Ironic Documentation

Release 17.1.1.dev30

OpenStack Foundation

Oct 22, 2024

CONTENTS

1 Introduction **1**

2 Installation Guide **3**
2.1 Bare Metal Service Installation Guide 3

3 Upgrade Guide **109**
3.1 Bare Metal Service Upgrade Guide 109

4 User Guide **121**
4.1 Bare Metal Service User Guide 121

5 Administrator Guide **129**
5.1 Drivers, Hardware Types and Hardware Interfaces 129
5.2 Administrators Guide 746

6 Configuration Guide **1735**
6.1 Configuration Reference 1735

7 Bare Metal API References **1905**
7.1 REST API Conceptual Guide 1905

8 Command References **1919**
8.1 Command References 1919

9 Contributor Guide **1925**
9.1 Developers Guide 1925

10 Release Notes **4563**

Python Module Index **4565**

Index **4569**

CHAPTER

ONE

INTRODUCTION

Ironic is an OpenStack project which provisions bare metal (as opposed to virtual) machines. It may be used independently or as part of an OpenStack Cloud, and integrates with the OpenStack Identity (keystone), Compute (nova), Network (neutron), Image (glance), and Object (swift) services.

The Bare Metal service manages hardware through both common (eg. PXE and IPMI) and vendor-specific remote management protocols. It provides the cloud operator with a unified interface to a heterogeneous fleet of servers while also providing the Compute service with an interface that allows physical servers to be managed as though they were virtual machines.

This documentation is continually updated and may not represent the state of the project at any specific prior release. To access documentation for a previous release of ironic, append the OpenStack release name to the URL; for example, the `ocata` release is available at https://docs.openstack.org/ironic/ocata/.

Found a bug in one of our projects? Please see *Bug Reporting and Triaging Guide*.

CHAPTER

TWO

INSTALLATION GUIDE

2.1 Bare Metal Service Installation Guide

The Bare Metal service is a collection of components that provides support to manage and provision physical machines.

This chapter assumes a working setup of OpenStack following the OpenStack Installation Guides. It contains the following sections:

2.1.1 Bare Metal service overview

The Bare Metal service, codenamed `ironic`, is a collection of components that provides support to manage and provision physical machines.

Bare Metal service components

The Bare Metal service includes the following components:

ironic-api A RESTful API that processes application requests by sending them to the ironic-conductor over remote procedure call (RPC). Can be run through WSGI or as a separate process.

ironic-conductor Adds/edits/deletes nodes; powers on/off nodes with IPMI or other vendor-specific protocol; provisions/deploys/cleans bare metal nodes.

ironic-conductor uses *drivers* to execute operations on hardware.

ironic-python-agent A python service which is run in a temporary ramdisk to provide ironic-conductor and ironic-inspector services with remote access, in-band hardware control, and hardware introspection.

Additionally, the Bare Metal service has certain external dependencies, which are very similar to other OpenStack services:

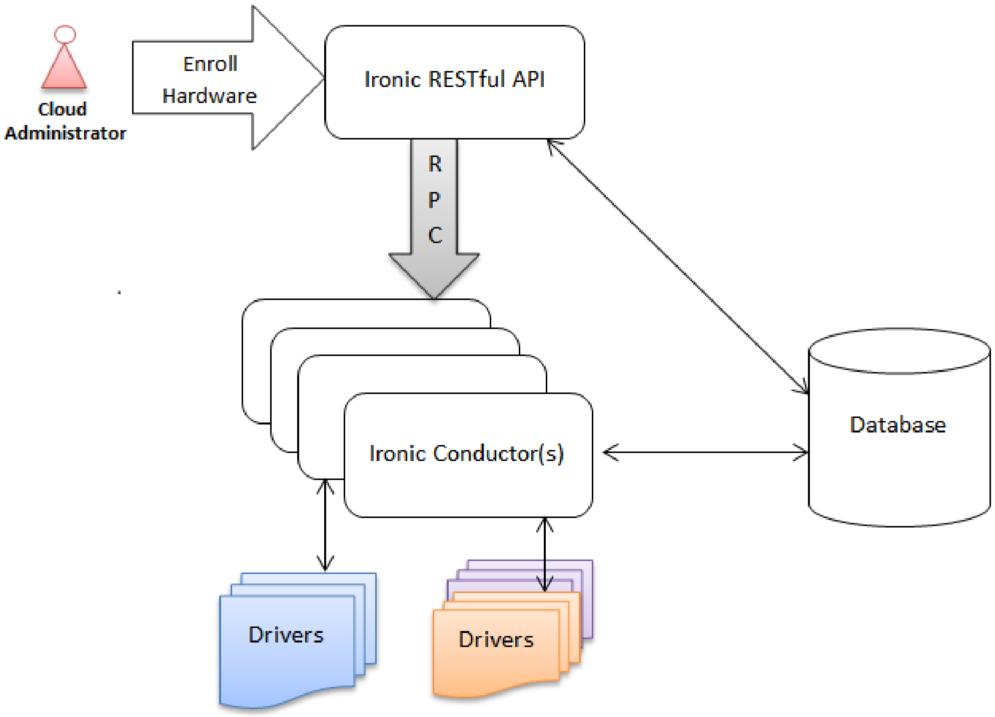
- A database to store hardware information and state. You can set the database back-end type and location. A simple approach is to use the same database back end as the Compute service. Another approach is to use a separate database back-end to further isolate bare metal resources (and associated metadata) from users.
- An oslo.messaging compatible queue, such as RabbitMQ. It may use the same implementation as that of the Compute service, but that is not a requirement. Used to implement RPC between ironic-api and ironic-conductor.

Deployment architecture

The Bare Metal RESTful API service is used to enroll hardware that the Bare Metal service will manage. A cloud administrator usually registers it, specifying their attributes such as MAC addresses and IPMI credentials. There can be multiple instances of the API service.

The *ironic-conductor* process does the bulk of the work. For security reasons, it is advisable to place it on an isolated host, since it is the only service that requires access to both the data plane and IPMI control plane.

There can be multiple instances of the conductor service to support various class of drivers and also to manage fail over. Instances of the conductor service should be on separate nodes. Each conductor can itself run many drivers to operate heterogeneous hardware. This is depicted in the following figure.



The API exposes a list of supported drivers and the names of conductor hosts servicing them.

Interaction with OpenStack components

The Bare Metal service may, depending upon configuration, interact with several other OpenStack services. This includes:

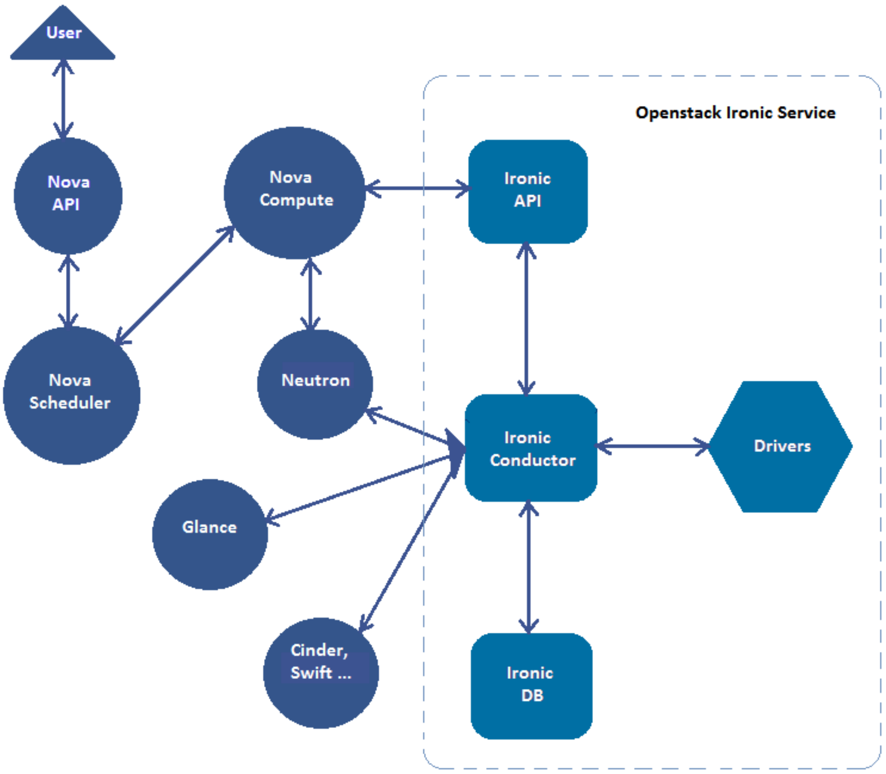
- the OpenStack Telemetry module (`ceilometer`) for consuming the IPMI metrics
- the OpenStack Identity service (`keystone`) for request authentication and to locate other OpenStack services
- the OpenStack Image service (`glance`) from which to retrieve images and image meta-data
- the OpenStack Networking service (`neutron`) for DHCP and network configuration

- the OpenStack Compute service (`nova`) works with the Bare Metal service and acts as a user-facing API for instance management, while the Bare Metal service provides the admin/operator API for hardware management. The OpenStack Compute service also provides scheduling facilities (matching flavors <-> images <-> hardware), tenant quotas, IP assignment, and other services which the Bare Metal service does not, in and of itself, provide.
- the OpenStack Object Storage (`swift`) provides temporary storage for the configdrive, user images, deployment logs and inspection data.

Logical architecture

The diagram below shows the logical architecture. It shows the basic components that form the Bare Metal service, the relation of the Bare Metal service with other OpenStack services and the logical flow of a boot instance request resulting in the provisioning of a physical server.

Figure 1.2. Logical Architecture



A users request to boot an instance is passed to the Compute service via the Compute API and the Compute Scheduler. The Compute service uses the *ironic virt driver* to hand over this request to the Bare Metal service, where the request passes from the Bare Metal API, to the Conductor, to a Driver to successfully provision a physical server for the user.

Just as the Compute service talks to various OpenStack services like Image, Network, Object Store etc to provision a virtual machine instance, here the Bare Metal service talks to the same OpenStack services for image, network and other resource needs to provision a bare metal instance.

See *Understanding Bare Metal Deployment* for a more detailed breakdown of a typical deployment process.

Associated projects

Optionally, one may wish to utilize the following associated projects for additional functionality:

python-ironicclient A command-line interface (CLI) and python bindings for interacting with the Bare Metal service.

ironic-ui Horizon dashboard, providing graphical interface (GUI) for the Bare Metal API.

ironic-inspector An associated service which performs in-band hardware introspection by PXE booting unregistered hardware into the ironic-python-agent ramdisk.

diskimage-builder A related project to help facilitate the creation of ramdisks and machine images, such as those running the ironic-python-agent.

bifrost A set of Ansible playbooks that automates the task of deploying a base image onto a set of known hardware using ironic in a standalone mode.

2.1.2 Reference Deploy Architectures

This section covers the way we recommend the Bare Metal service to be deployed and managed. It is assumed that a reader has already gone through *Bare Metal Service User Guide*. It may be also useful to try *Deploying Ironic with DevStack* first to get better familiar with the concepts used in this guide.

Common Considerations

This section covers considerations that are equally important to all described architectures.

- *Components*
- *Hardware and drivers*
    - *Power and management interfaces*
    - *Boot interface*
    - *Deploy interface*
    - *Hardware specifications*
- *Image types*
- *Local vs network boot*
- *Networking*
- *HA and Scalability*
    - *ironic-api*
    - *ironic-conductor*
        - *High availability*
        - *Performance*
        - *Disk space*
    - *Other services*

Components

As explained in *Bare Metal service overview*, the Bare Metal service has three components.

- The Bare Metal API service (`ironic-api`) should be deployed in a similar way as the control plane API services. The exact location will depend on the architecture used.
- The Bare Metal conductor service (`ironic-conductor`) is where most of the provisioning logic lives. The following considerations are the most important when deciding on the way to deploy it:
  - The conductor manages a certain proportion of nodes, distributed to it via a hash ring. This includes constantly polling these nodes for their current power state and hardware sensor data (if enabled and supported by hardware, see *Collecting sensor data* for an example).
  - The conductor needs access to the management controller of each node it manages.
  - The conductor co-exists with TFTP (for PXE) and/or HTTP (for iPXE) services that provide the kernel and ramdisk to boot the nodes. The conductor manages them by writing files to their root directories.
  - If serial console is used, the conductor launches console processes locally. If the `nova-serialproxy` service (part of the Compute service) is used, it has to be able to reach the conductors. Otherwise, they have to be directly accessible by the users.
  - There must be mutual connectivity between the conductor and the nodes being deployed or cleaned. See *Networking* for details.
- The provisioning ramdisk which runs the `ironic-python-agent` service on start up.

> **Warning:** The `ironic-python-agent` service is not intended to be used or executed anywhere other than a provisioning/cleaning/rescue ramdisk.

Hardware and drivers

The Bare Metal service strives to provide the best support possible for a variety of hardware. However, not all hardware is supported equally well. It depends on both the capabilities of hardware itself and the available drivers. This section covers various considerations related to the hardware interfaces. See *Enabling drivers and hardware types* for a detailed introduction into hardware types and interfaces before proceeding.

Power and management interfaces

The minimum set of capabilities that the hardware has to provide and the driver has to support is as follows:

1. getting and setting the power state of the machine
2. getting and setting the current boot device
3. booting an image provided by the Bare Metal service (in the simplest case, support booting using PXE and/or iPXE)

Note: Strictly speaking, it is possible to make the Bare Metal service provision nodes without some of these capabilities via some manual steps. It is not the recommended way of deployment, and thus it is not covered in this guide.

Once you make sure that the hardware supports these capabilities, you need to find a suitable driver. Most of enterprise-grade hardware has support for IPMI and thus can utilize *IPMI driver*. Some newer hardware also supports *Redfish driver*. Several vendors provide more specific drivers that usually provide additional capabilities. Check *Drivers, Hardware Types and Hardware Interfaces* to find the most suitable one.

Boot interface

The boot interface of a node manages booting of both the deploy ramdisk and the user instances on the bare metal node. The deploy interface orchestrates the deployment and defines how the image gets transferred to the target disk.

The main alternatives are to use PXE/iPXE or virtual media - see *Boot interfaces* for a detailed explanation. If a virtual media implementation is available for the hardware, it is recommended using it for better scalability and security. Otherwise, it is recommended to use iPXE, when it is supported by target hardware.

Deploy interface

There are two deploy interfaces in-tree, `iscsi` and `direct`. See *Deploy Interfaces* for explanation of the difference. With the `iscsi` deploy method, most of the deployment operations happen on the conductor. If the Object Storage service (swift) or RadosGW is present in the environment, it is recommended to use the `direct` deploy method for better scalability and reliability.

Hardware specifications

The Bare Metal services does not impose too many restrictions on the characteristics of hardware itself. However, keep in mind that

- By default, the Bare Metal service will pick the smallest hard drive that is larger than 4 GiB for deployment. Another hard drive can be used, but it requires setting *root device hints*.

  Note: This device does not have to match the boot device set in BIOS (or similar firmware).

- The machines should have enough RAM to fit the deployment/cleaning ramdisk to run. The minimum varies greatly depending on the way the ramdisk was built. For example, *tinyipa*, the TinyCoreLinux-based ramdisk used in the CI, only needs 400 MiB of RAM, while ramdisks built by *diskimage-builder* may require 3 GiB or more.

Image types

The Bare Metal service can deploy two types of images:

- *Whole-disk* images that contain a complete partitioning table with all necessary partitions and a bootloader. Such images are the most universal, but may be harder to build.
- *Partition images* that contain only the root partition. The Bare Metal service will create the necessary partitions and install a boot loader, if needed.

Warning: Partition images are only supported with GNU/Linux operating systems.

Warning: If you plan on using local boot, your partition images must contain GRUB2 bootloader tools to enable ironic to set up the bootloader during deploy.

Local vs network boot

The Bare Metal service supports booting user instances either using a local bootloader or using the drivers boot interface (e.g. via PXE or iPXE protocol in case of the `pxe` interface).

Network boot cannot be used with certain architectures (for example, when no tenant networks have access to the control plane).

Additional considerations are related to the `pxe` boot interface, and other boot interfaces based on it:

- Local boot makes nodes boot process independent of the Bare Metal conductor managing it. Thus, nodes are able to reboot correctly, even if the Bare Metal TFTP or HTTP service is down.
- Network boot (and iPXE) must be used when booting nodes from remote volumes, if the driver does not support attaching volumes out-of-band.

The default boot option for the cloud can be changed via the Bare Metal service configuration file, for example:

```
[deploy]
default_boot_option = local
```

This default can be overridden by setting the `boot_option` capability on a node. See *Local boot with partition images* for details.

Note: Currently, local boot is used by default. Its safer to set the `default_boot_option` explicitly.

Networking

There are several recommended network topologies to be used with the Bare Metal service. They are explained in depth in specific architecture documentation. However, several considerations are common for all of them:

- There has to be a *provisioning* network, which is used by nodes during the deployment process. If allowed by the architecture, this network should not be accessible by end users, and should not have access to the internet.
- There has to be a *cleaning* network, which is used by nodes during the cleaning process.
- There should be a *rescuing* network, which is used by nodes during the rescue process. It can be skipped if the rescue process is not supported.

Note: In the majority of cases, the same network should be used for cleaning, provisioning and rescue for simplicity.

Unless noted otherwise, everything in these sections apply to all three networks.

- The baremetal nodes must have access to the Bare Metal API while connected to the provisioning/cleaning/rescuing network.

  Note: Only two endpoints need to be exposed there:

  ```
  GET /v1/lookup
  POST /v1/heartbeat/[a-z0-9\-]+
  ```

  You may want to limit access from this network to only these endpoints, and make these endpoint not accessible from other networks.

- If the `pxe` boot interface (or any boot interface based on it) is used, then the baremetal nodes should have untagged (access mode) connectivity to the provisioning/cleaning/rescuing networks. It allows PXE firmware, which does not support VLANs, to communicate with the services required for provisioning.

  Note: It depends on the *network interface* whether the Bare Metal service will handle it automatically. Check the networking documentation for the specific architecture.

  Sometimes it may be necessary to disable the spanning tree protocol delay on the switch - see *DHCP during PXE or iPXE is inconsistent or unreliable*.

- The Baremetal nodes need to have access to any services required for provisioning/cleaning/rescue, while connected to the provisioning/cleaning/rescuing network. This may include:
  - a TFTP server for PXE boot and also an HTTP server when iPXE is enabled
  - either an HTTP server or the Object Storage service in case of the `direct` deploy interface and some virtual media boot interfaces

- The Baremetal Conductors need to have access to the booted baremetal nodes during provisioning/cleaning/rescue. A conductor communicates with an internal API, provided by **ironic-python-agent**, to conduct actions on nodes.

HA and Scalability

ironic-api

The Bare Metal API service is stateless, and thus can be easily scaled horizontally. It is recommended to deploy it as a WSGI application behind e.g. Apache or another WSGI container.

Note: This service accesses the ironic database for reading entities (e.g. in response to `GET /v1/nodes` request) and in rare cases for writing.

ironic-conductor

High availability

The Bare Metal conductor service utilizes the active/active HA model. Every conductor manages a certain subset of nodes. The nodes are organized in a hash ring that tries to keep the load spread more or less uniformly across the conductors. When a conductor is considered offline, its nodes are taken over by other conductors. As a result of this, you need at least 2 conductor hosts for an HA deployment.

Performance

Conductors can be resource intensive, so it is recommended (but not required) to keep all conductors separate from other services in the cloud. The minimum required number of conductors in a deployment depends on several factors:

- the performance of the hardware where the conductors will be running,
- the speed and reliability of the management controller of the bare metal nodes (for example, handling slower controllers may require having less nodes per conductor),
- the frequency, at which the management controllers are polled by the Bare Metal service (see the `sync_power_state_interval` option),
- the bare metal driver used for nodes (see *Hardware and drivers* above),
- the network performance,
- the maximum number of bare metal nodes that are provisioned simultaneously (see the `max_concurrent_builds` option for the Compute service).

We recommend a target of **100** bare metal nodes per conductor for maximum reliability and performance. There is some tolerance for a larger number per conductor. However, it was reported[1][2] that reliability degrades when handling approximately 300 bare metal nodes per conductor.

---

[1] http://lists.openstack.org/pipermail/openstack-dev/2017-June/118033.html

[2] http://lists.openstack.org/pipermail/openstack-dev/2017-June/118327.html

Disk space

Each conductor needs enough free disk space to cache images it uses. Depending on the combination of the deploy interface and the boot option, the space requirements are different:

- The deployment kernel and ramdisk are always cached during the deployment.
- When `[agent]image_download_source` is set to `http` and Glance is used, the conductor will download instances images locally to serve them from its HTTP server. Use `swift` to publish images using temporary URLs and convert them on the nodes side.

  When `[agent]image_download_source` is set to `local`, it will happen even for HTTP(s) URLs. For standalone case use `http` to avoid unnecessary caching of images.

  In both cases a cached image is converted to raw if `force_raw_images` is `True` (the default).

  Note: `image_download_source` can also be provided in the nodes `driver_info` or `instance_info`. See *Deploy with custom HTTP servers*.

- The `iscsi` deploy method always requires caching of the whole instance image locally during the deployment. The image has to be converted to the raw format, which may increase the required amount of disk space, as well as the CPU load.
- When network boot is used, the instance image kernel and ramdisk are cached locally while the instance is active.

Note: All images may be stored for some time after they are no longer needed. This is done to speed up simultaneous deployments of many similar images. The caching can be configured via the `image_cache_size` and `image_cache_ttl` configuration options in the `pxe` group.

Other services

When integrating with other OpenStack services, more considerations may need to be applied. This is covered in other parts of this guide.

Scenarios

Small cloud with trusted tenants

Story

As an operator I would like to build a small cloud with both virtual and bare metal instances or add bare metal provisioning to my existing small or medium scale single-site OpenStack cloud. The expected number of bare metal machines is less than 100, and the rate of provisioning and unprovisioning is expected to be low. All users of my cloud are trusted by me to not conduct malicious actions towards each other or the cloud infrastructure itself.

As a user I would like to occasionally provision bare metal instances through the Compute API by selecting an appropriate Compute flavor. I would like to be able to boot them from images provided by the Image service or from volumes provided by the Volume service.

Components

This architecture assumes an OpenStack installation with the following components participating in the bare metal provisioning:

- The Compute service manages bare metal instances.
- The Networking service provides DHCP for bare metal instances.
- The Image service provides images for bare metal instances.

The following services can be optionally used by the Bare Metal service:

- The Volume service provides volumes to boot bare metal instances from.
- The Bare Metal Introspection service simplifies enrolling new bare metal machines by conducting in-band introspection.

Node roles

An OpenStack installation in this guide has at least these three types of nodes:

- A *controller* node hosts the control plane services.
- A *compute* node runs the virtual machines and hosts a subset of Compute and Networking components.
- A *block storage* node provides persistent storage space for both virtual and bare metal nodes.

The *compute* and *block storage* nodes are configured as described in the installation guides of the Compute service and the Volume service respectively. The *controller* nodes host the Bare Metal service components.

Networking

The networking architecture will highly depend on the exact operating requirements. This guide expects the following existing networks: *control plane*, *storage* and *public*. Additionally, two more networks will be needed specifically for bare metal provisioning: *bare metal* and *management*.

Control plane network

The *control plane network* is the network where OpenStack control plane services provide their public API.

The Bare Metal API will be served to the operators and to the Compute service through this network.

Public network

The *public network* is used in a typical OpenStack deployment to create floating IPs for outside access to instances. Its role is the same for a bare metal deployment.

Note: Since, as explained below, bare metal nodes will be put on a flat provider network, it is also possible to organize direct access to them, without using floating IPs and bypassing the Networking service completely.

Bare metal network

The *Bare metal network* is a dedicated network for bare metal nodes managed by the Bare Metal service.

This architecture uses *flat bare metal networking*, in which both tenant traffic and technical traffic related to the Bare Metal service operation flow through this one network. Specifically, this network will serve as the *provisioning*, *cleaning* and *rescuing* network. It will also be used for introspection via the Bare Metal Introspection service. See *common networking considerations* for an in-depth explanation of the networks used by the Bare Metal service.

DHCP and boot parameters will be provided on this network by the Networking services DHCP agents.

For booting from volumes this network has to have a route to the *storage network*.

Management network

Management network is an independent network on which BMCs of the bare metal nodes are located.

The `ironic-conductor` process needs access to this network. The tenants of the bare metal nodes must not have access to it.

Note: The *direct deploy interface* and certain *Drivers, Hardware Types and Hardware Interfaces* require the *management network* to have access to the Object storage service backend.

Controllers

A *controller* hosts the OpenStack control plane services as described in the control plane design guide. While this architecture allows using *controllers* in a non-HA configuration, it is recommended to have at least three of them for HA. See *HA and Scalability* for more details.

Bare Metal services

The following components of the Bare Metal service are installed on a *controller* (see *components of the Bare Metal service*):

- The Bare Metal API service either as a WSGI application or the `ironic-api` process. Typically, a load balancer, such as HAProxy, spreads the load between the API instances on the *controllers*.

  The API has to be served on the *control plane network*. Additionally, it has to be exposed to the *bare metal network* for the ramdisk callback API.

- The `ironic-conductor` process. These processes work in active/active HA mode as explained in *HA and Scalability*, thus they can be installed on all *controllers*. Each will handle a subset of bare metal nodes.

  The `ironic-conductor` processes have to have access to the following networks:

  - *control plane* for interacting with other services
  - *management* for contacting nodes BMCs
  - *bare metal* for contacting deployment, cleaning or rescue ramdisks

- TFTP and HTTP service for booting the nodes. Each `ironic-conductor` process has to have a matching TFTP and HTTP service. They should be exposed only to the *bare metal network* and must not be behind a load balancer.

- The `nova-compute` process (from the Compute service). These processes work in active/active HA mode when dealing with bare metal nodes, thus they can be installed on all *controllers*. Each will handle a subset of bare metal nodes.

  Note: There is no 1-1 mapping between `ironic-conductor` and `nova-compute` processes, as they communicate only through the Bare Metal API service.

- The networking-baremetal ML2 plugin should be loaded into the Networking service to assist with binding bare metal ports.

  The ironic-neutron-agent service should be started as well.

- If the Bare Metal introspection is used, its `ironic-inspector` process has to be installed on all *controllers*. Each such process works as both Bare Metal Introspection API and conductor service. A load balancer should be used to spread the API load between *controllers*.

  The API has to be served on the *control plane network*. Additionally, it has to be exposed to the *bare metal network* for the ramdisk callback API.

Shared services

A *controller* also hosts two services required for the normal operation of OpenStack:

- Database service (MySQL/MariaDB is typically used, but other enterprise-grade database solutions can be used as well).

  All Bare Metal service components need access to the database service.

- Message queue service (RabbitMQ is typically used, but other enterprise-grade message queue brokers can be used as well).

  Both Bare Metal API (WSGI application or `ironic-api` process) and the `ironic-conductor` processes need access to the message queue service. The Bare Metal Introspection service does not need it.

Note: These services are required for all OpenStack services. If youre adding the Bare Metal service to your cloud, you may reuse the existing database and messaging queue services.

Bare metal nodes

Each bare metal node must be capable of booting from network, virtual media or other boot technology supported by the Bare Metal service as explained in *Boot interface*. Each node must have one NIC on the *bare metal network*, and this NIC (and **only** it) must be configured to be able to boot from network. This is usually done in the *BIOS setup* or a similar firmware configuration utility. There is no need to alter the boot order, as it is managed by the Bare Metal service. Other NICs, if present, will not be managed by OpenStack.

The NIC on the *bare metal network* should have untagged connectivity to it, since PXE firmware usually does not support VLANs - see *Networking* for details.

Storage

If your hardware **and** its bare metal *driver* support booting from remote volumes, please check the driver documentation for information on how to enable it. It may include routing *management* and/or *bare metal* networks to the *storage network*.

In case of the standard *PXE boot*, booting from remote volumes is done via iPXE. In that case, the Volume storage backend must support iSCSI protocol, and the *bare metal network* has to have a route to the *storage network*. See *Boot From Volume* for more details.

2.1.3 Install and configure the Bare Metal service

This section describes how to install and configure the Bare Metal service, code-named ironic.

Note that installation and configuration vary by distribution.

Install and configure for Red Hat Enterprise Linux and CentOS

This section describes how to install and configure the Bare Metal service for Red Hat Enterprise Linux 8 and CentOS 8.

Install and configure prerequisites

The Bare Metal service is a collection of components that provides support to manage and provision physical machines. You can configure these components to run on separate nodes or the same node. In this guide, the components run on one node, typically the Compute Services compute node.

It assumes that the Identity, Image, Compute, and Networking services have already been set up.

Set up the database for Bare Metal

The Bare Metal service stores information in a database. This guide uses the MySQL database that is used by other OpenStack services.

1. In MySQL, create an `ironic` database that is accessible by the `ironic` user. Replace `IRONIC_DBPASSWORD` with a suitable password:

```
# mysql -u root -p
mysql> CREATE DATABASE ironic CHARACTER SET utf8;
mysql> GRANT ALL PRIVILEGES ON ironic.* TO 'ironic'@'localhost' \
       IDENTIFIED BY 'IRONIC_DBPASSWORD';
mysql> GRANT ALL PRIVILEGES ON ironic.* TO 'ironic'@'%' \
       IDENTIFIED BY 'IRONIC_DBPASSWORD';
```

Install and configure components

1. Install from packages (using dnf)

```
# dnf install openstack-ironic-api openstack-ironic-conductor python3-
↪ironicclient
```

2. Enable services

```
# systemctl enable openstack-ironic-api openstack-ironic-conductor
# systemctl start openstack-ironic-api openstack-ironic-conductor
```

The Bare Metal service is configured via its configuration file. This file is typically located at `/etc/ironic/ironic.conf`.

Although some configuration options are mentioned here, it is recommended that you review all the *Sample Configuration File* so that the Bare Metal service is configured for your needs.

It is possible to set up an ironic-api and an ironic-conductor services on the same host or different hosts. Users also can add new ironic-conductor hosts to deal with an increasing number of bare metal nodes. But the additional ironic-conductor services should be at the same version as that of existing ironic-conductor services.

Configuring ironic-api service

1. The Bare Metal service stores information in a database. This guide uses the MySQL database that is used by other OpenStack services.

   Configure the location of the database via the `connection` option. In the following, replace `IRONIC_DBPASSWORD` with the password of your `ironic` user, and replace `DB_IP` with the IP address where the DB server is located:

   ```
   [database]

   # The SQLAlchemy connection string used to connect to the
   # database (string value)
   connection=mysql+pymysql://ironic:IRONIC_DBPASSWORD@DB_IP/ironic?
   ↪charset=utf8
   ```

2. Configure the ironic-api service to use the RabbitMQ message broker by setting the following option. Replace `RPC_*` with appropriate address details and credentials of RabbitMQ server:

   ```
   [DEFAULT]

   # A URL representing the messaging driver to use and its full
   # configuration. (string value)
   transport_url = rabbit://RPC_USER:RPC_PASSWORD@RPC_HOST:RPC_PORT/
   ```

   Alternatively, you can use JSON RPC for interactions between ironic-conductor and ironic-api. Enable it in the configuration and provide the keystone credentials to use for authentication:

   ```
   [DEFAULT]

   rpc_transport = json-rpc

   [json_rpc]

   # Authentication type to load (string value)
   auth_type = password

   # Authentication URL (string value)
   auth_url=https://IDENTITY_IP:5000/

   # Username (string value)
   username=ironic

   # User's password (string value)
   password=IRONIC_PASSWORD

   # Project name to scope to (string value)
   project_name=service

   # Domain ID containing project (string value)
   project_domain_id=default

   # User's domain id (string value)
   user_domain_id=default
   ```

   If you use port other than the default 8089 for JSON RPC, you have to configure it, for example:

```
[json_rpc]
port = 9999
```

3. Configure the ironic-api service to use these credentials with the Identity service. Replace `PUBLIC_IDENTITY_IP` with the public IP of the Identity server, `PRIVATE_IDENTITY_IP` with the private IP of the Identity server and replace `IRONIC_PASSWORD` with the password you chose for the `ironic` user in the Identity service:

```
[DEFAULT]

# Authentication strategy used by ironic-api: one of
# "keystone" or "noauth". "noauth" should not be used in a
# production environment because all authentication will be
# disabled. (string value)
auth_strategy=keystone

[keystone_authtoken]

# Authentication type to load (string value)
auth_type=password

# Complete public Identity API endpoint (string value)
www_authenticate_uri=http://PUBLIC_IDENTITY_IP:5000

# Complete admin Identity API endpoint. (string value)
auth_url=http://PRIVATE_IDENTITY_IP:5000

# Service username. (string value)
username=ironic

# Service account password. (string value)
password=IRONIC_PASSWORD

# Service tenant name. (string value)
project_name=service

# Domain name containing project (string value)
project_domain_name=Default

# User's domain name (string value)
user_domain_name=Default
```

4. Create the Bare Metal service database tables:

```
$ ironic-dbsync --config-file /etc/ironic/ironic.conf create_schema
```

5. Restart the ironic-api service:

   Fedora/RHEL8/CentOS8/SUSE:

```
sudo systemctl restart openstack-ironic-api
```

   Ubuntu:

```
sudo service ironic-api restart
```

Configuring ironic-api behind mod_wsgi

Bare Metal service comes with an example file for configuring the `ironic-api` service to run behind Apache with mod_wsgi.

1. Install the apache service:

   Fedora/RHEL8/CentOS8:

   ```
   sudo dnf install httpd
   ```

   Debian/Ubuntu:

   ```
   apt-get install apache2
   ```

   SUSE:

   ```
   zypper install apache2
   ```

2. Download the `etc/apache2/ironic` file from the Ironic project tree and copy it to the apache sites:

   Fedora/RHEL8/CentOS8:

   ```
   sudo cp etc/apache2/ironic /etc/httpd/conf.d/ironic.conf
   ```

   Debian/Ubuntu:

   ```
   sudo cp etc/apache2/ironic /etc/apache2/sites-available/ironic.conf
   ```

   SUSE:

   ```
   sudo cp etc/apache2/ironic /etc/apache2/vhosts.d/ironic.conf
   ```

3. Edit the recently copied `<apache-configuration-dir>/ironic.conf`:

   1. Modify the `WSGIDaemonProcess`, `APACHE_RUN_USER` and `APACHE_RUN_GROUP` directives to set the user and group values to an appropriate user on your server.
   2. Modify the `WSGIScriptAlias` directive to point to the automatically generated `ironic-api-wsgi` script that is located in *IRONIC_BIN* directory.
   3. Modify the `Directory` directive to set the path to the Ironic API code.
   4. Modify the `ErrorLog` and `CustomLog` to redirect the logs to the right directory (on Red Hat systems this is usually under /var/log/httpd).

4. Enable the apache `ironic` in site and reload:

   Fedora/RHEL8/CentOS8:

   ```
   sudo systemctl reload httpd
   ```

   Debian/Ubuntu:

   ```
   sudo a2ensite ironic
   sudo service apache2 reload
   ```

SUSE:

```
sudo systemctl reload apache2
```

Note: The file `ironic-api-wsgi` is automatically generated by pbr and is available in *IRONIC_BIN* directory. It should not be modified.

Configure another WSGI container

A slightly different approach has to be used for WSGI containers that cannot use `ironic-api-wsgi`. For example, for *gunicorn*:

```
gunicorn -b 0.0.0.0:6385 'ironic.api.wsgi:initialize_wsgi_app(argv=[])'
```

If you want to pass a configuration file, use:

```
gunicorn -b 0.0.0.0:6385 \
    'ironic.api.wsgi:initialize_wsgi_app(argv=["ironic-api", "--config-
↪file=/path/to/_ironic.conf"])'
```

Configuring ironic-conductor service

1. Replace `HOST_IP` with IP of the conductor host.

   ```
   [DEFAULT]

   # IP address of this host. If unset, will determine the IP
   # programmatically. If unable to do so, will use "127.0.0.1".
   # (string value)
   my_ip=HOST_IP
   ```

   Note: If a conductor host has multiple IPs, `my_ip` should be set to the IP which is on the same network as the bare metal nodes.

   ---

2. Configure the location of the database. Ironic-conductor should use the same configuration as ironic-api. Replace `IRONIC_DBPASSWORD` with the password of your `ironic` user, and replace DB_IP with the IP address where the DB server is located:

   ```
   [database]

   # The SQLAlchemy connection string to use to connect to the
   # database. (string value)
   connection=mysql+pymysql://ironic:IRONIC_DBPASSWORD@DB_IP/ironic?
   ↪charset=utf8
   ```

3. Configure the ironic-conductor service to use the RabbitMQ message broker by setting the following option. Ironic-conductor should use the same configuration as ironic-api. Replace `RPC_*` with appropriate address details and credentials of RabbitMQ server:

```
[DEFAULT]

# A URL representing the messaging driver to use and its full
# configuration. (string value)
transport_url = rabbit://RPC_USER:RPC_PASSWORD@RPC_HOST:RPC_PORT/
```

Alternatively, you can use JSON RPC for interactions between ironic-conductor and ironic-api. Enable it in the configuration and provide the keystone credentials to use for authenticating incoming requests (can be the same as for the API):

```
[DEFAULT]

rpc_transport = json-rpc

[keystone_authtoken]

# Authentication type to load (string value)
auth_type=password

# Complete public Identity API endpoint (string value)
www_authenticate_uri=http://PUBLIC_IDENTITY_IP:5000

# Complete admin Identity API endpoint. (string value)
auth_url=http://PRIVATE_IDENTITY_IP:5000

# Service username. (string value)
username=ironic

# Service account password. (string value)
password=IRONIC_PASSWORD

# Service tenant name. (string value)
project_name=service

# Domain name containing project (string value)
project_domain_name=Default

# User's domain name (string value)
user_domain_name=Default
```

You can optionally change the host and the port the JSON RPC service will bind to, for example:

```
[json_rpc]
host_ip = 192.168.0.10
port = 9999
```

Warning: Hostnames of ironic-conductor machines must be resolvable by ironic-api services when JSON RPC is used.

4. Configure credentials for accessing other OpenStack services.

   In order to communicate with other OpenStack services, the Bare Metal service needs to use service users to authenticate to the OpenStack Identity service when making requests to other services. These users credentials have to be configured in each configuration file section related to the corresponding service:

- `[neutron]` - to access the OpenStack Networking service
- `[glance]` - to access the OpenStack Image service
- `[swift]` - to access the OpenStack Object Storage service
- `[cinder]` - to access the OpenStack Block Storage service
- `[inspector]` - to access the OpenStack Bare Metal Introspection service
- `[service_catalog]` - a special section holding credentials the Bare Metal service will use to discover its own API URL endpoint as registered in the OpenStack Identity service catalog.

For simplicity, you can use the same service user for all services. For backward compatibility, this should be the same user configured in the `[keystone_authtoken]` section for the ironic-api service (see Configuring ironic-api service). However, this is not necessary, and you can create and configure separate service users for each service.

Under the hood, Bare Metal service uses `keystoneauth` library together with `Authentication plugin`, `Session` and `Adapter` concepts provided by it to instantiate service clients. Please refer to Keystoneauth documentation for supported plugins, their available options as well as Session- and Adapter-related options for authentication, connection and endpoint discovery respectively.

In the example below, authentication information for user to access the OpenStack Networking service is configured to use:

- Networking service is deployed in the Identity service region named `RegionTwo`, with only its `public` endpoint interface registered in the service catalog.
- HTTPS connection with specific CA SSL certificate when making requests
- the same service user as configured for ironic-api service
- dynamic `password` authentication plugin that will discover appropriate version of Identity service API based on other provided options
  - replace `IDENTITY_IP` with the IP of the Identity server, and replace `IRONIC_PASSWORD` with the password you chose for the `ironic` user in the Identity service

```
[neutron]

# Authentication type to load (string value)
auth_type = password

# Authentication URL (string value)
auth_url=https://IDENTITY_IP:5000/

# Username (string value)
username=ironic

# User's password (string value)
password=IRONIC_PASSWORD

# Project name to scope to (string value)
```

(continues on next page)

(continued from previous page)

```
project_name=service

# Domain ID containing project (string value)
project_domain_id=default

# User's domain id (string value)
user_domain_id=default

# PEM encoded Certificate Authority to use when verifying
# HTTPs connections. (string value)
cafile=/opt/stack/data/ca-bundle.pem

# The default region_name for endpoint URL discovery. (string
# value)
region_name = RegionTwo

# List of interfaces, in order of preference, for endpoint
# URL. (list value)
valid_interfaces=public
```

By default, in order to communicate with another service, the Bare Metal service will attempt to discover an appropriate endpoint for that service via the Identity services service catalog. The relevant configuration options from that service group in the Bare Metal service configuration file are used for this purpose. If you want to use a different endpoint for a particular service, specify this via the `endpoint_override` configuration option of that service group, in the Bare Metal services configuration file. Taking the previous Networking service example, this would be

```
[neutron]
...
endpoint_override = <NEUTRON_API_ADDRESS>
```

(Replace *<NEUTRON_API_ADDRESS>* with actual address of a specific Networking service endpoint.)

5. Configure enabled drivers and hardware types as described in *Enabling drivers and hardware types*.

   A. If you enabled any driver that uses *Direct deploy*, Swift backend for the Image service must be installed and configured, see *Configure the Image service for temporary URLs*. Ceph Object Gateway (RADOS Gateway) is also supported as the Image services backend, see *Ceph Object Gateway support*.

6. Configure the network for ironic-conductor service to perform node cleaning, see *Node cleaning* from the admin guide.

7. Restart the ironic-conductor service:

   Fedora/RHEL7/CentOS7/SUSE:

   ```
   sudo systemctl restart openstack-ironic-conductor
   ```

   Ubuntu:

   ```
   sudo service ironic-conductor restart
   ```

Install and configure for Ubuntu

This section describes how to install and configure the Bare Metal service for Ubuntu 14.04 (LTS).

Install and configure prerequisites

The Bare Metal service is a collection of components that provides support to manage and provision physical machines. You can configure these components to run on separate nodes or the same node. In this guide, the components run on one node, typically the Compute Services compute node.

It assumes that the Identity, Image, Compute, and Networking services have already been set up.

Set up the database for Bare Metal

The Bare Metal service stores information in a database. This guide uses the MySQL database that is used by other OpenStack services.

1. In MySQL, create an `ironic` database that is accessible by the `ironic` user. Replace `IRONIC_DBPASSWORD` with a suitable password:

   ```
   # mysql -u root -p
   mysql> CREATE DATABASE ironic CHARACTER SET utf8;
   mysql> GRANT ALL PRIVILEGES ON ironic.* TO 'ironic'@'localhost' \
          IDENTIFIED BY 'IRONIC_DBPASSWORD';
   mysql> GRANT ALL PRIVILEGES ON ironic.* TO 'ironic'@'%' \
          IDENTIFIED BY 'IRONIC_DBPASSWORD';
   ```

Install and configure components

1. Install from packages (using apt-get)

   ```
   # apt-get install ironic-api ironic-conductor python3-ironicclient
   ```

2. Enable services

   Services are enabled by default on Ubuntu.

The Bare Metal service is configured via its configuration file. This file is typically located at `/etc/ironic/ironic.conf`.

Although some configuration options are mentioned here, it is recommended that you review all the *Sample Configuration File* so that the Bare Metal service is configured for your needs.

It is possible to set up an ironic-api and an ironic-conductor services on the same host or different hosts. Users also can add new ironic-conductor hosts to deal with an increasing number of bare metal nodes. But the additional ironic-conductor services should be at the same version as that of existing ironic-conductor services.

Configuring ironic-api service

1. The Bare Metal service stores information in a database. This guide uses the MySQL database that is used by other OpenStack services.

   Configure the location of the database via the `connection` option. In the following, replace `IRONIC_DBPASSWORD` with the password of your `ironic` user, and replace `DB_IP` with the IP address where the DB server is located:

   ```
   [database]

   # The SQLAlchemy connection string used to connect to the
   # database (string value)
   connection=mysql+pymysql://ironic:IRONIC_DBPASSWORD@DB_IP/ironic?
   ↪charset=utf8
   ```

2. Configure the ironic-api service to use the RabbitMQ message broker by setting the following option. Replace `RPC_*` with appropriate address details and credentials of RabbitMQ server:

   ```
   [DEFAULT]

   # A URL representing the messaging driver to use and its full
   # configuration. (string value)
   transport_url = rabbit://RPC_USER:RPC_PASSWORD@RPC_HOST:RPC_PORT/
   ```

   Alternatively, you can use JSON RPC for interactions between ironic-conductor and ironic-api. Enable it in the configuration and provide the keystone credentials to use for authentication:

   ```
   [DEFAULT]

   rpc_transport = json-rpc

   [json_rpc]

   # Authentication type to load (string value)
   auth_type = password

   # Authentication URL (string value)
   auth_url=https://IDENTITY_IP:5000/

   # Username (string value)
   username=ironic

   # User's password (string value)
   password=IRONIC_PASSWORD

   # Project name to scope to (string value)
   project_name=service

   # Domain ID containing project (string value)
   project_domain_id=default

   # User's domain id (string value)
   user_domain_id=default
   ```

   If you use port other than the default 8089 for JSON RPC, you have to configure it, for example:

```
[json_rpc]
port = 9999
```

3. Configure the ironic-api service to use these credentials with the Identity service. Replace `PUBLIC_IDENTITY_IP` with the public IP of the Identity server, `PRIVATE_IDENTITY_IP` with the private IP of the Identity server and replace `IRONIC_PASSWORD` with the password you chose for the `ironic` user in the Identity service:

```
[DEFAULT]

# Authentication strategy used by ironic-api: one of
# "keystone" or "noauth". "noauth" should not be used in a
# production environment because all authentication will be
# disabled. (string value)
auth_strategy=keystone

[keystone_authtoken]

# Authentication type to load (string value)
auth_type=password

# Complete public Identity API endpoint (string value)
www_authenticate_uri=http://PUBLIC_IDENTITY_IP:5000

# Complete admin Identity API endpoint. (string value)
auth_url=http://PRIVATE_IDENTITY_IP:5000

# Service username. (string value)
username=ironic

# Service account password. (string value)
password=IRONIC_PASSWORD

# Service tenant name. (string value)
project_name=service

# Domain name containing project (string value)
project_domain_name=Default

# User's domain name (string value)
user_domain_name=Default
```

4. Create the Bare Metal service database tables:

```
$ ironic-dbsync --config-file /etc/ironic/ironic.conf create_schema
```

5. Restart the ironic-api service:

   Fedora/RHEL8/CentOS8/SUSE:

```
sudo systemctl restart openstack-ironic-api
```

   Ubuntu:

```
sudo service ironic-api restart
```

Configuring ironic-api behind mod_wsgi

Bare Metal service comes with an example file for configuring the `ironic-api` service to run behind Apache with mod_wsgi.

1. Install the apache service:

   Fedora/RHEL8/CentOS8:

   ```
   sudo dnf install httpd
   ```

   Debian/Ubuntu:

   ```
   apt-get install apache2
   ```

   SUSE:

   ```
   zypper install apache2
   ```

2. Download the `etc/apache2/ironic` file from the Ironic project tree and copy it to the apache sites:

   Fedora/RHEL8/CentOS8:

   ```
   sudo cp etc/apache2/ironic /etc/httpd/conf.d/ironic.conf
   ```

   Debian/Ubuntu:

   ```
   sudo cp etc/apache2/ironic /etc/apache2/sites-available/ironic.conf
   ```

   SUSE:

   ```
   sudo cp etc/apache2/ironic /etc/apache2/vhosts.d/ironic.conf
   ```

3. Edit the recently copied `<apache-configuration-dir>/ironic.conf`:

   1. Modify the `WSGIDaemonProcess`, `APACHE_RUN_USER` and `APACHE_RUN_GROUP` directives to set the user and group values to an appropriate user on your server.
   2. Modify the `WSGIScriptAlias` directive to point to the automatically generated `ironic-api-wsgi` script that is located in *IRONIC_BIN* directory.
   3. Modify the `Directory` directive to set the path to the Ironic API code.
   4. Modify the `ErrorLog` and `CustomLog` to redirect the logs to the right directory (on Red Hat systems this is usually under /var/log/httpd).

4. Enable the apache `ironic` in site and reload:

   Fedora/RHEL8/CentOS8:

   ```
   sudo systemctl reload httpd
   ```

   Debian/Ubuntu:

   ```
   sudo a2ensite ironic
   sudo service apache2 reload
   ```

SUSE:

```
sudo systemctl reload apache2
```

Note: The file `ironic-api-wsgi` is automatically generated by pbr and is available in *IRONIC_BIN* directory. It should not be modified.

Configure another WSGI container

A slightly different approach has to be used for WSGI containers that cannot use `ironic-api-wsgi`. For example, for *gunicorn*:

```
gunicorn -b 0.0.0.0:6385 'ironic.api.wsgi:initialize_wsgi_app(argv=[])'
```

If you want to pass a configuration file, use:

```
gunicorn -b 0.0.0.0:6385 \
    'ironic.api.wsgi:initialize_wsgi_app(argv=["ironic-api", "--config-
↪file=/path/to/_ironic.conf"])'
```

Configuring ironic-conductor service

1. Replace `HOST_IP` with IP of the conductor host.

   ```
   [DEFAULT]

   # IP address of this host. If unset, will determine the IP
   # programmatically. If unable to do so, will use "127.0.0.1".
   # (string value)
   my_ip=HOST_IP
   ```

   Note: If a conductor host has multiple IPs, `my_ip` should be set to the IP which is on the same network as the bare metal nodes.

   ---

2. Configure the location of the database. Ironic-conductor should use the same configuration as ironic-api. Replace `IRONIC_DBPASSWORD` with the password of your `ironic` user, and replace DB_IP with the IP address where the DB server is located:

   ```
   [database]

   # The SQLAlchemy connection string to use to connect to the
   # database. (string value)
   connection=mysql+pymysql://ironic:IRONIC_DBPASSWORD@DB_IP/ironic?
   ↪charset=utf8
   ```

3. Configure the ironic-conductor service to use the RabbitMQ message broker by setting the following option. Ironic-conductor should use the same configuration as ironic-api. Replace `RPC_*` with appropriate address details and credentials of RabbitMQ server:

```
[DEFAULT]

# A URL representing the messaging driver to use and its full
# configuration. (string value)
transport_url = rabbit://RPC_USER:RPC_PASSWORD@RPC_HOST:RPC_PORT/
```

Alternatively, you can use JSON RPC for interactions between ironic-conductor and ironic-api. Enable it in the configuration and provide the keystone credentials to use for authenticating incoming requests (can be the same as for the API):

```
[DEFAULT]

rpc_transport = json-rpc

[keystone_authtoken]

# Authentication type to load (string value)
auth_type=password

# Complete public Identity API endpoint (string value)
www_authenticate_uri=http://PUBLIC_IDENTITY_IP:5000

# Complete admin Identity API endpoint. (string value)
auth_url=http://PRIVATE_IDENTITY_IP:5000

# Service username. (string value)
username=ironic

# Service account password. (string value)
password=IRONIC_PASSWORD

# Service tenant name. (string value)
project_name=service

# Domain name containing project (string value)
project_domain_name=Default

# User's domain name (string value)
user_domain_name=Default
```

You can optionally change the host and the port the JSON RPC service will bind to, for example:

```
[json_rpc]
host_ip = 192.168.0.10
port = 9999
```

Warning: Hostnames of ironic-conductor machines must be resolvable by ironic-api services when JSON RPC is used.

4. Configure credentials for accessing other OpenStack services.

   In order to communicate with other OpenStack services, the Bare Metal service needs to use service users to authenticate to the OpenStack Identity service when making requests to other services. These users credentials have to be configured in each configuration file section related to the corresponding service:

- `[neutron]` - to access the OpenStack Networking service
- `[glance]` - to access the OpenStack Image service
- `[swift]` - to access the OpenStack Object Storage service
- `[cinder]` - to access the OpenStack Block Storage service
- `[inspector]` - to access the OpenStack Bare Metal Introspection service
- `[service_catalog]` - a special section holding credentials the Bare Metal service will use to discover its own API URL endpoint as registered in the OpenStack Identity service catalog.

For simplicity, you can use the same service user for all services. For backward compatibility, this should be the same user configured in the `[keystone_authtoken]` section for the ironic-api service (see Configuring ironic-api service). However, this is not necessary, and you can create and configure separate service users for each service.

Under the hood, Bare Metal service uses `keystoneauth` library together with `Authentication plugin`, `Session` and `Adapter` concepts provided by it to instantiate service clients. Please refer to Keystoneauth documentation for supported plugins, their available options as well as Session- and Adapter-related options for authentication, connection and endpoint discovery respectively.

In the example below, authentication information for user to access the OpenStack Networking service is configured to use:

- Networking service is deployed in the Identity service region named `RegionTwo`, with only its `public` endpoint interface registered in the service catalog.
- HTTPS connection with specific CA SSL certificate when making requests
- the same service user as configured for ironic-api service
- dynamic `password` authentication plugin that will discover appropriate version of Identity service API based on other provided options
    - replace `IDENTITY_IP` with the IP of the Identity server, and replace `IRONIC_PASSWORD` with the password you chose for the `ironic` user in the Identity service

```
[neutron]

# Authentication type to load (string value)
auth_type = password

# Authentication URL (string value)
auth_url=https://IDENTITY_IP:5000/

# Username (string value)
username=ironic

# User's password (string value)
password=IRONIC_PASSWORD

# Project name to scope to (string value)
```

(continues on next page)

(continued from previous page)

```
project_name=service

# Domain ID containing project (string value)
project_domain_id=default

# User's domain id (string value)
user_domain_id=default

# PEM encoded Certificate Authority to use when verifying
# HTTPs connections. (string value)
cafile=/opt/stack/data/ca-bundle.pem

# The default region_name for endpoint URL discovery. (string
# value)
region_name = RegionTwo

# List of interfaces, in order of preference, for endpoint
# URL. (list value)
valid_interfaces=public
```

By default, in order to communicate with another service, the Bare Metal service will attempt to discover an appropriate endpoint for that service via the Identity services service catalog. The relevant configuration options from that service group in the Bare Metal service configuration file are used for this purpose. If you want to use a different endpoint for a particular service, specify this via the `endpoint_override` configuration option of that service group, in the Bare Metal services configuration file. Taking the previous Networking service example, this would be

```
[neutron]
...
endpoint_override = <NEUTRON_API_ADDRESS>
```

(Replace *<NEUTRON_API_ADDRESS>* with actual address of a specific Networking service endpoint.)

5. Configure enabled drivers and hardware types as described in *Enabling drivers and hardware types*.

   A. If you enabled any driver that uses *Direct deploy*, Swift backend for the Image service must be installed and configured, see *Configure the Image service for temporary URLs*. Ceph Object Gateway (RADOS Gateway) is also supported as the Image services backend, see *Ceph Object Gateway support*.

6. Configure the network for ironic-conductor service to perform node cleaning, see *Node cleaning* from the admin guide.

7. Restart the ironic-conductor service:

   Fedora/RHEL7/CentOS7/SUSE:

   ```
   sudo systemctl restart openstack-ironic-conductor
   ```

   Ubuntu:

   ```
   sudo service ironic-conductor restart
   ```

Install and configure for openSUSE and SUSE Linux Enterprise

This section describes how to install and configure the Bare Metal service for openSUSE Leap 42.2 and SUSE Linux Enterprise Server 12 SP2.

Note: Installation of the Bare Metal service on openSUSE and SUSE Linux Enterprise Server is not officially supported. Nevertheless, installation should be possible.

Install and configure prerequisites

The Bare Metal service is a collection of components that provides support to manage and provision physical machines. You can configure these components to run on separate nodes or the same node. In this guide, the components run on one node, typically the Compute Services compute node.

It assumes that the Identity, Image, Compute, and Networking services have already been set up.

Set up the database for Bare Metal

The Bare Metal service stores information in a database. This guide uses the MySQL database that is used by other OpenStack services.

1. In MySQL, create an `ironic` database that is accessible by the `ironic` user. Replace `IRONIC_DBPASSWORD` with a suitable password:

   ```
   # mysql -u root -p
   mysql> CREATE DATABASE ironic CHARACTER SET utf8;
   mysql> GRANT ALL PRIVILEGES ON ironic.* TO 'ironic'@'localhost' \
          IDENTIFIED BY 'IRONIC_DBPASSWORD';
   mysql> GRANT ALL PRIVILEGES ON ironic.* TO 'ironic'@'%' \
          IDENTIFIED BY 'IRONIC_DBPASSWORD';
   ```

Install and configure components

1. Install from packages

   ```
   # zypper install openstack-ironic-api openstack-ironic-conductor␣
   ↪python3-ironicclient
   ```

2. Enable services

   ```
   # systemctl enable openstack-ironic-api openstack-ironic-conductor
   # systemctl start openstack-ironic-api openstack-ironic-conductor
   ```

The Bare Metal service is configured via its configuration file. This file is typically located at `/etc/ironic/ironic.conf`.

Although some configuration options are mentioned here, it is recommended that you review all the *Sample Configuration File* so that the Bare Metal service is configured for your needs.

It is possible to set up an ironic-api and an ironic-conductor services on the same host or different hosts. Users also can add new ironic-conductor hosts to deal with an increasing number of bare metal nodes. But the additional ironic-conductor services should be at the same version as that of existing ironic-conductor services.

Configuring ironic-api service

1. The Bare Metal service stores information in a database. This guide uses the MySQL database that is used by other OpenStack services.

   Configure the location of the database via the `connection` option. In the following, replace `IRONIC_DBPASSWORD` with the password of your `ironic` user, and replace `DB_IP` with the IP address where the DB server is located:

   ```
   [database]

   # The SQLAlchemy connection string used to connect to the
   # database (string value)
   connection=mysql+pymysql://ironic:IRONIC_DBPASSWORD@DB_IP/ironic?
   ↪charset=utf8
   ```

2. Configure the ironic-api service to use the RabbitMQ message broker by setting the following option. Replace `RPC_*` with appropriate address details and credentials of RabbitMQ server:

   ```
   [DEFAULT]

   # A URL representing the messaging driver to use and its full
   # configuration. (string value)
   transport_url = rabbit://RPC_USER:RPC_PASSWORD@RPC_HOST:RPC_PORT/
   ```

   Alternatively, you can use JSON RPC for interactions between ironic-conductor and ironic-api. Enable it in the configuration and provide the keystone credentials to use for authentication:

   ```
   [DEFAULT]

   rpc_transport = json-rpc

   [json_rpc]

   # Authentication type to load (string value)
   auth_type = password

   # Authentication URL (string value)
   auth_url=https://IDENTITY_IP:5000/

   # Username (string value)
   username=ironic

   # User's password (string value)
   password=IRONIC_PASSWORD

   # Project name to scope to (string value)
   project_name=service

   # Domain ID containing project (string value)
   project_domain_id=default

   ```

(continues on next page)

(continued from previous page)

```
# User's domain id (string value)
user_domain_id=default
```

If you use port other than the default 8089 for JSON RPC, you have to configure it, for example:

```
[json_rpc]
port = 9999
```

3. Configure the ironic-api service to use these credentials with the Identity service. Replace `PUBLIC_IDENTITY_IP` with the public IP of the Identity server, `PRIVATE_IDENTITY_IP` with the private IP of the Identity server and replace `IRONIC_PASSWORD` with the password you chose for the `ironic` user in the Identity service:

```
[DEFAULT]

# Authentication strategy used by ironic-api: one of
# "keystone" or "noauth". "noauth" should not be used in a
# production environment because all authentication will be
# disabled. (string value)
auth_strategy=keystone

[keystone_authtoken]

# Authentication type to load (string value)
auth_type=password

# Complete public Identity API endpoint (string value)
www_authenticate_uri=http://PUBLIC_IDENTITY_IP:5000

# Complete admin Identity API endpoint. (string value)
auth_url=http://PRIVATE_IDENTITY_IP:5000

# Service username. (string value)
username=ironic

# Service account password. (string value)
password=IRONIC_PASSWORD

# Service tenant name. (string value)
project_name=service

# Domain name containing project (string value)
project_domain_name=Default

# User's domain name (string value)
user_domain_name=Default
```

4. Create the Bare Metal service database tables:

```
$ ironic-dbsync --config-file /etc/ironic/ironic.conf create_schema
```

5. Restart the ironic-api service:

   Fedora/RHEL8/CentOS8/SUSE:

```
sudo systemctl restart openstack-ironic-api
```

Ubuntu:

```
sudo service ironic-api restart
```

Configuring ironic-api behind mod_wsgi

Bare Metal service comes with an example file for configuring the `ironic-api` service to run behind Apache with mod_wsgi.

1. Install the apache service:

   Fedora/RHEL8/CentOS8:

   ```
   sudo dnf install httpd
   ```

   Debian/Ubuntu:

   ```
   apt-get install apache2
   ```

   SUSE:

   ```
   zypper install apache2
   ```

2. Download the `etc/apache2/ironic` file from the Ironic project tree and copy it to the apache sites:

   Fedora/RHEL8/CentOS8:

   ```
   sudo cp etc/apache2/ironic /etc/httpd/conf.d/ironic.conf
   ```

   Debian/Ubuntu:

   ```
   sudo cp etc/apache2/ironic /etc/apache2/sites-available/ironic.conf
   ```

   SUSE:

   ```
   sudo cp etc/apache2/ironic /etc/apache2/vhosts.d/ironic.conf
   ```

3. Edit the recently copied `<apache-configuration-dir>/ironic.conf`:

   1. Modify the `WSGIDaemonProcess`, `APACHE_RUN_USER` and `APACHE_RUN_GROUP` directives to set the user and group values to an appropriate user on your server.

   2. Modify the `WSGIScriptAlias` directive to point to the automatically generated `ironic-api-wsgi` script that is located in *IRONIC_BIN* directory.

   3. Modify the `Directory` directive to set the path to the Ironic API code.

   4. Modify the `ErrorLog` and `CustomLog` to redirect the logs to the right directory (on Red Hat systems this is usually under /var/log/httpd).

4. Enable the apache `ironic` in site and reload:

   Fedora/RHEL8/CentOS8:

```
sudo systemctl reload httpd
```

Debian/Ubuntu:

```
sudo a2ensite ironic
sudo service apache2 reload
```

SUSE:

```
sudo systemctl reload apache2
```

Note: The file `ironic-api-wsgi` is automatically generated by pbr and is available in *IRONIC_BIN* directory. It should not be modified.

Configure another WSGI container

A slightly different approach has to be used for WSGI containers that cannot use `ironic-api-wsgi`. For example, for *gunicorn*:

```
gunicorn -b 0.0.0.0:6385 'ironic.api.wsgi:initialize_wsgi_app(argv=[])'
```

If you want to pass a configuration file, use:

```
gunicorn -b 0.0.0.0:6385 \
    'ironic.api.wsgi:initialize_wsgi_app(argv=["ironic-api", "--config-
↪file=/path/to/_ironic.conf"])'
```

Configuring ironic-conductor service

1. Replace `HOST_IP` with IP of the conductor host.

   ```
   [DEFAULT]

   # IP address of this host. If unset, will determine the IP
   # programmatically. If unable to do so, will use "127.0.0.1".
   # (string value)
   my_ip=HOST_IP
   ```

   Note: If a conductor host has multiple IPs, `my_ip` should be set to the IP which is on the same network as the bare metal nodes.

2. Configure the location of the database. Ironic-conductor should use the same configuration as ironic-api. Replace `IRONIC_DBPASSWORD` with the password of your `ironic` user, and replace DB_IP with the IP address where the DB server is located:

   ```
   [database]
   ```

(continues on next page)

(continued from previous page)

```
# The SQLAlchemy connection string to use to connect to the
# database. (string value)
connection=mysql+pymysql://ironic:IRONIC_DBPASSWORD@DB_IP/ironic?
↪charset=utf8
```

3. Configure the ironic-conductor service to use the RabbitMQ message broker by setting the following option. Ironic-conductor should use the same configuration as ironic-api. Replace `RPC_*` with appropriate address details and credentials of RabbitMQ server:

```
[DEFAULT]

# A URL representing the messaging driver to use and its full
# configuration. (string value)
transport_url = rabbit://RPC_USER:RPC_PASSWORD@RPC_HOST:RPC_PORT/
```

Alternatively, you can use JSON RPC for interactions between ironic-conductor and ironic-api. Enable it in the configuration and provide the keystone credentials to use for authenticating incoming requests (can be the same as for the API):

```
[DEFAULT]

rpc_transport = json-rpc

[keystone_authtoken]

# Authentication type to load (string value)
auth_type=password

# Complete public Identity API endpoint (string value)
www_authenticate_uri=http://PUBLIC_IDENTITY_IP:5000

# Complete admin Identity API endpoint. (string value)
auth_url=http://PRIVATE_IDENTITY_IP:5000

# Service username. (string value)
username=ironic

# Service account password. (string value)
password=IRONIC_PASSWORD

# Service tenant name. (string value)
project_name=service

# Domain name containing project (string value)
project_domain_name=Default

# User's domain name (string value)
user_domain_name=Default
```

You can optionally change the host and the port the JSON RPC service will bind to, for example:

```
[json_rpc]
host_ip = 192.168.0.10
port = 9999
```

> **Warning:** Hostnames of ironic-conductor machines must be resolvable by ironic-api services when JSON RPC is used.

4. Configure credentials for accessing other OpenStack services.

   In order to communicate with other OpenStack services, the Bare Metal service needs to use service users to authenticate to the OpenStack Identity service when making requests to other services. These users credentials have to be configured in each configuration file section related to the corresponding service:

   - `[neutron]` - to access the OpenStack Networking service
   - `[glance]` - to access the OpenStack Image service
   - `[swift]` - to access the OpenStack Object Storage service
   - `[cinder]` - to access the OpenStack Block Storage service
   - `[inspector]` - to access the OpenStack Bare Metal Introspection service
   - `[service_catalog]` - a special section holding credentials the Bare Metal service will use to discover its own API URL endpoint as registered in the OpenStack Identity service catalog.

   For simplicity, you can use the same service user for all services. For backward compatibility, this should be the same user configured in the `[keystone_authtoken]` section for the ironic-api service (see Configuring ironic-api service). However, this is not necessary, and you can create and configure separate service users for each service.

   Under the hood, Bare Metal service uses `keystoneauth` library together with `Authentication plugin`, `Session` and `Adapter` concepts provided by it to instantiate service clients. Please refer to Keystoneauth documentation for supported plugins, their available options as well as Session- and Adapter-related options for authentication, connection and endpoint discovery respectively.

   In the example below, authentication information for user to access the OpenStack Networking service is configured to use:

   - Networking service is deployed in the Identity service region named `RegionTwo`, with only its `public` endpoint interface registered in the service catalog.
   - HTTPS connection with specific CA SSL certificate when making requests
   - the same service user as configured for ironic-api service
   - dynamic `password` authentication plugin that will discover appropriate version of Identity service API based on other provided options
     - replace `IDENTITY_IP` with the IP of the Identity server, and replace `IRONIC_PASSWORD` with the password you chose for the `ironic` user in the Identity service

```
[neutron]

# Authentication type to load (string value)
auth_type = password

# Authentication URL (string value)
```

(continues on next page)

(continued from previous page)

```
auth_url=https://IDENTITY_IP:5000/

# Username (string value)
username=ironic

# User's password (string value)
password=IRONIC_PASSWORD

# Project name to scope to (string value)
project_name=service

# Domain ID containing project (string value)
project_domain_id=default

# User's domain id (string value)
user_domain_id=default

# PEM encoded Certificate Authority to use when verifying
# HTTPs connections. (string value)
cafile=/opt/stack/data/ca-bundle.pem

# The default region_name for endpoint URL discovery. (string
# value)
region_name = RegionTwo

# List of interfaces, in order of preference, for endpoint
# URL. (list value)
valid_interfaces=public
```

By default, in order to communicate with another service, the Bare Metal service will attempt to discover an appropriate endpoint for that service via the Identity services service catalog. The relevant configuration options from that service group in the Bare Metal service configuration file are used for this purpose. If you want to use a different endpoint for a particular service, specify this via the `endpoint_override` configuration option of that service group, in the Bare Metal services configuration file. Taking the previous Networking service example, this would be

```
[neutron]
...
endpoint_override = <NEUTRON_API_ADDRESS>
```

(Replace <*NEUTRON_API_ADDRESS*> with actual address of a specific Networking service endpoint.)

5. Configure enabled drivers and hardware types as described in *Enabling drivers and hardware types*.

   A. If you enabled any driver that uses *Direct deploy*, Swift backend for the Image service must be installed and configured, see *Configure the Image service for temporary URLs*. Ceph Object Gateway (RADOS Gateway) is also supported as the Image services backend, see *Ceph Object Gateway support*.

6. Configure the network for ironic-conductor service to perform node cleaning, see *Node cleaning* from the admin guide.

7. Restart the ironic-conductor service:

Fedora/RHEL7/CentOS7/SUSE:

```
sudo systemctl restart openstack-ironic-conductor
```

Ubuntu:

```
sudo service ironic-conductor restart
```

2.1.4 Create user images for the Bare Metal service

Bare Metal provisioning requires two sets of images: the deploy images and the user images. The *deploy images* are used by the Bare Metal service to prepare the bare metal server for actual OS deployment. Whereas the user images are installed on the bare metal server to be used by the end user. There are two types of user images:

partition images contain only the contents of the root partition. Additionally, two more images are used together with them: an image with a kernel and with an initramfs.

> **Warning:** To use partition images with local boot, Grub2 must be installed on them.

whole disk images contain a complete partition table with one or more partitions.

> **Warning:** The kernel/initramfs pair must not be used with whole disk images, otherwise theyll be mistaken for partition images.

Many distributions publish their own cloud images. These are usually whole disk images that are built for legacy boot mode (not UEFI), with Ubuntu being an exception (they publish images that work in both modes).

Building user images

disk-image-builder

The disk-image-builder can be used to create user images required for deployment and the actual OS which the user is going to run.

- Install diskimage-builder package (use virtualenv, if you dont want to install anything globally):

  ```
  # pip install diskimage-builder
  ```

- Build the image your users will run (Ubuntu image has been taken as an example):
  - Partition images

    ```
    $ disk-image-create ubuntu baremetal dhcp-all-interfaces grub2 -o␣
    ↪my-image
    ```

  - Whole disk images

```
$ disk-image-create ubuntu vm dhcp-all-interfaces -o my-image
```

with an EFI partition:

```
$ disk-image-create ubuntu vm block-device-efi dhcp-all-
↪interfaces -o my-image
```

The partition image command creates `my-image.qcow2`, `my-image.vmlinuz` and `my-image.initrd` files. The `grub2` element in the partition image creation command is only needed if local boot will be used to deploy `my-image.qcow2`, otherwise the images `my-image.vmlinuz` and `my-image.initrd` will be used for PXE booting after deploying the bare metal with `my-image.qcow2`. For whole disk images only the main image is used.

If you want to use Fedora image, replace `ubuntu` with `fedora` in the chosen command.

Virtual machine

Virtual machine software can also be used to build user images. There are different software options available, qemu-kvm is usually a good choice on linux platform, it supports emulating many devices and even building images for architectures other than the host machine by software emulation. VirtualBox is another good choice for non-linux host.

The procedure varies depending on the software used, but the steps for building an image are similar, the user creates a virtual machine, and installs the target system just like what is done for a real hardware. The system can be highly customized like partition layout, drivers or software shipped, etc.

Usually libvirt and its management tools are used to make interaction with qemu-kvm easier, for example, to create a virtual machine with `virt-install`:

```
$ virt-install --name centos8 --ram 4096 --vcpus=2 -f centos8.qcow2 \
> --cdrom CentOS-8-x86_64-1905-dvd1.iso
```

Graphic frontend like `virt-manager` can also be utilized.

The disk file can be used as user image after the system is set up and powered off. The path of the disk file varies depending on the software used, usually its stored in a user-selected part of the local file system. For qemu-kvm or GUI frontend building upon it, its typically stored at `/var/lib/libvirt/images`.

2.1.5 Building or downloading a deploy ramdisk image

Ironic depends on having an image with the ironic-python-agent (IPA) service running on it for controlling and deploying bare metal nodes.

Two kinds of images are published on every commit from every branch of ironic-python-agent (IPA)

- DIB images are suitable for production usage and can be downloaded from https://tarballs.openstack.org/ironic-python-agent/dib/files/.
  - For Train and older use CentOS 7 images.
  - For Ussuri and newer use CentOS 8 images.

Warning: CentOS 7 master images are no longer updated and must not be used.

Warning: The published images will not work for dhcp-less deployments since the simple-init element is not present. Check the DIB documentation to see how to build the image.

- TinyIPA images are suitable for CI and testing environments and can be downloaded from https://tarballs.openstack.org/ironic-python-agent/tinyipa/files/.

Building from source

Check the ironic-python-agent-builder project for information on how to build ironic-python-agent ramdisks.

2.1.6 Integration with other OpenStack services

Configure the Identity service for the Bare Metal service

1. Create the Bare Metal service user (for example, `ironic`). The service uses this to authenticate with the Identity service. Use the `service` tenant and give the user the `admin` role:

```
$ openstack user create --password IRONIC_PASSWORD \
    --email ironic@example.com ironic
$ openstack role add --project service --user ironic admin
```

2. You must register the Bare Metal service with the Identity service so that other OpenStack services can locate it. To register the service:

```
$ openstack service create --name ironic --description \
    "Ironic baremetal provisioning service" baremetal
```

3. Use the `id` property that is returned from the Identity service when registering the service (above), to create the endpoint, and replace `IRONIC_NODE` with your Bare Metal services API node:

```
$ openstack endpoint create --region RegionOne \
    baremetal admin http://$IRONIC_NODE:6385
$ openstack endpoint create --region RegionOne \
    baremetal public http://$IRONIC_NODE:6385
$ openstack endpoint create --region RegionOne \
    baremetal internal http://$IRONIC_NODE:6385
```

4. You may delegate limited privileges related to the Bare Metal service to your Users by creating Roles with the OpenStack Identity service. By default, the Bare Metal service expects the baremetal_admin and baremetal_observer Roles to exist, in addition to the default admin Role. There is no negative consequence if you choose not to create these Roles. They can be created with the following commands:

```
$ openstack role create baremetal_admin
$ openstack role create baremetal_observer
```

If you choose to customize the names of Roles used with the Bare Metal service, do so by changing the is_member, is_observer, and is_admin policy settings in `/etc/ironic/policy.yaml`.

More complete documentation on managing Users and Roles within your OpenStack deployment are outside the scope of this document, but may be found here.

5. You can further restrict access to the Bare Metal service by creating a separate baremetal Project, so that Bare Metal resources (Nodes, Ports, etc) are only accessible to members of this Project:

```
$ openstack project create baremetal
```

At this point, you may grant read-only access to the Bare Metal service API without granting any other access by issuing the following commands:

```
$ openstack user create \
    --domain default --project-domain default --project baremetal \
    --password PASSWORD USERNAME
$ openstack role add \
    --user-domain default --project-domain default --project␣
↪baremetal \
    --user USERNAME baremetal_observer
```

6. Further documentation is available elsewhere for the `openstack` command-line client and the Identity service. A *policy.yaml.sample* file, which enumerates the services default policies, is provided for your convenience with the Bare Metal Service.

Configure the Compute service to use the Bare Metal service

The Compute service needs to be configured to use the Bare Metal services driver. The configuration file for the Compute service is typically located at `/etc/nova/nova.conf`.

Note: As of the Newton release, it is possible to have multiple nova-compute services running the ironic virtual driver (in nova) to provide redundancy. Bare metal nodes are mapped to the services via a hash ring. If a service goes down, the available bare metal nodes are remapped to different services.

Once active, a node will stay mapped to the same nova-compute even when it goes down. The node is unable to be managed through the Compute API until the service responsible returns to an active state.

The following configuration file must be modified on the Compute services controller nodes and compute nodes.

1. Change these configuration options in the Compute service configuration file (for example, `/etc/nova/nova.conf`):

```
[default]

# Defines which driver to use for controlling virtualization.
# Enable the ironic virt driver for this compute instance.
compute_driver=ironic.IronicDriver

# Amount of memory in MB to reserve for the host so that it is always
# available to host processes.
# It is impossible to reserve any memory on bare metal nodes, so set
# this to zero.
```

(continues on next page)

(continued from previous page)

```
reserved_host_memory_mb=0

[filter_scheduler]

# Enables querying of individual hosts for instance information.
# Not possible for bare metal nodes, so set it to False.
track_instance_changes=False

[scheduler]

# This value controls how often (in seconds) the scheduler should
# attempt to discover new hosts that have been added to cells.
# If negative (the default), no automatic discovery will occur.
# As each bare metal node is represented by a separate host, it has
# to be discovered before the Compute service can deploy on it.
# The value here has to be carefully chosen based on a compromise
# between the enrollment speed and the load on the Compute scheduler.
# The recommended value of 2 minutes matches how often the Compute
# service polls the Bare Metal service for node information.
discover_hosts_in_cells_interval=120
```

Note: The alternative to setting the `discover_hosts_in_cells_interval` option is to run the following command on any Compute controller node after each node is enrolled:

```
nova-manage cell_v2 discover_hosts --by-service
```

---

2. Consider enabling the following option on controller nodes:

```
[filter_scheduler]

# Enabling this option is beneficial as it reduces re-scheduling␣
↪events
# for ironic nodes when scheduling is based on resource classes,
# especially for mixed hypervisor case with host_subset_size = 1.
# However enabling it will also make packing of VMs on hypervisors
# less dense even when scheduling weights are completely disabled.
#shuffle_best_same_weighed_hosts = false
```

3. Carefully consider the following option:

```
[compute]

# This option will cause nova-compute to set itself to a disabled␣
↪state
# if a certain number of consecutive build failures occur. This will
# prevent the scheduler from continuing to send builds to a compute
# service that is consistently failing. In the case of bare metal
# provisioning, however, a compute service is rarely the cause of␣
↪build
# failures. Furthermore, bare metal nodes, managed by a disabled
# compute service, will be remapped to a different one. That may cause
# the second compute service to also be disabled, and so on, until no
# compute services are active.
```

(continues on next page)

(continued from previous page)

```
# If this is not the desired behavior, consider increasing this value␣
↪or
# setting it to 0 to disable this behavior completely.
#consecutive_build_service_disable_threshold = 10
```

4. Change these configuration options in the `ironic` section. Replace:
   - `IRONIC_PASSWORD` with the password you chose for the `ironic` user in the Identity Service
   - `IRONIC_NODE` with the hostname or IP address of the ironic-api node
   - `IDENTITY_IP` with the IP of the Identity server

```
[ironic]

# Ironic authentication type
auth_type=password

# Keystone API endpoint
auth_url=http://IDENTITY_IP:5000/v3

# Ironic keystone project name
project_name=service

# Ironic keystone admin name
username=ironic

# Ironic keystone admin password
password=IRONIC_PASSWORD

# Ironic keystone project domain
# or set project_domain_id
project_domain_name=Default

# Ironic keystone user domain
# or set user_domain_id
user_domain_name=Default
```

5. On the Compute services controller nodes, restart the `nova-scheduler` process:

```
Fedora/RHEL8/CentOS8/SUSE:
  sudo systemctl restart openstack-nova-scheduler

Ubuntu:
  sudo service nova-scheduler restart
```

6. On the Compute services compute nodes, restart the `nova-compute` process:

```
Fedora/RHEL8/CentOS8/SUSE:
  sudo systemctl restart openstack-nova-compute

Ubuntu:
  sudo service nova-compute restart
```

Configure the Networking service for bare metal provisioning

You need to configure Networking so that the bare metal server can communicate with the Networking service for DHCP, PXE boot and other requirements. This section covers configuring Networking for a single flat network for bare metal provisioning.

It is recommended to use the baremetal ML2 mechanism driver and L2 agent for proper integration with the Networking service. Documentation regarding installation and configuration of the baremetal mechanism driver and L2 agent is available here.

For use with routed networks the baremetal ML2 components are required.

Note: When the baremetal ML2 components are *not* used, ports in the Networking service will have status: `DOWN`, and binding_vif_type: `binding_failed`. This was always the status for Bare Metal service `flat` network interface ports prior to the introduction of the baremetal ML2 integration. For a non-routed network, bare metal servers can still be deployed and are functional, despite this port binding state in the Networking service.

You will also need to provide Bare Metal service with the MAC address(es) of each node that it is provisioning; Bare Metal service in turn will pass this information to Networking service for DHCP and PXE boot configuration. An example of this is shown in the *Enrollment* section.

1. Install the networking-baremetal ML2 mechanism driver and L2 agent in the Networking service.
2. Edit `/etc/neutron/plugins/ml2/ml2_conf.ini` and modify these:

```
[ml2]
type_drivers = flat
tenant_network_types = flat
mechanism_drivers = openvswitch,baremetal

[ml2_type_flat]
flat_networks = physnet1

[securitygroup]
firewall_driver = neutron.agent.linux.iptables_firewall.
↪OVSHybridIptablesFirewallDriver
enable_security_group = True

[ovs]
bridge_mappings = physnet1:br-eth2
# Replace eth2 with the interface on the neutron node which you
# are using to connect to the bare metal server
```

3. Restart the `neutron-server` service, to load the new configuration.
4. Create and edit `/etc/neutron/plugins/ml2/ironic_neutron_agent.ini` and add the required configuration. For example:

```
[ironic]
project_domain_name = Default
project_name = service
user_domain_name = Default
password = password
username = ironic
```

(continues on next page)

(continued from previous page)

```
auth_url = http://identity-server.example.com/identity
auth_type = password
region_name = RegionOne
```

5. Make sure the `ironic-neutron-agent` service is started.
6. If neutron-openvswitch-agent runs with `ovs_neutron_plugin.ini` as the input config-file, edit `ovs_neutron_plugin.ini` to configure the bridge mappings by adding the [ovs] section described in the previous step, and restart the neutron-openvswitch-agent.
7. Add the integration bridge to Open vSwitch:

```
$ ovs-vsctl add-br br-int
```

8. Create the br-eth2 network bridge to handle communication between the OpenStack services (and the Bare Metal services) and the bare metal nodes using eth2. Replace eth2 with the interface on the network node which you are using to connect to the Bare Metal service:

```
$ ovs-vsctl add-br br-eth2
$ ovs-vsctl add-port br-eth2 eth2
```

9. Restart the Open vSwitch agent:

```
# service neutron-plugin-openvswitch-agent restart
```

10. On restarting the Networking service Open vSwitch agent, the veth pair between the bridges br-int and br-eth2 is automatically created.

    Your Open vSwitch bridges should look something like this after following the above steps:

```
$ ovs-vsctl show

    Bridge br-int
        fail_mode: secure
        Port "int-br-eth2"
            Interface "int-br-eth2"
                type: patch
                options: {peer="phy-br-eth2"}
        Port br-int
            Interface br-int
                type: internal
    Bridge "br-eth2"
        Port "phy-br-eth2"
            Interface "phy-br-eth2"
                type: patch
                options: {peer="int-br-eth2"}
        Port "eth2"
            Interface "eth2"
        Port "br-eth2"
            Interface "br-eth2"
                type: internal
    ovs_version: "2.3.0"
```

11. Create the flat network on which you are going to launch the instances:

```
$ openstack network create --project $TENANT_ID sharednet1 --share \
    --provider-network-type flat --provider-physical-network physnet1
```

12. Create the subnet on the newly created network:

```
$ openstack subnet create $SUBNET_NAME --network sharednet1 \
  --subnet-range $NETWORK_CIDR --ip-version 4 --gateway $GATEWAY_IP \
  --allocation-pool start=$START_IP,end=$END_IP --dhcp
```

Configuring services for bare metal provisioning using IPv6

Use of IPv6 addressing for baremetal provisioning requires additional configuration. This page covers the IPv6 specifics only. Please refer to *Configure tenant networks* and *Configure the Networking service for bare metal provisioning* for general networking configuration.

Configure ironic PXE driver for provisioning using IPv6 addressing

The ironic PXE driver operates in either IPv4 or IPv6 mode (IPv4 is the default). To enable IPv6 mode, set the `[pxe]/ip_version` option in the Bare Metal Services configuration file (`/etc/ironic/ironic.conf`) to `6`.

Note: Support for dual mode IPv4 and IPv6 operations is planned for a future version of ironic.

Provisioning with IPv6 stateless addressing

When using stateless addressing DHCPv6 does not provide addresses to the client. DHCPv6 however provides other configuration via DHCPv6 options such as the bootfile-url and bootfile-parameters.

Once the PXE driver is set to operate in IPv6 mode no further configuration is required in the Baremetal Service.

Creating networks and subnets in the Networking Service

When creating the Baremetal Service network(s) and subnet(s) in the Networking Services, subnets should have `ipv6-address-mode` set to `dhcpv6-stateless` and `ip-version` set to `6`. Depending on whether a router in the Networking Service is providing RAs (Router Advertisements) or not, the `ipv6-ra-mode` for the subnet(s) should either be set to `dhcpv6-stateless` or be left unset.

Note: If `ipv6-ra-mode` is left unset, an external router on the network is expected to provide RAs with the appropriate flags set for automatic addressing and other configuration.

Provisioning with IPv6 stateful addressing

When using stateful addressing DHCPv6 is providing both addresses and other configuration via DHCPv6 options such as the bootfile-url and bootfile- parameters.

The identity-association (IA) construct used by DHCPv6 is challenging when booting over the network. Firmware, and ramdisks typically end up using different DUID/IAID combinations and it is not always possible for one chain- booting stage to release its address before giving control to the next step. In case the DHCPv6 server is configured with static reservations only the result is that booting will fail because the DHCPv6 server has no addresses available. To get past this issue either configure the DHCPv6 server with multiple address reservations for each host, or use a dynamic range.

Note: Support for multiple address reservations requires dnsmasq version 2.81 or later. Some distributions may backport this feature to earlier dnsmasq version as part of the packaging, check the distributions release notes.

If a different (not dnsmasq) DHCPv6 server backend is used with the Networking service, use of multiple address reservations might not work.

Using the `flat` network interface

Due to the identity-association challenges with DHCPv6 provisioning using the `flat` network interface is not recommended. When ironic operates with the `flat` network interface the server instance port is used for provisioning and other operations. Ironic will not use multiple address reservations in this scenario. Because of this **it will not work in most cases**.

Using the `neutron` network interface

When using the `neutron` network interface the Baremetal Service will allocate multiple IPv6 addresses (4 addresses per port by default) on the service networks used for provisioning, cleaning, rescue and introspection. The number of addresses allocated can be controlled via the `[neutron]/dhcpv6_stateful_address_count` option in the Bare Metal Services configuration file (`/etc/ironic/ironic.conf`). Using multiple address reservations ensures that the DHCPv6 server can lease addresses to each step.

To enable IPv6 provisioning on neutron *flat* provider networks with no switch management, the `local_link_connection` field of baremetal ports must be set to `{'network_type': 'unmanaged'}`. The following example shows how to set the local_link_connection for operation on unmanaged networks:

```
baremetal port set \
  --local-link-connection network_type=unmanaged <port-uuid>
```

The use of multiple IPv6 addresses must also be enabled in the Networking Services dhcp agent configuration (`/etc/neutron/dhcp_agent.ini`) by setting the option `[DEFAULT]/dnsmasq_enable_addr6_list` to `True` (default `False` in Ussuri release).

Note: Support for multiple IPv6 address reservations in the dnsmasq backend was added to the Networking Service Ussuri release. It was also backported to the stable Train release.

Creating networks and subnets in the Networking Service

When creating the ironic service network(s) and subnet(s) in the Networking Service, subnets should have `ipv6-address-mode` set to `dhcpv6-stateful` and `ip-version` set to `6`. Depending on whether a router in the Networking Service is providing RAs (Router Advertisements) or not, the `ipv6-ra-mode` for the subnet(s) should be set to either `dhcpv6-stateful` or be left unset.

Note: If `ipv6-ra-mode` is left unset, an external router on the network is expected to provide RAs with the appropriate flags set for managed addressing and other configuration.

Configure the Image service for temporary URLs

Some drivers of the Baremetal service (in particular, any drivers using *Direct deploy* or *Ansible deploy* interfaces, and some virtual media drivers) require target user images to be available over clean HTTP(S) URL with no authentication involved (neither username/password-based, nor token-based).

When using the Baremetal service integrated in OpenStack, this can be achieved by specific configuration of the Image service and Object Storage service as described below.

1. Configure the Image service to have object storage as a backend for storing images. For more details, please refer to the Image service configuration guide.

   Note: When using Ceph+RadosGW for Object Storage service, images stored in Image service must be available over Object Storage service as well.

2. Enable TempURLs for the Object Storage account used by the Image service for storing images in the Object Storage service.

   1. Check if TempURLs are enabled:

      ```
      # executed under credentials of the user used by Image service
      # to access Object Storage service
      $ openstack object store account show
      +------------+---------------------------------------+
      | Field      | Value                                 |
      +------------+---------------------------------------+
      | Account    | AUTH_bc39f1d9dcf9486899088007789ae643 |
      | Bytes      | 536661727                             |
      | Containers | 1                                     |
      | Objects    | 19                                    |
      | properties | Temp-Url-Key='secret'                 |
      +------------+---------------------------------------+
      ```

   2. If property `Temp-Url-Key` is set, note its value.

   3. If property `Temp-Url-Key` is not set, you have to configure it (`secret` is used in the example below for the value):

```
$ openstack object store account set --property Temp-Url-
↪Key=secret
```

3. Optionally, configure the ironic-conductor service. The default configuration assumes that:

   1. the Object Storage service is implemented by swift,
   2. the Object Storage service URL is available from the service catalog,
   3. the project, used by the Image service to access the Object Storage, is the same as the project, used by the Bare Metal service to access it,
   4. the container, used by the Image service, is called `glance`.

   If any of these assumptions do not hold, you may want to change your configuration file (typically located at `/etc/ironic/ironic.conf`), for example:

```
[glance]

swift_endpoint_url = http://openstack/swift
swift_account = AUTH_bc39f1d9dcf9486899088007789ae643
swift_container = glance
swift_temp_url_key = secret
```

4. (Re)start the ironic-conductor service.

Enabling HTTPS

Enabling HTTPS in Swift

The drivers using virtual media use swift for storing boot images and node configuration information (contains sensitive information for Ironic conductor to provision bare metal hardware). By default, HTTPS is not enabled in swift. HTTPS is required to encrypt all communication between swift and Ironic conductor and swift and bare metal (via virtual media). It can be enabled in one of the following ways:

- Using an SSL termination proxy
- Using native SSL support in swift (recommended only for testing purpose by swift).

Enabling HTTPS in Image service

Ironic drivers usually use Image service during node provisioning. By default, image service does not use HTTPS, but it is required for secure communication. It can be enabled by making the following changes to `/etc/glance/glance-api.conf`:

1. Configuring SSL support
2. Restart the glance-api service:

```
Fedora/RHEL8/CentOS8/SUSE:
    sudo systemctl restart openstack-glance-api

Debian/Ubuntu:
    sudo service glance-api restart
```

See the Glance documentation, for more details on the Image service.

Enabling HTTPS communication between Image service and Object storage

This section describes the steps needed to enable secure HTTPS communication between Image service and Object storage when Object storage is used as the Backend.

To enable secure HTTPS communication between Image service and Object storage follow these steps:

1. *Enabling HTTPS in Swift*
2. Configure Swift Storage Backend
3. *Enabling HTTPS in Image service*

Enabling HTTPS communication between Image service and Bare Metal service

This section describes the steps needed to enable secure HTTPS communication between Image service and Bare Metal service.

To enable secure HTTPS communication between Bare Metal service and Image service follow these steps:

1. Edit `/etc/ironic/ironic.conf`:

   ```
   [glance]
   ...
   glance_cafile=/path/to/certfile
   ```

   Note: glance_cafile is an optional path to a CA certificate bundle to be used to validate the SSL certificate served by Image service.

   ---

2. If not using the keystone service catalog for the Image service API endpoint discovery, also edit the `endpoint_override` option to point to HTTPS URL of image service (replace `<GLANCE_API_ADDRESS>` with hostname[:port][path] of the Image service endpoint):

   ```
   [glance]
   ...
   endpoint_override = https://<GLANCE_API_ADDRESS>
   ```

3. Restart ironic-conductor service:

   ```
   Fedora/RHEL8/CentOS8/SUSE:
       sudo systemctl restart openstack-ironic-conductor

   Debian/Ubuntu:
       sudo service ironic-conductor restart
   ```

Configure the Bare Metal service for cleaning

Note: If you configured the Bare Metal service to do *Automated cleaning* (which is enabled by default), you will need to set the `cleaning_network` configuration option.

---

1. Note the network UUID (the *id* field) of the network you created in *Configure the Networking service for bare metal provisioning* or another network you created for cleaning:

   ```
   $ openstack network list
   ```

2. Configure the cleaning network UUID via the `cleaning_network` option in the Bare Metal service configuration file (`/etc/ironic/ironic.conf`). In the following, replace `NETWORK_UUID` with the UUID you noted in the previous step:

   ```
   [neutron]
   cleaning_network = NETWORK_UUID
   ```

3. Restart the Bare Metal services ironic-conductor:

   ```
   Fedora/RHEL8/CentOS8/SUSE:
     sudo systemctl restart openstack-ironic-conductor

   Ubuntu:
     sudo service ironic-conductor restart
   ```

Configure tenant networks

Below is an example flow of how to set up the Bare Metal service so that node provisioning will happen in a multi-tenant environment (which means using the `neutron` network interface as stated above):

1. Network interfaces can be enabled on ironic-conductor by adding them to the `enabled_network_interfaces` configuration option under the `default` section of the configuration file:

   ```
   [DEFAULT]
   ...
   enabled_network_interfaces=noop,flat,neutron
   ```

   Keep in mind that, ideally, all ironic-conductors should have the same list of enabled network interfaces, but it may not be the case during ironic-conductor upgrades. This may cause problems if one of the ironic-conductors dies and some node that is taken over is mapped to an ironic-conductor that does not support the nodes network interface. Any actions that involve calling the nodes driver will fail until that network interface is installed and enabled on that ironic-conductor.

2. It is recommended to set the default network interface via the `default_network_interface` configuration option under the `default` section of the configuration file:

   ```
   [DEFAULT]
   ...
   default_network_interface=neutron
   ```

This default value will be used for all nodes that dont have a network interface explicitly specified in the creation request.

If this configuration option is not set, the default network interface is determined by looking at the `[dhcp]dhcp_provider` configuration option value. If it is `neutron`, then `flat` network interface becomes the default, otherwise `noop` is the default.

3. Define a provider network in the Networking service, which we shall refer to as the provisioning network. Using the `neutron` network interface requires that `provisioning_network` and `cleaning_network` configuration options are set to valid identifiers (UUID or name) of networks in the Networking service. If these options are not set correctly, cleaning or provisioning will fail to start. There are two ways to set these values:

   - Under the `neutron` section of ironic configuration file:

     ```
     [neutron]
     cleaning_network = $CLEAN_UUID_OR_NAME
     provisioning_network = $PROVISION_UUID_OR_NAME
     ```

   - Under `provisioning_network` and `cleaning_network` keys of the nodes `driver_info` field as `driver_info['provisioning_network']` and `driver_info['cleaning_network']` respectively.

Note: If these `provisioning_network` and `cleaning_network` values are not specified in nodes *driver_info* then ironic falls back to the configuration in the `neutron` section.

Please refer to *Configure the Bare Metal service for cleaning* for more information about cleaning.

Warning: Please make sure that the Bare Metal service has exclusive access to the provisioning and cleaning networks. Spawning instances by non-admin users in these networks and getting access to the Bare Metal services control plane is a security risk. For this reason, the provisioning and cleaning networks should be configured as non-shared networks in the `admin` tenant.

Note: When using the `flat` network interface, bare metal instances are normally spawned onto the provisioning network. This is not supported with the `neutron` interface and the deployment will fail. Please ensure a different network is chosen in the Networking service when a bare metal instance is booted from the Compute service.

Note: The provisioning and cleaning networks may be the same network or distinct networks. To ensure that communication between the Bare Metal service and the deploy ramdisk works, it is important to ensure that security groups are disabled for these networks, *or* that the default security groups allow:

- DHCP
- TFTP
- egress port used for the Bare Metal service (6385 by default)
- ingress port used for ironic-python-agent (9999 by default)

- if using *iSCSI deploy*, the ingress port used for iSCSI (3260 by default)
- if using *Direct deploy*, the egress port used for the Object Storage service (typically 80 or 443)
- if using iPXE, the egress port used for the HTTP server running on the ironic-conductor nodes (typically 80).

---

4. This step is optional and applicable only if you want to use security groups during provisioning and/or cleaning of the nodes. If not specified, default security groups are used.

    1. Define security groups in the Networking service, to be used for provisioning and/or cleaning networks.
    2. Add the list of these security group UUIDs under the `neutron` section of ironic-conductors configuration file as shown below:

    ```
    [neutron]
    ...
    cleaning_network=$CLEAN_UUID_OR_NAME
    cleaning_network_security_groups=[$LIST_OF_CLEAN_SECURITY_GROUPS]
    provisioning_network=$PROVISION_UUID_OR_NAME
    provisioning_network_security_groups=[$LIST_OF_PROVISION_SECURITY_
     ↪GROUPS]
    ```

    Multiple security groups may be applied to a given network, hence, they are specified as a list. The same security group(s) could be used for both provisioning and cleaning networks.

> **Warning:** If security groups are configured as described above, do not set the port_security_enabled flag to False for the corresponding Networking services network or port. This will cause the deploy to fail.
>
> For example: if `provisioning_network_security_groups` configuration option is used, ensure that port_security_enabled flag for the provisioning network is set to True. This flag is set to True by default; make sure not to override it by manually setting it to False.

5. Install and configure a compatible ML2 mechanism driver which supports bare metal provisioning for your switch. See ML2 plugin configuration manual for details.
6. Restart the ironic-conductor and ironic-api services after the modifications:

    - Fedora/RHEL8/CentOS8:

    ```
    sudo systemctl restart openstack-ironic-api
    sudo systemctl restart openstack-ironic-conductor
    ```

    - Ubuntu:

    ```
    sudo service ironic-api restart
    sudo service ironic-conductor restart
    ```

7. Make sure that the ironic-conductor is reachable over the provisioning network by trying to download a file from a TFTP server on it, from some non-control-plane server in that network:

```
tftp $TFTP_IP -c get $FILENAME
```

where FILENAME is the file located at the TFTP server.

See *Multi-tenancy in the Bare Metal service* for required node configuration.

Add images to the Image service

1. Build or download the user images as described in *Create user images for the Bare Metal service*.
2. Add the user images to the Image service

   Load all the images created in the below steps into the Image service, and note the image UUIDs in the Image service for each one as it is generated.

   For *partition images*:

   - Add the kernel and ramdisk images to the Image service:

     ```
     $ openstack image create my-kernel --public \
       --disk-format aki --container-format aki --file my-image.vmlinuz
     ```

     Store the image uuid obtained from the above step as `MY_VMLINUZ_UUID`.

     ```
     $ openstack image create my-image.initrd --public \
       --disk-format ari --container-format ari --file my-image.initrd
     ```

     Store the image UUID obtained from the above step as `MY_INITRD_UUID`.

   - Add the *my-image* to the Image service which is going to be the OS that the user is going to run. Also associate the above created images with this OS image. These two operations can be done by executing the following command:

     ```
     $ openstack image create my-image --public \
       --disk-format qcow2 --container-format bare --property \
       kernel_id=$MY_VMLINUZ_UUID --property \
       ramdisk_id=$MY_INITRD_UUID --file my-image.qcow2
     ```

   For *whole disk images*, skip uploading and configuring kernel and ramdisk images completely, proceed directly to uploading the main image:

   ```
   $ openstack image create my-whole-disk-image --public \
     --disk-format qcow2 --container-format bare \
     --file my-whole-disk-image.qcow2
   ```

   > **Warning:** The kernel/initramfs pair must not be set for whole disk images, otherwise theyll be mistaken for partition images.

3. Build or download the deploy images

   The deploy images are used initially for preparing the server (creating disk partitions) before the actual OS can be deployed.

   There are several methods to build or download deploy images, please read the *Building or downloading a deploy ramdisk image* section.

4. Add the deploy images to the Image service

   Add the deployment kernel and ramdisk images to the Image service:

   ```
   $ openstack image create deploy-vmlinuz --public \
     --disk-format aki --container-format aki \
     --file ironic-python-agent.vmlinuz
   ```

   Store the image UUID obtained from the above step as `DEPLOY_VMLINUZ_UUID`.

   ```
   $ openstack image create deploy-initrd --public \
     --disk-format ari --container-format ari \
     --file ironic-python-agent.initramfs
   ```

   Store the image UUID obtained from the above step as `DEPLOY_INITRD_UUID`.

Create flavors for use with the Bare Metal service

Youll need to create a special bare metal flavor in the Compute service. The flavor is mapped to the bare metal node through the nodes `resource_class` field (available starting with Bare Metal API version 1.21). A flavor can request *exactly one* instance of a bare metal resource class.

Note that when creating the flavor, its useful to add the `RAM_MB` and `CPU` properties as a convenience to users, although they are not used for scheduling. The `DISK_GB` property is also not used for scheduling, but is still used to determine the root partition size.

1. Change these to match your hardware:

   ```
   $ RAM_MB=1024
   $ CPU=2
   $ DISK_GB=100
   ```

2. Create the bare metal flavor by executing the following command:

   ```
   $ openstack flavor create --ram $RAM_MB --vcpus $CPU --disk $DISK_GB \
     my-baremetal-flavor
   ```

   Note: You can add `--id <id>` to specify an ID for the flavor.

See the docs on this command for other options that may be specified.

After creation, associate each flavor with one custom resource class. The name of a custom resource class that corresponds to a nodes resource class (in the Bare Metal service) is:

- the bare metal nodes resource class all upper-cased
- prefixed with `CUSTOM_`
- all punctuation replaced with an underscore

For example, if the resource class is named `baremetal-small`, associate the flavor with this custom resource class via:

```
$ openstack flavor set --property resources:CUSTOM_BAREMETAL_SMALL=1 my-
↪baremetal-flavor
```

Another set of flavor properties must be used to disable scheduling based on standard properties for a bare metal flavor:

```
$ openstack flavor set --property resources:VCPU=0 my-baremetal-flavor
$ openstack flavor set --property resources:MEMORY_MB=0 my-baremetal-flavor
$ openstack flavor set --property resources:DISK_GB=0 my-baremetal-flavor
```

Example

If you want to define a class of nodes called `baremetal.with-GPU`, start with tagging some nodes with it:

```
$ baremetal node set <node> --resource-class baremetal.with-GPU
```

> **Warning:** It is possible to **add** a resource class to `active` nodes, but it is not possible to **replace** an existing resource class on them.

Then you can update your flavor to request the resource class instead of the standard properties:

```
$ openstack flavor set --property resources:CUSTOM_BAREMETAL_WITH_GPU=1 my-
↪baremetal-flavor
$ openstack flavor set --property resources:VCPU=0 my-baremetal-flavor
$ openstack flavor set --property resources:MEMORY_MB=0 my-baremetal-flavor
$ openstack flavor set --property resources:DISK_GB=0 my-baremetal-flavor
```

Note how `baremetal.with-GPU` in the nodes `resource_class` field becomes `CUSTOM_BAREMETAL_WITH_GPU` in the flavors properties.

Scheduling based on traits

Starting with the Queens release, the Compute service supports scheduling based on qualitative attributes using traits. Starting with Bare Metal REST API version 1.37, it is possible to assign a list of traits to each bare metal node. Traits assigned to a bare metal node will be assigned to the corresponding resource provider in the Compute service placement API.

When creating a flavor in the Compute service, required traits may be specified via flavor properties. The Compute service will then schedule instances only to bare metal nodes with all of the required traits.

Traits can be either standard or custom. Standard traits are listed in the os_traits library. Custom traits must meet the following requirements:

- prefixed with `CUSTOM_`
- contain only upper case characters A to Z, digits 0 to 9, or underscores
- no longer than 255 characters in length

A bare metal node can have a maximum of 50 traits.

Example

To add the standard trait `HW_CPU_X86_VMX` and a custom trait `CUSTOM_TRAIT1` to a node:

```
$ baremetal node add trait <node> CUSTOM_TRAIT1 HW_CPU_X86_VMX
```

Then, update the flavor to require these traits:

```
$ openstack flavor set --property trait:CUSTOM_TRAIT1=required my-
↪baremetal-flavor
$ openstack flavor set --property trait:HW_CPU_X86_VMX=required my-
↪baremetal-flavor
```

2.1.7 Set up the drivers for the Bare Metal service

Enabling drivers and hardware types

Introduction

The Bare Metal service delegates actual hardware management to **drivers**. *Drivers*, also called *hardware types*, consist of *hardware interfaces*: sets of functionality dealing with some aspect of bare metal provisioning in a vendor-specific way. There are generic **hardware types** (eg. `redfish` and `ipmi`), and vendor-specific ones (eg. `ilo` and `irmc`).

Note: Starting with the Rocky release, the terminologies *driver*, *dynamic driver*, and *hardware type* have the same meaning in the scope of Bare Metal service.

Enabling hardware types

Hardware types are enabled in the configuration file of the **ironic-conductor** service by setting the `enabled_hardware_types` configuration option, for example:

```
[DEFAULT]
enabled_hardware_types = ipmi,redfish
```

Due to the drivers dynamic nature, they also require configuring enabled hardware interfaces.

Note: All available hardware types and interfaces are listed in setup.cfg file in the source code tree.

Enabling hardware interfaces

There are several types of hardware interfaces:

bios manages configuration of the BIOS settings of a bare metal node. This interface is vendor-specific and can be enabled via the `enabled_bios_interfaces` option:

```
[DEFAULT]
enabled_hardware_types = <hardware_type_name>
enabled_bios_interfaces = <bios_interface_name>
```

See *BIOS Configuration* for details.

boot manages booting of both the deploy ramdisk and the user instances on the bare metal node. See *Boot interfaces* for details.

Boot interface implementations are often vendor specific, and can be enabled via the `enabled_boot_interfaces` option:

```
[DEFAULT]
enabled_hardware_types = ipmi,ilo
enabled_boot_interfaces = pxe,ilo-virtual-media
```

Boot interfaces with `pxe` in their name require *Configuring PXE and iPXE*. There are also a few hardware-specific boot interfaces - see *Drivers, Hardware Types and Hardware Interfaces* for their required configuration.

console manages access to the serial console of a bare metal node. See *Configuring Web or Serial Console* for details.

deploy defines how the image gets transferred to the target disk. See *Deploy Interfaces* for an explanation of the difference between supported deploy interfaces `direct` and `iscsi`.

The deploy interfaces can be enabled as follows:

```
[DEFAULT]
enabled_hardware_types = ipmi,redfish
enabled_deploy_interfaces = iscsi,direct
```

Additionally,

- the `iscsi` deploy interface requires *Configuring iSCSI-based drivers*
- the `direct` deploy interface requires the Object Storage service or an HTTP service

inspect implements fetching hardware information from nodes. Can be implemented out-of-band (via contacting the nodes BMC) or in-band (via booting a ramdisk on a node). The latter implementation is called `inspector` and uses a separate service called ironic-inspector. Example:

```
[DEFAULT]
enabled_hardware_types = ipmi,ilo,irmc
enabled_inspect_interfaces = ilo,irmc,inspector
```

See *Hardware Inspection* for more details.

management provides additional hardware management actions, like getting or setting boot devices. This interface is usually vendor-specific, and its name often matches the name of the hardware type (with `ipmitool` being a notable exception). For example:

```
[DEFAULT]
enabled_hardware_types = ipmi,redfish,ilo,irmc
enabled_management_interfaces = ipmitool,redfish,ilo,irmc
```

Using `ipmitool` requires *Configuring IPMI support*. See *Drivers, Hardware Types and Hardware Interfaces* for the required configuration of each driver.

network connects/disconnects bare metal nodes to/from virtual networks. See *Configure tenant networks* for more details.

power runs power actions on nodes. Similar to the management interface, it is usually vendor-specific, and its name often matches the name of the hardware type (with `ipmitool` being again an exception). For example:

```
[DEFAULT]
enabled_hardware_types = ipmi,redfish,ilo,irmc
enabled_power_interfaces = ipmitool,redfish,ilo,irmc
```

Using `ipmitool` requires *Configuring IPMI support*. See *Drivers, Hardware Types and Hardware Interfaces* for the required configuration of each driver.

raid manages building and tearing down RAID on nodes. Similar to inspection, it can be implemented either out-of-band or in-band (via `agent` implementation). See *RAID Configuration* for details. For example:

```
[DEFAULT]
enabled_hardware_types = ipmi,redfish,ilo,irmc
enabled_raid_interfaces = agent,no-raid
```

storage manages the interaction with a remote storage subsystem, such as the Block Storage service, and helps facilitate booting from a remote volume. This interface ensures that volume target and connector information is updated during the lifetime of a deployed instance. See *Boot From Volume* for more details.

This interface defaults to a `noop` driver as it is considered an opt-in interface which requires additional configuration by the operator to be usable.

For example:

```
[DEFAULT]
enabled_hardware_types = ipmi,irmc
enabled_storage_interfaces = cinder,noop
```

vendor is a place for vendor extensions to be exposed in API. See *Vendor Methods* for details.

```
[DEFAULT]
enabled_hardware_types = ipmi,redfish,ilo,irmc
enabled_vendor_interfaces = ipmitool,no-vendor
```

Here is a complete configuration example, enabling two generic protocols, IPMI and Redfish, with a few additional features:

```
[DEFAULT]
enabled_hardware_types = ipmi,redfish
enabled_boot_interfaces = pxe
enabled_console_interfaces = ipmitool-socat,no-console
```

(continues on next page)

(continued from previous page)

```
enabled_deploy_interfaces = iscsi,direct
enabled_inspect_interfaces = inspector
enabled_management_interfaces = ipmitool,redfish
enabled_network_interfaces = flat,neutron
enabled_power_interfaces = ipmitool,redfish
enabled_raid_interfaces = agent
enabled_storage_interfaces = cinder,noop
enabled_vendor_interfaces = ipmitool,no-vendor
```

Note that some interfaces have implementations named `no-<TYPE>` where `<TYPE>` is the interface type. These implementations do nothing and return errors when used from API.

Hardware interfaces in multi-conductor environments

When enabling hardware types and their interfaces, make sure that for every enabled hardware type, the whole set of enabled interfaces matches for all conductors. However, different conductors can have different hardware types enabled.

For example, you can have two conductors with the following configuration respectively:

```
[DEFAULT]
enabled_hardware_types = ipmi
enabled_deploy_interfaces = direct
enabled_power_interfaces = ipmitool
enabled_management_interfaces = ipmitool
```

```
[DEFAULT]
enabled_hardware_types = redfish
enabled_deploy_interfaces = iscsi
enabled_power_interfaces = redfish
enabled_management_interfaces = redfish
```

But you cannot have two conductors with the following configuration respectively:

```
[DEFAULT]
enabled_hardware_types = ipmi,redfish
enabled_deploy_interfaces = direct
enabled_power_interfaces = ipmitool,redfish
enabled_management_interfaces = ipmitool,redfish
```

```
[DEFAULT]
enabled_hardware_types = redfish
enabled_deploy_interfaces = iscsi
enabled_power_interfaces = redfish
enabled_management_interfaces = redfish
```

This is because the `redfish` hardware type will have different enabled *deploy* interfaces on these conductors. It would have been fine, if the second conductor had `enabled_deploy_interfaces = direct` instead of `iscsi`.

This situation is not detected by the Bare Metal service, but it can cause inconsistent behavior in the API, when node functionality will depend on which conductor it gets assigned to.

Note: We dont treat this as an error, because such *temporary* inconsistency is inevitable during a rolling upgrade or a configuration update.

Configuring interface defaults

When an operator does not provide an explicit value for one of the interfaces (when creating a node or updating its driver), the default value is calculated as described in *Defaults for hardware interfaces*. It is also possible to override the defaults for any interfaces by setting one of the options named `default_<IFACE>_interface`, where `<IFACE>` is the interface name. For example:

```
[DEFAULT]
default_deploy_interface = direct
default_network_interface = neutron
```

This configuration forces the default *deploy* interface to be `direct` and the default *network* interface to be `neutron` for all hardware types.

The defaults are calculated and set on a node when creating it or updating its hardware type. Thus, changing these configuration options has no effect on existing nodes.

Warning: The default interface implementation must be configured the same way across all conductors in the cloud, except maybe for a short period of time during an upgrade or configuration update. Otherwise the default implementation will depend on which conductor handles which node, and this mapping is not predictable or even persistent.

Warning: These options should be used with care. If a hardware type does not support the provided default implementation, its users will have to always provide an explicit value for this interface when creating a node.

Configuring PXE and iPXE

DHCP server setup

A DHCP server is required by PXE/iPXE client. You need to follow steps below.

1. Set the `[dhcp]/dhcp_provider` to `neutron` in the Bare Metal Services configuration file (`/etc/ironic/ironic.conf`):

   Note: Refer *Configure tenant networks* for details. The `dhcp_provider` configuration is already set by the configuration defaults, and when you create subnet, DHCP is also enabled if you do not add any dhcp options at openstack subnet create command.

2. Enable DHCP in the subnet of PXE network.
3. Set the ip address range in the subnet for DHCP.

Note: Refer *Configure the Networking service for bare metal provisioning* for details about the two precedent steps.

4. Connect the openstack DHCP agent to the external network through the OVS bridges and the interface `eth2`.

   Note: Refer *Configure the Networking service for bare metal provisioning* for details. You do not require this part if br-int, br-eth2 and eth2 are already connected.

5. Configure the host ip at `br-eth2`. If it locates at `eth2`, do below:

```
ip addr del 192.168.2.10/24 dev eth2
ip addr add 192.168.2.10/24 dev br-eth2
```

   Note: Replace eth2 with the interface on the network node which you are using to connect to the Bare Metal service.

TFTP server setup

In order to deploy instances via PXE, a TFTP server needs to be set up on the Bare Metal service nodes which run the `ironic-conductor`.

1. Make sure the tftp root directory exist and can be written to by the user the `ironic-conductor` is running as. For example:

```
sudo mkdir -p /tftpboot
sudo chown -R ironic /tftpboot
```

2. Install tftp server:

   Ubuntu:

```
sudo apt-get install xinetd tftpd-hpa
```

   RHEL8/CentOS8/Fedora:

```
sudo dnf install tftp-server xinetd
```

   SUSE:

```
sudo zypper install tftp xinetd
```

3. Using xinetd to provide a tftp server setup to serve `/tftpboot`. Create or edit `/etc/xinetd.d/tftp` as below:

```
service tftp
{
  protocol        = udp
  port            = 69
```

(continues on next page)

(continued from previous page)

```
    socket_type     = dgram
    wait            = yes
    user            = root
    server          = /usr/sbin/in.tftpd
    server_args     = -v -v -v -v -v --map-file /tftpboot/map-file /
↪tftpboot
    disable         = no
    # This is a workaround for Fedora, where TFTP will listen only on
    # IPv6 endpoint, if IPv4 flag is not used.
    flags           = IPv4
}
```

and restart the `xinetd` service:

Ubuntu:

```
sudo service xinetd restart
```

Fedora/RHEL8/CentOS8/SUSE:

```
sudo systemctl restart xinetd
```

Note: In certain environments the networks MTU may cause TFTP UDP packets to get fragmented. Certain PXE firmwares struggle to reconstruct the fragmented packets which can cause significant slow down or even prevent the server from PXE booting. In order to avoid this, TFTPd provides an option to limit the packet size so that it they do not get fragmented. To set this additional option in the server_args above:

```
--blocksize <MAX MTU minus 32>
```

---

4. Create a map file in the tftp boot directory (`/tftpboot`):

```
echo 're ^(/tftpboot/) /tftpboot/\2' > /tftpboot/map-file
echo 're ^/tftpboot/ /tftpboot/' >> /tftpboot/map-file
echo 're ^(^/) /tftpboot/\1' >> /tftpboot/map-file
echo 're ^([^/]) /tftpboot/\1' >> /tftpboot/map-file
```

UEFI PXE - Grub setup

In order to deploy instances with PXE on bare metal nodes which support UEFI, perform these additional steps on the ironic conductor node to configure the PXE UEFI environment.

1. Install Grub2 and shim packages:

   Ubuntu (18.04LTS and later):

```
sudo apt-get install grub-efi-amd64-signed shim-signed
```

   RHEL8/CentOS8/Fedora:

```
sudo dnf install grub2-efi shim
```

SUSE:

```
sudo zypper install grub2-x86_64-efi shim
```

2. Copy grub and shim boot loader images to `/tftpboot` directory:

   Ubuntu (18.04LTS and later):

   ```
   sudo cp /usr/lib/shim/shimx64.efi.signed /tftpboot/bootx64.efi
   sudo cp /usr/lib/grub/x86_64-efi-signed/grubnetx64.efi.signed /
   ↪tftpboot/grubx64.efi
   ```

   Fedora:

   ```
   sudo cp /boot/efi/EFI/fedora/shim.efi /tftpboot/bootx64.efi
   sudo cp /boot/efi/EFI/fedora/grubx64.efi /tftpboot/grubx64.efi
   ```

   RHEL8/CentOS8:

   ```
   sudo cp /boot/efi/EFI/centos/shim.efi /tftpboot/bootx64.efi
   sudo cp /boot/efi/EFI/centos/grubx64.efi /tftpboot/grubx64.efi
   ```

   SUSE:

   ```
   sudo cp /usr/lib64/efi/shim.efi /tftpboot/bootx64.efi
   sudo cp /usr/lib/grub2/x86_64-efi/grub.efi /tftpboot/grubx64.efi
   ```

3. Create master grub.cfg:

   Ubuntu: Create grub.cfg under `/tftpboot/grub` directory:

   ```
   GRUB_DIR=/tftpboot/grub
   ```

   Fedora: Create grub.cfg under `/tftpboot/EFI/fedora` directory:

   ```
   GRUB_DIR=/tftpboot/EFI/fedora
   ```

   RHEL8/CentOS8: Create grub.cfg under `/tftpboot/EFI/centos` directory:

   ```
   GRUB_DIR=/tftpboot/EFI/centos
   ```

   SUSE: Create grub.cfg under `/tftpboot/boot/grub` directory:

   ```
   GRUB_DIR=/tftpboot/boot/grub
   ```

   Create directory `GRUB_DIR`:

   ```
   sudo mkdir -p $GRUB_DIR
   ```

   This file is used to redirect grub to baremetal node specific config file. It redirects it to specific grub config file based on DHCP IP assigned to baremetal node.

   ```
   set default=master
   set timeout=5
   set hidden_timeout_quiet=false

   ```

(continues on next page)

(continued from previous page)

```
menuentry "master"  {
configfile /tftpboot/$net_default_mac.conf
}
```

Change the permission of grub.cfg:

```
sudo chmod 644 $GRUB_DIR/grub.cfg
```

4. Update the bare metal node with `boot_mode:uefi` capability in nodes properties field. See *Boot mode support* for details.
5. Make sure that bare metal node is configured to boot in UEFI boot mode and boot device is set to network/pxe.

Note: Some drivers, e.g. `ilo`, `irmc` and `redfish`, support automatic setting of the boot mode during deployment. This step is not required for them. Please check *Drivers, Hardware Types and Hardware Interfaces* for information on whether your driver requires manual UEFI configuration.

Legacy BIOS - Syslinux setup

In order to deploy instances with PXE on bare metal using Legacy BIOS boot mode, perform these additional steps on the ironic conductor node.

1. Install the syslinux package with the PXE boot images:

   Ubuntu (16.04LTS and later):

   ```
   sudo apt-get install syslinux-common pxelinux
   ```

   RHEL8/CentOS8/Fedora:

   ```
   sudo dnf install syslinux-tftpboot
   ```

   SUSE:

   ```
   sudo zypper install syslinux
   ```

2. Copy the PXE image to `/tftpboot`. The PXE image might be found at[1]:

   Ubuntu (16.04LTS and later):

   ```
   sudo cp /usr/lib/PXELINUX/pxelinux.0 /tftpboot
   ```

   RHEL8/CentOS8/SUSE:

   ```
   sudo cp /usr/share/syslinux/pxelinux.0 /tftpboot
   ```

3. If whole disk images need to be deployed via PXE-netboot, copy the chain.c32 image to `/tftpboot` to support it:

---

[1] On **Fedora/RHEL** the `syslinux-tftpboot` package already installs the library modules and PXE image at `/tftpboot`. If the TFTP server is configured to listen to a different directory you should copy the contents of `/tftpboot` to the configured directory

Ubuntu (16.04LTS and later):

```
sudo cp /usr/lib/syslinux/modules/bios/chain.c32 /tftpboot
```

Fedora:

```
sudo cp /boot/extlinux/chain.c32 /tftpboot
```

RHEL8/CentOS8/SUSE:

```
sudo cp /usr/share/syslinux/chain.c32 /tftpboot/
```

4. If the version of syslinux is **greater than** 4 we also need to make sure that we copy the library modules into the `/tftpboot` directory[2]?. For example, for Ubuntu run:

```
sudo cp /usr/lib/syslinux/modules/*/ldlinux.* /tftpboot
```

5. Update the bare metal node with `boot_mode:bios` capability in nodes properties field. See *Boot mode support* for details.

6. Make sure that bare metal node is configured to boot in Legacy BIOS boot mode and boot device is set to network/pxe.

iPXE setup

If you will be using iPXE to boot instead of PXE, iPXE needs to be set up on the Bare Metal service node(s) where `ironic-conductor` is running.

1. Make sure these directories exist and can be written to by the user the `ironic-conductor` is running as. For example:

```
sudo mkdir -p /tftpboot
sudo mkdir -p /httpboot
sudo chown -R ironic /tftpboot
sudo chown -R ironic /httpboot
```

2. Create a map file in the tftp boot directory (`/tftpboot`):

```
echo 'r ^([^/]) /tftpboot/\1' > /tftpboot/map-file
echo 'r ^(/tftpboot/) /tftpboot/\2' >> /tftpboot/map-file
```

3. Set up TFTP and HTTP servers.

   These servers should be running and configured to use the local /tftpboot and /httpboot directories respectively, as their root directories. (Setting up these servers is outside the scope of this install guide.)

   These root directories need to be mounted locally to the `ironic-conductor` services, so that the services can access them.

   The Bare Metal services configuration file (/etc/ironic/ironic.conf) should be edited accordingly to specify the TFTP and HTTP root directories and server addresses. For example:

---

[2] http://www.syslinux.org/wiki/index.php/Library_modules

```
[pxe]

# Ironic compute node's tftp root path. (string value)
tftp_root=/tftpboot

# IP address of Ironic compute node's tftp server. (string
# value)
tftp_server=192.168.0.2

[deploy]
# Ironic compute node's http root path. (string value)
http_root=/httpboot

# Ironic compute node's HTTP server URL. Example:
# http://192.1.2.3:8080 (string value)
http_url=http://192.168.0.2:8080
```

See also: *Deploying outside of the provisioning network*.

4. Install the iPXE package with the boot images:

   Ubuntu:

   ```
   apt-get install ipxe
   ```

   RHEL8/CentOS8/Fedora:

   ```
   dnf install ipxe-bootimgs
   ```

   Note: SUSE does not provide a package containing iPXE boot images. If you are using SUSE or if the packaged version of the iPXE boot image doesnt work, you can download a prebuilt one from http://boot.ipxe.org or build one image from source, see http://ipxe.org/download for more information.

5. Copy the iPXE boot image (`undionly.kpxe` for **BIOS** and `ipxe.efi` for **UEFI**) to `/tftpboot`. The binary might be found at:

   Ubuntu:

   ```
   cp /usr/lib/ipxe/{undionly.kpxe,ipxe.efi,snponly.efi} /tftpboot
   ```

   Fedora/RHEL8/CentOS8:

   ```
   cp /usr/share/ipxe/{undionly.kpxe,ipxe-x86_64.efi,ipxe-snponly-x86_64.
   ↪efi} /tftpboot
   ```

   Note: `snponly` variants may not be available for all distributions.

6. Enable/Configure iPXE overrides in the Bare Metal Services configuration file **if required** (/etc/ironic/ironic.conf):

```
[pxe]

# Neutron bootfile DHCP parameter. (string value)
ipxe_bootfile_name=undionly.kpxe

# Bootfile DHCP parameter for UEFI boot mode. (string value)
uefi_ipxe_bootfile_name=ipxe.efi

# Template file for PXE configuration. (string value)
ipxe_config_template=$pybasedir/drivers/modules/ipxe_config.template
```

Note: Most UEFI systems have integrated networking which means the `[pxe]uefi_ipxe_bootfile_name` setting should be set to `snponly.efi` or `ipxe-snponly-x86_64.efi` if its available for your distribution.

Note: Setting the iPXE parameters noted in the code block above to no value, in other words setting a line to something like `ipxe_bootfile_name=` will result in ironic falling back to the default values of the non-iPXE PXE settings. This is for backwards compatability.

7. Ensure iPXE is the default PXE, if applicable.

   In earlier versions of ironic, a `[pxe]ipxe_enabled` setting allowing operators to declare the behavior of the conductor to exclusively operate as if only iPXE was to be used. As time moved on, iPXE functionality was moved to its own `ipxe` boot interface.

   If you want to emulate that same hehavior, set the following in the configuration file (/etc/ironic/ironic.conf):

   ```
   [DEFAULT]
   default_boot_interface=ipxe
   enabled_boot_interfaces=ipxe,pxe
   ```

   Note: The `[DEFAULT]enabled_boot_interfaces` setting may be exclusively set to `ipxe`, however ironic has multiple interfaces available depending on the hardware types available for use.

8. It is possible to configure the Bare Metal service in such a way that nodes will boot into the deploy image directly from Object Storage. Doing this avoids having to cache the images on the ironic-conductor host and serving them via the ironic-conductors *HTTP server*. This can be done if:

   1. the Image Service is used for image storage;
   2. the images in the Image Service are internally stored in Object Storage;
   3. the Object Storage supports generating temporary URLs for accessing objects stored in it. Both the OpenStack Swift and RADOS Gateway provide support for this.
      - See *Ceph Object Gateway support* on how to configure the Bare Metal Service with RADOS Gateway as the Object Storage.

Configure this by setting the `[pxe]/ipxe_use_swift` configuration option to `True` as follows:

```
[pxe]

# Download deploy images directly from swift using temporary
# URLs. If set to false (default), images are downloaded to
# the ironic-conductor node and served over its local HTTP
# server. Applicable only when 'ipxe_enabled' option is set to
# true. (boolean value)
ipxe_use_swift=True
```

Although the *HTTP server* still has to be deployed and configured (as it will serve iPXE boot script and boot configuration files for nodes), such configuration will shift some load from ironic-conductor hosts to the Object Storage service which can be scaled horizontally.

Note that when SSL is enabled on the Object Storage service you have to ensure that iPXE firmware on the nodes can indeed boot from generated temporary URLs that use HTTPS protocol.

9. Restart the `ironic-conductor` process:

   Fedora/RHEL8/CentOS8/SUSE:

   ```
   sudo systemctl restart openstack-ironic-conductor
   ```

   Ubuntu:

   ```
   sudo service ironic-conductor restart
   ```

PXE multi-architecture setup

It is possible to deploy servers of different architecture by one conductor. To use this feature, architecture-specific boot and template files must be configured using the configuration options `[pxe]pxe_bootfile_name_by_arch` and `[pxe]pxe_config_template_by_arch` respectively, in the Bare Metal services configuration file (/etc/ironic/ironic.conf).

These two options are dictionary values; the key is the architecture and the value is the boot (or config template) file. A nodes `cpu_arch` property is used as the key to get the appropriate boot file and template file. If the nodes `cpu_arch` is not in the dictionary, the configuration options (in [pxe] group) `pxe_bootfile_name`, `pxe_config_template`, `uefi_pxe_bootfile_name` and `uefi_pxe_config_template` will be used instead.

In the following example, since x86 and x86_64 keys are not in the `pxe_bootfile_name_by_arch` or `pxe_config_template_by_arch` options, x86 and x86_64 nodes will be deployed by pxelinux.0 or bootx64.efi, depending on the nodes `boot_mode` capability (bios or uefi). However, aarch64 nodes will be deployed by grubaa64.efi, and ppc64 nodes by bootppc64:

```
[pxe]

# Bootfile DHCP parameter. (string value)
pxe_bootfile_name=pxelinux.0

# On ironic-conductor node, template file for PXE
```

(continues on next page)

(continued from previous page)

```
# configuration. (string value)
pxe_config_template = $pybasedir/drivers/modules/pxe_config.template

# Bootfile DHCP parameter for UEFI boot mode. (string value)
uefi_pxe_bootfile_name=bootx64.efi

# On ironic-conductor node, template file for PXE
# configuration for UEFI boot loader. (string value)
uefi_pxe_config_template=$pybasedir/drivers/modules/pxe_grub_config.
↪template

# Bootfile DHCP parameter per node architecture. (dict value)
pxe_bootfile_name_by_arch=aarch64:grubaa64.efi,ppc64:bootppc64

# On ironic-conductor node, template file for PXE
# configuration per node architecture. For example:
# aarch64:/opt/share/grubaa64_pxe_config.template (dict value)
pxe_config_template_by_arch=aarch64:pxe_grubaa64_config.template,ppc64:pxe_
↪ppc64_config.template
```

Note: The grub implementation may vary on different architecture, you may need to tweak the pxe config template for a specific arch. For example, grubaa64.efi shipped with CentoOS7 does not support `linuxefi` and `initrdefi` commands, youll need to switch to use `linux` and `initrd` command instead.

Note: A `[pxe]ipxe_bootfile_name_by_arch` setting is available for multi-arch iPXE based deployment, and defaults to the same behavior as the comperable `[pxe]pxe_bootfile_by_arch` setting for standard PXE.

PXE timeouts tuning

Because of its reliance on UDP-based protocols (DHCP and TFTP), PXE is particularly vulnerable to random failures during the booting stage. If the deployment ramdisk never calls back to the bare metal conductor, the build will be aborted, and the node will be moved to the `deploy failed` state, after the deploy callback timeout. This timeout can be changed via the *`conductor.deploy_callback_timeout`* configuration option.

Starting with the Train release, the Bare Metal service can retry PXE boot if it takes too long. The timeout is defined via *`pxe.boot_retry_timeout`* and must be smaller than the `deploy_callback_timeout`, otherwise it will have no effect.

For example, the following configuration sets the overall timeout to 60 minutes, allowing two retries after 20 minutes:

```
[conductor]
deploy_callback_timeout = 3600

[pxe]
boot_retry_timeout = 1200
```

Configuring IPMI support

Installing ipmitool command

To enable one of the drivers that use IPMI protocol for power and management actions (for example, `ipmi`), the `ipmitool` command must be present on the service node(s) where `ironic-conductor` is running. On most distros, it is provided as part of the `ipmitool` package. Source code is available at http://ipmitool.sourceforge.net/.

> **Warning:** Certain distros, notably Mac OS X and SLES, install `openipmi` instead of `ipmitool` by default. This driver is not compatible with `openipmi` as it relies on error handling options not provided by this tool.

Please refer to the *IPMI driver* for information on how to use IPMItool-based drivers.

Validation and troubleshooting

Check that you can connect to, and authenticate with, the IPMI controller in your bare metal server by running `ipmitool`:

```
ipmitool -I lanplus -H <ip-address> -U <username> -P <password> chassis␣
↪power status
```

where `<ip-address>` is the IP of the IPMI controller you want to access. This is not the bare metal nodes main IP. The IPMI controller should have its own unique IP.

If the above command doesnt return the power status of the bare metal server, check that

- `ipmitool` is installed and is available via the `$PATH` environment variable.
- The IPMI controller on your bare metal server is turned on.
- The IPMI controller credentials and IP address passed in the command are correct.
- The conductor node has a route to the IPMI controller. This can be checked by just pinging the IPMI controller IP from the conductor node.

IPMI configuration

If there are slow or unresponsive BMCs in the environment, the `min_command_interval` configuration option in the `[ipmi]` section may need to be raised. The default is fairly conservative, as setting this timeout too low can cause older BMCs to crash and require a hard-reset.

Collecting sensor data

Bare Metal service supports sending IPMI sensor data to Telemetry with certain hardware types, such as `ipmi`, `ilo` and `irmc`. By default, support for sending IPMI sensor data to Telemetry is disabled. If you want to enable it, you should make the following two changes in `ironic.conf`:

```
[conductor]
send_sensor_data = true
[oslo_messaging_notifications]
driver = messagingv2
```

If you want to customize the sensor types which will be sent to Telemetry, change the `send_sensor_data_types` option. For example, the below settings will send information about temperature, fan, voltage from sensors to the Telemetry service:

```
send_sensor_data_types=Temperature,Fan,Voltage
```

Supported sensor types are defined by the Telemetry service, currently these are `Temperature`, `Fan`, `Voltage`, `Current`. Special value `All` (the default) designates all supported sensor types.

Configuring iSCSI-based drivers

Ensure that the `qemu-img` and `iscsiadm` tools are installed on the **ironic-conductor** host(s).

2.1.8 Enrollment

After all the services have been properly configured, you should enroll your hardware with the Bare Metal service, and confirm that the Compute service sees the available hardware. The nodes will be visible to the Compute service once they are in the `available` provision state.

Note: After enrolling nodes with the Bare Metal service, the Compute service will not be immediately notified of the new resources. The Compute services resource tracker syncs periodically, and so any changes made directly to the Bare Metal services resources will become visible in the Compute service only after the next run of that periodic task. More information is in the *Troubleshooting* section.

Note: Any bare metal node that is visible to the Compute service may have a workload scheduled to it, if both the `power` and `management` interfaces pass the `validate` check. If you wish to exclude a node from the Compute services scheduler, for instance so that you can perform maintenance on it, you can set the node to maintenance mode. For more information see the *Maintenance mode* section.

Choosing a driver

When enrolling a node, the most important information to supply is *driver*. See *Enabling drivers and hardware types* for a detailed explanation of bare metal drivers, hardware types and interfaces. The `driver list` command can be used to list all drivers enabled on all hosts:

```
baremetal driver list
+---------------------+-----------------------+
| Supported driver(s) | Active host(s)        |
+---------------------+-----------------------+
| ipmi                | localhost.localdomain |
+---------------------+-----------------------+
```

The specific driver to use should be picked based on actual hardware capabilities and expected features. See *Drivers, Hardware Types and Hardware Interfaces* for more hints on that.

Each driver has a list of *driver properties* that need to be specified via the nodes `driver_info` field, in order for the driver to operate on node. This list consists of the properties of the hardware interfaces that the driver uses. These driver properties are available with the `driver property list` command:

```
$ baremetal driver property list ipmi
+----------------------+---------------------------------------------------
↪----------------------------------------------------------+
| Property             | Description                                       ␣
↪                                                          |
+----------------------+---------------------------------------------------
↪----------------------------------------------------------+
| ipmi_address         | IP address or hostname of the node. Required.     ␣
↪                                                          |
| ipmi_password        | password. Optional.                               ␣
↪                                                          |
| ipmi_username        | username; default is NULL user. Optional.         ␣
↪                                                          |
| ...                  | ...                                               ␣
↪                                                          |
| deploy_kernel        | UUID (from Glance) of the deployment kernel.␣
↪Required.                                                         |
| deploy_ramdisk       | UUID (from Glance) of the ramdisk that is mounted␣
↪at boot time. Required.                                   |
+----------------------+---------------------------------------------------
↪----------------------------------------------------------+
```

The properties marked as required must be supplied either during node creation or shortly after. Some properties may only be required for certain features.

Note on API versions

Starting with API version 1.11, the Bare Metal service added a new initial provision state of `enroll` to its state machine. When this or later API version is used, new nodes get this state instead of `available`.

Existing automation tooling that use an API version lower than 1.11 are not affected, since the initial provision state is still `available`. However, using API version 1.11 or above may break existing automation tooling with respect to node creation.

The default API version used by (the most recent) python-ironicclient is 1.9, but it may change in the future and should not be relied on.

In the examples below we will use version 1.11 of the Bare metal API. This gives us the following advantages:

- Explicit power credentials validation before leaving the `enroll` state.
- Running node cleaning before entering the `available` state.
- Not exposing half-configured nodes to the scheduler.

To set the API version for all commands, you can set the environment variable `IRONIC_API_VERSION`. For the OpenStackClient baremetal plugin, set the `OS_BAREMETAL_API_VERSION` variable to the same value. For example:

```
$ export IRONIC_API_VERSION=1.11
$ export OS_BAREMETAL_API_VERSION=1.11
```

Enrollment process

Creating a node

This section describes the main steps to enroll a node and make it available for provisioning. Some steps are shown separately for illustration purposes, and may be combined if desired.

1. Create a node in the Bare Metal service with the `node create` command. At a minimum, you must specify the driver name (for example, `ipmi`).

   This command returns the node UUID along with other information about the node. The nodes provision state will be `enroll`:

```
$ export OS_BAREMETAL_API_VERSION=1.11
$ baremetal node create --driver ipmi
+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| uuid         | dfc6189f-ad83-4261-9bda-b27258eb1987 |
| driver_info  | {}                                   |
| extra        | {}                                   |
| driver       | ipmi                                 |
| chassis_uuid |                                      |
| properties   | {}                                   |
| name         | None                                 |
+--------------+--------------------------------------+

$ baremetal node show dfc6189f-ad83-4261-9bda-b27258eb1987
+------------------------+--------------------------------------+
| Property               | Value                                |
+------------------------+--------------------------------------+
| target_power_state     | None                                 |
| extra                  | {}                                   |
| last_error             | None                                 |
| maintenance_reason     | None                                 |
| provision_state        | enroll                               |
| uuid                   | dfc6189f-ad83-4261-9bda-b27258eb1987 |
```

(continues on next page)

(continued from previous page)

```
| console_enabled        | False                                |
| target_provision_state | None                                 |
| provision_updated_at   | None                                 |
| maintenance            | False                                |
| power_state            | None                                 |
| driver                 | ipmi                                 |
| properties             | {}                                   |
| instance_uuid          | None                                 |
| name                   | None                                 |
| driver_info            | {}                                   |
| ...                    | ...                                  |
+------------------------+--------------------------------------+
```

A node may also be referred to by a logical name as well as its UUID. A name can be assigned to the node during its creation by adding the `-n` option to the `node create` command or by updating an existing node with the `node set` command. See *Logical Names* for examples.

2. Starting with API version 1.31 (and `python-ironicclient` 1.13), you can pick which hardware interface to use with nodes that use hardware types. Each interface is represented by a node field called `<IFACE>_interface` where `<IFACE>` in the interface type, e.g. `boot`. See *Enabling drivers and hardware types* for details on hardware interfaces.

   An interface can be set either separately:

```
$ baremetal node set $NODE_UUID --deploy-interface direct --raid-
↪interface agent
```

   or set during node creation:

```
$ baremetal node create --driver ipmi \
    --deploy-interface direct \
    --raid-interface agent
```

   If no value is provided for some interfaces, *Defaults for hardware interfaces* are used instead.

3. Update the node `driver_info` with the required driver properties, so that the Bare Metal service can manage the node:

```
$ baremetal node set $NODE_UUID \
    --driver-info ipmi_username=$USER \
    --driver-info ipmi_password=$PASS \
    --driver-info ipmi_address=$ADDRESS
```

Note: If IPMI is running on a port other than 623 (the default). The port must be added to `driver_info` by specifying the `ipmi_port` value. Example:

```
$ baremetal node set $NODE_UUID --driver-info ipmi_port=$PORT_NUMBER
```

   You may also specify all `driver_info` parameters during node creation by passing the **driver-info** option multiple times:

```
$ baremetal node create --driver ipmi \
    --driver-info ipmi_username=$USER \
```

(continues on next page)

(continued from previous page)

```
    --driver-info ipmi_password=$PASS \
    --driver-info ipmi_address=$ADDRESS
```

See *Choosing a driver* above for details on driver properties.

4. Specify a deploy kernel and ramdisk compatible with the nodes driver, for example:

```
$ baremetal node set $NODE_UUID \
    --driver-info deploy_kernel=$DEPLOY_VMLINUZ_UUID \
    --driver-info deploy_ramdisk=$DEPLOY_INITRD_UUID
```

See *Add images to the Image service* for details.

5. Optionally you can specify the provisioning and/or cleaning network UUID or name in the nodes `driver_info`. The `neutron` network interface requires both `provisioning_network` and `cleaning_network`, while the `flat` network interface requires the `cleaning_network` to be set either in the configuration or on the nodes. For example:

```
$ baremetal node set $NODE_UUID \
    --driver-info cleaning_network=$CLEAN_UUID_OR_NAME \
    --driver-info provisioning_network=$PROVISION_UUID_OR_NAME
```

See *Configure tenant networks* for details.

6. You must also inform the Bare Metal service of the network interface cards which are part of the node by creating a port with each NICs MAC address. These MAC addresses are passed to the Networking service during instance provisioning and used to configure the network appropriately:

```
$ baremetal port create $MAC_ADDRESS --node $NODE_UUID
```

Note: When it is time to remove the node from the Bare Metal service, the command used to remove the port is `baremetal port delete <port uuid>`. When doing so, it is important to ensure that the baremetal node is not in `maintenance` as guarding logic to prevent orphaning Neutron Virtual Interfaces (VIFs) will be overriden.

Adding scheduling information

1. Assign a *resource class* to the node. A *resource class* should represent a class of hardware in your data center, that corresponds to a Compute flavor.

   For example, lets split hardware into these three groups:

   1. nodes with a lot of RAM and powerful CPU for computational tasks,
   2. nodes with powerful GPU for OpenCL computing,
   3. smaller nodes for development and testing.

   We can define three resource classes to reflect these hardware groups, named `large-cpu`, `large-gpu` and `small` respectively. Then, for each node in each of the hardware groups, well set their `resource_class` appropriately via:

```
$ baremetal node set $NODE_UUID --resource-class $CLASS_NAME
```

The `--resource-class` argument can also be used when creating a node:

```
$ baremetal node create --driver $DRIVER --resource-class $CLASS_NAME
```

To use resource classes for scheduling you need to update your flavors as described in *Create flavors for use with the Bare Metal service*.

Note: This is not required for standalone deployments, only for those using the Compute service for provisioning bare metal instances.

---

2. Update the nodes properties to match the actual hardware of the node:

```
$ baremetal node set $NODE_UUID \
    --property cpus=$CPU_COUNT \
    --property memory_mb=$RAM_MB \
    --property local_gb=$DISK_GB
```

As above, these can also be specified at node creation by passing the **property** option to `node create` multiple times:

```
$ baremetal node create --driver ipmi \
    --driver-info ipmi_username=$USER \
    --driver-info ipmi_password=$PASS \
    --driver-info ipmi_address=$ADDRESS \
    --property cpus=$CPU_COUNT \
    --property memory_mb=$RAM_MB \
    --property local_gb=$DISK_GB
```

These values can also be discovered during *Hardware Inspection*.

Warning: The value provided for the `local_gb` property must match the size of the root device youre going to deploy on. By default **ironic-python-agent** picks the smallest disk which is not smaller than 4 GiB.

If you override this logic by using root device hints (see *Specifying the disk for deployment (root device hints)*), the `local_gb` value should match the size of picked target disk.

3. If you wish to perform more advanced scheduling of the instances based on hardware capabilities, you may add metadata to each node that will be exposed to the Compute scheduler (see: ComputeCapabilitiesFilter). A full explanation of this is outside of the scope of this document. It can be done through the special `capabilities` member of node properties:

```
$ baremetal node set $NODE_UUID \
    --property capabilities=key1:val1,key2:val2
```

Some capabilities can also be discovered during *Hardware Inspection*.

4. If you wish to perform advanced scheduling of instances based on qualitative attributes of bare metal nodes, you may add traits to each bare metal node that will be exposed to the Compute scheduler (see: *Scheduling based on traits* for a more in-depth discussion of traits in the Bare

Metal service). For example, to add the standard trait `HW_CPU_X86_VMX` and a custom trait `CUSTOM_TRAIT1` to a node:

```
$ baremetal node add trait $NODE_UUID \
    CUSTOM_TRAIT1 HW_CPU_X86_VMX
```

Validating node information

1. To check if Bare Metal service has the minimum information necessary for a nodes driver to be functional, you may `validate` it:

```
$ baremetal node validate $NODE_UUID
+------------+--------+--------+
| Interface  | Result | Reason |
+------------+--------+--------+
| boot       | True   |        |
| console    | True   |        |
| deploy     | True   |        |
| inspect    | True   |        |
| management | True   |        |
| network    | True   |        |
| power      | True   |        |
| raid       | True   |        |
| storage    | True   |        |
+------------+--------+--------+
```

If the node fails validation, each driver interface will return information as to why it failed:

```
$ baremetal node validate $NODE_UUID
+------------+--------+-----------------------------------------------
↪--------------------------------------------------------------------
↪-----------------+
| Interface  | Result | Reason                                       ␣
↪                                                                  ␣
↪                 |
+------------+--------+-----------------------------------------------
↪--------------------------------------------------------------------
↪-----------------+
| boot       | True   |                                              ␣
↪                                                                  ␣
↪                 |
| console    | None   | not supported                                ␣
↪                                                                  ␣
↪                 |
| deploy     | False  | Cannot validate iSCSI deploy. Some parameters␣
↪were missing in node's instance_info. Missing are: ['root_gb',
↪'image_source']        |
| inspect    | True   |                                              ␣
↪                                                                  ␣
↪                 |
| management | False  | Missing the following IPMI credentials in node
↪'s driver_info: ['ipmi_address'].                                 ␣
↪                 |
| network    | True   |                                              ␣
↪                                                                  ␣
↪                 |
```

(continues on next page)

(continued from previous page)

```
| power       | False   | Missing the following IPMI credentials in node
↪'s driver_info: ['ipmi_address'].                                   ␣
↪                  |
| raid        | None    | not supported                              ␣
↪                                                                    ␣
↪                  |
| storage     | True    |                                            ␣
↪                                                                    ␣
↪                  |
+-------------+--------+----------------------------------------------------
↪----------------------------------------------------------------------
↪------------------+
```

When using the Compute Service with the Bare Metal service, it is safe to ignore the deploy interfaces validation error due to lack of image information. You may continue the enrollment process. This information will be set by the Compute Service just before deploying, when an instance is requested:

```
$ baremetal node validate $NODE_UUID
+-------------+--------+----------------------------------------------------
↪----------------------------------------------------------------------
↪-----------------------------------------------+
| Interface   | Result | Reason                                          ␣
↪                                                                    ␣
↪                                               |
+-------------+--------+----------------------------------------------------
↪----------------------------------------------------------------------
↪-----------------------------------------------+
| boot        | False   | Cannot validate image information for node␣
↪because one or more parameters are missing from its instance_info.␣
↪Missing are: ['ramdisk', 'kernel', 'image_source'] |
| console     | True    |                                            ␣
↪                                                                    ␣
↪                                               |
| deploy      | False   | Cannot validate image information for node␣
↪because one or more parameters are missing from its instance_info.␣
↪Missing are: ['ramdisk', 'kernel', 'image_source'] |
| inspect     | True    |                                            ␣
↪                                                                    ␣
↪                                               |
| management | True    |                                             ␣
↪                                                                    ␣
↪                                               |
| network     | True    |                                            ␣
↪                                                                    ␣
↪                                               |
| power       | True    |                                            ␣
↪                                                                    ␣
↪                                               |
| raid        | None    | not supported                              ␣
↪                                                                    ␣
↪                                               |
| storage     | True    |                                            ␣
↪                                                                    ␣
↪                                               |
```

(continues on next page)

(continued from previous page)

```
+-------------+--------+-----------------------------------------------
↪--------------------------------------------------------------------
↪------------------------------------------------+
```

Making node available for deployment

In order for nodes to be available for deploying workloads on them, nodes must be in the `available` provision state. To do this, nodes created with API version 1.11 and above must be moved from the `enroll` state to the `manageable` state and then to the `available` state. This section can be safely skipped, if API version 1.10 or earlier is used (which is the case by default).

After creating a node and before moving it from its initial provision state of `enroll`, basic power and port information needs to be configured on the node. The Bare Metal service needs this information because it verifies that it is capable of controlling the node when transitioning the node from `enroll` to `manageable` state.

To move a node from `enroll` to `manageable` provision state:

```
$ baremetal node manage $NODE_UUID
$ baremetal node show $NODE_UUID
+------------------------+---------------------------------------------
↪-------------------+
| Property               | Value                                      ␣
↪                   |
+------------------------+---------------------------------------------
↪-------------------+
| ...                    | ...                                        ␣
↪                   |
| provision_state        | manageable                                 ␣
↪                   | <- verify correct state
| uuid                   | 0eb013bb-1e4b-4f4c-94b5-2e7468242611       ␣
↪                   |
| ...                    | ...                                        ␣
↪                   |
+------------------------+---------------------------------------------
↪-------------------+
```

Note: Since it is an asynchronous call, the response for `baremetal node manage` will not indicate whether the transition succeeded or not. You can check the status of the operation via `baremetal node show`. If it was successful, `provision_state` will be in the desired state. If it failed, there will be information in the nodes `last_error`.

When a node is moved from the `manageable` to `available` provision state, the node will go through automated cleaning if configured to do so (see *Configure the Bare Metal service for cleaning*).

To move a node from `manageable` to `available` provision state:

```
$ baremetal node provide $NODE_UUID
$ baremetal node show $NODE_UUID
+------------------------+---------------------------------------------
↪-------------------+
```

(continues on next page)

(continued from previous page)

```
| Property               | Value                                                ␣
↪                   |
+------------------------+----------------------------------------------------
↪-------------------+
| ...                    | ...                                                  ␣
↪                   |
| provision_state        | available                                            ␣
↪                   | < - verify correct state
| uuid                   | 0eb013bb-1e4b-4f4c-94b5-2e7468242611                 ␣
↪                   |
| ...                    | ...                                                  ␣
↪                   |
+------------------------+----------------------------------------------------
↪-------------------+
```

For more details on the Bare Metal services state machine, see the *Ironics State Machine* documentation.

Mapping nodes to Compute cells

If the Compute service is used for scheduling, and the `discover_hosts_in_cells_interval` was not set as described in *Configure the Compute service to use the Bare Metal service*, then log into any controller node and run the following command to map the new node(s) to Compute cells:

```
nova-manage cell_v2 discover_hosts
```

Logical names

A node may also be referred to by a logical name as well as its UUID. Names can be assigned either during its creation by adding the `-n` option to the `node create` command or by updating an existing node with the `node set` command.

Node names must be unique, and conform to:

- rfc952
- rfc1123
- wiki_hostname

The node is named example in the following examples:

```
$ baremetal node create --driver ipmi --name example
```

or

```
$ baremetal node set $NODE_UUID --name example
```

Once assigned a logical name, a node can then be referred to by name or UUID interchangeably:

```
$ baremetal node create --driver ipmi --name example
+--------------+---------------------------------------+
| Property     | Value                                 |
+--------------+---------------------------------------+
```

(continues on next page)

(continued from previous page)

```
| uuid         | 71e01002-8662-434d-aafd-f068f69bb85e |
| driver_info  | {}                                   |
| extra        | {}                                   |
| driver       | ipmi                                 |
| chassis_uuid |                                      |
| properties   | {}                                   |
| name         | example                              |
+--------------+--------------------------------------+

$ baremetal node show example
+------------------------+--------------------------------------+
| Property               | Value                                |
+------------------------+--------------------------------------+
| target_power_state     | None                                 |
| extra                  | {}                                   |
| last_error             | None                                 |
| updated_at             | 2015-04-24T16:23:46+00:00            |
| ...                    | ...                                  |
| instance_info          | {}                                   |
+------------------------+--------------------------------------+
```

Defaults for hardware interfaces

For *hardware types*, users can request one of enabled implementations when creating or updating a node as explained in *Creating a node*.

When no value is provided for a certain interface when creating a node, or changing a nodes hardware type, the default value is used. You can use the driver details command to list the current enabled and default interfaces for a hardware type (for your deployment):

```
$ baremetal driver show ipmi
+-------------------------------+----------------+
| Field                         | Value          |
+-------------------------------+----------------+
| default_boot_interface        | pxe            |
| default_console_interface     | no-console     |
| default_deploy_interface      | iscsi          |
| default_inspect_interface     | no-inspect     |
| default_management_interface  | ipmitool       |
| default_network_interface     | flat           |
| default_power_interface       | ipmitool       |
| default_raid_interface        | no-raid        |
| default_vendor_interface      | no-vendor      |
| enabled_boot_interfaces       | pxe            |
| enabled_console_interfaces    | no-console     |
| enabled_deploy_interfaces     | iscsi, direct  |
| enabled_inspect_interfaces    | no-inspect     |
| enabled_management_interfaces | ipmitool       |
| enabled_network_interfaces    | flat, noop     |
| enabled_power_interfaces      | ipmitool       |
| enabled_raid_interfaces       | no-raid, agent |
| enabled_vendor_interfaces     | no-vendor      |
| hosts                         | ironic-host-1  |
| name                          | ipmi           |
```

(continues on next page)

(continued from previous page)

```
| type                           | dynamic         |
+--------------------------------+-----------------+
```

The defaults are calculated as follows:

1. If the `default_<IFACE>_interface` configuration option (where `<IFACE>` is the interface name) is set, its value is used as the default.

   If this implementation is not compatible with the nodes hardware type, an error is returned to a user. An explicit value has to be provided for the nodes `<IFACE>_interface` field in this case.

2. Otherwise, the first supported implementation that is enabled by an operator is used as the default.

   A list of supported implementations is calculated by taking the intersection between the implementations supported by the nodes hardware type and implementations enabled by the `enabled_<IFACE>_interfaces` option (where `<IFACE>` is the interface name). The calculation preserves the order of items, as provided by the hardware type.

   If the list of supported implementations is not empty, the first one is used. Otherwise, an error is returned to a user. In this case, an explicit value has to be provided for the `<IFACE>_interface` field.

See *Enabling drivers and hardware types* for more details on configuration.

Example

Consider the following configuration (shortened for simplicity):

```
[DEFAULT]
enabled_hardware_types = ipmi,redfish
enabled_console_interfaces = no-console,ipmitool-shellinabox
enabled_deploy_interfaces = iscsi,direct
enabled_management_interfaces = ipmitool,redfish
enabled_power_interfaces = ipmitool,redfish
default_deploy_interface = direct
```

A new node is created with the `ipmi` driver and no interfaces specified:

```
$ export OS_BAREMETAL_API_VERSION=1.31
$ baremetal node create --driver ipmi
+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| uuid         | dfc6189f-ad83-4261-9bda-b27258eb1987 |
| driver_info  | {}                                   |
| extra        | {}                                   |
| driver       | ipmi                                 |
| chassis_uuid |                                      |
| properties   | {}                                   |
| name         | None                                 |
+--------------+--------------------------------------+
```

Then the defaults for the interfaces that will be used by the node in this example are calculated as follows:

deploy An explicit value of `direct` is provided for `default_deploy_interface`, so it is used.

power No default is configured. The `ipmi` hardware type supports only `ipmitool` power. The intersection between supported power interfaces and values provided in the `enabled_power_interfaces` option has only one item: `ipmitool`. It is used.

console No default is configured. The `ipmi` hardware type supports the following console interfaces: `ipmitool-socat`, `ipmitool-shellinabox` and `no-console` (in this order). Of these three, only two are enabled: `no-console` and `ipmitool-shellinabox` (order does not matter). The intersection contains `ipmitool-shellinabox` and `no-console`. The first item is used, and it is `ipmitool-shellinabox`.

management Following the same calculation as *power*, the `ipmitool` management interface is used.

Hardware Inspection

The Bare Metal service supports hardware inspection that simplifies enrolling nodes - please see *Hardware Inspection* for details.

Tenant Networks and Port Groups

See *Multi-tenancy in the Bare Metal service* and *Port groups support*.

2.1.9 Using Bare Metal service as a standalone service

This guide explains how to configure and use the Bare Metal service standalone, i.e. without other OpenStack services. In this mode users are interacting with the bare metal API directly, not though OpenStack Compute.

Configuration

This guide covers manual configuration of the Bare Metal service in the standalone mode. Alternatively, Bifrost can be used for automatic configuration.

Service settings

It is possible to use the Bare Metal service without other OpenStack services. You should make the following changes to `/etc/ironic/ironic.conf`:

1. Choose an authentication strategy which supports standalone, one option is `noauth`:

   ```
   [DEFAULT]
   auth_strategy=noauth
   ```

   Another option is `http_basic` where the credentials are stored in an Apache htpasswd format file:

   ```
   [DEFAULT]
   auth_strategy=http_basic
   http_basic_auth_user_file=/etc/ironic/htpasswd
   ```

Only the `bcrypt` format is supported, and the Apache *htpasswd* utility can be used to populate the file with entries, for example:

```
htpasswd -nbB myName myPassword >> /etc/ironic/htpasswd
```

2. If you want to disable the Networking service, you should have your network pre-configured to serve DHCP and TFTP for machines that youre deploying. To disable it, change the following lines:

```
[dhcp]
dhcp_provider=none
```

Note: If you disabled the Networking service and the driver that you use is supported by at most one conductor, PXE boot will still work for your nodes without any manual config editing. This is because you know all the DHCP options that will be used for deployment and can set up your DHCP server appropriately.

If you have multiple conductors per driver, it would be better to use Networking since it will do all the dynamically changing configurations for you.

3. If you want to disable using a messaging broker between conductor and API processes, switch to JSON RPC instead:

```
[DEFAULT]
rpc_transport = json-rpc
```

JSON RPC also has its own authentication strategy. If it is not specified then the stategy defaults to `[DEFAULT] auth_strategy`. The following will set JSON RPC to `noauth`:

```
[json_rpc]
auth_strategy = noauth
```

For `http_basic` the conductor server needs a credentials file to validate requests:

```
[json_rpc]
auth_strategy = http_basic
http_basic_auth_user_file = /etc/ironic/htpasswd-json-rpc
```

The API server also needs client-side credentials to be specified:

```
[json_rpc]
auth_type = http_basic
username = myName
password = myPassword
```

Using CLI

To use the baremetal CLI, set up these environment variables. If the `noauth` authentication strategy is being used, the value `none` must be set for OS_AUTH_TYPE. OS_ENDPOINT is the URL of the ironic-api process. For example:

```
export OS_AUTH_TYPE=none
export OS_ENDPOINT=http://localhost:6385/
```

If the `http_basic` authentication strategy is being used, the value `http_basic` must be set for OS_AUTH_TYPE. For example:

```
export OS_AUTH_TYPE=http_basic
export OS_ENDPOINT=http://localhost:6385/
export OS_USERNAME=myUser
export OS_PASSWORD=myPassword
```

Enrollment

Preparing images

If you dont use Image service, its possible to provide images to Bare Metal service via a URL.

At the moment, only two types of URLs are acceptable instead of Image service UUIDs: HTTP(S) URLs (for example, http://my.server.net/images/img) and file URLs (file:///images/img).

There are however some limitations for different hardware interfaces:

- If youre using *Direct deploy* with HTTP(s) URLs, you have to provide the Bare Metal service with the a checksum of your instance image.

  MD5 is used by default for backward compatibility reasons. To compute an MD5 checksum, you can use the following command:

  ```
  $ md5sum image.qcow2
  ed82def8730f394fb85aef8a208635f6  image.qcow2
  ```

  Alternatively, use a SHA256 checksum or any other algorithm supported by the Pythons hashlib, e.g.:

  ```
  $ sha256sum image.qcow2
  9f6c942ad81690a9926ff530629fb69a82db8b8ab267e2cbd59df417c1a28060 ␣
  ↪image.qcow2
  ```

- *Direct deploy* started supporting `file://` images in the Victoria release cycle, before that only HTTP(s) had been supported.

> **Warning:** File images must be accessible to every conductor! Use a shared file system if you have more than one conductor. The ironic CLI tool will not transfer the file from a local machine to the conductor(s).

Note: The Bare Metal service tracks content changes for non-Glance images by checking their modification date and time. For example, for HTTP image, if Last-Modified header value from response to a HEAD request to http://my.server.net/images/deploy.ramdisk is greater than cached image modification time, Ironic will re-download the content. For file:// images, the file system modification time is used.

If the HTTP server does not provide the last modification date and time, the image will be redownloaded every time it is used.

Enrolling nodes

1. Create a node in Bare Metal service. At minimum, you must specify the driver name (for example, `ipmi`). You can also specify all the required driver parameters in one command. This will return the node UUID:

```
$ baremetal node create --driver ipmi \
    --driver-info ipmi_address=ipmi.server.net \
    --driver-info ipmi_username=user \
    --driver-info ipmi_password=pass \
    --driver-info deploy_kernel=file:///images/deploy.vmlinuz \
    --driver-info deploy_ramdisk=http://my.server.net/images/deploy.
↪ramdisk
+--------------+-------------------------------------------------------
↪--------------------+
| Property     | Value                                                ␣
↪                    |
+--------------+-------------------------------------------------------
↪--------------------+
| uuid         | be94df40-b80a-4f63-b92b-e9368ee8d14c                 ␣
↪                    |
| driver_info  | {u'deploy_ramdisk': u'http://my.server.net/images/
↪deploy.ramdisk',          |
|              | u'deploy_kernel': u'file:///images/deploy.vmlinuz', u
↪'ipmi_address':     |
|              | u'ipmi.server.net', u'ipmi_username': u'user', u
↪'ipmi_password':          |
|              | u'******'}                                           ␣
↪                    |
| extra        | {}                                                   ␣
↪                    |
| driver       | ipmi                                                 ␣
↪                    |
| chassis_uuid |                                                      ␣
↪                    |
| properties   | {}                                                   ␣
↪                    |
+--------------+-------------------------------------------------------
↪--------------------+
```

   Note that here deploy_kernel and deploy_ramdisk contain links to images instead of Image service UUIDs.

2. As in case of Compute service, you can also provide `capabilities` to node properties, but they will be used only by Bare Metal service (for example, boot mode). Although you dont need

to add properties like `memory_mb`, `cpus` etc. as Bare Metal service will require UUID of a node youre going to deploy.

3. Then create a port to inform Bare Metal service of the network interface cards which are part of the node by creating a port with each NICs MAC address. In this case, theyre used for naming of PXE configs for a node:

```
baremetal port create $MAC_ADDRESS --node $NODE_UUID
```

Deploying

Populating instance_info

Image information

You need to specify image information in the nodes `instance_info` (see *Create user images for the Bare Metal service*):

- `image_source` - URL of the whole disk or root partition image, mandatory.
- `root_gb` - size of the root partition, required for partition images.

Note: Older versions of the Bare Metal service used to require a positive integer for `root_gb` even for whole-disk images. You may want to set it for compatibility.

- `image_checksum` - MD5 checksum of the image specified by `image_source`, only required for `http://` images when using *Direct deploy*.

Note: Additional checksum support exists via the `image_os_hash_algo` and `image_os_hash_value` fields. They may be used instead of the `image_checksum` field.

Warning: If your operating system is running in FIPS 140-2 mode, MD5 will not be available, and you **must** use SHA256 or another modern algorithm.

Starting with the Stein release of ironic-python-agent can also be a URL to a checksums file, e.g. one generated with:

```
cd /path/to/http/root
md5sum *.img > checksums
```

- `kernel`, `ramdisk` - HTTP(s) or file URLs of the kernel and initramfs of the target OS. Must be added **only** for partition images.

For example:

```
baremetal node set $NODE_UUID \
    --instance-info image_source=$IMG \
    --instance-info image_checksum=$MD5HASH \
```

(continues on next page)

(continued from previous page)

```
    --instance-info kernel=$KERNEL \
    --instance-info ramdisk=$RAMDISK \
    --instance-info root_gb=10
```

With a SHA256 hash:

```
baremetal node set $NODE_UUID \
    --instance-info image_source=$IMG \
    --instance-info image_os_hash_algo=sha256 \
    --instance-info image_os_hash_value=$SHA256HASH \
    --instance-info kernel=$KERNEL \
    --instance-info ramdisk=$RAMDISK \
    --instance-info root_gb=10
```

With a whole disk image:

```
baremetal node set $NODE_UUID \
    --instance-info image_source=$IMG \
    --instance-info image_checksum=$MD5HASH
```

Note: For iLO drivers, fields that should be provided are:

- `ilo_deploy_iso` under `driver_info`;
- `ilo_boot_iso`, `image_source`, `root_gb` under `instance_info`.

When using low RAM nodes with `http://` images that are not in the RAW format, you may want them cached locally, converted to raw and served from the conductors HTTP server:

```
baremetal node set $NODE_UUID --instance-info image_download_source=local
```

For software RAID with whole-disk images, the root UUID of the root partition has to be provided so that the bootloader can be correctly installed:

```
baremetal node set $NODE_UUID --instance-info image_rootfs_uuid=<uuid>
```

Capabilities

- *Boot mode* can be specified per instance:

  ```
  baremetal node set $NODE_UUID \
      --instance-info capabilities='{"boot_mode": "uefi"}'
  ```

  Otherwise, the `boot_mode` capability from the nodes `properties` will be used.

  Warning: The two settings must not contradict each other.

  Note: This capability was introduced in the Wallaby release series, previously ironic used a separate `instance_info/deploy_boot_mode` field instead.

- To override the *boot option* used for this instance, set the `boot_option` capability:

  ```
  baremetal node set $NODE_UUID \
      --instance-info capabilities='{"boot_option": "local"}'
  ```

- Starting with the Ussuri release, you can set *root device hints* per instance:

  ```
  baremetal node set $NODE_UUID \
      --instance-info root_device='{"wwn": "0x4000cca77fc4dba1"}'
  ```

  This setting overrides any previous setting in `properties` and will be removed on undeployment.

Overriding a hardware interface

Non-admins with temporary access to a node, may wish to specify different node interfaces. However, allowing them to set these interface values directly on the node is problematic, as there is no automated way to ensure that the original interface values are restored.

In order to temporarily override a hardware interface, simply set the appropriate value in `instance_info`. For example, if youd like to override a nodes storage interface, run the following:

```
baremetal node set $NODE_UUID --instance-info storage_interface=cinder
```

`instance_info` values persist until after a node is cleaned.

Note: This feature is available starting with the Wallaby release.

Deployment

1. Validate that all parameters are correct:

   ```
   $ baremetal node validate $NODE_UUID
   +------------+--------+-----------------------------------------------
   ↪-----------------+
   | Interface  | Result | Reason                                        ␣
   ↪                 |
   +------------+--------+-----------------------------------------------
   ↪-----------------+
   | boot       | True   |                                               ␣
   ↪                 |
   | console    | False  | Missing 'ipmi_terminal_port' parameter in node
   ↪'s driver_info.  |
   | deploy     | True   |                                               ␣
   ↪                 |
   | inspect    | True   |                                               ␣
   ↪                 |
   | management | True   |                                               ␣
   ↪                 |
   ```

(continues on next page)

(continued from previous page)

```
| network     | True    |                                                    ␣
↪                   |
| power       | True    |                                                    ␣
↪                   |
| raid        | True    |                                                    ␣
↪                   |
| storage     | True    |                                                    ␣
↪                   |
+-------------+---------+-------------------------------------------------------
↪-----------------+
```

2. Now you can start the deployment, run:

```
baremetal node deploy $NODE_UUID
```

3. You can provide a configdrive as a JSON or as an ISO image, e.g.:

```
baremetal node deploy $NODE_UUID \
    --config-drive '{"meta_data": {"public_keys": {"0": "ssh key␣
↪contents"}}}'
```

   See *Enabling the configuration drive (configdrive)* for details.

4. Starting with the Wallaby release you can also request custom deploy steps, see *Requesting steps* for details.

Ramdisk booting

Advanced operators, specifically ones working with ephemeral workloads, may find it more useful to explicitly treat a node as one that would always boot from a Ramdisk. See *Booting a Ramdisk or an ISO* for details.

Other references

- *Enabling local boot without Compute*

2.1.10 Enabling the configuration drive (configdrive)

The Bare Metal service supports exposing a configuration drive image to the instances.

The configuration drive is used to store instance-specific metadata and is present to the instance as a disk partition labeled `config-2`. The configuration drive has a maximum size of 64MB. One use case for using the configuration drive is to expose a networking configuration when you do not use DHCP to assign IP addresses to instances.

The configuration drive is usually used in conjunction with the Compute service, but the Bare Metal service also offers a standalone way of using it. The following sections will describe both methods.

When used with Compute service

To enable the configuration drive for a specific request, pass `--config-drive true` parameter to the **`nova boot`** command, for example:

```
nova boot --config-drive true --flavor baremetal --image test-image␣
↪instance-1
```

Its also possible to enable the configuration drive automatically on all instances by configuring the `OpenStack Compute service` to always create a configuration drive by setting the following option in the `/etc/nova/nova.conf` file, for example:

```
[DEFAULT]
...

force_config_drive=True
```

In some cases, you may wish to pass a user customized script when deploying an instance. To do this, pass `--user-data /path/to/file` to the **`nova boot`** command.

When used standalone

When used without the Compute service, the operator needs to create a configuration drive and provide the file or HTTP URL to the Bare Metal service.

For the format of the configuration drive, Bare Metal service expects a `gzipped` and `base64` encoded ISO 9660[1] file with a `config-2` label. The baremetal client can generate a configuration drive in the expected format. Just pass a directory path containing the files that will be injected into it via the `--config-drive` parameter of the `baremetal node deploy` command, for example:

```
baremetal node deploy $node_identifier --config-drive /dir/configdrive_
↪files
```

Starting with the Stein release and *ironicclient* 2.7.0, you can request building a configdrive on the server side by providing a JSON with keys `meta_data`, `user_data` and `network_data` (all optional), e.g.:

```
baremetal node deploy $node_identifier \
    --config-drive '{"meta_data": {"hostname": "server1.cluster"}}'
```

Configuration drive storage in an object store

Under normal circumstances, the configuration drive can be stored in the Bare Metal service when the size is less than 64KB. Optionally, if the size is larger than 64KB there is support to store it in a swift endpoint. Both swift and radosgw use swift-style APIs.

The following option in `/etc/ironic/ironic.conf` enables swift as an object store backend to store config drive. This uses the Identity service to establish a session between the Bare Metal service and the Object Storage service.

---

[1] A configuration drive could also be a data block with a VFAT filesystem on it instead of ISO 9660. But its unlikely that it would be needed since ISO 9660 is widely supported across operating systems.

```
[deploy]
...

configdrive_use_object_store = True
```

Use the following options in `/etc/ironic/ironic.conf` to enable radosgw. Credentials in the swift section are needed because radosgw will not use the Identity service and relies on radosgws username and password authentication instead.

```
[deploy]
...

configdrive_use_object_store = True

[swift]
...

username = USERNAME
password = PASSWORD
auth_url = http://RADOSGW_IP:8000/auth/v1
```

If the *Direct deploy* is being used, edit `/etc/glance/glance-api.conf` to store the instance images in respective object store (radosgw or swift) as well:

```
[glance_store]
...

swift_store_user = USERNAME
swift_store_key = PASSWORD
swift_store_auth_address = http://RADOSGW_OR_SWIFT_IP:PORT/auth/v1
```

Accessing the configuration drive data

When the configuration drive is enabled, the Bare Metal service will create a partition on the instance disk and write the configuration drive image onto it. The configuration drive must be mounted before use. This is performed automatically by many tools, such as cloud-init and cloudbase-init. To mount it manually on a Linux distribution that supports accessing devices by labels, simply run the following:

```
mkdir -p /mnt/config
mount /dev/disk/by-label/config-2 /mnt/config
```

If the guest OS doesnt support accessing devices by labels, you can use other tools such as `blkid` to identify which device corresponds to the configuration drive and mount it, for example:

```
CONFIG_DEV=$(blkid -t LABEL="config-2" -odevice)
mkdir -p /mnt/config
mount $CONFIG_DEV /mnt/config
```

Cloud-init integration

The configuration drive can be especially useful when used with cloud-init, but in order to use it we should follow some rules:

- `Cloud-init` data should be organized in the expected format.
- Since the Bare Metal service uses a disk partition as the configuration drive, it will only work with cloud-init version >= 0.7.5.
- `Cloud-init` has a collection of data source modules, so when building the image with disk-image-builder we have to define `DIB_CLOUD_INIT_DATASOURCES` environment variable and set the appropriate sources to enable the configuration drive, for example:

  ```
  DIB_CLOUD_INIT_DATASOURCES="ConfigDrive, OpenStack" disk-image-create␣
  ↪-o fedora-cloud-image fedora baremetal
  ```

  For more information see how to configure cloud-init data sources.

2.1.11 Advanced features

Local boot with partition images

The Bare Metal service supports local boot with partition images, meaning that after the deployment the nodes subsequent reboots wont happen via PXE or Virtual Media. Instead, it will boot from a local boot loader installed on the disk.

Note: Whole disk images, on the contrary, support only local boot, and use it by default.

Its important to note that in order for this to work the image being deployed with Bare Metal service **must** contain `grub2` installed within it.

Enabling the local boot is different when Bare Metal service is used with Compute service and without it. The following sections will describe both methods.

Enabling local boot with Compute service

To enable local boot we need to set a capability on the bare metal node, for example:

```
baremetal node set <node-uuid> --property capabilities="boot_option:local"
```

Nodes having `boot_option` set to `local` may be requested by adding an `extra_spec` to the Compute service flavor, for example:

```
nova flavor-key baremetal set capabilities:boot_option="local"
```

Note: If the node is configured to use `UEFI`, Bare Metal service will create an `EFI partition` on the disk and switch the partition table format to `gpt`. The `EFI partition` will be used later by the boot loader (which is installed from the deploy ramdisk).

Enabling local boot without Compute

Since adding `capabilities` to the nodes properties is only used by the nova scheduler to perform more advanced scheduling of instances, we need a way to enable local boot when Compute is not present. To do that we can simply specify the capability via the `instance_info` attribute of the node, for example:

```
baremetal node set <node-uuid> --instance-info capabilities='{"boot_option
↪": "local"}'
```

Specifying the disk for deployment (root device hints)

The Bare Metal service supports passing hints to the deploy ramdisk about which disk it should pick for the deployment. The list of supported hints is:

- model (STRING): device identifier
- vendor (STRING): device vendor
- serial (STRING): disk serial number
- size (INT): size of the device in GiB

Note: A nodes local_gb property is often set to a value 1 GiB less than the actual disk size to account for partitioning (this is how DevStack, TripleO and Ironic Inspector work, to name a few). However, in this case `size` should be the actual size. For example, for a 128 GiB disk `local_gb` will be 127, but size hint will be 128.

- wwn (STRING): unique storage identifier
- wwn_with_extension (STRING): unique storage identifier with the vendor extension appended
- wwn_vendor_extension (STRING): unique vendor storage identifier
- rotational (BOOLEAN): whether its a rotational device or not. This hint makes it easier to distinguish HDDs (rotational) and SSDs (not rotational) when choosing which disk Ironic should deploy the image onto.
- hctl (STRING): the SCSI address (Host, Channel, Target and Lun), e.g 1:0:0:0
- by_path (STRING): the alternate device name corresponding to a particular PCI or iSCSI path, e.g /dev/disk/by-path/pci-0000:00
- name (STRING): the device name, e.g /dev/md0

Warning: The root device hint name should only be used for devices with constant names (e.g RAID volumes). For SATA, SCSI and IDE disk controllers this hint is not recommended because the order in which the device nodes are added in Linux is arbitrary, resulting in devices like /dev/sda and /dev/sdb switching around at boot time.

To associate one or more hints with a node, update the nodes properties with a `root_device` key, for example:

```
baremetal node set <node-uuid> --property root_device='{"wwn":
↪"0x4000cca77fc4dba1"}'
```

That will guarantee that Bare Metal service will pick the disk device that has the `wwn` equal to the specified wwn value, or fail the deployment if it can not be found.

Note: Starting with the Ussuri release, root device hints can be specified per-instance, see *Using Bare Metal service as a standalone service*.

The hints can have an operator at the beginning of the value string. If no operator is specified the default is `==` (for numerical values) and `s==` (for string values). The supported operators are:

- For numerical values:
  - `=` equal to or greater than. This is equivalent to `>=` and is supported for legacy reasons
  - `==` equal to
  - `!=` not equal to
  - `>=` greater than or equal to
  - `>` greater than
  - `<=` less than or equal to
  - `<` less than
- For strings (as python comparisons):
  - `s==` equal to
  - `s!=` not equal to
  - `s>=` greater than or equal to
  - `s>` greater than
  - `s<=` less than or equal to
  - `s<` less than
  - `<in>` substring
- For collections:
  - `<all-in>` all elements contained in collection
  - `<or>` find one of these

Examples are:

- Finding a disk larger or equal to 60 GiB and non-rotational (SSD):

```
baremetal node set <node-uuid> --property root_device='{"size": ">= 60
↪", "rotational": false}'
```

- Finding a disk whose vendor is `samsung` or `winsys`:

```
baremetal node set <node-uuid> --property root_device='{"vendor": "
↪<or> samsung <or> winsys"}'
```

Note: If multiple hints are specified, a device must satisfy all the hints.

Appending kernel parameters to boot instances

The Bare Metal service supports passing custom kernel parameters to boot instances to fit users requirements. The way to append the kernel parameters is depending on how to boot instances.

Network boot

Currently, the Bare Metal service supports assigning unified kernel parameters to PXE booted instances by:

- Modifying the `[pxe]/pxe_append_params` configuration option, for example:

  ```
  [pxe]

  pxe_append_params = quiet splash
  ```

- Copying a template from shipped templates to another place, for example:

  ```
  https://opendev.org/openstack/ironic/src/branch/master/ironic/drivers/
  ↪modules/pxe_config.template
  ```

  Making the modifications and pointing to the custom template via the configuration options: `[pxe]/pxe_config_template` and `[pxe]/uefi_pxe_config_template`.

Local boot

For local boot instances, users can make use of configuration drive (see *Enabling the configuration drive (configdrive)*) to pass a custom script to append kernel parameters when creating an instance. This is more flexible and can vary per instance. Here is an example for grub2 with ubuntu, users can customize it to fit their use case:

```
#!/usr/bin/env python
import os

# Default grub2 config file in Ubuntu
grub_file = '/etc/default/grub'
# Add parameters here to pass to instance.
kernel_parameters = ['quiet', 'splash']
grub_cmd = 'GRUB_CMDLINE_LINUX'
old_grub_file = grub_file+'~'
os.rename(grub_file, old_grub_file)
cmdline_existed = False
with open(grub_file, 'w') as writer, \
       open(old_grub_file, 'r') as reader:
       for line in reader:
           key = line.split('=')[0]
           if key == grub_cmd:
               #If there is already some value:
```

(continues on next page)

(continued from previous page)

```
                if line.strip()[-1] == '"':
                    line = line.strip()[:-1] + ' ' + ' '.join(kernel_
↪parameters) + '"'
                cmdline_existed = True
            writer.write(line)
        if not cmdline_existed:
            line = grub_cmd + '=' + '"' + ' '.join(kernel_parameters) + '"'
            writer.write(line)

os.remove(old_grub_file)
os.system('update-grub')
os.system('reboot')
```

Console

In order to change default console configuration in the Bare Metal service configuration file (`[pxe]` section in `/etc/ironic/ironic.conf`), include the serial port terminal and serial speed. Serial speed must be the same as the serial configuration in the BIOS settings, so that the operating system boot process can be seen in the serial console or web console. Following examples represent possible parameters for serial and web console respectively.

- Node serial console. The console parameter `console=ttyS0,115200n8` uses `ttyS0` for console output at `115200bps, 8bit, non-parity`, e.g.:

  ```
  [pxe]

  # Additional append parameters for baremetal PXE boot.
  pxe_append_params = nofb nomodeset vga=normal console=ttyS0,115200n8
  ```

- For node web console configuration is similar with the addition of `ttyX` parameter, see example:

  ```
  [pxe]

  # Additional append parameters for baremetal PXE boot.
  pxe_append_params = nofb nomodeset vga=normal console=tty0␣
  ↪console=ttyS0,115200n8
  ```

For detailed information on how to add consoles see the reference documents kernel params and serial console. In case of local boot the Bare Metal service is not able to control kernel boot parameters. To configure console locally, follow Local boot section above.

Boot mode support

Some of the bare metal hardware types (namely, `redfish`, `ilo` and generic `ipmi`) support setting boot mode (Legacy BIOS or UEFI).

Note: Setting boot mode support in generic `ipmi` driver is coupled with setting boot device. That makes boot mode support in the `ipmi` driver incomplete.

Note: In this chapter we will distinguish *ironic node* from *bare metal node*. The difference is that *ironic node* refers to a logical node, as it is configured in ironic, while *bare metal node* indicates the hardware machine that ironic is managing.

The following rules apply in order when ironic manages node boot mode:

- If the hardware type (or bare metal node) does not implement reading current boot mode of the bare metal node, then ironic assumes that boot mode is not set on the bare metal node
- If boot mode is not set on ironic node and bare metal node boot mode is unknown (not set, cant be read etc.), ironic node boot mode is set to the value of the *[deploy]/default_boot_mode* option
- If boot mode is set on a bare metal node, but is not set on ironic node, bare metal node boot mode is set on ironic node
- If boot mode is set on ironic node, but is not set on the bare metal node, ironic node boot mode is attempted to be set on the bare metal node (failure to set boot mode on the bare metal node will not fail ironic node deployment)
- If different boot modes appear on to be set ironic node and on the bare metal node, ironic node boot mode is attempted to be set on the bare metal node (failure to set boot mode on the bare metal node will fail ironic node deployment)

Warning: If a bare metal node does not support setting boot mode, then the operator needs to make sure that boot mode configuration is consistent between ironic node and the bare metal node.

The boot modes can be configured in the Bare Metal service in the following way:

- Only one boot mode (either `uefi` or `bios`) can be configured for the node.
- If the operator wants a node to boot always in `uefi` mode or `bios` mode, then they may use `capabilities` parameter within `properties` field of an bare metal node. The operator must manually set the appropriate boot mode on the bare metal node.

  To configure a node in `uefi` mode, then set `capabilities` as below:

  ```
  openstack baremetal node set <node-uuid> --property capabilities=
  ↪'boot_mode:uefi'
  ```

  Nodes having `boot_mode` set to `uefi` may be requested by adding an `extra_spec` to the Compute service flavor:

  ```
  nova flavor-key ironic-test-3 set capabilities:boot_mode="uefi"
  nova boot --flavor ironic-test-3 --image test-image instance-1
  ```

  If `capabilities` is used in `extra_spec` as above, nova scheduler (`ComputeCapabilitiesFilter`) will match only bare metal nodes which have the `boot_mode` set appropriately in `properties/capabilities`. It will filter out rest of the nodes.

  The above facility for matching in the Compute service can be used in heterogeneous environments where there is a mix of `uefi` and `bios` machines, and operator wants to provide a choice to the user regarding boot modes. If the flavor doesnt contain `boot_mode` and `boot_mode` is configured for bare metal nodes, then nova scheduler will consider all nodes and user may get either `bios` or `uefi` machine.

Some hardware support setting secure boot mode, see *UEFI secure boot mode* for details.

Choosing the disk label

Note: The term `disk label` is historically used in Ironic and was taken from parted. Apparently everyone seems to have a different word for `disk label` - these are all the same thing: disk type, partition table, partition map and so on

Ironic allows operators to choose which disk label they want their bare metal node to be deployed with when Ironic is responsible for partitioning the disk; therefore choosing the disk label does not apply when the image being deployed is a `whole disk image`.

There are some edge cases where someone may want to choose a specific disk label for the images being deployed, including but not limited to:

- For machines in `bios` boot mode with disks larger than 2 terabytes its recommended to use a `gpt` disk label. Thats because a capacity beyond 2 terabytes is not addressable by using the MBR partitioning type. But, although GPT claims to be backward compatible with legacy BIOS systems thats not always the case.
- Operators may want to force the partitioning to be always MBR (even if the machine is deployed with boot mode `uefi`) to avoid breakage of applications and tools running on those instances.

The disk label can be configured in two ways; when Ironic is used with the Compute service or in standalone mode. The following bullet points and sections will describe both methods:

- When no disk label is provided Ironic will configure it according to the boot mode (see *Boot mode support*); `bios` boot mode will use `msdos` and `uefi` boot mode will use `gpt`.
- Only one disk label - either `msdos` or `gpt` - can be configured for the node.

When used with Compute service

When Ironic is used with the Compute service the disk label should be set to nodes `properties/capabilities` field and also to the flavor which will request such capability, for example:

```
baremetal node set <node-uuid> --property capabilities='disk_label:gpt'
```

As for the flavor:

```
nova flavor-key baremetal set capabilities:disk_label="gpt"
```

When used in standalone mode

When used without the Compute service, the disk label should be set directly to the nodes `instance_info` field, as below:

```
baremetal node set <node-uuid> --instance-info capabilities='{"disk_label
↪": "gpt"}'
```

Notifications

The Bare Metal service supports the emission of notifications, which are messages sent on a message broker (like RabbitMQ or anything else supported by the oslo messaging library) that indicate various events which occur, such as when a node changes power states. These can be consumed by an external service reading from the message bus. For example, Searchlight is an OpenStack service that uses notifications to index (and make searchable) resources from the Bare Metal service.

Notifications are disabled by default. For a complete list of available notifications and instructions for how to enable them, see the *Notifications*.

Configuring node web console

See *Configuring Web or Serial Console*.

2.1.12 Troubleshooting

Once all the services are running and configured properly, and a node has been enrolled with the Bare Metal service and is in the `available` provision state, the Compute service should detect the node as an available resource and expose it to the scheduler.

Note: There is a delay, and it may take up to a minute (one periodic task cycle) for the Compute service to recognize any changes in the Bare Metal services resources (both additions and deletions).

In addition to watching `nova-compute` log files, you can see the available resources by looking at the list of Compute hypervisors. The resources reported therein should match the bare metal node properties, and the Compute service flavor.

Here is an example set of commands to compare the resources in Compute service and Bare Metal service:

```
$ baremetal node list
+--------------------------------------+---------------+-------------+-----
↪---------------+-------------+
| UUID                                 | Instance UUID | Power State |␣
↪Provisioning State | Maintenance |
+--------------------------------------+---------------+-------------+-----
↪---------------+-------------+
| 86a2b1bb-8b29-4964-a817-f90031debddb | None          | power off   |␣
↪available          | False       |
+--------------------------------------+---------------+-------------+-----
↪---------------+-------------+
```

(continues on next page)

(continued from previous page)

```
$ baremetal node show 86a2b1bb-8b29-4964-a817-f90031debddb
+------------------------+-------------------------------------------------
↪---------------------+
| Property               | Value                                          ␣
↪                     |
+------------------------+-------------------------------------------------
↪---------------------+
| instance_uuid          | None                                           ␣
↪                     |
| properties             | {u'memory_mb': u'1024', u'cpu_arch': u'x86_64',␣
↪u'local_gb': u'10',  |
|                        | u'cpus': u'1'}                                 ␣
↪                     |
| maintenance            | False                                          ␣
↪                     |
| driver_info            | { [SNIP] }                                     ␣
↪                     |
| extra                  | {}                                             ␣
↪                     |
| last_error             | None                                           ␣
↪                     |
| created_at             | 2014-11-20T23:57:03+00:00                      ␣
↪                     |
| target_provision_state | None                                           ␣
↪                     |
| driver                 | ipmi                                           ␣
↪                     |
| updated_at             | 2014-11-21T00:47:34+00:00                      ␣
↪                     |
| instance_info          | {}                                             ␣
↪                     |
| chassis_uuid           | 7b49bbc5-2eb7-4269-b6ea-3f1a51448a59           ␣
↪                     |
| provision_state        | available                                      ␣
↪                     |
| reservation            | None                                           ␣
↪                     |
| power_state            | power off                                      ␣
↪                     |
| console_enabled        | False                                          ␣
↪                     |
| uuid                   | 86a2b1bb-8b29-4964-a817-f90031debddb           ␣
↪                     |
+------------------------+-------------------------------------------------
↪---------------------+

$ nova hypervisor-list
+--------------------------------------+-----------------------------------
↪---+-------+---------+
| ID                                   | Hypervisor hostname              ␣
↪   | State | Status  |
+--------------------------------------+-----------------------------------
↪---+-------+---------+
| 584cfdc8-9afd-4fbb-82ef-9ff25e1ad3f3 | 86a2b1bb-8b29-4964-a817-
↪f90031debddb | up    | enabled |
```

(continues on next page)

(continued from previous page)

```
+--------------------------------------+-------------------------------------
↪---+-------+---------+

$ nova hypervisor-show 584cfdc8-9afd-4fbb-82ef-9ff25e1ad3f3
+--------------------------+---------------------------------------+
| Property                 | Value                                 |
+--------------------------+---------------------------------------+
| cpu_info                 | baremetal cpu                         |
| current_workload         | 0                                     |
| disk_available_least     | -                                     |
| free_disk_gb             | 10                                    |
| free_ram_mb              | 1024                                  |
| host_ip                  | [ SNIP ]                              |
| hypervisor_hostname      | 86a2b1bb-8b29-4964-a817-f90031debddb |
| hypervisor_type          | ironic                                |
| hypervisor_version       | 1                                     |
| id                       | 1                                     |
| local_gb                 | 10                                    |
| local_gb_used            | 0                                     |
| memory_mb                | 1024                                  |
| memory_mb_used           | 0                                     |
| running_vms              | 0                                     |
| service_disabled_reason  | -                                     |
| service_host             | my-test-host                          |
| service_id               | 6                                     |
| state                    | up                                    |
| status                   | enabled                               |
| vcpus                    | 1                                     |
| vcpus_used               | 0                                     |
+--------------------------+---------------------------------------+
```

Maintenance mode

Maintenance mode may be used if you need to take a node out of the resource pool. Putting a node in maintenance mode will prevent Bare Metal service from executing periodic tasks associated with the node. This will also prevent Compute service from placing a tenant instance on the node by not exposing the node to the nova scheduler. Nodes can be placed into maintenance mode with the following command.

```
$ baremetal node maintenance set $NODE_UUID
```

A maintenance reason may be included with the optional `--reason` command line option. This is a free form text field that will be displayed in the `maintenance_reason` section of the `node show` command.

```
$ baremetal node maintenance set $UUID --reason "Need to add ram."

$ baremetal node show $UUID

+------------------------+--------------------------------------+
| Property               | Value                                |
+------------------------+--------------------------------------+
| target_power_state     | None                                 |
```

(continues on next page)

(continued from previous page)

```
| extra                  | {}                                    |
| last_error             | None                                  |
| updated_at             | 2015-04-27T15:43:58+00:00             |
| maintenance_reason     | Need to add ram.                      |
| ...                    | ...                                   |
| maintenance            | True                                  |
| ...                    | ...                                   |
+------------------------+---------------------------------------+
```

To remove maintenance mode and clear any `maintenance_reason` use the following command.

```
$ baremetal node maintenance unset $NODE_UUID
```

2.1.13 Next steps

Your OpenStack environment now includes the Bare Metal service.

CHAPTER

THREE

UPGRADE GUIDE

3.1 Bare Metal Service Upgrade Guide

This document outlines various steps and notes for operators to consider when upgrading their ironic-driven clouds from previous versions of OpenStack.

The Bare Metal (ironic) service is tightly coupled with the ironic driver that is shipped with the Compute (nova) service. Some special considerations must be taken into account when upgrading your cloud.

Both offline and rolling upgrades are supported.

3.1.1 Plan your upgrade

- Rolling upgrades are available starting with the Pike release; that is, when upgrading from Ocata. This means that it is possible to do an upgrade with minimal to no downtime of the Bare Metal API.
- Upgrades are only supported between two consecutive named releases. This means that you cannot upgrade Ocata directly into Queens; you need to upgrade into Pike first.
- The release notes should always be read carefully when upgrading the Bare Metal service. Specific upgrade steps and considerations are documented there.
- The Bare Metal service should always be upgraded before the Compute service.

  Note: The ironic virt driver in nova always uses a specific version of the ironic REST API. This API version may be one that was introduced in the same development cycle, so upgrading nova first may result in nova being unable to use the Bare Metal API.

- Make a backup of your database. Ironic does not support downgrading of the database. Hence, in case of upgrade failure, restoring the database from a backup is the only choice.
- Before starting your upgrade, it is best to ensure that all nodes have reached, or are in, a stable `provision_state`. Nodes in states with long running processes such as deploying or cleaning, may fail, and may require manual intervention to return them to the available hardware pool. This is most likely in cases where a timeout has occurred or a service was terminated abruptly. For a visual diagram detailing states and possible state transitions, please see *Ironics State Machine*.

3.1.2 Offline upgrades

In an offline (or cold) upgrade, the Bare Metal service is not available during the upgrade, because all the services have to be taken down.

When upgrading the Bare Metal service, the following steps should always be taken in this order:

1. upgrade the ironic-python-agent image
2. update ironic code, without restarting services
3. run database schema migrations via `ironic-dbsync upgrade`
4. restart ironic-conductor and ironic-api services

Once the above is done, do the following:

- update any applicable configuration options to stop using any deprecated features or options, and perform any required work to transition to alternatives. All the deprecated features and options will be supported for one release cycle, so should be removed before your next upgrade is performed.
- upgrade python-ironicclient along with any other services connecting to the Bare Metal service as a client, such as nova-compute
- run the `ironic-dbsync online_data_migrations` command to make sure that data migrations are applied. The command lets you limit the impact of the data migrations with the `--max-count` option, which limits the number of migrations executed in one run. You should complete all of the migrations as soon as possible after the upgrade.

> **Warning:** You will not be able to start an upgrade to the release after this one, until this has been completed for the current release. For example, as part of upgrading from Ocata to Pike, you need to complete Pikes data migrations. If this not done, you will not be able to upgrade to Queens it will not be possible to execute Queens database schema updates.

3.1.3 Rolling upgrades

To Reduce downtime, the services can be upgraded in a rolling fashion, meaning to upgrade one or a few services at a time to minimize impact.

Rolling upgrades are available starting with the Pike release. This feature makes it possible to upgrade between releases, such as Ocata to Pike, with minimal to no downtime of the Bare Metal API.

Requirements

To facilitate an upgrade in a rolling fashion, you need to have a highly-available deployment consisting of at least two ironic-api and two ironic-conductor services. Use of a load balancer to balance requests across the ironic-api services is recommended, as it allows for a minimal impact to end users.

Concepts

There are four aspects of the rolling upgrade process to keep in mind:

- API and RPC version pinning, and versioned object backports
- online data migrations
- graceful service shutdown
- API load balancer draining

API & RPC version pinning and versioned object backports

Through careful RPC versioning, newer services are able to talk to older services (and vice-versa). The `[DEFAULT]/pin_release_version` configuration option is used for this. It should be set (pinned) to the release version that the older services are using. The newer services will backport RPC calls and objects to their appropriate versions from the pinned release. If the `IncompatibleObjectVersion` exception occurs, it is most likely due to an incorrect or unspecified `[DEFAULT]/pin_release_version` configuration value. For example, when `[DEFAULT]/pin_release_version` is not set to the older release version, no conversion will happen during the upgrade.

For the ironic-api service, the API version is pinned via the same `[DEFAULT]/pin_release_version` configuration option as above. When pinned, the new ironic-api services will not service any API requests with Bare Metal API versions that are higher than what the old ironic-api services support. HTTP status code 406 is returned for such requests. This prevents new features (available in new API versions) from being used until after the upgrade has been completed.

Online data migrations

To make database schema migrations less painful to execute, we have implemented process changes to facilitate upgrades.

- All data migrations are banned from schema migration scripts.
- Schema migration scripts only update the database schema.
- Data migrations must be done at the end of the rolling upgrade process, after the schema migration and after the services have been upgraded to the latest release.

All data migrations are performed using the `ironic-dbsync online_data_migrations` command. It can be run as a background process so that it does not interrupt running services; however it must be run to completion for a cold upgrade if the intent is to make use of new features immediately.

(You would also execute the same command with services turned off if you are doing a cold upgrade).

This data migration must be completed. If not, you will not be able to upgrade to future releases. For example, if you had upgraded from Ocata to Pike but did not do the data migrations, you will not be able to upgrade from Pike to Queens. (More precisely, you will not be able to apply Queens schema migrations.)

Graceful conductor service shutdown

The ironic-conductor service is a Python process listening for messages on a message queue. When the operator sends the SIGTERM signal to the process, the service stops consuming messages from the queue, so that no additional work is picked up. It completes any outstanding work and then terminates. During this process, messages can be left on the queue and will be processed after the Python process starts back up. This gives us a way to shutdown a service using older code, and start up a service using newer code with minimal impact.

Note: This was tested with RabbitMQ messaging backend and may vary with other backends.

Nodes that are being acted upon by an ironic-conductor process, which are not in a stable state, may encounter failures. Node failures that occur during an upgrade are likely due to timeouts, resulting from delays involving messages being processed and acted upon by a conductor during long running, multi-step processes such as deployment or cleaning.

API load balancer draining

If you are using a load balancer for the ironic-api services, we recommend that you redirect requests to the new API services and drain off of the ironic-api services that have not yet been upgraded.

Rolling upgrade process

Before maintenance window

- Upgrade the ironic-python-agent image
- Using the new release (ironic code), execute the required database schema updates by running the database upgrade command: `ironic-dbsync upgrade`. These schema change operations should have minimal or no effect on performance, and should not cause any operations to fail (but please check the release notes). You can:
  - install the new release on an existing system
  - install the new release in a new virtualenv or a container

  At this point, new columns and tables may exist in the database. These database schema changes are done in a way that both the old and new (N and N+1) releases can perform operations against the same schema.

Note: Ironic bases its API, RPC and object storage format versions on the `[DEFAULT]/pin_release_version` configuration option. It is advisable to automate the deployment of changes in configuration files to make the process less error prone and repeatable.

During maintenance window

1. All ironic-conductor services should be upgraded first. Ensure that at least one ironic-conductor service is running at all times. For every ironic-conductor, either one by one or a few at a time:

    - shut down the service. Messages from the ironic-api services to the conductors are load-balanced by the message queue and a hash-ring, so the only thing you need to worry about is to shut the service down gracefully (using `SIGTERM` signal) to make sure it will finish all the requests being processed before shutting down.
    - upgrade the installed version of ironic and dependencies
    - set the `[DEFAULT]/pin_release_version` configuration option value to the version you are upgrading from (that is, the old version). Based on this setting, the new ironic-conductor services will downgrade any RPC communication and data objects to conform to the old service. For example, if you are upgrading from Ocata to Pike, set this value to `ocata`.
    - start the service

2. The next service to upgrade is ironic-api. Ensure that at least one ironic-api service is running at all times. You may want to start another temporary instance of the older ironic-api to handle the load while you are upgrading the original ironic-api services. For every ironic-api service, either one by one or a few at a time:

    - in HA deployment you are typically running them behind a load balancer (for example HAProxy), so you need to take the service instance out of the balancer
    - shut it down
    - upgrade the installed version of ironic and dependencies
    - set the `[DEFAULT]/pin_release_version` configuration option value to the version you are upgrading from (that is, the old version). Based on this setting, the new ironic-api services will downgrade any RPC communication and data objects to conform to the old service. In addition, the new services will return HTTP status code 406 for any requests with newer API versions that the old services did not support. This prevents new features (available in new API versions) from being used until after the upgrade has been completed. For example, if you are upgrading from Ocata to Pike, set this value to `ocata`.
    - restart the service
    - add it back into the load balancer

    After upgrading all the ironic-api services, the Bare Metal service is running in the new version but with downgraded RPC communication and database object storage formats. New features (in new API versions) are not supported, because they could fail when objects are in the downgraded object formats and some internal RPC API functions may still not be available.

3. For all the ironic-conductor services, one at a time:

    - remove the `[DEFAULT]/pin_release_version` configuration option setting
    - restart the ironic-conductor service

4. For all the ironic-api services, one at a time:

    - remove the `[DEFAULT]/pin_release_version` configuration option setting
    - restart the ironic-api service

After maintenance window

Now that all the services are upgraded, the system is able to use the latest version of the RPC protocol and able to access all the features of the new release.

- Update any applicable configuration options to stop using any deprecated features or options, and perform any required work to transition to alternatives. All the deprecated features and options will be supported for one release cycle, so should be removed before your next upgrade is performed.
- Upgrade `python-ironicclient` along with other services connecting to the Bare Metal service as a client, such as `nova-compute`.

  Warning: A `nova-compute` instance tries to attach VIFs to all active instances on start up. Make sure that for all active nodes there is at least one running `ironic-conductor` process to manage them. Otherwise the instances will be moved to the `ERROR` state on the `nova-compute` start up.

- Run the `ironic-dbsync online_data_migrations` command to make sure that data migrations are applied. The command lets you limit the impact of the data migrations with the `--max-count` option, which limits the number of migrations executed in one run. You should complete all of the migrations as soon as possible after the upgrade.

  Warning: Note that you will not be able to start an upgrade to the next release after this one, until this has been completed for the current release. For example, as part of upgrading from Ocata to Pike, you need to complete Pikes data migrations. If this not done, you will not be able to upgrade to Queens it will not be possible to execute Queens database schema updates.

3.1.4 Upgrading from Ocata to Pike

1. Use the `ironic-dbsync online_data_migrations` command from the 9.1.1 (or newer) release. The one from older (9.0.0 - 9.1.0) releases could cause a a ports physical_network information to be deleted from the database.
2. It is required to set the `resource_class` field for nodes registered with the Bare Metal service *before* using the Pike version of the Compute service. See *Enrollment* for details.
3. It is recommended to move from old-style classic drivers to the new hardware types after the upgrade to Pike. We expect the classic drivers to be deprecated in the Queens release and removed in the Rocky release. See *Upgrading to Hardware Types* for the details on the migration.

Other upgrade instructions are in the Pike release notes.

Upgrading to Hardware Types

Starting with the Rocky release, the Bare Metal service does not support *classic drivers* any more. If you still use *classic drivers*, please upgrade to *hardware types* immediately. Please see *Enabling drivers and hardware types* for details on *hardware types* and *hardware interfaces*.

Planning the upgrade

It is necessary to figure out which hardware types and hardware interfaces correspond to which classic drivers used in your deployment. The following table lists the classic drivers with their corresponding hardware types and the boot, deploy, inspect, management, and power hardware interfaces:

Classic Driver	Hardware Type	Boot	Deploy	Inspect	Management	Power
agent_ilo	ilo	ilo-virtual-media	direct	ilo	ilo	ilo
agent_ipmitool	ipmi	pxe	direct	inspector	ipmitool	ipmitool
agent_ipmitool_socat	ipmi	pxe	direct	inspector	ipmitool	ipmitool
agent_irmc	irmc	irmc-virtual-media	direct	irmc	irmc	irmc
iscsi_ilo	ilo	ilo-virtual-media	iscsi	ilo	ilo	ilo
iscsi_irmc	irmc	irmc-virtual-media	iscsi	irmc	irmc	irmc
pxe_drac	idrac	pxe	iscsi	idrac	idrac	idrac
pxe_drac_inspector	idrac	pxe	iscsi	inspector	idrac	idrac
pxe_ilo	ilo	ilo-pxe	iscsi	ilo	ilo	ilo
pxe_ipmitool	ipmi	pxe	iscsi	inspector	ipmitool	ipmitool
pxe_ipmitool_socat	ipmi	pxe	iscsi	inspector	ipmitool	ipmitool
pxe_irmc	irmc	irmc-pxe	iscsi	irmc	irmc	irmc
pxe_snmp	snmp	pxe	iscsi	no-inspect	fake	snmp

Note: The `inspector` *inspect* interface was only used if explicitly enabled in the configuration. Otherwise, `no-inspect` was used.

Note: `pxe_ipmitool_socat` and `agent_ipmitool_socat` use `ipmitool-socat` *console* interface (the default for the `ipmi` hardware type), while `pxe_ipmitool` and `agent_ipmitool` use `ipmitool-shellinabox`. See *Console* for details.

For out-of-tree drivers you may need to reach out to their maintainers or figure out the appropriate interfaces by researching the source code.

Configuration

You will need to enable hardware types and interfaces that correspond to your currently enabled classic drivers. For example, if you have the following configuration in your `ironic.conf`:

```
[DEFAULT]
enabled_drivers = pxe_ipmitool,agent_ipmitool
```

You will have to add this configuration as well:

```
[DEFAULT]
enabled_hardware_types = ipmi
enabled_boot_interfaces = pxe
enabled_deploy_interfaces = iscsi,direct
enabled_management_interfaces = ipmitool
enabled_power_interfaces = ipmitool
```

Note: For every interface type there is an option `default_<INTERFACE>_interface`, where `<INTERFACE>` is the interface type name. For example, one can make all nodes use the `direct` deploy method by default by setting:

```
[DEFAULT]
default_deploy_interface = direct
```

Migrating nodes

After the required items are enabled in the configuration, each nodes `driver` field has to be updated to a new value. You may need to also set new values for some or all interfaces:

```
export OS_BAREMETAL_API_VERSION=1.31

for uuid in $(baremetal node list --driver pxe_ipmitool -f value -c UUID);␣
↪do
    baremetal node set <node> --driver ipmi --deploy-interface iscsi
done

for uuid in $(baremetal node list --driver agent_ipmitool -f value -c␣
↪UUID); do
    baremetal node set <node> --driver ipmi --deploy-interface direct
done
```

See *Enrollment* for more details on setting hardware types and interfaces.

Warning: It is not recommended to change the interfaces for `active` nodes. If absolutely needed, the nodes have to be put in the maintenance mode first:

```
baremetal node maintenance set $UUID \
    --reason "Changing driver and/or hardware interfaces"
# do the update, validate its correctness
baremetal node maintenance unset $UUID
```

Other interfaces

Care has to be taken to migrate from classic drivers using non-default interfaces. This chapter covers a few of the most commonly used.

Ironic Inspector

Some classic drivers, notably `pxe_ipmitool`, `agent_ipmitool` and `pxe_drac_inspector`, use ironic-inspector for their *inspect* interface.

The same functionality is available for all hardware types, but the appropriate `inspect` interface has to be enabled in the Bare Metal service configuration file, for example:

```
[DEFAULT]
enabled_inspect_interfaces = inspector,no-inspect
```

See *Enabling drivers and hardware types* for more details.

Note: The configuration option `[inspector]enabled` does not affect hardware types.

Then you can tell your nodes to use this interface, for example:

```
export OS_BAREMETAL_API_VERSION=1.31
for uuid in $(baremetal node list --driver ipmi -f value -c UUID); do
    baremetal node set <node> --inspect-interface inspector
done
```

Note: A node configured with the IPMI hardware type, will use the inspector inspection implementation automatically if it is enabled. This is not the case for the most of the vendor drivers.

Console

Several classic drivers, notably `pxe_ipmitool_socat` and `agent_ipmitool_socat`, use socat-based serial console implementation.

For the `ipmi` hardware type it is used by default, if enabled in the configuration file:

```
[DEFAULT]
enabled_console_interfaces = ipmitool-socat,no-console
```

If you want to use the `shellinabox` implementation instead, it has to be enabled as well:

```
[DEFAULT]
enabled_console_interfaces = ipmitool-shellinabox,no-console
```

Then you need to update some or all nodes to use it explicitly. For example, to update all nodes use:

```
export OS_BAREMETAL_API_VERSION=1.31
for uuid in $(baremetal node list --driver ipmi -f value -c UUID); do
    baremetal node set <node> --console-interface ipmitool-shellinabox
done
```

RAID

Many classic drivers, including `pxe_ipmitool` and `agent_ipmitool` use the IPA-based in-band RAID implementation by default.

For the hardware types it is not used by default. To use it, you need to enable it in the configuration first:

```
[DEFAULT]
enabled_raid_interfaces = agent,no-raid
```

Then you can update those nodes that support in-band RAID to use the `agent` RAID interface. For example, to update all nodes use:

```
export OS_BAREMETAL_API_VERSION=1.31
for uuid in $(baremetal node list --driver ipmi -f value -c UUID); do
    baremetal node set <node> --raid-interface agent
done
```

Note: The ability of a node to use the `agent` RAID interface depends on the ramdisk (more specifically, a hardware manager used in it), not on the driver.

Network and storage

The network and storage interfaces have always been dynamic, and thus do not require any special treatment during upgrade.

Vendor

Classic drivers are allowed to use the `VendorMixin` functionality to combine and expose several node or driver vendor passthru methods from different vendor interface implementations in one driver.

This is no longer possible with hardware types.

With hardware types, a vendor interface can only have a single active implementation from the list of vendor interfaces supported by a given hardware type.

Ironic no longer has in-tree drivers (both classic and hardware types) that rely on this `VendorMixin` functionality support. However if you are using an out-of-tree classic driver that depends on it, youll need to do the following in order to use vendor passthru methods from different vendor passthru implementations:

1. While creating a new hardware type to replace your classic driver, specify all vendor interface implementations your classic driver was using to build its `VendorMixin` as supported vendor interfaces (property `supported_vendor_interfaces` of the Python class that defines your hardware type).

2. Ensure all required vendor interfaces are enabled in the ironic configuration file under the `[DEFAULT]enabled_vendor_interfaces` option. You should also consider setting the `[DEFAULT]default_vendor_interface` option to specify the vendor interface for nodes that do not have one set explicitly.
3. Before invoking a specific vendor passthru method, make sure that the nodes vendor interface is set to the interface with the desired vendor passthru method. For example, if you want to invoke the vendor passthru method `vendor_method_foo()` from `vendor_foo` vendor interface:

```
# set the vendor interface to 'vendor_foo`
baremetal node set <node> --vendor-interface vendor_foo

# invoke the vendor passthru method
baremetal node passthru call <node> vendor_method_foo
```

3.1.5 Upgrading from Newton to Ocata

There are no specific upgrade instructions other than the Ocata release notes.

3.1.6 Upgrading from Mitaka to Newton

There are no specific upgrade instructions other than the Newton release notes.

3.1.7 Upgrading from Liberty to Mitaka

There are no specific upgrade instructions other than the Mitaka release notes.

3.1.8 Upgrading from Kilo to Liberty

In-band Inspection

If you used in-band inspection with **ironic-discoverd**, it is highly recommended that you switch to using **ironic-inspector**, which is a newer (and compatible on API level) version of the same service. You have to install **python-ironic-inspector-client** during the upgrade. This package contains a client module for the in-band inspection service, which was previously part of the **ironic-discoverd** package. Ironic Liberty supports the **ironic-discoverd** service, but does not support its in-tree client module. Please refer to ironic-inspector version support matrix for details on which ironic versions are compatible with which **ironic-inspector/ironic-discoverd** versions.

The discoverd to inspector upgrade procedure is as follows:

- Install **ironic-inspector** on the machine where you have **ironic-discoverd** (usually the same as conductor).
- Update the **ironic-inspector** configuration file to stop using deprecated configuration options, as marked by the comments in the example.conf. It is recommended you move the configuration file to `/etc/ironic-inspector/inspector.conf`.
- Shutdown **ironic-discoverd**, and start **ironic-inspector**.
- During upgrade of each conductor instance:

1. Shutdown the conductor.
2. Uninstall **ironic-discoverd**, install **python-ironic-inspector-client**.
3. Update the conductor.
4. Update `ironic.conf` to use `[inspector]` section instead of `[discoverd]` (option names are the same).
5. Start the conductor.

3.1.9 Upgrading from Juno to Kilo

When upgrading a cloud from Juno to Kilo, users must ensure the nova service is upgraded prior to upgrading the ironic service. Additionally, users need to set a special config flag in nova prior to upgrading to ensure the newer version of nova is not attempting to take advantage of new ironic features until the ironic service has been upgraded. The steps for upgrading your nova and ironic services are as follows:

- Edit nova.conf and ensure force_config_drive=False is set in the [DEFAULT] group. Restart nova-compute if necessary.
- Install new nova code, run database migrations.
- Install new python-ironicclient code.
- Restart nova services.
- Install new ironic code, run database migrations, restart ironic services.
- Edit nova.conf and set force_config_drive to your liking, restarting nova-compute if necessary.

Note that during the period between novas upgrade and ironics upgrades, instances can still be provisioned to nodes. However, any attempt by users to specify a config drive for an instance will cause an error until ironics upgrade has completed.

Cleaning

A new feature starting from Kilo cycle is support for the automated cleaning of nodes between workloads to ensure the node is ready for another workload. This can include erasing the hard drives, updating firmware, and other steps. For more information, see *Automated cleaning*.

If ironic is configured with automated cleaning enabled (defaults to True) and neutron is set as the DHCP provider (also the default), you will need to set the *cleaning_network_uuid* option in the ironic configuration file before starting the ironic service. See *Configure the Bare Metal service for cleaning* for information on how to set up the cleaning network for ironic.

CHAPTER

FOUR

USER GUIDE

4.1 Bare Metal Service User Guide

Ironic is an OpenStack project which provisions bare metal (as opposed to virtual) machines. It may be used independently or as part of an OpenStack Cloud, and integrates with the OpenStack Identity (keystone), Compute (nova), Network (neutron), Image (glance) and Object (swift) services.

When the Bare Metal service is appropriately configured with the Compute and Network services, it is possible to provision both virtual and physical machines through the Compute services API. However, the set of instance actions is limited, arising from the different characteristics of physical servers and switch hardware. For example, live migration can not be performed on a bare metal instance.

The community maintains reference drivers that leverage open-source technologies (eg. PXE and IPMI) to cover a wide range of hardware. Ironics pluggable driver architecture also allows hardware vendors to write and contribute drivers that may improve performance or add functionality not provided by the community drivers.

4.1.1 Why Provision Bare Metal

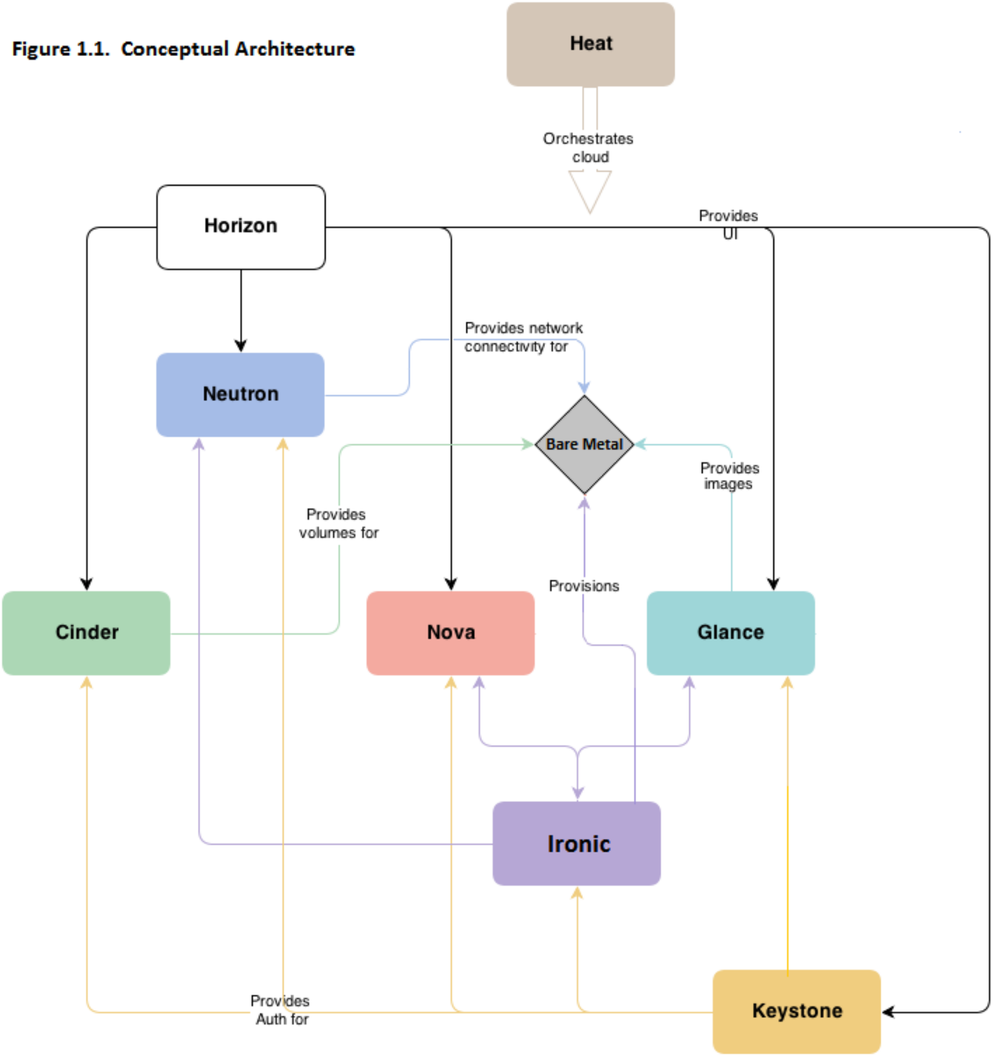
Here are a few use-cases for bare metal (physical server) provisioning in cloud; there are doubtless many more interesting ones:

- High-performance computing clusters
- Computing tasks that require access to hardware devices which cant be virtualized
- Database hosting (some databases run poorly in a hypervisor)
- Single tenant, dedicated hardware for performance, security, dependability and other regulatory requirements
- Or, rapidly deploying a cloud infrastructure

4.1.2 Conceptual Architecture

The following diagram shows the relationships and how all services come into play during the provisioning of a physical server. (Note that Ceilometer and Swift can be used with Ironic, but are missing from this diagram.)

Figure 1.1. Conceptual Architecture


Heat
Orchestrates cloud
Horizon
Provides UI
Provides network connectivity for
Neutron
Bare Metal
Provides images
Provides volumes for
Provisions
Cinder
Nova
Glance
Ironic
Provides Auth for
Keystone

4.1.3 Key Technologies for Bare Metal Hosting

Preboot Execution Environment (PXE)

PXE is part of the Wired for Management (WfM) specification developed by Intel and Microsoft. The PXE enables systems BIOS and network interface card (NIC) to bootstrap a computer from the network in place of a disk. Bootstrapping is the process by which a system loads the OS into local memory so that it can be executed by the processor. This capability of allowing a system to boot over a network simplifies server deployment and server management for administrators.

Dynamic Host Configuration Protocol (DHCP)

DHCP is a standardized networking protocol used on Internet Protocol (IP) networks for dynamically distributing network configuration parameters, such as IP addresses for interfaces and services. Using PXE, the BIOS uses DHCP to obtain an IP address for the network interface and to locate the server that stores the network bootstrap program (NBP).

Network Bootstrap Program (NBP)

NBP is equivalent to GRUB (GRand Unified Bootloader) or LILO (LInux LOader) - loaders which are traditionally used in local booting. Like the boot program in a hard drive environment, the NBP is responsible for loading the OS kernel into memory so that the OS can be bootstrapped over a network.

Trivial File Transfer Protocol (TFTP)

TFTP is a simple file transfer protocol that is generally used for automated transfer of configuration or boot files between machines in a local environment. In a PXE environment, TFTP is used to download NBP over the network using information from the DHCP server.

Intelligent Platform Management Interface (IPMI)

IPMI is a standardized computer system interface used by system administrators for out-of-band management of computer systems and monitoring of their operation. It is a method to manage systems that may be unresponsive or powered off by using only a network connection to the hardware rather than to an operating system.

4.1.4 Understanding Bare Metal Deployment

What happens when a boot instance request comes in? The below diagram walks through the steps involved during the provisioning of a bare metal instance.

These pre-requisites must be met before the deployment process:

- Dependent packages to be configured on the Bare Metal service node(s) where ironic-conductor is running like tftp-server, ipmi, syslinux etc for bare metal provisioning.
- Nova must be configured to make use of the bare metal service endpoint and compute driver should be configured to use ironic driver on the Nova compute node(s).
- Flavors to be created for the available hardware. Nova must know the flavor to boot from.
- Images to be made available in Glance. Listed below are some image types required for successful bare metal deployment:
    - bm-deploy-kernel
    - bm-deploy-ramdisk
    - user-image
    - user-image-vmlinuz
    - user-image-initrd

- Hardware to be enrolled via the bare metal API service.

Deploy Process

This describes a typical bare metal node deployment within OpenStack using PXE to boot the ramdisk. Depending on the ironic driver interfaces used, some of the steps might be marginally different, however the majority of them will remain the same.

1. A boot instance request comes in via the Nova API, through the message queue to the Nova scheduler.
2. Nova scheduler applies filters and finds the eligible hypervisor. The nova scheduler also uses the flavors `extra_specs`, such as `cpu_arch`, to match the target physical node.
3. Nova compute manager claims the resources of the selected hypervisor.
4. Nova compute manager creates (unbound) tenant virtual interfaces (VIFs) in the Networking service according to the network interfaces requested in the nova boot request. A caveat here is, the MACs of the ports are going to be randomly generated, and will be updated when the VIF is attached to some node to correspond to the node network interface cards (or bonds) MAC.
5. A spawn task is created by the nova compute which contains all the information such as which image to boot from etc. It invokes the `driver.spawn` from the virt layer of Nova compute. During the spawn process, the virt driver does the following:
    1. Updates the target ironic node with the information about deploy image, instance UUID, requested capabilities and various flavor properties.
    2. Validates nodes power and deploy interfaces, by calling the ironic API.
    3. Attaches the previously created VIFs to the node. Each neutron port can be attached to any ironic port or port group, with port groups having higher priority than ports. On ironic side, this work is done by the network interface. Attachment here means saving the VIF identifier into ironic port or port group and updating VIF MAC to match the ports or port groups MAC, as described in bullet point 4.
    4. Generates config drive, if requested.
6. Novas ironic virt driver issues a deploy request via the Ironic API to the Ironic conductor servicing the bare metal node.
7. Virtual interfaces are plugged in and Neutron API updates DHCP port to set PXE/TFTP options. In case of using `neutron` network interface, ironic creates separate provisioning ports in the Networking service, while in case of `flat` network interface, the ports created by nova are used both for provisioning and for deployed instance networking.
8. The ironic nodes boot interface prepares (i)PXE configuration and caches deploy kernel and ramdisk.
9. The ironic nodes management interface issues commands to enable network boot of a node.
10. The ironic nodes deploy interface caches the instance image, kernel and ramdisk if needed (it is needed in case of netboot for example).
11. The ironic nodes power interface instructs the node to power on.
12. The node boots the deploy ramdisk.

13. Depending on the exact driver used, the deploy ramdisk downloads the image from a URL (*Direct deploy*) or the conductor uses SSH to execute commands (*Ansible deploy*). The URL can be generated by Swift API-compatible object stores, for example Swift itself or RadosGW, or provided by a user.

    The image deployment is done.

14. The nodes boot interface switches pxe config to refer to instance images (or, in case of local boot, sets boot device to disk), and asks the ramdisk agent to soft power off the node. If the soft power off by the ramdisk agent fails, the bare metal node is powered off via IPMI/BMC call.

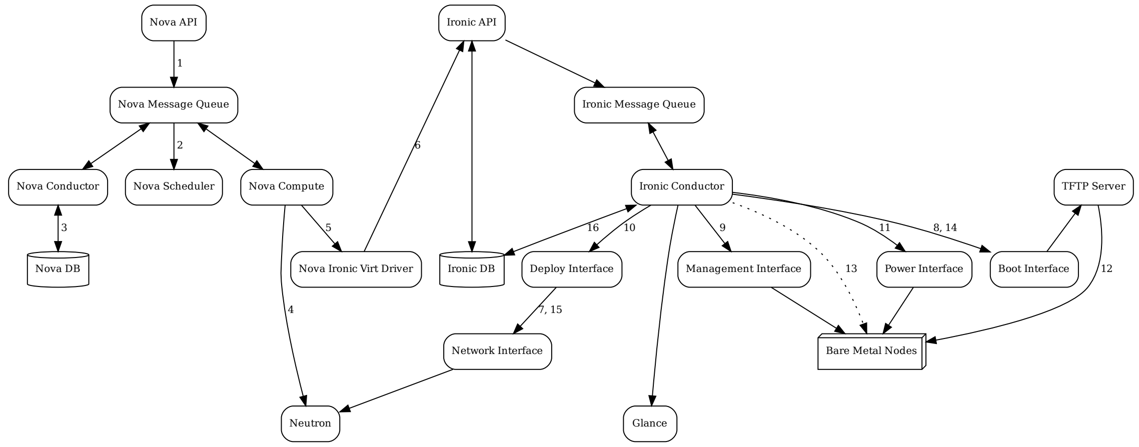
15. The deploy interface triggers the network interface to remove provisioning ports if they were created, and binds the tenant ports to the node if not already bound. Then the node is powered on.

    Note: There are 2 power cycles during bare metal deployment; the first time the node is powered-on when ramdisk is booted, the second time after the image is deployed.

    ---

16. The bare metal nodes provisioning state is updated to `active`.

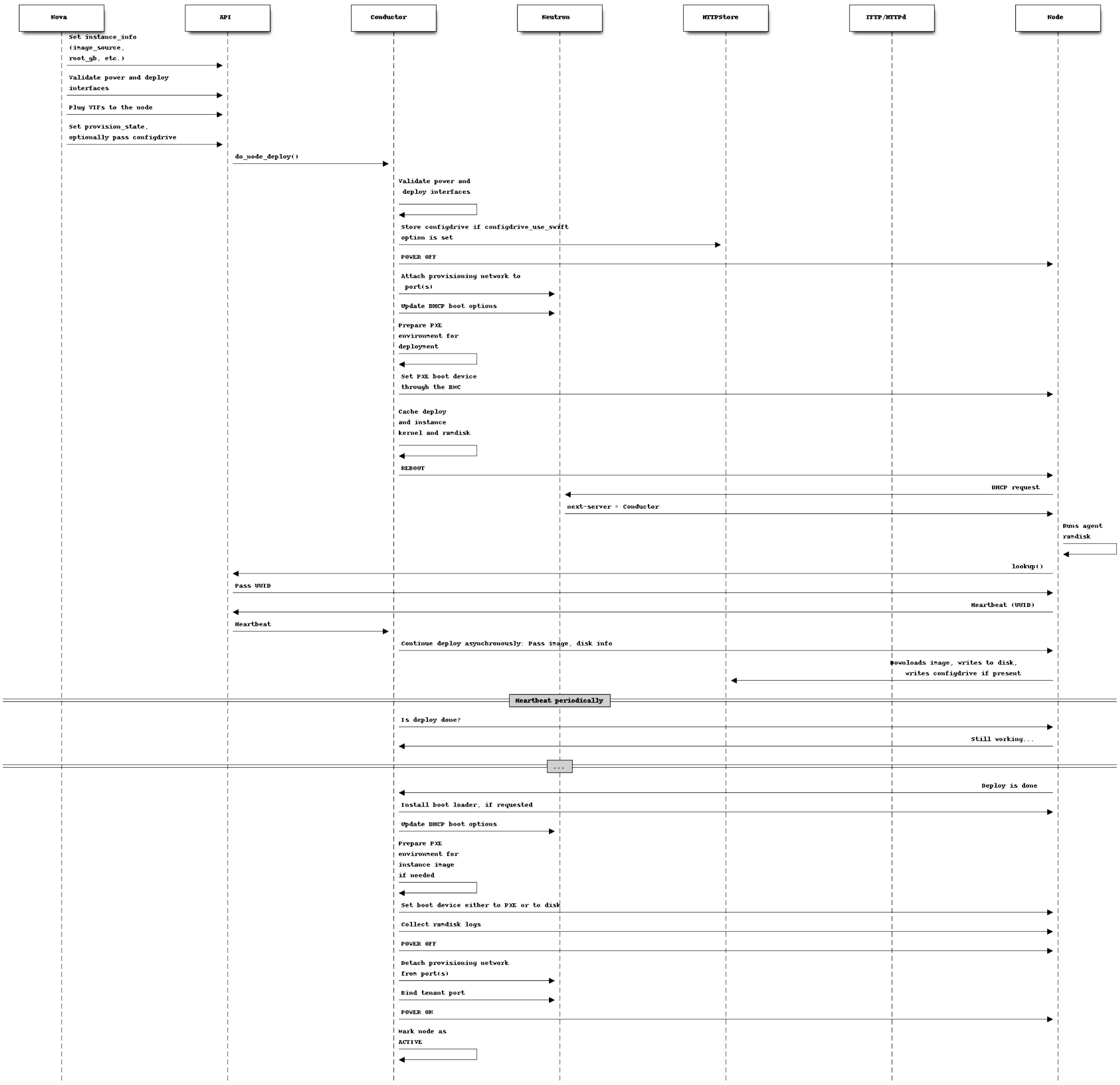
Below is the diagram that describes the above process.


Nova API
1
Nova Message Queue
2
Nova Conductor
Nova Scheduler
Nova Compute
3
Nova DB
4
5
Nova Ironic Virt Driver
6
Ironic API
Ironic DB
Ironic Message Queue
Ironic Conductor
16
10
Deploy Interface
7, 15
Network Interface
Neutron
Glance
9
Management Interface
13
11
Power Interface
8, 14
Boot Interface
TFTP Server
12
Bare Metal Nodes

The following two examples describe what ironic is doing in more detail, leaving out the actions performed by nova and some of the more advanced options.

Example 1: PXE Boot and Direct Deploy Process

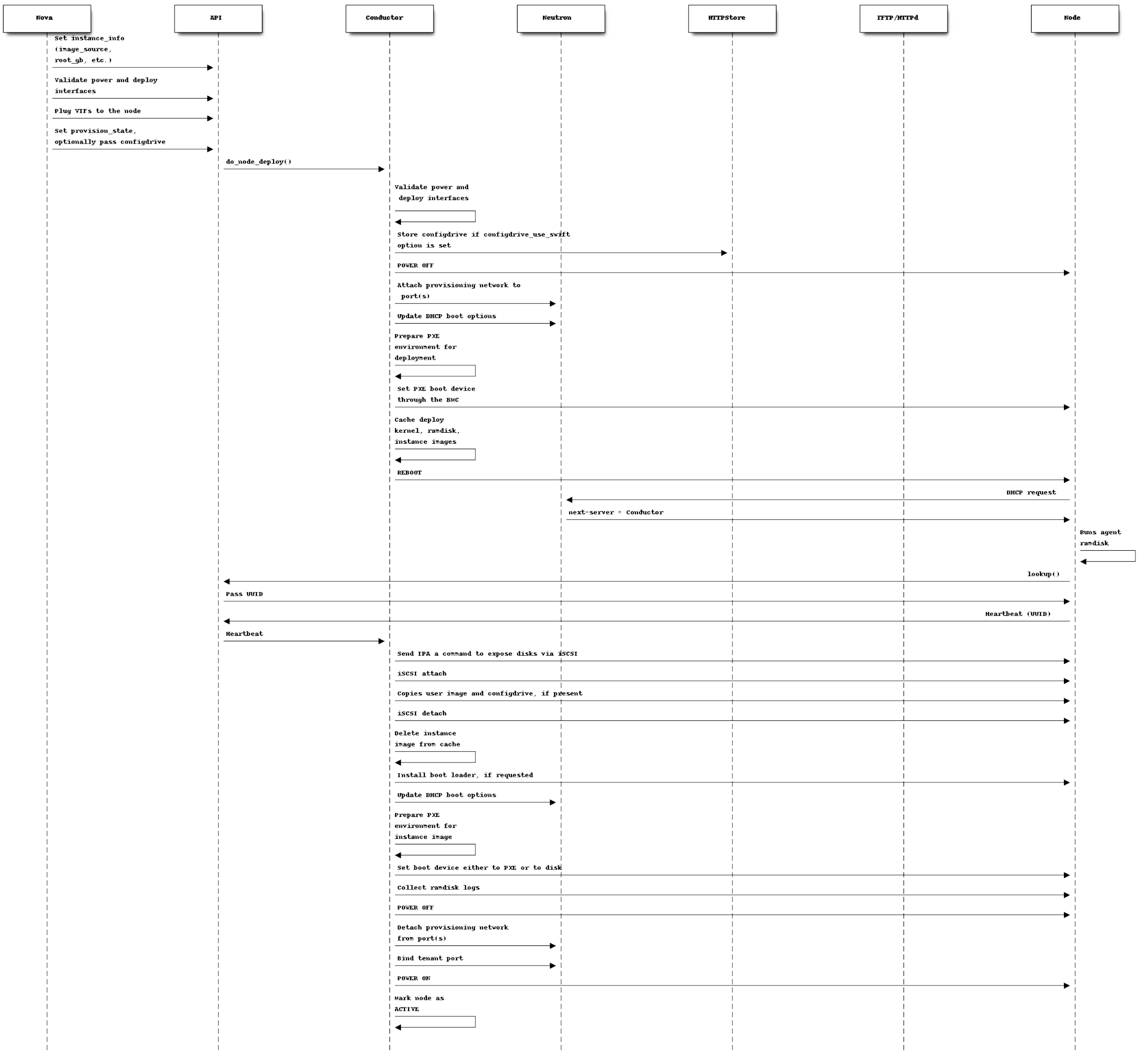
This process is how *Direct deploy* works.


Nova
API
Conductor
Neutron
HTTPStore
TFTP/HTTPd
Node
Set instance_info (image_source, root_gb, etc.)
Validate power and deploy interfaces
Plug VIFs to the node
Set provision_state, optionally pass configdrive
do_node_deploy()
Validate power and deploy interfaces
Store configdrive if configdrive_use_swift option is set
POWER OFF
Attach provisioning network to port(s)
Update DHCP boot options
Prepare PXE environment for deployment
Set PXE boot device through the BMC
Cache deploy and instance kernel and ramdisk
REBOOT
DHCP request
next-server = Conductor
Runs agent ramdisk
lookup()
Pass UUID
Heartbeat (UUID)
Heartbeat
Continue deploy asynchronously: Pass image, disk info
Downloads image, writes to disk, writes configdrive if present
Heartbeat periodically
Is deploy done?
Still working...
...
Deploy is done
Install boot loader, if requested
Update DHCP boot options
Prepare PXE environment for instance image if needed
Set boot device either to PXE or to disk
Collect ramdisk logs
POWER OFF
Detach provisioning network from port(s)
Bind tenant port
POWER ON
Mark node as ACTIVE

(From a talk and slides)

Example 2: PXE Boot and iSCSI Deploy Process

This process is how the currently deprecated *iSCSI deploy* works.


Nova
API
Conductor
Neutron
HTTPStore
TFTP/HTTPd
Node
Set instance_info (image_source, root_gb, etc.)
Validate power and deploy interfaces
Plug VIFs to the node
Set provision_state, optionally pass configdrive
do_node_deploy()
Validate power and deploy interfaces
Store configdrive if configdrive_use_swift option is set
POWER OFF
Attach provisioning network to port(s)
Update DHCP boot options
Prepare PXE environment for deployment
Set PXE boot device through the BMC
Cache deploy kernel, ramdisk, instance images
REBOOT
DHCP request
next-server = Conductor
Runs agent ramdisk
lookup()
Pass UUID
Heartbeat (UUID)
Heartbeat
Send IPA a command to expose disks via iSCSI
iSCSI attach
Copies user image and configdrive, if present
iSCSI detach
Delete instance image from cache
Install boot loader, if requested
Update DHCP boot options
Prepare PXE environment for instance image
Set boot device either to PXE or to disk
Collect ramdisk logs
POWER OFF
Detach provisioning network from port(s)
Bind tenant port
POWER ON
Mark node as ACTIVE

(From a talk and slides)

CHAPTER

FIVE

ADMINISTRATOR GUIDE

5.1 Drivers, Hardware Types and Hardware Interfaces

5.1.1 Generic Interfaces

Boot interfaces

The boot interface manages booting of both the deploy ramdisk and the user instances on the bare metal node.

The *PXE boot* interface is generic and works with all hardware that supports booting from network. Alternatively, several vendors provide *virtual media* implementations of the boot interface. They work by pushing an ISO image to the nodes management controller, and do not require either PXE or iPXE. Check your driver documentation at *Drivers, Hardware Types and Hardware Interfaces* for details.

PXE boot

The `pxe` boot interface uses PXE or iPXE to deliver the target kernel/ramdisk pair. PXE uses relatively slow and unreliable TFTP protocol for transfer, while iPXE uses HTTP. The downside of iPXE is that its less common, and usually requires bootstrapping using PXE first.

The `pxe` boot interface works by preparing a PXE/iPXE environment for a node on the file system, then instructing the DHCP provider (for example, the Networking service) to boot the node from it. See *Example 2: PXE Boot and iSCSI Deploy Process* and *Example 1: PXE Boot and Direct Deploy Process* for a better understanding of the whole deployment process.

Note: Both PXE and iPXE are configured differently, when UEFI boot is used instead of conventional BIOS boot. This is particularly important for CPU architectures that do not have BIOS support at all.

The `pxe` boot interface is used by default for many hardware types, including `ipmi`. Some hardware types, notably `ilo` and `irmc` have their specific implementations of the PXE boot interface.

Additional configuration is required for this boot interface - see *Configuring PXE and iPXE* for details.

Common options

Enable persistent boot device for deploy/clean operation

Ironic uses non-persistent boot for cleaning/deploying phases as default. For some drivers, a persistent change is far more costly than a non-persistent one, so this can bring performance improvements.

Set the flag `force_persistent_boot_device` to `True` in the nodes `driver_info`:

```
$ baremetal node set --driver-info force_persistent_boot_device=True <node>
```

Note: Its recommended to check if the nodes state has not changed as there is no way of locking the node between these commands.

Once the flag is present, the next cleaning and deploy steps will be done with persistent boot for that node.

Deploy Interfaces

A *deploy* interface plays a critical role in the provisioning process. It orchestrates the whole deployment and defines how the image gets transferred to the target disk.

Direct deploy

With `direct` deploy interface, the deploy ramdisk fetches the image from an HTTP location. It can be an object storage (swift or RadosGW) temporary URL or a user-provided HTTP URL. The deploy ramdisk then copies the image to the target disk. See *direct deploy diagram* for a detailed explanation of how this deploy interface works.

You can specify this deploy interface when creating or updating a node:

```
baremetal node create --driver ipmi --deploy-interface direct
baremetal node set <NODE> --deploy-interface direct
```

Note: For historical reasons the `direct` deploy interface is sometimes called `agent`. This is because before the Kilo release **ironic-python-agent** used to only support this deploy interface.

Deploy with custom HTTP servers

The `direct` deploy interface can also be configured to use with custom HTTP servers set up at ironic conductor nodes, images will be cached locally and made accessible by the HTTP server.

To use this deploy interface with a custom HTTP server, set `image_download_source` to `http` in the `[agent]` section.

```
[agent]
...
image_download_source = http
...
```

This configuration affects *glance* and `file://` images. If you want `http(s)://` images to also be cached and served locally, use instead:

```
[agent]
image_download_source = local
```

Note: This option can also be set per node in `driver_info`:

```
baremetal node set <node> --driver-info image_download_source=local
```

or per instance in `instance_info`:

```
baremetal node set <node> --instance-info image_download_source=local
```

You need to set up a workable HTTP server at each conductor node which with `direct` deploy interface enabled, and check http related options in the ironic configuration file to match the HTTP server configurations.

```
[deploy]
http_url = http://example.com
http_root = /httpboot
```

Note: See also: *Deploying outside of the provisioning network*.

Each HTTP server should be configured to follow symlinks for images accessible from HTTP service. Please refer to configuration option `FollowSymLinks` if you are using Apache HTTP server, or `disable_symlinks` if Nginx HTTP server is in use.

Ansible deploy

This interface is similar to `direct` in the sense that the image is downloaded by the ramdisk directly from the image store (not from ironic-conductor host), but the logic of provisioning the node is held in a set of Ansible playbooks that are applied by the `ironic-conductor` service handling the node. While somewhat more complex to set up, this deploy interface provides greater flexibility in terms of advanced node preparation during provisioning.

This interface is supported by most but not all hardware types declared in ironic. However this deploy interface is not enabled by default. To enable it, add `ansible` to the list of enabled deploy interfaces in `enabled_deploy_interfaces` option in the `[DEFAULT]` section of ironics configuration file:

```
[DEFAULT]
...
enabled_deploy_interfaces = iscsi,direct,ansible
...
```

Once enabled, you can specify this deploy interface when creating or updating a node:

```
baremetal node create --driver ipmi --deploy-interface ansible
baremetal node set <NODE> --deploy-interface ansible
```

For more information about this deploy interface, its features and how to use it, see *Ansible deploy interface*.

Ansible deploy interface

Ansible is a mature and popular automation tool, written in Python and requiring no agents running on the node being configured. All communications with the node are by default performed over secure SSH transport.

The `ansible` deploy interface uses Ansible playbooks to define the deployment logic. It is not based on Ironic Python Agent (IPA) and does not generally need IPA to be running in the deploy ramdisk.

Overview

The main advantage of this deploy interface is extended flexibility in regards to changing and adapting node deployment logic for specific use cases, via Ansible tooling that is already familiar to operators.

It can be used to shorten the usual feature development cycle of

- implementing logic in ironic,
- implementing logic in IPA,
- rebuilding deploy ramdisk,
- uploading deploy ramdisk to Glance/HTTP storage,
- reassigning deploy ramdisk to nodes,
- restarting ironic-conductor service(s) and
- running a test deployment

by using a stable deploy ramdisk and not requiring ironic-conductor restarts (see *Extending playbooks*).

The main disadvantage of this deploy interface is the synchronous manner of performing deployment/cleaning tasks. A separate `ansible-playbook` process is spawned for each node being provisioned or cleaned, which consumes one thread from the thread pool available to the `ironic-conductor` process and blocks this thread until the node provisioning or cleaning step is finished or fails. This has to be taken into account when planning an ironic deployment that enables this deploy interface.

Each action (deploy, clean) is described by a single playbook with roles, which is run whole during deployment, or tag-wise during cleaning. Control of cleaning steps is through tags and auxiliary clean steps file. The playbooks for actions can be set per-node, as can the clean steps file.

Features

Similar to deploy interfaces relying on Ironic Python Agent (IPA), this deploy interface also depends on the deploy ramdisk calling back to ironic APIs `heartbeat` endpoint.

However, the driver is currently synchronous, so only the first heartbeat is processed and is used as a signal to start `ansible-playbook` process.

User images

Supports whole-disk images and partition images:

- compressed images are downloaded to RAM and converted to disk device;
- raw images are streamed to disk directly.

For partition images the driver will create root partition, and, if requested, ephemeral and swap partitions as set in nodes `instance_info` by the Compute service or operator. The create partition table will be of `msdos` type by default, the nodes `disk_label` capability is honored if set in nodes `instance_info` (see also *Choosing the disk label*).

Configdrive partition

Creating a configdrive partition is supported for both whole disk and partition images, on both `msdos` and `GPT` labeled disks.

Root device hints

Root device hints are currently supported in their basic form only, with exact matches (see *Specifying the disk for deployment (root device hints)* for more details). If no root device hint is provided for the node, the first device returned as part of `ansible_devices` fact is used as root device to create partitions on or write the whole disk image to.

Node cleaning

Cleaning is supported, both automated and manual. The driver has two default clean steps:

- wiping device metadata
- disk shredding

Their priority can be overridden via `[deploy]\erase_devices_metadata_priority` and `[deploy]\erase_devices_priority` options, respectively, in the ironic configuration file.

As in the case of this driver all cleaning steps are known to the ironic-conductor service, booting the deploy ramdisk is completely skipped when there are no cleaning steps to perform.

Note: Aborting cleaning steps is not supported.

Logging

Logging is implemented as custom Ansible callback module, that makes use of `oslo.log` and `oslo.config` libraries and can re-use logging configuration defined in the main ironic configuration file to set logging for Ansible events, or use a separate file for this purpose.

It works best when `journald` support for logging is enabled.

Requirements

Ansible Tested with, and targets, Ansible 2.5.x

Bootstrap image requirements

- password-less sudo permissions for the user used by Ansible
- python 2.7.x
- openssh-server
- GNU coreutils
- utils-linux
- parted
- gdisk
- qemu-utils
- python-requests (for ironic callback and streaming image download)
- python-netifaces (for ironic callback)

A set of scripts to build a suitable deploy ramdisk based on TinyCore Linux and `tinyipa` ramdisk, and an element for `diskimage-builder` can be found in ironic-staging-drivers project but will be eventually migrated to the new ironic-python-agent-builder project.

Setting up your environment

1. Install ironic (either as part of OpenStack or standalone)
   - If using ironic as part of OpenStack, ensure that the Image service is configured to use the Object Storage service as backend, and the Bare Metal service is configured accordingly, see *Configure the Image service for temporary URLs*.
2. Install Ansible version as specified in `ironic/driver-requirements.txt` file
3. Edit ironic configuration file
   A. Add `ansible` to the list of deploy interfaces defined in `[DEFAULT]\enabled_deploy_interfaces` option.
   B. Ensure that a hardware type supporting `ansible` deploy interface is enabled in `[DEFAULT]\enabled_hardware_types` option.

   C. Modify options in the `[ansible]` section of ironics configuration file if needed (see *Configuration file*).

4. (Re)start ironic-conductor service
5. Build suitable deploy kernel and ramdisk images
6. Upload them to Glance or put in your HTTP storage
7. Create new or update existing nodes to use the enabled driver of your choice and populate *Driver properties for the Node* when different from defaults.
8. Deploy the node as usual.

Ansible-deploy options

Configuration file

Driver options are configured in `[ansible]` section of ironic configuration file, for their descriptions and default values please see configuration file sample.

Driver properties for the Node

Set them per-node via `baremetal node set` command, for example:

```
baremetal node set <node> \
    --deploy-interface ansible \
    --driver-info ansible_username=stack \
    --driver-info ansible_key_file=/etc/ironic/id_rsa
```

ansible_username User name to use for Ansible to access the node. Default is taken from `[ansible]/default_username` option of the ironic configuration file (defaults to `ansible`).

ansible_key_file Private SSH key used to access the node. Default is taken from `[ansible]/default_key_file` option of the ironic configuration file. If neither is set, the default private SSH keys of the user running the `ironic-conductor` process will be used.

ansible_deploy_playbook Playbook to use when deploying this node. Default is taken from `[ansible]/default_deploy_playbook` option of the ironic configuration file (defaults to `deploy.yaml`).

ansible_shutdown_playbook Playbook to use to gracefully shutdown the node in-band. Default is taken from `[ansible]/default_shutdown_playbook` option of the ironic configuration file (defaults to `shutdown.yaml`).

ansible_clean_playbook Playbook to use when cleaning the node. Default is taken from `[ansible]/default_clean_playbook` option of the ironic configuration file (defaults to `clean.yaml`).

ansible_clean_steps_config Auxiliary YAML file that holds description of cleaning steps used by this node, and defines playbook tags in `ansible_clean_playbook` file corresponding to each cleaning step. Default is taken from `[ansible]/default_clean_steps_config` option of the ironic configuration file (defaults to `clean_steps.yaml`).

ansible_python_interpreter Absolute path to the python interpreter on the managed machine. Default is taken from `[ansible]/default_python_interpreter` option of the ironic configuration file. Ansible uses `/usr/bin/python` by default.

Customizing the deployment logic

Expected playbooks directory layout

The `[ansible]\playbooks_path` option in the ironic configuration file is expected to have a standard layout for an Ansible project with some additions:

```
<playbooks_path>
|
\_ inventory
\_ add-ironic-nodes.yaml
\_ roles
 \_ role1
 \_ role2
 \_ ...
|
\_callback_plugins
 \_ ...
|
\_ library
 \_ ...
```

The extra files relied by this driver are:

inventory Ansible inventory file containing a single entry of `conductor ansible_connection=local`. This basically defines an alias to `localhost`. Its purpose is to make logging for tasks performed by Ansible locally and referencing the localhost in playbooks more intuitive. This also suppresses warnings produced by Ansible about `hosts` file being empty.

add-ironic-nodes.yaml This file contains an Ansible play that populates in-memory Ansible inventory with access information received from the ansible-deploy interface, as well as some per-node variables. Include it in all your custom playbooks as the first play.

The default `deploy.yaml` playbook is using several smaller roles that correspond to particular stages of deployment process:

- `discover` - e.g. set root device and image target
- `prepare` - if needed, prepare system, for example create partitions
- `deploy` - download/convert/write user image and configdrive
- `configure` - post-deployment steps, e.g. installing the bootloader

Some more included roles are:

- `shutdown` - used to gracefully power the node off in-band
- `clean` - defines cleaning procedure, with each clean step defined as separate playbook tag.

Extending playbooks

Most probably youd start experimenting like this:

1. Create a copy of `deploy.yaml` playbook *in the same folder*, name it distinctively.
2. Create Ansible roles with your customized logic in `roles` folder.
   A. In your custom deploy playbook, replace the `prepare` role with your own one that defines steps to be run *before* image download/writing. This is a good place to set facts overriding those provided/omitted by the driver, like `ironic_partitions` or `ironic_root_device`, and create custom partitions or (software) RAIDs.
   B. In your custom deploy playbook, replace the `configure` role with your own one that defines steps to be run *after* image is written to disk. This is a good place for example to configure the bootloader and add kernel options to avoid additional reboots.
   C. Use those new roles in your new playbook.
3. Assign the custom deploy playbook youve created to the nodes `driver_info/ansible_deploy_playbook` field.
4. Run deployment.
   A. No ironic-conductor restart is necessary.
   B. A new deploy ramdisk must be built and assigned to nodes only when you want to use a command/script/package not present in the current deploy ramdisk and you can not or do not want to install those at runtime.

Variables you have access to

This driver will pass the single JSON-ified extra var argument to Ansible (as in `ansible-playbook -e ..`). Those values are then accessible in your plays as well (some of them are optional and might not be defined):

```
ironic:
  nodes:
  - ip: "<IPADDRESS>"
    name: "<NODE_UUID>"
    user: "<USER ANSIBLE WILL USE>"
    extra: "<COPY OF NODE's EXTRA FIELD>"
  image:
    url: "<URL TO FETCH THE USER IMAGE FROM>"
    disk_format: "<qcow2|raw|...>"
    container_format: "<bare|...>"
    checksum: "<hash-algo:hashstring>"
    mem_req: "<REQUIRED FREE MEMORY TO DOWNLOAD IMAGE TO RAM>"
    tags: "<LIST OF IMAGE TAGS AS DEFINED IN GLANCE>"
    properties: "<DICT OF IMAGE PROPERTIES AS DEFINED IN GLANCE>"
  configdrive:
    type: "<url|file>"
    location: "<URL OR PATH ON CONDUCTOR>"
  partition_info:
    label: "<msdos|gpt>"
    preserve_ephemeral: "<bool>"
```

(continues on next page)

(continued from previous page)

```
    ephemeral_format: "<FILESYSTEM TO CREATE ON EPHEMERAL PARTITION>"
    partitions: "<LIST OF PARTITIONS IN FORMAT EXPECTED BY PARTED MODULE>"
  raid_config: "<COPY OF NODE's TARGET_RAID_CONFIG FIELD>"
```

`ironic.nodes` List of dictionaries (currently of only one element) that will be used by `add-ironic-nodes.yaml` play to populate in-memory inventory. It also contains a copy of nodes `extra` field so you can access it in the playbooks. The Ansibles host is set to nodes UUID.

`ironic.image` All fields of nodes `instance_info` that start with `image_` are passed inside this variable. Some extra notes and fields:

- `mem_req` is calculated from image size (if available) and config option `[ansible]extra_memory`.
- if `checksum` is not in the form `<hash-algo>:<hash-sum>`, hashing algorithm is assumed to be `md5` (default in Glance).
- `validate_certs` - boolean (`yes/no`) flag that turns validating image store SSL certificate on or off (default is yes). Governed by `[ansible]image_store_insecure` option in ironic configuration file.
- `cafile` - custom CA bundle to use for validating image store SSL certificate. Takes value of `[ansible]image_store_cafile` if that is defined. Currently is not used by default playbooks, as Ansible has no way to specify the custom CA bundle to use for single HTTPS actions, however you can use this value in your custom playbooks to for example upload and register this CA in the ramdisk at deploy time.
- `client_cert` - cert file for client-side SSL authentication. Takes value of `[ansible]image_store_certfile` option if defined. Currently is not used by default playbooks, however you can use this value in your custom playbooks.
- `client_key` - private key file for client-side SSL authentication. Takes value of `[ansible]image_store_keyfile` option if defined. Currently is not used by default playbooks, however you can use this value in your custom playbooks.

`ironic.partition_info.partitions` Optional. List of dictionaries defining partitions to create on the node in the form:

```
partitions:
- name: "<NAME OF PARTITION>"
  unit: "<UNITS FOR SIZE>"
  size: "<SIZE OF THE PARTITION>"
  type: "<primary|extended|logical>"
  align: "<ONE OF PARTED_SUPPORTED OPTIONS>"
  format: "<PARTITION TYPE TO SET>"
  flags:
    flag_name: "<bool>"
```

The driver will populate this list from `root_gb`, `swap_mb` and `ephemeral_gb` fields of `instance_info`. The driver will also prepend the `bios_grub`-labeled partition when deploying on GPT-labeled disk, and pre-create a 64 MiB partition for configdrive if it is set in `instance_info`.

Please read the documentation included in the `ironic_parted` modules source for more info on the module and its arguments.

ironic.partition_info.ephemeral_format Optional. Taken from `instance_info`, it defines file system to be created on the ephemeral partition. Defaults to the value of `[pxe]\default_ephemeral_format` option in ironic configuration file.

ironic.partition_info.preserve_ephemeral Optional. Taken from the `instance_info`, it specifies if the ephemeral partition must be preserved or rebuilt. Defaults to `no`.

ironic.raid_config Taken from the `target_raid_config` if not empty, it specifies the RAID configuration to apply.

As usual for Ansible playbooks, you also have access to standard Ansible facts discovered by `setup` module.

Included custom Ansible modules

The provided `playbooks_path/library` folder includes several custom Ansible modules used by default implementation of `deploy` and `prepare` roles. You can use these modules in your playbooks as well.

stream_url Streaming download from HTTP(S) source to the disk device directly, tries to be compatible with Ansibles `get_url` module in terms of module arguments. Due to the low level of such operation it is not idempotent.

ironic_parted creates partition tables and partitions with `parted` utility. Due to the low level of such operation it is not idempotent. Please read the documentation included in the modules source for more information about this module and its arguments. The name is chosen so that the `parted` module included in Ansible is not shadowed.

Ramdisk deploy

The ramdisk interface is intended to provide a mechanism to deploy an instance where the item to be deployed is in reality a ramdisk. It is documented separately, see *Booting a Ramdisk or an ISO*.

iSCSI deploy

Warning: This deploy interface is deprecated and will be removed in the Xena release cycle. Please use *direct deploy* instead.

With `iscsi` deploy interface, the deploy ramdisk publishes the nodes hard drive as an iSCSI share. The ironic-conductor then copies the image to this share. See *iSCSI deploy diagram* for a detailed explanation of how this deploy interface works.

This interface is used by default, if enabled (see *Enabling hardware interfaces*). You can specify it explicitly when creating or updating a node:

```
baremetal node create --driver ipmi --deploy-interface iscsi
baremetal node set <NODE> --deploy-interface iscsi
```

5.1.2 Hardware Types

iBMC driver

Overview

The `ibmc` driver is targeted for Huawei V5 series rack server such as 2288H V5, CH121 V5. The iBMC hardware type enables the user to take advantage of features of Huawei iBMC to control Huawei server.

The `ibmc` hardware type supports the following Ironic interfaces:

- Management Interface: Boot device management
- Power Interface: Power management
- *RAID Interface*: RAID controller and disk management
- *Vendor Interface*: ibmc passthru interfaces

Prerequisites

The HUAWEI iBMC Client library should be installed on the ironic conductor node(s).

For example, it can be installed with `pip`:

```
sudo pip install python-ibmcclient
```

Enabling the iBMC driver

1. Add `ibmc` to the list of `enabled_hardware_types`, `enabled_power_interfaces`, `enabled_vendor_interfaces` and `enabled_management_interfaces` in `/etc/ironic/ironic.conf`. For example:

   ```
   [DEFAULT]
   ...
   enabled_hardware_types = ibmc
   enabled_power_interfaces = ibmc
   enabled_management_interfaces = ibmc
   enabled_raid_interfaces = ibmc
   enabled_vendor_interfaces = ibmc
   ```

2. Restart the ironic conductor service:

   ```
   sudo service ironic-conductor restart

   # Or, for RDO:
   sudo systemctl restart openstack-ironic-conductor
   ```

Registering a node with the iBMC driver

Nodes configured to use the driver should have the `driver` property set to `ibmc`.

The following properties are specified in the nodes `driver_info` field:

- `ibmc_address`:

  The URL address to the ibmc controller. It must include the authority portion of the URL, and can optionally include the scheme. If the scheme is missing, https is assumed. For example: https://ibmc.example.com. This is required.

- `ibmc_username`:

  User account with admin/server-profile access privilege. This is required.

- `ibmc_password`:

  User account password. This is required.

- `ibmc_verify_ca`:

  If ibmc_address has the **https** scheme, the driver will use a secure (TLS) connection when talking to the ibmc controller. By default (if this is set to True), the driver will try to verify the host certificates. This can be set to the path of a certificate file or directory with trusted certificates that the driver will use for verification. To disable verifying TLS, set this to False. This is optional.

The `baremetal node create` command can be used to enroll a node with the `ibmc` driver. For example:

```
baremetal node create --driver ibmc
  --driver-info ibmc_address=https://example.com \
  --driver-info ibmc_username=admin \
  --driver-info ibmc_password=password
```

For more information about enrolling nodes see *Enrollment* in the install guide.

RAID Interface

Currently, only RAID controller which supports OOB management can be managed.

See *RAID Configuration* for more information on Ironic RAID support.

The following properties are supported by the iBMC raid interface implementation, `ibmc`:

Mandatory properties

- `size_gb`: Size in gigabytes (integer) for the logical disk. Use `MAX` as `size_gb` if this logical disk is supposed to use the rest of the space available.
- `raid_level`: RAID level for the logical disk. Valid values are `JBOD`, `0`, `1`, `5`, `6`, `1+0`, `5+0` and `6+0`. And it is possible that some RAID controllers can only support a subset RAID levels.

Note: RAID level `2` is not supported by `iBMC` driver.

Optional properties

- `is_root_volume`: Optional. Specifies whether this disk is a root volume. By default, this is `False`.
- `volume_name`: Optional. Name of the volume to be created. If this is not specified, it will be N/A.

Backing physical disk hints

See *RAID Configuration* for more information on backing disk hints.

These are machine-independent properties. The hints are specified for each logical disk to help Ironic find the desired disks for RAID configuration.

- `share_physical_disks`
- `disk_type`
- `interface_type`
- `number_of_physical_disks`

Backing physical disks

These are HUAWEI RAID controller dependent properties:

- `controller`: Optional. Supported values are: RAID storage id, RAID storage name or RAID controller name. If a bare metal server have more than one controller, this is mandatory. Typical values would look like:
  - RAID Storage Id: `RAIDStorage0`
  - RAID Storage Name: `RAIDStorage0`
  - RAID Controller Name: `RAID Card1 Controller`.
- `physical_disks`: Optional. Supported values are: disk-id, disk-name or disk serial number. Typical values for hdd disk would look like:
  - Disk Id: `HDDPlaneDisk0`
  - Disk Name: `Disk0`.
  - Disk SerialNumber: `38DGK77LF77D`

Delete RAID configuration

For `delete_configuration` step, `ibmc` will do:

- delete all logical disks
- delete all hot-spare disks

Logical disks creation priority

Logical Disks creation priority based on three properties:

- `share_physical_disks`
- `physical_disks`
- `size_gb`

The logical disks creation priority strictly follow the table below, if multiple logical disks have the same priority, then they will be created with the same order in `logical_disks` array.

Share physical disks	Specified Physical Disks	Size
no	yes	int|max
no	no	int
yes	yes	int
yes	yes	max
yes	no	int
yes	no	max
no	no	max

Physical disks choice strategy

Note: physical-disk-group: a group of physical disks which have been used by some logical-disks with same RAID level.

- If no `physical_disks` are specified, the waste least strategy will be used to choose the physical disks.
  - waste least disk capacity: when using disks with different capacity, it will cause a waste of disk capacity. This is to avoid with highest priority.
  - using least total disk capacity: for example, we can create 400G RAID 5 with both 5 100G-disks and 3 200G-disks. 5 100G disks is a better strategy because it uses a 500G capacity totally. While 3 200G-disks are 600G totally.
  - using least disk count: finally, if waste capacity and total disk capacity are both the same (it rarely happens?), we will choose the one with the minimum number of disks.
- when `share_physical_disks` option is present, `ibmc` driver will create logical disk upon existing physical-disk-group list first. Only when no existing physical-disk-group matches, then it chooses unused physical disks with same strategy described above. When multiple exists physical-disk-groups matches, it will use waste least strategy too, the bigger capacity left the better. For example, to create a logical disk shown below on a `ibmc` server which has two RAID5 logical disks already. And the shareable capacity of this two logical-disks are 500G and 300G, then `ibmc` driver will choose the second one.

```
{
  "logical_disks": [
      {
          "controller": "RAID Card1 Controller",
```

(continues on next page)

(continued from previous page)

```
            "raid_level": "5",
            "size_gb": 100,
            "share_physical_disks": true
        }
    ]
}
```

And the `ibmc` server has two RAID5 logical disks already.

- When `size_gb` is set to `MAX`, `ibmc` driver will auto work through all possible cases and choose the best solution which has the biggest capacity and use least capacity. For example: to create a RAID 5+0 logical disk with MAX size in a server has 9 200G-disks, it will finally choose 8 disks + span-number 2 but not 9 disks + span-number 3. Although they both have 1200G capacity totally, but the former uses only 8 disks and the latter uses 9 disks. If you want to choose the latter solution, you can specified the disk count to use by adding `number_of_physical_disks` option.

```
{
   "logical_disks": [
       {
           "controller": "RAID Card1 Controller",
           "raid_level": "5+0",
           "size_gb": "MAX"
       }
   ]
}
```

Examples

In a typical scenario we may want to create:

- RAID 5, 500G, root OS volume with 3 disks
- RAID 5, rest available space, data volume with rest disks

```
{
  "logical_disks": [
      {
          "volume_name": "os_volume",
          "controller": "RAID Card1 Controller",
          "is_root_volume": "True",
          "physical_disks": [
              "Disk0",
              "Disk1",
              "Disk2"
          ],
          "raid_level": "5",
          "size_gb": "500"
      },
      {
          "volume_name": "data_volume",
          "controller": "RAID Card1 Controller",
          "raid_level": "5",
          "size_gb": "MAX"
```

(continues on next page)

(continued from previous page)

```
        }
    ]
}
```

Vendor Interface

The `ibmc` hardware type provides vendor passthru interfaces shown below:

Method Name	HTTP Method	Description
boot_up_seq	GET	Query boot up sequence
get_raid_controller_list	GET	Query RAID controller summary info

PXE Boot and iSCSI Deploy Process with Ironic Standalone Environment

iDRAC driver

Overview

col and the standard Distributed Management Task Force (DMTF) Redfish protocol to perform all of its functions.

Ironic Features

Prerequisites

Enabling

and match WSMAN and Redfish interfaces.

Interface	Supported Implementations
`bios`	`idrac-wsman`, `idrac-redfish`, `no-bios`
`boot`	`ipxe`, `pxe`, `idrac-redfish-virtual-media`
`console`	`no-console`
`deploy`	`iscsi`, `direct`, `ansible`, `ramdisk`
`inspect`	`idrac-wsman`, `idrac`, `idrac-redfish`, `inspector`, `no-inspect`
`management`	`idrac-wsman`, `idrac`, `idrac-redfish`
`network`	`flat`, `neutron`, `noop`
`power`	`idrac-wsman`, `idrac`, `idrac-redfish`
`raid`	`idrac-wsman`, `idrac`, `idrac-redfish`, `no-raid`
`rescue`	`no-rescue`, `agent`
`storage`	`noop`, `cinder`, `external`
`vendor`	`idrac-wsman`, `idrac`, `idrac-redfish`, `no-vendor`

Note: `idrac` is the legacy name of the WSMAN interface. It has been deprecated in favor of `idrac-wsman` and may be removed in a future release.

Protocol-specific Properties

interfaces, where some use WSMAN and others use Redfish, both the WSMAN and Redfish properties must be supplied.

Enrolling

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Note: If using WSMAN for the management interface, then WSMAN must be used for the power interface. The same applies to Redfish. It is currently not possible to use Redfish for one and WSMAN for the other.

BIOS Interface

Example

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Inspect Interface

is performed using the Dell WSMAN or Redfish protocol directly without affecting the operation of the system being inspected.

rently set `pxe_enabled` on the ports. The user should ensure that `pxe_enabled` is set correctly on the ports following inspection with the `idrac-redfish` inspect interface.

Management Interface

Import and export configuration

and available options see *Storage setup*.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```

```

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

SCP and its capabilities, see SCP_Reference_Guide.

directly from the storage location.

Storage setup

Note: Use of TLS is strongly advised.

Swift configuration

HTTP configuration

RAID Interface

Note: When using `idrac-redfish` for RAID interface iDRAC firmware greater than 4.40.00.00 is required.

Mandatory properties

Note: `JBOD` and `2` are not supported, and will fail with reason: Cannot calculate spans for RAID level.

Optional properties

Backing physical disk hints

Backing physical disks

Note: `physical_disks` is a mandatory parameter if the property `size_gb` is set to `MAX`.

Examples

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Manual RAID Invocation

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Vendor Interface

idrac-wsman

Method Name	HTTP Method	Description
`abandon_bios_config`	`DELETE`	Abandon a BIOS configuration job.
`commit_bios_config`	`POST`	Commit a BIOS configuration job submitted through `set_bios_config`. Required argument: `reboot` - indicates whether a reboot job should be automatically created with the config job. Returns a dictionary containing the `job_id` key with the ID of the newly created config job, and the `reboot_required` key indicating whether the node needs to be rebooted to execute the config job.
`get_bios_config`	`GET`	Returns a dictionary containing the nodes BIOS settings.
`list_unfinished_jobs`	`GET`	Returns a dictionary containing the key `unfinished_jobs`; its value is a list of dictionaries. Each dictionary represents an unfinished config job object.
`set_bios_config`	`POST`	Change the BIOS configuration on a node. Required argument: a dictionary of `{AttributeName: NewValue}`. Returns a dictionary containing the `is_commit_required` key indicating whether `commit_bios_config` needs to be called to apply the changes and the `is_reboot_required` value indicating whether the server must also be rebooted. Possible values are `true` and `false`.

Examples

Get BIOS Config

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Set BIOS Config

(continues on next page)

(continued from previous page)

Commit BIOS Changes

ically by the `commit_bios_config` call. If the reboot argument is not supplied, the job is still created, however it remains in the `scheduled` state until a reboot is performed. The reboot can be initiated through the Ironic power API.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Abandon BIOS Changes

Change Boot Mode

(OS) software. The OS loader uses basic services provided by the system BIOS to locate and load OS modules into system memory. After booting the system, the BIOS and embedded management controllers execute system management algorithms, which monitor and optimize the condition of the underlying hardware. BIOS configuration settings enable fine-tuning of the performance, power management, and reliability features of the system.

agement tasks as a traditional BIOS. However, UEFI does change the interfaces and data structures the BIOS uses to interact with I/O device firmware and operating system software. The primary intent of UEFI is to eliminate shortcomings in the traditional BIOS environment, enabling system firmware to continue scaling with industry trends.

(continues on next page)

(continued from previous page)

idrac-redfish

Method Name	HTTP Method	Description
`eject_media`	POST	Eject a virtual media device. If no device is provided then all attached devices will be ejected. Optional argument: `boot_device` - the boot device to eject, either, `cd`, `dvd`, `usb` or `floppy`.

Known Issues

Nodes go into maintenance mode

`conf`.

PXE reset with factory_reset BIOS clean step

`clean_failed` state on the node or `deploy_failed` if you attempt to deploy a node after this step. For now, the only solution is for the operator to manually restore the PXE settings of the server for it to PXE boot again, properly. The problem is caused by the fact that with the `UEFI boot mode`, the `idrac` uses BIOS settings to manage PXE configuration. This is not the case with the `BIOS boot mode` where the PXE configuration is handled as a configuration job on the integrated NIC itself, independently of the BIOS settings.

WSMAN vendor passthru timeout

Timeout when powering off

out to 90 seconds by setting the retry count to 18 as follows:

Unable to mount remote share with iDRAC firmware 4.40.00.00 or newer

til then can adjust settings in iDRAC to use plug-in type HTML5. In iDRAC web UI go to Configuration -> Virtual Console and select Plug-in Type to HTML5.

iLO driver

Overview

`ilo` hardware type supports ProLiant Gen10 systems which have iLO 5 management engine. iLO5 conforms to Redfish API and hence hardware type `redfish` (see *Redfish driver*) is also an option for this kind of hardware but it lacks the iLO specific features.

Hardware type

driver). For information on how to enable the `ilo` and `ilo5` hardware type, see *Enabling hardware types*.

Note: Only HPE ProLiant Gen10 servers supports hardware type `redfish`.

Hardware interfaces

iPXE respectively for deployment(just like *PXE boot*). These interfaces do not require iLO Advanced license. They can be enabled by using the `[DEFAULT]enabled_boot_interfaces` option in `ironic.conf` as given below:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Node configuration

`UEFI.`

Note: If configuration values for `ca_file`, `client_port` and `client_timeout` are not provided in the `driver_info` of the node, the corresponding config variables defined under `[ilo]` section in ironic.conf will be used.

Prerequisites

sion.

Different configuration for ilo hardware type

Glance Configuration

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Web server configuration on conductor

(continues on next page)

(continued from previous page)

intermediate files. The default value for `use_web_server_for_images` is False.

Note: HTTPS is strongly recommended over HTTP web server configuration for security enhancement. The `ilo-virtual-media` boot interface will send the instances configdrive over an encrypted channel if web server is HTTPS enabled. However for `ilo-uefi-https` boot interface HTTPS webserver is mandatory as this interface only supports HTTPS URLs.

Enable driver

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Optional functionalities for the `ilo` hardware type

Boot mode support

(continues on next page)

(continued from previous page)

flavor doesnt contain `boot_mode` then nova scheduler will not consider boot mode as a placement criteria, hence user may get either a BIOS or UEFI machine that matches with user specified flavors.

and the deploy images `boot_iso` property in glance should contain the glance UUID of the boot ISO. For building boot ISO, add `iso` element to the diskimage-builder command to build the image. For example:

UEFI Secure Boot Support

`ilo-virtual-media` boot interface, it is recommended that `boot_iso` property for user image contains the glance UUID of the boot ISO. If `boot_iso` property is not updated in glance for the user image, it would create the `boot_iso` using bootloader from the deploy iso. This `boot_iso` will be able to boot the user image in UEFI secure boot environment only if the bootloader is signed and can validate digital signatures of user image kernel.

Boot for Linux on HP ProLiant servers for additional details.

Node Cleaning Support

Supported Automated Cleaning Operations

disks visible to SSA in Proliant servers only with the ramdisk created using diskimage-builder from Ocata release. By default, this step is disabled. See *Disk Erase Support* for more details.

priority should be updated in ironic.conf.

(continues on next page)

(continued from previous page)

Supported Manual Cleaning Operations

note that this operation cannot be performed using the `ilo-virtual-media` boot interface as it needs this type of advanced license already active to use virtual media to boot into to start cleaning operation. Virtual media is an advanced feature. If an advanced license is already active and the user wants to overwrite the current license key, for example in case of a multi-server activation key delivered with a flexible-quantity kit or after completing an Activation Key Agreement (AKA), then the driver can still be used for executing this cleaning step.

are: `ilo`, `cpld`, `power_pic`, `bios` and `chassis`. Please refer to below table for their commonly used descriptions.

more information on usage.

Node Deployment Customization

`chassis`. This step is part of `management` interface. Please refer to below table for their commonly used descriptions.

more information on usage.

Example of using deploy template with the Compute service

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```
```

`CUSTOM_HYPERTHREADING_ON` will be executed during the deployment of the scheduled node, causing Hyperthreading to be enabled in the nodes BIOS configuration.

Hardware Inspection Support

Note:

unable to get the disk size, it raises an error. This feature is available in proliantutils release version >= 2.2.0.

rameters are mandatory to be given in driver_info for SNMPv3 inspection:

fix the actual parameters and then re-inspect so that iLO can recompute the overall security status. If the all security params, whose `security_status` is `Risk`, have the `Ignore` field set to `True`, then iLO sets the overall security status value as `Ignored`. All the security params must have the `security_status` as `Ok` for the `overall_security_status` to have the value as `Ok`.

components so that firmware is updated for all the components using latest SPP (Service Provider Pack) ISO and then re-inspect to get the security status again.

is replaced in the capability name. Examples: {gpu_Nvidia_count: 1}, {gpu_0x102b_count: 1}.

otherwise) NICs for Gen8 and Gen9 servers and ironic ports are created for all of them. Inspection logs a warning if the node under inspection is Gen8 or Gen9.

rately and re-inspect to see the security status of the parameters.

(continues on next page)

(continued from previous page)

Swiftless deploy for intermediate images

the boot ISO. A local HTTP(S) web server on each conductor node needs to be configured. Please refer to *Web server configuration on conductor* for more information. The HTTPS web server needs to be enabled (instead of HTTP web server) in order to send management information and images in encrypted channel over HTTPS.

Note: This feature assumes that the user inputs are on Glance which uses swift as backend. If swift dependency has to be eliminated, please refer to *HTTP(S) Based Deploy Support* also.

Deploy Process

HTTP(S) Based Deploy Support

the bare metal nodes.

Deploy Process

Support for iLO driver with Standalone Ironic

Configuration

scribed in *Swiftless deploy for intermediate images*.

Deploy Process

Netboot with glance and swift

Localboot with glance and swift for partition images

Localboot with glance and swift

Netboot in swiftless deploy for intermediate images

Localboot in swiftless deploy for intermediate images

Netboot with HTTP(S) based deploy

Localboot with HTTP(S) based deploy

Netboot in standalone ironic

Localboot in standalone ironic

Activating iLO Advanced license as manual clean step

in the `manageable` state again. User can follow steps from *Manual cleaning* to initiate manual cleaning operation on a node.

(continues on next page)

(continued from previous page)

Attribute	Description
`interface`	Interface of clean step, here `management`
`step`	Name of clean step, here `activate_license`
`args`	Keyword-argument entry (<name>: <value>) being passed to clean step
`args.ilo_license_key`	iLO Advanced license key to activate enterprise features. This is mandatory.

Initiating firmware update as manual clean step

state again. A user can follow steps from *Manual cleaning* to initiate manual cleaning operation on a node.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Attribute	Description
`interface`	Interface of clean step, here `management`
`step`	Name of clean step, here `update_firmware`
`args`	Keyword-argument entry (<name>: <value>) being passed to clean step
`args.firmware_update_mode`	Mode (or mechanism) of out-of-band firmware update. Supported value is `ilo`. This is mandatory.
`args.firmware_images`	Ordered list of dictionaries of images to be flashed. This is mandatory.

(continues on next page)

(continued from previous page)

purpose, the account is generally `service` and the container is generally `ironic` and `ilo` driver uses a container named `ironic_ilo_container` for their own purpose.

processing error could happen during image download, image checksum verification or image extraction. The logic is to process each of the firmware files and update them on the devices only if all the files are processed successfully. If, during the update (uploading and flashing) process, an update fails, then the remaining updates, if any, in the list will be aborted. But it is recommended to triage and fix the failure and re-attempt the manual clean step `update_firmware` for the aborted `firmware_images`.

where things were left off or where things failed. You can then fix or work around and then try again. A common cause of update failure is HPE Secure Digital Signature check failure for the firmware image file.

(continues on next page)

(continued from previous page)

Smart Update Manager (SUM) based firmware update

on SUM based firmware update.

Note: `update_firmware_sum` clean step requires the agent ramdisk with `Proliant Hardware Manager` from the proliantutils version 2.5.0 or higher. See *DIB support for Proliant Hardware Manager* to create the agent ramdisk with `Proliant Hardware Manager`.

Note: `flash_firmware_sum` deploy step requires the agent ramdisk with `Proliant Hardware Manager` from the proliantutils version 2.9.5 or higher. See *DIB support for Proliant Hardware Manager* to create the agent ramdisk with `Proliant Hardware Manager`.

Attribute	Description
`interface`	Interface of the clean step, here `management`
`step`	Name of the clean step, here `update_firmware_sum`
`args`	Keyword-argument entry (<name>: <value>) being passed to the clean step

and their update status. The log object will be named with the following pattern:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

tion or image extraction. In case of a failure, check Ironic conductor logs carefully to see if there are any validation or firmware processing related errors which may help in root cause analysis or gaining an understanding of where things were left off or where things failed. You can then fix or work around and then try again.

Note: Refer Guidelines for SPP ISO for steps to get SPP (Service Pack for ProLiant) ISO.

RAID Support

scheduling:

DIB support for Proliant Hardware Manager

---

[1] *ironic-python-agent-builder*: https://docs.openstack.org/ironic-python-agent-builder/latest/install/index.html

Disk Erase Support

ported by SSA.

ssacli supported erase method. If Sanitize erase is not supported on the Smart Storage Controller the disks are erased using One-pass erase (overwrite with zeros).

band for more information on enabling/disabling a clean step.

Firmware based UEFI iSCSI boot from volume support

ing state so it wouldnt take much time setting the iSCSI target as persistent device.

formed using iPXE. See *Boot From Volume* for more details.

and `uefi` boot modes, the virtual media driver only supports uefi boot mode, and that attempting to use iscsi boot at the same time with a bios volume will result in an error.

BIOS configuration support

examples.

Note: Prior to the Stein release the user is required to reboot the node manually in order for the settings to take into effect. Starting with the Stein release, iLO drivers reboot the node after running clean steps related to the BIOS configuration. The BIOS settings are cached and the clean step is marked as success only if all the requested settings are applied without any failure. If application of any of the settings fails, the clean step is marked as failed and the settings are not cached.

Configuration

values are `Enabled`, `Disabled`.

devices. Allowed values are `Enabled`, `Disabled`.

Certificate based validation in iLO

grated Lights-Out Security Technology Brief. Use iLO hostname or IP address as a Common Name (CN) while generating Certificate Signing Request (CSR). Use the same value as *ilo_address* while enrolling node to Bare Metal service to avoid SSL certificate validation errors related to hostname mismatch.

Rescue mode support

Inject NMI support

(continues on next page)

(continued from previous page)

Note: This feature is supported on HPE ProLiant Gen9 servers and beyond.

Soft power operation support

Note: The configuration `[conductor]soft_power_off_timeout` is used as a default timeout value when no timeout is provided while invoking hard or soft power operations.

Note: Server POST state is used to track the power status of HPE ProLiant Gen9 servers and beyond.

Out of Band RAID Support

RAID Configuration for more information.

(continues on next page)

(continued from previous page)

Note: Supported raid levels for `ilo5` hardware type are: 0, 1, 5, 6, 10, 50, 60

IPv6 support

(continues on next page)

(continued from previous page)

Note: No configuration changes (in e.g. ironic.conf) are required in order to support IPv6.

Out of Band Sanitize Disk Erase Support

Note: In average 300GB HDD with default pattern overwrite would take approx. 9 hours and 300GB SSD with default pattern block would take approx. 30 seconds to complete the erase.

Out of Band One Button Secure Erase Support

ployment settings profiles. See HPE Gen10 Security Reference Guide for more information.

Note:

cess along with the credentials of the server, which needs to be regained by the administrator. The process can take up to a day or two to fully erase and reset all user data.

Note: Do not perform any iLO 5 configuration changes until this process is completed.

UEFI-HTTPS Boot support

for more information.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Layer 3 or DHCP-less ramdisk booting

Intel IPMI driver

Overview

figuring the CPU to run at 3 distinct operating points or profiles.

Config	Cores	Base Freq (GHz)
Base	24	2.4
Config 1	20	2.5
Config 2	16	2.7

devices.

Glossary

Enabling the IntelIPMI hardware type

(continues on next page)

(continued from previous page)

Registering a node with the IntelIPMI driver

(continues on next page)

(continued from previous page)

Features of the `intel-ipmi` hardware type

Intel SST-PP

spectively. The input value must be a string.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

IPMI driver

Overview

ling IPMI-enabled devices.

Glossary

Enabling the IPMI hardware type

(continues on next page)

(continued from previous page)

Registering a node with the IPMI driver

Note: It is highly recommend that you setup a username and password for your BMC.

(continues on next page)

(continued from previous page)

Advanced configuration

Single/Double bridging functionality

Note: A version of IPMItool higher or equal to 1.8.12 is required to use the bridging functionality.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Changing the version of the IPMI protocol

Cipher suites

cent versions (e.g. the one used in RHEL 8.2) are switching to suite 17.

(continues on next page)

(continued from previous page)

```
```

be used from the list provided (from last to first):

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Static boot order configuration

Vendor Differences

could be something as simple as different interpretation of the standard.

disk subsystem is requested **in UEFI mode**. This is contrary to BIOS mode where the same BMCs expect the selector to be a value of `0x08`.

tomatically be recorded in the `properties` field `vendor`. When this is set to a value of `supermicro`, Ironic will navigate the UEFI behavior difference enabling the UEFI to be requested with boot to disk.

(continues on next page)

(continued from previous page)

```
```

be sure to include the `chassis bootparam get 5` output value along with the `mc info` output from your BMC.

(continues on next page)

(continued from previous page)

iRMC driver

Overview

Prerequisites

Hardware Type

Hardware interfaces

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Node configuration

Configuration via `driver_info`

ule)

tack. When set to `False`, Ironic user must take enough care around infrastructure environment in terms of security. (e.g. make sure network between Ironic conductor and iRMC is secure) rectory. For iRMC to recongnize certification file, Ironic user must run `openssl rehash <path_to_dir>`.

be blank, otherwise python-scciclient will encounter an communication error. If you are using such old version python-scciclient, the `irmc_snmp_auth_password` and `irmc_snmp_priv_password` properties will be ignored. If you want to set passwords, please update python-scciclient to some newer version (>= 0.10.1).

Configuration via `properties`

Configuration via `ironic.conf`

Override `ironic.conf` configuration via `driver_info`

Optional functionalities for the `irmc` hardware type

UEFI Secure Boot Support

Node Cleaning Support

Supported Automated Cleaning Operations

priority `0`. Set its priority to a positive integer to enable it. The recommended value is `10`.

Boot from Remote Volume

of iRMC. It supports iSCSI and FibreChannel.

Configuration

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```
```

mand:

Supported hardware

Hardware Inspection Support

Note: SNMP requires being enabled in ServerViewő iRMC S4 Web Server(Network SettingsSNMP section).

Configuration

loaded from here.

Supported properties

Note:

state.

(continues on next page)

(continued from previous page)

RAID configuration Support

Note:

Configuration

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Note: RAID 1+0 and 5+0 in iRMC driver does not support property `physical_disks` in `target_raid_config` during create raid configuration yet. See following example:

(continues on next page)

(continued from previous page)

Supported properties

Note:

ated RAID on iRMC server.

BIOS configuration Support

Configuration

This supports following options: `true`, `false`.

Supported platforms

Redfish driver

Overview

Prerequisites

Enabling the Redfish driver

(continues on next page)

(continued from previous page)

Registering a node with the Redfish driver

erwise ironic will pick the only available ComputerSystem automatically. For example: /redfish/v1/Systems/1.

tificate file or directory with trusted certificates that the driver will use for verification. To disable verifying TLS, set this to False. This is optional.

(continues on next page)

(continued from previous page)

Boot mode support

Note: Boot mode management is the optional part of the Redfish specification. Not all Redfish-compliant BMCs might implement it. In that case it remains the responsibility of the operator to configure proper boot mode to their bare metal nodes.

UEFI secure boot

Out-Of-Band inspection

ramdisk.

Note: The `redfish` inspect interface relies on the optional parts of the Redfish specification. Not all Redfish-compliant BMCs might serve the required information, in which case bare metal node inspection will fail.

Note: The `local_gb` property cannot always be discovered, for example, when a node does not have local storage or the Redfish implementation does not support the required schema. In this case the property will be set to 0.

Virtual media boot

physical CD/DVD. The node can then boot from that virtual drive into the operating system residing on the image.

ages associated with the ironic node.

Note: iDRAC firmware before 4.40.10.00 (on Intel systems) and 6.00.00.00 (on AMD systems) requires a non-standard Redfish call to boot from virtual media. Consider upgrading to 6.00.00.00, otherwise you **must** use the `idrac` hardware type and the `idrac-redfish-virtual-media` boot interface with older iDRAC firmware instead. For simplicity Ironic restricts both AMD and Intel systems before firmware version 6.00.00.00. See *iDRAC driver* for more details on this hardware type.

serves the same purpose.

Pre-built ISO images

Note: OpenStack Image service (glance) image IDs and `file://` links are also accepted.

Configuring an ESP image

to get them.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Note: If you use an architecture other than x86-64, youll need to adjust the destination paths.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```
```

Note: Unlike in the script above, these paths are case-sensitive!

Virtual Media Ramdisk

on how to enable and configure it.

dance with the `ramdisk` deployment interface behavior, once booted the machine will have a `provision_state` of `ACTIVE`.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Layer 3 or DHCP-less ramdisk booting

Firmware update using manual cleaning

If a failure occurs, the cleaning step immediately fails which may result in some updates not being applied. If the node is placed into maintenance mode while a firmware update cleaning step is running that is performing multiple firmware updates, the update in progress will complete, and processing of the remaining updates will pause. When the node is taken out of maintenance mode, processing of the remaining updates will continue.

cating that the update was successful. This allows the BMC time to fully reset before further operations are carried out against it. To cause the cleaning step to wait after applying an update, an optional `wait` argument may be specified in the firmware image dictionary. The value of this argument indicates the number of seconds to wait following the update. If the `wait` argument is not specified, then this is equivalent to `wait 0`, meaning that it will not wait and immediately proceed with the next firmware update if there is one, or complete the cleaning step if not.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Attribute	Description
`interface`	Interface of the cleaning step. Must be `management` for firmware update
`step`	Name of cleaning step. Must be `update_firmware` for firmware update
`args`	Keyword-argument entry (<name>: <value>) being passed to cleaning step
`args.firmware_images`	Ordered list of dictionaries of firmware images to be applied

(continues on next page)

(continued from previous page)

Note: Only `http` and `https` URLs are currently supported in the `url` argument.

Note: At the present time, targets for the firmware update cannot be specified. In testing, the BMC applied the update to all applicable targets on the node. It is assumed that the BMC knows what components a given firmware image is applicable to.

lar tool on a server that has network access to the BMC, try downloading the firmware to verify that the URLs are correct and that the web server is configured properly.

step that installs two firmware updates. The first updates the BMC firmware followed by a five minute wait to allow the BMC time to start back up. The second updates the firmware on all applicable NICs.:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Note: Firmware updates may take some time to complete. If a firmware update cleaning step consistently times out, then consider performing fewer firmware updates in the cleaning step or increasing `clean_callback_timeout` in ironic.conf to increase the timeout value.

SNMP driver

Note: Unlike most of the other power interfaces, the SNMP power interface does not have a corresponding management interface. The SNMP hardware type uses the `noop` management interface instead.

List of supported devices

Manufacturer	Model	Supported?	Driver name
APC	AP7920	Yes	apc_masterswitch
APC	AP9606	Yes	apc_masterswitch
APC	AP9225	Yes	apc_masterswitchplus
APC	AP7155	Yes	apc_rackpdu
APC	AP7900	Yes	apc_rackpdu
APC	AP7901	Yes	apc_rackpdu
APC	AP7902	Yes	apc_rackpdu
APC	AP7911a	Yes	apc_rackpdu
APC	AP7921	Yes	apc_rackpdu
APC	AP7922	Yes	apc_rackpdu
APC	AP7930	Yes	apc_rackpdu
APC	AP7931	Yes	apc_rackpdu
APC	AP7932	Yes	apc_rackpdu
APC	AP7940	Yes	apc_rackpdu
APC	AP7941	Yes	apc_rackpdu
APC	AP7951	Yes	apc_rackpdu
APC	AP7960	Yes	apc_rackpdu
APC	AP7990	Yes	apc_rackpdu
APC	AP7998	Yes	apc_rackpdu
APC	AP8941	Yes	apc_rackpdu
APC	AP8953	Yes	apc_rackpdu
APC	AP8959	Yes	apc_rackpdu
APC	AP8961	Yes	apc_rackpdu
APC	AP8965	Yes	apc_rackpdu
Aten	all?	Yes	aten
CyberPower	all?	Untested	cyberpower
EatonPower	all?	Untested	eatonpower
Teltronix	all?	Yes	teltronix
BayTech	MRP27	Yes	baytech_mrp27

Software Requirements

Enabling the SNMP Hardware Type

below:

(continues on next page)

(continued from previous page)

Ironic Node Configuration

provided. In the latter case `md5` is the default.

ing message authentication. Default is `none` unless `snmp_priv_key` is provided. In the latter case `des` is the default.

(continues on next page)

(continued from previous page)

XClarity driver

Overview

Prerequisites

Enabling the XClarity driver

Registering a node with the XClarity driver

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Changing Hardware Interfaces

(continues on next page)

(continued from previous page)

```
```

Changing Hardware Type

not work:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Static boot order configuration

vice will not change the boot device for you, leaving the pre-configured boot order.

(continues on next page)

(continued from previous page)

Overview

of a ramdisk, the process of booting this ramdisk on the node.

Drivers

installation actions like setting up a bootloader for local boot support.

and doing any post-deploy actions.

figure the Image service for temporary URLs.

Requirements

Using proxies for image download

Overview

Steps to enable proxies

imum cached file size as images can be pretty big. If you have HTTPS enabled in swift (see swift deployment guide), it is possible to configure the proxy server to talk to swift via HTTPS to download the image, store it in the cache unencrypted and return it to the node via HTTPS again. Because the image will be stored unencrypted in the cache, this approach is recommended for images that do not contain sensitive information. Refer to your proxy servers documentation to complete this step.

cache entries for the same image, based on the query part of the URL (as it contains some query parameters that change each time it is regenerated).

when the URL is used for the image download. You can think of it as roughly the time needed for

IPA ramdisk to startup and begin download. This value is used to check if the swift temporary URL duration is large enough to let the image download begin. Also if temporary URL caching is enabled, this will determine if a cached entry will still be valid when the download starts. It is used only if `[glance]swift_temp_url_cache_enabled` is `True`.

the proxy server as the query in its URL will change. The value of this option must be greater than or equal to `[glance]swift_temp_url_expected_download_start_delay`.

Advanced configuration

Out-of-band vs. in-band power off on deploy

`driver_info` field and set the `deploy_forces_oob_reboot` parameter with the value of **True**. For example, the below command sets this configuration in a specific node:

(continues on next page)

(continued from previous page)

```

```

Overview

discovered ethernet MACs. Operators will have to manually delete the Bare Metal service ports for which physical media is not connected. This is required due to the bug 1405131.

Capabilities discovery

Support.

(continues on next page)

(continued from previous page)

```

```

In-band inspection

a wide range of hardware. In-band inspection is using the ironic-inspector project.

Overview

node.

Deploy Steps

dered by priority and executed on the node when the node is moved to the `deploying` state.

Order of execution

is used: Power, Management, Deploy, BIOS, and RAID interfaces.

Agent steps

In-band steps

Requesting steps

(continues on next page)

(continued from previous page)

```
```

Writing a Deploy Step

FAQ

What deploy step is running?

(continues on next page)

(continued from previous page)

Troubleshooting

node.

Deploy Templates

Deploy step format

(continues on next page)

(continued from previous page)

```
```

(continues on next page)

(continued from previous page)

Matching deploy templates

scheduling when the Bare Metal service is used with the Compute service.

of those templates will not be reflected in the nodes configuration unless it is redeployed or rebuilt. Similarly, if a node is rebuilt and the set of matching deploy templates has changed since the initial deployment, then the resulting configuration of the node may be different from the initial deployment.

Overriding default deploy steps

be executed with the specified priority and arguments. If the steps priority is zero, the step will not be executed.

Creating a deploy template via API

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Creating a deploy template via baremetal client

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Example of use with the Compute service

Note: The deploy steps used in this example are for example purposes only.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```
```

configuration.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

ployment.

Overview

Automated cleaning

time.

priority and executed on the node when the node is moved to `cleaning` state, if automated cleaning is enabled.

workload is assigned to the nodes). For a full understanding of all state transitions into cleaning, please see *Ironics State Machine*.

Enabling automated cleaning

Cleaning steps

lution order is used: Power, Management, Deploy, BIOS, and RAID interfaces.

Management Interface

Name	Details	Priority	Stoppable	Arguments
`clear_job_queue`	Clear the job queue.	0	no	
`known_good_state`	Reset the iDRAC, Clear the job queue.	0	no	
`reset_idrac`	Reset the iDRAC.	0	no	

Name	Details	Priority	Stoppable	Arguments
`clear_job_queue`	Clear iDRAC job queue.	0	no	
`clear_secure_boot_keys`	Clear all secure boot keys.	0	no	
`export_configuration`	Export the configuration of the server. Exports the configuration of the server against which the step is run and stores it in specific format in indicated location. Uses Dells Server Configuration Profile (SCP) from *sushy-oem-idrac* library to get ALL configuration for cloning. **param task** A task from TaskManager. **param export_configuration_location** URL of location to save the configuration to.	0	no	`export_configuration_location` (*required*) URL of location to save the configuration to.
`import_configuration`	Import and apply the configuration to the server. Gets pre-created configuration from storage by given location and imports that into given server. Uses Dells Server Configuration Profile (SCP). **param task** A task from TaskManager. **param import_configuration_location** URL of location to fetch desired configuration from.	0	no	`import_configuration_location` (*required*) URL of location to fetch desired configuration from.
`import_export_configuration`	Import and export configuration in one go. Gets pre-created configuration from storage by given name and imports that into given server. After that ex-	0	no	`export_configuration_location` (*required*) URL of location to save the configuration to. `import_configuration_location` (*required*) URL of location to fetch desired configura-

Name	Details	Priority	Stoppable	Arguments
`clear_job_queue`	Clear the job queue.	0	no	
`known_good_state`	Reset the iDRAC, Clear the job queue.	0	no	
`reset_idrac`	Reset the iDRAC.	0	no	

Name	Details	Priority	Stoppable	Arguments
`activate_license`	Activates iLO Advanced license.	0	no	`ilo_license_key` (*required*) The HPE iLO Advanced license key to activate enterprise features.
`clear_secure_boot_keys`	Clear all secure boot keys. Clears all the secure boot keys. This operation is supported only on HP Proliant Gen9 and above servers.	0	no	
`reset_bios_to_default`	Resets the BIOS settings to default values. Resets BIOS to default settings. This operation is currently supported only on HP Proliant Gen9 and above servers.	10	no	
`reset_ilo`	Resets the iLO.	0	no	
`reset_ilo_credential`	Resets the iLO password.	30	no	
`reset_secure_boot_keys_to_default`	Reset secure boot keys to manufacturing defaults. Resets the secure boot keys to manufacturing defaults. This operation is supported only on HP Proliant Gen9 and above servers.	20	no	
`update_firmware`	Updates the firmware.	0	no	`firmware_images` (*required*) This argument represents the ordered list of JSON dictionaries of firmware images. Each firmware image dictionary consists of three mandatory fields, namely url, checksum and component. These fields represent firmware image location URL, md5 checksum of image file and firmware component type respectively. The supported firmware URL schemes are file, http, https and swift. The supported values for firmware component are ilo, cpld, power_pic, bios and chassis. The firmware images will be applied (in the order given) one by one on the baremetal server. For more information, see https://docs.openstack.org/ironic/latest/admin/drivers/ilo.html#initiating-firmware-update-as-manual-clean-s

Name	Details	Priority	Stoppable	Arguments
`activate_license`	Activates iLO Advanced license.	0	no	`ilo_license_key` (*required*) The HPE iLO Advanced license key to activate enterprise features.
`clear_secure_boot_keys`	Clear all secure boot keys. Clears all the secure boot keys. This operation is supported only on HP Proliant Gen9 and above servers.	0	no	
`erase_devices`	Erase all the drives on the node. This method performs out-of-band sanitize disk erase on all the supported physical drives in the node. This erase cannot be performed on logical drives.	0	no	`erase_pattern` Dictionary of disk type and corresponding erase pattern to be used to perform specific out-of-band sanitize disk erase. Supported values are, for hdd: (overwrite, crypto, zero), for ssd: (block, crypto, zero). Default pattern is: {hdd: overwrite, ssd: block}.
`one_button_secure_erase`	Erase the whole system securely. The One-button secure erase process resets iLO and deletes all licenses stored there, resets BIOS settings, and deletes all Active Health System (AHS) and warranty data stored on the system. It also erases supported non-volatile storage data and deletes any deployment setting profiles.	0	no	
`reset_bios_to_default`	Resets the BIOS settings to default values. Resets BIOS to default settings. This operation is currently supported only on HP Proliant Gen9 and above servers.	10	no	
`reset_ilo`	Resets the iLO.	0	no	
`reset_ilo_credential`	Resets the iLO password.	30	no	
`reset_secure_boot_keys_to_default`	Reset secure boot keys to manufacturing defaults. Resets the secure boot keys to manufacturing defaults. This operation is supported only on HP Proliant Gen9 and above servers.	20	no	
`update_firmware`	Updates the firmware.	0	no	`firmware_images` (*required*) This argument represents the ordered list of JSON dictionaries of

Name	Details	Priority	Stoppable	Arguments
`restore_irmc_bios_config`	Restore BIOS config for a node.	0	no	

Name	Details	Priority	Stoppable	Arguments
`clear_secure_boot_keys`	Clear all secure boot keys.	0	no	
`reset_secure_boot_keys_to_default`	Reset secure boot keys to manufacturing defaults.	0	no	
`update_firmware`	Updates the firmware on the node.	0	no	`firmware_images` (*required*) A list of firmware images to apply.

Bios Interface

Name	Details	Priority	Stoppable	Arguments
`apply_configuration`	Apply the BIOS settings to the node.	0	no	`settings` (*required*) A list of BIOS settings to be applied
`factory_reset`	Reset the BIOS settings of the node to the factory default.	0	no	

Name	Details	Priority	Stoppable	Arguments
`apply_configuration`	Apply the BIOS configuration to the node **param task** a TaskManager instance containing the node to act on **param settings** List of BIOS settings to apply **raises** DRACOperationError upon an error from python-dracclient	0	no	`settings` (*required*) List of BIOS settings to apply
`factory_reset`	Reset the BIOS settings of the node to the factory default. This uses the Lifecycle Controller configuration to perform BIOS configuration reset. Leveraging the python-dracclient methods already available.	0	no	

Name	Details	Priority	Stoppable	Arguments
`apply_configuration`	Applies the provided configuration on the node.	0	no	`settings` (*required*) Dictionary with current BIOS configuration.
`factory_reset`	Reset the BIOS settings to factory configuration.	0	no	

Name	Details	Priority	Stoppable	Arguments
`apply_configuration`	Applies BIOS configuration on the given node. This method takes the BIOS settings from the settings param and applies BIOS configuration on the given node. After the BIOS configuration is done, self.cache_bios_settings() may be called to sync the nodes BIOS-related information with the BIOS configuration applied on the node. It will also validate the given settings before applying any settings and manage failures when setting an invalid BIOS config. In the case of needing password to update the BIOS config, it will be taken from the driver_info properties.	0	no	`settings` (*required*) Dictionary containing the BIOS configuration.

Name	Details	Priority	Stoppable	Arguments
`apply_configuration`	Apply the BIOS settings to the node.	0	no	`settings` (*required*) A list of BIOS settings to be applied
`factory_reset`	Reset the BIOS settings of the node to the factory default.	0	no	

Raid Interface

Name	Details	Priority	Stoppable	Arguments
`create_configuration`	Create a RAID configuration on a bare metal using agent ramdisk. This method creates a RAID configuration on the given node.	0	no	
`delete_configuration`	Deletes RAID configuration on the given node.	0	no	

Name	Details	Priority	Stoppable	Arguments
`create_configuration`	Create a RAID configuration. This method creates a RAID configuration on the given node.	0	no	`create_nonroot_volumes` This specifies whether to create the non-root volumes. Defaults to *True*. `create_root_volume` This specifies whether to create the root volume. Defaults to *True*. `delete_existing` Setting this to True indicates to delete existing RAID configuration prior to creating the new configuration. Default value is False.
`delete_configuration`	Delete the RAID configuration.	0	no	

Name	Details	Priority	Stoppable	Arguments
`create_configuration`	Create the RAID configuration. This method creates the RAID configuration on the given node.	0	no	`create_nonroot_volumes` This specifies whether to create the non-root volumes. Defaults to *True*. `create_root_volume` This specifies whether to create the root volume. Defaults to *True*. `delete_existing` Setting this to True indicates to delete existing RAID configuration prior to creating the new configuration. Default value is False.
`delete_configuration`	Delete the RAID configuration.	0	no	

Name	Details	Priority	Stoppable	Arguments
`create_configuration`	Create RAID configuration on the node. This method creates the RAID configuration as read from node.target_raid_config. This method by default will create all logical disks.	0	no	`create_nonroot_volumes` This specifies whether to create the non-root volumes. Defaults to *True*. `create_root_volume` This specifies whether to create the root volume. Defaults to *True*. `delete_existing` Setting this to *True* indicates to delete existing RAID configuration prior to creating the new configuration. Default value is *False*.
`delete_configuration`	Delete RAID configuration on the node.	0	no	

Name	Details	Priority	Stoppable	Arguments
`create_configuration`	Create the RAID configuration. This method creates the RAID configuration on the given node.	0	no	`create_nonroot_volumes` This specifies whether to create the non-root volumes. Defaults to *True*. `create_root_volume` This specifies whether to create the root volume. Defaults to *True*. `delete_existing` Setting this to True indicates to delete existing RAID configuration prior to creating the new configuration. Default value is False.
`delete_configuration`	Delete the RAID configuration.	0	no	

Name	Details	Priority	Stoppable	Arguments
`create_configuration`	Create a RAID configuration on a bare metal using agent ramdisk. This method creates a RAID configuration on the given node.	0	no	`create_nonroot_volumes` This specifies whether to create the non-root volumes. Defaults to *True*. `create_root_volume` This specifies whether to create the root volume. Defaults to *True*.
`delete_configuration`	Delete the RAID configuration.	0	no	

Name	Details	Priority	Stoppable	Arguments
`create_configuration`	Create the RAID configuration. This method creates the RAID configuration on the given node.	0	no	`create_nonroot_volumes` This specifies whether to create the non-root volumes. Defaults to *True*. `create_root_volume` This specifies whether to create the root volume.Defaults to *True*.
`delete_configuration`	Delete the RAID configuration.	0	no	

Name	Details	Priority	Stoppable	Arguments
`create_configuration`	Create RAID configuration on the node. This method creates the RAID configuration as read from node.target_raid_config. This method by default will create all logical disks.	0	no	`create_nonroot_volumes` This specifies whether to create the non-root volumes. Defaults to *True*. `create_root_volume` This specifies whether to create the root volume. Defaults to *True*. `delete_existing` Setting this to *True* indicates to delete existing RAID configuration prior to creating the new configuration. Default value is *False*.
`delete_configuration`	Delete RAID configuration on the node.	0	no	

Manual cleaning

ing a manual clean, the operator specifies the cleaning steps to be performed. Manual cleaning can only be performed when a node is in the `manageable` state. Once the manual cleaning is finished, the node will be put in the `manageable` state again.

Setup

Starting manual cleaning via API

A cleaning step is represented by a dictionary (JSON), in the form:

(continues on next page)

(continued from previous page)

ing <name>: <value>.

error message.

(continues on next page)

(continued from previous page)

Starting manual cleaning via openstack metal CLI

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Cleaning Network

tenant network. For steps to set up the cleaning network, please see *Configure the Bare Metal service for cleaning*.

In-band vs out-of-band

In-band

mal cleaning configuration, only erasing disks. However, you can add your own cleaning steps and/or override default cleaning steps with a custom Hardware Manager.

Out-of-band

and hardware itself.

FAQ

How are cleaning steps ordered?

olution order is used:

How do I skip a cleaning step?

How do I change the priority of a cleaning step?

the following configuration option:

(continues on next page)

(continued from previous page)

```
```

by using the `nvme-cli` utility. This behavior can be controlled using the following configuration option (by default it is set to True):

ations, use the following configuration option:

Overriding step priority

(continues on next page)

(continued from previous page)

What cleaning step is running?

(continues on next page)

(continued from previous page)

```
```

Should I disable automated cleaning?

time consuming process. To mitigate this, we suggest using NVMe drives with support for NVMe Secure Erase (based on `nvme-cli` format command) or ATA drives with support for cryptographic ATA Security Erase, as typically the erase_devices step in the deploy interface takes the longest time to complete of all cleaning steps.

Why cant I power on/off a node while its cleaning?

Troubleshooting

tor should validate that no permanent damage has been done to the node and no processes are still running on it before removing the maintenance mode.

Note: Older versions of ironic may put the node to maintenance even when no clean step has been running.

Overview

been deployed by another Bare Metal service installation or deployed via other means.

How it works

adopt a node.

ment of nodes for a conductor that has failed.

ISO image and then places any PXE or virtual media configuration necessary for the node should it be required.

should ensure that any supplied configuration defining the node is sufficient for the continued operation of the node moving forward. Such as, if the node is configured to network boot via instance_info/boot_option=netboot, then appropriate driver specific node configuration should be set to support this capability.

Possible Risk

pre-existing configuration.

is effectively wiped.

compatibility issues may exist as a result.

How to use

Note: The power state that the ironic-conductor observes upon the first successful power state check, as part of the transition to the `manageable` state will be enforced with a node that has been adopted. This means a node that is in `power off` state will, by default, have the power state enforced as `power off` moving forward, unless an administrator actively changes the power state using the Bare Metal service.

Requirements

to retrieve the pertinent files. Inability to do so will result in the adoption failing, and the node being placed in the `adopt failed` state.

Example

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Note: In the above example, the image_source setting must reference a valid image or file, however that image or file can ultimately be empty.

Note: The above example utilizes a capability that defines the boot operation to be local. It is recommended to define the node as such unless network booting is desired.

Note: The above example will fail a re-deployment as a fake image is defined and no instance_info/image_checksum value is defined. As such any actual attempt to write the image out will fail as the image_checksum value is only validated at time of an actual deployment operation.

Note: A user may wish to assign an instance_uuid to a node, which could be used to match an instance in the Compute service. Doing so is not required for the proper operation of the Bare Metal service.

Note: In Newton, coupled with API version 1.20, the concept of a network_interface was introduced. A user of this feature may wish to add new nodes with a network_interface of `noop` and then change the interface at a later point and time.

Troubleshooting

dation step. Validation steps are dependent upon what driver is selected for the node.

without cleaning occurring to preserve the nodes current state. Example:

Overview

the BMC, or tracking the dismantling of servers from their racks.

portant difference to nodes which have the `maintenance` flag set).

How to use

Note: An exception are nodes which are in `available`. For backwards compatibility reasons, these nodes need to be moved to `manageable` first. Trying to set the `retired` flag for `available` nodes will result in an error.

(continues on next page)

(continued from previous page)

needs to be removed first. This can be done via:

Overview

the bare metal during manual cleaning.

Prerequisites

with some caveats - see *Software RAID* for details.

Build agent ramdisk which supports RAID configuration

should be used for HPE Proliant Servers.

Note: For in-band software RAID, the agent ramdisk does not need to be bundled with a hardware manager as the generic hardware manager in the Ironic Python Agent already provides (basic) support for software RAID.

RAID configuration JSON format

Target RAID configuration

cleaning.

```

```

(continues on next page)

(continued from previous page)

Mandatory properties

disks are specified (see below).

Optional properties

pable of retrieving it. This is `false` by default.

Backing physical disk hints

ual details for each bare metal node. None of these options are supported for software RAID.

Backing physical disks

S.M.A.R.T. status, physical location). The values for these properties are hardware dependent.

peat the same hint if necessary).

Note: If properties from both Backing physical disk hints or Backing physical disks are specified, they should be consistent with each other. If they are not consistent, then the RAID configuration will fail (because the appropriate backing physical disks could not be found).

Examples for `target_raid_config`

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Current RAID configuration

ical disk after they were created on the bare metal node. It contains details like RAID controller used, the backing physical disks used, WWN of each logical disk, etc. It also contains information about each physical disk found on the bare metal node.

Workflow

mation.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Software RAID

a software RAID configuration example in *Examples for target_raid_config*.

ond one can be 0, 1, 1+0, 5, or 6. As the first RAID device will be the deployment device, enforcing a RAID-1 reduces the risk of ending up with a non-booting node in case of a disk failure.

(continues on next page)

(continued from previous page)

bedded in the images initrd).

the deployed user image. Depending on how the partitions are mounted, the content of the partitions may get out of sync, e.g. when new kernels are installed or the bootloader is updated, so measures to keep these partitions in sync need to be taken.

Image requirements

tem on the first partition. Starting with Ussuri, the image can also have additional metadata to point Ironic to the partition with the root file system: for this, the image needs to set the `rootfs_uuid` property with the file system UUID of the root file system. One way to extract this UUID from an existing image is to download the image, mount it as a loopback device, and use `blkid`:

(continues on next page)

(continued from previous page)

Using RAID in nova flavor for scheduling

Developer documentation

mation, see Ironic Python Agent Hardware Manager documentation.

set `node.raid_config`.

(continues on next page)

(continued from previous page)

Overview

ual cleaning.

Prerequisites

Enabling hardware types

Enabling hardware interface

Retrieve BIOS settings

`json` is added as suffix to above command, it returns BIOS settings as following:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Configure BIOS settings

Factory reset

(continues on next page)

(continued from previous page)

Apply BIOS configuration

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

tion is a dictionary with `name` and `value` keys.

Note: When applying BIOS settings to a node, vendor-specific driver may take the given BIOS settings from the argument and compare them with the current BIOS settings on the node and only apply when there is a difference.

Overview

configured appropriately.

Note: The rescue operation is currently supported only when tenant networks use DHCP to obtain IP addresses.

ample, if there is a need to perform manual password reset or data recovery in the event of some failure, rescue operation can be used.

Configuring The Bare Metal Service

deploy environments, but an example of how to do this is outlined below:

booting (the default). This can be the same network as your cleaning or tenant network (for flat network). For an example of how to configure new networks with Bare Metal Service, see the *Configure the Networking service for bare metal provisioning* documentation.

(continues on next page)

(continued from previous page)

Overview

duced.

How this works - From Ironics point of view

booted.

the integrated iSCSI initiators are to connect to the supplied volume target information.

as the baremetals boot from volume disk upon requesting the instance.

tion of standard network traffic and instance network traffic. In the iPXE case, this is not possible as the OS userspace re-configures the iSCSI connection after detection inside the OS ramdisk boot.

different interface. This is a common pattern in iSCSI based deployments in the physical realm.

Prerequisites

Conductor Configuration

Node Configuration

Storage Interface

iSCSI Configuration

Qualifying Name (IQN) that is unique to your SAN. For example, to create a volume connector for iSCSI:

(continues on next page)

(continued from previous page)

```

```

Image Creation

(continues on next page)

(continued from previous page)

```
```

Note:

Advanced Topics

Use without the Compute Service

them, it is not explicitly required, and can be performed manually.

(continues on next page)

(continued from previous page)

Note: A `boot-index` value of `0` represents the boot volume for a node. As the `boot-index` is per-node in sequential order, only one boot volume is permitted for each node.

Use Without Cinder

age interface which does not contain logic to determine if the node should or could boot from a remote volume.

scenario.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

may not support all forms of volume target configuration. As of the Rocky release, the bare metal service does not support writing an Operating System image to a remote boot from volume target, so that also must be ensured by the user in advance.

Cinder Multi-attach

Compute service, as of the Pike release, does not yet have support to leverage multi-attach. Concurrently, multi-attach requires the backend volume driver running as part of the Block Storage service to contain support for multi-attach volumes.

tested until there is Compute service integration as well as volume driver support.

use of volumes that are being reported as `in-use` if they do not explicitly support multi-attach.

Overview

to allow provisioning of nodes in a separate provisioning network. The result of this is that multiple tenants can use nodes in an isolated fashion. However, this configuration does not support trunk ports belonging to multiple networks.

Concepts

Network interfaces

life cycle. This interface requires Networking service support for the switches attached to the baremetal servers so they can be programmed.

Local link connection

the information to plug the specified port to the tenant network.

Field	Description
`switch_id`	Required. Identifies a switch and can be a MAC address or an OpenFlow-based `datapath_id`.
`port_id`	Required. Port ID on the switch/Smart NIC, for example, Gig0/1, rep0-0.
`switch_info`	Optional. Used to distinguish different switch models or other vendor-specific identifier. Some ML2 plugins may require this field.
`hostname`	Required in case of a Smart NIC port. Hostname of Smart NIC device.

Note: This isnt applicable to Infiniband ports because the network topology is discoverable by the Infiniband Subnet Manager. If specified, local_link_connection information will be ignored. If port is Smart NIC port then:

Physical networks

port groups in the Bare Metal service. A ports physical network field is optional, and if not set then any virtual port may be mapped to that port, provided that no free Bare Metal port with a suitable physical network assignment exists.

Configuring the Bare Metal service

Configuring nodes

vices ML2 driver:

(continues on next page)

(continued from previous page)

cess for deriving the ports MAC address ($HW_MAC_ADDRESS); it is vendor specific. For example, Mellanox ConnectX Family Devices prefix is ff:00:00:00:00:00:02:00:00:02:c9:00. If port GUID was f4:52:14:03:00:38:39:81 the client ID would be ff:00:00:00:00:00:02:00:00:02:c9:00:f4:52:14:03:00:38:39:81. Mellanox ConnectX Family Devices HW_MAC_ADDRESS consists of 6 bytes; the port GUIDs lower 3 and higher 3 bytes. In this example it would be f4:52:14:38:39:81. Putting it all together, create an Infiniband port as follows:

(continues on next page)

(continued from previous page)

Configuring the Networking service

determined by the Bare Metal service network interfaces you have enabled and which top of rack switches you have in your environment.

`flat` network interface

This driver understands that the switch should be already configured by the admin, and will mark the networking service ports as successfully bound as nothing else needs to be done.

`neutron` network interface

of rack switch in the environment must be installed and enabled.

below describe how to make use of them in the Bare Metal service.

Switch-side configuration

ties that will be configured on the ironic side, as bonding mode and properties may be named differently on your switch, or have possible values different from the ones described in kernel documentation on bonding. Please refer to your switch configuration documentation for more details.

bers to be used by themselves, you need to set port groups `standalone_ports_supported` value to be `False` in ironic, as it is `True` by default.

Physical networks

Port groups configuration in the Bare Metal service

not be configured.

how to configure bonding via configdrive, refer to cloud-init documentation and code. cloud-init version 0.7.7 or later is required for bonding configuration to work.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Port groups that dont have any ports will be ignored.

service configuration file.

Link aggregation/teaming on windows

transmit hash policy, MII link monitoring interval, and of which links the bond consists. The information in InstanceMetadata will be used afterwards to generate the config drive.

Overview

Node web console

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```
↪
```

(continues on next page)

↪

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

```
↪---------------------------------+
```

Node serial console

(continues on next page)

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

```
↪----------------------------------+
```

to achieve that, you need to follow the documentation for Serial Console from the Compute service.

Configuring HA

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

billing or usage system, a monitoring data store, or other OpenStack services. This page describes how to enable notifications and the different kinds of notifications that ironic may emit. The external consumer will see notifications emitted by ironic as JSON objects structured in the following manner:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Configuration

fications are emitted. For example, if the option is set to warning, all notifications with priority level warning, error, or critical are emitted, but not notifications with priority level debug or info. For information about the semantics of each log level, see the OpenStack logging standards[1]. If this option is unset, no notifications will be emitted. The priority level of each available notification is documented below.

---

[1] https://wiki.openstack.org/wiki/LoggingStandards#Log_level_definitions

mation, see the documentation of your chosen message bus, such as the RabbitMQ documentation[2].

Versioning

---

[2] https://www.rabbitmq.com/documentation.html

Available notifications

ironic-api notifications

Resources CRUD notifications

that is emitted at ERROR level.

---

[3] https://en.wikipedia.org/wiki/Create,_read,_update_and_delete

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```

```

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Node maintenance notifications

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

ironic-conductor notifications

Node console notifications

takes over a node that was being managed by another ironic-conductor. start and end notifications have INFO level, error has ERROR. Example of node console notification:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

baremetal.node.power_set

or when it fails to set the power state if a change is requested.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

baremetal.node.power_state_corrected

tion level info.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```
```

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

baremetal.node.provision_set

triggered the state change:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Overview

is only known to the deployer and operator of the infrastructure.

New York City.

How it works

ating conductor and as such if a conductor has a `[conductor]conductor_group` configuration option defined in its *ironic.conf* configuration file, the conductor will then be limited to only managing nodes with a matching `conductor_group` string.

Note: Any conductor without a `[conductor]conductor_group` setting will only manage baremetal nodes without a `conductor_group` value set upon node creation. If no such conductor is present when conductor groups are configured, node creation will fail unless a `conductor_group` is specified upon node creation.

How to use

vice should be aware of. It is not intended as a How-To guide for securing a data center or an OpenStack deployment.

Image Checksums

checksums, and those deployment interfaces are for specific use cases which Ironic users leverage, outside of the general use case capabilities provided by the `direct` deployment interface.

Note: Use of the node `instance_info/image_checksum` field is discouraged for integrated OpenStack Users as usage of the matching Glance Image Service field is also deprecated. That being said, Ironic retains this feature by popular demand while also enabling also retain simplified operator interaction. The newer field values supported by Glance are also specifically supported by Ironic as `instance_info/image_os_hash_value` for checksum values and `instance_info/image_os_hash_algo` field for the checksum algorithm.

disable_support_for_checksum_files setting.

REST API: user roles and policy settings

and responsibilities through the roles. Please see *Secure RBAC*.

`driver_info` unmasked for users with administrative privileges, apply following changes to policy configuration file:

as described above.

Multi-tenancy

affect the next tenant.

Network Interactions

tity, Compute, and Networking services, so as to provide tenant-network isolation. Additional documentation on network multi-tenancy is available.

Lingering Effects

not erased between uses).

the utility ramdisk used during the cleaning phase. See details in the *Firmware security* section.

Firmware security

administrative access to the underlying hardware.

deleting their instance and allowing the server to be allocated to another user.

ever, the service does not ship with any code that will validate the integrity of, or make any modifications to, system or device firmware or firmware settings.

form the specific actions necessary within that environment to ensure the integrity of each servers firmware.

UEFI secure boot mode

Compatible images

Enabling with OpenStack Compute

the user regarding secure boot. If the flavor doesnt contain `secure_boot` then nova scheduler will not consider secure boot mode as a placement criteria, hence user may get a secure boot capable machine that matches with user specified flavors but deployment would not use its secure boot capability. Secure boot deploy would happen only when it is explicitly specified through flavor.

Enabling standalone

(continues on next page)

(continued from previous page)

Other considerations

Internal networks

Management interface technologies

protocol is not secure. If IPMI is enabled, in most cases a local OS administrator is able to work in-band with IPMI settings without specifying any credentials, as this is a DCMI specification requirement.

Tenant network isolation

API endpoints for RAM disk use

Disk Images

passed through `qemu-img`. When in doubt, use a `raw` image which you can verify is in the desirable and expected state.

should not be utilized. This check, by default, occurs only through images which transverse *through* the conductor.

along with network traffic to facilitate the transfer. This check is enabled by default on Zed releases and earlier, but can be disabled using the `[conductor]conductor_always_validates_images` configuration option.

ies all disk images through the conductor. This setting is also available in the node `driver_info` and `instance_info` fields.

Mitigating Factors to disk images

a System scoped Member, or Project scoped Owner-Member, or a Project scoped Lessee-Admin via the `baremetal:node:update_instance_info` policy permission rule. Before the Wallaby release of OpenStack, this was restricted to `admin` and `baremetal_admin` roles and remains similarly restrictive in the newer Secure RBAC model. »»»> 8491abb92 (Harden all image handling and conversion code)

Nova returns No valid host was found Error

nance mode:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

source of the failures, then re-enable it:

(continues on next page)

(continued from previous page)

request will result in a No valid host was found error. It is hence sensible to check if Placement is aware of resource providers (nodes) for the requested resource class with:

(continues on next page)

(continued from previous page)

```
↪-----------------+--------------------------------+
```

(continues on next page)

(continued from previous page)

```
↪----------------+-------------------------------+
```

(continues on next page)

(continued from previous page)

```
↪-----------------+--------------------------------+
```

ported this provider to placement. Potential explanations include:

`memory_mb` and `local_gb`. Example of valid properties:

(continues on next page)

(continued from previous page)

```
↪                                              |
```

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

ror messages in Ironic conductor log, it means the conductor run into a special error during deployment. So you can check the log carefully to fix or work around and then try again.

Patching the Deploy Ramdisk

youve built your ramdisk). But its also possible to quickly modify an already built ramdisk.

Note: On a systemd-based system you can use the `systemd-nspawn` tool (from the `systemd-container` package) to create a lightweight container from the unpacked filesystem tree:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Note: You dont need to modify the kernel (e.g. `tinyipa-master.vmlinuz`), only the ramdisk part.

API Errors

Retrieving logs from the deploy ramdisk

deploy ramdisk when the deployment fails and save it on the local filesystem at `/var/log/ironic/deploy`.

(continues on next page)

(continued from previous page)

Note: The *instance_uuid* field is not required for deploying a node when Ironic is configured to be used in standalone mode. If present it will be appended to the name.

Accessing the log data

When storing in the local filesystem

Note: When saving the logs to the filesystem, operators may want to enable some form of rotation for the logs to avoid disk space problems.

When storing in Swift

(continues on next page)

(continued from previous page)

The contents of the log file

DHCP during PXE or iPXE is inconsistent or unreliable

this issue you should set the switch port that connects to your baremetal nodes as an edge or PortFast type port. Configured in this way the switch port will move to forwarding mode as soon as the link is established. An example on how to do that for a Cisco Nexus switch is:

Why does X issue occur when I am using LACP bonding with iPXE?

this for only the single port which is used for network booting.

iPXE has stopped responding to LACP messages from the peer port, which occurs as part of the process of booting a ramdisk and iPXE handing over control to a full operating-system, switches typically begin a timer to determine how to handle the failure. This is because, depending on the mode of LACP, this can be interpreted as a switch or network fabric failure.

unavailable in introspection.

attempt to navigate the worst-known default hold-down timers to help ensure a deployment does not fail due to a short-lived transitory network connectivity failure in the form of a switch port having moved to a temporary blocking state. Where applicable and possible, many of these patches have been backported to supported releases, however users of the iSCSI deployment interface will see the least capability for these sorts of situations to be handled automatically. These patches also require that the switchport has an eventual fallback to a non-bonded mode. If the port remains in a blocking state, then traffic will be unable to flow and the deloyment is likely to time out.

establish the Link Aggregate. This is instead of being treated as if its possibly another switch.

IPMI errors

Enable IPMI over LAN

Troubleshooting lanplus interface

(continues on next page)

(continued from previous page)

Why are my nodes stuck in a -ing state?

environment and operating configuration.

What can cause these sorts of failures?

can use the iostat tool to identify the percentage of CPU time spent waiting on storage devices.

settings, cause threads to be stuck in a blocking wait state, which is realistically undetectable short the operating system logging connectivity errors or even lock manager access errors.

ure, is when an `ls /path/to/nfs` command hangs for a period of time. In such cases, the Storage Administrator should be consulted and network connectivity investigated for errors before trying to recover to proceed.

The bad news for IO related failures

Note: Ironics conductor, upon restart, clears reservations for nodes which were previously managed by the conductor before restart.

the state of an IO failure, again dependent upon site and server configuration.

File Size != Disk Size

spends in `deploying` and `deploy wait` states.

ence issues here as the conductor will cache the image to be written which takes place when the `[agent]image_download_source` is set to `http` instead of `swift`.

device.

Note: The QCOW2 image conversion utility does consume quite a bit of memory when converting images or writing them to the end storage device. This is because the files are not sequential in nature, and must be re-assembled from an internal block mapping. Internally Ironic limits this to 1GB of RAM. Operators performing large numbers of deployments may wish to explore the `direct` deployment interface in these sorts of cases in order to minimize the conductor becoming a limiting factor due to memory and network IO.

Why are my nodes stuck in a wait state?

conductor will time out and the node will eventually move to a `failed` state. Depending on the configuration and the circumstances, however, a node can stay in a `wait` state for a long time or even never time out. The list of such wait states includes:

Communication issues between the conductor and the node

call back. Examples include wrong ciphers which will make ipmitool get stuck or BMCs in a state where they accept commands, but dont do the requested task (or only a part of it, like shutting off, but not starting). It is useful in these cases to see via a ping or the console if and which action the node is performing. If the node does not seem to react to the requests sent be the conductor, it may be worthwhile to try the corresponding action out-of-band, e.g. confirm that power on/off commands work when directly sent to the BMC. The section on *IPMI errors*. above gives some additional points to check. In some situations, a BMC reset may be necessary.

Ironic Python Agent stuck

be helpful to connect to the IPA and inspect its logs, see the trouble shooting guide of the ironic-python-agent (IPA) on how to do this.

Deployments fail with failed to update MAC address

ately reply.

unexpected glitch, and a previous entry is still present in Neutron.

How did I get here?

in the Bare Metal service.

sult in the VIF not being cleaned up from Neutron.

they are transitory from cleaning, provisioning, rescuing, or even inspection, getting the node to the `available` state wil unblock your delete operation, that is unless there is a tenant VIF attahment. In that case, the vif will need to be removed from with-in the Bare Metal service using the `baremetal node vif detach` command.

How do I resolve this?

inventory typo, or possibly even a duplicate MAC address exists, which could also produce the same basic error message.

My test VM image does not deploy mount point does not exist

What is likely occuring

a Linux OS image

unexpected internal structure.

networking and possibly installing user keys. Unfortunately, these images often lack drivers and firmware required for many different types of physical hardware which makes using them very problematic. Additionally, images such as Cirros do not have any contents in the root filesystem (i.e. an empty filesystem), as they are designed for the `ramdisk` to write the contents to disk upon boot.

How do I not encounter this issue?

of these published cloud images, also support auto-configuration of networking AND population of user keys.

Issues with autoconfigured TLS

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```

```

Clock skew

with the timestamp of a log message.

Since ironic-python-agent is likely operating in UTC, the hardware clock should also be set in UTC.

Note: Microsoft Windows uses local time by default, so a machine that has previously run Windows will likely have wrong time.

I changed ironic.conf, and now I cant edit my nodes.

abled interfaces list. The result of this is that the `ironic-conductor` cannot spawn a `task` using the composed driver, as a portion of the driver is no longer enabled. This makes it difficult to edit or update the node if the settings have been changed.

Example failure

How to fix this?

is encountered.:

Note: There are additional paths one can take to remedy this sort of issue, however we encourage operators to be mindful of operational consistency when making major configuration changes.

Im getting Out of Memory errors

complete failure of the machine. Unfortunately this can cause unpredictable behavior.

How did I get here?

der, requires a considerable amount of memory to efficently re-assemble and write-out a disk to a device, or to simply convert the format such as to a `raw` image.

ally memory pressure alone from buffers will not cause an out of memory condition, but the multiple conversions or deployments running at the same time CAN cause extreme memory pressure and risk the system running out of memory.

How do I resolve this?

parameters.

Ironic says my Image is Invalid

age service will necessitate uploading a new image as that property cannot be changed in the image service *after* creation of an image.

inherent risk that the image they are attempting to use may have a bad or malicious structure. Image safety checks are generally performed as the deployment process begins and stages artifacts, however a late stage check is performed when needed by the ironic-python-agent.

Baremetal Power Sync

forced on the hardware and if it is set to `false` the hardware state will be forced on the database. If this periodic task is enabled, it runs at an interval defined by the *`conductor.sync_power_state_interval`* config option for those nodes which are not in maintenance.

Compute-Baremetal Power Sync

the `nova-compute` process. In case of the compute driver being baremetal driver, this sync will happen between the databases of the compute and baremetal services. Since the sync happens on the `nova-compute` process, the state in the compute database will be forced on the baremetal database in case of inconsistencies. Hence a node which was put down using the compute service API cannot be brought up through the baremetal service API since the power sync task will regard the compute services knowledge of the power state as the source of truth. In order to get around this disadvantage of the compute-baremetal power sync, baremetal service does power state change callbacks to the compute service using external events.

Power State Change Callbacks to the Compute Service

its database. By conveying all the power state changes to the compute service, the baremetal service becomes the source of truth thus preventing the compute service from forcing wrong power states on the physical instance during the compute-baremetal power sync. It also adds the possibility of bringing up/down a physical instance through the baremetal service API even if it was put down/up through the compute service API.

be able to send notifications to the compute service and it will fall back to the behaviour of the compute service forcing power states on the baremetal service during the power sync. See `nova` group for more details on the available config options.

Note: The baremetal service sends notifications to the compute service only if the target power state is `power on` or `power off`. Other error and `None` states will be ignored. In situations where the power state change is originally coming from the compute service, the notification will still be sent by the baremetal service and it will be a no-op on the compute service side with a debug log stating the node is already powering on/off.

Note: Although an exclusive lock is used when sending notifications to the compute service, there can still be a race condition if the compute-baremetal power sync happens to happen a nano-second before the power state change event is received from the baremetal service in which case the power state from compute services database will be forced on the node.

Setting the Owner and Lessee

(continues on next page)

(continued from previous page)

Configuring the Bare Metal Service Policy

(continues on next page)

(continued from previous page)

Ports

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Allocations

policy rules that allow non-admins to use allocations effectively:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Deployment and Metalsmith

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

happens first) and is only shut down before rebooting into the final instance. Depending on the configuration, this mode can save several reboots and is particularly useful for scenarios where nodes are enrolled, prepared and provisioned within a short period of time.

Enabling

Inspection

the only local storage contents being those in memory. It is suported by `pxe`, `ipxe`, `redfish-virtual-media` and `ilo-virtual-media` boot interfaces.

Configuration

(continues on next page)

(continued from previous page)

Creating a ramdisk

Booting a ramdisk

(continues on next page)

(continued from previous page)

Note: The requirement to pass `image_source` is artificial and will be fixed in a future version of the Bare Metal service.

Booting an ISO

(continues on next page)

(continued from previous page)

after the boot of the Operating System. Operators wishing to leverage this functionality for actions such as OS installation should explore use of the standard `ramdisk` `deploy_interface` along with the `instance_info/kernel_append_params` setting to pass arbitrary settings such as a mirror URL for the initial ramdisk to load data from. This is a limitation of iPXE and the overall boot process of the operating system where memory allocated by iPXE is released.

Limitations

visioning and cleaning networks

Configuration

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Creating an OS Image

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Configuring the OS Image in glance

is a mountable OS disk.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```
```

Creating a bare metal server

template set in `instance_info` takes precedence over the one specified via the OS image in glance. If no kickstart template is specified (via the nodes `instance_info` or `ks_template` glance image property), the default kickstart template will be used to deploy the OS.

(continues on next page)

(continued from previous page)

```
```

Limitations

Ceph Object Gateway support

Overview

Configure Ironic and Glance with RADOS Gateway

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Building images for Windows

when the instance is spawned on hardware servers (Bare metals).

Requirements:

Preparation:

Implementation:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Emitting Software Metrics

Configuring the Bare Metal Service to Enable Metrics

Enabling metrics in ironic-api and ironic-conductor

also supply connection information in the ironic configuration file:

Enabling metrics in ironic-python-agent

in your ironic configuration file on all ironic-conductor hosts:

ured in the ironic configuration file as well:

Types of Metrics Emitted

the Bare Metal deployment. This estimate may be used to determine if a deployer needs to scale their metrics backend to handle the additional load before enabling metrics. To see which metrics have changed names or have been removed between releases, refer to the ironic release notes.

Note: With the default statsd configuration, each timing metric may create additional metrics due to how statsd handles timing metrics. For more information, see statds documentation on metric types.

Adding New Metrics

a metric is changed or removed to alert deployers of the change.

API Audit Logging

tification_driver = messagingv2) or can be routed to a log file (*[oslo_messaging_notifications]/driver* = *log*).

Enabling API Audit Logging

ration options for the Bare Metal service are included in the etc/ironic/ironic_api_audit_map.conf.sample file. To understand CADF format specified in ironic_api_audit_map.conf file refer to CADF Format.:

Sample Audit Event

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Bare Metal Service state report (via Guru Meditation Reports)

Guru Meditation Report (GMR for short). GMR provides useful debugging information that can be used to obtain an accurate view on the current live state of the system. For example, what threads are running, what configuration parameters are in effect, and more. The eventlet backdoor facility provides an interactive shell interface for any eventlet based process, allowing an administrator to telnet to a pre-defined port and execute a variety of commands.

Configuration

Generating a GMR

Structure of a GMR

Agent Token

Purpose

ken can be viewed as a session identifier or authentication token.

How it works

Note: In the case of the token being embedded with virtual media, it is read from a configuration file with-in the image. Ideally this should be paired with Swift temporary URLs.

the `ironic-python-agent`. As of the Victoria release, use of Agent Token is required for all agents and the previously available setting to force this functionality to be manditory, `[DEFAULT]require_agent_token` no longer has any effect.

With Virtual Media

With PXE/iPXE/etc.

Agent Configuration

action, but can be asserted via the embedded configuration for the agent in the ramdisk. This setting is also available via kernel command line as `ipa-agent-token-required`.

Deploying without BMC Credentials

dentials.

How it works

Enabling

(continues on next page)

(continued from previous page)

Limitations

Layer 3 or DHCP-less ramdisk booting

not only the unreliable TFTP protocol, but DHCP altogether.

be configured accordingly.

(continues on next page)

(continued from previous page)

Note: If desired, some interfaces can still be configured to use DHCP.

Hardware type support

Configuring network data

work service and also works in standalone mode.

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Note: Some fields are redundant with the port information. Were looking into simplifying the format, but currently all these fields are mandatory.

(continues on next page)

(continued from previous page)

Deploying outside of the provisioning network

Tuning Ironic

Memory Utilization

memory map to re-assemble the image contents into a coherent stream of data. This entire process also stresses the kernel buffers and cache.

API

may differ. Naturally there are configuration and performance trade-offs.

the Victoria cycle, a direct invocation of the `ironic-api` program will only launch a maximum of four workers.

up in the front-end webserver and be released to the `ironic-api` as prior requests complete. In environments with long running synchronous calls, such as use of the vendor passthru interface, this can be very problematic.

number of worker threads. At the same time, the scheduler will focus on worker processes as the threads are greenthreads.

Note: Host operating systems featuring in-memory de-duplication should see an improvement in the overall memory footprint with multiple processes, but this is not something the development team has measured and will vary based upon multiple factors.

sponsible conductors are allocated across the cluster. In other words, your amount of memory WILL increase corresponding to the number of nodes managed by each ironic conductor. It is important to understand that features such as conductor groups means that only matching portions of nodes will be considered for the hash ring if needed.

Conductor

tial amount of work all at once.

names and addresses for JSON-RPC or use a central oslo.messaging supported message bus in order for Webserver API to Conductor API communication to be functional.

ers.

tecture.

image block structure and layout, and what deploy interface is being used.

Database

need to be examined in order to return a result set.

Adding indexes

(continues on next page)

(continued from previous page)

Note: The indexes noted have been added automatically by Xena versions of Ironic and later. They are provided here as an example and operators can add them manually prior with versions of Ironic. The database upgrade for the Xena release of Ironic which adds these indexes are only aware of being able to skip index creation if it already exists on MySQL/MariaDB.

Note: It may be possible to use LOCK = NONE. Basic testing indicates this takes a little bit longer, but shouldnt result in the database table becoming write locked during the index creation. If the database engine cannot support this, then the index creation will fail.

tically reduce the result set generation time for the remainder of the query. As of the composition of this document, we do not ship compound indexes in Ironic as we feel the most general benefit is single column indexes, and depending on data present, an operator may wish to explore compound indexes with their database administrator, as comound indexes can also have negative performance impacts if improperly constructed.

the index. At the same time, queries with less field matches, or in different orders will exhibit different performance as the compound index may not be able to be consulted.

Indexes will not fix everything

due to the object model and the need to extract multiple pieces of data at varying levels of the data model to handle cases such as upgrades, the entire result set is downloaded and transformed which is an overhead you do not experience with a command line database client.

What can I do?

from the host running the `ironic-conductor`.

der extreme memory pressure this may still be sub-optimal. Before changing this setting, it is highly advised to consult with your resident Unix wizard or even the Ironic development team in upstream IRC. This feature was added in the Wallaby development cycle.

deployments. See *Using proxies for image download*.

Secure RBAC

Suggested Reading

was started, to provide consistency and a good starting place for most users who need a higher level of granularity.

Historical Context - How we reached our access model

System Scoped

Or to be more precise, tell the policy enforcement framework the information necessary to make decisions.

cal to a system scoped `reader`. In these concepts, the `admin` is allowed to create/delete objects/items. The `reader` is allowed to read details about items and is intended for users who may need an account with read-only access for or front-line support purposes.

cial fields like those to disable cleaning.

Project Scoped

Legacy Behavior

through `system` scoped and `project` scoped requests.

Note:

Note: While Ironic has had the concept of an `owner` and a `lessee`, they are *NOT* used by default. They require custom policy configuration files to be used in the legacy operating mode.

Supported Endpoints

How Project Scoped Works

are in charge of the node. Regardless of the use model that the fields and mechanics support, these fields are to support humans, and possibly services where applicable.

lessees progative, but by default there are actions and field updates that cannot be performed by default. This is also governed by access level within a project.

Field value visibility restrictions

ther and against additional policies. This threshold is controlled with the `baremetal:node:get:filter_threshold`.

Field update restrictions

Allocations

to control access for the creation process, any project member does have the inherent prilege of requesting an allocation. That being said, their allocation request will require physical nodes to be owned or leased to the `project_id` through the `node` fields `owner` or `lessee`.

which are not allocated or deployed, but they cannot reprovision nodes when using only a `member` role. This limitation is not the case for project-scoped users with the `admin` role.

cations will automatically populate an owner. System scoped request are not subjected to this restriction, and operators may change the default restriction via the `baremetal:allocation:create_restricted` policy.

Pratical differences

level. For most users who used a `baremetal` project, or other custom project via a custom policy file, along with a custom role name such as `baremetal_admin`, this will require changing the user to be a `system` scoped user with `admin` privilges.

the node, and then do they have access to the specific resource?.

What is an owner or lessee?

trator from having to correct ownership records should a project need to be removed due to an individuals departure.

How do I assign an owner?

Note: With the default access policy, an `owner` is able to change the assigned `lessee` of a node. However the `lessee` is unable to do the same.

How do I assign a lessee?

What is the difference between an owner and lessee?

Deploy Steps

CHAPTER

SIX

CONFIGURATION GUIDE

6.1 Configuration Reference

Many aspects of the Bare Metal service are specific to the environment it is deployed in. The following pages describe configuration options that can be used to adjust the service to your particular situation.

6.1.1 Configuration Options

The following is an overview of all available configuration options in Ironic. For a sample configuration file, refer to *Sample Configuration File*.

DEFAULT

`run_external_periodic_tasks`

Type boolean

Default `True`

Some periodic tasks can be run in a separate process. Should we run them here?

`backdoor_port`

Type string

Default `<None>`

Enable eventlet backdoor. Acceptable values are 0, <port>, and <start>:<end>, where 0 results in listening on a random tcp port number; <port> results in listening on the specified port number (and not enabling backdoor if that port is in use); and <start>:<end> results in listening on the smallest unused port number within the specified range of port numbers. The chosen port is displayed in the services log file.

`backdoor_socket`

Type string

Default `<None>`

Enable eventlet backdoor, using the provided path as a unix socket that can receive connections. This option is mutually exclusive with backdoor_port in that only one should be provided. If both are provided then the existence of this option overrides the usage of that option. Inside the path {pid} will be replaced with the PID of the current process.

log_options

Type boolean

Default `True`

Enables or disables logging values of all registered options when starting a service (at DEBUG level).

graceful_shutdown_timeout

Type integer

Default `60`

Specify a timeout after which a gracefully shutdown server will exit. Zero value means endless wait.

rpc_conn_pool_size

Type integer

Default `30`

Minimum Value 1

Size of RPC connection pool.

Table 1: Deprecated Variations

Group	Name
DEFAULT	rpc_conn_pool_size

conn_pool_min_size

Type integer

Default `2`

The pool size limit for connections expiration policy

conn_pool_ttl

Type integer

Default `1200`

The time-to-live in sec of idle connections in the pool

executor_thread_pool_size

Type integer

Default `64`

Size of executor thread pool when executor is threading or eventlet.

Table 2: Deprecated Variations

Group	Name
DEFAULT	rpc_thread_pool_size

rpc_response_timeout

Type integer

Default `60`

Seconds to wait for a response from a call.

`transport_url`

Type string

Default `rabbit://`

The network address and optional user credentials for connecting to the messaging backend, in URL format. The expected format is:

driver://[user:pass@]host:port[,[userN:passN@]hostN:portN]/virtual_host?query

Example: rabbit://rabbitmq:password@127.0.0.1:5672//

For full details on the fields in the URL see the documentation of oslo_messaging.TransportURL at https://docs.openstack.org/oslo.messaging/latest/reference/transport.html

`control_exchange`

Type string

Default `openstack`

The default exchange under which topics are scoped. May be overridden by an exchange name specified in the transport_url option.

`rpc_ping_enabled`

Type boolean

Default `False`

Add an endpoint to answer to ping calls. Endpoint is named oslo_rpc_server_ping

`debug`

Type boolean

Default `False`

Mutable This option can be changed without restarting.

If set to true, the logging level will be set to DEBUG instead of the default INFO level.

`log_config_append`

Type string

Default `<None>`

Mutable This option can be changed without restarting.

The name of a logging configuration file. This file is appended to any existing logging configuration files. For details about logging configuration files, see the Python logging module documentation. Note that when logging configuration files are used then all logging configuration is set in the configuration file and other logging configuration options are ignored (for example, log-date-format).

Table 3: Deprecated Variations

Group	Name
DEFAULT	log-config
DEFAULT	log_config

log_date_format

Type string

Default `%Y-%m-%d %H:%M:%S`

Defines the format string for %(asctime)s in log records. Default: the value above . This option is ignored if log_config_append is set.

log_file

Type string

Default `<None>`

(Optional) Name of log file to send logging output to. If no default is set, logging will go to stderr as defined by use_stderr. This option is ignored if log_config_append is set.

Table 4: Deprecated Variations

Group	Name
DEFAULT	logfile

log_dir

Type string

Default `<None>`

(Optional) The base directory used for relative log_file paths. This option is ignored if log_config_append is set.

Table 5: Deprecated Variations

Group	Name
DEFAULT	logdir

watch_log_file

Type boolean

Default `False`

Uses logging handler designed to watch file system. When log file is moved or removed this handler will open a new log file with specified path instantaneously. It makes sense only if log_file option is specified and Linux platform is used. This option is ignored if log_config_append is set.

use_syslog

Type boolean

Default `False`

Use syslog for logging. Existing syslog format is DEPRECATED and will be changed later to honor RFC5424. This option is ignored if log_config_append is set.

use_journal

Type boolean

Default `False`

Enable journald for logging. If running in a systemd environment you may wish to enable journal support. Doing so will use the journal native protocol which includes structured metadata in addition to log messages.This option is ignored if log_config_append is set.

syslog_log_facility

Type string

Default `LOG_USER`

Syslog facility to receive log lines. This option is ignored if log_config_append is set.

use_json

Type boolean

Default `False`

Use JSON formatting for logging. This option is ignored if log_config_append is set.

use_stderr

Type boolean

Default `False`

Log output to standard error. This option is ignored if log_config_append is set.

use_eventlog

Type boolean

Default `False`

Log output to Windows Event Log.

log_rotate_interval

Type integer

Default `1`

The amount of time before the log files are rotated. This option is ignored unless log_rotation_type is setto interval.

log_rotate_interval_type

Type string

Default `days`

Valid Values Seconds, Minutes, Hours, Days, Weekday, Midnight

Rotation interval type. The time of the last file change (or the time when the service was started) is used when scheduling the next rotation.

max_logfile_count

Type integer

Default `30`

Maximum number of rotated log files.

`max_logfile_size_mb`

Type integer

Default `200`

Log file maximum size in MB. This option is ignored if log_rotation_type is not set to size.

`log_rotation_type`

Type string

Default `none`

Valid Values interval, size, none

Log rotation type.

Possible values

interval Rotate logs at predefined time intervals.

size Rotate logs once they reach a predefined size.

none Do not rotate log files.

`logging_context_format_string`

Type string

Default `%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s%(message)s`

Format string to use for log messages with context. Used by oslo_log.formatters.ContextFormatter

`logging_default_format_string`

Type string

Default `%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s%(message)s`

Format string to use for log messages when context is undefined. Used by oslo_log.formatters.ContextFormatter

`logging_debug_format_suffix`

Type string

Default `%(funcName)s %(pathname)s:%(lineno)d`

Additional data to append to log message when logging level for the message is DEBUG. Used by oslo_log.formatters.ContextFormatter

`logging_exception_prefix`

Type string

Default `%(asctime)s.%(msecs)03d %(process)d ERROR %(name)s %(instance)s`

Prefix each line of exception output with this format. Used by oslo_log.formatters.ContextFormatter

logging_user_identity_format

Type string

Default `%(user)s %(tenant)s %(domain)s %(user_domain)s %(project_domain)s`

Defines the format string for %(user_identity)s that is used in logging_context_format_string. Used by oslo_log.formatters.ContextFormatter

default_log_levels

Type list

Default `['amqp=WARNING', 'amqplib=WARNING', 'qpid.messaging=INFO', 'oslo.messaging=INFO', 'oslo_messaging=INFO', 'sqlalchemy=WARNING', 'stevedore=INFO', 'eventlet.wsgi.server=INFO', 'iso8601=WARNING', 'requests=WARNING', 'glanceclient=WARNING', 'urllib3.connectionpool=WARNING', 'keystonemiddleware.auth_token=INFO', 'keystoneauth.session=INFO', 'openstack=WARNING']`

List of package logging levels in logger=LEVEL pairs. This option is ignored if log_config_append is set.

publish_errors

Type boolean

Default `False`

Enables or disables publication of error events.

instance_format

Type string

Default `"[instance: %(uuid)s] "`

The format for an instance that is passed with the log message.

instance_uuid_format

Type string

Default `"[instance: %(uuid)s] "`

The format for an instance UUID that is passed with the log message.

rate_limit_interval

Type integer

Default `0`

Interval, number of seconds, of log rate limiting.

`rate_limit_burst`

Type integer

Default `0`

Maximum number of logged messages per rate_limit_interval.

`rate_limit_except_level`

Type string

Default `CRITICAL`

Log level name used by rate limiting: CRITICAL, ERROR, INFO, WARNING, DEBUG or empty string. Logs with level greater or equal to rate_limit_except_level are not filtered. An empty string means that all levels are filtered.

`fatal_deprecations`

Type boolean

Default `False`

Enables or disables fatal status of deprecations.

`auth_strategy`

Type string

Default `keystone`

Valid Values noauth, keystone, http_basic

Authentication strategy used by ironic-api. noauth should not be used in a production environment because all authentication will be disabled.

Possible values

noauth no authentication

keystone use the Identity service for authentication

http_basic HTTP basic authentication

`http_basic_auth_user_file`

Type string

Default `/etc/ironic/htpasswd`

Path to Apache format user authentication file used when auth_strategy=http_basic

`debug_tracebacks_in_api`

Type boolean

Default `False`

Return server tracebacks in the API response for any error responses. WARNING: this is insecure and should not be used in a production environment.

`pecan_debug`

Type boolean

Default `False`

Enable pecan debug mode. WARNING: this is insecure and should not be used in a production environment.

default_resource_class

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Resource class to use for new nodes when no resource class is provided in the creation request.

enabled_hardware_types

Type list

Default `['ipmi']`

Specify the list of hardware types to load during service initialization. Missing hardware types, or hardware types which fail to initialize, will prevent the conductor service from starting. This option defaults to a recommended set of production-oriented hardware types. A complete list of hardware types present on your system may be found by enumerating the ironic.hardware.types entrypoint.

enabled_bios_interfaces

Type list

Default `['no-bios']`

Specify the list of bios interfaces to load during service initialization. Missing bios interfaces, or bios interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one bios interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented bios interfaces. A complete list of bios interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.bios entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled bios interfaces on every ironic-conductor service.

default_bios_interface

Type string

Default `<None>`

Default bios interface to be used for nodes that do not have bios_interface field set. A complete list of bios interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.bios entrypoint.

enabled_boot_interfaces

Type list

Default `['pxe']`

Specify the list of boot interfaces to load during service initialization. Missing boot interfaces, or boot interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one boot interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented boot interfaces. A complete list of boot interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.boot entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled boot interfaces on every ironic-conductor service.

`default_boot_interface`

Type string

Default `<None>`

Default boot interface to be used for nodes that do not have boot_interface field set. A complete list of boot interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.boot entrypoint.

`enabled_console_interfaces`

Type list

Default `['no-console']`

Specify the list of console interfaces to load during service initialization. Missing console interfaces, or console interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one console interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented console interfaces. A complete list of console interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.console entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled console interfaces on every ironic-conductor service.

`default_console_interface`

Type string

Default `<None>`

Default console interface to be used for nodes that do not have console_interface field set. A complete list of console interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.console entrypoint.

`enabled_deploy_interfaces`

Type list

Default `['direct']`

Specify the list of deploy interfaces to load during service initialization. Missing deploy interfaces, or deploy interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one deploy interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented deploy interfaces. A complete list of deploy interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.deploy entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled deploy interfaces on every ironic-conductor service.

`default_deploy_interface`

Type string

Default `<None>`

Default deploy interface to be used for nodes that do not have deploy_interface field set. A complete list of deploy interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.deploy entrypoint.

`enabled_inspect_interfaces`

Type list

Default `['no-inspect']`

Specify the list of inspect interfaces to load during service initialization. Missing inspect interfaces, or inspect interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one inspect interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented inspect interfaces. A complete list of inspect interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.inspect entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled inspect interfaces on every ironic-conductor service.

`default_inspect_interface`

Type string

Default `<None>`

Default inspect interface to be used for nodes that do not have inspect_interface field set. A complete list of inspect interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.inspect entrypoint.

`enabled_management_interfaces`

Type list

Default `['ipmitool']`

Specify the list of management interfaces to load during service initialization. Missing management interfaces, or management interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one management interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented management interfaces. A complete list of management interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.management entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled management interfaces on every ironic-conductor service.

`default_management_interface`

Type string

Default `<None>`

Default management interface to be used for nodes that do not have management_interface field set. A complete list of management interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.management entrypoint.

`enabled_network_interfaces`

Type list

Default `['flat', 'noop']`

Specify the list of network interfaces to load during service initialization. Missing network interfaces, or network interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one network interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented network interfaces. A complete list of network interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.network entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled network interfaces on every ironic-conductor service.

`default_network_interface`

Type string

Default `<None>`

Default network interface to be used for nodes that do not have network_interface field set. A complete list of network interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.network entrypoint.

`enabled_power_interfaces`

Type list

Default `['ipmitool']`

Specify the list of power interfaces to load during service initialization. Missing power interfaces, or power interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one power interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented power interfaces. A complete list of power interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.power entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled power interfaces on every ironic-conductor service.

`default_power_interface`

Type string

Default `<None>`

Default power interface to be used for nodes that do not have power_interface field set. A complete list of power interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.power entrypoint.

`enabled_raid_interfaces`

Type list

Default `['agent', 'no-raid']`

Specify the list of raid interfaces to load during service initialization. Missing raid interfaces, or raid interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one raid interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented raid interfaces. A complete list of raid interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.raid entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled raid interfaces on every ironic-conductor service.

`default_raid_interface`

Type string

Default `<None>`

Default raid interface to be used for nodes that do not have raid_interface field set. A complete list of raid interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.raid entrypoint.

`enabled_rescue_interfaces`

Type list

Default `['no-rescue']`

Specify the list of rescue interfaces to load during service initialization. Missing rescue interfaces, or rescue interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one rescue interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented rescue interfaces. A complete list of rescue interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.rescue entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled rescue interfaces on every ironic-conductor service.

`default_rescue_interface`

Type string

Default `<None>`

Default rescue interface to be used for nodes that do not have rescue_interface field set. A complete list of rescue interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.rescue entrypoint.

`enabled_storage_interfaces`

Type list

Default `['cinder', 'noop']`

Specify the list of storage interfaces to load during service initialization. Missing storage interfaces, or storage interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one storage interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented storage interfaces. A complete list of storage interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.storage entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled storage interfaces on every ironic-conductor service.

`default_storage_interface`

Type string

Default `noop`

Default storage interface to be used for nodes that do not have storage_interface field set. A complete list of storage interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.storage entrypoint.

`enabled_vendor_interfaces`

Type list

Default `['ipmitool', 'no-vendor']`

Specify the list of vendor interfaces to load during service initialization. Missing vendor interfaces, or vendor interfaces which fail to initialize, will prevent the ironic-conductor service from starting. At least one vendor interface that is supported by each enabled hardware type must be enabled here, or the ironic-conductor service will not start. Must not be an empty list. The default value is a recommended set of production-oriented vendor interfaces. A complete list of vendor interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.vendor entrypoint. When setting this value, please make sure that every enabled hardware type will have the same set of enabled vendor interfaces on every ironic-conductor service.

`default_vendor_interface`

Type string

Default `<None>`

Default vendor interface to be used for nodes that do not have vendor_interface field set. A complete list of vendor interfaces present on your system may be found by enumerating the ironic.hardware.interfaces.vendor entrypoint.

`log_in_db_max_size`

Type integer

Default `4096`

Max number of characters of any node last_error/maintenance_reason pushed to database.

`hash_partition_exponent`

Type integer

Default `5`

Exponent to determine number of hash partitions to use when distributing load across conductors. Larger values will result in more even distribution of load and less load when rebalancing the ring, but more memory usage. Number of partitions per conductor is (2^hash_partition_exponent). This determines the granularity of rebalancing: given 10 hosts, and an exponent of the 2, there are 40 partitions in the ring.A few thousand partitions should make rebalancing smooth in most cases. The default is suitable for up to a few hundred conductors. Configuring for too many partitions has a negative impact on CPU usage.

`hash_ring_reset_interval`

Type integer

Default `15`

Time (in seconds) after which the hash ring is considered outdated and is refreshed on the next access.

`hash_ring_algorithm`

Type string

Default `md5`

Valid Values shake_128, sha3_224, shake_256, sha3_384, sha1, md5, sha3_256, blake2s, sha256, blake2b, sha3_512, sha512, sha224, sha384

Advanced Option Intended for advanced users and not used by the majority of users, and might have a significant effect on stability and/or performance.

Hash function to use when building the hash ring. If running on a FIPS system, do not use md5. WARNING: all ironic services in a cluster MUST use the same algorithm at all times. Changing the algorithm requires an offline update.

`force_raw_images`

Type boolean

Default `True`

Mutable This option can be changed without restarting.

If True, convert backing images to raw disk image format.

`raw_image_growth_factor`

Type floating point

Default `2.0`

Minimum Value 1.0

The scale factor used for estimating the size of a raw image converted from compact image formats such as QCOW2. Default is 2.0, must be greater than 1.0.

`isolinux_bin`

Type string

Default `/usr/lib/syslinux/isolinux.bin`

Path to isolinux binary file.

`isolinux_config_template`

Type string

Default `$pybasedir/common/isolinux_config.template`

Template file for isolinux configuration file.

`grub_config_path`

Type string

Default `/boot/grub/grub.cfg`

GRUB2 configuration file location on the UEFI ISO images produced by ironic. The default value is usually incorrect and should not be relied on. If you use a GRUB2 image from a certain distribution, use a distribution-specific path here, e.g. EFI/ubuntu/grub.cfg

`grub_config_template`

Type string

Default `$pybasedir/common/grub_conf.template`

Template file for grub configuration file.

ldlinux_c32

Type string

Default `<None>`

Path to ldlinux.c32 file. This file is required for syslinux 5.0 or later. If not specified, the file is looked for in /usr/lib/syslinux/modules/bios/ldlinux.c32 and /usr/share/syslinux/ldlinux.c32.

esp_image

Type string

Default `<None>`

Path to EFI System Partition image file. This file is recommended for creating UEFI bootable ISO images efficiently. ESP image should contain a FAT12/16/32-formatted file system holding EFI boot loaders (e.g. GRUB2) for each hardware architecture ironic needs to boot. This option is only used when neither ESP nor ISO deploy image is configured to the node being deployed in which case ironic will attempt to fetch ESP image from the configured location or extract ESP image from UEFI-bootable deploy ISO image.

parallel_image_downloads

Type boolean

Default `False`

Mutable This option can be changed without restarting.

Run image downloads and raw format conversions in parallel.

my_ip

Type string

Default `127.0.0.1`

This option has a sample default set, which means that its actual default value may vary from the one documented above.

IPv4 address of this host. If unset, will determine the IP programmatically. If unable to do so, will use 127.0.0.1. NOTE: This field does accept an IPv6 address as an override for templates and URLs, however it is recommended that [DEFAULT]my_ipv6 is used along with DNS names for service URLs for dual-stack environments.

my_ipv6

Type string

Default `2001:db8::1`

This option has a sample default set, which means that its actual default value may vary from the one documented above.

IP address of this host using IPv6. This value must be supplied via the configuration and cannot be adequately programmatically determined like the [DEFAULT]my_ip parameter for IPv4.

notification_level

Type string

Default `<None>`

Valid Values debug, info, warning, error, critical

Specifies the minimum level for which to send notifications. If not set, no notifications will be sent. The default is for this option to be unset.

Possible values

debug debug level

info info level

warning warning level

error error level

critical critical level

`versioned_notifications_topics`

Type list

Default `['ironic_versioned_notifications']`

Specifies the topics for the versioned notifications issued by Ironic.

The default value is fine for most deployments and rarely needs to be changed. However, if you have a third-party service that consumes versioned notifications, it might be worth getting a topic for that service. Ironic will send a message containing a versioned notification payload to each topic queue in this list.

The list of versioned notifications is visible in https://docs.openstack.org/ironic/latest/admin/notifications.html

`pybasedir`

Type string

Default `/usr/lib/python/site-packages/ironic/ironic`

This option has a sample default set, which means that its actual default value may vary from the one documented above.

Directory where the ironic python module is installed.

`bindir`

Type string

Default `$pybasedir/bin`

Directory where ironic binaries are installed.

`state_path`

Type string

Default `$pybasedir`

Top-level directory for maintaining ironics state.

`default_portgroup_mode`

Type string

Default `active-backup`

Mutable This option can be changed without restarting.

Default mode for portgroups. Allowed values can be found in the linux kernel documentation on bonding: https://www.kernel.org/doc/Documentation/networking/bonding.txt.

host

Type string

Default `localhost`

This option has a sample default set, which means that its actual default value may vary from the one documented above.

Name of this node. This can be an opaque identifier. It is not necessarily a hostname, FQDN, or IP address. However, the node name must be valid within an AMQP key, and if using ZeroMQ (will be removed in the Stein release), a valid hostname, FQDN, or IP address.

pin_release_version

Type string

Default `<None>`

Valid Values wallaby, victoria, 9.2, 17.0, 16.2, 16.1, 16.0, 15.1, 15.0, 14.0, 13.0, 12.2, 12.1, 12.0, 11.1, 11.0, 10.1, 10.0

Mutable This option can be changed without restarting.

Used for rolling upgrades. Setting this option downgrades (or pins) the Bare Metal API, the internal ironic RPC communication, and the database objects to their respective versions, so they are compatible with older services. When doing a rolling upgrade from version N to version N+1, set (to pin) this to N. To unpin (default), leave it unset and the latest versions will be used.

Possible values

wallaby	wallaby release
victoria	victoria release
9.2	9.2 release
17.0	17.0 release
16.2	16.2 release
16.1	16.1 release
16.0	16.0 release
15.1	15.1 release
15.0	15.0 release
14.0	14.0 release
13.0	13.0 release
12.2	12.2 release

12.1 12.1 release

12.0 12.0 release

11.1 11.1 release

11.0 11.0 release

10.1 10.1 release

10.0 10.0 release

rpc_transport

Type string

Default `oslo`

Valid Values oslo, json-rpc

Which RPC transport implementation to use between conductor and API services

Possible values

oslo use oslo.messaging transport

json-rpc use JSON RPC transport

minimum_memory_warning_only

Type boolean

Default `False`

Mutable This option can be changed without restarting.

Setting to govern if Ironic should only warn instead of attempting to hold back the request in order to prevent the exhaustion of system memory.

minimum_required_memory

Type integer

Default `1024`

Mutable This option can be changed without restarting.

Minimum memory in MiB for the system to have available prior to starting a memory intensive process on the conductor.

minimum_memory_wait_time

Type integer

Default `15`

Mutable This option can be changed without restarting.

Seconds to wait between retries for free memory before launching the process. This, combined with `memory_wait_retries` allows the conductor to determine how long we should attempt to directly retry.

minimum_memory_wait_retries

Type integer

Default `6`

Mutable This option can be changed without restarting.

Number of retries to hold onto the worker before failing or returning the thread to the pool if the conductor can automatically retry.

rootwrap_config

Type string

Default `/etc/ironic/rootwrap.conf`

Path to the rootwrap configuration file to use for running commands as root.

tempdir

Type string

Default `/tmp`

This option has a sample default set, which means that its actual default value may vary from the one documented above.

Temporary working directory, default is Python temp dir.

webserver_verify_ca

Type string

Default `True`

Mutable This option can be changed without restarting.

CA certificates to be used for certificate verification. This can be either a Boolean value or a path to a CA_BUNDLE file.If set to True, the certificates present in the standard path are used to verify the host certificates.If set to False, the conductor will ignore verifying the SSL certificate presented by the host.If its a path, conductor uses the specified certificate for SSL verification. If the path does not exist, the behavior is same as when this value is set to True i.e the certificates present in the standard path are used for SSL verification.Defaults to True.

webserver_connection_timeout

Type integer

Default `60`

Connection timeout when accessing remote web servers with images.

agent

manage_agent_boot

Type boolean

Default `True`

Whether Ironic will manage booting of the agent ramdisk. If set to False, you will need to configure your mechanism to allow booting the agent ramdisk.

memory_consumed_by_agent

Type integer

Default `0`

Mutable This option can be changed without restarting.

The memory size in MiB consumed by agent when it is booted on a bare metal node. This is used for checking if the image can be downloaded and deployed on the bare metal node after booting agent ramdisk. This may be set according to the memory consumed by the agent ramdisk image.

`stream_raw_images`

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Whether the agent ramdisk should stream raw images directly onto the disk or not. By streaming raw images directly onto the disk the agent ramdisk will not spend time copying the image to a tmpfs partition (therefore consuming less memory) prior to writing it to the disk. Unless the disk where the image will be copied to is really slow, this option should be set to True. Defaults to True.

`post_deploy_get_power_state_retries`

Type integer

Default `6`

Number of times to retry getting power state to check if bare metal node has been powered off after a soft power off.

`post_deploy_get_power_state_retry_interval`

Type integer

Default `5`

Amount of time (in seconds) to wait between polling power state after trigger soft poweroff.

`agent_api_version`

Type string

Default `v1`

API version to use for communicating with the ramdisk agent.

`deploy_logs_collect`

Type string

Default `on_failure`

Valid Values always, on_failure, never

Mutable This option can be changed without restarting.

Whether Ironic should collect the deployment logs on deployment failure (on_failure), always or never.

Possible values

always always collect the logs

on_failure only collect logs if there is a failure

never never collect logs

`deploy_logs_storage_backend`

Type string

Default `local`

Valid Values local, swift

Mutable This option can be changed without restarting.

The name of the storage backend where the logs will be stored.

Possible values

local store the logs locally

swift store the logs in Object Storage service

`deploy_logs_local_path`

Type string

Default `/var/log/ironic/deploy`

Mutable This option can be changed without restarting.

The path to the directory where the logs should be stored, used when the deploy_logs_storage_backend is configured to local.

`deploy_logs_swift_container`

Type string

Default `ironic_deploy_logs_container`

Mutable This option can be changed without restarting.

The name of the Swift container to store the logs, used when the deploy_logs_storage_backend is configured to swift.

`deploy_logs_swift_days_to_expire`

Type integer

Default `30`

Mutable This option can be changed without restarting.

Number of days before a log object is marked as expired in Swift. If None, the logs will be kept forever or until manually deleted. Used when the deploy_logs_storage_backend is configured to swift.

`image_download_source`

Type string

Default `http`

Valid Values swift, http, local

Mutable This option can be changed without restarting.

Specifies whether direct deploy interface should try to use the image source directly or if ironic should cache the image on the conductor and serve it from ironics own http server.

Possible values

swift IPA ramdisk retrieves instance image from the Object Storage service.

http IPA ramdisk retrieves instance image from HTTP service served at conductor nodes.

local Same as http, but HTTP images are also cached locally, converted and served from the conductor

`command_timeout`

Type integer

Default `60`

Mutable This option can be changed without restarting.

Timeout (in seconds) for IPA commands.

`max_command_attempts`

Type integer

Default `3`

This is the maximum number of attempts that will be done for IPA commands that fails due to network problems.

`command_wait_attempts`

Type integer

Default `100`

Number of attempts to check for asynchronous commands completion before timing out.

`command_wait_interval`

Type integer

Default `6`

Number of seconds to wait for between checks for asynchronous commands completion.

`neutron_agent_poll_interval`

Type integer

Default `2`

Mutable This option can be changed without restarting.

The number of seconds Neutron agent will wait between polling for device changes. This value should be the same as CONF.AGENT.polling_interval in Neutron configuration.

neutron_agent_max_attempts

Type integer

Default `100`

Max number of attempts to validate a Neutron agent status before raising network error for a dead agent.

neutron_agent_status_retry_interval

Type integer

Default `10`

Wait time in seconds between attempts for validating Neutron agent status.

require_tls

Type boolean

Default `False`

Mutable This option can be changed without restarting.

If set to True, callback URLs without https:// will be rejected by the conductor.

certificates_path

Type string

Default `/var/lib/ironic/certificates`

Path to store auto-generated TLS certificates used to validate connections to the ramdisk.

verify_ca

Type string

Default `True`

Path to the TLS CA to validate connection to the ramdisk. Set to True to use the system default CA storage. Set to False to disable validation. Ignored when automatic TLS setup is used.

api_ca_file

Type string

Default `<None>`

Path to the TLS CA that is used to start the bare metal API. In some boot methods this file can be passed to the ramdisk.

anaconda

default_ks_template

Type string

Default `$pybasedir/drivers/modules/ks.cfg.template`

Mutable This option can be changed without restarting.

kickstart template to use when no kickstart template is specified in the instance_info or the glance OS image.

ansible

`ansible_extra_args`

Type string

Default `<None>`

Extra arguments to pass on every invocation of Ansible.

`verbosity`

Type integer

Default `<None>`

Minimum Value 0

Maximum Value 4

Set ansible verbosity level requested when invoking ansible-playbook command. 4 includes detailed SSH session logging. Default is 4 when global debug is enabled and 0 otherwise.

`ansible_playbook_script`

Type string

Default `ansible-playbook`

Path to ansible-playbook script. Default will search the $PATH configured for user running ironic-conductor process. Provide the full path when ansible-playbook is not in $PATH or installed in not default location.

`playbooks_path`

Type string

Default `$pybasedir/drivers/modules/ansible/playbooks`

Path to directory with playbooks, roles and local inventory.

`config_file_path`

Type string

Default `$pybasedir/drivers/modules/ansible/playbooks/ansible.cfg`

Path to ansible configuration file. If set to empty, system default will be used.

`post_deploy_get_power_state_retries`

Type integer

Default `6`

Minimum Value 0

Number of times to retry getting power state to check if bare metal node has been powered off after a soft power off. Value of 0 means do not retry on failure.

`post_deploy_get_power_state_retry_interval`

Type integer

Default `5`

Minimum Value 0

Amount of time (in seconds) to wait between polling power state after trigger soft poweroff.

extra_memory

Type integer

Default `10`

Extra amount of memory in MiB expected to be consumed by Ansible-related processes on the node. Affects decision whether image will fit into RAM.

image_store_insecure

Type boolean

Default `False`

Skip verifying SSL connections to the image store when downloading the image. Setting it to True is only recommended for testing environments that use self-signed certificates.

image_store_cafile

Type string

Default `<None>`

Specific CA bundle to use for validating SSL connections to the image store. If not specified, CA available in the ramdisk will be used. Is not used by default playbooks included with the driver. Suitable for environments that use self-signed certificates.

image_store_certfile

Type string

Default `<None>`

Client cert to use for SSL connections to image store. Is not used by default playbooks included with the driver.

image_store_keyfile

Type string

Default `<None>`

Client key to use for SSL connections to image store. Is not used by default playbooks included with the driver.

default_username

Type string

Default `ansible`

Name of the user to use for Ansible when connecting to the ramdisk over SSH. It may be overridden by per-node ansible_username option in nodes driver_info field.

default_key_file

Type string

Default `<None>`

Absolute path to the private SSH key file to use by Ansible by default when connecting to the ramdisk over SSH. Default is to use default SSH keys configured for the user running the ironic-conductor service. Private keys with password must be pre-loaded into ssh-agent. It may be overridden by per-node ansible_key_file option in nodes driver_info field.

`default_deploy_playbook`

Type string

Default `deploy.yaml`

Path (relative to $playbooks_path or absolute) to the default playbook used for deployment. It may be overridden by per-node ansible_deploy_playbook option in nodes driver_info field.

`default_shutdown_playbook`

Type string

Default `shutdown.yaml`

Path (relative to $playbooks_path or absolute) to the default playbook used for graceful in-band shutdown of the node. It may be overridden by per-node ansible_shutdown_playbook option in nodes driver_info field.

`default_clean_playbook`

Type string

Default `clean.yaml`

Path (relative to $playbooks_path or absolute) to the default playbook used for node cleaning. It may be overridden by per-node ansible_clean_playbook option in nodes driver_info field.

`default_clean_steps_config`

Type string

Default `clean_steps.yaml`

Path (relative to $playbooks_path or absolute) to the default auxiliary cleaning steps file used during the node cleaning. It may be overridden by per-node ansible_clean_steps_config option in nodes driver_info field.

`default_python_interpreter`

Type string

Default `<None>`

Absolute path to the python interpreter on the managed machines. It may be overridden by per-node ansible_python_interpreter option in nodes driver_info field. By default, ansible uses /usr/bin/python

api

host_ip

Type host address

Default `0.0.0.0`

The IP address or hostname on which ironic-api listens.

port

Type port number

Default `6385`

Minimum Value 0

Maximum Value 65535

The TCP port on which ironic-api listens.

max_limit

Type integer

Default `1000`

Mutable This option can be changed without restarting.

The maximum number of items returned in a single response from a collection resource.

public_endpoint

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Public URL to use when building the links to the API resources (for example, https://ironic.rocks:6384). If None the links will be built using the requests host URL. If the API is operating behind a proxy, you will want to change this to represent the proxys URL. Defaults to None. Ignored when proxy headers parsing is enabled via [oslo_middleware]enable_proxy_headers_parsing option.

api_workers

Type integer

Default `<None>`

Number of workers for OpenStack Ironic API service. The default is equal to the number of CPUs available, but not more than 4. One worker is used if the CPU number cannot be detected.

enable_ssl_api

Type boolean

Default `False`

Enable the integrated stand-alone API to service requests via HTTPS instead of HTTP. If there is a front-end service performing HTTPS offloading from the service, this option should be False; note, you will want to enable proxy headers parsing with [oslo_middleware]enable_proxy_headers_parsing option or configure [api]public_endpoint option to set URLs in responses to the SSL terminated one.

restrict_lookup

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Whether to restrict the lookup API to only nodes in certain states.

ramdisk_heartbeat_timeout

Type integer

Default `300`

Mutable This option can be changed without restarting.

Maximum interval (in seconds) for agent heartbeats.

network_data_schema

Type string

Default `$pybasedir/api/controllers/v1/network-data-schema.json`

Schema for network data used by this deployment.

audit

enabled

Type boolean

Default `False`

Enable auditing of API requests (for ironic-api service).

audit_map_file

Type string

Default `/etc/ironic/api_audit_map.conf`

Path to audit map file for ironic-api service. Used only when API audit is enabled.

ignore_req_list

Type string

Default `''`

Comma separated list of Ironic REST API HTTP methods to be ignored during audit logging. For example: auditing will not be done on any GET or POST requests if this is set to GET,POST. It is used only when API audit is enabled.

cinder

`action_retries`

Type integer

Default `3`

Number of retries in the case of a failed action (currently only used when detaching volumes).

`action_retry_interval`

Type integer

Default `5`

Retry interval in seconds in the case of a failed action (only specific actions are retried).

`auth_url`

Type unknown type

Default `<None>`

Authentication URL

`auth_type`

Type unknown type

Default `<None>`

Authentication type to load

Table 6: Deprecated Variations

Group	Name
cinder	auth_plugin

`cafile`

Type string

Default `<None>`

PEM encoded Certificate Authority to use when verifying HTTPs connections.

`certfile`

Type string

Default `<None>`

PEM encoded client certificate cert file

`collect_timing`

Type boolean

Default `False`

Collect per-API call timing information.

`connect_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for connection errors.

`connect_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for connection errors. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`default_domain_id`

Type unknown type

Default `<None>`

Optional domain ID to use with v3 and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`default_domain_name`

Type unknown type

Default `<None>`

Optional domain name to use with v3 API and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`domain_id`

Type unknown type

Default `<None>`

Domain ID to scope to

`domain_name`

Type unknown type

Default `<None>`

Domain name to scope to

`endpoint_override`

Type string

Default `<None>`

Always use this endpoint URL for requests for this client. NOTE: The unversioned endpoint should be specified here; to request a particular API version, use the *version*, *min-version*, and/or *max-version* options.

`insecure`

Type boolean

Default `False`

Verify HTTPS connections.

`keyfile`

Type string

Default `<None>`

PEM encoded client certificate key file

`max_version`

Type string

Default `<None>`

The maximum major version of a given API, intended to be used as the upper bound of a range with min_version. Mutually exclusive with version.

`min_version`

Type string

Default `<None>`

The minimum major version of a given API, intended to be used as the lower bound of a range with max_version. Mutually exclusive with version. If min_version is given with no max_version it is as if max version is latest.

`password`

Type unknown type

Default `<None>`

Users password

`project_domain_id`

Type unknown type

Default `<None>`

Domain ID containing project

`project_domain_name`

Type unknown type

Default `<None>`

Domain name containing project

`project_id`

Type unknown type

Default `<None>`

Project ID to scope to

Table 7: Deprecated Variations

Group	Name
cinder	tenant-id
cinder	tenant_id

`project_name`

Type unknown type

Default `<None>`

Project name to scope to

Table 8: Deprecated Variations

Group	Name
cinder	tenant-name
cinder	tenant_name

`region_name`

Type string

Default `<None>`

The default region_name for endpoint URL discovery.

`retries`

Type integer

Default `3`

Client retries in the case of a failed request connection.

`service_name`

Type string

Default `<None>`

The default service_name for endpoint URL discovery.

`service_type`

Type string

Default `volumev3`

The default service_type for endpoint URL discovery.

`split_loggers`

Type boolean

Default `False`

Log requests to multiple loggers.

`status_code_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for retriable HTTP status codes.

`status_code_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for retriable status codes. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

system_scope

Type unknown type

Default `<None>`

Scope for system operations

tenant_id

Type unknown type

Default `<None>`

Tenant ID

tenant_name

Type unknown type

Default `<None>`

Tenant Name

timeout

Type integer

Default `<None>`

Timeout value for http requests

trust_id

Type unknown type

Default `<None>`

Trust ID

user_domain_id

Type unknown type

Default `<None>`

Users domain id

user_domain_name

Type unknown type

Default `<None>`

Users domain name

user_id

Type unknown type

Default `<None>`

User id

username

Type unknown type

Default `<None>`

Username

Table 9: Deprecated Variations

Group	Name
cinder	user-name
cinder	user_name

`valid_interfaces`

Type list

Default `['internal', 'public']`

List of interfaces, in order of preference, for endpoint URL.

`version`

Type string

Default `<None>`

Minimum Major API version within a given Major API version for endpoint URL discovery. Mutually exclusive with min_version and max_version

conductor

`workers_pool_size`

Type integer

Default `100`

Minimum Value 3

The size of the workers greenthread pool. Note that 2 threads will be reserved by the conductor itself for handling heart beats and periodic tasks. On top of that, *sync_power_state_workers* will take up to 7 green threads with the default value of 8.

`heartbeat_interval`

Type integer

Default `10`

Seconds between conductor heart beats.

`heartbeat_timeout`

Type integer

Default `60`

Maximum Value 315576000

Mutable This option can be changed without restarting.

Maximum time (in seconds) since the last check-in of a conductor. A conductor is considered inactive when this time has been exceeded.

sync_power_state_interval

Type integer

Default `60`

Interval between syncing the node power state to the database, in seconds. Set to 0 to disable syncing.

check_provision_state_interval

Type integer

Default `60`

Minimum Value 0

Interval between checks of provision timeouts, in seconds. Set to 0 to disable checks.

check_rescue_state_interval

Type integer

Default `60`

Minimum Value 1

Interval (seconds) between checks of rescue timeouts.

check_allocations_interval

Type integer

Default `60`

Minimum Value 0

Interval between checks of orphaned allocations, in seconds. Set to 0 to disable checks.

deploy_callback_timeout

Type integer

Default `1800`

Minimum Value 0

Timeout (seconds) to wait for a callback from a deploy ramdisk. Set to 0 to disable timeout.

force_power_state_during_sync

Type boolean

Default `True`

Mutable This option can be changed without restarting.

During sync_power_state, should the hardware power state be set to the state recorded in the database (True) or should the database be updated based on the hardware state (False).

power_state_sync_max_retries

Type integer

Default `3`

During sync_power_state failures, limit the number of times Ironic should try syncing the hardware node power state with the node power state in DB

`sync_power_state_workers`

Type integer

Default `8`

Minimum Value 1

The maximum number of worker threads that can be started simultaneously to sync nodes power states from the periodic task.

`periodic_max_workers`

Type integer

Default `8`

Maximum number of worker threads that can be started simultaneously by a periodic task. Should be less than RPC thread pool size.

`node_locked_retry_attempts`

Type integer

Default `3`

Number of attempts to grab a node lock.

`node_locked_retry_interval`

Type integer

Default `1`

Seconds to sleep between node lock attempts.

`send_sensor_data`

Type boolean

Default `False`

Enable sending sensor data message via the notification bus

`send_sensor_data_interval`

Type integer

Default `600`

Minimum Value 1

Seconds between conductor sending sensor data message to ceilometer via the notification bus.

`send_sensor_data_workers`

Type integer

Default `4`

Minimum Value 1

The maximum number of workers that can be started simultaneously for send data from sensors periodic task.

send_sensor_data_wait_timeout

Type integer

Default `300`

The time in seconds to wait for send sensors data periodic task to be finished before allowing periodic call to happen again. Should be less than send_sensor_data_interval value.

send_sensor_data_types

Type list

Default `['ALL']`

List of comma separated meter types which need to be sent to Ceilometer. The default value, ALL, is a special value meaning send all the sensor data.

send_sensor_data_for_undeployed_nodes

Type boolean

Default `False`

The default for sensor data collection is to only collect data for machines that are deployed, however operators may desire to know if there are failures in hardware that is not presently in use. When set to true, the conductor will collect sensor information from all nodes when sensor data collection is enabled via the send_sensor_data setting.

sync_local_state_interval

Type integer

Default `180`

When conductors join or leave the cluster, existing conductors may need to update any persistent local state as nodes are moved around the cluster. This option controls how often, in seconds, each conductor will check for nodes that it should take over. Set it to 0 (or a negative value) to disable the check entirely.

configdrive_swift_container

Type string

Default `ironic_configdrive_container`

Name of the Swift container to store config drive data. Used when configdrive_use_object_store is True.

configdrive_swift_temp_url_duration

Type integer

Default `<None>`

Minimum Value 60

The timeout (in seconds) after which a configdrive temporary URL becomes invalid. Defaults to deploy_callback_timeout if it is set, otherwise to 1800 seconds. Used when configdrive_use_object_store is True.

inspect_wait_timeout

Type integer

Default `1800`

Minimum Value 0

Timeout (seconds) for waiting for node inspection. 0 - unlimited.

automated_clean

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Enables or disables automated cleaning. Automated cleaning is a configurable set of steps, such as erasing disk drives, that are performed on the node to ensure it is in a baseline state and ready to be deployed to. This is done after instance deletion as well as during the transition from a manageable to available state. When enabled, the particular steps performed to clean a node depend on which driver that node is managed by; see the individual drivers documentation for details. NOTE: The introduction of the cleaning operation causes instance deletion to take significantly longer. In an environment where all tenants are trusted (eg, because there is only one tenant), this option could be safely disabled.

allow_provisioning_in_maintenance

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Whether to allow nodes to enter or undergo deploy or cleaning when in maintenance mode. If this option is set to False, and a node enters maintenance during deploy or cleaning, the process will be aborted after the next heartbeat. Automated cleaning or making a node available will also fail. If True (the default), the process will begin and will pause after the node starts heartbeating. Moving it from maintenance will make the process continue.

clean_callback_timeout

Type integer

Default `1800`

Minimum Value 0

Timeout (seconds) to wait for a callback from the ramdisk doing the cleaning. If the timeout is reached the node will be put in the clean failed provision state. Set to 0 to disable timeout.

rescue_callback_timeout

Type integer

Default `1800`

Minimum Value 0

Timeout (seconds) to wait for a callback from the rescue ramdisk. If the timeout is reached the node will be put in the rescue failed provision state. Set to 0 to disable timeout.

soft_power_off_timeout

Type integer

Default `600`

Minimum Value 1

Mutable This option can be changed without restarting.

Timeout (in seconds) of soft reboot and soft power off operation. This value always has to be positive.

`power_state_change_timeout`

Type integer

Default `60`

Minimum Value 2

Mutable This option can be changed without restarting.

Number of seconds to wait for power operations to complete, i.e., so that a baremetal node is in the desired power state. If timed out, the power operation is considered a failure.

`power_failure_recovery_interval`

Type integer

Default `300`

Minimum Value 0

Interval (in seconds) between checking the power state for nodes previously put into maintenance mode due to power synchronization failure. A node is automatically moved out of maintenance mode once its power state is retrieved successfully. Set to 0 to disable this check.

`conductor_group`

Type string

Default `''`

Name of the conductor group to join. Can be up to 255 characters and is case insensitive. This conductor will only manage nodes with a matching conductor_group field set on the node.

`allow_deleting_available_nodes`

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Allow deleting nodes which are in state available. Defaults to True.

`enable_mdns`

Type boolean

Default `False`

Whether to enable publishing the baremetal API endpoint via multicast DNS.

`deploy_kernel`

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Glance ID, http:// or file:// URL of the kernel of the default deploy image.

`deploy_ramdisk`

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Glance ID, http:// or file:// URL of the initramfs of the default deploy image.

`rescue_kernel`

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Glance ID, http:// or file:// URL of the kernel of the default rescue image.

`rescue_ramdisk`

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Glance ID, http:// or file:// URL of the initramfs of the default rescue image.

`rescue_password_hash_algorithm`

Type string

Default `sha256`

Valid Values sha256, sha512

Mutable This option can be changed without restarting.

Password hash algorithm to be used for the rescue password.

`require_rescue_password_hashed`

Type boolean

Default `False`

Mutable This option can be changed without restarting.

Option to cause the conductor to not fallback to an un-hashed version of the rescue password, permitting rescue with older ironic-python-agent ramdisks.

`bootloader`

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Glance ID, http:// or file:// URL of the EFI system partition image containing EFI boot loader. This image will be used by ironic when building UEFI-bootable ISO out of kernel and ramdisk. Required for UEFI boot from partition images.

`clean_step_priority_override`

Type unknown type

Default `{}`

Priority to run automated clean steps for both in-band and out of band clean steps, provided in interface.step_name:priority format, e.g. deploy.erase_devices_metadata:123. The option can be specified multiple times to define priorities for multiple steps. If set to 0, this specific step will not run during cleaning. If unset for an inband clean step, will use the priority set in the ramdisk.

`disable_deep_image_inspection`

Type boolean

Default `False`

Security Option to permit an operator to disable file content inspections. Under normal conditions, the conductor will inspect requested image contents which are transferred through the conductor. Disabling this option is not advisable and opens the risk of unsafe images being processed which may allow an attacker to leverage unsafe features in various disk image formats to perform a variety of unsafe and potentially compromising actions. This option is *not* mutable, and requires a service restart to change.

`conductor_always_validates_images`

Type boolean

Default `True`

Security Option to enable the conductor to *always* inspect the image content of any requested deploy, even if the deployment would have normally bypassed the conductors cache. When this is set to False, the Ironic-Python-Agent is responsible for any necessary image checks. Setting this to True will result in a higher utilization of resources (disk space, network traffic) as the conductor will evaluate *all* images. This option is *not* mutable, and requires a service restart to change. This option requires [conductor]disable_deep_image_inspection to be set to False.

`permitted_image_formats`

Type list

Default `['raw', 'qcow2', 'iso']`

Mutable This option can be changed without restarting.

The supported list of image formats which are permitted for deployment with Ironic. If an image format outside of this list is detected, the image validation logic will fail the deployment process.

`disable_file_checksum`

Type boolean

Default `False`

Deprecated Security option: In the default case, image files have their checksums verified before undergoing additional conductor side actions such as image conversion. Enabling this option opens the risk of files being replaced at the source without the users knowledge.

Warning: This option is deprecated for removal. Its value may be silently ignored in the future.

`disable_support_for_checksum_files`

Type boolean

Default `False`

Security option: By default Ironic will attempt to retrieve a remote checksum file via HTTP(S) URL in order to validate an image download. This is functionality aligning with ironic-python-agent support for standalone users. Disabling this functionality by setting this option to True will create a more secure environment, however it may break users in an unexpected fashion.

console

`terminal`

Type string

Default `shellinaboxd`

Path to serial console terminal program. Used only by Shell In A Box console.

`terminal_cert_dir`

Type string

Default `<None>`

Directory containing the terminal SSL cert (PEM) for serial console access. Used only by Shell In A Box console.

`terminal_pid_dir`

Type string

Default `<None>`

Directory for holding terminal pid files. If not specified, the temporary directory will be used.

`terminal_timeout`

Type integer

Default `600`

Minimum Value 0

Timeout (in seconds) for the terminal session to be closed on inactivity. Set to 0 to disable timeout. Used only by Socat console.

`subprocess_checking_interval`

Type integer

Default `1`

Time interval (in seconds) for checking the status of console subprocess.

`subprocess_timeout`

Type integer

Default `10`

Time (in seconds) to wait for the console subprocess to start.

`kill_timeout`

Type integer

Default `1`

Time (in seconds) to wait for the console subprocess to exit before sending SIGKILL signal.

`socat_address`

Type ip address

Default `$my_ip`

IP address of Socat service running on the host of ironic conductor. Used only by Socat console.

`port_range`

Type string

Default `10000:20000`

This option has a sample default set, which means that its actual default value may vary from the one documented above.

A range of ports available to be used for the console proxy service running on the host of ironic conductor, in the form of <start>:<stop>. This option is used by both Shellinabox and Socat console

cors

`allowed_origin`

Type list

Default `<None>`

Indicate whether this resource may be shared with the domain received in the requests origin header. Format: <protocol>://<host>[:<port>], no trailing slash. Example: https://horizon.example.com

`allow_credentials`

Type boolean

Default `True`

Indicate that the actual request can include user credentials

`expose_headers`

Type list

Default `[]`

Indicate which headers are safe to expose to the API. Defaults to HTTP Simple Headers.

`max_age`

Type integer

Default `3600`

Maximum cache age of CORS preflight requests.

`allow_methods`

Type list

Default `['OPTIONS', 'GET', 'HEAD', 'POST', 'PUT', 'DELETE', 'TRACE', 'PATCH']`

Indicate which methods can be used during the actual request.

`allow_headers`

Type list

Default `[]`

Indicate which header field names may be used during the actual request.

database

`sqlite_synchronous`

Type boolean

Default `True`

If True, SQLite uses synchronous mode.

Table 10: Deprecated Variations

Group	Name
DEFAULT	sqlite_synchronous

`backend`

Type string

Default `sqlalchemy`

The back end to use for the database.

Table 11: Deprecated Variations

Group	Name
DEFAULT	db_backend

`connection`

Type string

Default `<None>`

The SQLAlchemy connection string to use to connect to the database.

Table 12: Deprecated Variations

Group	Name
DEFAULT	sql_connection
DATABASE	sql_connection
sql	connection

`slave_connection`

Type string

Default `<None>`

The SQLAlchemy connection string to use to connect to the slave database.

`mysql_sql_mode`

Type string

Default `TRADITIONAL`

The SQL mode to be used for MySQL sessions. This option, including the default, overrides any server-set SQL mode. To use whatever SQL mode is set by the server configuration, set this to no value. Example: mysql_sql_mode=

`mysql_enable_ndb`

Type boolean

Default `False`

If True, transparently enables support for handling MySQL Cluster (NDB).

`connection_recycle_time`

Type integer

Default `3600`

Connections which have been present in the connection pool longer than this number of seconds will be replaced with a new one the next time they are checked out from the pool.

Table 13: Deprecated Variations

Group	Name
DATABASE	idle_timeout
database	idle_timeout
DEFAULT	sql_idle_timeout
DATABASE	sql_idle_timeout
sql	idle_timeout

`max_pool_size`

Type integer

Default `5`

Maximum number of SQL connections to keep open in a pool. Setting a value of 0 indicates no limit.

Table 14: Deprecated Variations

Group	Name
DEFAULT	sql_max_pool_size
DATABASE	sql_max_pool_size

`max_retries`

Type integer

Default `10`

Maximum number of database connection retries during startup. Set to -1 to specify an infinite retry count.

Table 15: Deprecated Variations

Group	Name
DEFAULT	sql_max_retries
DATABASE	sql_max_retries

`retry_interval`

Type integer

Default `10`

Interval between retries of opening a SQL connection.

Table 16: Deprecated Variations

Group	Name
DEFAULT	sql_retry_interval
DATABASE	reconnect_interval

`max_overflow`

Type integer

Default `50`

If set, use this value for max_overflow with SQLAlchemy.

Table 17: Deprecated Variations

Group	Name
DEFAULT	sql_max_overflow
DATABASE	sqlalchemy_max_overflow

`connection_debug`

Type integer

Default `0`

Minimum Value 0

Maximum Value 100

Verbosity of SQL debugging information: 0=None, 100=Everything.

Table 18: Deprecated Variations

Group	Name
DEFAULT	sql_connection_debug

connection_trace

Type boolean

Default `False`

Add Python stack traces to SQL as comment strings.

Table 19: Deprecated Variations

Group	Name
DEFAULT	sql_connection_trace

pool_timeout

Type integer

Default `<None>`

If set, use this value for pool_timeout with SQLAlchemy.

Table 20: Deprecated Variations

Group	Name
DATABASE	sqlalchemy_pool_timeout

use_db_reconnect

Type boolean

Default `False`

Enable the experimental use of database reconnect on connection lost.

db_retry_interval

Type integer

Default `1`

Seconds between retries of a database transaction.

db_inc_retry_interval

Type boolean

Default `True`

If True, increases the interval between retries of a database operation up to db_max_retry_interval.

db_max_retry_interval

Type integer

Default `10`

If db_inc_retry_interval is set, the maximum seconds between retries of a database operation.

`db_max_retries`

Type integer

Default `20`

Maximum retries in case of connection error or deadlock error before error is raised. Set to -1 to specify an infinite retry count.

`connection_parameters`

Type string

Default `''`

Optional URL parameters to append onto the connection URL at connect time; specify as param1=value1&param2=value2&

`mysql_engine`

Type string

Default `InnoDB`

MySQL engine to use.

deploy

`http_url`

Type string

Default `<None>`

ironic-conductor nodes HTTP server URL. Example: http://192.1.2.3:8080

`http_root`

Type string

Default `/httpboot`

ironic-conductor nodes HTTP root path.

`external_http_url`

Type string

Default `<None>`

URL of the ironic-conductor nodes HTTP server for boot methods such as virtual media, where images could be served outside of the provisioning network. Does not apply when Swift is used. Defaults to http_url.

`external_callback_url`

Type string

Default `<None>`

Agent callback URL of the bare metal API for boot methods such as virtual media, where images could be served outside of the provisioning network. Defaults to the configuration from [service_catalog].

`enable_ata_secure_erase`

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Whether to support the use of ATA Secure Erase during the cleaning process. Defaults to True.

`enable_nvme_secure_erase`

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Whether to support the use of NVMe Secure Erase during the cleaning process. Currently nvme-cli format command is supported with user-data and crypto modes, depending on device capabilities.Defaults to True.

`erase_devices_priority`

Type integer

Default `<None>`

Mutable This option can be changed without restarting.

Priority to run in-band erase devices via the Ironic Python Agent ramdisk. If unset, will use the priority set in the ramdisk (defaults to 10 for the GenericHardwareManager). If set to 0, will not run during cleaning.

`erase_devices_metadata_priority`

Type integer

Default `<None>`

Mutable This option can be changed without restarting.

Priority to run in-band clean step that erases metadata from devices, via the Ironic Python Agent ramdisk. If unset, will use the priority set in the ramdisk (defaults to 99 for the GenericHardwareManager). If set to 0, will not run during cleaning.

`delete_configuration_priority`

Type integer

Default `<None>`

Mutable This option can be changed without restarting.

Priority to run in-band clean step that erases RAID configuration from devices, via the Ironic Python Agent ramdisk. If unset, will use the priority set in the ramdisk (defaults to 0 for the GenericHardwareManager). If set to 0, will not run during cleaning.

`create_configuration_priority`

Type integer

Default `<None>`

Mutable This option can be changed without restarting.

Priority to run in-band clean step that creates RAID configuration from devices, via the Ironic Python Agent ramdisk. If unset, will use the priority set in the ramdisk (defaults to 0 for the GenericHardwareManager). If set to 0, will not run during cleaning.

`shred_random_overwrite_iterations`

Type integer

Default `1`

Minimum Value 0

Mutable This option can be changed without restarting.

During shred, overwrite all block devices N times with random data. This is only used if a device could not be ATA Secure Erased. Defaults to 1.

`shred_final_overwrite_with_zeros`

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Whether to write zeros to a nodes block devices after writing random data. This will write zeros to the device even when deploy.shred_random_overwrite_iterations is 0. This option is only used if a device could not be ATA Secure Erased. Defaults to True.

`continue_if_disk_secure_erase_fails`

Type boolean

Default `False`

Mutable This option can be changed without restarting.

Defines what to do if a secure erase operation (NVMe or ATA) fails during cleaning in the Ironic Python Agent. If False, the cleaning operation will fail and the node will be put in `clean failed` state. If True, shred will be invoked and cleaning will continue.

`disk_erasure_concurrency`

Type integer

Default `1`

Minimum Value 1

Mutable This option can be changed without restarting.

Defines the target pool size used by Ironic Python Agent ramdisk to erase disk devices. The number of threads created to erase disks will not exceed this value or the number of disks to be erased.

`power_off_after_deploy_failure`

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Whether to power off a node after deploy failure. Defaults to True.

`default_boot_option`

Type string

Default `local`

Valid Values netboot, local

Mutable This option can be changed without restarting.

Default boot option to use when no boot option is requested in nodes driver_info. Defaults to local. Prior to the Ussuri release, the default was netboot.

Possible values

netboot boot from a network

local local boot

`default_boot_mode`

Type string

Default `bios`

Valid Values uefi, bios

Mutable This option can be changed without restarting.

Default boot mode to use when no boot mode is requested in nodes driver_info, capabilities or in the *instance_info* configuration. Currently the default boot mode is bios, but it will be changed to uefi in the future. It is recommended to set an explicit value for this option. This option only has effect when management interface supports boot mode management

Possible values

uefi UEFI boot mode

bios Legacy BIOS boot mode

`configdrive_use_object_store`

Type boolean

Default `False`

Mutable This option can be changed without restarting.

Whether to upload the config drive to object store. Set this option to True to store config drive in a swift endpoint.

Table 21: Deprecated Variations

Group	Name
conductor	configdrive_use_swift

`http_image_subdir`

Type string

Default `agent_images`

The name of subdirectory under ironic-conductor nodes HTTP root path which is used to place instance images for the direct deploy interface, when local HTTP service is incorporated to provide instance image instead of swift tempurls.

`fast_track`

Type boolean

Default `False`

Mutable This option can be changed without restarting.

Whether to allow deployment agents to perform lookup, heartbeat operations during initial states of a machine lifecycle and by-pass the normal setup procedures for a ramdisk. This feature also enables power operations which are part of deployment processes to be bypassed if the ramdisk has performed a heartbeat operation using the fast_track_timeout setting.

`fast_track_timeout`

Type integer

Default `300`

Minimum Value 0

Maximum Value 300

Mutable This option can be changed without restarting.

Seconds for which the last heartbeat event is to be considered valid for the purpose of a fast track sequence. This setting should generally be less than the number of seconds for Power-On Self Test and typical ramdisk start-up. This value should not exceed the [api]ramdisk_heartbeat_timeout setting.

`erase_skip_read_only`

Type boolean

Default `False`

Mutable This option can be changed without restarting.

If the ironic-python-agent should skip read-only devices when running the erase_devices clean step where block devices are zeroed out. This requires ironic-python-agent 6.0.0 or greater. By default a read-only device will cause non-metadata based cleaning operations to fail due to the possible operational security risk of data being retained between deployments of the bare metal node.

`ramdisk_image_download_source`

Type string

Default `local`

Valid Values http, local

Mutable This option can be changed without restarting.

Specifies whether a boot iso image should be served from its own original location using the image source url directly, or if ironic should cache the image on the conductor and serve it from ironics own http server.

Possible values

http In case the ramdisk is already a bootable iso, using this option it will be directly provided by an external HTTP service using its full url.

local This is the default behavior. The image is downloaded, prepared and cached locally, to be served from the conductor.

dhcp

`dhcp_provider`

Type string

Default `neutron`

DHCP provider to use. neutron uses Neutron, and none uses a no-op provider.

disk_partitioner

`check_device_interval`

Type integer

Default `1`

After Ironic has completed creating the partition table, it continues to check for activity on the attached iSCSI device status at this interval prior to copying the image to the node, in seconds

`check_device_max_retries`

Type integer

Default `20`

The maximum number of times to check that the device is not accessed by another process. If the device is still busy after that, the disk partitioning will be treated as having failed.

disk_utils

`efi_system_partition_size`

Type integer

Default `200`

Size of EFI system partition in MiB when configuring UEFI systems for local boot.

`bios_boot_partition_size`

Type integer

Default `1`

Size of BIOS Boot partition in MiB when configuring GPT partitioned systems for local boot in BIOS.

`dd_block_size`

Type string

Default `1M`

Block size to use when writing to the nodes disk.

`partition_detection_attempts`

Type integer

Default `3`

Minimum Value 1

Maximum attempts to detect a newly created partition.

`partprobe_attempts`

Type integer

Default `10`

Maximum number of attempts to try to read the partition.

`image_convert_memory_limit`

Type integer

Default `2048`

Memory limit for qemu-img convert in MiB. Implemented via the address space resource limit.

`image_convert_attempts`

Type integer

Default `3`

Number of attempts to convert an image.

`image_convert_memory_limit`

Type integer

Default `2048`

Memory limit for qemu-img convert in MiB. Implemented via the address space resource limit.

`image_convert_attempts`

Type integer

Default `3`

Number of attempts to convert an image.

drac

query_raid_config_job_status_interval

Type integer

Default `120`

Minimum Value 1

Interval (in seconds) between periodic RAID job status checks to determine whether the asynchronous RAID configuration was successfully finished or not.

boot_device_job_status_timeout

Type integer

Default `30`

Minimum Value 1

Maximum amount of time (in seconds) to wait for the boot device configuration job to transition to the correct state to allow a reboot or power on to complete.

config_job_max_retries

Type integer

Default `240`

Minimum Value 1

Maximum number of retries for the configuration job to complete successfully.

query_import_config_job_status_interval

Type integer

Default `60`

Minimum Value 0

Number of seconds to wait between checking for completed import configuration task

bios_factory_reset_timeout

Type integer

Default `600`

Minimum Value 1

Maximum time (in seconds) to wait for factory reset of BIOS settings to complete.

glance

allowed_direct_url_schemes

Type list

Default `[]`

A list of URL schemes that can be downloaded directly via the direct_url. Currently supported schemes: [file].

auth_url

Type unknown type

Default `<None>`

Authentication URL

auth_type

Type unknown type

Default `<None>`

Authentication type to load

Table 22: Deprecated Variations

Group	Name
glance	auth_plugin

cafile

Type string

Default `<None>`

PEM encoded Certificate Authority to use when verifying HTTPs connections.

certfile

Type string

Default `<None>`

PEM encoded client certificate cert file

collect_timing

Type boolean

Default `False`

Collect per-API call timing information.

connect_retries

Type integer

Default `<None>`

The maximum number of retries that should be attempted for connection errors.

connect_retry_delay

Type floating point

Default `<None>`

Delay (in seconds) between two retries for connection errors. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`default_domain_id`

Type unknown type

Default `<None>`

Optional domain ID to use with v3 and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`default_domain_name`

Type unknown type

Default `<None>`

Optional domain name to use with v3 API and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`domain_id`

Type unknown type

Default `<None>`

Domain ID to scope to

`domain_name`

Type unknown type

Default `<None>`

Domain name to scope to

`endpoint_override`

Type string

Default `<None>`

Always use this endpoint URL for requests for this client. NOTE: The unversioned endpoint should be specified here; to request a particular API version, use the *version*, *min-version*, and/or *max-version* options.

`insecure`

Type boolean

Default `False`

Verify HTTPS connections.

`keyfile`

Type string

Default `<None>`

PEM encoded client certificate key file

max_version

Type string

Default `<None>`

The maximum major version of a given API, intended to be used as the upper bound of a range with min_version. Mutually exclusive with version.

min_version

Type string

Default `<None>`

The minimum major version of a given API, intended to be used as the lower bound of a range with max_version. Mutually exclusive with version. If min_version is given with no max_version it is as if max version is latest.

num_retries

Type integer

Default `0`

Number of retries when downloading an image from glance.

password

Type unknown type

Default `<None>`

Users password

project_domain_id

Type unknown type

Default `<None>`

Domain ID containing project

project_domain_name

Type unknown type

Default `<None>`

Domain name containing project

project_id

Type unknown type

Default `<None>`

Project ID to scope to

Table 23: Deprecated Variations

Group	Name
glance	tenant-id
glance	tenant_id

`project_name`

Type unknown type

Default `<None>`

Project name to scope to

Table 24: Deprecated Variations

Group	Name
glance	tenant-name
glance	tenant_name

`region_name`

Type string

Default `<None>`

The default region_name for endpoint URL discovery.

`service_name`

Type string

Default `<None>`

The default service_name for endpoint URL discovery.

`service_type`

Type string

Default `image`

The default service_type for endpoint URL discovery.

`split_loggers`

Type boolean

Default `False`

Log requests to multiple loggers.

`status_code_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for retriable HTTP status codes.

`status_code_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for retriable status codes. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`swift_account`

Type string

Default `<None>`

The account that Glance uses to communicate with Swift. The format is AUTH_uuid. uuid is the UUID for the account configured in the glance-api.conf. For example: AUTH_a422b2-91f3-2f46-74b7-d7c9e8958f5d30. If not set, the default value is calculated based on the ID of the project used to access Swift (as set in the [swift] section). Swift temporary URL format: endpoint_url/api_version/account/container/object_id

swift_api_version

Type string

Default `v1`

The Swift API version to create a temporary URL for. Defaults to v1. Swift temporary URL format: endpoint_url/api_version/account/container/object_id

swift_container

Type string

Default `glance`

The Swift container Glance is configured to store its images in. Defaults to glance, which is the default in glance-api.conf. Swift temporary URL format: endpoint_url/api_version/account/container/object_id

swift_endpoint_url

Type string

Default `<None>`

The endpoint (scheme, hostname, optional port) for the Swift URL of the form endpoint_url/api_version/account/container/object_id. Do not include trailing /. For example, use https://swift.example.com. If using RADOS Gateway, endpoint may also contain /swift path; if it does not, it will be appended. Used for temporary URLs, will be fetched from the service catalog, if not provided.

swift_store_multiple_containers_seed

Type integer

Default `0`

This should match a config by the same name in the Glance configuration file. When set to 0, a single-tenant store will only use one container to store all images. When set to an integer value between 1 and 32, a single-tenant store will use multiple containers to store images, and this value will determine how many containers are created.

swift_temp_url_cache_enabled

Type boolean

Default `False`

Whether to cache generated Swift temporary URLs. Setting it to true is only useful when an image caching proxy is used. Defaults to False.

swift_temp_url_duration

Type integer

Default `1200`

The length of time in seconds that the temporary URL will be valid for. Defaults to 20 minutes. If some deploys get a 401 response code when trying to download from the temporary URL, try raising this duration. This value must be greater than or equal to the value for swift_temp_url_expected_download_start_delay

swift_temp_url_expected_download_start_delay

Type integer

Default `0`

Minimum Value 0

This is the delay (in seconds) from the time of the deploy request (when the Swift temporary URL is generated) to when the IPA ramdisk starts up and URL is used for the image download. This value is used to check if the Swift temporary URL duration is large enough to let the image download begin. Also if temporary URL caching is enabled this will determine if a cached entry will still be valid when the download starts. swift_temp_url_duration value must be greater than or equal to this options value. Defaults to 0.

swift_temp_url_key

Type string

Default `<None>`

The secret token given to Swift to allow temporary URL downloads. Required for temporary URLs. For the Swift backend, the key on the service project (as set in the [swift] section) is used by default.

system_scope

Type unknown type

Default `<None>`

Scope for system operations

tenant_id

Type unknown type

Default `<None>`

Tenant ID

tenant_name

Type unknown type

Default `<None>`

Tenant Name

timeout

Type integer

Default `<None>`

Timeout value for http requests

trust_id

Type unknown type

Default `<None>`

Trust ID

`user_domain_id`

Type unknown type

Default `<None>`

Users domain id

`user_domain_name`

Type unknown type

Default `<None>`

Users domain name

`user_id`

Type unknown type

Default `<None>`

User id

`username`

Type unknown type

Default `<None>`

Username

Table 25: Deprecated Variations

Group	Name
glance	user-name
glance	user_name

`valid_interfaces`

Type list

Default `['internal', 'public']`

List of interfaces, in order of preference, for endpoint URL.

`version`

Type string

Default `<None>`

Minimum Major API version within a given Major API version for endpoint URL discovery. Mutually exclusive with min_version and max_version

healthcheck

path

Type string

Default `/healthcheck`

The path to respond to healtcheck requests on.

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.

detailed

Type boolean

Default `False`

Show more detailed information as part of the response. Security note: Enabling this option may expose sensitive details about the service being monitored. Be sure to verify that it will not violate your security policies.

backends

Type list

Default `[]`

Additional backends that can perform health checks and report that information back as part of a request.

disable_by_file_path

Type string

Default `<None>`

Check the presence of a file to determine if an application is running on a port. Used by DisableByFileHealthcheck plugin.

disable_by_file_paths

Type list

Default `[]`

Check the presence of a file based on a port to determine if an application is running on a port. Expects a port:path list of strings. Used by DisableByFilesPortsHealthcheck plugin.

enabled

Type boolean

Default `False`

Enable the health check endpoint at /healthcheck. Note that this is unauthenticated. More information is available at https://docs.openstack.org/oslo.middleware/latest/reference/healthcheck_plugins.html.

ilo

`client_timeout`

Type integer

Default `60`

Timeout (in seconds) for iLO operations

`client_port`

Type port number

Default `443`

Minimum Value 0

Maximum Value 65535

Port to be used for iLO operations

`swift_ilo_container`

Type string

Default `ironic_ilo_container`

The Swift iLO container to store data.

`swift_object_expiry_timeout`

Type integer

Default `900`

Amount of time in seconds for Swift objects to auto-expire.

`use_web_server_for_images`

Type boolean

Default `False`

Set this to True to use http web server to host floppy images and generated boot ISO. This requires http_root and http_url to be configured in the [deploy] section of the config file. If this is set to False, then Ironic will use Swift to host the floppy images and generated boot_iso.

`clean_priority_reset_ilo`

Type integer

Default `0`

Priority for reset_ilo clean step.

`clean_priority_reset_bios_to_default`

Type integer

Default `10`

Priority for reset_bios_to_default clean step.

`clean_priority_reset_secure_boot_keys_to_default`

Type integer

Default `20`

Priority for reset_secure_boot_keys clean step. This step will reset the secure boot keys to manufacturing defaults.

`clean_priority_clear_secure_boot_keys`

Type integer

Default `0`

Priority for clear_secure_boot_keys clean step. This step is not enabled by default. It can be enabled to clear all secure boot keys enrolled with iLO.

`clean_priority_reset_ilo_credential`

Type integer

Default `30`

Priority for reset_ilo_credential clean step. This step requires ilo_change_password parameter to be updated in nodess driver_info with the new password.

`power_wait`

Type integer

Default `2`

Amount of time in seconds to wait in between power operations

`oob_erase_devices_job_status_interval`

Type integer

Default `300`

Minimum Value 10

Interval (in seconds) between periodic erase-devices status checks to determine whether the asynchronous out-of-band erase-devices was successfully finished or not. On an average, a 300GB HDD with default pattern overwrite would take approximately 9 hours and 300GB SSD with default pattern block would take approx. 30 seconds to complete sanitize disk erase.

`ca_file`

Type string

Default `<None>`

CA certificate file to validate iLO.

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.
>
> **Reason** Its being replaced by new configuration parameter verify_ca.

`verify_ca`

Type string

Default `True`

CA certificate to validate iLO. This can be either a Boolean value, a path to a CA_BUNDLE file or directory with certificates of trusted CAs. If set to True the driver will verify the host certificates; if False the driver will ignore verifying the SSL certificate. If its a path the driver will use the specified certificate or one of the certificates in the directory. Defaults to True.

`default_boot_mode`

Type string

Default `auto`

Valid Values auto, bios, uefi

Default boot mode to be used in provisioning when boot_mode capability is not provided in the properties/capabilities of the node. The default is auto for backward compatibility. When auto is specified, default boot mode will be selected based on boot mode settings on the system.

Possible values

auto based on boot mode settings on the system

bios BIOS boot mode

uefi UEFI boot mode

`file_permission`

Type integer

Default `420`

File permission for swift-less image hosting with the octal permission representation of file access permissions. This setting defaults to `644`, or as the octal number `0o644` in Python. This setting must be set to the octal number representation, meaning starting with `0o`.

`kernel_append_params`

Type string

Default `nofb nomodeset vga=normal`

Mutable This option can be changed without restarting.

Additional kernel parameters to pass down to the instance kernel. These parameters can be consumed by the kernel or by the applications by reading /proc/cmdline. Mind severe cmdline size limit! Can be overridden by *instance_info/kernel_append_params* property.

inspector

`auth_url`

Type unknown type

Default `<None>`

Authentication URL

`auth_type`

Type unknown type

Default `<None>`

Authentication type to load

Table 26: Deprecated Variations

Group	Name
inspector	auth_plugin

cafile

Type string

Default `<None>`

PEM encoded Certificate Authority to use when verifying HTTPs connections.

callback_endpoint_override

Type string

Default `<None>`

endpoint to use as a callback for posting back introspection data when boot is managed by ironic. Standard keystoneauth options are used by default.

certfile

Type string

Default `<None>`

PEM encoded client certificate cert file

collect_timing

Type boolean

Default `False`

Collect per-API call timing information.

connect_retries

Type integer

Default `<None>`

The maximum number of retries that should be attempted for connection errors.

connect_retry_delay

Type floating point

Default `<None>`

Delay (in seconds) between two retries for connection errors. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

default_domain_id

Type unknown type

Default `<None>`

Optional domain ID to use with v3 and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

default_domain_name

Type unknown type

Default `<None>`

Optional domain name to use with v3 API and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

domain_id

Type unknown type

Default `<None>`

Domain ID to scope to

domain_name

Type unknown type

Default `<None>`

Domain name to scope to

endpoint_override

Type string

Default `<None>`

Always use this endpoint URL for requests for this client. NOTE: The unversioned endpoint should be specified here; to request a particular API version, use the *version*, *min-version*, and/or *max-version* options.

extra_kernel_params

Type string

Default `''`

extra kernel parameters to pass to the inspection ramdisk when boot is managed by ironic (not ironic-inspector). Pairs key=value separated by spaces.

insecure

Type boolean

Default `False`

Verify HTTPS connections.

keyfile

Type string

Default `<None>`

PEM encoded client certificate key file

max_version

Type string

Default `<None>`

The maximum major version of a given API, intended to be used as the upper bound of a range with min_version. Mutually exclusive with version.

`min_version`

Type string

Default `<None>`

The minimum major version of a given API, intended to be used as the lower bound of a range with max_version. Mutually exclusive with version. If min_version is given with no max_version it is as if max version is latest.

`password`

Type unknown type

Default `<None>`

Users password

`power_off`

Type boolean

Default `True`

whether to power off a node after inspection finishes

`project_domain_id`

Type unknown type

Default `<None>`

Domain ID containing project

`project_domain_name`

Type unknown type

Default `<None>`

Domain name containing project

`project_id`

Type unknown type

Default `<None>`

Project ID to scope to

Table 27: Deprecated Variations

Group	Name
inspector	tenant-id
inspector	tenant_id

`project_name`

Type unknown type

Default `<None>`

Project name to scope to

Table 28: Deprecated Variations

Group	Name
inspector	tenant-name
inspector	tenant_name

`region_name`

Type string

Default `<None>`

The default region_name for endpoint URL discovery.

`require_managed_boot`

Type boolean

Default `False`

require that the in-band inspection boot is fully managed by ironic. Set this to True if your installation of ironic-inspector does not have a separate PXE boot environment.

`service_name`

Type string

Default `<None>`

The default service_name for endpoint URL discovery.

`service_type`

Type string

Default `baremetal-introspection`

The default service_type for endpoint URL discovery.

`split_loggers`

Type boolean

Default `False`

Log requests to multiple loggers.

`status_code_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for retriable HTTP status codes.

`status_code_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for retriable status codes. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

status_check_period

Type integer

Default `60`

period (in seconds) to check status of nodes on inspection

system_scope

Type unknown type

Default `<None>`

Scope for system operations

tenant_id

Type unknown type

Default `<None>`

Tenant ID

tenant_name

Type unknown type

Default `<None>`

Tenant Name

timeout

Type integer

Default `<None>`

Timeout value for http requests

trust_id

Type unknown type

Default `<None>`

Trust ID

user_domain_id

Type unknown type

Default `<None>`

Users domain id

user_domain_name

Type unknown type

Default `<None>`

Users domain name

user_id

Type unknown type

Default `<None>`

User id

`username`

Type unknown type

Default `<None>`

Username

Table 29: Deprecated Variations

Group	Name
inspector	user-name
inspector	user_name

`valid_interfaces`

Type list

Default `['internal', 'public']`

List of interfaces, in order of preference, for endpoint URL.

`version`

Type string

Default `<None>`

Minimum Major API version within a given Major API version for endpoint URL discovery. Mutually exclusive with min_version and max_version

ipmi

`command_retry_timeout`

Type integer

Default `60`

Mutable This option can be changed without restarting.

Maximum time in seconds to retry retryable IPMI operations. (An operation is retryable, for example, if the requested operation fails because the BMC is busy.) Setting this too high can cause the sync power state periodic task to hang when there are slow or unresponsive BMCs.

`min_command_interval`

Type integer

Default `5`

Mutable This option can be changed without restarting.

Minimum time, in seconds, between IPMI operations sent to a server. There is a risk with some hardware that setting this too low may cause the BMC to crash. Recommended setting is 5 seconds.

use_ipmitool_retries

Type boolean

Default `False`

When set to True and the parameters are supported by ipmitool, the number of retries and the retry interval are passed to ipmitool as parameters, and ipmitool will do the retries. When set to False, ironic will retry the ipmitool commands. Recommended setting is False

kill_on_timeout

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Kill *ipmitool* process invoked by ironic to read node power state if *ipmitool* process does not exit after *command_retry_timeout* timeout expires. Recommended setting is True

disable_boot_timeout

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Default timeout behavior whether ironic sends a raw IPMI command to disable the 60 second timeout for booting. Setting this option to False will NOT send that command, the default value is True. It may be overridden by per-node ipmi_disable_boot_timeout option in nodes driver_info field.

additional_retryable_ipmi_errors

Type multi-valued

Default `''`

Mutable This option can be changed without restarting.

Additional errors ipmitool may encounter, specific to the environment it is run in.

debug

Type boolean

Default `False`

Mutable This option can be changed without restarting.

Enables all ipmi commands to be executed with an additional debugging output. This is a separate option as ipmitool can log a substantial amount of misleading text when in this mode.

cipher_suite_versions

Type list

Default `[]`

List of possible cipher suites versions that can be supported by the hardware in case the field *cipher_suite* is not set for the node.

irmc

`remote_image_share_root`

Type string

Default `/remote_image_share_root`

Ironic conductor nodes NFS or CIFS root path

`remote_image_server`

Type string

Default `<None>`

IP of remote image server

`remote_image_share_type`

Type string

Default `CIFS`

Valid Values CIFS, NFS

Share type of virtual media

Possible values

CIFS CIFS (Common Internet File System) protocol

NFS NFS (Network File System) protocol

`remote_image_share_name`

Type string

Default `share`

share name of remote_image_server

`remote_image_user_name`

Type string

Default `<None>`

User name of remote_image_server

`remote_image_user_password`

Type string

Default `<None>`

Password of remote_image_user_name

`remote_image_user_domain`

Type string

Default `''`

Domain name of remote_image_user_name

`port`

Type port number

Default `443`

Minimum Value 0

Maximum Value 65535

Valid Values 443, 80

Port to be used for iRMC operations

Possible values

443 port 443

80 port 80

`auth_method`

Type string

Default `basic`

Valid Values basic, digest

Authentication method to be used for iRMC operations

Possible values

basic Basic authentication

digest Digest authentication

`client_timeout`

Type integer

Default `60`

Timeout (in seconds) for iRMC operations

`sensor_method`

Type string

Default `ipmitool`

Valid Values ipmitool, scci

Sensor data retrieval method.

Possible values

ipmitool IPMItool

scci Fujitsu SCCI (ServerView Common Command Interface)

`snmp_version`

Type string

Default `v2c`

Valid Values v1, v2c, v3

SNMP protocol version

Possible values

v1 SNMPv1

v2c SNMPv2c

v3 SNMPv3

`snmp_port`

Type port number

Default `161`

Minimum Value 0

Maximum Value 65535

SNMP port

`snmp_community`

Type string

Default `public`

SNMP community. Required for versions v1 and v2c

`snmp_security`

Type string

Default `<None>`

SNMP security name. Required for version v3. Will be ignored if driver_info/irmc_snmp_user is set.

`snmp_polling_interval`

Type integer

Default `10`

SNMP polling interval in seconds

`snmp_auth_proto`

Type string

Default `sha`

Valid Values sha, sha256, sha384, sha512

SNMPv3 message authentication protocol ID. Required for version v3. Will be ignored if the version of python-scciclient is before 0.10.1. The valid options are sha, sha256, sha384 and sha512, while sha is the only supported protocol in iRMC S4 and S5, and from iRMC S6, sha256, sha384 and sha512 are supported, but sha is not supported any more.

Possible values

sha Secure Hash Algorithm 1, supported in iRMC S4 and S5.

sha256 Secure Hash Algorithm 2 with 256 bits digest, only supported in iRMC S6.

sha384 Secure Hash Algorithm 2 with 384 bits digest, only supported in iRMC S6.

sha512 Secure Hash Algorithm 2 with 512 bits digest, only supported in iRMC S6.

`snmp_priv_proto`

Type string

Default `aes`

Valid Values aes

SNMPv3 message privacy (encryption) protocol ID. Required for version v3. Will be ignored if the version of python-scciclient is before 0.10.1. aes is supported.

Possible values

aes Advanced Encryption Standard

`clean_priority_restore_irmc_bios_config`

Type integer

Default `0`

Priority for restore_irmc_bios_config clean step.

`gpu_ids`

Type list

Default `[]`

List of vendor IDs and device IDs for GPU device to inspect. List items are in format vendorID/deviceID and separated by commas. GPU inspection will use this value to count the number of GPU device in a node. If this option is not defined, then leave out pci_gpu_devices in capabilities property. Sample gpu_ids value: 0x1000/0x0079,0x2100/0x0080

`fpga_ids`

Type list

Default `[]`

List of vendor IDs and device IDs for CPU FPGA to inspect. List items are in format vendorID/deviceID and separated by commas. CPU inspection will use this value to find existence of CPU FPGA in a node. If this option is not defined, then leave out CUSTOM_CPU_FPGA in node traits. Sample fpga_ids value: 0x1000/0x0079,0x2100/0x0080

query_raid_config_fgi_status_interval

Type integer

Default `300`

Minimum Value 1

Interval (in seconds) between periodic RAID status checks to determine whether the asynchronous RAID configuration was successfully finished or not. Foreground Initialization (FGI) will start 5 minutes after creating virtual drives.

ironic_lib

fatal_exception_format_errors

Type boolean

Default `False`

Used if there is a formatting error when generating an exception message (a programming error). If True, raise an exception; if False, use the unformatted message.

Table 30: Deprecated Variations

Group	Name
DEFAULT	fatal_exception_format_errors

root_helper

Type string

Default `sudo ironic-rootwrap /etc/ironic/rootwrap.conf`

Command that is prefixed to commands that are run as root. If not specified, no commands are run as root.

iscsi

portal_port

Type port number

Default `3260`

Minimum Value 0

Maximum Value 65535

Mutable This option can be changed without restarting.

The port number on which the iSCSI portal listens for incoming connections.

conv_flags

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Flags that need to be sent to the dd command, to control the conversion of the original file when copying to the host. It can contain several options separated by commas.

`verify_attempts`

Type integer

Default `3`

Minimum Value 1

Mutable This option can be changed without restarting.

Maximum attempts to verify an iSCSI connection is active, sleeping 1 second between attempts. Defaults to 3.

json_rpc

`auth_strategy`

Type string

Default `<None>`

Valid Values noauth, keystone, http_basic

Authentication strategy used by JSON RPC. Defaults to the global auth_strategy setting.

Possible values

noauth no authentication

keystone use the Identity service for authentication

http_basic HTTP basic authentication

`http_basic_auth_user_file`

Type string

Default `/etc/ironic/htpasswd-json-rpc`

Path to Apache format user authentication file used when auth_strategy=http_basic

`host_ip`

Type host address

Default `::`

The IP address or hostname on which JSON RPC will listen.

`port`

Type port number

Default `8089`

Minimum Value 0

Maximum Value 65535

The port to use for JSON RPC

`use_ssl`

Type boolean

Default `False`

Whether to use TLS for JSON RPC

`http_basic_username`

Type string

Default `<None>`

Name of the user to use for HTTP Basic authentication client requests.

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.
>
> **Reason** Use username instead

`http_basic_password`

Type string

Default `<None>`

Password to use for HTTP Basic authentication client requests.

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.
>
> **Reason** Use password instead

`auth_url`

Type unknown type

Default `<None>`

Authentication URL

`auth_type`

Type unknown type

Default `<None>`

Authentication type to load

Table 31: Deprecated Variations

Group	Name
json_rpc	auth_plugin

`cafile`

Type string

Default `<None>`

PEM encoded Certificate Authority to use when verifying HTTPs connections.

`certfile`

Type string

Default `<None>`

PEM encoded client certificate cert file

`collect_timing`

Type boolean

Default `False`

Collect per-API call timing information.

`default_domain_id`

Type unknown type

Default `<None>`

Optional domain ID to use with v3 and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`default_domain_name`

Type unknown type

Default `<None>`

Optional domain name to use with v3 API and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`domain_id`

Type unknown type

Default `<None>`

Domain ID to scope to

`domain_name`

Type unknown type

Default `<None>`

Domain name to scope to

`insecure`

Type boolean

Default `False`

Verify HTTPS connections.

`keyfile`

Type string

Default `<None>`

PEM encoded client certificate key file

`password`

Type unknown type

Default `<None>`

Users password

`project_domain_id`

Type unknown type

Default `<None>`

Domain ID containing project

`project_domain_name`

Type unknown type

Default `<None>`

Domain name containing project

`project_id`

Type unknown type

Default `<None>`

Project ID to scope to

Table 32: Deprecated Variations

Group	Name
json_rpc	tenant-id
json_rpc	tenant_id

`project_name`

Type unknown type

Default `<None>`

Project name to scope to

Table 33: Deprecated Variations

Group	Name
json_rpc	tenant-name
json_rpc	tenant_name

split_loggers

Type boolean

Default `False`

Log requests to multiple loggers.

system_scope

Type unknown type

Default `<None>`

Scope for system operations

tenant_id

Type unknown type

Default `<None>`

Tenant ID

tenant_name

Type unknown type

Default `<None>`

Tenant Name

timeout

Type integer

Default `<None>`

Timeout value for http requests

trust_id

Type unknown type

Default `<None>`

Trust ID

user_domain_id

Type unknown type

Default `<None>`

Users domain id

user_domain_name

Type unknown type

Default `<None>`

Users domain name

user_id

Type unknown type

Default `<None>`

User id

`username`

Type unknown type

Default `<None>`

Username

Table 34: Deprecated Variations

Group	Name
json_rpc	user-name
json_rpc	user_name

keystone_authtoken

`www_authenticate_uri`

Type string

Default `<None>`

Complete public Identity API endpoint. This endpoint should not be an admin endpoint, as it should be accessible by all end users. Unauthenticated clients are redirected to this endpoint to authenticate. Although this endpoint should ideally be unversioned, client support in the wild varies. If youre using a versioned v2 endpoint here, then this should *not* be the same endpoint the service user utilizes for validating tokens, because normal end users may not be able to reach that endpoint.

Table 35: Deprecated Variations

Group	Name
keystone_authtoken	auth_uri

`auth_uri`

Type string

Default `<None>`

Complete public Identity API endpoint. This endpoint should not be an admin endpoint, as it should be accessible by all end users. Unauthenticated clients are redirected to this endpoint to authenticate. Although this endpoint should ideally be unversioned, client support in the wild varies. If youre using a versioned v2 endpoint here, then this should *not* be the same endpoint the service user utilizes for validating tokens, because normal end users may not be able to reach that endpoint. This option is deprecated in favor of www_authenticate_uri and will be removed in the S release.

Warning: This option is deprecated for removal since Queens. Its value may be silently ignored in the future.

> **Reason** The auth_uri option is deprecated in favor of www_authenticate_uri and will be removed in the S release.

auth_version

Type string

Default `<None>`

API version of the Identity API endpoint.

interface

Type string

Default `internal`

Interface to use for the Identity API endpoint. Valid values are public, internal (default) or admin.

delay_auth_decision

Type boolean

Default `False`

Do not handle authorization requests within the middleware, but delegate the authorization decision to downstream WSGI components.

http_connect_timeout

Type integer

Default `<None>`

Request timeout value for communicating with Identity API server.

http_request_max_retries

Type integer

Default `3`

How many times are we trying to reconnect when communicating with Identity API Server.

cache

Type string

Default `<None>`

Request environment key where the Swift cache object is stored. When auth_token middleware is deployed with a Swift cache, use this option to have the middleware share a caching backend with swift. Otherwise, use the `memcached_servers` option instead.

certfile

Type string

Default `<None>`

Required if identity server requires client certificate

keyfile

Type string

Default `<None>`

Required if identity server requires client certificate

`cafile`

Type string

Default `<None>`

A PEM encoded Certificate Authority to use when verifying HTTPs connections. Defaults to system CAs.

`insecure`

Type boolean

Default `False`

Verify HTTPS connections.

`region_name`

Type string

Default `<None>`

The region in which the identity server can be found.

`memcached_servers`

Type list

Default `<None>`

Optionally specify a list of memcached server(s) to use for caching. If left undefined, tokens will instead be cached in-process.

Table 36: Deprecated Variations

Group	Name
keystone_authtoken	memcache_servers

`token_cache_time`

Type integer

Default `300`

In order to prevent excessive effort spent validating tokens, the middleware caches previously-seen tokens for a configurable duration (in seconds). Set to -1 to disable caching completely.

`memcache_security_strategy`

Type string

Default `None`

Valid Values None, MAC, ENCRYPT

(Optional) If defined, indicate whether token data should be authenticated or authenticated and encrypted. If MAC, token data is authenticated (with HMAC) in the cache. If ENCRYPT, token data is encrypted and authenticated in the cache. If the value is not one of these options or empty, auth_token will raise an exception on initialization.

memcache_secret_key

Type string

Default `<None>`

(Optional, mandatory if memcache_security_strategy is defined) This string is used for key derivation.

memcache_pool_dead_retry

Type integer

Default `300`

(Optional) Number of seconds memcached server is considered dead before it is tried again.

memcache_pool_maxsize

Type integer

Default `10`

(Optional) Maximum total number of open connections to every memcached server.

memcache_pool_socket_timeout

Type integer

Default `3`

(Optional) Socket timeout in seconds for communicating with a memcached server.

memcache_pool_unused_timeout

Type integer

Default `60`

(Optional) Number of seconds a connection to memcached is held unused in the pool before it is closed.

memcache_pool_conn_get_timeout

Type integer

Default `10`

(Optional) Number of seconds that an operation will wait to get a memcached client connection from the pool.

memcache_use_advanced_pool

Type boolean

Default `False`

(Optional) Use the advanced (eventlet safe) memcached client pool. The advanced pool will only work under python 2.x.

include_service_catalog

Type boolean

Default `True`

(Optional) Indicate whether to set the X-Service-Catalog header. If False, middleware will not ask for service catalog on token validation and will not set the X-Service-Catalog header.

`enforce_token_bind`

Type string

Default `permissive`

Used to control the use and type of token binding. Can be set to: disabled to not check token binding. permissive (default) to validate binding information if the bind type is of a form known to the server and ignore it if not. strict like permissive but if the bind type is unknown the token will be rejected. required any form of token binding is needed to be allowed. Finally the name of a binding method that must be present in tokens.

`service_token_roles`

Type list

Default `['service']`

A choice of roles that must be present in a service token. Service tokens are allowed to request that an expired token can be used and so this check should tightly control that only actual services should be sending this token. Roles here are applied as an ANY check so any role in this list must be present. For backwards compatibility reasons this currently only affects the allow_expired check.

`service_token_roles_required`

Type boolean

Default `False`

For backwards compatibility reasons we must let valid service tokens pass that dont pass the service_token_roles check as valid. Setting this true will become the default in a future release and should be enabled if possible.

`service_type`

Type string

Default `<None>`

The name or type of the service as it appears in the service catalog. This is used to validate tokens that have restricted access rules.

`auth_type`

Type unknown type

Default `<None>`

Authentication type to load

Table 37: Deprecated Variations

Group	Name
keystone_authtoken	auth_plugin

`auth_section`

Type unknown type

Default `<None>`

Config Section from which to load plugin specific options

mdns

`registration_attempts`

Type integer

Default `5`

Minimum Value 1

Number of attempts to register a service. Currently has to be larger than 1 because of race conditions in the zeroconf library.

`lookup_attempts`

Type integer

Default `3`

Minimum Value 1

Number of attempts to lookup a service.

`params`

Type unknown type

Default `{}`

Additional parameters to pass for the registered service.

`interfaces`

Type list

Default `<None>`

List of IP addresses of interfaces to use for mDNS. Defaults to all interfaces on the system.

metrics

`backend`

Type string

Default `noop`

Valid Values noop, statsd

Backend to use for the metrics system.

`prepend_host`

Type boolean

Default `False`

Prepend the hostname to all metric names. The format of metric names is [global_prefix.][host_name.]prefix.metric_name.

prepend_host_reverse

Type boolean

Default `True`

Split the prepended host value by . and reverse it (to better match the reverse hierarchical form of domain names).

global_prefix

Type string

Default `<None>`

Prefix all metric names with this value. By default, there is no global prefix. The format of metric names is [global_prefix.][host_name.]prefix.metric_name.

agent_backend

Type string

Default `noop`

Backend for the agent ramdisk to use for metrics. Default possible backends are noop and statsd.

agent_prepend_host

Type boolean

Default `False`

Prepend the hostname to all metric names sent by the agent ramdisk. The format of metric names is [global_prefix.][uuid.][host_name.]prefix.metric_name.

agent_prepend_uuid

Type boolean

Default `False`

Prepend the nodes Ironic uuid to all metric names sent by the agent ramdisk. The format of metric names is [global_prefix.][uuid.][host_name.]prefix.metric_name.

agent_prepend_host_reverse

Type boolean

Default `True`

Split the prepended host value by . and reverse it for metrics sent by the agent ramdisk (to better match the reverse hierarchical form of domain names).

agent_global_prefix

Type string

Default `<None>`

Prefix all metric names sent by the agent ramdisk with this value. The format of metric names is [global_prefix.][uuid.][host_name.]prefix.metric_name.

metrics_statsd

`statsd_host`

Type string

Default `localhost`

Host for use with the statsd backend.

`statsd_port`

Type port number

Default `8125`

Minimum Value 0

Maximum Value 65535

Port to use with the statsd backend.

`agent_statsd_host`

Type string

Default `localhost`

Host for the agent ramdisk to use with the statsd backend. This must be accessible from networks the agent is booted on.

`agent_statsd_port`

Type port number

Default `8125`

Minimum Value 0

Maximum Value 65535

Port for the agent ramdisk to use with the statsd backend.

molds

`storage`

Type string

Default `swift`

Configuration mold storage location. Supports swift and http. By default swift.

`user`

Type string

Default `<None>`

User for http Basic auth. By default set empty.

`password`

Type string

Default `<None>`

Password for http Basic auth. By default set empty.

retry_attempts

Type integer

Default `3`

Retry attempts for saving or getting configuration molds.

retry_interval

Type integer

Default `3`

Retry interval for saving or getting configuration molds.

neutron

add_all_ports

Type boolean

Default `False`

Mutable This option can be changed without restarting.

Option to enable transmission of all ports to neutron when creating ports for provisioning, cleaning, or rescue. This is done without IP addresses assigned to the port, and may be useful in some bonded network configurations.

auth_url

Type unknown type

Default `<None>`

Authentication URL

auth_type

Type unknown type

Default `<None>`

Authentication type to load

Table 38: Deprecated Variations

Group	Name
neutron	auth_plugin

cafile

Type string

Default `<None>`

PEM encoded Certificate Authority to use when verifying HTTPs connections.

`certfile`

Type string

Default `<None>`

PEM encoded client certificate cert file

`cleaning_network`

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Neutron network UUID or name for the ramdisk to be booted into for cleaning nodes. Required for neutron network interface. It is also required if cleaning nodes when using flat network interface or neutron DHCP provider. If a name is provided, it must be unique among all networks or cleaning will fail.

Table 39: Deprecated Variations

Group	Name
neutron	cleaning_network_uuid

`cleaning_network_security_groups`

Type list

Default `[]`

Mutable This option can be changed without restarting.

List of Neutron Security Group UUIDs to be applied during cleaning of the nodes. Optional for the neutron network interface and not used for the flat or noop network interfaces. If not specified, default security group is used.

`collect_timing`

Type boolean

Default `False`

Collect per-API call timing information.

`connect_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for connection errors.

`connect_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for connection errors. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`default_domain_id`

Type unknown type

Default `<None>`

Optional domain ID to use with v3 and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`default_domain_name`

Type unknown type

Default `<None>`

Optional domain name to use with v3 API and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`dhcpv6_stateful_address_count`

Type integer

Default `4`

Mutable This option can be changed without restarting.

Number of IPv6 addresses to allocate for ports created for provisioning, cleaning, rescue or inspection on DHCPv6-stateful networks. Different stages of the chain-loading process will request addresses with different CLID/IAID. Due to non-identical identifiers multiple addresses must be reserved for the host to ensure each step of the boot process can successfully lease addresses.

`domain_id`

Type unknown type

Default `<None>`

Domain ID to scope to

`domain_name`

Type unknown type

Default `<None>`

Domain name to scope to

`endpoint_override`

Type string

Default `<None>`

Always use this endpoint URL for requests for this client. NOTE: The unversioned endpoint should be specified here; to request a particular API version, use the *version*, *min-version*, and/or *max-version* options.

`insecure`

Type boolean

Default `False`

Verify HTTPS connections.

`inspection_network`

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Neutron network UUID or name for the ramdisk to be booted into for in-band inspection of nodes. If a name is provided, it must be unique among all networks or inspection will fail.

`inspection_network_security_groups`

Type list

Default `[]`

Mutable This option can be changed without restarting.

List of Neutron Security Group UUIDs to be applied during the node inspection process. Optional for the neutron network interface and not used for the flat or noop network interfaces. If not specified, the default security group is used.

`keyfile`

Type string

Default `<None>`

PEM encoded client certificate key file

`max_version`

Type string

Default `<None>`

The maximum major version of a given API, intended to be used as the upper bound of a range with min_version. Mutually exclusive with version.

`min_version`

Type string

Default `<None>`

The minimum major version of a given API, intended to be used as the lower bound of a range with max_version. Mutually exclusive with version. If min_version is given with no max_version it is as if max version is latest.

`password`

Type unknown type

Default `<None>`

Users password

`port_setup_delay`

Type integer

Default `0`

Minimum Value 0

Mutable This option can be changed without restarting.

Delay value to wait for Neutron agents to setup sufficient DHCP configuration for port.

`project_domain_id`

Type unknown type

Default `<None>`

Domain ID containing project

`project_domain_name`

Type unknown type

Default `<None>`

Domain name containing project

`project_id`

Type unknown type

Default `<None>`

Project ID to scope to

Table 40: Deprecated Variations

Group	Name
neutron	tenant-id
neutron	tenant_id

`project_name`

Type unknown type

Default `<None>`

Project name to scope to

Table 41: Deprecated Variations

Group	Name
neutron	tenant-name
neutron	tenant_name

`provisioning_network`

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Neutron network UUID or name for the ramdisk to be booted into for provisioning nodes. Required for neutron network interface. If a name is provided, it must be unique among all networks or deploy will fail.

Table 42: Deprecated Variations

Group	Name
neutron	provisioning_network_uuid

provisioning_network_security_groups

Type list

Default `[]`

Mutable This option can be changed without restarting.

List of Neutron Security Group UUIDs to be applied during provisioning of the nodes. Optional for the neutron network interface and not used for the flat or noop network interfaces. If not specified, default security group is used.

region_name

Type string

Default `<None>`

The default region_name for endpoint URL discovery.

request_timeout

Type integer

Default `45`

Mutable This option can be changed without restarting.

Timeout for request processing when interacting with Neutron. This value should be increased if neutron port action timeouts are observed as neutron performs pre-commit validation prior returning to the API client which can take longer than normal client/server interactions.

rescuing_network

Type string

Default `<None>`

Mutable This option can be changed without restarting.

Neutron network UUID or name for booting the ramdisk for rescue mode. This is not the network that the rescue ramdisk will use post-boot the tenant network is used for that. Required for neutron network interface, if rescue mode will be used. It is not used for the flat or noop network interfaces. If a name is provided, it must be unique among all networks or rescue will fail.

rescuing_network_security_groups

Type list

Default `[]`

Mutable This option can be changed without restarting.

List of Neutron Security Group UUIDs to be applied during the node rescue process. Optional for the neutron network interface and not used for the flat or noop network interfaces. If not specified, the default security group is used.

retries

Type integer

Default `3`

Mutable This option can be changed without restarting.

DEPRECATED: Client retries in the case of a failed request.

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.
>
> **Reason** Replaced by status_code_retries and status_code_retry_delay.

`service_name`

Type string

Default `<None>`

The default service_name for endpoint URL discovery.

`service_type`

Type string

Default `network`

The default service_type for endpoint URL discovery.

`split_loggers`

Type boolean

Default `False`

Log requests to multiple loggers.

`status_code_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for retriable HTTP status codes.

`status_code_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for retriable status codes. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`system_scope`

Type unknown type

Default `<None>`

Scope for system operations

`tenant_id`

Type unknown type

Default `<None>`

Tenant ID

tenant_name

Type unknown type

Default `<None>`

Tenant Name

timeout

Type integer

Default `<None>`

Timeout value for http requests

trust_id

Type unknown type

Default `<None>`

Trust ID

user_domain_id

Type unknown type

Default `<None>`

Users domain id

user_domain_name

Type unknown type

Default `<None>`

Users domain name

user_id

Type unknown type

Default `<None>`

User id

username

Type unknown type

Default `<None>`

Username

Table 43: Deprecated Variations

Group	Name
neutron	user-name
neutron	user_name

valid_interfaces

Type list

Default `['internal', 'public']`

List of interfaces, in order of preference, for endpoint URL.

`version`

Type string

Default `<None>`

Minimum Major API version within a given Major API version for endpoint URL discovery. Mutually exclusive with min_version and max_version

nova

`auth_url`

Type unknown type

Default `<None>`

Authentication URL

`auth_type`

Type unknown type

Default `<None>`

Authentication type to load

Table 44: Deprecated Variations

Group	Name
nova	auth_plugin

`cafile`

Type string

Default `<None>`

PEM encoded Certificate Authority to use when verifying HTTPs connections.

`certfile`

Type string

Default `<None>`

PEM encoded client certificate cert file

`collect_timing`

Type boolean

Default `False`

Collect per-API call timing information.

`connect_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for connection errors.

`connect_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for connection errors. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`default_domain_id`

Type unknown type

Default `<None>`

Optional domain ID to use with v3 and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`default_domain_name`

Type unknown type

Default `<None>`

Optional domain name to use with v3 API and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`domain_id`

Type unknown type

Default `<None>`

Domain ID to scope to

`domain_name`

Type unknown type

Default `<None>`

Domain name to scope to

`endpoint_override`

Type string

Default `<None>`

Always use this endpoint URL for requests for this client. NOTE: The unversioned endpoint should be specified here; to request a particular API version, use the *version*, *min-version*, and/or *max-version* options.

`insecure`

Type boolean

Default `False`

Verify HTTPS connections.

`keyfile`

Type string

Default `<None>`

PEM encoded client certificate key file

`max_version`

Type string

Default `<None>`

The maximum major version of a given API, intended to be used as the upper bound of a range with min_version. Mutually exclusive with version.

`min_version`

Type string

Default `<None>`

The minimum major version of a given API, intended to be used as the lower bound of a range with max_version. Mutually exclusive with version. If min_version is given with no max_version it is as if max version is latest.

`password`

Type unknown type

Default `<None>`

Users password

`project_domain_id`

Type unknown type

Default `<None>`

Domain ID containing project

`project_domain_name`

Type unknown type

Default `<None>`

Domain name containing project

`project_id`

Type unknown type

Default `<None>`

Project ID to scope to

Table 45: Deprecated Variations

Group	Name
nova	tenant-id
nova	tenant_id

`project_name`

Type unknown type

Default `<None>`

Project name to scope to

Table 46: Deprecated Variations

Group	Name
nova	tenant-name
nova	tenant_name

`region_name`

Type string

Default `<None>`

The default region_name for endpoint URL discovery.

`send_power_notifications`

Type boolean

Default `True`

Mutable This option can be changed without restarting.

When set to True, it will enable the support for power state change callbacks to nova. This option should be set to False in deployments that do not have the openstack compute service.

`service_name`

Type string

Default `<None>`

The default service_name for endpoint URL discovery.

`service_type`

Type string

Default `compute`

The default service_type for endpoint URL discovery.

`split_loggers`

Type boolean

Default `False`

Log requests to multiple loggers.

`status_code_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for retriable HTTP status codes.

`status_code_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for retriable status codes. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`system_scope`

Type unknown type

Default `<None>`

Scope for system operations

`tenant_id`

Type unknown type

Default `<None>`

Tenant ID

`tenant_name`

Type unknown type

Default `<None>`

Tenant Name

`timeout`

Type integer

Default `<None>`

Timeout value for http requests

`trust_id`

Type unknown type

Default `<None>`

Trust ID

`user_domain_id`

Type unknown type

Default `<None>`

Users domain id

`user_domain_name`

Type unknown type

Default `<None>`

Users domain name

`user_id`

Type unknown type

Default `<None>`

User id

`username`

Type unknown type

Default `<None>`

Username

Table 47: Deprecated Variations

Group	Name
nova	user-name
nova	user_name

`valid_interfaces`

Type list

Default `['internal', 'public']`

List of interfaces, in order of preference, for endpoint URL.

`version`

Type string

Default `<None>`

Minimum Major API version within a given Major API version for endpoint URL discovery. Mutually exclusive with min_version and max_version

oslo_concurrency

`disable_process_locking`

Type boolean

Default `False`

Enables or disables inter-process locks.

Table 48: Deprecated Variations

Group	Name
DEFAULT	disable_process_locking

`lock_path`

Type string

Default `<None>`

Directory to use for lock files. For security, the specified directory should only be writable by the user running the processes that need locking. Defaults to environment variable OSLO_LOCK_PATH. If external locks are used, a lock path must be set.

Table 49: Deprecated Variations

Group	Name
DEFAULT	lock_path

oslo_messaging_amqp

`container_name`

Type string

Default `<None>`

Name for the AMQP container. must be globally unique. Defaults to a generated UUID

Table 50: Deprecated Variations

Group	Name
amqp1	container_name

`idle_timeout`

Type integer

Default `0`

Timeout for inactive connections (in seconds)

Table 51: Deprecated Variations

Group	Name
amqp1	idle_timeout

`trace`

Type boolean

Default `False`

Debug: dump AMQP frames to stdout

Table 52: Deprecated Variations

Group	Name
amqp1	trace

`ssl`

Type boolean

Default `False`

Attempt to connect via SSL. If no other ssl-related parameters are given, it will use the systems CA-bundle to verify the servers certificate.

`ssl_ca_file`

Type string

Default `''`

CA certificate PEM file used to verify the servers certificate

Table 53: Deprecated Variations

Group	Name
amqp1	ssl_ca_file

ssl_cert_file

Type string

Default `''`

Self-identifying certificate PEM file for client authentication

Table 54: Deprecated Variations

Group	Name
amqp1	ssl_cert_file

ssl_key_file

Type string

Default `''`

Private key PEM file used to sign ssl_cert_file certificate (optional)

Table 55: Deprecated Variations

Group	Name
amqp1	ssl_key_file

ssl_key_password

Type string

Default `<None>`

Password for decrypting ssl_key_file (if encrypted)

Table 56: Deprecated Variations

Group	Name
amqp1	ssl_key_password

ssl_verify_vhost

Type boolean

Default `False`

By default SSL checks that the name in the servers certificate matches the hostname in the transport_url. In some configurations it may be preferable to use the virtual hostname instead, for example if the server uses the Server Name Indication TLS extension (rfc6066) to provide a certificate per virtual host. Set ssl_verify_vhost to True if the servers SSL certificate uses the virtual host name instead of the DNS name.

sasl_mechanisms

Type string

Default `''`

Space separated list of acceptable SASL mechanisms

Table 57: Deprecated Variations

Group	Name
amqp1	sasl_mechanisms

`sasl_config_dir`

Type string

Default `''`

Path to directory that contains the SASL configuration

Table 58: Deprecated Variations

Group	Name
amqp1	sasl_config_dir

`sasl_config_name`

Type string

Default `''`

Name of configuration file (without .conf suffix)

Table 59: Deprecated Variations

Group	Name
amqp1	sasl_config_name

`sasl_default_realm`

Type string

Default `''`

SASL realm to use if no realm present in username

`connection_retry_interval`

Type integer

Default `1`

Minimum Value 1

Seconds to pause before attempting to re-connect.

`connection_retry_backoff`

Type integer

Default `2`

Minimum Value 0

Increase the connection_retry_interval by this many seconds after each unsuccessful failover attempt.

`connection_retry_interval_max`

Type integer

Default `30`

Minimum Value 1

Maximum limit for connection_retry_interval + connection_retry_backoff

`link_retry_delay`

Type integer

Default `10`

Minimum Value 1

Time to pause between re-connecting an AMQP 1.0 link that failed due to a recoverable error.

`default_reply_retry`

Type integer

Default `0`

Minimum Value -1

The maximum number of attempts to re-send a reply message which failed due to a recoverable error.

`default_reply_timeout`

Type integer

Default `30`

Minimum Value 5

The deadline for an rpc reply message delivery.

`default_send_timeout`

Type integer

Default `30`

Minimum Value 5

The deadline for an rpc cast or call message delivery. Only used when caller does not provide a timeout expiry.

`default_notify_timeout`

Type integer

Default `30`

Minimum Value 5

The deadline for a sent notification message delivery. Only used when caller does not provide a timeout expiry.

`default_sender_link_timeout`

Type integer

Default `600`

Minimum Value 1

The duration to schedule a purge of idle sender links. Detach link after expiry.

`addressing_mode`

Type string

Default `dynamic`

Indicates the addressing mode used by the driver. Permitted values: legacy - use legacy non-routable addressing routable - use routable addresses dynamic - use legacy addresses if the message bus does not support routing otherwise use routable addressing

`pseudo_vhost`

Type boolean

Default `True`

Enable virtual host support for those message buses that do not natively support virtual hosting (such as qpidd). When set to true the virtual host name will be added to all message bus addresses, effectively creating a private subnet per virtual host. Set to False if the message bus supports virtual hosting using the hostname field in the AMQP 1.0 Open performative as the name of the virtual host.

`server_request_prefix`

Type string

Default `exclusive`

address prefix used when sending to a specific server

Table 60: Deprecated Variations

Group	Name
amqp1	server_request_prefix

`broadcast_prefix`

Type string

Default `broadcast`

address prefix used when broadcasting to all servers

Table 61: Deprecated Variations

Group	Name
amqp1	broadcast_prefix

`group_request_prefix`

Type string

Default `unicast`

address prefix when sending to any server in group

Table 62: Deprecated Variations

Group	Name
amqp1	group_request_prefix

`rpc_address_prefix`

Type string

Default `openstack.org/om/rpc`

Address prefix for all generated RPC addresses

`notify_address_prefix`

Type string

Default `openstack.org/om/notify`

Address prefix for all generated Notification addresses

`multicast_address`

Type string

Default `multicast`

Appended to the address prefix when sending a fanout message. Used by the message bus to identify fanout messages.

`unicast_address`

Type string

Default `unicast`

Appended to the address prefix when sending to a particular RPC/Notification server. Used by the message bus to identify messages sent to a single destination.

`anycast_address`

Type string

Default `anycast`

Appended to the address prefix when sending to a group of consumers. Used by the message bus to identify messages that should be delivered in a round-robin fashion across consumers.

`default_notification_exchange`

Type string

Default `<None>`

Exchange name used in notification addresses. Exchange name resolution precedence: Target.exchange if set else default_notification_exchange if set else control_exchange if set else notify

`default_rpc_exchange`

Type string

Default `<None>`

Exchange name used in RPC addresses. Exchange name resolution precedence: Target.exchange if set else default_rpc_exchange if set else control_exchange if set else rpc

`reply_link_credit`

Type integer

Default `200`

Minimum Value 1

Window size for incoming RPC Reply messages.

`rpc_server_credit`

Type integer

Default `100`

Minimum Value 1

Window size for incoming RPC Request messages

`notify_server_credit`

Type integer

Default `100`

Minimum Value 1

Window size for incoming Notification messages

`pre_settled`

Type multi-valued

Default `rpc-cast`

Default `rpc-reply`

Send messages of this type pre-settled. Pre-settled messages will not receive acknowledgement from the peer. Note well: pre-settled messages may be silently discarded if the delivery fails. Permitted values: rpc-call - send RPC Calls pre-settled rpc-reply- send RPC Replies pre-settled rpc-cast - Send RPC Casts pre-settled notify - Send Notifications pre-settled

oslo_messaging_kafka

`kafka_max_fetch_bytes`

Type integer

Default `1048576`

Max fetch bytes of Kafka consumer

`kafka_consumer_timeout`

Type floating point

Default `1.0`

Default timeout(s) for Kafka consumers

`pool_size`

Type integer

Default `10`

Pool Size for Kafka Consumers

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.
>
> **Reason** Driver no longer uses connection pool.

conn_pool_min_size

Type integer

Default `2`

The pool size limit for connections expiration policy

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.
>
> **Reason** Driver no longer uses connection pool.

conn_pool_ttl

Type integer

Default `1200`

The time-to-live in sec of idle connections in the pool

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.
>
> **Reason** Driver no longer uses connection pool.

consumer_group

Type string

Default `oslo_messaging_consumer`

Group id for Kafka consumer. Consumers in one group will coordinate message consumption

producer_batch_timeout

Type floating point

Default `0.0`

Upper bound on the delay for KafkaProducer batching in seconds

producer_batch_size

Type integer

Default `16384`

Size of batch for the producer async send

compression_codec

Type string

Default `none`

Valid Values none, gzip, snappy, lz4, zstd

The compression codec for all data generated by the producer. If not set, compression will not be used. Note that the allowed values of this depend on the kafka version

enable_auto_commit

Type boolean

Default `False`

Enable asynchronous consumer commits

max_poll_records

Type integer

Default `500`

The maximum number of records returned in a poll call

security_protocol

Type string

Default `PLAINTEXT`

Valid Values PLAINTEXT, SASL_PLAINTEXT, SSL, SASL_SSL

Protocol used to communicate with brokers

sasl_mechanism

Type string

Default `PLAIN`

Mechanism when security protocol is SASL

ssl_cafile

Type string

Default `''`

CA certificate PEM file used to verify the server certificate

ssl_client_cert_file

Type string

Default `''`

Client certificate PEM file used for authentication.

ssl_client_key_file

Type string

Default `''`

Client key PEM file used for authentication.

ssl_client_key_password

Type string

Default `''`

Client key password file used for authentication.

oslo_messaging_notifications

driver

Type multi-valued

Default `''`

The Drivers(s) to handle sending notifications. Possible values are messaging, messagingv2, routing, log, test, noop

Table 63: Deprecated Variations

Group	Name
DEFAULT	notification_driver

transport_url

Type string

Default `<None>`

A URL representing the messaging driver to use for notifications. If not set, we fall back to the same configuration used for RPC.

Table 64: Deprecated Variations

Group	Name
DEFAULT	notification_transport_url

topics

Type list

Default `['notifications']`

AMQP topic used for OpenStack notifications.

Table 65: Deprecated Variations

Group	Name
rpc_notifier2	topics
DEFAULT	notification_topics

retry

Type integer

Default `-1`

The maximum number of attempts to re-send a notification message which failed to be delivered due to a recoverable error. 0 - No retry, -1 - indefinite

oslo_messaging_rabbit

`amqp_durable_queues`

Type boolean

Default `False`

Use durable queues in AMQP.

`amqp_auto_delete`

Type boolean

Default `False`

Auto-delete queues in AMQP.

Table 66: Deprecated Variations

Group	Name
DEFAULT	amqp_auto_delete

`ssl`

Type boolean

Default `False`

Connect over SSL.

Table 67: Deprecated Variations

Group	Name
oslo_messaging_rabbit	rabbit_use_ssl

`ssl_version`

Type string

Default `''`

SSL version to use (valid only if SSL enabled). Valid values are TLSv1 and SSLv23. SSLv2, SSLv3, TLSv1_1, and TLSv1_2 may be available on some distributions.

Table 68: Deprecated Variations

Group	Name
oslo_messaging_rabbit	kombu_ssl_version

`ssl_key_file`

Type string

Default `''`

SSL key file (valid only if SSL enabled).

Table 69: Deprecated Variations

Group	Name
oslo_messaging_rabbit	kombu_ssl_keyfile

`ssl_cert_file`

Type string

Default `''`

SSL cert file (valid only if SSL enabled).

Table 70: Deprecated Variations

Group	Name
oslo_messaging_rabbit	kombu_ssl_certfile

`ssl_ca_file`

Type string

Default `''`

SSL certification authority file (valid only if SSL enabled).

Table 71: Deprecated Variations

Group	Name
oslo_messaging_rabbit	kombu_ssl_ca_certs

`heartbeat_in_pthread`

Type boolean

Default `True`

Run the health check heartbeat thread through a native python thread by default. If this option is equal to False then the health check heartbeat will inherit the execution model from the parent process. For example if the parent process has monkey patched the stdlib by using eventlet/greenlet then the heartbeat will be run through a green thread.

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.

`kombu_reconnect_delay`

Type floating point

Default `1.0`

How long to wait before reconnecting in response to an AMQP consumer cancel notification.

Table 72: Deprecated Variations

Group	Name
DEFAULT	kombu_reconnect_delay

kombu_compression

Type string

Default `<None>`

EXPERIMENTAL: Possible values are: gzip, bz2. If not set compression will not be used. This option may not be available in future versions.

kombu_missing_consumer_retry_timeout

Type integer

Default `60`

How long to wait a missing client before abandoning to send it its replies. This value should not be longer than rpc_response_timeout.

Table 73: Deprecated Variations

Group	Name
oslo_messaging_rabbit	kombu_reconnect_timeout

kombu_failover_strategy

Type string

Default `round-robin`

Valid Values round-robin, shuffle

Determines how the next RabbitMQ node is chosen in case the one we are currently connected to becomes unavailable. Takes effect only if more than one RabbitMQ node is provided in config.

rabbit_login_method

Type string

Default `AMQPLAIN`

Valid Values PLAIN, AMQPLAIN, RABBIT-CR-DEMO

The RabbitMQ login method.

Table 74: Deprecated Variations

Group	Name
DEFAULT	rabbit_login_method

rabbit_retry_interval

Type integer

Default `1`

How frequently to retry connecting with RabbitMQ.

`rabbit_retry_backoff`

Type integer

Default `2`

How long to backoff for between retries when connecting to RabbitMQ.

Table 75: Deprecated Variations

Group	Name
DEFAULT	rabbit_retry_backoff

`rabbit_interval_max`

Type integer

Default `30`

Maximum interval of RabbitMQ connection retries. Default is 30 seconds.

`rabbit_ha_queues`

Type boolean

Default `False`

Try to use HA queues in RabbitMQ (x-ha-policy: all). If you change this option, you must wipe the RabbitMQ database. In RabbitMQ 3.0, queue mirroring is no longer controlled by the x-ha-policy argument when declaring a queue. If you just want to make sure that all queues (except those with auto-generated names) are mirrored across all nodes, run: rabbitmqctl set_policy HA ^(?!amq.).* {ha-mode: all}

Table 76: Deprecated Variations

Group	Name
DEFAULT	rabbit_ha_queues

`rabbit_transient_queues_ttl`

Type integer

Default `1800`

Minimum Value 1

Positive integer representing duration in seconds for queue TTL (x-expires). Queues which are unused for the duration of the TTL are automatically deleted. The parameter affects only reply and fanout queues.

`rabbit_qos_prefetch_count`

Type integer

Default `0`

Specifies the number of messages to prefetch. Setting to zero allows unlimited messages.

`heartbeat_timeout_threshold`

Type integer

Default `60`

Number of seconds after which the Rabbit broker is considered down if heartbeats keep-alive fails (0 disables heartbeat).

heartbeat_rate

Type integer

Default `2`

How often times during the heartbeat_timeout_threshold we check the heartbeat.

direct_mandatory_flag

Type boolean

Default `True`

(DEPRECATED) Enable/Disable the RabbitMQ mandatory flag for direct send. The direct send is used as reply, so the MessageUndeliverable exception is raised in case the client queue does not exist.MessageUndeliverable exception will be used to loop for a timeout to lets a chance to sender to recover.This flag is deprecated and it will not be possible to deactivate this functionality anymore

> **Warning:** This option is deprecated for removal. Its value may be silently ignored in the future.
>
> **Reason** Mandatory flag no longer deactivable.

enable_cancel_on_failover

Type boolean

Default `False`

Enable x-cancel-on-ha-failover flag so that rabbitmq server will cancel and notify consumerswhen queue is down

oslo_middleware

enable_proxy_headers_parsing

Type boolean

Default `False`

Whether the application is behind a proxy or not. This determines if the middleware should parse the headers or not.

oslo_policy

`enforce_scope`

Type boolean

Default `False`

This option controls whether or not to enforce scope when evaluating policies. If `True`, the scope of the token used in the request is compared to the `scope_types` of the policy being enforced. If the scopes do not match, an `InvalidScope` exception will be raised. If `False`, a message will be logged informing operators that policies are being invoked with mismatching scope.

`enforce_new_defaults`

Type boolean

Default `False`

This option controls whether or not to use old deprecated defaults when evaluating policies. If `True`, the old deprecated defaults are not going to be evaluated. This means if any existing token is allowed for old defaults but is disallowed for new defaults, it will be disallowed. It is encouraged to enable this flag along with the `enforce_scope` flag so that you can get the benefits of new defaults and `scope_type` together

`policy_file`

Type string

Default `policy.yaml`

The relative or absolute path of a file that maps roles to permissions for a given service. Relative paths must be specified in relation to the configuration file setting this option.

Table 77: Deprecated Variations

Group	Name
DEFAULT	policy_file

`policy_default_rule`

Type string

Default `default`

Default rule. Enforced when a requested rule is not found.

Table 78: Deprecated Variations

Group	Name
DEFAULT	policy_default_rule

`policy_dirs`

Type multi-valued

Default `policy.d`

Directories where policy configuration files are stored. They can be relative to any directory in the search path defined by the config_dir option, or absolute paths. The file defined by policy_file must exist for these directories to be searched. Missing or empty directories are ignored.

Table 79: Deprecated Variations

Group	Name
DEFAULT	policy_dirs

remote_content_type

Type string

Default `application/x-www-form-urlencoded`

Valid Values application/x-www-form-urlencoded, application/json

Content Type to send and receive data for REST based policy check

remote_ssl_verify_server_crt

Type boolean

Default `False`

server identity verification for REST based policy check

remote_ssl_ca_crt_file

Type string

Default `<None>`

Absolute path to ca cert file for REST based policy check

remote_ssl_client_crt_file

Type string

Default `<None>`

Absolute path to client cert for REST based policy check

remote_ssl_client_key_file

Type string

Default `<None>`

Absolute path client key file REST based policy check

profiler

enabled

Type boolean

Default `False`

Enable the profiling for all services on this node.

Default value is False (fully disable the profiling feature).

Possible values:

- True: Enables the feature
- False: Disables the feature. The profiling cannot be started via this project operations. If the profiling is triggered by another project, this project part will be empty.

Table 80: Deprecated Variations

Group	Name
profiler	profiler_enabled

`trace_sqlalchemy`

Type boolean

Default `False`

Enable SQL requests profiling in services.

Default value is False (SQL requests wont be traced).

Possible values:

- True: Enables SQL requests profiling. Each SQL query will be part of the trace and can the be analyzed by how much time was spent for that.
- False: Disables SQL requests profiling. The spent time is only shown on a higher level of operations. Single SQL queries cannot be analyzed this way.

`hmac_keys`

Type string

Default `SECRET_KEY`

Secret key(s) to use for encrypting context data for performance profiling.

This string value should have the following format: <key1>[,<key2>,<keyn>], where each key is some random string. A user who triggers the profiling via the REST API has to set one of these keys in the headers of the REST API call to include profiling results of this node for this particular project.

Both enabled flag and hmac_keys config options should be set to enable profiling. Also, to generate correct profiling information across all services at least one key needs to be consistent between OpenStack projects. This ensures it can be used from client side to generate the trace, containing information from all possible resources.

`connection_string`

Type string

Default `messaging://`

Connection string for a notifier backend.

Default value is `messaging://` which sets the notifier to oslo_messaging.

Examples of possible values:

- `messaging://` - use oslo_messaging driver for sending spans.
- `redis://127.0.0.1:6379` - use redis driver for sending spans.
- `mongodb://127.0.0.1:27017` - use mongodb driver for sending spans.

- `elasticsearch://127.0.0.1:9200` - use elasticsearch driver for sending spans.
- `jaeger://127.0.0.1:6831` - use jaeger tracing as driver for sending spans.

es_doc_type

Type string

Default `notification`

Document type for notification indexing in elasticsearch.

es_scroll_time

Type string

Default `2m`

This parameter is a time value parameter (for example: es_scroll_time=2m), indicating for how long the nodes that participate in the search will maintain relevant resources in order to continue and support it.

es_scroll_size

Type integer

Default `10000`

Elasticsearch splits large requests in batches. This parameter defines maximum size of each batch (for example: es_scroll_size=10000).

socket_timeout

Type floating point

Default `0.1`

Redissentinel provides a timeout option on the connections. This parameter defines that timeout (for example: socket_timeout=0.1).

sentinel_service_name

Type string

Default `mymaster`

Redissentinel uses a service name to identify a master redis service. This parameter defines the name (for example: `sentinal_service_name=mymaster`).

filter_error_trace

Type boolean

Default `False`

Enable filter traces that contain error/exception to a separated place.

Default value is set to False.

Possible values:

- True: Enable filter traces that contain error/exception.
- False: Disable the filter.

pxe

pxe_append_params

Type string

Default `nofb nomodeset vga=normal`

Mutable This option can be changed without restarting.

Additional append parameters for baremetal PXE boot.

default_ephemeral_format

Type string

Default `ext4`

Mutable This option can be changed without restarting.

Default file system format for ephemeral partition, if one is created.

images_path

Type string

Default `/var/lib/ironic/images/`

On the ironic-conductor node, directory where images are stored on disk.

instance_master_path

Type string

Default `/var/lib/ironic/master_images`

On the ironic-conductor node, directory where master instance images are stored on disk. Setting to the empty string disables image caching.

image_cache_size

Type integer

Default `20480`

Maximum size (in MiB) of cache for master images, including those in use.

image_cache_ttl

Type integer

Default `10080`

Maximum TTL (in minutes) for old master images in cache.

pxe_config_template

Type string

Default `$pybasedir/drivers/modules/pxe_config.template`

Mutable This option can be changed without restarting.

On ironic-conductor node, template file for PXE loader configuration.

ipxe_config_template

Type string

Default `$pybasedir/drivers/modules/ipxe_config.template`

Mutable This option can be changed without restarting.

On ironic-conductor node, template file for iPXE operations.

`uefi_pxe_config_template`

Type string

Default `$pybasedir/drivers/modules/pxe_grub_config.template`

Mutable This option can be changed without restarting.

On ironic-conductor node, template file for PXE configuration for UEFI boot loader. Generally this is used for GRUB specific templates.

`pxe_config_template_by_arch`

Type dict

Default `{}`

Mutable This option can be changed without restarting.

On ironic-conductor node, template file for PXE configuration per node architecture. For example: aarch64:/opt/share/grubaa64_pxe_config.template

`tftp_server`

Type string

Default `$my_ip`

IP address of ironic-conductor nodes TFTP server.

`tftp_root`

Type string

Default `/tftpboot`

ironic-conductor nodes TFTP root path. The ironic-conductor must have read/write access to this path.

`tftp_master_path`

Type string

Default `/tftpboot/master_images`

On ironic-conductor node, directory where master TFTP images are stored on disk. Setting to the empty string disables image caching.

`dir_permission`

Type integer

Default `<None>`

The permission that will be applied to the TFTP folders upon creation. This should be set to the permission such that the tftpserver has access to read the contents of the configured TFTP folder. This setting is only required when the operating systems umask is restrictive such that ironic-conductor is creating files that cannot be read by the TFTP server. Setting to <None> will result in the operating systems umask to be utilized for the creation of new tftp folders. It is recommended that an octal representation is specified. For example: 0o755

`pxe_bootfile_name`

Type string

Default `pxelinux.0`

Bootfile DHCP parameter.

`pxe_config_subdir`

Type string

Default `pxelinux.cfg`

Directory in which to create symbolic links which represent the MAC or IP address of the ports on a node and allow boot loaders to load the PXE file for the node. This directory name is relative to the PXE or iPXE folders.

`uefi_pxe_bootfile_name`

Type string

Default `bootx64.efi`

Bootfile DHCP parameter for UEFI boot mode.

`ipxe_bootfile_name`

Type string

Default `undionly.kpxe`

Bootfile DHCP parameter.

`uefi_ipxe_bootfile_name`

Type string

Default `ipxe.efi`

Bootfile DHCP parameter for UEFI boot mode. If you experience problems with booting using it, try snponly.efi.

`pxe_bootfile_name_by_arch`

Type dict

Default `{}`

Bootfile DHCP parameter per node architecture. For example: aarch64:grubaa64.efi

`ipxe_bootfile_name_by_arch`

Type dict

Default `{}`

Bootfile DHCP parameter per node architecture. For example: aarch64:ipxe_aa64.efi

`ipxe_boot_script`

Type string

Default `$pybasedir/drivers/modules/boot.ipxe`

On ironic-conductor node, the path to the main iPXE script file.

`ipxe_timeout`

Type integer

Default `0`

Timeout value (in seconds) for downloading an image via iPXE. Defaults to 0 (no timeout)

`boot_retry_timeout`

Type integer

Default `<None>`

Minimum Value 60

Timeout (in seconds) after which PXE boot should be retried. Must be less than [conductor]deploy_callback_timeout. Disabled by default.

`boot_retry_check_interval`

Type integer

Default `90`

Minimum Value 1

Interval (in seconds) between periodic checks on PXE boot retry. Has no effect if boot_retry_timeout is not set.

`ip_version`

Type string

Default `4`

Valid Values 4, 6

Mutable This option can be changed without restarting.

The IP version that will be used for PXE booting. Defaults to 4. EXPERIMENTAL

Possible values

4 IPv4

6 IPv6

`ipxe_use_swift`

Type boolean

Default `False`

Mutable This option can be changed without restarting.

Download deploy and rescue images directly from swift using temporary URLs. If set to false (default), images are downloaded to the ironic-conductor node and served over its local HTTP server. Applicable only when ipxe compatible boot interface is used.

`enable_netboot_fallback`

Type boolean

Default `False`

Mutable This option can be changed without restarting.

If True, generate a PXE environment even for nodes that use local boot. This is useful when the driver cannot switch nodes to local boot, e.g. with SNMP or with Redfish on machines that cannot do persistent boot. Mostly useful for standalone ironic since Neutron will prevent incorrect PXE boot.

redfish

`connection_attempts`

Type integer

Default `5`

Minimum Value 1

Maximum number of attempts to try to connect to Redfish

`connection_retry_interval`

Type integer

Default `4`

Minimum Value 1

Number of seconds to wait between attempts to connect to Redfish

`connection_cache_size`

Type integer

Default `1000`

Minimum Value 0

Maximum Redfish client connection cache size. Redfish driver would strive to reuse authenticated BMC connections (obtained through Redfish Session Service). This option caps the maximum number of connections to maintain. The value of *0* disables client connection caching completely.

`auth_type`

Type string

Default `auto`

Valid Values basic, session, auto

Redfish HTTP client authentication method.

Possible values

basic Use HTTP basic authentication

session Use HTTP session authentication

auto Try HTTP session authentication first, fall back to basic HTTP authentication

`use_swift`

Type boolean

Default `True`

Mutable This option can be changed without restarting.

Upload generated ISO images for virtual media boot to Swift, then pass temporary URL to BMC for booting the node. If set to false, images are placed on the ironic-conductor node and served over its local HTTP server.

`swift_container`

Type string

Default `ironic_redfish_container`

Mutable This option can be changed without restarting.

The Swift container to store Redfish driver data. Applies only when *use_swift* is enabled.

`swift_object_expiry_timeout`

Type integer

Default `900`

Mutable This option can be changed without restarting.

Amount of time in seconds for Swift objects to auto-expire. Applies only when *use_swift* is enabled.

`kernel_append_params`

Type string

Default `nofb nomodeset vga=normal`

Mutable This option can be changed without restarting.

Additional kernel parameters to pass down to the instance kernel. These parameters can be consumed by the kernel or by the applications by reading /proc/cmdline. Mind severe cmdline size limit! Can be overridden by *instance_info/kernel_append_params* property.

`file_permission`

Type integer

Default `420`

File permission for swift-less image hosting with the octal permission representation of file access permissions. This setting defaults to `644`, or as the octal number `0o644` in Python. This setting must be set to the octal number representation, meaning starting with `0o`.

`firmware_update_status_interval`

Type integer

Default `60`

Minimum Value 0

Number of seconds to wait between checking for completed firmware update tasks

`firmware_update_fail_interval`

Type integer

Default `60`

Minimum Value 0

Number of seconds to wait between checking for failed firmware update tasks

`raid_config_status_interval`

Type integer

Default `60`

Minimum Value 0

Number of seconds to wait between checking for completed raid config tasks

`raid_config_fail_interval`

Type integer

Default `60`

Minimum Value 0

Number of seconds to wait between checking for failed raid config tasks

service_catalog

`auth_url`

Type unknown type

Default `<None>`

Authentication URL

`auth_type`

Type unknown type

Default `<None>`

Authentication type to load

Table 81: Deprecated Variations

Group	Name
service_catalog	auth_plugin

`cafile`

Type string

Default `<None>`

PEM encoded Certificate Authority to use when verifying HTTPs connections.

`certfile`

Type string

Default `<None>`

PEM encoded client certificate cert file

`collect_timing`

Type boolean

Default `False`

Collect per-API call timing information.

`connect_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for connection errors.

`connect_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for connection errors. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`default_domain_id`

Type unknown type

Default `<None>`

Optional domain ID to use with v3 and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`default_domain_name`

Type unknown type

Default `<None>`

Optional domain name to use with v3 API and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`domain_id`

Type unknown type

Default `<None>`

Domain ID to scope to

`domain_name`

Type unknown type

Default `<None>`

Domain name to scope to

`endpoint_override`

Type string

Default `<None>`

Always use this endpoint URL for requests for this client. NOTE: The unversioned endpoint should be specified here; to request a particular API version, use the *version*, *min-version*, and/or *max-version* options.

`insecure`

Type boolean

Default `False`

Verify HTTPS connections.

`keyfile`

Type string

Default `<None>`

PEM encoded client certificate key file

`max_version`

Type string

Default `<None>`

The maximum major version of a given API, intended to be used as the upper bound of a range with min_version. Mutually exclusive with version.

`min_version`

Type string

Default `<None>`

The minimum major version of a given API, intended to be used as the lower bound of a range with max_version. Mutually exclusive with version. If min_version is given with no max_version it is as if max version is latest.

`password`

Type unknown type

Default `<None>`

Users password

`project_domain_id`

Type unknown type

Default `<None>`

Domain ID containing project

`project_domain_name`

Type unknown type

Default `<None>`

Domain name containing project

`project_id`

Type unknown type

Default `<None>`

Project ID to scope to

Table 82: Deprecated Variations

Group	Name
service_catalog	tenant-id
service_catalog	tenant_id

`project_name`

Type unknown type

Default `<None>`

Project name to scope to

Table 83: Deprecated Variations

Group	Name
service_catalog	tenant-name
service_catalog	tenant_name

`region_name`

Type string

Default `<None>`

The default region_name for endpoint URL discovery.

`service_name`

Type string

Default `<None>`

The default service_name for endpoint URL discovery.

`service_type`

Type string

Default `baremetal`

The default service_type for endpoint URL discovery.

`split_loggers`

Type boolean

Default `False`

Log requests to multiple loggers.

status_code_retries

Type integer

Default `<None>`

The maximum number of retries that should be attempted for retriable HTTP status codes.

status_code_retry_delay

Type floating point

Default `<None>`

Delay (in seconds) between two retries for retriable status codes. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

system_scope

Type unknown type

Default `<None>`

Scope for system operations

tenant_id

Type unknown type

Default `<None>`

Tenant ID

tenant_name

Type unknown type

Default `<None>`

Tenant Name

timeout

Type integer

Default `<None>`

Timeout value for http requests

trust_id

Type unknown type

Default `<None>`

Trust ID

user_domain_id

Type unknown type

Default `<None>`

Users domain id

user_domain_name

Type unknown type

Default `<None>`

Users domain name

`user_id`

Type unknown type

Default `<None>`

User id

`username`

Type unknown type

Default `<None>`

Username

Table 84: Deprecated Variations

Group	Name
service_catalog	user-name
service_catalog	user_name

`valid_interfaces`

Type list

Default `['internal', 'public']`

List of interfaces, in order of preference, for endpoint URL.

`version`

Type string

Default `<None>`

Minimum Major API version within a given Major API version for endpoint URL discovery. Mutually exclusive with min_version and max_version

snmp

`power_timeout`

Type integer

Default `10`

Seconds to wait for power action to be completed

`reboot_delay`

Type integer

Default `0`

Minimum Value 0

Time (in seconds) to sleep between when rebooting (powering off and on again)

`udp_transport_timeout`

Type floating point

Default `1.0`

Minimum Value 0.0

Response timeout in seconds used for UDP transport. Timeout should be a multiple of 0.5 seconds and is applicable to each retry.

`udp_transport_retries`

Type integer

Default `5`

Minimum Value 0

Maximum number of UDP request retries, 0 means no retries.

ssl

`ca_file`

Type string

Default `<None>`

CA certificate file to use to verify connecting clients.

Table 85: Deprecated Variations

Group	Name
DEFAULT	ssl_ca_file

`cert_file`

Type string

Default `<None>`

Certificate file to use when starting the server securely.

Table 86: Deprecated Variations

Group	Name
DEFAULT	ssl_cert_file

`key_file`

Type string

Default `<None>`

Private key file to use when starting the server securely.

Table 87: Deprecated Variations

Group	Name
DEFAULT	ssl_key_file

version

Type string

Default `<None>`

SSL version to use (valid only if SSL enabled). Valid values are TLSv1 and SSLv23. SSLv2, SSLv3, TLSv1_1, and TLSv1_2 may be available on some distributions.

ciphers

Type string

Default `<None>`

Sets the list of available ciphers. value should be a string in the OpenSSL cipher list format.

swift

auth_url

Type unknown type

Default `<None>`

Authentication URL

auth_type

Type unknown type

Default `<None>`

Authentication type to load

Table 88: Deprecated Variations

Group	Name
swift	auth_plugin

cafile

Type string

Default `<None>`

PEM encoded Certificate Authority to use when verifying HTTPs connections.

certfile

Type string

Default `<None>`

PEM encoded client certificate cert file

collect_timing

Type boolean

Default `False`

Collect per-API call timing information.

`connect_retries`

Type integer

Default `<None>`

The maximum number of retries that should be attempted for connection errors.

`connect_retry_delay`

Type floating point

Default `<None>`

Delay (in seconds) between two retries for connection errors. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`default_domain_id`

Type unknown type

Default `<None>`

Optional domain ID to use with v3 and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`default_domain_name`

Type unknown type

Default `<None>`

Optional domain name to use with v3 API and v2 parameters. It will be used for both the user and project domain in v3 and ignored in v2 authentication.

`domain_id`

Type unknown type

Default `<None>`

Domain ID to scope to

`domain_name`

Type unknown type

Default `<None>`

Domain name to scope to

`endpoint_override`

Type string

Default `<None>`

Always use this endpoint URL for requests for this client. NOTE: The unversioned endpoint should be specified here; to request a particular API version, use the *version*, *min-version*, and/or *max-version* options.

insecure

Type boolean

Default `False`

Verify HTTPS connections.

keyfile

Type string

Default `<None>`

PEM encoded client certificate key file

max_version

Type string

Default `<None>`

The maximum major version of a given API, intended to be used as the upper bound of a range with min_version. Mutually exclusive with version.

min_version

Type string

Default `<None>`

The minimum major version of a given API, intended to be used as the lower bound of a range with max_version. Mutually exclusive with version. If min_version is given with no max_version it is as if max version is latest.

password

Type unknown type

Default `<None>`

Users password

project_domain_id

Type unknown type

Default `<None>`

Domain ID containing project

project_domain_name

Type unknown type

Default `<None>`

Domain name containing project

project_id

Type unknown type

Default `<None>`

Project ID to scope to

Table 89: Deprecated Variations

Group	Name
swift	tenant-id
swift	tenant_id

project_name

Type unknown type

Default `<None>`

Project name to scope to

Table 90: Deprecated Variations

Group	Name
swift	tenant-name
swift	tenant_name

region_name

Type string

Default `<None>`

The default region_name for endpoint URL discovery.

service_name

Type string

Default `<None>`

The default service_name for endpoint URL discovery.

service_type

Type string

Default `object-store`

The default service_type for endpoint URL discovery.

split_loggers

Type boolean

Default `False`

Log requests to multiple loggers.

status_code_retries

Type integer

Default `<None>`

The maximum number of retries that should be attempted for retriable HTTP status codes.

status_code_retry_delay

Type floating point

Default `<None>`

Delay (in seconds) between two retries for retriable status codes. If not set, exponential retry starting with 0.5 seconds up to a maximum of 60 seconds is used.

`swift_max_retries`

Type integer

Default `2`

Maximum number of times to retry a Swift request, before failing.

`system_scope`

Type unknown type

Default `<None>`

Scope for system operations

`tenant_id`

Type unknown type

Default `<None>`

Tenant ID

`tenant_name`

Type unknown type

Default `<None>`

Tenant Name

`timeout`

Type integer

Default `<None>`

Timeout value for http requests

`trust_id`

Type unknown type

Default `<None>`

Trust ID

`user_domain_id`

Type unknown type

Default `<None>`

Users domain id

`user_domain_name`

Type unknown type

Default `<None>`

Users domain name

`user_id`

Type unknown type

Default `<None>`

User id

`username`

Type unknown type

Default `<None>`

Username

Table 91: Deprecated Variations

Group	Name
swift	user-name
swift	user_name

`valid_interfaces`

Type list

Default `['internal', 'public']`

List of interfaces, in order of preference, for endpoint URL.

`version`

Type string

Default `<None>`

Minimum Major API version within a given Major API version for endpoint URL discovery. Mutually exclusive with min_version and max_version

xclarity

`manager_ip`

Type string

Default `<None>`

IP address of the XClarity Controller. Configuration here is deprecated and will be removed in the Stein release. Please update the driver_info field to use xclarity_manager_ip instead

`username`

Type string

Default `<None>`

Username for the XClarity Controller. Configuration here is deprecated and will be removed in the Stein release. Please update the driver_info field to use xclarity_username instead

`password`

Type string

Default `<None>`

Password for XClarity Controller username. Configuration here is deprecated and will be removed in the Stein release. Please update the driver_info field to use xclarity_password instead

`port`

Type port number

Default `443`

Minimum Value 0

Maximum Value 65535

Port to be used for XClarity Controller connection.

6.1.2 Policies

Warning: JSON formatted policy files were deprecated in the Wallaby development cycle due to the Victoria deprecation by the `olso.policy` library. Use the oslopolicy-convert-json-to-yaml tool to convert the existing JSON to YAML formatted policy file in backward compatible way.

The following is an overview of all available policies in Ironic. For a sample configuration file, refer to *Ironic Policy*.

ironic.api

`admin_api`

Default `role:admin or role:administrator`

Legacy rule for cloud admin access

`public_api`

Default `is_public_api:True`

Internal flag for public API routes

`show_password`

Default `!`

Show or mask secrets within node driver information in API responses

`show_instance_secrets`

Default `!`

Show or mask secrets within instance information in API responses

`is_member`

Default `(project_domain_id:default or project_domain_id:None) and (project_name:demo or project_name:baremetal)`

May be used to restrict access to specific projects

is_observer

Default `rule:is_member and (role:observer or role:baremetal_observer)`

Read-only API access

is_admin

Default `rule:admin_api or (rule:is_member and role:baremetal_admin)`

Full read/write API access

is_node_owner

Default `project_id:%(node.owner)s`

Owner of node

is_node_lessee

Default `project_id:%(node.lessee)s`

Lessee of node

is_allocation_owner

Default `project_id:%(allocation.owner)s`

Owner of allocation

baremetal:node:create

Default `role:admin and system_scope:all`

Operations

- **POST** `/nodes`

Scope Types

- **system**

Create Node records

baremetal:node:list

Default `role:reader`

Operations

- **GET** `/nodes`
- **GET** `/nodes/detail`

Scope Types

- **system**
- **project**

Retrieve multiple Node records, filtered by an explicit owner or the client project_id

baremetal:node:list_all

Default `role:reader and system_scope:all`

Operations

- **GET** `/nodes`
- **GET** `/nodes/detail`

Scope Types

- **system**

Retrieve multiple Node records

`baremetal:node:get`

Default `(role:reader and system_scope:all) or (role:reader and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **GET** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Retrieve a single Node record

`baremetal:node:get:filter_threshold`

Default `role:reader and system_scope:all`

Operations

- **GET** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Filter to allow operators to govern the threshold where information should be filtered. Non-authorized users will be subjected to additional API policy checks for API content response bodies.

`baremetal:node:get:last_error`

Default `(role:reader and system_scope:all) or (role:reader and project_id:%(node.owner)s)`

Operations

- **GET** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if the node last_error field is masked from APIclients with insufficent privileges.

`baremetal:node:get:reservation`

Default `(role:reader and system_scope:all) or (role:reader and project_id:%(node.owner)s)`

Operations

- **GET** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if the node reservation field is masked from APIclients with insufficent privileges.

`baremetal:node:get:driver_internal_info`

Default `(role:reader and system_scope:all) or (role:reader and project_id:%(node.owner)s)`

Operations

- **GET** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if the node driver_internal_info field is masked from API clients with insufficent privileges.

`baremetal:node:get:driver_info`

Default `(role:reader and system_scope:all) or (role:reader and project_id:%(node.owner)s)`

Operations

- **GET** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if the driver_info field is masked from APIclients with insufficent privileges.

`baremetal:node:update:driver_info`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node driver_info field can be updated via the API clients.

`baremetal:node:update:properties`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node properties field can be updated via the API clients.

`baremetal:node:update:chassis_uuid`

Default `role:admin and system_scope:all`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node chassis_uuid field can be updated via the API clients.

`baremetal:node:update:instance_uuid`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node instance_uuid field can be updated via the API clients.

`baremetal:node:update:lessee`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node lessee field can be updated via the API clients.

baremetal:node:update:owner

Default `role:member and system_scope:all`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node owner field can be updated via the API clients.

baremetal:node:update:driver_interfaces

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node driver and driver interfaces field can be updated via the API clients.

baremetal:node:update:network_data

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node driver_info field can be updated via the API clients.

baremetal:node:update:conductor_group

Default `role:member and system_scope:all`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node conductor_group field can be updated via the API clients.

`baremetal:node:update:name`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node name field can be updated via the API clients.

`baremetal:node:update:retired`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Governs if node retired and retired reason can be updated by API clients.

`baremetal:node:update`

Default `(role:member and system_scope:all) or (role:member and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Generalized update of node records

`baremetal:node:update_extra`

Default `(role:member and system_scope:all) or (role:member and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**

- **project**

Update Node extra field

`baremetal:node:update_instance_info`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Update Node instance_info field

`baremetal:node:update_owner_provisioned`

Default `role:admin and system_scope:all`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**

Update Node owner even when Node is provisioned

`baremetal:node:delete`

Default `role:admin and system_scope:all`

Operations

- **DELETE** `/nodes/{node_ident}`

Scope Types

- **system**
- **project**

Delete Node records

`baremetal:node:validate`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **GET** `/nodes/{node_ident}/validate`

Scope Types

- **system**
- **project**

Request active validation of Nodes

`baremetal:node:set_maintenance`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **PUT** `/nodes/{node_ident}/maintenance`

Scope Types

- **system**
- **project**

Set maintenance flag, taking a Node out of service

`baremetal:node:clear_maintenance`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **DELETE** `/nodes/{node_ident}/maintenance`

Scope Types

- **system**
- **project**

Clear maintenance flag, placing the Node into service again

`baremetal:node:get_boot_device`

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **GET** `/nodes/{node_ident}/management/boot_device`
- **GET** `/nodes/{node_ident}/management/boot_device/supported`

Scope Types

- **system**
- **project**

Retrieve Node boot device metadata

`baremetal:node:set_boot_device`

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **PUT** `/nodes/{node_ident}/management/boot_device`

Scope Types

- **system**
- **project**

Change Node boot device

`baremetal:node:get_indicator_state`

Default `(role:reader and system_scope:all) or (role:reader and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **GET** `/nodes/{node_ident}/management/indicators/{component}/{indicator}`
- **GET** `/nodes/{node_ident}/management/indicators`

Scope Types

- **system**
- **project**

Retrieve Node indicators and their states

`baremetal:node:set_indicator_state`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PUT** `/nodes/{node_ident}/management/indicators/{component}/{indicator}`

Scope Types

- **system**
- **project**

Change Node indicator state

`baremetal:node:inject_nmi`

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **PUT** `/nodes/{node_ident}/management/inject_nmi`

Scope Types

- **system**
- **project**

Inject NMI for a node

`baremetal:node:get_states`

Default `(role:reader and system_scope:all) or (role:reader and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **GET** `/nodes/{node_ident}/states`

Scope Types

- **system**
- **project**

View Node power and provision state

baremetal:node:set_power_state

Default `(role:member and system_scope:all) or (role:member and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **PUT** `/nodes/{node_ident}/states/power`

Scope Types

- **system**
- **project**

Change Node power status

baremetal:node:set_provision_state

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **PUT** `/nodes/{node_ident}/states/provision`

Scope Types

- **system**
- **project**

Change Node provision status

baremetal:node:set_raid_state

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PUT** `/nodes/{node_ident}/states/raid`

Scope Types

- **system**
- **project**

Change Node RAID status

`baremetal:node:get_console`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **GET** `/nodes/{node_ident}/states/console`

Scope Types

- **system**
- **project**

Get Node console connection information

`baremetal:node:set_console_state`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s)`

Operations

- **PUT** `/nodes/{node_ident}/states/console`

Scope Types

- **system**
- **project**

Change Node console status

`baremetal:node:vif:list`

Default `(role:reader and system_scope:all) or (role:reader and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **GET** `/nodes/{node_ident}/vifs`

Scope Types

- **system**
- **project**

List VIFs attached to node

`baremetal:node:vif:attach`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **POST** `/nodes/{node_ident}/vifs`

Scope Types

- **system**
- **project**

Attach a VIF to a node

`baremetal:node:vif:detach`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **DELETE** `/nodes/{node_ident}/vifs/{node_vif_ident}`

Scope Types

- **system**
- **project**

Detach a VIF from a node

`baremetal:node:traits:list`

Default `(role:reader and system_scope:all) or (role:reader and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **GET** `/nodes/{node_ident}/traits`

Scope Types

- **system**
- **project**

List node traits

`baremetal:node:traits:set`

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **PUT** `/nodes/{node_ident}/traits`
- **PUT** `/nodes/{node_ident}/traits/{trait}`

Scope Types

- **system**
- **project**

Add a trait to, or replace all traits of, a node

`baremetal:node:traits:delete`

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **DELETE** `/nodes/{node_ident}/traits`
- **DELETE** `/nodes/{node_ident}/traits/{trait}`

Scope Types

- **system**
- **project**

Remove one or all traits from a node

`baremetal:node:bios:get`

Default `(role:reader and system_scope:all) or (role:reader and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **GET** `/nodes/{node_ident}/bios`
- **GET** `/nodes/{node_ident}/bios/{setting}`

Scope Types

- **system**
- **project**

Retrieve Node BIOS information

`baremetal:node:disable_cleaning`

Default `role:admin and system_scope:all`

Operations

- **PATCH** `/nodes/{node_ident}`

Scope Types

- **system**

Disable Node disk cleaning

`baremetal:port:get`

Default `(role:reader and system_scope:all) or (role:reader and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **GET** `/ports/{port_id}`
- **GET** `/nodes/{node_ident}/ports`
- **GET** `/nodes/{node_ident}/ports/detail`
- **GET** `/portgroups/{portgroup_ident}/ports`
- **GET** `/portgroups/{portgroup_ident}/ports/detail`

Scope Types

- **system**
- **project**

Retrieve Port records

`baremetal:port:list`

Default `role:reader`

Operations

- **GET** `/ports`
- **GET** `/ports/detail`

Scope Types

- **system**
- **project**

Retrieve multiple Port records, filtered by owner

`baremetal:port:list_all`

Default `role:reader and system_scope:all`

Operations

- **GET** `/ports`
- **GET** `/ports/detail`

Scope Types

- **system**
- **project**

Retrieve multiple Port records

`baremetal:port:create`

Default `(role:admin and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **POST** `/ports`

Scope Types

- **system**
- **project**

Create Port records

`baremetal:port:delete`

Default `(role:admin and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **DELETE** `/ports/{port_id}`

Scope Types

- **system**
- **project**

Delete Port records

`baremetal:port:update`

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **PATCH** `/ports/{port_id}`

Scope Types

- **system**
- **project**

Update Port records

`baremetal:portgroup:get`

Default `(role:reader and system_scope:all) or (role:reader and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **GET** `/portgroups`
- **GET** `/portgroups/detail`
- **GET** `/portgroups/{portgroup_ident}`
- **GET** `/nodes/{node_ident}/portgroups`
- **GET** `/nodes/{node_ident}/portgroups/detail`

Scope Types

- **system**
- **project**

Retrieve Portgroup records

`baremetal:portgroup:create`

Default `(role:admin and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **POST** `/portgroups`

Scope Types

- **system**
- **project**

Create Portgroup records

`baremetal:portgroup:delete`

Default `(role:admin and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **DELETE** `/portgroups/{portgroup_ident}`

Scope Types

- **system**
- **project**

Delete Portgroup records

`baremetal:portgroup:update`

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s)`

Operations

- **PATCH** `/portgroups/{portgroup_ident}`

Scope Types

- **system**
- **project**

Update Portgroup records

`baremetal:portgroup:list`

Default `role:reader`

Operations

- **GET** `/portgroups`
- **GET** `/portgroups/detail`

Scope Types

- **system**
- **project**

Retrieve multiple Port records, filtered by owner

`baremetal:portgroup:list_all`

Default `role:reader and system_scope:all`

Operations

- **GET** `/portgroups`
- **GET** `/portgroups/detail`

Scope Types

- **system**
- **project**

Retrieve multiple Port records

`baremetal:chassis:get`

Default `role:reader and system_scope:all`

Operations

- **GET** `/chassis`
- **GET** `/chassis/detail`
- **GET** `/chassis/{chassis_id}`

Scope Types

- **system**

Retrieve Chassis records

`baremetal:chassis:create`

Default `role:admin and system_scope:all`

Operations

- **POST** `/chassis`

Scope Types

- **system**

Create Chassis records

`baremetal:chassis:delete`

Default `role:admin and system_scope:all`

Operations

- **DELETE** `/chassis/{chassis_id}`

Scope Types

- **system**

Delete Chassis records

`baremetal:chassis:update`

Default `role:member and system_scope:all`

Operations

- **PATCH** `/chassis/{chassis_id}`

Scope Types

- **system**

Update Chassis records

`baremetal:driver:get`

Default `role:reader and system_scope:all`

Operations

- **GET** `/drivers`
- **GET** `/drivers/{driver_name}`

Scope Types

- **system**

View list of available drivers

`baremetal:driver:get_properties`

Default `role:reader and system_scope:all`

Operations

- **GET** `/drivers/{driver_name}/properties`

Scope Types

- **system**

View driver-specific properties

`baremetal:driver:get_raid_logical_disk_properties`

Default `role:reader and system_scope:all`

Operations

- **GET** `/drivers/{driver_name}/raid/logical_disk_properties`

Scope Types

- **system**

View driver-specific RAID metadata

`baremetal:node:vendor_passthru`

Default `role:admin and system_scope:all`

Operations

- **GET** `nodes/{node_ident}/vendor_passthru/methods`
- **GET** `nodes/{node_ident}/vendor_passthru?method={method_name}`
- **PUT** `nodes/{node_ident}/vendor_passthru?method={method_name}`
- **POST** `nodes/{node_ident}/vendor_passthru?method={method_name}`
- **PATCH** `nodes/{node_ident}/vendor_passthru?method={method_name}`
- **DELETE** `nodes/{node_ident}/vendor_passthru?method={method_name}`

Scope Types

- **system**

- **project**

Access vendor-specific Node functions

`baremetal:driver:vendor_passthru`

Default `role:admin and system_scope:all`

Operations

- **GET** `drivers/{driver_name}/vendor_passthru/methods`
- **GET** `drivers/{driver_name}/vendor_passthru?method={method_name}`
- **PUT** `drivers/{driver_name}/vendor_passthru?method={method_name}`
- **POST** `drivers/{driver_name}/vendor_passthru?method={method_name}`
- **PATCH** `drivers/{driver_name}/vendor_passthru?method={method_name}`
- **DELETE** `drivers/{driver_name}/vendor_passthru?method={method_name}`

Scope Types

- **system**

Access vendor-specific Driver functions

`baremetal:node:ipa_heartbeat`

Default <empty string>

Operations

- **POST** `/heartbeat/{node_ident}`

Receive heartbeats from IPA ramdisk

`baremetal:driver:ipa_lookup`

Default <empty string>

Operations

- **GET** `/lookup`

Access IPA ramdisk functions

`baremetal:volume:list_all`

Default `role:reader and system_scope:all`

Operations

- **GET** `/volume/connectors`
- **GET** `/volume/targets`
- **GET** `/nodes/{node_ident}/volume/connectors`
- **GET** `/nodes/{node_ident}/volume/targets`

Scope Types

- **system**
- **project**

Retrieve a list of all Volume connector and target records

`baremetal:volume:list`

Default `role:reader`

Operations

- **GET** `/volume/connectors`
- **GET** `/volume/targets`
- **GET** `/nodes/{node_ident}/volume/connectors`
- **GET** `/nodes/{node_ident}/volume/targets`

Scope Types

- **system**
- **project**

Retrieve a list of Volume connector and target records

`baremetal:volume:get`

Default `(role:reader and system_scope:all) or (role:reader and (project_id:%(node.owner)s or project_id:%(node.lessee)s))`

Operations

- **GET** `/volume`
- **GET** `/volume/connectors`
- **GET** `/volume/connectors/{volume_connector_id}`
- **GET** `/volume/targets`
- **GET** `/volume/targets/{volume_target_id}`
- **GET** `/nodes/{node_ident}/volume`
- **GET** `/nodes/{node_ident}/volume/connectors`
- **GET** `/nodes/{node_ident}/volume/targets`

Scope Types

- **system**
- **project**

Retrieve Volume connector and target records

`baremetal:volume:create`

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **POST** `/volume/connectors`
- **POST** `/volume/targets`

Scope Types

- **system**
- **project**

Create Volume connector and target records

`baremetal:volume:delete`

Default `(role:member and system_scope:all) or (role:admin and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **DELETE** `/volume/connectors/{volume_connector_id}`
- **DELETE** `/volume/targets/{volume_target_id}`

Scope Types

- **system**
- **project**

Delete Volume connector and target records

`baremetal:volume:update`

Default `(role:member and system_scope:all) or (role:member and project_id:%(node.owner)s) or (role:admin and project_id:%(node.lessee)s)`

Operations

- **PATCH** `/volume/connectors/{volume_connector_id}`
- **PATCH** `/volume/targets/{volume_target_id}`

Scope Types

- **system**
- **project**

Update Volume connector and target records

`baremetal:volume:view_target_properties`

Default `(role:reader and system_scope:all) or (role:admin)`

Operations

- **GET** `/volume/connectors/{volume_connector_id}`
- **GET** `/volume/targets/{volume_target_id}`

Scope Types

- **system**

- **project**

Ability to view volume target properties

`baremetal:conductor:get`

Default `role:reader and system_scope:all`

Operations

- **GET** `/conductors`
- **GET** `/conductors/{hostname}`

Scope Types

- **system**

Retrieve Conductor records

`baremetal:allocation:get`

Default `(role:reader and system_scope:all) or (role:reader and project_id:%(allocation.owner)s)`

Operations

- **GET** `/allocations/{allocation_id}`
- **GET** `/nodes/{node_ident}/allocation`

Scope Types

- **system**
- **project**

Retrieve Allocation records

`baremetal:allocation:list`

Default `role:reader`

Operations

- **GET** `/allocations`

Scope Types

- **system**
- **project**

Retrieve multiple Allocation records, filtered by owner

`baremetal:allocation:list_all`

Default `role:reader and system_scope:all`

Operations

- **GET** `/allocations`

Scope Types

- **system**
- **project**

Retrieve multiple Allocation records

`baremetal:allocation:create`

Default `(role:member and system_scope:all) or (role:member)`

Operations

- **POST** `/allocations`

Scope Types

- **system**
- **project**

Create Allocation records

`baremetal:allocation:create_restricted`

Default `role:member and system_scope:all`

Operations

- **POST** `/allocations`

Scope Types

- **system**
- **project**

Create Allocation records with a specific owner.

`baremetal:allocation:delete`

Default `(role:member and system_scope:all) or (role:member and project_id:%(allocation.owner)s)`

Operations

- **DELETE** `/allocations/{allocation_id}`
- **DELETE** `/nodes/{node_ident}/allocation`

Scope Types

- **system**
- **project**

Delete Allocation records

`baremetal:allocation:update`

Default `(role:member and system_scope:all) or (role:member and project_id:%(allocation.owner)s)`

Operations

- **PATCH** `/allocations/{allocation_id}`

Scope Types

- **system**
- **project**

Change name and extra fields of an allocation

`baremetal:allocation:create_pre_rbac`

Default `(rule:is_member and role:baremetal_admin) or (is_admin_project:True and role:admin)`

Operations

- **PATCH** `/allocations/{allocation_id}`

Scope Types

- **project**

Logical restrictor to prevent legacy allocation rule missuse - Requires blank allocations to originate from the legacy baremetal_admin.

`baremetal:events:post`

Default `role:admin and system_scope:all`

Operations

- **POST** `/events`

Scope Types

- **system**

Post events

`baremetal:deploy_template:get`

Default `role:reader and system_scope:all`

Operations

- **GET** `/deploy_templates`
- **GET** `/deploy_templates/{deploy_template_ident}`

Scope Types

- **system**

Retrieve Deploy Template records

`baremetal:deploy_template:create`

Default `role:admin and system_scope:all`

Operations

- **POST** `/deploy_templates`

Scope Types

- **system**

Create Deploy Template records

`baremetal:deploy_template:delete`

Default `role:admin and system_scope:all`

Operations

- **DELETE** `/deploy_templates/{deploy_template_ident}`

Scope Types

- **system**

Delete Deploy Template records

`baremetal:deploy_template:update`

Default `role:admin and system_scope:all`

Operations

- **PATCH** `/deploy_templates/{deploy_template_ident}`

Scope Types

- **system**

Update Deploy Template records

CHAPTER

SEVEN

BARE METAL API REFERENCES

Ironics REST API has changed since its first release, and continues to evolve to meet the changing needs of the community. Here we provide a conceptual guide as well as more detailed reference documentation.

7.1 REST API Conceptual Guide

7.1.1 Versioning

The ironic REST API supports two types of versioning:

- major versions, which have dedicated urls.
- microversions, which can be requested through the use of the `X-OpenStack-Ironic-API-Version` header.

There is only one major version supported currently, v1. As such, most URLs in this documentation are written with the /v1/ prefix.

Starting with the Kilo release, ironic supports microversions. In this context, a version is defined as a string of 2 integers separated by a dot: **X.Y**. Here `X` is a major version, always equal to `1`, and `Y` is a minor version. Server minor version is increased every time the API behavior is changed (note *Exceptions from Versioning*).

Note: Nova versioning documentation has a nice guide for developers on when to bump an API version.

The server indicates its minimum and maximum supported API versions in the `X-OpenStack-Ironic-API-Minimum-Version` and `X-OpenStack-Ironic-API-Maximum-Version` headers respectively, returned with every response. Client may request a specific API version by providing `X-OpenStack-Ironic-API-Version` header with request.

The requested microversion determines both the allowable requests and the response format for all requests. A resource may be represented differently based on the requested microversion.

If no version is requested by the client, the minimum supported version will be assumed. In this way, a client is only exposed to those API features that are supported in the requested (explicitly or implicitly) API version (again note *Exceptions from Versioning*, they are not covered by this rule).

We recommend clients that require a stable API to always request a specific version of API that they have been tested against.

Note: A special value `latest` can be requested instead a numerical microversion, which always requests the newest supported API version from the server.

REST API Versions History

REST API Version History

1.72 (Wallaby, 17.0)

Add support for `agent_status` and `agent_status_message` to /v1/heartbeat. These fields are used for external installation tools, such as Anaconda, to report back status.

1.71 (Wallaby, 17.0)

Signifier of the API supporting keystone `system` scoped roles and access controls. This is an informational flag for clients to be aware of the servers capability.

1.70 (Wallaby, 17.0)

Add support for `disable_ramdisk` parameter to provisioning endpoint `/v1/nodes/{node_ident}/states/provision`.

1.69 (Wallaby, 16.2)

Add support for `deploy_steps` parameter to provisioning endpoint `/v1/nodes/{node_ident}/states/provision`. Available and optional when target is active or rebuild.

1.68 (Victoria, 16.0)

Added the `agent_verify_ca` parameter to the ramdisk heartbeat API.

1.67 (Victoria, 15.1)

Add support for the mutually exclusive `port_uuid` and `portgroup_uuid` fields by having the node vif_attach API accept those values within `vif_info`. If one is specified, then Ironic will attempt to attach a VIF to the relative port or portgroup.

1.66 (Victoria, 15.1)

Add `network_data` field to the node object, that will be used by stand-alone ironic to pass L3 network configuration information to ramdisk.

1.65 (Ussuri, 15.0)

Added `lessee` field to the node object. The field should match the `project_id` of the intended lessee. If an allocation has an owner, then the allocation process will only match the allocation with a node that has the same `owner` or `lessee`.

1.64 (Ussuri, 15.0)

Added the `network_type` to the port objects `local_link_connection` field. The `network_type` can be set to either `managed` or `unmanaged`. When the type is `unmanaged` other fields are not required. Use `unmanaged` when the neutron `network_interface` is required, but the network is in fact a flat network where no actual switch management is done.

1.63 (Ussuri, 15.0)

Added the following new endpoints for indicator management:

- `GET /v1/nodes/<node_ident>/management/indicators` to list all available indicators names for each of the hardware component. Currently known components are: `chassis`, `system`, `disk`, `power` and `nic`.
- `GET /v1/nodes/<node_ident>/management/indicators/<component>/<indicator_ident>` to retrieve all indicators and their states for the hardware component.
- `PUT /v1/nodes/<node_ident>/management/indicators/<component>/<indicator_ident>` change state of the desired indicators of the component.

1.62 (Ussuri, 15.0)

This version of the API is to signify capability of an ironic deployment to support the `agent token` functionality with the `ironic-python-agent`.

1.61 (Ussuri, 14.0)

Added `retired` field to the node object to mark nodes for retirement. If set, this flag will move nodes to `manageable` upon automatic cleaning. `manageable` nodes which have this flag set cannot be moved to available. Also added `retired_reason` to specify the retirement reason.

1.60 (Ussuri, 14.0)

Added `owner` field to the allocation object. The field should match the `project_id` of the intended owner. If the `owner` field is set, the allocation process will only match the allocation with a node that has the same `owner` field set.

1.59 (Ussuri, 14.0)

Added the ability to specify a `vendor_data` dictionary field in the `configdrive` parameter submitted with the deployment of a node. The value is a dictionary which is served as `vendor_data2.json` in the config drive.

1.58 (Train, 12.2.0)

Added the ability to backfill allocations for already deployed nodes by creating an allocation with `node` set.

1.57 (Train, 12.2.0)

Added the following new endpoint for allocation:

- `PATCH /v1/allocations/<allocation_ident>` that allows updating `name` and `extra` fields for an existing allocation.

1.56 (Stein, 12.1.0)

Added the ability for the `configdrive` parameter submitted with the deployment of a node, to include a `meta_data`, `network_data` and `user_data` dictionary fields. Ironic will now use the supplied data to create a configuration drive for the user. Prior uses of the `configdrive` field are unaffected.

1.55 (Stein, 12.1.0)

Added the following new endpoints for deploy templates:

- `GET /v1/deploy_templates` to list all deploy templates.
- `GET /v1/deploy_templates/<deploy template identifier>` to retrieve details of a deploy template.
- `POST /v1/deploy_templates` to create a deploy template.
- `PATCH /v1/deploy_templates/<deploy template identifier>` to update a deploy template.
- `DELETE /v1/deploy_templates/<deploy template identifier>` to delete a deploy template.

1.54 (Stein, 12.1.0)

Added new endpoints for external `events`:

- POST /v1/events for creating events. (This endpoint is only intended for internal consumption.)

1.53 (Stein, 12.1.0)

Added `is_smartnic` field to the port object to enable Smart NIC port creation in addition to local link connection attributes `port_id` and `hostname`.

1.52 (Stein, 12.1.0)

Added allocation API, allowing reserving a node for deployment based on resource class and traits. The new endpoints are:

- `POST /v1/allocations` to request an allocation.
- `GET /v1/allocations` to list all allocations.
- `GET /v1/allocations/<ID or name>` to retrieve the allocation details.
- `GET /v1/nodes/<ID or name>/allocation` to retrieve an allocation associated with the node.
- `DELETE /v1/allocations/<ID or name>` to remove the allocation.
- `DELETE /v1/nodes/<ID or name>/allocation` to remove an allocation associated with the node.

Also added a new field `allocation_uuid` to the node resource.

1.51 (Stein, 12.1.0)

Added `description` field to the node object to enable operators to store any information relates to the node. The field is limited to 4096 characters.

1.50 (Stein, 12.1.0)

Added `owner` field to the node object to enable operators to store information in relation to the owner of a node. The field is up to 255 characters and MAY be used in a later point in time to allow designation and deligation of permissions.

1.49 (Stein, 12.0.0)

Added new endpoints for retrieving conductors information, and added a `conductor` field to node object.

1.48 (Stein, 12.0.0)

Added `protected` field to the node object to allow protecting deployed nodes from undeploying, rebuilding or deletion. Also added `protected_reason` to specify the reason of making the node protected.

1.47 (Stein, 12.0.0)

Added `automated_clean` field to the node object, enabling cleaning per node.

1.46 (Rocky, 11.1.0)

Added `conductor_group` field to the node and the node response, as well as support to the API to return results by matching the parameter.

1.45 (Rocky, 11.1.0)

Added `reset_interfaces` parameter to nodes PATCH request, to specify whether to reset hardware interfaces to their defaults on drivers update.

1.44 (Rocky, 11.1.0)

Added `deploy_step` to the node object, to indicate the current deploy step (if any) being performed on the node.

1.43 (Rocky, 11.0.0)

Added `?detail=` boolean query to the API list endpoints to provide a more RESTful alternative to the existing `/nodes/detail` and similar endpoints.

1.42 (Rocky, 11.0.0)

Added `fault` to the node object, to indicate currently detected fault on the node.

1.41 (Rocky, 11.0.0)

Added support to abort inspection of a node in the `inspect wait` state.

1.40 (Rocky, 11.0.0)

Added BIOS properties as sub resources of nodes:

- GET /v1/nodes/<node_ident>/bios
- GET /v1/nodes/<node_ident>/bios/<setting_name>

Added `bios_interface` field to the node object to allow getting and setting the interface.

1.39 (Rocky, 11.0.0)

Added `inspect wait` to available provision states. A node is shown as `inspect wait` instead of `inspecting` during asynchronous inspection.

1.38 (Queens, 10.1.0)

Added provision_state verbs `rescue` and `unrescue` along with the following states: `rescue`, `rescue failed`, `rescue wait`, `rescuing`, `unrescue failed`, and `unrescuing`. After rescuing a node, it will be left in the `rescue` state running a rescue ramdisk, configured with the `rescue_password`, and listening with ssh on the specified network interfaces. Unrescuing a node will return it to `active`.

Added `rescue_interface` to the node object, to allow setting the rescue interface for a dynamic driver.

1.37 (Queens, 10.1.0)

Adds support for node traits, with the following new endpoints.

- GET /v1/nodes/<node identifier>/traits lists the traits for a node.
- PUT /v1/nodes/<node identifier>/traits sets all traits for a node.
- PUT /v1/nodes/<node identifier>/traits/<trait> adds a trait to a node.
- DELETE /v1/nodes/<node identifier>/traits removes all traits from a node.
- DELETE /v1/nodes/<node identifier>/traits/<trait> removes a trait from a node.

A nodes traits are also included the following node query and list responses:

- GET /v1/nodes/<node identifier>
- GET /v1/nodes/detail
- GET /v1/nodes?fields=traits

Traits cannot be specified on node creation, nor can they be updated via a PATCH request on the node.

1.36 (Queens, 10.0.0)

Added `agent_version` parameter to deploy heartbeat request for version negotiation with Ironic Python Agent features.

1.35 (Queens, 9.2.0)

Added ability to provide `configdrive` when node is updated to `rebuild` provision state.

1.34 (Pike, 9.0.0)

Adds a `physical_network` field to the port object. All ports in a portgroup must have the same value in their `physical_network` field.

1.33 (Pike, 9.0.0)

Added `storage_interface` field to the node object to allow getting and setting the interface.

Added `default_storage_interface` and `enabled_storage_interfaces` fields to the driver object to show the information.

1.32 (Pike, 9.0.0)

Added new endpoints for remote volume configuration:

- GET /v1/volume as a root for volume resources
- GET /v1/volume/connectors for listing volume connectors
- POST /v1/volume/connectors for creating a volume connector
- GET /v1/volume/connectors/<UUID> for showing a volume connector
- PATCH /v1/volume/connectors/<UUID> for updating a volume connector
- DELETE /v1/volume/connectors/<UUID> for deleting a volume connector
- GET /v1/volume/targets for listing volume targets
- POST /v1/volume/targets for creating a volume target
- GET /v1/volume/targets/<UUID> for showing a volume target
- PATCH /v1/volume/targets/<UUID> for updating a volume target
- DELETE /v1/volume/targets/<UUID> for deleting a volume target

Volume resources also can be listed as sub resources of nodes:

- GET /v1/nodes/<node identifier>/volume
- GET /v1/nodes/<node identifier>/volume/connectors
- GET /v1/nodes/<node identifier>/volume/targets

1.31 (Ocata, 7.0.0)

Added the following fields to the node object, to allow getting and setting interfaces for a dynamic driver:

- boot_interface
- console_interface
- deploy_interface
- inspect_interface
- management_interface
- power_interface
- raid_interface
- vendor_interface

1.30 (Ocata, 7.0.0)

Added dynamic driver APIs:

- GET /v1/drivers now accepts a `type` parameter (optional, one of `classic` or `dynamic`), to limit the result to only classic drivers or dynamic drivers (hardware types). Without this parameter, both classic and dynamic drivers are returned.
- GET /v1/drivers now accepts a `detail` parameter (optional, one of `True` or `False`), to show all fields for a driver. Defaults to `False`.
- GET /v1/drivers now returns an additional `type` field to show if the driver is classic or dynamic.
- GET /v1/drivers/<name> now returns an additional `type` field to show if the driver is classic or dynamic.
- GET /v1/drivers/<name> now returns additional fields that are null for classic drivers, and set as following for dynamic drivers:
    - The value of the default_<interface-type>_interface is the entrypoint name of the calculated default interface for that type:
        - default_boot_interface
        - default_console_interface
        - default_deploy_interface
        - default_inspect_interface
        - default_management_interface
        - default_network_interface
        - default_power_interface
        - default_raid_interface
        - default_vendor_interface

- The value of the enabled_<interface-type>_interfaces is a list of entrypoint names of the enabled interfaces for that type:
  - enabled_boot_interfaces
  - enabled_console_interfaces
  - enabled_deploy_interfaces
  - enabled_inspect_interfaces
  - enabled_management_interfaces
  - enabled_network_interfaces
  - enabled_power_interfaces
  - enabled_raid_interfaces
  - enabled_vendor_interfaces

1.29 (Ocata, 7.0.0)

Add a new management API to support inject NMI, PUT /v1/nodes/(node_ident)/management/inject_nmi.

1.28 (Ocata, 7.0.0)

Add /v1/nodes/<node identifier>/vifs endpoint for attach, detach and list of VIFs.

1.27 (Ocata, 7.0.0)

Add `soft rebooting` and `soft power off` as possible values for the `target` field of the power state change payload, and also add `timeout` field to it.

1.26 (Ocata, 7.0.0)

Add portgroup `mode` and `properties` fields.

1.25 (Ocata, 7.0.0)

Add possibility to unset chassis_uuid from a node.

1.24 (Ocata, 7.0.0)

Added new endpoints /v1/nodes/<node>/portgroups and /v1/portgroups/<portgroup>/ports. Added new field `port.portgroup_uuid`.

1.23 (Ocata, 7.0.0)

Added /v1/portgroups/ endpoint.

1.22 (Newton, 6.1.0)

Added endpoints for deployment ramdisks.

1.21 (Newton, 6.1.0)

Add node `resource_class` field.

1.20 (Newton, 6.1.0)

Add node `network_interface` field.

1.19 (Newton, 6.1.0)

Add `local_link_connection` and `pxe_enabled` fields to the port object.

1.18 (Newton, 6.1.0)

Add `internal_info` readonly field to the port object, that will be used by ironic to store internal port-related information.

1.17 (Newton, 6.0.0)

Addition of provision_state verb `adopt` which allows an operator to move a node from `manageable` state to `active` state without performing a deployment operation on the node. This is intended for nodes that have already been deployed by external means.

1.16 (Mitaka, 5.0.0)

Add ability to filter nodes by driver.

1.15 (Mitaka, 5.0.0)

Add ability to do manual cleaning when a node is in the manageable provision state via PUT v1/nodes/<identifier>/states/provision, target:clean, clean_steps:[].

1.14 (Liberty, 4.2.0)

Make the following endpoints discoverable via Ironic API:

- /v1/nodes/<UUID or logical name>/states
- /v1/drivers/<driver name>/properties

1.13 (Liberty, 4.2.0)

Add a new verb `abort` to the API used to abort nodes in `CLEANWAIT` state.

1.12 (Liberty, 4.2.0)

This API version adds the following abilities:

- Get/set `node.target_raid_config` and to get `node.raid_config`.
- Retrieve the logical disk properties for the driver.

1.11 (Liberty, 4.0.0, breaking change)

Newly registered nodes begin in the `enroll` provision state by default, instead of `available`. To get them to the `available` state, the `manage` action must first be run to verify basic hardware control. On success the node moves to `manageable` provision state. Then the `provide` action must be run. Automated cleaning of the node is done and the node is made `available`.

1.10 (Liberty, 4.0.0)

Logical node names support all RFC 3986 unreserved characters. Previously only valid fully qualified domain names could be used.

1.9 (Liberty, 4.0.0)

Add ability to filter nodes by provision state.

1.8 (Liberty, 4.0.0)

Add ability to return a subset of resource fields.

1.7 (Liberty, 4.0.0)

Add node `clean_step` field.

1.6 (Kilo)

Add *Hardware Inspection* process: introduce `inspecting` and `inspectfail` provision states, and `inspect` action that can be used when a node is in `manageable` provision state.

1.5 (Kilo)

Add logical node names that can be used to address a node in addition to the node UUID. Name is expected to be a valid fully qualified domain name in this version of API.

1.4 (Kilo)

Add `manageable` state and `manage` transition, which can be used to move a node to `manageable` state from `available`. The node cannot be deployed in `manageable` state. This change is mostly a preparation for future inspection work and introduction of `enroll` provision state.

1.3 (Kilo)

Add node `driver_internal_info` field.

1.2 (Kilo, breaking change)

Renamed NOSTATE (`None` in Python, `null` in JSON) node state to `available`. This is needed to reduce confusion around `None` state, especially when future additions to the state machine land.

1.1 (Kilo)

This was the initial version when API versioning was introduced. Includes the following changes from Kilo release cycle:

- Add node `maintenance_reason` field and an API endpoint to set/unset the node maintenance mode.
- Add sync and async support for vendor passthru methods.
- Vendor passthru endpoints support different HTTP methods, not only `POST`.
- Make vendor methods discoverable via the Ironic API.
- Add logic to store the config drive passed by Nova.

This has been the minimum supported version since versioning was introduced.

1.0 (Juno)

This version denotes Juno API and was never explicitly supported, as API versioning was not implemented in Juno, and 1.1 became the minimum supported version in Kilo.

Exceptions from Versioning

The following API-visible things are not covered by the API versioning:

- Current node state is always exposed as it is, even if not supported by the requested API version, with exception of `available` state, which is returned in version 1.1 as `None` (in Python) or `null` (in JSON).
- Data within free-form JSON attributes: `properties`, `driver_info`, `instance_info`, `driver_internal_info` fields on a node object; `extra` fields on all objects.
- Addition of new drivers.
- All vendor passthru methods.

CHAPTER

EIGHT

COMMAND REFERENCES

Here are references for commands not elsewhere documented.

8.1 Command References

Here are references for commands not elsewhere documented.

8.1.1 ironic-dbsync

The **ironic-dbsync** utility is used to create the database schema tables that the ironic services will use for storage. It can also be used to upgrade existing database tables when migrating between different versions of ironic.

The Alembic library is used to perform the database migrations.

Options

This is a partial list of the most useful options. To see the full list, run the following:

```
ironic-dbsync --help
```

-h, --help
Show help message and exit.

--config-dir `<DIR>`
Path to a config directory with configuration files.

--config-file `<PATH>`
Path to a configuration file to use.

-d, --debug
Print debugging output.

--version
Show the programs version number and exit.

upgrade, stamp, revision, version, create_schema, online_data_migrations
The *command* to run.

Usage

Options for the various *commands* for **ironic-dbsync** are listed when the `-h` or `--help` option is used after the command.

For example:

```
ironic-dbsync create_schema --help
```

Information about the database is read from the ironic configuration file used by the API server and conductor services. This file must be specified with the `--config-file` option:

```
ironic-dbsync --config-file /path/to/ironic.conf create_schema
```

The configuration file defines the database backend to use with the *connection* database option:

```
[database]
connection=mysql+pymysql://root@localhost/ironic
```

If no configuration file is specified with the `--config-file` option, **ironic-dbsync** assumes an SQLite database.

Command Options

ironic-dbsync is given a command that tells the utility what actions to perform. These commands can take arguments. Several commands are available:

create_schema

-h, --help
Show help for create_schema and exit.

This command will create database tables based on the most current version. It assumes that there are no existing tables.

An example of creating database tables with the most recent version:

```
ironic-dbsync --config-file=/etc/ironic/ironic.conf create_schema
```

online_data_migrations

-h, --help
Show help for online_data_migrations and exit.

--max-count <NUMBER>
The maximum number of objects (a positive value) to migrate. Optional. If not specified, all the objects will be migrated (in batches of 50 to avoid locking the database for long periods of time).

--option <MIGRATION.KEY=VALUE>
If a migration accepts additional parameters, they can be passed via this argument. It can be specified several times.

This command will migrate objects in the database to their most recent versions. This command must be successfully run (return code 0) before upgrading to a future release.

It returns:

- 1 (not completed) if there are still pending objects to be migrated. Before upgrading to a newer release, this command must be run until 0 is returned.
- 0 (success) after migrations are finished or there are no data to migrate
- 127 (error) if max-count is not a positive value or an option is invalid
- 2 (error) if the database is not compatible with this release. This command needs to be run using the previous release of ironic, before upgrading and running it with this release.

revision

`-h, --help`
Show help for revision and exit.

`-m` `<MESSAGE>`**`, --message`** `<MESSAGE>`
The message to use with the revision file.

`--autogenerate`
Compares table metadata in the application with the status of the database and generates migrations based on this comparison.

This command will create a new revision file. You can use the `--message` option to comment the revision.

This is really only useful for ironic developers making changes that require database changes. This revision file is used during database migration and will specify the changes that need to be made to the database tables. Further discussion is beyond the scope of this document.

stamp

`-h, --help`
Show help for stamp and exit.

`--revision` `<REVISION>`
The revision number.

This command will stamp the revision table with the version specified with the `--revision` option. It will not run any migrations.

upgrade

-h, --help
Show help for upgrade and exit.

--revision <REVISION>
The revision number to upgrade to.

This command will upgrade existing database tables to the most recent version, or to the version specified with the `--revision` option.

Before this `upgrade` is invoked, the command **`ironic-dbsync online_data_migrations`** must have been successfully run using the previous version of ironic (if you are doing an upgrade as opposed to a new installation of ironic). If it wasnt run, the database will not be compatible with this recent version of ironic, and this command will return 2 (error).

If there are no existing tables, then new tables are created, beginning with the oldest known version, and successively upgraded using all of the database migration files, until they are at the specified version. Note that this behavior is different from the *create_schema* command that creates the tables based on the most recent version.

An example of upgrading to the most recent table versions:

```
ironic-dbsync --config-file=/etc/ironic/ironic.conf upgrade
```

Note: This command is the default if no command is given to **`ironic-dbsync`**.

Warning: The upgrade command is not compatible with SQLite databases since it uses ALTER TABLE commands to upgrade the database tables. SQLite supports only a limited subset of ALTER TABLE.

version

-h, --help
Show help for version and exit.

This command will output the current database version.

8.1.2 ironic-status

Synopsis

```
ironic-status <category> <command> [<args>]
```

Description

ironic-status is a tool that provides routines for checking the status of a Ironic deployment.

Options

The standard pattern for executing a **ironic-status** command is:

```
ironic-status <category> <command> [<args>]
```

Run without arguments to see a list of available command categories:

```
ironic-status
```

Categories are:

- `upgrade`

Detailed descriptions are below.

You can also run with a category argument such as `upgrade` to see a list of all commands in that category:

```
ironic-status upgrade
```

These sections describe the available categories and arguments for **ironic-status**.

Upgrade

ironic-status upgrade check Performs a release-specific readiness check before restarting services with new code. This command expects to have complete configuration and access to databases and services.

Return Codes

Return code	Description
0	All upgrade readiness checks passed successfully and there is nothing to do.
1	At least one check encountered an issue and requires further investigation. This is considered a warning but the upgrade may be OK.
2	There was an upgrade status check failure that needs to be investigated. This should be considered something that stops an upgrade.
255	An unexpected error occurred.

History of Checks

12.0.0 (Stein)

- Adds a check for compatibility of the object versions with the release of ironic.

Wallaby

- Adds a check to validate the configured policy file is not JSON based as JSON based policies have been deprecated.

CHAPTER
NINE

CONTRIBUTOR GUIDE

9.1 Developers Guide

9.1.1 Getting Started

If you are new to ironic, this section contains information that should help you get started as a developer working on the project or contributing to the project.

So You Want to Contribute

This document provides some necessary points for developers to consider when writing and reviewing Ironic code. The checklist will help developers get things right.

Getting Started

If youre completely new to OpenStack and want to contribute to the ironic project, please start by familiarizing yourself with the Infra Teams Developer Guide. This will help you get your accounts set up in Launchpad and Gerrit, familiarize you with the workflow for the OpenStack continuous integration and testing systems, and help you with your first commit.

LaunchPad

Most of the tools used for OpenStack require a launchpad.net ID for authentication. Ironic previously used to track work on Launchpad, but we have not done so since migrating to Storyboard.

See also:

- https://launchpad.net

Storyboard

The ironic project moved from Launchpad to StoryBoard for work and task tracking. This provides an aggregate view called a Project Group and individual Projects. A good starting place is the project group representing the whole of the ironic community, as opposed to the ironic project storyboard which represents ironic as a repository.

See *Bug Reporting and Triaging Guide* for more details on how we track bugs.

Internet Relay Chat IRC

Daily contributor discussions take place on IRC in the #openstack-ironic channel on the OFTC IRC network.

Please feel free to join us at ircs://irc.oftc.net:6697 and join our channel!

Additional information on getting connected can be found in the OpenStack community contribution guide.

Everything Ironic

Ironic is a community of projects centered around the primary project repository ironic, which help facilitate the deployment and management of bare metal resources.

This means there are a number of different repositories that fall into the responsibility of the project team and the community. Some of the repositories may not seem strictly hardware related, but they may be tools or things to just make an aspect easier.

Related Projects

There are several projects that are tightly integrated with ironic and which are developed by the same community.

See also:

- Bifrost Documentation
- Ironic Inspector Documentation
- Ironic Lib Documentation
- Ironic Python Agent (IPA) Documentation
- Ironic Client Documentation
- Ironic Inspector Client Documentation

Useful Links

Bug/Task tracker https://storyboard.openstack.org/#!/project/943

Mailing list (prefix Subject line with `[ironic]`) http://lists.openstack.org/cgi-bin/mailman/listinfo/openstack-discuss

Code Hosting https://opendev.org/openstack/ironic

Code Review https://review.opendev.org/#/q/status:open+project:openstack/ironic,n,z

Whiteboard https://etherpad.openstack.org/p/IronicWhiteBoard

Weekly Meeting Agenda https://wiki.openstack.org/wiki/Meetings/Ironic#Agenda_for_next_meeting

Adding New Features

Ironic tracks new features using RFEs (Requests for Feature Enhancements) instead of blueprints. These are stories with rfe tag, and they should be submitted before a spec or code is proposed.

When a member of the ironic-core team decides that the proposal is worth implementing, a spec (if needed) and code should be submitted, referencing the RFE task or story ID number. Contributors are welcome to submit a spec and/or code before the RFE is approved, however those patches will not land until the RFE is approved.

Feature Submission Process

1. Submit a bug report on the ironic StoryBoard. There are two fields that must be filled: Title and Description. Tasks can be added and are associated with a project. If you cant describe it in a sentence or two, it may mean that you are either trying to capture more than one RFE at once, or that you are having a hard time defining what you are trying to solve at all. This may also be a sign that your feature may require a specification document.
2. Describe the proposed change in the Description field. The description should provide enough details for a knowledgeable developer to understand what is the existing problem in the current platform that needs to be addressed, or what is the enhancement that would make the platform more capable, both from a functional and a non-functional standpoint.
3. Submit the story, add an rfe tag to it and assign yourself or whoever is going to work on this feature.
4. As soon as a member of the team acknowledges the story, we will move the story to the Review state. As time goes on, Discussion about the RFE, and whether to approve it will occur.
5. Contributors will evaluate the RFE and may advise the submitter to file a spec in the ironic-specs repository to elaborate on the feature request. Typically this is when an RFE requires extra scrutiny, more design discussion, etc. For the spec submission process, please see the *Ironic Specs Process*. A specific task should be created to track the creation of a specification.
6. If a spec is not required, once the discussion has happened and there is positive consensus among the ironic-core team on the RFE, the RFE is approved, and its tag will move from rfe to rfe-approved. This means that the feature is approved and the related code may be merged.

7. If a spec is required, the spec must be submitted (with a new task as part of the story referenced as Task in the commit message), reviewed, and merged before the RFE will be approved (and the tag changed to rfe-approved).
8. The tasks then goes through the usual process first to Review when the spec/code is being worked on, then Merged when it is implemented.
9. If the RFE is rejected, the ironic-core team will move the story to Invalid status.

Change Tracking

We track our stories and tasks in Storyboard.

https://storyboard.openstack.org/#!/project/ironic

When working on an RFE, please be sure to tag your commits properly: Story: #xxxx or Task: #xxxx. It is also helpful to set a consistent review topic, such as story/xxxx for all patches related to the RFE.

If the RFE spans across several projects (e.g. ironic and python-ironicclient), but the main work is going to happen within ironic, please use the same story for all the code youre submitting, there is no need to create a separate RFE in every project.

Note: RFEs may only be approved by members of the ironic-core team.

Note: While not strictly required for minor changes and fixes, it is highly preferred by the Ironic community that any change which needs to be backported, have a recorded Story and Task in Storyboard.

Managing Change Sets

If you would like some help, or if you (or some members of your team) are unable to continue working on the feature, updating and maintaining the changes, please let the rest of the ironic community know. You could leave a comment in one or more of the changes/patches, bring it up in IRC, the weekly meeting, or on the OpenStack development email list. Communicating this will make other contributors aware of the situation and allow for others to step forward and volunteer to continue with the work.

In the event that a contributor leaves the community, do not expect the contributors changes to be continued unless someone volunteers to do so.

Getting Your Patch Merged

Within the Ironic project, we generally require two core reviewers to sign-off (+2) change sets. We also will generally recognize non-core (+1) reviewers, and sometimes even reverse our decision to merge code based upon their reviews.

We recognize that some repositories have less visibility, as such it is okay to ask for a review in our IRC channel. Please be prepared to stay in IRC for a little while in case we have questions.

Sometimes we may also approve patches with a single core reviewer. This is generally discouraged, but sometimes necessary. When we do so, we try to explain why we do so. As a patch submitter, it equally helps us to understand why the change is important. Generally, more detail and context helps us understand the change faster.

Timeline Expectations

As with any large project, it does take time for features and changes to be merged in any of the project repositories. This is largely due to limited review bandwidth coupled with varying reviewer priorities and focuses.

When establishing an understanding of complexity, the following things should be kept in mind.

- Generally, small and minor changes can gain consensus and merge fairly quickly. These sorts of changes would be: bug fixes, minor documentation updates, follow-up changes.
- Medium changes generally consist of driver feature parity changes, where one driver is working to match functionality of another driver.
  - These changes generally only require an RFE for the purposes of tracking and correlating the change.
  - Documentation updates are expected to be submitted with or immediately following the initial change set.
- Larger or controversial changes generally take much longer to merge. This is often due to the necessity of reviewers to gain additional context and for change sets to be iterated upon to reach a state where there is consensus. These sorts of changes include: database, object, internal interface additions, RPC, rest API changes.
  - These changes will very often require specifications to reach consensus, unless there are pre-existing patterns or code already present.
  - These changes may require many reviews and iterations, and can also expect to be impacted by merge conflicts as other code or features are merged.
  - These changes must typically be split into a series of changes. Reviewers typically shy away from larger single change sets due to increased difficulty in reviewing.
  - Do not expect any API or user-visible data model changes to merge after the API client freeze. Some substrate changes may merge if not user visible.
- You should expect complex features, such as cross-project features or integration, to take longer than a single development cycle to land.
  - Building consensus is vital.
  - Often these changes are controversial or have multiple considerations that need to be worked through in the specification process, which may cause the design to change. As such, it may take months to reach consensus over design.
  - These features are best broken into larger chunks and tackled in an incremental fashion.

Live Upgrade Related Concerns

See *Rolling Upgrades*.

Driver Internal Info

The `driver_internal_info` node field was introduced in the Kilo release. It allows driver developers to store internal information that can not be modified by end users. Here is the list of existing common and agent driver attributes:

- Common attributes:
  - `is_whole_disk_image`: A Boolean value to indicate whether the user image contains ramdisk/kernel.
  - `clean_steps`: An ordered list of clean steps that will be performed on the node.
  - `deploy_steps`: An ordered list of deploy steps that will be performed on the node. Support for deploy steps was added in the `11.1.0` release.
  - `instance`: A list of dictionaries containing the disk layout values.
  - `root_uuid_or_disk_id`: A String value of the bare metal nodes root partition uuid or disk id.
  - `persistent_boot_device`: A String value of device from `ironic.common.boot_devices`.
  - `is_next_boot_persistent`: A Boolean value to indicate whether the next boot device is `persistent_boot_device`.
- Agent driver attributes:
  - `agent_url`: A String value of IPA API URL so that Ironic can talk to IPA ramdisk.
  - `hardware_manager_version`: A String value of the version of the hardware manager in IPA ramdisk.
  - `target_raid_config`: A Dictionary containing the target RAID configuration. This is a copy of the same name attribute in Node object. But this one is never actually saved into DB and is only read by IPA ramdisk.

Note: These are only some fields in use. Other vendor drivers might expose more `driver_internal_info` properties, please check their development documentation and/or module docstring for details. It is important for developers to make sure these properties follow the precedent of prefixing their variable names with a specific interface name (e.g., ilo_bar, drac_xyz), so as to minimize or avoid any conflicts between interfaces.

Ironic Specs Process

Specifications must follow the template which can be found at specs/template.rst, which is quite self-documenting. Specifications are proposed by adding them to the *specs/approved* directory, adding a soft link to it from the *specs/not-implemented* directory, and posting it for review to Gerrit. For more information, please see the README.

The same Gerrit process as with source code, using the repository ironic-specs, is used to add new specifications.

All approved specifications are available at: https://specs.openstack.org/openstack/ironic-specs. If a specification has been approved but not completed within one or more releases since the approval, it may be re-reviewed to make sure it still makes sense as written.

Ironic specifications are part of the *RFE (Requests for Feature Enhancements) process*. You are welcome to submit patches associated with an RFE, but they will have a -2 (do not merge) until the specification has been approved. This is to ensure that the patches dont get accidentally merged beforehand. You will still be able to get reviewer feedback and push new patch sets, even with a -2. The list of core reviewers for the specifications is small but mighty. (This is not necessarily the same list of core reviewers for code patches.)

Changes to existing specs

For approved but not-completed specs:

- cosmetic cleanup, fixing errors, and changing the definition of a feature can be done to the spec.

For approved and completed specs:

- changing a previously approved and completed spec should only be done for cosmetic cleanup or fixing errors.
- changing the definition of the feature should be done in a new spec.

Please see the Ironic specs process wiki page for further reference.

Bug Reporting

Bugs can reported via our Task and Bug tracking tool Storyboard.

When filing bugs, please include as much detail as possible, and dont be shy.

Essential pieces of information are generally:

- Contents of the node - *baremetal node show <uuid>*
- Steps to reproduce the issue.
- Exceptions and surrounding lines from the logs.
- Versions of ironic, ironic-python-agent, and any other coupled components.

Please also set your expectations of what *should* be happening. Statements of user expectations are how we understand what is occuring and how we learn new use cases!

Project Team Leader Duties

The `Project Team Leader` or `PTL` is elected each development cycle by the contributors to the ironic community.

Think of this person as your primary contact if you need to try and rally the project, or have a major issue that requires attention.

They serve a role that is mainly oriented towards trying to drive the technical discussion forward and managing the idiosyncrasies of the project. With this responsibility, they are considered a public face of the project and are generally obliged to try and provide project updates and outreach communication.

All common PTL duties are enumerated here in the PTL guide.

Tasks like release management or preparation for a release are generally delegated with-in the team. Even outreach can be delegated, and specifically there is no rule stating that any member of the community cant propose a release, clean-up release notes or documentation, or even get on the occasional stage.

Bug Reporting and Triaging Guide

StoryBoard

All ironic projects use StoryBoard for tracking both bugs and enhancement requests (RFE). The ironic project group lists all our projects.

Note: Ironic is developed as part of OpenStack and therefore uses the `openstack/` namespace.

StoryBoard is somewhat different from traditional bug tracking systems because every *story* is not linked to a project itself, but rather through its *tasks*. A story represents an issue you are facing or an enhancement you want to see, while tasks represent individual action items which can span several projects. When creating a story, youll also need to create the first task. If unsure, create a task against `openstack/ironic`.

Reporting Guide

We are constantly receiving a lot of requests, so its important to file a meaningful story for it to be acted upon. A good story:

- specifies **why** a change is needed.
- explains how to reproduce the described condition.

  Note: Please try to provide a reproducer based on unit tests, *devstack* or bifrost. While we try our best to support users using other installers and distributions, it may be non-trivial without deep knowledge of them. If youre using a commercial distribution or a product, please try contacting support first.

- should be understandable without additional context. For example, if you see an exception, we will need the full traceback.

- should not be too verbose either. Unfortunately, we cannot process a few days worth of system logs to find the problems, we expect your collaboration.
- is not a question or a support request. Please see *So You Want to Contribute* for the ways to contact us.
- provides a way to contact the reporter. Please follow the comments and expect follow-up emails, but ideally also be on IRC for questions.

An enhancement request additionally:

- benefits the overall project, not just one consumer. If you have a case that is specific to your requirements, think about ways to make ironic extensible to be able to cover it.
- does not unnecessary increase the project scope. Consider if your idea can be implemented without changing ironic or its projects, maybe it actually should?

Triaging Guide

The bug triaging process involves checking new stories to make sure they are actionable by the team. This guide is mostly targeting the project team, but we would appreciate if reporters could partly self-triage their own requests.

- Determine if the request is valid and complete. Use the checklist in the *Reporting Guide* for that.
- Is the request a bug report or an enhancement request (an RFE)? The difference is often subtle, the key question to answer is if the described behavior is expected.

  Add an `rfe` tag to all enhancement requests and propose it for the RFE Review section of the weekly meeting.
- Does the RFE obviously require a spec? Usually this is decided when an RFE is reviewed during the meeting, but some requests are undoubtedly complex, involve changing a lot of critical parts and thus demand a spec.

  Add a `needs-spec` tag to enhancement requests that obviously need a spec. Otherwise leave it until the meeting.
- Apply additional tags:
  - All hardware type specific stories should receive a corresponding tag (e.g. `ipmi`, `idrac`, etc).
  - API-related stories should have an `api` tag.
  - CI issues should have a `gate` tag.

The next actions **must only** be done by a core team member (or an experienced full-time contributor appoined by the PTL):

- Can the RFE be automatically approved? It happens if the RFE requests an implementation of a driver feature that is already implemented for other drivers and does not pose additional complexity.

  If the RFE can be automatically approved, apply the `rfe-approved` tag. If unsure, never apply the tag! Talk to the PTL instead.
- Does the RFE have a corresponding spec approved? If yes, apply the `rfe-approved` tag.
- In the end, apply the `ironic-triaged` tag to make the story as triaged.

Expiring Bugs

While we hope to fix all issues that our consumers hit, it is unfortunately not realistic. Stories **may** be closed by marking all their tasks `INVALID` in the following cases:

- No solution has been proposed in 1 calendar year.
- Additional information has been requested from the reporter, and no update has been provided in 1 calendar month.
- The request no longer aligns with the direction of the project.

Note: As usual, common sense should be applied when closing stories.

Developer Quick-Start

This is a quick walkthrough to get you started developing code for Ironic. This assumes you are already familiar with submitting code reviews to an OpenStack project.

The gate currently runs the unit tests under Python 3.6, 3.7, and 3.8. It is strongly encouraged to run the unit tests locally prior to submitting a patch.

Note: Do not run unit tests on the same environment as devstack due to conflicting configuration with system dependencies.

Note: This document is compatible with Python (3.6, 3.7, 3.8), Debian buster (10.8), Ubuntu Focal Fossa (20.04 LTS), RHEL8/CentOS Stream, openSUSE/SLE 15, and Fedora (33). When referring to different versions of Python and OS distributions, this is explicitly stated.

See also:

https://docs.openstack.org/infra/manual/developers.html#development-workflow

Prepare Development System

System Prerequisites

The following packages cover the prerequisites for a local development environment on most current distributions. Instructions for getting set up with non-default versions of Python and on older distributions are included below as well.

- Ubuntu/Debian:

```
sudo apt-get install build-essential python3-dev libssl-dev python3-
↪pip libmysqlclient-dev libxml2-dev libxslt-dev libpq-dev git git-
↪review libffi-dev gettext ipmitool psmisc graphviz libjpeg-dev
```

- RHEL/CentOS/Fedora:

```
sudo dnf install python3-devel openssl-devel python3-pip mysql-devel␣
↪libxml2-devel libxslt-devel postgresql-devel git git-review libffi-
↪devel gettext ipmitool psmisc graphviz gcc libjpeg-turbo-devel
```

- openSUSE/SLE:

```
sudo zypper install git git-review libffi-devel libmysqlclient-devel␣
↪libopenssl-devel libxml2-devel libxslt-devel postgresql-devel␣
↪python3-devel python-nose python3-pip gettext-runtime psmisc
```

To run the tests locally, it is a requirement that your terminal emulator supports unicode with the `en_US.UTF8` locale. If you use locale-gen to manage your locales, make sure you have enabled `en_US.UTF8` in `/etc/locale.gen` and rerun `locale-gen`.

Python Prerequisites

We suggest to use at least tox 3.9, if your distribution has an older version, you can install it using pip system-wise or better per user using the user option that by default will install the binary under $HOME/.local/bin, so you need to be sure to have that path in $PATH; for example:

```
pip install tox --user
```

will install tox as ~/.local/bin/tox

You may need to explicitly upgrade virtualenv if youve installed the one from your OS distribution and it is too old (tox will complain). You can upgrade it individually, if you need to:

```
pip install -U virtualenv --user
```

Running Unit Tests Locally

If you havent already, Ironic source code should be pulled directly from git:

```
# from your home or source directory
cd ~
git clone https://opendev.org/openstack/ironic
cd ironic
```

Running Unit and Style Tests

All unit tests should be run using tox. To run Ironics entire test suite:

```
# to run the py3 unit tests, and the style tests
tox
```

To run a specific test or tests, use the -e option followed by the tox target name. For example:

```
# run the unit tests under py36 and also run the pep8 tests
tox -epy36 -epep8
```

You may pass options to the test programs using positional arguments. To run a specific unit test, this passes the desired test (regex string) to stestr:

```
# run a specific test for Python 3.6
tox -epy36 -- test_conductor
```

Debugging unit tests

In order to break into the debugger from a unit test we need to insert a breaking point to the code:

```
import pdb; pdb.set_trace()
```

Then run `tox` with the debug environment as one of the following:

```
tox -e debug
tox -e debug test_file_name
tox -e debug test_file_name.TestClass
tox -e debug test_file_name.TestClass.test_name
```

For more information see the oslotest documentation.

Database Setup

The unit tests need a local database setup, you can use `tools/test-setup.sh` to set up the database the same way as setup in the OpenStack test systems.

Additional Tox Targets

There are several additional tox targets not included in the default list, such as the target which builds the documentation site. See the `tox.ini` file for a complete listing of tox targets. These can be run directly by specifying the target name:

```
# generate the documentation pages locally
tox -edocs

# generate the sample configuration file
tox -egenconfig
```

Exercising the Services Locally

In addition to running automated tests, sometimes it can be helpful to actually run the services locally, without needing a server in a remote datacenter.

If you would like to exercise the Ironic services in isolation within your local environment, you can do this without starting any other OpenStack services. For example, this is useful for rapidly prototyping and debugging interactions over the RPC channel, testing database migrations, and so forth.

Here we describe two ways to install and configure the dependencies, either run directly on your local machine or encapsulated in a virtual machine or container.

Step 1: Create a Python virtualenv

1. If you havent already downloaded the source code, do that first:

   ```
   cd ~
   git clone https://opendev.org/openstack/ironic
   cd ironic
   ```

2. Create the Python virtualenv:

   ```
   tox -evenv --notest --develop -r
   ```

3. Activate the virtual environment:

   ```
   . .tox/venv/bin/activate
   ```

4. Install the *openstack* client command utility:

   ```
   pip install python-openstackclient
   ```

5. Install the *baremetal* client:

   ```
   pip install python-ironicclient
   ```

   Note: You can install python-ironicclient from source by cloning the git repository and running *pip install .* while in the root of the cloned repository.

   ---

6. Export some ENV vars so the client will connect to the local services that youll start in the next section:

   ```
   export OS_AUTH_TYPE=none
   export OS_ENDPOINT=http://localhost:6385/
   ```

Next, install and configure system dependencies.

Step 2: Install System Dependencies Locally

This step will install MySQL on your local system. This may not be desirable in some situations (eg, youre developing from a laptop and do not want to run a MySQL server on it all the time). If you want to use SQLite, skip it and do not set the `connection` option.

1. Install mysql-server:

   Ubuntu/Debian:

   ```
   sudo apt-get install mysql-server
   ```

   RHEL/CentOS/Fedora:

   ```
   sudo dnf install mariadb mariadb-server
   sudo systemctl start mariadb.service
   ```

   openSUSE/SLE:: sudo zypper install mariadb sudo systemctl start mysql.service

If using MySQL, you need to create the initial database:

```
mysql -u root -pMYSQL_ROOT_PWD -e "create schema ironic"
```

Note: if you choose not to install mysql-server, ironic will default to using a local sqlite database. The database will then be stored in `ironic/ironic.sqlite`.

---

2. Create a configuration file within the ironic source directory:

```
# generate a sample config
tox -egenconfig

# copy sample config and modify it as necessary
cp etc/ironic/ironic.conf.sample etc/ironic/ironic.conf.local

# disable auth since we are not running keystone here
sed -i "s/#auth_strategy = keystone/auth_strategy = noauth/" etc/
↪ironic/ironic.conf.local

# use the 'fake-hardware' test hardware type
sed -i "s/#enabled_hardware_types = .*/enabled_hardware_types = fake-
↪hardware/" etc/ironic/ironic.conf.local

# use the 'fake' deploy and boot interfaces
sed -i "s/#enabled_deploy_interfaces = .*/enabled_deploy_interfaces =␣
↪fake/" etc/ironic/ironic.conf.local
sed -i "s/#enabled_boot_interfaces = .*/enabled_boot_interfaces =␣
↪fake/" etc/ironic/ironic.conf.local

# enable both fake and ipmitool management and power interfaces
sed -i "s/#enabled_management_interfaces = .*/enabled_management_
↪interfaces = fake,ipmitool/" etc/ironic/ironic.conf.local
sed -i "s/#enabled_power_interfaces = .*/enabled_power_interfaces =␣
↪fake,ipmitool/" etc/ironic/ironic.conf.local

# change the periodic sync_power_state_interval to a week, to avoid␣
↪getting NodeLocked exceptions
sed -i "s/#sync_power_state_interval = 60/sync_power_state_interval =␣
↪604800/" etc/ironic/ironic.conf.local

# if you opted to install mysql-server, switch the DB connection from␣
↪sqlite to mysql
sed -i "s/#connection = .*/connection = mysql\+pymysql:\/\/root:MYSQL_
↪ROOT_PWD@localhost\/ironic/" etc/ironic/ironic.conf.local

# use JSON RPC to avoid installing rabbitmq locally
sed -i "s/#rpc_transport = oslo/rpc_transport = json-rpc/" etc/ironic/
↪ironic.conf.local
```

Step 3: Start the Services

From within the python virtualenv, run the following command to prepare the database before you start the ironic services:

```
# initialize the database for ironic
ironic-dbsync --config-file etc/ironic/ironic.conf.local create_schema
```

Next, open two new terminals for this section, and run each of the examples here in a separate terminal. In this way, the services will *not* be run as daemons; you can observe their output and stop them with Ctrl-C at any time.

1. Start the API service in debug mode and watch its output:

   ```
   cd ~/ironic
   . .tox/venv/bin/activate
   ironic-api -d --config-file etc/ironic/ironic.conf.local
   ```

2. Start the Conductor service in debug mode and watch its output:

   ```
   cd ~/ironic
   . .tox/venv/bin/activate
   ironic-conductor -d --config-file etc/ironic/ironic.conf.local
   ```

Step 4: Interact with the running services

You should now be able to interact with ironic via the python client, which is present in the python virtualenv, and observe both services debug outputs in the other two windows. This is a good way to test new features or play with the functionality without necessarily starting DevStack.

To get started, export the following variables to point the client at the local instance of ironic and disable the authentication:

```
export OS_AUTH_TYPE=none
export OS_ENDPOINT=http://127.0.0.1:6385
```

Then list the available commands and resources:

```
# get a list of available commands
openstack help baremetal

# get the list of drivers currently supported by the available conductor(s)
baremetal driver list

# get a list of nodes (should be empty at this point)
baremetal node list
```

Here is an example walkthrough of creating a node:

```
MAC="aa:bb:cc:dd:ee:ff"   # replace with the MAC of a data port on your␣
↪node
IPMI_ADDR="1.2.3.4"       # replace with a real IP of the node BMC
IPMI_USER="admin"         # replace with the BMC's user name
IPMI_PASS="pass"          # replace with the BMC's password
```

(continues on next page)

(continued from previous page)

```
# enroll the node with the fake hardware type and IPMI-based power and
# management interfaces. Note that driver info may be added at node
# creation time with "--driver-info"
NODE=$(baremetal node create \
       --driver fake-hardware \
       --management-interface ipmitool \
       --power-interface ipmitool \
       --driver-info ipmi_address=$IPMI_ADDR \
       --driver-info ipmi_username=$IPMI_USER \
       -f value -c uuid)

# driver info may also be added or updated later on
baremetal node set $NODE --driver-info ipmi_password=$IPMI_PASS

# add a network port
baremetal port create $MAC --node $NODE

# view the information for the node
baremetal node show $NODE

# request that the node's driver validate the supplied information
baremetal node validate $NODE

# you have now enrolled a node sufficiently to be able to control
# its power state from ironic!
baremetal node power on $NODE
```

If you make some code changes and want to test their effects, simply stop the services with Ctrl-C and restart them.

Step 5: Fixing your test environment

If you are testing changes that add or remove python entrypoints, or making significant changes to ironics python modules, or simply keep the virtualenv around for a long time, your development environment may reach an inconsistent state. It may help to delete cached .pyc files, update dependencies, reinstall ironic, or even recreate the virtualenv. The following commands may help with that, but are not an exhaustive troubleshooting guide:

```
# clear cached pyc files
cd ~/ironic/ironic
find ./ -name '*.pyc' | xargs rm

# reinstall ironic modules
cd ~/ironic
. .tox/venv/bin/activate
pip uninstall ironic
pip install -e .

# install and upgrade ironic and all python dependencies
cd ~/ironic
. .tox/venv/bin/activate
pip install -U -e .
```

Deploying Ironic with DevStack

DevStack may be configured to deploy Ironic, setup Nova to use the Ironic driver and provide hardware resources (network, baremetal compute nodes) using a combination of OpenVSwitch and libvirt. It is highly recommended to deploy on an expendable virtual machine and not on your personal work station. Deploying Ironic with DevStack requires a machine running Ubuntu 16.04 (or later) or Fedora 24 (or later). Make sure your machine is fully up to date and has the latest packages installed before beginning this process.

The ironic-tempest-plugin is necessary if you want to run integration tests, the section *Ironic with ironic-tempest-plugin* tells the extra steps you need to enable it in DevStack.

See also:

https://docs.openstack.org/devstack/latest/

Note: The devstack demo tenant is now granted the baremetal_observer role and thereby has read-only access to ironics API. This is sufficient for all the examples below. Should you want to create or modify bare metal resources directly (ie. through ironic rather than through nova) you will need to use the devstack admin tenant.

Devstack will no longer create the user stack with the desired permissions, but does provide a script to perform the task:

```
git clone https://opendev.org/openstack/devstack.git devstack
sudo ./devstack/tools/create-stack-user.sh
```

Switch to the stack user and clone DevStack:

```
sudo su - stack
git clone https://opendev.org/openstack/devstack.git devstack
```

Ironic

Create devstack/local.conf with minimal settings required to enable Ironic. An example local.conf that enables both `direct` and `iscsi` *deploy interfaces* and uses the `ipmi` hardware type by default:

```
cd devstack
cat >local.conf <<END
[[local|localrc]]
# Credentials
ADMIN_PASSWORD=password
DATABASE_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password
SWIFT_HASH=password
SWIFT_TEMPURL_KEY=password

# Enable Ironic plugin
enable_plugin ironic https://opendev.org/openstack/ironic

```

(continues on next page)

(continued from previous page)

```
# Disable nova novnc service, ironic does not support it anyway.
disable_service n-novnc

# Enable Swift for the direct deploy interface.
enable_service s-proxy
enable_service s-object
enable_service s-container
enable_service s-account

# Disable Horizon
disable_service horizon

# Disable Cinder
disable_service cinder c-sch c-api c-vol

# Swift temp URL's are required for the direct deploy interface
SWIFT_ENABLE_TEMPURLS=True

# Create 3 virtual machines to pose as Ironic's baremetal nodes.
IRONIC_VM_COUNT=3
IRONIC_BAREMETAL_BASIC_OPS=True
DEFAULT_INSTANCE_TYPE=baremetal

# Enable additional hardware types, if needed.
#IRONIC_ENABLED_HARDWARE_TYPES=ipmi,fake-hardware
# Don't forget that many hardware types require enabling of additional
# interfaces, most often power and management:
#IRONIC_ENABLED_MANAGEMENT_INTERFACES=ipmitool,fake
#IRONIC_ENABLED_POWER_INTERFACES=ipmitool,fake
# The 'ipmi' hardware type's default deploy interface is 'iscsi'.
# This would change the default to 'direct':
#IRONIC_DEFAULT_DEPLOY_INTERFACE=direct

# Change this to alter the default driver for nodes created by devstack.
# This driver should be in the enabled list above.
IRONIC_DEPLOY_DRIVER=ipmi

# The parameters below represent the minimum possible values to create
# functional nodes.
IRONIC_VM_SPECS_RAM=2048
IRONIC_VM_SPECS_DISK=10

# Size of the ephemeral partition in GB. Use 0 for no ephemeral partition.
IRONIC_VM_EPHEMERAL_DISK=0

# To build your own IPA ramdisk from source, set this to True
IRONIC_BUILD_DEPLOY_RAMDISK=False

VIRT_DRIVER=ironic

# By default, DevStack creates a 10.0.0.0/24 network for instances.
# If this overlaps with the hosts network, you may adjust with the
# following.
NETWORK_GATEWAY=10.1.0.1
FIXED_RANGE=10.1.0.0/24
FIXED_NETWORK_SIZE=256
```

(continues on next page)

(continued from previous page)

```
# Log all output to files
LOGFILE=$HOME/devstack.log
LOGDIR=$HOME/logs
IRONIC_VM_LOG_DIR=$HOME/ironic-bm-logs

END
```

Ironic with ironic-tempest-plugin

Using the stack user, clone the ironic-tempest-plugin repository in the same directory you cloned DevStack:

```
git clone https://opendev.org/openstack/ironic-tempest-plugin.git
```

An example local.conf that enables the ironic tempest plugin and Ironic can be found below. The `TEMPEST_PLUGINS` variable needs to have the absolute path to the ironic-tempest-plugin folder, otherwise the plugin wont be installed. Ironic will have enabled both `direct` and `iscsi` *deploy interfaces* and uses the `ipmi` hardware type by default:

```
cd devstack
cat >local.conf <<END
[[local|localrc]]
# Credentials
ADMIN_PASSWORD=password
DATABASE_PASSWORD=password
RABBIT_PASSWORD=password
SERVICE_PASSWORD=password
SERVICE_TOKEN=password
SWIFT_HASH=password
SWIFT_TEMPURL_KEY=password

# Enable Ironic plugin
enable_plugin ironic https://opendev.org/openstack/ironic

# Disable nova novnc service, ironic does not support it anyway.
disable_service n-novnc

# Enable Swift for the direct deploy interface.
enable_service s-proxy
enable_service s-object
enable_service s-container
enable_service s-account

# Disable Horizon
disable_service horizon

# Disable Cinder
disable_service cinder c-sch c-api c-vol

# Swift temp URL's are required for the direct deploy interface
SWIFT_ENABLE_TEMPURLS=True
```

(continues on next page)

(continued from previous page)

```
# Create 3 virtual machines to pose as Ironic's baremetal nodes.
IRONIC_VM_COUNT=3
IRONIC_BAREMETAL_BASIC_OPS=True
DEFAULT_INSTANCE_TYPE=baremetal

# Enable additional hardware types, if needed.
#IRONIC_ENABLED_HARDWARE_TYPES=ipmi,fake-hardware
# Don't forget that many hardware types require enabling of additional
# interfaces, most often power and management:
#IRONIC_ENABLED_MANAGEMENT_INTERFACES=ipmitool,fake
#IRONIC_ENABLED_POWER_INTERFACES=ipmitool,fake
# The 'ipmi' hardware type's default deploy interface is 'iscsi'.
# This would change the default to 'direct':
#IRONIC_DEFAULT_DEPLOY_INTERFACE=direct

# Change this to alter the default driver for nodes created by devstack.
# This driver should be in the enabled list above.
IRONIC_DEPLOY_DRIVER=ipmi

# The parameters below represent the minimum possible values to create
# functional nodes.
IRONIC_VM_SPECS_RAM=2048
IRONIC_VM_SPECS_DISK=10

# Size of the ephemeral partition in GB. Use 0 for no ephemeral partition.
IRONIC_VM_EPHEMERAL_DISK=0

# To build your own IPA ramdisk from source, set this to True
IRONIC_BUILD_DEPLOY_RAMDISK=False

VIRT_DRIVER=ironic

# By default, DevStack creates a 10.0.0.0/24 network for instances.
# If this overlaps with the hosts network, you may adjust with the
# following.
NETWORK_GATEWAY=10.1.0.1
FIXED_RANGE=10.1.0.0/24
FIXED_NETWORK_SIZE=256

# Log all output to files
LOGFILE=$HOME/devstack.log
LOGDIR=$HOME/logs
IRONIC_VM_LOG_DIR=$HOME/ironic-bm-logs
TEMPEST_PLUGINS="/opt/stack/ironic-tempest-plugin"

END
```

Note: Some tests may be skipped depending on the configuration of your environment, they may be reliant on a driver or a capability that you did not configure.

Deployment

Note: Git protocol requires access to port 9418, which is not a standard port that corporate firewalls always allow. If you are behind a firewall or on a proxy that blocks Git protocol, modify the `enable_plugin` line to use `https://` instead of `git://` and add `GIT_BASE=https://opendev.org` to the credentials:

```
GIT_BASE=https://opendev.org

# Enable Ironic plugin
enable_plugin ironic https://opendev.org/openstack/ironic
```

Note: When the `ipmi` hardware type is used and IRONIC_IS_HARDWARE variable is `false` devstack will automatically set up VirtualBMC to control the power state of the virtual baremetal nodes.

Note: When running QEMU as non-root user (e.g. `qemu` on Fedora or `libvirt-qemu` on Ubuntu), make sure `IRONIC_VM_LOG_DIR` points to a directory where QEMU will be able to write. You can verify this with, for example:

```
# on Fedora
sudo -u qemu touch $HOME/ironic-bm-logs/test.log
# on Ubuntu
sudo -u libvirt-qemu touch $HOME/ironic-bm-logs/test.log
```

Note: To check out an in-progress patch for testing, you can add a Git ref to the `enable_plugin` line. For instance:

```
enable_plugin ironic https://opendev.org/openstack/ironic refs/changes/46/
↪295946/15
```

For a patch in review, you can find the ref to use by clicking the Download button in Gerrit. You can also specify a different git repo, or a branch or tag:

```
enable_plugin ironic https://github.com/openstack/ironic stable/kilo
```

For more details, see the devstack plugin interface documentation.

Run stack.sh:

```
./stack.sh
```

Source credentials, create a key, and spawn an instance as the `demo` user:

```
. ~/devstack/openrc

# query the image id of the default cirros image
```

(continues on next page)

(continued from previous page)

```
image=$(openstack image show $DEFAULT_IMAGE_NAME -f value -c id)

# create keypair
ssh-keygen
openstack keypair create --public-key ~/.ssh/id_rsa.pub default

# spawn instance
openstack server create --flavor baremetal --image $image --key-name␣
↪default testing
```

Note: Because devstack create multiple networks, we need to pass an additional parameter `--nic net-id` to the nova boot command when using the admin account, for example:

```
net_id=$(openstack network list | egrep "$PRIVATE_NETWORK_NAME"'[^-]' |␣
↪awk '{ print $2 }')

openstack server create --flavor baremetal --nic net-id=$net_id --image
↪$image --key-name default testing
```

You should now see a Nova instance building:

```
openstack server list --long
+----------+---------+--------+------------+-------------+----------+------
↪------+----------+-------------------+------+------------+
| ID       | Name    | Status | Task State | Power State | Networks |␣
↪Image Name | Image ID | Availability Zone | Host | Properties |
+----------+---------+--------+------------+-------------+----------+------
↪------+----------+-------------------+------+------------+
| a2c7f812 | testing | BUILD  | spawning   | NOSTATE     |          |␣
↪cirros-0.3 | 44d4092a | nova              |      |            |
| -e386-4a |         |        |            |             |          | .5-
↪x86_64- | -51ac-47 |                   |      |            |
| 22-b393- |         |        |            |             |          | disk␣
↪      | 51-9c50- |                   |      |            |
| fe1802ab |         |        |            |             |          |     ␣
↪      | fd6e2050 |                   |      |            |
| d56e     |         |        |            |             |          |     ␣
↪      | faa1     |                   |      |            |
+----------+---------+--------+------------+-------------+----------+------
↪------+----------+-------------------+------+------------+
```

Nova will be interfacing with Ironic conductor to spawn the node. On the Ironic side, you should see an Ironic node associated with this Nova instance. It should be powered on and in a wait call-back provisioning state:

```
baremetal node list
+--------------------------------------+--------+--------------------------
↪------------+-------------+--------------------+-------------+
| UUID                                 | Name   | Instance UUID            ␣
↪            | Power State | Provisioning State | Maintenance |
+--------------------------------------+--------+--------------------------
↪------------+-------------+--------------------+-------------+
| 9e592cbe-e492-4e4f-bf8f-4c9e0ad1868f | node-0 | None                     ␣
↪            | power off   | None               | False       |
```

(continues on next page)

(continued from previous page)

```
| ec0c6384-cc3a-4edf-b7db-abde1998be96 | node-1 | None                    ␣
↪             | power off   | None               | False       |
| 4099e31c-576c-48f8-b460-75e1b14e497f | node-2 | a2c7f812-e386-4a22-b393-
↪fe1802abd56e | power on    | wait call-back     | False       |
+--------------------------------------+--------+-------------------------
↪-------------+-------------+--------------------+-------------+
```

At this point, Ironic conductor has called to libvirt (via virtualbmc) to power on a virtual machine, which will PXE + TFTP boot from the conductor node and progress through the Ironic provisioning workflow. One libvirt domain should be active now:

```
sudo virsh list --all
 Id    Name                           State
----------------------------------------------------
 2     node-2                         running
 -     node-0                         shut off
 -     node-1                         shut off
```

This provisioning process may take some time depending on the performance of the host system, but Ironic should eventually show the node as having an active provisioning state:

```
baremetal node list
+--------------------------------------+--------+-------------------------
↪-------------+-------------+--------------------+-------------+
| UUID                                 | Name   | Instance UUID           ␣
↪             | Power State | Provisioning State | Maintenance |
+--------------------------------------+--------+-------------------------
↪-------------+-------------+--------------------+-------------+
| 9e592cbe-e492-4e4f-bf8f-4c9e0ad1868f | node-0 | None                    ␣
↪             | power off   | None               | False       |
| ec0c6384-cc3a-4edf-b7db-abde1998be96 | node-1 | None                    ␣
↪             | power off   | None               | False       |
| 4099e31c-576c-48f8-b460-75e1b14e497f | node-2 | a2c7f812-e386-4a22-b393-
↪fe1802abd56e | power on    | active             | False       |
+--------------------------------------+--------+-------------------------
↪-------------+-------------+--------------------+-------------+
```

This should also be reflected in the Nova instance state, which at this point should be ACTIVE, Running and an associated private IP:

```
openstack server list --long
+----------+---------+--------+------------+-------------+---------------+-
↪-----------+----------+-------------------+------+------------+
| ID       | Name    | Status | Task State | Power State | Networks      |␣
↪Image Name | Image ID | Availability Zone | Host | Properties |
+----------+---------+--------+------------+-------------+---------------+-
↪-----------+----------+-------------------+------+------------+
| a2c7f812 | testing | ACTIVE | none       | Running     | private=10.1. |␣
↪cirros-0.3 | 44d4092a | nova              |      |            |
| -e386-4a |         |        |            |             | 0.4, fd7d:1f3 |␣
↪.5-x86_64- | -51ac-47 |                   |      |            |
| 22-b393- |         |        |            |             | c:4bf1:0:f816 |␣
↪disk       | 51-9c50- |                   |      |            |
| fe1802ab |         |        |            |             | :3eff:f39d:6d |␣
↪           | fd6e2050 |                   |      |            |
```

(continues on next page)

(continued from previous page)

```
| d56e     |         |        |            |             | 94           |␣
↪            | faa1     |                   |      |            |
+----------+---------+--------+------------+-------------+---------------+-
↪------------+----------+-------------------+------+------------+
```

The server should now be accessible via SSH:

```
ssh cirros@10.1.0.4
$
```

Running Tempest tests

After *Deploying Ironic with DevStack* with the ironic-tempest-plugin enabled, one might want to run integration tests against the running cloud. The Tempest project is the project that offers an integration test suite for OpenStack.

First, navigate to Tempest directory:

```
cd /opt/stack/tempest
```

To run all tests from the Ironic plugin, execute the following command:

```
tox -e all -- ironic
```

To limit the amount of tests that you would like to run, you can use a regex. For instance, to limit the run to a single test file, the following command can be used:

```
tox -e all -- ironic_tempest_plugin.tests.scenario.test_baremetal_basic_ops
```

Debugging Tempest tests

It is sometimes useful to step through the test code, line by line, especially when the error output is vague. This can be done by running the tests in debug mode and using a debugger such as pdb.

For example, after editing the *test_baremetal_basic_ops* file and setting up the pdb traces you can invoke the `run_tempest.sh` script in the Tempest directory with the following parameters:

```
./run_tempest.sh -N -d ironic_tempest_plugin.tests.scenario.test_baremetal_
↪basic_ops
```

- The *-N* parameter tells the script to run the tests in the local environment (without a virtualenv) so it can find the Ironic tempest plugin.
- The *-d* parameter enables the debug mode, allowing it to be used with pdb.

For more information about the supported parameters see:

```
./run_tempest.sh --help
```

Note: Always be careful when running debuggers in time sensitive code, they may cause timeout errors that werent there before.

OSProfiler Tracing in Ironic

OSProfiler is an OpenStack cross-project profiling library. It is being used among OpenStack projects to look at performance issues and detect bottlenecks. For details on how OSProfiler works and how to use it in ironic, please refer to OSProfiler Support Documentation.

Building developer documentation

If you would like to build the documentation locally, eg. to test your documentation changes before uploading them for review, run these commands to build the documentation set:

- On your local machine:

```
# activate your development virtualenv
. .tox/venv/bin/activate

# build the docs
tox -edocs

#Now use your browser to open the top-level index.html located at:

ironic/doc/build/html/index.html
```

- On a remote machine:

```
# Go to the directory that contains the docs
cd ~/ironic/doc/source/

# Build the docs
tox -edocs

# Change directory to the newly built HTML files
cd ~/ironic/doc/build/html/

# Create a server using python on port 8000
python -m SimpleHTTPServer 8000

#Now use your browser to open the top-level index.html located at:

http://your_ip:8000
```

Developer FAQ (frequently asked questions)

Here are some answers to frequently-asked questions from IRC and elsewhere.

- *How do I*
  - *create a migration script template?*
  - *know if a release note is needed for my change?*
  - *create a new release note?*
  - *update a release note?*
  - *get a decision on something?*
  - *add support for GMRs to new executables and extending the GMR?*

How do I

create a migration script template?

Using the `ironic-dbsync revision` command, e.g:

```
$ cd ironic
$ tox -evenv -- ironic-dbsync revision -m \"create foo table\"
```

It will create an empty alembic migration. For more information see the alembic documentation.

know if a release note is needed for my change?

Reno documentation contains a description of what can be added to each section of a release note. If, after reading this, youre still unsure about whether to add a release note for your change or not, keep in mind that it is intended to contain information for deployers, so changes to unit tests or documentation are unlikely to require one.

create a new release note?

By running `reno` command via tox, e.g:

```
$ tox -e venv -- reno new version-foo
  venv create: /home/foo/ironic/.tox/venv
  venv installdeps: -r/home/foo/ironic/test-requirements.txt
  venv develop-inst: /home/foo/ironic
  venv runtests: PYTHONHASHSEED='0'
  venv runtests: commands[0] | reno new version-foo
  Created new notes file in releasenotes/notes/version-foo-
↪ecb3875dc1cbf6d9.yaml
    venv: commands succeeded
    congratulations :)
```

(continues on next page)

(continued from previous page)

```
$ git status
  On branch test
  Untracked files:
    (use "git add <file>..." to include in what will be committed)

    releasenotes/notes/version-foo-ecb3875dc1cbf6d9.yaml
```

Then edit the result file. Note that:

- we prefer to use present tense in release notes. For example, a release note should say Adds support for feature foo, not Added support for feature foo. (We use adds instead of add because grammatically, it is ironic adds support, not ironic add support.)
- any variant of English spelling (American, British, Canadian, Australian) is acceptable. The release note itself should be consistent and not have different spelling variants of the same word.

For more information see the reno documentation.

update a release note?

If this is a release note that pertains to something that was fixed on master or an intermediary release (during a development cycle, that hasnt been branched yet), you can go ahead and update it by submitting a patch.

If it is the release note of an ironic release that has branched, it can be updated but we will only allow it in extenuating circumstances. (It can be updated by *only* updating the file in that branch. DO NOT update the file in master and cherry-pick it. If you do, see how the mess was cleaned up.)

get a decision on something?

You have an issue and would like a decision to be made. First, make sure that the issue hasnt already been addressed, by looking at documentation, stories, specifications, or asking. Information and links can be found on the Ironic wiki page.

There are several ways to solicit comments and opinions:

- bringing it up at the weekly Ironic meeting
- bringing it up on IRC
- bringing it up on the mailing list (add [Ironic] to the Subject of the email)

If there are enough core folks at the weekly meeting, after discussing an issue, voting could happen and a decision could be made. The problem with IRC or the weekly meeting is that feedback will only come from the people that are actually present.

To inform (and solicit feedback from) more people about an issue, the preferred process is:

1. bring it up on the mailing list
2. after some period of time has elapsed (and depending on the thread activity), someone should propose a solution via gerrit. (E.g. the person that started the thread if no one else steps up.) The proposal should be made in the git repository that is associated with the issue. (For instance, this decision process was proposed as a documentation patch to the ironic repository.)

3. In the email thread, dont forget to provide a link to the proposed patch!
4. The discussion then moves to the proposed patch. If this is a big decision, we could declare that some percentage of the cores should vote on it before landing it.

(This process was suggested in an email thread about process for making decisions.)

add support for GMRs to new executables and extending the GMR?

For more information, see the oslo.reports documentation page.

Contributor Vision

Background

During the Rocky Project Teams Gathering (Feburary/March 2018), The contributors in the room at that time took a few minutes to write out each contributors vision of where they see ironic in five years time.

After everyone had a chance to spend a few minutes writing, we went around the room and gave every contributor the chance to read their vision and allow other contributors to ask questions to better understand what each individual contributor wrote. While we were doing that, we also took time to capture the common themes.

This entire exercise did result in some laughs and a common set of words, and truly helped to ensure that the entire team proceeded to use the same words to describe various aspects as the sessions progressed during the week. We also agreed that we should write a shared vision, to have something to reference and remind us of where we want to go as a community.

Rocky Vision: For 2022-2023

Common Themes

Below is an entirely unscientific summary of common themes that arose during the discussion among fourteen contributors.

- Contributors picked a time between 2020, and 2023.
- 4 Contributors foresee ironic being the leading Open Source baremetal deployment technology
- 2 Contributors foresee ironic reaching feature parity with Nova.
- 2 Contributors foresee users moving all workloads to the cloud
- 1 Contributor foresees Kubernetes and Container integration being the major focus of Bare Metal as a Service further down the road.
- 2 Contributors foresee greater composible hardware being more common.
- 1 Contributor foresees ironic growing into or supporting CMDBs.
- 2 Contributors foresee that features are more micro-service oriented.
- 2 Contributors foresee that ironic supported all of the possible baremetal management needs
- 1 Contributor foresees standalone use being more common.

- 2 Contributors foresee the ironics developer community growing
- 2 Contributors foresee that auto-discovery will be more common.
- 2 Contributors foresee ironic being used for devices beyond servers, such as lightbulbs, IOT, etc.

Vision Statement

The year is 2022. Were meeting to plan the Z release of Ironic. We stopped to reflect upon the last few years of Ironics growth, how we had come such a long way to become the defacto open source baremetal deployment technology. How we had grown our use cases, and support for consumers such as containers, and users who wished to managed specialized fleets of composed machines.

New contributors and their different use cases have brought us closer to parity with virtual machines. Everyday were gaining word of more operators adopting the ironic communitys CMDB integration to leverage hardware discovery. Weve heard of operators deploying racks upon racks of new hardware by just connecting the power and network cables, and from there the operators have discovered time to write the worlds greatest operator novel with the time saved in commissioning new racks of hardware.

Time has brought us closer and taught us to be more collaborative across the community, and we look forward to our next release together.

Comparison to the 2018 OpenStack Technical Vision

In late-2018, the OpenStack Technical composed a technical vision of what OpenStack clouds should look like. While every component differs, and cloudy interactions change dramatically the closer to physical hardware one gets, there are a few areas where Ironic could use some improvement.

This list is largely for the purposes of help wanted. It is also important to note that Ironic as a project has a vision document for itself.

The Pillars of Cloud - Self Service

- Ironics mechanisms and tooling are low level infrastructure mechanisms and as such there has never been a huge emphasis or need on making Ironic be capable of offering direct multi-tenant interaction. Most users interact with the bare metal managed by Ironic via Nova, which abstracts away many of these issues. Eventually, we should offer direct multi-tenancy which is not oriented towards admin-only.

Design Goals - Built-in Reliability and Durability

- Ironic presently considers in-flight operations as failed upon the restart of a controller that was previously performing a task, because we do not know the current status of the task upon re-start. In some cases, this makes sense, but potentially requires administrative intervention in the worst of cases. In a perfect universe, Ironic conductors would validate their perception, in case tasks actually finished.

Design Goals - Graphical User Interface

- While a graphical interface was developed for Horizon in the form of ironic-ui, currently ironic-ui receives only minimal housekeeping. As Ironic has evolved, ironic-ui is stuck on version *1.34* and knows nothing of our evolution since. Ironic ultimately needs a contributor with sufficient time to pick up `ironic-ui` or to completely replace it as a functional and customizable user interface.

The following pages describe the architecture of the Bare Metal service and may be helpful to anyone working on or with the service, but are written primarily for developers.

System Architecture

High Level description

An Ironic deployment will be composed of the following components:

- An admin-only RESTful API service, by which privileged users, such as cloud operators and other services within the cloud control plane, may interact with the managed bare metal servers.
- A Conductor service, which does the bulk of the work. Functionality is exposed via the API service. The Conductor and API services communicate via RPC.
- A Database and DB API for storing the state of the Conductor and Drivers.
- A Deployment Ramdisk or Deployment Agent, which provide control over the hardware which is not available remotely to the Conductor. A ramdisk should be built which contains one of these agents, eg. with diskimage-builder. This ramdisk can be booted on-demand.

Note: The agent is never run inside a tenant instance.

Drivers

The internal driver API provides a consistent interface between the Conductor service and the driver implementations. A driver is defined by a *hardware type* deriving from the AbstractHardwareType class, defining supported *hardware interfaces*. See *Enabling drivers and hardware types* for a more detailed explanation. See *Pluggable Drivers* for an explanation on how to write new hardware types and interfaces.

Driver-Specific Periodic Tasks

Drivers may run their own periodic tasks, i.e. actions run repeatedly after a certain amount of time. Such a task is created by using the periodic decorator on an interface method. For example

```
from futurist import periodics

class FakePower(base.PowerInterface):
    @periodics.periodic(spacing=42)
    def task(self, manager, context):
        pass  # do something
```

Here the `spacing` argument is a period in seconds for a given periodic task. For example spacing=5 means every 5 seconds.

Driver-Specific Steps

Drivers may have specific steps that may need to be executed or offered to a user to execute in order to perform specific configuration tasks.

These steps should ideally be located on the management interface to enable consistent user experience of the hardware type. What should be avoided is duplication of existing interfaces such as the deploy interface to enable vendor specific cleaning or deployment steps.

Message Routing

Each Conductor registers itself in the database upon start-up, and periodically updates the timestamp of its record. Contained within this registration is a list of the drivers which this Conductor instance supports. This allows all services to maintain a consistent view of which Conductors and which drivers are available at all times.

Based on their respective driver, all nodes are mapped across the set of available Conductors using a consistent hashing algorithm. Node-specific tasks are dispatched from the API tier to the appropriate conductor using conductor-specific RPC channels. As Conductor instances join or leave the cluster, nodes may be remapped to different Conductors, thus triggering various driver actions such as take-over or clean-up.

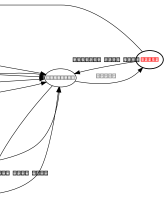
Ironics State Machine

State Machine Diagram

The diagram below shows the provisioning states that an Ironic node goes through during the lifetime of a node. The diagram also depicts the events that transition the node to different states.

Stable states are highlighted with a thicker border. All transitions from stable states are initiated by API requests. There are a few other API-initiated-transitions that are possible from non-stable states. The events for these API-initiated transitions are indicated with (via API). Internally, the conductor initiates the other transitions (depicted in gray).

State Descriptions



enroll (sta

This is the state that all node start off in whe created using API version 1.11 or newer. When a node is in the `enroll` state, the only thing ironic knows about it is that it exists, and ironic cannot take any further action by itself. Once a node has its driver/interfaces and their required information set in `node.driver_info`, the node can be transitioned to the `verifying` state by setting the nodes provision state using the `manage` verb.

verifying
ironi will validate that it can manage the nod using the information given in `node.driver_info` and with either the driver/hardware type and interfaces it has been assigned. This involves going out and confirming that the credentials work to access whatever node control mechanism they talk to.

manageab
Onc ironi has verified that it can manage the nod using the driver/interfaces and credentials passed in at node create time, the node will be transitioned to the `manageable` state. From `manageable`, nodes can transition to:

- `man` (thro `cle` by setting the nod

provision state using the `clean` verb.

- `man` (thro `ins` by setting the node provision state using the `inspect` verb.
- `ava` (thro `cle` if automati clea ing is enable by setting the nodes provision state using the `provide` verb.
- `act` (thro `adc` by setting the node provision state us-

ing the `adopt` verb.

`man` is the state that a node shou be mov into whe any updates need to be made to it such as changes to fields in driver_info and updates to networking information on ironic ports assigned to the node.

`man` is also the only stable state that can be transition to, from these failure states:

- `adc fai`
- `cle fai`
- `ins fai`

inspecting

`ins` will utilize node introspec tion to update hard derived node properties to reflect the current state of the hardware. Typically, the node will transition to `manageable` if inspection is synchronous, or `inspect wait` if asynchronous. The node will transition to `inspect failed` if error occurred.

inspect wa

This is the provision state used whe an asyn chro inspection is in progress. A successfully inspected node shall transition to `manageable` state.

inspect fai

This is the state a node will mov into whe inspec tion of the node fails. From here the node can transitioned to:

- `ins` by setting the node provision

state using the `inspect` verb.

- `man` by setting the node provision state using the `manage` verb

cleaning

Nod in the `cle` state are being scru and re-progran into a known configuration.

Whe a node is in the `cle` state it mea that the conductor is executing the clean step (for out-of-band clean steps) or preparing the environment (building PXE configuration files, configuring the DHCP, etc) to boot the ramdisk for running in-band clean steps.

clean wai

Just like the `cle` state the node in the `cle` `wai` state are being scrubbed and reprogrammed. The difference is that in the `clean wait` state the conductor is waiting for the ramdisk to boot or the clean step which is running in-band to finish.

The clea ing process of a node in the `cle` `wai` state can be interrupted by setting the nodes provision state using the `abort` verb if the task that is running allows it.

available (

After node have been success fully preconfigured and cleaned, they are moved into the `available` state and are ready to be provisioned. From `available`, nodes can transition to:

- `act` (thro `dep` by setting the node provision state using the `active` verb.
- `man` by setting the node provision state using the `manage` verb

deploying
Nod in `dep` are being prepared to run a work load on them. This consists of running a series of tasks, such as:

- Setting appropriate BIO configuration

- Partitioning driv and laying dow file system
- Creating any additiona resour (nod spec networ

config, a config drive partition, etc.) that may be required by additional subsystems.

wait call-

Just like the `dep` state the node in `wai` `cal` are being deployed. The difference is that in `wait call-back` the conductor is waiting for the ramdisk to boot or execute parts of the deployment which need to run in-band on the node (for example, installing the bootloader, or writing the image to the disk).

The deploy men of a node in `wai` `cal` can be interrupted by setting the nodes provision state using the `deleted` verb.

deploy fai

This is the state a node will mov into whe a deploy ment fails, for example a timeout waiting for the ramdisk to PXE boot. From here the node can be transitioned to:

- `act` (thro `dep` by setting the node provision state using either the `active` or `rebuild` verbs.

- `ava` (thro `del` and `cle` by setting the node provision

state using the `deleted` verb.

active (sta

Nod in `act` have a work load running on them ironi may

collect out-of-band sensor information (including power state) on a regular basis. Nodes in `active` can transition to:

- `ava` (thro `del` and `cle` by setting the node provision

state using the `deleted` verb.

- `act` (thro `dep`

by setting the node provision state using the `rebuild` verb.

- `res` (thro `res` by setting the node provision state using the `rescue` verb.

deleting

Nod in `del` state are being torn down from running an active workload. In `deleting`, ironic tears down and removes any configuration and resources it added in `deploying` or `rescuing`.

error (stal

This is the state a node will

mov into whe dele ing an

active deployment fails. From `error`, nodes can transition to:

- `ava` (thro `del` and `cle` by setting the node provision

state using the `deleted` verb.

adopting

This state allows ironi to take over man agemen of a

baremetal node with an existing workload on it. Ordinarily when a baremetal node is enrolled and managed by ironic, it must transition through `cleaning` and `deploying` to reach `active` state. However, those baremetal nodes that have an existing workload on them, do not need to be deployed or cleaned again, so this transition allows these nodes to move directly from `manageable` to `active`.

rescuing

Nod in `res` are being prepare to

perform rescue operations. This consists of running a series of tasks, such as:

- Setting appropriate BIO configurations
- Creating any additiona resour (nod spec network config, etc.) that may be required by additional subsystems.

rescue wa

Just like the `res` state the node in `res wai` are being rescued. The difference is that in `rescue wait` the conductor is waiting for the ramdisk to boot or execute parts of the rescue which need to run in-band on the node (for example, setting the password for user named `rescue`).

The rescue operation of a node in `res` `wai` can be aborted by setting the nodes provision state using the `abort` verb.

rescue fai

This is the state a node will mov into whe a rescue operation fails, for example a timeout waiting for the ramdisk to PXE boot. From here the node can be transitioned to:

- `res` (thro `res` by setting the node provision state using the `rescue` verb.
- `act` (thro `unr`

by setting the node provision state us-

ing the `unrescue` verb.

- `ava`
  (thro
  `del`
  by setting the node provision state us-

ing the `deleted` verb.

rescue (sta
Nod in `res` have a rescue ramd running on them. Iron

may collect out-of-band sensor information (including power state) on a regular basis. Nodes in `rescue` can transition to:

- `act`
  (thro
  `unr`
  by setting the

node provision state using the `unrescue` verb.

- `ava` (thro `del` by setting the node provision state using the `deleted` verb.

unrescuin

Nod in `unr` are being prepare to transition to `active` state from `rescue` state. This consists of running a series of tasks, such as setting appropriate BIOS configurations such as changing boot device.

unrescue f

This is the state a node will mov into whe an unrescue operation fails. From here the node can be transitioned to:

- `res` (thro `res` by setting the node provision state using the `rescue` verb.
- `act` (thro `unr` by setting the node provision state using the `unrescue` verb.
- `ava` (thro `del` by setting the node provision state using the `deleted` verb.

Developing New Notifications

Iron notifications are even intend for consum tion by external services. Notifications are sent to these services over a message bus by oslo.messagings Notifier class. For more information about configuring notifications and available notifications, see *Notifications*.

Iron also has a set of base class that assist in clear defining the notification itself, the payload, and the other fields not auto-generated by oslo (level, event_type and publisher_id). Below describes how to use these base classes to add a new notification to ironic.

Adding a new notification to ironic

To add a new notification to ironi a

new versioned notification class should be created by subclassing the NotificationBase class to define the notification itself and the NotificationPayloadBase class to define which fields the new notification will contain inside its payload. You may also define a schema to allow the payload to be automatically populated by the fields of an ironic object. Heres an example:

```
↪#
↪T
↪i
↪o
↪w
↪f
↪y
↪w
↪t
↪u
↪i
↪y
↪s
@ba
↪I
↪r
cla
↪E
↪I
␣
↪␣
↪␣
↪␣
↪
↪#
↪V
↪1
↪0
↪I
↪v
␣
↪␣
↪␣
↪␣
↪V
↪=
↪
↪'
↪0
↪'
␣
↪␣
↪␣
↪␣
↪f
↪=
↪
↪{
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
```

(continues on next page)

(continued from previous page)

```
↪StringField(),

↪StringField()
```

(continues on next page)

(continued from previous page)

```
↪')
```

```
@ba
↪I
↪r
cla
↪E
↪N
␣
↪␣
↪␣
↪␣
↪
↪#
↪V
↪1
↪0
↪I
↪V
␣
↪␣
↪␣
↪␣
↪V
↪=
↪
↪'
↪0
↪'
␣
↪␣
↪␣
↪␣
↪f
↪=
↪
↪{
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪'
↪'
↪f
↪O
↪'

␣
↪␣
↪␣
↪␣
↪}
↪
```

(continues on next page)

(continued from previous page)

```
↪to reuse other objects'

↪populate_schema with
```

(continues on next page)

(continued from previous page)

```
↪'example_obj', 'a_useful_field')
```

(continues on next page)

(continued from previous page)

```
↪StringField(),
↪StringField(nullable=True)
```

```
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪'
↪u
↪f
↪'
↪f

␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪'
↪e
↪f
↪'
↪f

␣
↪␣
↪␣
↪}
↪
```

Note that both the payload and notification class are oslo versioned objects. Modifications to these require a version bump so that consumers of notifications know when the notifications have changed.

SCH defines how to populate the payload field Its an optional attribute that subclasses may use to easily populate notifications with data from other objects.

It is a dictionary whe every key valu pair has the following format:

```
↪<
↪f
↪n
↪:
↪(
↪<
↪s
↪n
↪,
↪
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
```

(continues on next page)

```
↪           <field_of_the_data_source>)
```

(continued from previous page)

The `<pa` is the nam whe the data will be store in the payload object; this field has to be defined as a field of the payload. The `<data_source_name>` shall refer to name of the parameter passed as kwarg to the payloads `populate_schema()` call and this object will be used as the source of the data. The `<field_of_the_data_source>` shall be a valid field of the passed argument.

The SCH need to be applied with the `pop` call before the notification can be emitted.

The valu of the `pay` `<pa` field will be set by the `<da` `<fi`

field. The `<data_source_name>` will not be part of the payload object internal or external representation.

Payload fields that are not set by the SCH can be filled in the same way as in any versioned object.

Ther to create a payload you wou do som thin like the following. Note that if you choose to define a schema in the SCHEMA class variable, you must populate the schema by calling `populate_schema(example_obj=my_example_obj)` before emitting the notification is allowed:

```
my_
↪e
↪o
↪=
↪E
↪
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪                    a_useful_field='important',
```

(continues on next page)

(continued from previous page)

```
↪                       not_useful_field='blah')

↪the SCHEMA and is a
```

```
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣


↪#
↪a
↪e
↪f
↪i
↪o
↪s
↪i
↪'
↪n
↪a
↪p
↪o


↪#
↪n
↪f
↪i
↪t
↪c
↪f
my_
↪n
↪p
↪=
↪E
↪e
↪f
↪'
↪'


↪#
↪p
↪t
↪s
↪w
↪t
↪E
↪f
```

(continues on next page)

(continued from previous page)

```
my_
↪n
↪p
↪p
↪s
↪o
↪e
↪o
```

You then create the notification with the oslo required fields (event_type, publisher_id, and level, all sender fields needed by oslo that are defined in the ironic notification base classes) and emit it:

```
not
↪=
↪E
␣
↪␣
↪␣
↪␣
↪e
↪t
↪E
↪'
↪o
↪'
↪
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪a
↪'
↪S
↪'
↪␣
↪s
↪NotificationStatus.START),
```

(continues on next page)

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪p
↪N
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪s
↪'
↪C
↪'
↪
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪h
↪'
↪'
↪
␣
↪␣
↪␣
↪␣
↪l
↪N
↪D
↪
␣
↪␣
↪␣
↪␣
↪p
↪n
↪p
not
↪e
```

Whe
spec
ifying
the
ever

obj will spec ify the object being acted on, `action` will be a string describing what action is being performed on that object, and `status` will be one of start, end, error, or success. start and end are used to indicate when actions that are not immediate begin and succeed. success is used to indicate when actions that are immediate succeed. error is used to indicate when any type of action fails, regardless of whether its immediate or not. As a result of specifying these parameters, event_type will be formatted as `baremetal.<object>.<action>.<status>` on the message bus.

This example will send the following notification over the message bus:

```
↪{
␣
↪␣
↪␣
↪␣
↪
↪"
↪"
↪
↪"
↪"
↪
␣
↪␣
↪␣
↪␣
↪
↪"
↪"
↪{
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
```

(continues on next page)

(continued from previous page)

```
↪",
```

```
↪0",
```

```
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪o
↪n
↪"
↪"

␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪o
↪v
↪"
↪"

␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪o
↪d
↪"
↪{
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
```

(continues on next page)

```
↪"a_useful_field":"important",
```

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"an_extra_field":"hello"
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪}
↪
␣
↪␣
↪␣
↪␣
↪}
↪,
↪
␣
↪␣
↪␣
↪␣
↪
↪"
↪t
↪"
↪"
↪e
↪o
↪d
↪s
↪s
↪",
␣
↪␣
↪␣
↪␣
↪
↪"
↪i
↪"
↪"
↪c
↪h
↪"
```

(continues on next page)

(continued from previous page)

```
␣
↪}
↪
```

About OSProfiler

OSProfiler is an Ope Stac cros proj profiling li brar

Its API provides different ways to add a new trace point. Trace points contain two messages (start and stop). Messages like below are sent to a collector:

```
↪{
␣
↪␣
↪␣
↪␣
↪
↪"
↪"
↪
↪<
↪n
↪-
↪(
↪
␣
↪␣
↪␣
↪␣
↪
↪"
↪i
↪"
↪
↪<
↪,
↪
␣
↪␣
↪␣
↪␣
↪
↪"
↪i
↪"
↪
```

(continues on next page)

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪
↪"
↪i
↪"
↪
↪<
↪,
↪
␣
↪␣
↪␣
↪␣
↪
↪"
↪"
↪
↪<
↪
}
↪
```

The field are defined as follows

bas -

<uui that is same for all trace poin that be- long

to one trace. This is used to simplify the process of retrieving all trace points (related to one trace) from the collector.

par -

<uui of parent trace poin

tra
-

<uui of current trace poin

inf
-

the dictionary that contains user information passed when calling profiler start() & stop() methods.

The profiler uses ceilo ter as a centralized collector.

Two other alternatives for ceilometer are pure MongoDB driver and Elasticsearch.

A notifier

is setup to send notifications to ceilometer using oslo.messaging and ceilometer API is used to retrieve all messages related to one trace.

OSProfiler has entry poin that allows the user to retrieve information about traces and present it in HTML/JSON using CLI.

For more details see OSProfiler Cross proje profiling library.

How to Use OSProfiler with Ironic in Devstack

To use or test OSProfiler in ironi the user need to setu

Devstack with OSProfiler and ceilometer. In addition to the setup described at *Deploying Ironic with DevStack*, the user needs to do the following:

Add the following to loc to enable OSProfiler and

ceilometer:

```
ena
↪p
↪p
↪h
↪/
↪o
↪o
↪o
↪p
ena
↪p
↪c
↪h
↪/
↪o
↪o
↪o
↪c
```

(continues on next page)

(continued from previous page)

```
ena
↪p
↪o
↪h
↪/
↪o
↪o
↪o
↪o


↪#
↪E
↪t
↪f
↪s
CEI
↪N
↪T
↪p
ENA
↪S
↪c
↪a
↪c
↪a
ENA
↪S
↪c
↪a
↪c
↪c
ENA
↪S
↪c
↪a
↪e
↪c
↪a
↪n
ENA
↪S
↪c
↪a
```

Run
stac

Onc
De-
vs-
tack
en-
vi-
ron-
men

is setu edit iro con to

set the following profiler options and restart ironic services:

```
[pr
ena
↪=
↪T
hma
↪k
↪=
↪S
↪K
↪
↪#
↪d
↪v
↪u
↪a
↪s
↪O
↪p
tra
↪s
↪=
↪T
```

In order to trac ironi using OS-Profiler, use oper stac

client to run baremetal commands with `--os-profile SECRET_KEY`.

For example, the following

will caus a <tra id> to

be printed after node list:

```
↪$
↪o
↪-
↪-
↪o
↪p
↪S
↪K
↪b
↪n
↪l
```

Out-put of the above com man will in-clud the fol-low-ing:

```
Tra
↪I
↪
↪<
↪i
↪
Dis
↪t
↪w
↪c
osp
↪t
↪s
↪-
↪-
↪h
↪
↪<
↪i
↪
```

(continues on next page)

(continued from previous page)

The trac results can be seen using this comman

```
↪$
↪o
↪t
↪s
↪-
↪-
↪h
↪
↪<
↪i
↪
```

The trac results can be save in a file with --c fil option:

```
↪$
↪o
↪t
↪s
↪-
↪-
↪h
↪
↪<
↪i
↪␣
↪-
↪-
```

(continues on next page)

```
↪out trace.html
```

(continued from previous page)

The trace results shov the time spen in iron api, iron cond and

db calls. More detailed db tracing is enabled if `trace_sqlalchemy` is set to true.

References

- OS-Profiler Cros proj profiling librar
- *Deploying Iron with DevStack*

Rolling Upgrades

The ironi (iron api and ironi conc service support rolli upgrad

starting with a rolling upgrade from the Ocata to the Pike release. This describes the design of rolling upgrades, followed by notes for developing new features or modifying an IronicObject.

Design

Rolling upgrades between releases

Iron follows the relea cycl with inter release mod The release

are semantic-versioned, in the form <major>.<minor>.<patch>. We refer to a `named release` of ironic as the release associated with a development cycle like Pike.

In addition, ironi follows the standard

deprecation

policy, which says that the deprecation period must be at least three months and a cycle boundary. This means that there will never be anything that is both deprecated *and* removed between two named releases.

Roll upgrad will be support betwee

- nam releas N to N+1 (star ing with N == Oca
- any nam releas to its latest revision containing backported bug fixes. Because those bug fixes can contain improvements to the upgrade process, the operator should patch the system before upgrading between named releases.
- mos recent

nam releas N (and sem releas new than N) to master. As with the above bullet point, there may be a bug or a feature introduced on a master branch, that we want to remove before publishing a named release. Deprecation policy allows to do this in a 3 month time frame. If the feature was included and removed in intermediate releases, there should be a release note added, with instructions on how to do a rolling upgrade to master from an affected release or release span. This would typically instruct the operator to upgrade to a particular intermediate release, before upgrading to master.

Rolling upgrade process

Iron support rolli upgrad as describ in the *upgrad guid*

The upgrad process will caus the ironi service to be running the `FromVer` and `ToVer` releases in this order:

0.

Upgrad ironi code and run datab sche migrations via the `ironic-dbsync upgrade` command.

1. Upgrad code and resta ironi cond service one at a time
2. Upgrad code and resta ironi api service one at a time
3. Unpin API RPC and object

version so that the services can now use the latest versions in `ToVer`. This is done via updating the configuration option described below in *API, RPC and object version pinning* and then restarting the services. ironic-conductor services should be restarted first, followed by the ironic-api services. This is to ensure that when new functionality is exposed on the unpinned API service (via API micro version), it is available on the backend.

step	ironic-api	ironic-conductor
0	all FromVer	all FromVer
1.1	all FromVer	some FromVer, some ToVer-pinned
1.2	all FromVer	all ToVer-pinned
2.1	some FromVer, some ToVer-pinned	all ToVer-pinned
2.2	all ToVer-pinned	all ToVer-pinned
3.1	all ToVer-pinned	some ToVer-pinned, some ToVer
3.2	all ToVer-pinned	all ToVer
3.3	some ToVer-pinned, some ToVer	all ToVer
3.4	all ToVer	all ToVer

Policy for changes to the DB model

The policy for chan to the DB mod is as follows

- Add new item to the DB mod is support

- The dropping of colu or tables and correspon ing objects fields is subject to ironics deprecation policy. But its alembic script has to wait one more deprecation period, otherwise an `unknown column` exception will be thrown when `FromVer` services access the DB. This is because **`ironic-dbsync upgrade`** upgrades the DB schema but `FromVer` services still contain the dropped field in their SQLAlchemy DB model.

- An `ale` `op.` `alt` to renam or resize a column is not allowed. Instead, split it into multiple operations, with one operation per release cycle (to maintain compatibility with an old SQLAlchemy model). For example, to rename a column, add the new column in release N, then remove the old column in release N+1.

- Som implementation of SQL `ALT` `TAB` such as addi foreign keys in PostgreSQL, may impose table locks and cause downtime. If the change cannot be avoided and the impact is significant (e.g. the table can be frequently accessed and/or store a large dataset), these cases must be mentioned in the release notes.

API, RPC and object version pinning

For the iron service to be running old and new release at the same time during a rolling upgrade, the services need to be able to handle different API, RPC and object versions.

This versioning is handled via the configuration option: `[DEFAULT]/pin_release_version`. It is used to pin the API, RPC and IronicObject (e.g., Node, Conductor, Chassis, Port, and Portgroup) versions for all the ironic services.

The default valu of emp indicate that iron api and ironi conductor will use the latest versions of API, RPC and IronicObjects. Its possible values are releases, named (e.g. `ocata`) or sem-versioned (e.g. `7.0`).

Internally in com mon ironi main tains a map ping that indicates the API, RPC and IronicObject versions associated with each release. This mapping is maintained manually.

During a rolli upgrad the service using the new release will set the configuration option value to be the name (or version) of the old release. This will indicate to the services running the new release, which API, RPC and object versions that they should be compatible with, in order to communicate with the services using the old release.

Handling API versions

Whe the (new service is pinn the max

imun API version it supports will be the pinned version which the older service supports (as described above at *API, RPC and object version pinning*). The ironic-api service returns HTTP status code 406 for any requests with API versions that are higher than this maximum version.

Handling RPC versions

Con ductorA sets the `ver` variable to the desired (latest or pinned) RPC API version and passes it to the `RPCClient` as an initialization parameter. This variable is then used to determine the maximum requested message version that the `RPCClient` can send.

Eac RPC call can custom the reques accord ing to this `version_cap`. The *Ironic RPC versions* section below has more details about this.

Handling IronicObject versions

Internally ironi services deal with IronicObjects in their latest versions. Only at these boundaries, when the IronicObject enters or leaves the service, do we deal with object versioning:

- getting objects from the data convert to latest version
- saving objects to the data if pinn save in pinn version; else save in latest version
-

serializing objects (to send over RPC if pinn send pinned version; else send latest version

- deserializing objects (receiving objects from RPC): convert to latest version

The ironi api service also has to handle API reques base on whether or how a feature is supported by the API version and object versions. For example, when the ironic-api service is pinned, it can only allow actions that are available to the objects pinned version, and cannot allow actions that are only available for the latest version of that object.

To support this:

- All the data tables (SQ models) of the IronicObjects have a column named `version`. The value is the version of the object that is saved in the database.
- The meth `Irc` `get` returns the target version. If pinn the pinned version is returned. Otherwise, the latest version is returned.
- The meth `Irc` `con` converts the object into the target version. The target version may be a newer or older version than the existing version of the object. The bulk of the work is done in the helper method `IronicObject._convert_to_version()`. Subclasses that have new versions redefine this to perform the actual conversions.

In the following,

- The old releas is `Fro` it uses version 1.14 of a Node object.
- The new releas is `ToV` It uses version 1.15 of a Node object this has a deprecated `extra` field and a new `meta` field that replaces `extra`.
- db_ and db_ are the data representations of thos node fields.

Getting objects from the database (API/conductor < DB)

Both ironi api and ironi conc service read values from the data

These values are converted to IronicObjects via the method `IronicObject._from_db_object()`. This method always returns the IronicObject in its latest version, even if it was in an older version in the database. This is done regardless of the service being pinned or not.

Note that if an object is converte to a later version that IronicObject will retain any changes (in its `_changed_fields` field) resulting from that conversion. This is needed in case the object gets saved later, in the latest version.

For example, if the node in the data is in version 1.14 and has db_obj[extra] set:

- a `Fro` service will get a Nod with node = db_c (and

no knowledge of node.meta since it doesnt exist)

- a `ToV` service (pin or unpinn will get a Nod with
  - node = db_c
  - node = Non
  - node = [met extra]

Saving objects to the database (API/conductor > DB)

The version used for saving Iron icObjects to the the data is determined as follows:

- For an unpinn service the object is save in its latest version. Since objects are always in their latest version, no conversions are needed.
- For a pinn service the object is save in its pinn version. Since objects are always in their latest version, the object needs to be converted to the pinned version before being saved.

The

meth `Iro` `do_` handles this logi returning a dictionary of changed fields and their new values (similar to the existing `oslo.versionedobjects.VersionedObject.obj_get_changes()`). Since we do not keep track internally, of the database version of an object, the objects `version` field will always be part of these changes.

The *Roll upgrad process* (at step 3.1) ensure that by the time an object can be saved in its latest version, all services are running the newer release (although some may still be pinned) and can handle the latest object versions.

An interesting situation can occur whe the services are as described in step 3.1. It is possible for an IronicObject to be saved in a newer version and subsequently get saved in an older version. For example, a `ToVer` unpinned conductor might save a node in version 1.5. A subsequent request may cause a `ToVer` pinned conductor to replace and save the same node in version 1.4!

Sending objects via RPC (API/conductor -> RPC)

Whe a service mak an RPC reques any Iron icOb jects that are sent as part of that request are serialized into entities or primitives via `IronicObjectSerializer.serialize_entity()`. The version used for objects being serialized is as follows:

- For an unpinn service the object is serialized to its latest version. Since objects are always in their latest version, no conversions are needed.
- For a pinn service the object is serialized to its pinned version. Since objects are always in their latest version, the object is converted to the pinned version before being serialized. The converted object includes changes that resulted from the

conversion; this is needed so that the service at the other end of the RPC request has the necessary information if that object will be saved to the database.

Receiving objects via RPC (API/conductor <- RPC)

Whe a service receiv an RPC reques any entities that are part of the request need to be deserialized (via `oslo.versionedobjects.VersionedObjectSerializer.deserialize_entity()`). For entities that represent IronicObjects, we want the deserialization process (via `IronicObjectSerializer._process_object()`) to result in IronicObjects that are in their latest version, regardless of the version they were sent in and regardless of whether the receiving service is pinned or not. Again, any objects that are converted will retain the changes that resulted from the conversion, useful if that object is later saved to the database.

For example, a `Fro` iron api coul issue an `upd` RPC request with a node in version 1.4, where node.extra was changed (so node._changed_fields = [extra]). This node will be serialized in version 1.4. The receiving `ToVer` pinned ironic-conductor deserializes it and converts it to version 1.5. The resulting node will have node.meta set (to the changed value from node.extra in v1.4), node.extra = None, and node._changed_fields = [meta, extra].

When developing a new feature or modifying an IronicObject

Whe addi a new feature or char ing an Iron icOb ject, they need to be coded so that things work during a rolling upgrade.

The following describ areas whe the code may need to be changed, as well as some points to keep in mind when developing code.

ironic-api

During a rolli upgrad the new, pinn ironi api is talking to a new conductor that might also be pinned. There may also be old ironic-api services. So the new, pinned ironic-api service needs to act like it was the older service:

- New features shou not be mad avai able unless they are somehow totally supported in the old and new releases. Pinning the API version is in place to handle this.
  - If, for wha ever reason, the API version pinning does prevent a request from being handled that cannot or should not be handled, it should be coded so that the response has HTTP status code 406 (Not Acceptable). This is the same response to requests that have an incorrect (old) version specified.

Ironic RPC versions

Whe the signature (argumen of an RPC

meth is char

or new methods are added, the following needs to be considered:

- The RPC version mus be incremen and be the sam

value for both the client (`ironic/conductor/rpcapi.py`, used by ironic-api) and the server (`ironic/conductor/manager.py`, used by ironic-conductor). It should also be updated in `ironic/common/release_mappings.py`.

- Until ther is a major version bum new arguments of an RPC method can only be added as optional. Existing arguments cannot be removed or changed in incompatible ways with the method in older RPC versions.

- iron api (clie side sets a version cap (by pass ing the version cap to the constructor of oslo_messaging.RPCClient). This pinning is in place during a rolling upgrade when the `[DEFAULT]/pin_release_version` configuration option is set.

- New RPC methods are not available whe the service is pinned to the older release version. In this case, the corresponding REST API function should return a server error or implement alternative behaviours.
- Methods whic chan arguments shou run `cli` `can` to see if the version of the request is compatible with the version cap of the RPC Client. Otherwise the request needs to be created to work with a previous version that is supported.
- ironi cond (serv side) shou tolerate olde version of requests in order to keep working during the rolling upgrade process. The behaviour of ironic-conductor

will depend on the input parameters passed from the client-side.

- Old methods can be remov only after they are no longer used by a previous named release.

Object versions

Whe subclass of `irc` `obj` `bas` `Irc` are modified, the following needs to be considered:

- Any char of field or char in signature of remot methods needs a bump of the object version. The object versions are also maintained in `ironic/`

`common/release_mappings.py`.

- New objects mus be adde to `irc com rel py.` Also for the first releases they should be excluded from the version check by adding their class names to the `NEW_MODELS` list in `ironic/cmd/dbsync.py`.
- The argumen of remot methods (met ods whic are remoted to the conductor via RPC) can only be added as optional. They cannot be removed or changed in an incompatible way (to the previous release).
- Fiel type cannot be char Instea create a new field and deprecate the old one.
-

Ther is a unit test that generates the hash of an object using its fields and the signatures of its remotable methods. Objects that have a version bump need to be updated in the expected_object_fingerprints dictionary; otherwise this test will fail. A failed test can also indicate to the developer that their change(s) to an object require a version bump.

- Whe new version objects communicate with old version objects and when reading or writing to the database, `ironic.objects.base.IronicObject._convert_to_version()` will be called to convert objects to the target version. Objects should implement their own ._convert_to_version() to remove or alter fields which were added or changed after the target version:

(continues on next page)

```
↪              remove_unavailable_fields=True):
```

(continued from previous page)

```
↪of the
















↪version may be
```

(continues on next page)

(continued from previous page)

```
↪
↪two
↪unavailable
```

(continues on next page)

(continued from previous page)

```
↪set to True in this case.







↪unavailable fields to their







↪saved in the DB. (If
```

(continues on next page)

```
↪magic will not know to
```

(continued from previous page)

```
↪remove_unavailable_fields is set to
```

(continues on next page)

(continued from previous page)

```
↪unavailable fields to appropriate





↪for DB interactions.
```

This meth mus han- dle:

– con- vert- ing from an olde ver- sion to a new ver- sion

– con- vert-ing fron a new version to an olde version

– mak ing sure whe converting, that you take into consideration other object fields that may have been affected by a field (value) only available in a newer version. For example, if field new is only available in Node version 1.5 and Node.affected = Node.new+3, when converting to 1.4 (an older version), you may need to change the value of Node.affected too.

Online data migrations

The `irc` `onl` com man will perform online data migrations

Kee in min the

Policy for char to the DB mod Future incompatible changes in SQLAlchemy models, like removing or renaming columns and tables can break rolling upgrades (when ironic services are run with different release versions simultaneously). It is forbidden to remove these database resources when they may still be used by the previous named release.

Whe creating new Alen bic migrations whic modify existing models, make sure that any new columns default to NULL. Test the migration out on a non-empty database to make sure that any new constraints dont cause the database to be locked out for normal operations.

You can find an over on wha DDI operations may caus downtime in https://dev.mysql.com/doc/refman/5.7/en/innodb-create-index-overview.html. (You should also check older, widely deployed InnoDB versions for issues.) In the case of PostgreSQL, adding a foreign key may lock a whole table for writes.

Mak sure

to add a releas note if ther are any dow relat concerns.

Back-filling default values, and migrating data betwee columns or between tables must be implemented inside an online migration script. A script is a database API method (added to `ironic/db/api.py` and `ironic/db/sqlalchemy/api.py`) which takes two arguments:

- context: an admin context
- max this is used to limit the quer It is the

maximum number of objects to migrate; >= 0. If zero, all the objects will be migrated.

It returns a two-tuple

- the total number of objects that need to be migrated, at the start of the method, and
- the number of migrated objects.

In this method the version column can be used to select and update old objects.

The

meth nam shou be adde to the list of `ONL` in `irc` `cmd`

`dbsync.py`.

The meth shou be removed in the next nam release after

this one.

After online data migrations are com plete and the SQL

models no longer contain old fields, old columns can be removed from the database. This takes at least 3 releases, since we have to wait until the previous named release no longer contains references to the old schema. Before removing any resources from the database by modifying the schema, make sure that your implementation checks that all objects in the affected tables have been migrated. This check can be implemented using the version column.

ironic-dbsync upgrade command

The `irc` `upg` comman first chec that the versions of the objects are compatible with the (new) release of ironic, before it will make any DB schema changes. If one or more objects are not compatible, the upgrade will not be performed.

This chec is done by comparing the objects `ver` field in the database with the expected (or supported) versions of these objects. The supported versions are the versions specified in `ironic.common.release_mappings.RELEASE_MAPPING`. The newly created tables cannot pass this check and thus have to be excluded by adding their object class names (e.g. `Node`) to `ironic.cmd.dbsync.NEW_MODELS`.

Role Based Access Control - Testing

The Role Base Access control testing is a minor departure from the Ironic standard pattern of entirely python based unit testing. In part this was done for purposes of speed and to keep the declaration of the test context.

This also lend itself to be very useful due to the nature of A/B testing which is requried to properly migrate the Ironic project from a project scoped universe where an `admin project` is utilized as the authenticating factor coupled with two custom roles, `baremetal_admin`, and `baremetal_observer`.

As a contributor looking back after getting a over a thousand additional tests in place using this method, it definitely helped the speed at which these were created, and then ported to support additional.

How these tests work

These tests execute API calls thro

the API laye using the appropriate verb and header, which settings to prevent the `keystonemiddleware` from intercepting and replacing the headers were passing. Ultimately this is a feature, and it helps quite a bit.

The second aspect of how this worl is were moc ing the conductor RPC `get_topic_for` and `get_random_topic_for` methods. These calls raise Temporary Unavailable, since trying to execute the entire interaction into the conductor is moderately pointless because all policy enforement is located with-in the API layer.

At the sam time wiri everythin up to go fron API to conductor code and back would have been a heavier lift. As such, the tests largely look for one of the following error codes.

- 200 - Got the item from the

API
-

This
is

an database driven interaction.

- 201
  -

  Cre-
  ated
  -

  This
  is
  data
  driv
  in-
  ter-

action. These are rare.

- 204
  -

  Ac-
  cept
  -

  This
  is
  a
  data
  driv
  in-

teraction. These are rare.

- **403 - For**
  ac-
  cess
  was
  de-
  nied
- **404 - Not**
  Iron
  be-
  com
  scop
  awa
  thes

are generally only in the drivers API endpoints behavior. In System scope aware Project scoped configuration, i.e. later RBAC tests, this will become the dominant response for project scoped users as responding with a 403 if they could be an owner or lessee would provide insight into the existence of a node.

- **503 - Ser** whe a reques *has* been successfully authenticate and would have been sent along to the conductor.

How to make changes or review these tests?

The tests cycle thro the various endpoin and repeating patterns are clearly visible. Typically this means a given endpoint is cycled through with the same basic test using slightly different parameters such as different authentication parameters. When it comes to system scope aware tests supporting node `owners` and `lessee`, these tests will cycle a little more with slightly different attributes as the operation is not general against a shared common node, but different nodes.

Som tests

will test bod contents or attribu som will validate the number of records returned. This is important later with `owner` and `lessee` having slightly different views of the universe.

Som general rule apply

- Admins can do thin at leas as far as thei scop or rights apply. Remember: owner and lessee admins are closer to System scoped Admin Members.
- Members can do som thin but not everythin
- Rea

ers can al- way read but as we get into sensitive data later on such as fields containing infrastucture internal addresses, these values will become hidden and additional tests will examine this.

- Thir part or external/ Admins will find noth ing but sadness in empty lists, 403, 404, or even 500 errors.

What is/will be tested?

The idea is to in esse test as muc as possible, how ever as these tests Role Based Access Control related capabilities will come in a series of phases, styles vary a little.

The

first phas is `"le` In esse thes are partiall programatically generated and then human reviewed and values populated with expected values.

The second phas is remar ably similar to `leg` It is the safety net where we execute the `legacy` tests with the updated `oslo.policy` configuration to help enforce scopes. These tests will intentionally begin to fail in phase three.

The thir phas is the implemen tation of System scop awareness for the API. In this process, as various portions of the API are made system scope aware. The `legacy` tests are marked as `deprecated` which signals to the second phase test sequences that they are **expected** to fail. New `system scoped` tests are also implemented which are matched up by name to the `legacy` tests. The major difference being some header values, and a user with a `member` role in the `system` scope now has some rights.

The

forth phas is implemen taito of `own` and `les` awa proj

scoping. The testing approach is similar, however it is much more of a shotgun approach. We test what we know should work, and what know should not work, but we do not have redundant testing for each role as `admin` users are also `members`, and since the policy rules are designed around thresholds of access, it just made no sense to run the same test for admin and members, where member was the threshold. These thresholds will vary with the proposed default policy. The forth scope also tests a third party external admin as a negative test to ensure that we are also denying access to resources appropriately.

Thes page contain information for PTL cros proj liaiso

and core reviewers.

Releasing Ironic Projects

Sinc the respon sibility for releas will mov betwee people, we document that process here.

A full list of proj that ironi manages is available in the governance site.

Who is responsible for releases?

The current PTL is ultimate responsible for making sure code gets released. They may choose to delegate this responsibility to a liaison, which is documented in the cross-project liaison wiki.

Any one may submit a releas reques per the process below, but the PTL or liaison must +1 the request for it to be processed.

Release process

Release are managed by the Ope Stac release team The release process is documented in the Project Team Guide.

What do we have to release?

The iron proj has a number of deliverable under its governance. The ultimate source of truth for this is projects.yaml in the governance repository. These deliverables have varying release models, and these are defined in the deliverables YAML files in the releases repository.

In general, iron deliverable

follow the cycl with inter

release model.

Non-client libraries

The following deliverables are non-clien librar

- iron lib
- metalsmit
- sush

Client libraries

The following deliverables are clien librar

- pyth iron
- pyth iron insp clier
- sush cli

Normal release

The following deliver able are Neu tron plugins:

- netw bare
- netw gene swit

The following deliver able are Hori zon plug-ins:

- ironi ui

The following deliverables are Tem pest plug-ins:

- ironi temp plug

The following deliverables are services or treat as

such:

- bifro
- ironi
- ironi insp
- ironi

pron expo

- ironi pyth agen

Independent

The following deliverable are releas independently:

- ironi pyth agen buil
- molt iron
- sush tools
- tenk
- virtualbm

Not released

The following deliverable do not need to be released:

- ironi insp spec
- ironi spec

Things to do before releasing

- Review the unrelease release note if the proj uses them. Make sure they follow our *standards*, are coherent, and have proper grammar. Combine release notes if necessary (for example, a release note for a feature and another release note to add to that feature may be combined).
- For ironi release

only not ironi insp release if any new API microversions have been added since the last release, update the REST API version history (`doc/source/contributor/webapi-version-history.rst`) to indicate that they were part of the new release.

- To support rolli upgrad add this new release version (and release name if it is a named release) into `ironic/common/release_mappings.py`:
  - in `REL` mak a copy of the `mas` entry, and renam the first `master` entry to the new semver release version.
  - If this is a nam release

add a `REL` entry for the named release. Its value should be the same as that of the latest semver one (that you just added above).

It is important to do this before a stable/ bran is made (or if the grenade switch is made to use the latest release from stable as the old release). Otherwise, once it is made, CI (the grenade job that tests new-release -> master) will fail.

- Che for any ope patc that are clos to be merg or release critical.

  This usually includes important bug fixes and/

features that wed like to release, including the related documentation.

How to propose a release

The steps that lead to a release proposa are main manual, while proposing the release itself is almost a 100% automated process, accomplished by following the next steps:

- Clon the oper stacl repo i-tory. This is whe deliverables are tracked and all the automation resides.
  - Under the `del` directory you can see

yam files for each deliverable (i.e. subproject) grouped by release cycles.

– The `_in` directory contains yam files for deliverables that are not bound to (official) cycles (e.g. ironic-python-agent-builder).

- To chec the char were abou to re-leas we can use the tox environment `list-unreleased-changes`, with this syntax:

```
↪<deliverable>
```

The

ser

ar-
gu-
men
is
a
re-
lease
se-
ries
(i.e.
mas-
ter

or train, not stable/ussuri or stable/train).

For
ex-
am-
ple,
as-
sum
ing
were
in
the
mair
di-
rec-
tory

of the releases repository, to check the changes in the ussuri series for ironic-python-agent type:

```
↪agent
```

- To
up-
date
the
de-
liv-

erable file for the new release, we use a scripted process in the form of a tox environment called `new-release`.

To get familiar with it and see all the options tions type

Now base on the list of char we foun in the prec dent step and the release notes, we need to decide on whether the next version will be major, minor (feature) or patch (bugfix).

Note

that in this case `ser` is a code nam (trai ussuri) not a branch. That is also valid for the current development branch (master) that takes the code name of the future stable release, for example if the future stable release code name is wallaby, we need to use wallaby as `series`.

The `--s` `arg` is used only for bran ing in the end of a cycle, independent projects are not branched this way though.

The `--i` option is used to create an intermediate bugfix branch following the new release model for ironic projects.

To propose the

releas use the scrip to update the deliverable file, then commit the change, and propose it for review.

For example, to propose a minor releas for ironi in the master branch (current development branch), considering that the code name of the future stable release is wallaby, use:

Remen ber to use a mea ingful topi

usually using

the name of the deliverable, the new version and the branch, if applicable.

A gooc commit message title shou also includ the sam

for example Release ironic 1.2.3 for ussuri

- As an optiona step we can use `tox -e lis` to doub

check the changes before submitting them for review.

Also `tox -e val` (it migh take a whil to run base on the

number of changes) does some some sanity-checks, but since everything is scripted, there shouldnt be any issue.

All the scrip are design and main taine by the release team

in case of questions or doubts or if any errors should arise, you can reach to them in the IRC channel `#openstack-release`; all release liaisons should be present there.

- After the chan is up for review the PTL or a

release liaison will have to approve it before it can get approved by the release team. Then, it will be processed automatically by zuul.

Things to do after releasing

When a release is done that results in a stable branch

Whe a release is done that results in a

stable bran

for the project, several changes need to be made.

The releas automa tion will push a number of char that

need to be approved. This includes:

- In the new stable bran
  - a char to poin `.git` at the bran
  - a char to update the upper constrai file used

`tox`

by

- In the master bran
  - updating the release note RST to include the new

branch.

The generated RST does not include the version rang in

the title, so we typically submit a follow-up patch to do that. An example of this patch is here.

  - update the *templat* in *.zuu* or *zuul*

The

update is necessary to use the job for the next release *openstack-python3-<next_release>-jobs*. An example of this patch is here.

We need to submit patc for chan in the stable bran to:

- update the ironi devstack plugin to poin at the branched tarball for IPA. An example of this patch is here.
- update links in the doc-

umentation (`ir` `doc` `sou`

) to point to the branched versions of any openstack projects (that branch) documents. As of Pike release, the only outlier is diskimage-builder.

- set appropriate default for `TEM` and `TEM` in `dev`

`lib/ironic` to make sure that unsupported API tempest tests are skipped on stable branches. E.g. patch 495319.

We need to submit patc for char on master to:

- create an emp commit with a `Sem` tag to bum

the

generated minor version. See example and pbr documentation for details.

- to support rolli upgrad sinc the releas was a nam

release, we need to make these changes. Note that we need to wait until *after* the switch in grenade is made to test the latest release (N) with master (e.g. for stable/queens). Doing these changes sooner after the ironic release and before the switch when grenade is testing the prior release (N-1) with master, will cause the tests to fail. (You may want to ask/remind infra/qa team, as to when they will do this switch.)

  - In `irc` `com` `rel` `py`, dele any entries fron `REL` asso-

ciated with the oldest named release. Since we support upgrades between adjacent named releases, the master branch will only support upgrades from the most recent named release to master.

  - remov any DB migration scrip fron `irc` `cmd` `dbs`

`ONL`

and remove the corresponding code from ironic. (These migration scripts are used to migrate from an old release to this latest release; they shouldnt be needed after that.)

– remov any mod class nam from `irc` `cmd` `dbs` `NEW`

Ironic Tempest plugin

As **iron temp plug** is bran less, we need to submit a patc

adding stable jobs to its master branch. Example for Queens.

Bifrost

Bifr need to be update to install dependencies using the stable branch. Example for Victoria. The upper constraints file referenced in `scripts/install-deps.sh` needs to be updated to the new release.

For all releases

For all releas whe or not it results in a stable branch:

- update the spec repo to mark any spec complete in the release as implemented.
- remov any -2s on patc that were bloc until

after the release.

Ironic Governance Structure

The iron proj man ages a num ber of repo itories that contribute to our mission. The full list of repositories that ironic manages is available in the governance site.

What belongs in ironic governance?

For a repo itory to be part of the Iron proj

- It mus com ply with the TCs rules for a new proj

- It mus not be intend for use with only a single

vendors hardware. A library that implements a standard to manage hardware from multiple vendors (such as IPMI or redfish) is okay.

- It mus align with Ironics mission statemen

  Lack of contributor diversity is a chic egg prob

lem, and as such a repository where only a single company is contributing is okay, with the hope that other companies will contribute after joining the ironic project.

Repositories that are no

long main taine shou be prun from governance regularly.

Proposing a new project to ironic governance

Brin the proposa to the iron week meet ing to discuss with the team.

9.1. Dri

Iron ics com munity includ man hard ware vendors who contribute drivers that enable more advanced functionality when Ironic is used in conjunction with that hardware. To do this, the Ironic developer community is committed to standardizing on a Python Driver API that meets the common needs of all hardware vendors, and evolving this API without breaking backwards compatibility. However, it is sometimes necessary for driver authors to implement functionality - and expose it through the REST API - that can not be done through any existing API.

To facilitate that, we also provide the mea for API calls to be passed through ironic and directly to the driver. Some guidelines on how to implement this are provided below. Driver authors are strongly encouraged to talk with the developer community about any implementation using this functionality.

Pluggable Drivers

Iron support a pluggabl driv mod This allows contributors to easily add new drivers, and operators to use third-party drivers or write their own. A driver is built at runtime from a *hardware type* and *hardware interfaces*. See *Enabling drivers and hardware types* for a detailed explanation of these concepts.

Har ware type and interface are load by the `iro` service

during initialization from the setuptools entrypoints `ironic.hardware.types` and `ironic.hardware.interfaces.<INTERFACE>` where `<INTERFACE>` is an interface type (for example, `deploy`). Only hardware types listed in the configuration option `enabled_hardware_types` and interfaces listed in configuration options `enabled_<INTERFACE>_interfaces` are loaded. A complete list of hardware types available on the system may be found by enumerating this entrypoint by running the following python script:

```
↪#
↪/
↪u
↪b
↪e
↪p

imp
↪p
↪a
↪p
pri
↪[
↪n
↪f
↪p
↪i
↪p
↪i
↪e
↪p
↪"
↪h
↪t
↪"
↪if not p.name.startswith("fake")]
```

A list of drive enable in a running Iron service may be found by issuing the following command against that API end point:

```
bar
↪d
↪l
```

Writing a hardware type

A hardware type is a Pyth class inheriting *iro dri* *hardware_type.AbstractHardwareType* and listed in the setuptools entry point `ironic.hardware.types`. Most of the real world hardware types inherit *ironic.drivers.generic.GenericHardware* instead. This helper class provides useful implementations for interfaces that are usually the same for all hardware types, such as `deploy`.

The minimum requir interface are:

- *boot* that specifies how to boot ram and instan on the hardware. A generic `pxe` implementation is provided by the `GenericHardware` base class.
- *deploy* that

orches trate the deploy men A few common implementations are provided by the `GenericHardware` base class.

As of the Rock release a deploy interface shou decorate its deploy method to indicate that it is a deploy step. Conventionally, the deploy method uses a priority of 100.

Note Mos of the hardware type shou not override this interface.

- *pow* implemen pow actions for the hard ware Thes common implementations may be used, if supported by the hardware:
  - *irc dri moc ipm IPM*
  - *irc dri moc rec pow Rec*

  Otherwise you need to write your own implementation by subclassing *ironic.drivers.base.PowerInterface* and providing missing methods.

Not Pow

actions in Iron are bloc ing - methods of a power interface should not return until the power action is finished or errors out.

- *man age- men* im- ple- men ad- di- tiona out- of- banc man agement actions, such as setting a boot device. A few common implementations exist and may be used, if supported by the hardware:
  - `iro dri moc ipm IPM`
  - `iro dri moc rec man Rec`

  Som hard ware type such as

snm do not support out-of-banc management. They use the fake implementation in *`ironic.drivers.modules.fake.FakeManagement`* instead.

Otherwise you need to write your own implementation by subclassing *`ironic.drivers.base.ManagementInterface`* and providing missing methods.

Combine the interfaces in a hardware type by populating the lists of supported interfaces. These lists are prioritized, with the most preferred implementation first. For example:

```
cla
 ↪M
 ↪G


␣
 ↪␣
 ↪␣
 ↪␣
 ↪@
```

(continues on next page)

(continued from previous page)

```
↪management interfaces."""
```

```
␣
↪␣
↪␣
↪␣
↪d
↪s
↪m
↪i
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪"
↪"
↪o
↪s
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪r
↪[
↪␣
↪i
↪I
␣
↪␣
↪␣
↪␣
↪@
␣
↪␣
↪␣
↪␣
↪d
↪s
↪p
↪i
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪"
↪"
↪o
```

(continues on next page)

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪r
↪[
↪␣
↪i
↪I
```

Note: In this example, all interfaces, except for `management` and `power` are taken from the `GenericHardware` base class.

Finally give the new hardware type and new interface hum

friendly names and create entry points for them in the `setup.cfg` file:

```
iro
↪h
↪t
↪=
␣
↪␣
↪␣
↪␣
↪m
↪h
↪=
↪i
↪d
↪m
↪h
iro
↪h
↪i
↪p
↪=
```

(continues on next page)

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪m
↪p
↪=
↪i
↪d
↪m
↪m
↪h
iro
↪h
↪i
↪m
↪=
␣
↪␣
↪␣
↪␣
↪m
↪m
↪=
↪i
↪d
↪m
↪m
↪h
```

Supported Drivers

For a list of sup-port driv (tho that are con-tin-u-ousl tested on every upstream commit) please consult the *drivers page*.

Node Vendor Passthru

Driv may implemen a pass API whic is accessible via the `/v1/nodes/<Node UUID or Name>/vendor_passthru?method={METHOD}` endpoint. Beyond basic checking, Ironic does not introspect the message body and simply passes it through to the relevant driver.

A meth

- can support one or more HTT methods (for example, GET, POST)
- is asyn chro or synchro
  - For asyn chro methods,

a 202 (Accept HTT status code

is returned to indicate that the request was received, accepted and is being acted upon. No body is returned in the response.

– For synchro methods, a 200 (OK HTT status code is

returned to indicate that the request was fulfilled. The response may include a body.

- can requir an exclusive lock on the node This only

occurs if the method doesnt specify require_exclusive_lock=False in the decorator. If an exclusive lock is held on the node, other requests for the node will be delayed and may fail with an HTTP 409 (Conflict) error code.

This endpoin expose a node driv

directl and as such it

is expressly not part of Ironics standard REST API. There is only a single HTTP endpoint exposed, and the semantics of the message body are determined solely by the driver. Ironic makes no guarantees about backwards compatibility; this is solely up to the discretion of each drivers author.

To get information abou all the methods avai able via

the vendor_passthru endpoint for a particular node, you can issue an HTTP GET request:

```
GET
↪/
↪v
↪n
↪
↪<
↪U
↪o
↪n
↪/
↪v
↪p
↪m
```

The respor JSO body will contain information for each method, such as the methods name, a description, the HTTP methods supported, and whether its asynchronous or synchronous.

Driver Vendor Passthru

Driv may implemen an API for reques not relate to any node, at `/v1/drivers/<driver name>/vendor_passthru?method={METHOD}`.

A meth

- can support one or more HTT methods (for example, GET, POST)
- is asyn chro or synchro
  - For asyn chro

methods, a 202 (Accept HTT status code is returned to indicate that the request was received, accepted and is being acted upon. No body is returned in the response.

- For synchro methods, a 200 (OK HTT status code is returned to indicate that the request was fulfilled. The response may include a body.

Note: Unlike methods in *Node Vendor Passthru*, a request does not lock any resource, so it will not delay other requests and will not fail with an HTTP 409 (Conflict) error code.

Iron mak no guarantees abou the semantics of the message BODY sent to this endpoint. That is left up to each drivers author.

To get information abou all the meth ods avai able via

the driver vendor_passthru endpoint, you can issue an HTTP GET request:

```
GET
↪/
↪v
↪d
↪
↪<
↪n
↪/
↪v
↪p
↪m
```

The response JSO body will contain information for each

method, such as the methods name, a description, the HTTP methods supported, and whether its asynchronous or synchronous.

Vendor Methods

This document is a quic tutorial on writing vendor specific methods to a driver.

The first thin to note is that the Iron API support two vendor endpoints: A driver vendor passthru and a node vendor passthru.

- The `Ven` allows hardware type to expose a custom top-level functionality which is not specific to a Node. For example, lets say the driver *ipmi* exposed a method called *authentication_types* that would return what are the authentication types supported. It could be accessed via the Ironic API like:

(continues on next page)

(continued from previous page)

for the default `vendor` interface implementation for a given hardware type. This limitation will be lifted in the future.

- The node vendor pass allows driv to expose cus tom

functionality on per-node basis. For example the same driver *ipmi* exposing a method called *send_raw* that would send raw bytes to the BMC, the method also receives a parameter called *raw_bytes* which the value would be the bytes to be sent. It could be accessed via the Ironic API like:

(continues on next page)

```
↪<address>:<port>/v1/nodes/<node UUID>/vendor_passthru/send_raw
```

(continued from previous page)

Writing Vendor Methods

Writ ing a custom vendor meth in Iron shou be simple.

The first thing to do is write a class inheriting from the VendorInterface class:

```
cla
 ↪E

␣
 ↪␣
 ↪␣
 ↪␣
 ↪d
 ↪g
 ↪p
␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪r
 ↪
 ↪{
 ↪

␣
 ↪␣
 ↪␣
 ↪␣
 ↪d
 ↪v
 ↪␣
 ↪t
 ↪␣
 ↪*
```

(continues on next page)

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪p
```

The *get_* is a meth that all driv interface have it shou

return a dictionary of <property>:<description> telling in the description whether that property is required or optional so the node can be manageable by that driver. For example, a required property for a *ipmi* driver would be *ipmi_address* which is the IP address or hostname of the node. We are returning an empty dictionary in our example to make it simpler.

The *validate* meth is responsible for validating the parameters passed to the vendor methods. Ironic will not introspect into what is passed to the drivers, its up to the developers writing the vendor method to validate that data.

Lets extend the *Exam-*

pleV *dor* class to support two methods, the *authentication_types* which will be exposed on the driver vendor passthru endpoint; And the *send_raw* method that will be exposed on the node vendor passthru endpoint:

```
cla
↪E

␣
↪␣
↪␣
↪␣
↪d
↪g
↪p
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪r
↪
↪{
↪


␣
↪␣
↪␣
↪␣
↪d
↪v
↪␣
↪t
↪␣
↪m
↪␣
↪*
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪i
↪m
↪=
↪
↪'
↪r
↪':
```

(continues on next page)

(continued from previous page)

```
↪'raw_bytes' not in kwargs:

↪  raise MissingParameterValue()
```

(continues on next page)

(continued from previous page)

```
↪bytes')
```

```
␣
↪␣
↪␣
↪␣
↪@
↪p
↪'
↪'
␣
↪␣
↪␣
↪␣
↪d
↪s
↪r
↪␣
↪t
↪␣
↪*
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪r
↪b
↪=
↪k
↪g
↪'

␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪.
↪.
↪.
↪
```

That
it!

Writ
ing
a
node
or

driv vendor pass meth is prett muc the same, the only difference is how you decorate the methods and the first parameter of the method (ignoring self). A method decorated with the *@passthru* decorator should expect a Task object as first parameter and a method decorated with the *@driver_passthru* decorator should expect a Context object as first parameter.

Both decorators accept thes parameters:

- http_ A list of wha the HTT methods supporte by that vendor function. To know what HTTP method that function was invoked with, a *http_method* parameter will be present in the *kwargs*. Supported HTTP methods are *POST*, *PUT*, *GET* and *PATCH*.
- meth By default the meth nam is

the nam of the pyth function, if you want to use a different name this parameter is where this name can be set. For example:

- descrip tion: A strin containing a nice descrip tion about what that method is supposed to do. Defaults to (empty string).
- asyn A bool valu to determin

whe
this
meth
shou
run

asynchronously or synchronously. Defaults to True (Asynchronously).

Not

This parameter was previousl calle asyn

The node vendor pass decorator (@*p*
also accept the

following parameter:

- require A bool valu determining whe this meth shou

require an exclusive lock on a node between validate() and the beginning of method execution. For synchronous methods, the lock on the node would also be kept for the duration of method execution. Defaults to True.

Wa

Plea avoi having a synchro meth for slow runn operations **or** if the method does talk to a BMC; BMCs are flaky and very easy to break.

Wa

Each asyn chro reques consum a work threa in the `irc` process. This can lead to starvation of the thread pool, resulting in a denial of service.

Give the new vendor interface implementation a human-friendly name and create an entry point for it in the `setup.cfg`:

```
iro
↪h
↪i
↪v
↪=
␣
↪␣
↪␣
↪␣
↪e
↪=
↪i
↪d
↪m
↪e
```

Finally add it to the list of supported vendor interfaces for relevant hardware types, for example:

```
cla
↪E
↪G
␣
↪␣
↪␣
↪␣
↪.
↪.
↪.
↪
␣
↪␣
↪␣
↪␣
↪@
␣
↪␣
↪␣
↪␣
↪d
↪s
↪v
↪i
```

(continues on next page)

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪r
↪[
↪E
```

Backwards Compatibility

Ther is no requir men that char to a vendor meth be backwards compatible. However, for your users sakes, we highly recommend that you do so.

If you are char ing the exception being raise you migh want to ensure that the same HTTP code is being returned to the user.

For non- back com patibility, plea mak sure you add a release note that indicates this.

Developing BIOS Interface

To support a driv specific BIO interface it is necessary to create a class inheriting from the `BIOSInterface` class:

```
fro
↪i
↪i
↪b

cla
↪E
↪B


␣
↪␣
↪␣
↪␣
↪d
↪g
↪p
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪r
↪
↪{
↪
```

(continues on next page)

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪d
↪v
↪␣
↪t

␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪p
```

See *Plug gabl Driv* for a detaile explanation of hard

ware type and interface.

The `get` and `val` are methods that all driv interface have

The hardware interface that supports BIOS settings should also implement the following three methods:

- Im-

ple-
men
a
meth
nam
cac
This
meth
store
BIO
set-
ting

to the `bios_settings` table during cleaning operations and updates the `bios_settings` table when `apply_configuration` or `factory_reset` are successfully called.

(continues on next page)

(continued from previous page)

```
↪DriverLoadError(
↪   driver=self.__class__.__name__,
↪   reason=_("Unable to import driver library"))
```

(continues on next page)

(continued from previous page)

```
↪driver_info(task.node)
```

```
↪settings(node_info)
```

(continues on next page)

```
↪delete_list, nochange_list = (
```

(continued from previous page)

```
↪sync_node_setting(settings))

↪create(

↪    task.context, node_id, create_list)
```

(continues on next page)

(continued from previous page)

```
↪save(
↪  task.context, node_id, update_list)
```

(continues on next page)

(continued from previous page)

```
↪= []
↪in delete_list:
↪   delete_names.append(setting.name)
```

(continues on next page)

```
↪delete(
```

(continued from previous page)

```
↪  task.context, node_id, delete_names)
```

Not dri cli is vendor specific library to control and

manage the bare metal hardware, for example: python-dracclient, sushy.

- Implement a meth nam fac This meth need to use the

`clean_step` decorator. It resets BIOS settings to factory default on the given node. It calls `cache_bios_settings` automatically to update existing `bios_settings` table once successfully executed.

```
↪driver_info(task.node)
```

- Implemen a meth

nam
app
This
meth
need
to
use
the

clean_step decorator. It takes the given BIOS settings and applies them on the node. It also calls `cache_bios_settings` automatically to update existing `bios_settings` table after successfully applying given settings on the node.

```
↪':  (
```

(continues on next page)

(continued from previous page)

```
↪': True
```

(continues on next page)

(continued from previous page)

```
↪driver_info(task.node)

↪ settings)
```

The `set` parameter is a list of BIO settings to be configured. for example:

```
```

(continues on next page)

(continued from previous page)

```

↪

```

(continues on next page)

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

```
```

Third Party Continuous Integration

Note: This document is a work-in-progress. Unfilled sections will be worked in follow-up patchsets. This version is to get a basic outline and index done so that we can then build on it. (krtaylor)

This documen provide tips and guid lines for third party driv developers setting up their continuous integration test systems.

CI Architecture Overview

Requirements Cookbook

Sizing

Infrastructure

This section describ wha char youl need to mak to a your CI system to add an ironic job.

jenkins changes

nodepool changes

neutron changes

pre-test hook

cleanup hook

Ironic

Hardware Pool Management

Problem

If you are using actual hardware as target machin

for your CI testing then the problem of two jobs trying to use the name target arises. If you have one target machine and a maximum number of one jobs running on your ironic pipeline at a time, then you wont run into this problem. However, one target may not handle the load of ironics daily patch submissions.

Solutions

Zuul v3

Molten Iron

molt iron is a tool that

allows you to reserv hard ware from a pool at the last minute to use in your job. Once finished testing, you can unreserve the hardware making it available for the next test job.

Tips and Tricks

Optimize Run Time

Image Server

Other References

Developing a new Deploy Step

To support customi deploy men step implemen a new method in an interface class and use the decorator `deploy_step` defined in `ironic/drivers/base.py`. For example, we will implement a `do_nothing` deploy step in the `AgentDeploy` class.

```
cla
↪A
↪␣
↪b
↪D
␣
↪␣
↪␣
↪␣
↪.
↪.
↪.
↪
```

(continues on next page)

(continued from previous page)

```
↪': (

↪  "This is a test argument."
```

```
␣
↪␣
↪␣
↪␣
↪@
↪d
↪s
↪␣
↪a
↪{
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪'
↪a
↪'
↪
↪{
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪'
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪,
```

(continues on next page)

(continued from previous page)

```
↪': True
```

```
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪'

␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪}
↪

␣
↪␣
↪␣
↪␣
↪}
↪)

␣
↪␣
↪␣
↪␣
↪d
↪d
↪n
↪␣
↪t
↪␣
↪*

␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪r
↪N
```

After deploy men of the bare node chec the update de-

ploy steps:

```
bar
↪n
↪s
↪
↪$
↪i
↪-
↪f
↪j
↪-
↪c
↪d
↪i
↪i
```

The abov com man outputs the dri as following:

```
↪{
␣
↪␣
↪
↪"
↪i
↪i
↪"
↪
↪{
␣
↪␣
↪␣
↪␣
↪.
↪.
↪.
↪
```

(continues on next page)

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪
↪"
↪s
↪"
↪[
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪{
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪"
↪2
↪
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪"
↪
↪"
↪"
↪
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪"
↪
↪"
↪n
↪",
```

(continues on next page)

(continued from previous page)

```
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪"
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪{
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪
↪"
↪arg":
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪{
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪,
```

(continues on next page)

(continued from previous page)

```
↪  "description": "This is a test argument."

↪}
```

(continues on next page)

(continued from previous page)

```
↪
↪
```

(continues on next page)

(continued from previous page)

```
␣
 ↪␣
 ↪␣
 ↪␣
 ↪]
 ↪
␣
 ↪␣
 ↪␣
 ↪␣
 ↪
 ↪"
 ↪s
 ↪i
 ↪"
 ↪1
␣
 ↪␣
 ↪}
 ↪
}
 ↪
```

Note: Similarly, clean steps can be implemented using the `clean_step` decorator.

In-banc deploy step (deploy step that are run inside the ramdisk) have to be implemented in a custom IPA hardware manager. All in-band deploy steps must have priorities between 41 and 99, see *Agent steps* for details.

9.1. Net wor Integra tio

In order to test the integration betwee the Bare Metal and Networking services, support has been added to devstack to mimic an external physical switch. Here we include a recommended configuration for devstack to bring up this environment.

Ironic multitenant networking and DevStack

This guid will walk you thro using Ope Stac Iron with the ML2 `networking-generic-switch` plugin.

Using VMs as baremetal servers

This scenari show how to setu Devstack to use Iron integration with VMs as baremetal servers and ML2 `networking-generic-switch` that interacts with OVS.

DevStack Configuration

The following is `loc` `con` that will setu Devstack with 3 VMs that are registered in ironic. `networking-generic-switch` driver will be installed and configured in Neutron.

```
[[l
↪#
↪C
↪i
↪f
↪i
↪d
↪p
↪
ena
↪p
↪i
↪h
↪/
↪o
↪o
↪o
↪i
```

(continues on next page)

(continued from previous page)

```
↪#
↪I
↪n
↪g
↪s
↪N
↪M
↪d
↪t
↪i
↪w
↪O
ena
↪p
↪n
↪g
↪s
↪h
↪/
↪o
↪o
↪o
↪n
↪g
↪s


↪#
↪A
↪l
↪l
↪i
↪w
↪r
↪I
↪n
IRO
↪U
↪L
↪L

IRO
↪E
↪N
↪I
↪n
IRO
↪N
↪I


↪#
↪c
OVS
↪P
↪B
```

(continues on next page)

(continued from previous page)

```
PHY
 ↪N
IRO
 ↪P
 ↪N
 ↪N
 ↪p
IRO
 ↪P
 ↪S
 ↪P
 ↪0
 ↪5
 ↪0
 ↪2
IRO
 ↪P
 ↪S
 ↪G
 ↪0
 ↪5
 ↪1

Q_
 ↪P
ENA
 ↪T
 ↪V
Q_
 ↪M
 ↪T
 ↪N
 ↪T
TEN
 ↪V
 ↪R


 ↪#
 ↪C
ADM
 ↪P
RAB
 ↪P
DAT
 ↪P
SER
 ↪P
SER
 ↪T
SWI
 ↪H
SWI
 ↪T
 ↪K
```

(continues on next page)

(continued from previous page)

```
↪#
↪E
↪I
↪A
↪a
↪I
↪C
ena
↪s
↪i
ena
↪s
↪i
↪a
ena
↪s
↪i
↪c


↪#
↪D
↪n
↪n
↪s
↪␣
↪i
↪d
↪n
↪s
↪i
↪a
↪
dis
↪s
↪n
↪n


↪#
↪E
↪S
↪f
↪t
↪d
↪d
↪i
↪
ena
↪s
↪s
↪p
ena
↪s
↪s
↪o
```

(continues on next page)

(continued from previous page)

```
ena
↪s
↪s
↪c
ena
↪s
↪s
↪a


↪#
↪D
↪H
dis
↪s
↪h


↪#
↪D
↪C
dis
↪s
↪c
↪c
↪s
↪c
↪a
↪c
↪v


↪#
↪D
↪T
dis
↪s
↪t


↪#
↪S
↪t
↪U
↪'
↪a
↪r
↪f
↪t
↪d
↪d
↪i
↪
SWI
↪E
↪T
```

(continues on next page)

(continued from previous page)

```
↪#
↪C
↪3
↪v
↪m
↪t
↪p
↪a
↪I
↪'
↪b
↪n
↪
IRO
↪V
↪C
IRO
↪B
↪B
↪O

↪#
↪E
↪a
↪h
↪t
↪␣
↪i
↪n
↪

↪#
↪E
↪H
↪T
↪f
↪h

↪#
↪D
↪'
↪f
↪t
↪m
↪h
↪t
↪r
↪e
↪o
↪a

↪#
↪i
↪␣
↪m
↪o
↪p
↪a
↪m
```

(continues on next page)

(continued from previous page)

```
↪#
↪E
↪M
↪I
↪f

↪#
↪E
↪P
↪I
↪f

↪#
↪T
↪d
↪d
↪i
↪i
↪
↪'
↪'
↪␣
↪y
↪c
↪u
↪'direct' with

↪#
↪D
↪D
↪I

↪#
↪C
↪t
↪t
↪a
↪t
↪d
↪d
↪f
↪n
↪c
↪b
↪d
↪

↪#
↪T
↪d
↪s
↪b
↪i
↪t
↪e
↪l
↪a
↪
```

(continues on next page)

(continued from previous page)

```
↪ephemeral partition.
```

```
IRO
↪D
↪D


↪#
↪T
↪p
↪b
↪r
↪t
↪m
↪p
↪v
↪t
↪c

↪#
↪f
↪n
↪
IRO
↪V
↪S
↪R
IRO
↪V
↪S
↪D


↪#
↪S
↪o
↪t
↪e
↪p
↪i
↪G
↪␣
↪U
↪0
↪f
↪n

IRO
↪V
↪E
↪D


↪#
↪T
↪b
↪y
↪o
↪I
↪r
↪f
↪s
↪␣
↪s
↪t
```

(continues on next page)

(continued from previous page)

```
IRO
↪B
↪D
↪R

VIR
↪D


↪#
↪B
↪d
↪␣
↪D
↪c
↪a
↪1
↪0
↪0
↪0
↪2
↪n
↪for instances.


↪#
↪I
↪t
↪o
↪w
↪t
↪h
↪n
↪␣
↪y
↪m
↪a
↪w
↪the


↪#
↪f
↪
NET
↪G
↪1
↪0
↪1
FIX
↪R
↪1
↪0
↪0
↪2
FIX
↪N
↪S
```

(continues on next page)

(continued from previous page)

```
↪#
↪L
↪a
↪o
↪t
↪f
LOG
↪$
↪d
↪l
LOG
↪$
↪l
IRO
↪V
↪L
↪D
↪$
↪i
↪b
↪l
```

9.1. Boo fron Vol

Star ing with the Pike releas it is also possible to use DevStack for testing booting from Cinder volumes with VMs.

Ironic Boot-from-Volume with DevStack

This guid show how to setup DevStack for enablin boot from volume feature, which has been supported from the Pike release.

This scenario show how to setup DevStack to enable node to boot from volumes managed by cinder with VMs as baremetal servers.

DevStack Configuration

The following is `loc` `con` that will setup DevStack with 3 VMs that are registered in ironic. A volume connector with IQN is created for each node. These connectors can be used to connect volumes created by cinder. The detailed description for DevStack is at *Deploying Ironic with DevStack*.

```
[[l

ena
 ↪p
 ↪i
 ↪h
 ↪/
 ↪o
 ↪o
 ↪o
 ↪i

IRO
 ↪S
 ↪I


 ↪#
 ↪C
ADM
 ↪P
DAT
 ↪P
RAB
 ↪P
SER
 ↪P
SER
 ↪T
SWI
 ↪H
SWI
 ↪T
 ↪K


 ↪#
 ↪E
 ↪N
 ↪w
 ↪i
 ↪r
 ↪b
 ↪I
 ↪a
 ↪d
 ↪n
 ↪n
 ↪
dis
 ↪s
 ↪n
 ↪n
dis
 ↪s
 ↪n
 ↪n
```

(continues on next page)

(continued from previous page)

```
ena
↪s
↪q
↪s
ena
↪s
↪q
↪a
ena
↪s
↪q
↪d
ena
↪s
↪q
↪l
ena
↪s
↪q
↪m
ena
↪s
↪n


↪#
↪E
↪S
↪f
↪t
↪d
↪d
↪i
↪
ena
↪s
↪s
↪p
ena
↪s
↪s
↪o
ena
↪s
↪s
↪c
ena
↪s
↪s
↪a


↪#
↪D
↪H
dis
↪s
↪h
```

(continues on next page)

(continued from previous page)

```
↪#
↪D
↪H
dis
↪s
↪h
↪h
↪a
↪h
↪a
↪c
↪h
↪a
↪c
↪h
↪e


↪#
↪S
↪t
↪U
↪'
↪a
↪r
↪f
↪t
↪d
↪d
↪i
↪
SWI
↪E
↪T


↪#
↪C
↪3
↪v
↪m
↪t
↪p
↪a
↪I
↪'
↪b
↪n
↪
IRO
↪V
↪C
IRO
↪B
↪B
↪O
```

(continues on next page)

(continued from previous page)

```
DEF
↪I
↪T


↪#
↪E
↪a
↪h
↪t
↪␣
↪i
↪n
↪

↪#
↪E
↪H
↪T
↪f
↪h

↪#
↪D
↪'
↪f
↪t
↪m
↪h
↪t
↪r
↪e
↪o
↪a

↪#
↪i
↪␣
↪m
↪o
↪p
↪a
↪m

↪#
↪E
↪M
↪I
↪f

↪#
↪E
↪P
↪I
↪f

↪#
↪T
↪d
↪d
↪i
↪i
↪
```

(continues on next page)

(continued from previous page)

```
↪#
↪D
↪D
↪I

↪#
↪C
↪t
↪t
↪a
↪t
↪d
↪d
↪f
↪n
↪c
↪b
↪d
↪

↪#
↪T
↪d
↪s
↪b
↪i
↪t
↪e
↪l
↪a
↪
IRO
↪D
↪D

↪#
↪T
↪p
↪b
↪r
↪t
↪m
↪p
↪v
↪t
↪c

↪#
↪f
↪n
↪
IRO
↪V
↪S
↪R
```

(continues on next page)

(continued from previous page)

```
↪ephemeral partition.
```

```
↪True
```

(continues on next page)

```
IRO
 ↪V
 ↪S
 ↪D

 ↪#
 ↪S
 ↪o
 ↪t
 ↪e
 ↪p
 ↪i
 ↪G
 ↪␣
 ↪U
 ↪0
 ↪f
 ↪n

IRO
 ↪V
 ↪E
 ↪D

 ↪#
 ↪T
 ↪b
 ↪y
 ↪o
 ↪I
 ↪r
 ↪f
 ↪s
 ↪␣
 ↪s
 ↪t
 ↪t

IRO
 ↪B
 ↪D
 ↪R

VIR
 ↪D

 ↪#
 ↪B
 ↪d
 ↪␣
 ↪D
 ↪c
 ↪a
 ↪1
 ↪0
 ↪0
 ↪0
 ↪2
 ↪n
```

```
↪for instances.
```

(continued from previous page)

```
↪#
↪I
↪t
↪o
↪w
↪t
↪h
↪n
↪␣
↪y
↪m
↪a
↪w
↪the

↪#
↪f
↪
NET
↪G
↪1
↪0
↪1
FIX
↪R
↪1
↪0
↪0
↪2
FIX
↪N
↪S


↪#
↪L
↪a
↪o
↪t
↪f
LOG
↪$
↪d
↪l
LOG
↪$
↪l
IRO
↪V
↪L
↪D
↪$
↪i
↪b
↪l
```

After the environmen is buil you can create a

volume with cinder and request an instance with the volume to nova:

```
.
↪␣
↪~
↪/
↪d
↪o


↪#
↪q
↪t
↪i
↪i
↪o
↪t
↪d
↪c
↪i
ima
↪$
↪i
↪s
↪
↪$
↪I
↪N
↪-
↪f
↪v
↪-
↪c
↪i


↪#
↪c
↪k
ssh
↪k
```

(continues on next page)

(continued from previous page)

```
↪default

↪volume -f value -c id)

↪key-name default testing
```

```
ope
↪k
↪c
↪-
↪-
↪p
↪k
↪~
↪/
↪.
↪s
↪i
↪r
↪p

↪#
↪c
↪v
vol
↪$
↪v
↪c
↪-
↪-
↪i
↪
↪$
↪-
↪-
↪s
↪1
↪m

↪#
↪s
↪i
ope
↪s
↪c
↪-
↪-
↪f
↪b
↪-
↪-
↪v
↪
↪$
↪-
↪-
```

You can also run an integration test that an instan

is booted from a remote volume with tempest in the environment:

```
cd␣
↪/
↪o
↪s
↪t
tox
↪-
↪e
↪a
↪p
↪-
↪-
↪␣
↪i
↪t
↪p
↪t
↪s
↪t
↪baremetal_boot_from_volume
```

Plea note that the storage interface will only indicate errors based upon the state of the node and the configuration present. As such a node does not exclusively have to boot via a remote volume, and as such *validate* actions upon nodes may be slightly misleading. If an appropriate *volume target* is defined, no error should be returned for the boot interface.

9.1. Iror Ser Pyt API Refer- enc

ironic

ironic package

Subpackages

ironic.api package

Subpackages

ironic.api.controllers package

Subpackages

ironic.api.controllers.v1 package

Submodules

ironic.api.controllers.v1.allocation module

class i

Base `pec` `res` `Res`

RES controll for allocations

delete

Dele an

allocation.

Parame

all UUI or logical nam of an allocation.

get_all

Retriev a list of allocations

Parame

- **nod** UUI or nam of a nod to get only allocations for that node.

- **res** Filter by requeste resource class

- **sta** Filter by allocation state

- **mar** pagination mark for large data sets.

- **lim** maximum number of resource to return in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** column to

sort re- sults by. De- fault id.

- **sor** di- rec- tion to sort. asc or desc De- fault asc.
- **fie** Op- tiona a list with a spec i- fied set of field of the resource to be returned.
- **own** Fil- ter by own

get_one
Re- triev in- for- ma- tion abou

the give al- lo- ca- tion.

Parame

- **all** UUI or log- i- cal nam of an al- lo- ca- tion.
- **fie** Op- tion a list with a spec i- fied set of field

of the resource to be returned.

invalic

patch(*a*

Up- date an ex- ist- ing al- lo- ca-

tion.

Parame

- **all** UUI or logical nam of an allocation.
- **pat** a json PAT document to apply to this allocation.

post (*al*

Create a new allocation.

Parame

all an allocation with

the
re-
ques
body

class i

Base
pec
res
Res

RES
con-
troll
for
al-
lo-
ca-
tions

delete

get_all

invalic

ironic.

ironic.

ironic.

ironic.

ironic.api.controllers.v1.bios module

class i

Base
pec
res
Res

REST con-troll for bios

get_all
List node bios set-tings

get_one
Re-triev in-for-ma-tion abou the give bios set-ting.

Parame
set
Log-i-cal nam of the set-ting to re-triev

ironic.

`ironic.`

Buil a dict containing a bios setting valu

ironic.api.controllers.v1.chassis module

class `i`

Base `pec` `res` `Res`

RES controll for Cha sis.

delete

Dele a chas sis.

Parame

cha

UUI of a chas sis.

detail

Retriev a list of chas sis with

detail.

Parame

- **mar** pagination mark for large data sets.
- **lim** maximum number of resour to return in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** column to sort results by. Default id.
- **sor** direction to

sort. asc or desc De- fault asc.

get_al

Re- triev a list of chas sis.

Parame

- **mar** pag- i- na- tion mark for larg data sets.
- **lim** max i- mun num ber of re- sour to re- turn in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** col- umn

to sort results by. Default: id.

- **sor** direction to sort. asc or desc Default: asc.
- **fie** Optiona a list with a specified set of field

of the resource to be returned.

get_one

Retriev information about the given chassis.

Paramet

- **cha** UUI of a chassis.
- **fie** Optiona a list with a specified set of field

of the resource to be returned.

invalic

nodes =

Exposed node as a sub-elem of chassis

patch (*c*

Updated an existing chassis.

Paramet

-

cha
UUI
of
a
chas
sis.

-

pat
a
json
PAT
doc-
u-
men
to
ap-
ply
to
this
chas

sis.

post (*ch*
Cre-
ate
a
new
chas-
sis.

Paramet
cha
a
chas-
sis
with
the
re-
ques
body

`ironic.`

`ironic.`

ironic.api.controllers.v1.collection module

`ironic.`

Return a link to the next subset of the collection.

`ironic.`

Return whe collection has more item

`ironic.`

Buil a collection dict including the

next link for

paging support.

Paramet

- **ite** List of unsanitized item to include in the

collection

- **ite** Nam of dict key for item valu
- **lim** Paging limi
- **url** Base URL for building next link
- **fie** Optiona

field to use for sanitize function

- **san** Optiona sanitize function run on each item item

changes will be done in-place

- **key** Key nam for buil ing next URL

- **kwa** othe arguments pass to `get`

Parm sa Dictionary with additiona ar-gu-men to be pass

to the sanitizer.

Returns

A dict con-tain-ing `ite` and `nex` val-ues

ironic.api.controllers.v1.conductor module

class `i`

Base `pec` `res` `Res`

RES con-troll for con-duc-tors.

get_all

Re-triev a list of con-duc-tors.

Paramet

- **mar** pagination mark for large data sets.
- **lim** maximum number of resour to return in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** column to sort results by. Default: id.
- **sor** direction to sort. asc or desc De-

fault asc.

- **fie** Optiona a list with a speci- fied set of field

of the resource to be returned.

- **det** Optiona bool to in- di- cate whe re- triev a list

of conductors with detail.

get_one Re- triev in- for- ma- tion abou the give con- duc- tor.

Paramet

- **hos**

host nam of a conductor.

- **fie** Optiona a list with a speci-fied set of field

of the resource to be returned.

invalic

ironic.

ironic.

ironic.api.controllers.v1.deploy_template module

class i

Base pec res Res

RES controll for deploy

tem-
plate

delete

Dele
a
de-
ploy
tem-
plate

Paramet

tem

UUI
or
log-
i-
cal
nam
of
a
de-
ploy
tem-
plate

get_all

Re-
triev
a
list
of
de-
ploy
tem-
plate

Paramet

- **mar**
  pag-
  i-
  na-
  tion
  mark
  for
  large
  data
  sets.
-

lim maximum number of resour to return in a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** column to sort results by. Default id.
- **sor** direction to sort. asc or desc Default asc.
- **fie** Optiona a list with a speci-

fied set of field

of the resource to be returned.

- **det** Optional bool to indicate whe retriev a list

of deploy templates with detail.

get_one Retriev information about the give deploy template

Paramet

- **tem** UUI or logical nam of a deploy template

- **fie** Optiona a list with a speci fied set of field

of the resource to be returned.

invalic

patch(*t*

Update an existing deploy template

Paramet

- **tem** UUI or logical nam of a deploy template
- **pat** a

json PAT document to apply to this de-

ploy template.

post (*te*

Create a new deploy template

Parame

tem

a deploy template with the request body

`ironic.`

`ironic.`

Add links to the deploy template

`ironic.`

Argument validator to check template for duplicate steps

`ironic.`

`ironic.`

`ironic.`

Remov sensitive and unrequest data

Will only keep the field specified in the `fie` parameter.

Paramet

`fie`

(*li*
of
str
list of field to preserve or Non to

preserve them all

ironic.api.controllers.v1.driver module

class i

Base
pec
res
Res

RES controll for drive pass

This controll allow vendors to expose cros node functionality in the Ironic API. Ironic will merely relay the message from here to the specified driver, no introspection will be made in the message body.

methods
Retriev information

ma-
tion
abou
ven-
dor
meth
ods
of
the

given driver.

Paramet

dri
nam
of
the
drive

Returns

dic-
tio-
nary
with
<ver
dor
meth
nam
meta
data
en-
tries

Raises

Driv
Not-
Foun
if
the
drive
nam
is
in-
valid
or
the
drive

cannot be loaded.

class i

Base
`pec`
`res`

Res

logical

Returns the logical disk properties for the driv

Paramet

dri

Nam of the driv

Returns

A dictionary containing the properties that can

be mentioned for logical disks and a textual description for them.

Raises

UnsupportedDrivtension if the driv does

sup-
port

RAID configuration.

Raises
No-
tAc-
cept
able
if
re-
ques
ver-
sion
of
the
API
is

less than 1.12.

Raises
Driv
Not-
Foun
if
driv
is
not
load
on
any
of
the
con-

ductors.

class i

Base
pec
res
Res

RES
con-
troll
for
Driv

get_all
Re-
triev
a

list of driv

get_one

Retriev a single driv

propert

Retriev property information of the give driv

Paramet

dri nam of the driv

Returns

dictionary with <pro erty nam description en tries

Raises

Driv Not-Foun (HT 404) if

the driver name is invalid or

the driver cannot be loaded.

raid =

Exposed RAI as a sub-elem of driv

vendor_

`ironic.`

Convert driv type info to a dict.

Paramet

- **nam** name of a hardware type
- **hos** list of conductor

host nam driv is active on.

- **det** bool whe to include detaile info such as the type

field and default/enabled interfaces fields.

- **int** optiona list of dicts of hardware interface info

Returns
dict representing the driv object.

`ironic.`
This meth

hide field that were added in new API versions.

Certain field were introduced at certain API versions. Thes

fields are only made available when the requests API version matches or exceeds the versions when these fields were introduced.

`ironic.`

Con vert drive and hard ware type to an API- seria object.

Paramet

- **har** dict map ping

hard-
ware
type
nam
to
con-
duc-
tor
host

names.

- **det**
  bool
  whe
  to
  in-
  clud
  de-
  taile
  info
  such
  as
  the
  type

field and default/enabled interfaces fields.

Returns
an
API-
seria
drive
col-
lec-
tion
ob-
ject.

ironic.api.controllers.v1.event module

class i

Base
pec
res
Res

RES
con-
troll
for
Even

post (*ev*

ironic.

Validator for even

ironic.api.controllers.v1.node module

class i

Base pec res Res

get (*nod*

Get the current boot device for a node

Paramet

nod

the UUI or logical nam of a node

Returns

a json object

containing:

boot_
the boot device, one of *iro com boo* or Non if it

is unknown.

persist
Whe the boot device will persist to all future boot

or not, None if it is unknown.

put (*nod*
Set the boot device for a node

Set the boot device to use

on next reboot of the node

Paramet

- **`nod`** the UUI or logical nam of a node
- **`boo`** the boot device one of *`irc com boc`*
- **`per`** Boo valu True if the boot device will persist to

all future boots, False if not. Default: False.

`support`
Get

a list of the supported boot devices

Parame

nod the UUI or logical nam of a node

Returns A json object with the list of supported boot devices

class i

Base obj

class i

Base pec res Res

get_all

Get node hard

ware component and their indicators.

Paramet

nod the UUI or logical nam of a node

Returns

A json object of hardware component (*ir com com*

as keys with indicator IDs (from *get_supported_indicators*) as values.

get_one

Get node hardware component indicator and its

state.

Parame

- **nod** the UU or logical nam of a nod
- **ind** Indicator ID (as reporte by *get_*

Returns

a dict with the state key and one of mod as a valu

put (*nod*

Set node hardware component indicator to the

desired state.

Parame

- **nod** the UUI or logical nam of a nod
- **ind** Indicator ID (as reporte by *get_*
- **sta** Indicator state one of mod

class i

Bas
pec
res

`Res`

put (*nod*

Inject NM for a node

Inject NM (Nor Mas able Interrupt) for a node immediately.

Parame

nod the UUI or logical nam of a node

Raises

Not-Foun if reques version of the API does sup-inject nmi.

port

Raises
HTT For- bid- den if the pol- icy is not au- tho- rized

Raises
Nod Not- Fou if the node is not foun

Raises
Nod Lock if the node is lock by an- othe con- duc- tor.

Raises
Un- sup- port- ed- Driv ten- sion if

the node driv does sup-

port management or management.inject_nmi.

Raises

In- valic Pa- ram- e- ter- Valu whe the wron driv info is

specified or an invalid boot device is specified.

Raises

Miss ing- Pa- ram- e- ter- Valu if miss ing sup- plied info

class i

Base pec res Res

get (*nod*

Get con- nec- tion in- formation abou the console

Paramet

nod UUI or logical nam of a node

put (*nod* Star and stop the node console

Paramet

- **nod** UUI or logical nam of a node
- **ena** Boo valu whe to enable or disable

the console

class i

Base pec res Res

delete
Re- mov the node from main te- nanc mod

Paramet
nod
the UUI or log- i- cal nam of a node

put (*nod*
Put the node in main te- nanc mod

Paramet

- **nod** the UUI or

logical_
of
a
node

- **rea**
Optiona
the
reason
why
its
in
maintenanc

class i

Base
pec
res
Res

boot_de
Exposed
boot
as
a
sub-
elem
of
management

indicat
Exposed
indicators
as
a
sub-
elem

of man-age-

ment

inject_

Ex-pose in-ject_ as a sub-elem of man-age-men

class i

Base pec res Res

console

Ex-pose con-sole as a sub-elem of state

get (*nod*

List the state of the node

Paramet

nod the UUI or log-ical_ of a node

power (*r*

Set the pow state of the node

Paramet

- **nod** the UUI or logical nam of a node
- **tar** The desired pow state of the node
- **tim** timeout (in seconds pos- i- tive integer

(>

0) for any power state. `None` indicates to use default timeout.

Raises
Clie
ror
(HT
409)
if
a
pow
op-
er-
a-
tion
is
al-

ready in progress.

Raises
In-
valid
State
(HT
400)
if
the
re-
ques
tar-
get
state
is

not valid or if the node is in CLEANING state.

Raises
No-
tAc-
cept
able
(HT
406)
for
soft
re-
boot
soft
pow
off

or timeout parameter, if requested version of the API is less than 1.27.

Raises
In-

valic (HT 400) if time out valu is less than 1.

provisi

Asy chro trigger the provisioning of the node

This will set the target provision state of the node and

a background task will begin which actually applies the state change. This call will return a 202 (Accepted) indicating the request was accepted and is in progress; the client should continue to GET the status of this node to observe the status of the requested action.

Paramet

- **nod** UUI or logical nam of a nod

- **tar** The desire provision state of the node or verb
- **con** Optiona A gzip and base encode configdrive or

a dict to build a configdrive from. Only valid when setting provision state to active or rebuild.

- **cle** An ordere list of clea ing step that will be performed on the node. A cleaning step is a dictionary with required keys interface and step, and optional key args. If specified, the value for args is a keyword variable argument dictionary that is passed to the cleaning step method.:

```
↪.., <argn>: <valuen>} }
```

For example (this isnt a real example, this clea ing step doesnt exist):

This is requir (and only vali whe target is clea

- **`dep`** A list of deploy steps that

will be perform on the node. A deploy step is a dictionary with required keys interface, step, priority and args. If specified, the value for args is a keyword variable argument dictionary that is passed to the deploy step method.:

```
↪.., <argn>: <valuen>}
```

For ex

ample (this isnt a real example, this deploy step doesnt exist):

This is

used only whe target is active or rebuil and is optional.

- **`res`** A strin representing the pass wor to be set inside the rescue environment. This is required (and only valid), when target is rescue.

- **`dis`** Whe to skip boot ing ram for clea ing.

Raises
Nod Locl (HT 409) if the node is current

lock

Raises
Clie ror (HT 409) if the node is already being provisioned.

Raises
Invalid Parameter Valu (HT 400) if validation of clean_steps, deploy_steps or power driver interface fails.

Raises
Invalid State (HT 400) if the requested transition is not possible from the current state.

Raises
Nod Mai

tenanc (HT 400) if operation cannot be performed because the node is in maintenance mode.

Raises
NoFductorWor (HT 503) if no work ers are avai able

Raises
NotAcceptable (HT 406) if the API version specified does not allow the requested state transition or parameters.

raid(*no*
Set the target raid config of

the nod

Parame

- **nod** the UU or logical nam of a nod
- **tar** Desire target RA configuration of the

node. It may be an empty dictionary as well.

Raises

Unsupport ed-Driv tension if the nod driv does sup-

port RAID configuration.

Raises

Invali

Parameter-Valu if validation of

target raid config fails.

Raises

NotAcceptable if requested version of the API is

less than 1.12.

class i

Base pec res Res

delete

Remove one or all trait from a node

Parame

tra

Strin valu trait

to re-mov from a node or Non If

None, all traits are removed.

get_all

List node trait

put (*trai*

Add a trait to a node

Paramet

- **tra** Strin valu trait to add to a node or Non Mu-tu-

ally exclusive with traits. If not None, adds this trait to the node.

- **tra** List of Strin trait to set for a node

or
Non
Mu-

tually exclusive with trait. If not None, replaces the nodes traits with this list.

class i

Base
pec
res
Res

delete
Detach
a
VIF
from
this
node

Parame
vif
The
ID
of
a
VIF
to
detach

get_all
Get
a
list
of
attach
VIFs

post (*vi*
Attach
a
VIF
to
this
node

Parame
vif
a

dictionary of information abou a VIF.

It must have an id key, whose value is a unique identifier for that VIF.

class i

Base pec res Res

RES controll for Vendor-Pass

This controll allow vendors to expose a custom func

tionality in the Ironic API. Ironic will merely relay the message from here to the appropriate driver, no introspection will be made in the message body.

methods

Retriev information abou

vendor meth ods of the

given node.

Paramet

nod

UUI or logical nam of a node

Returns

dictionary with <ver dor meth nam meta data entries

Raises

Nod Not-Four if the node is not foun

class i

Base pec res Res

RES controll for Nod

delete

Dele a node

Paramet

nod UUI or logical nam of a node

detail

Retriev a list of node with detail.

Paramet

- **cha** Optiona UUI of a chassis, to get only node for that chassis.

- **ins** Op- tiona UUI of an in- stanc to find the node as-

sociated with that instance.

- **ass** Op- tiona bool whe to re- turn a list of as- so-

ciated or unassociated nodes. May be combined with other parameters.

- **mai** Op- tiona bool valu that in- di- cates whe to get node

in maintenance mode (True), or not in maintenance mode (False).

- **ret** Optiona bool valu that indicates whe to get node which are retired.

- **pro** Optiona strin valu to get only node in that provi sion state.

- **mar** pagination mark for large data sets.

- **lim** maximum number of resour to return in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** column to sort results by. Default: id.
- **sor** direction to sort. asc or desc Default asc.
- **dri** Optional strin valu to get only node using that drive
- **res** Optional strin valu

to get only node with that re- sour

- **fau** Op- tiona strin valu to get only node with that fault

- **con** Op- tiona strin valu to get only node with that con- duc-

tor_group.

- **own** Op- tiona strin valu that set the own who node are to be retrurned.

- **les**
  Op-
  tiona
  strin
  valu
  that
  set
  the
  lesse
  who
  node
  are
  to

be returned.

- **pro**
  Op-
  tiona
  strin
  valu
  that
  set
  the
  proj
  -

  lesse
  or

owner - whose nodes are to be returned.

- **des**
  Op-
  tiona
  strin
  valu
  to
  get
  only
  node
  with
  de-
  scrip
  tion

field contains matching value.

from_ch
A
flag

to in- di- cate if the re- ques to this con-

troller are coming from the top-level resource Chassis

get_all

Re- triev a list of node

Parame

- **cha** Op- tiona UUI of a chas- sis, to get only node for

that chassis.

- **ins** Op- tiona UUI of an instan to find the node as-

sociated with that instance.

- **ass** Optiona bool whe to return a list of associated or unassociated nodes. May be combined with other parameters.

- **mai** Optiona bool valu that indicates whe to get node in maintenance mode (True), or not in maintenance mode (False).

- **ret** Optiona bool valu that indicates whe to get

re-

tired nodes.

- **pro** Op- tiona strin valu to get only node in that pro- vi-

sion state.

- **mar** pag- i- na- tion mark for large data sets.
- **lim** max i- mum num ber of re- sour to re- turn in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** col- umn to

sort results by. Default id.

- **sor** direction to sort. asc or desc Default asc.
- **dri** Optional string value to get only node using that drive
- **res** Optional string value to get only node with that resour
-

con Op- tion strin valu to get only node with that con- duc-

tor_group.

- **con** Op- tion strin valu to get only node man aged by that

conductor.

- **own** Op- tion strin valu that set the own who node are to

be retrurned.

- **les** Op- tion strin valu

that set the lesse who node are to

be returned.

- **pro** Optiona strin valu that set the proj - lesse or

owner - whose nodes are to be returned.

- **fie** Optiona a list with a spec ified set of field

of the resource to be returned.

- **fau** Optiona strin valu to get only node with field contains matching value.

that faul

- **des** Optiona strin valu to get only node with description

get_one

Retriev information about the given node

Paramet

- **nod** UUI or logical nam of a node
- **fie** Optiona a list with a

spec
i-
fied
set
of
field

of the resource to be returned.

invalic

mainter
Ex-
pose
mair
te-
nanc
as
a
sub-
elem
of
node

managen
Ex-
pose
man
age-
men
as
a
sub-
elem
of
node

patch(*r*
Up-
date
an
ex-
ist-
ing
node

Paramet

- **nod**
UUI
or
log-

ical nam of a nod

- res whe to reset hard ware interface to thei de-

faults. Only valid when updating the driver field.

- pat a json PAT document to apply to this nod

post (*no*
Create a new nod

Parame
nod a nod with the request

bod

states

Ex-
pose
the
state
con-
troll
ac-
tion
as
a
sub-
elem
of

nodes

validat

Val-
i-
date
the
driv
in-
ter-
face
us-
ing
the
node
UUI

or name.

Note
that
the
node
in-
ter-
face
is
dep-
re-
cate
in
favo
of

the node interface

Parame

-

nod UUI or nam of a node

- **nod** UUI of a node

vendor_ A resource used for vendors to expose a custom

functionality in the API

`ironic.`

`ironic.`

This meth hide field that were adde in new API versions

Certain node field were in-

tro-
duce
at
cer-
tain
API
ver-
sion

These fields are only made available when the requests API version matches or exceeds the versions when these fields were introduced.

`ironic.`

Add
links
to
the
in-
di-
ca-
tor.

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

Re-
mov
sen-

sitive and unreques data

Will only keep the field specified in the `fie` parameter.

Paramet

- **`fie`** (*`li of str`*) list of field to preserv or `Non` to preserve them all
- **`cdi`** Context dictionary for policy

values evalu-

ation. If not provided, it will be executed by the method, however for enumarting node lists, it is more efficent to provide.

- **sho** A bool valu to allow external single eval-

uation of policy instead of once per node. Default None.

- **sho** A bool valu to allow external evalu-a-

tion of policy instead of once per node. Default None.

- **eva** A bool valu to allow external evalu-ation of policy instead of once per node. Default None.

ironic.

ironic.

ironic.

ironic.

When creating an object, reject field that appear

in newer versions.

ironic.

ironic.

Cha provision state nam for API back ward compatibility.

Paramet

obj

The dict being returned to

the API clien that is

to be updated by this method.

`ironic.`

Val- i- date node net- work field

This meth val- i- date net- work data con- fig- u- ra- tion agai

JSON schema.

Paramet

net

a net- work field to val- i- date

Raises

In- valid if net- work data is not sche

com

ironic.api.controllers.v1.notification_utils module

`ironic.`

Help for emitting API end notifications

Paramet

- `con` request context.
- `obj` resource rpc object.
- `act` Action string to go in the Even Type
- `kwa`

kwa
to
use
whe
creating
the
notification payload.

ironic.

Help
for
emit
ting
API
start
notifications

Paramet

- **con**
  request
  context.
- **obj**
  resour
  rpc
  object.
- **act**
  Action
  strin

to go in the Even Typo

- **kwa** kwa to use whe creating the notification payload.

ironic.

Con text man agen to handle any error notifications.

Paramet

- **con** request context.
-

obj resource rpc object.

- **act** Action string to go in the Even Type

- **kwa** kwa to use whe creating the notifica-

tion payload.

ironic.api.controllers.v1.port module

class i

Base pec res Res

RES controll for Port

advance

delete

Dele
a
port

Parame

por
UUI
of
a
port

Raises

Op-
er-
a-
tionl
Per-
mit-
ted,
HTT
Not-
Foun

detail

Re-
triev
a
list
of
port
with
de-
tail.

Note
that
the
node
in-
ter-
face
is
dep-
re-
cated
in
favo
of

the node interface

Parame

- **nod** UUI or nam of a node to get only port for that

node.

- **nod** UUI of a node to get only port for that node

- **add** MA ad- dres of a port to get the port whic has

this MAC address.

- **por** UUI or nam of a

port- grou to get only port for

that portgroup.

- **mar** pag- i- na- tion mark for larg data sets.

- **lim** max i- mun num ber of re- sour to re- turn in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** col- umn to sort re- sults by. De- fault id.

- **sor**

direction to sort. asc or desc Default asc.

Raises

NotAcceptable HTT NotFour

`get_all`

Retriev a list of port

Note that the node interface is deprecated in favo of

the node interface

Parame

- `nod` UUI or nam

of a node to get only port for that node.

- **nod** UUI of a node to get only port for that node
- **add** MA addres of a port to get the port whic has this MAC address.
- **mar** pagination mar for large data sets.

- **lim** max imun num ber of resour to return in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** column to sort results by. Defaul id.
- **sor** direction to sort. asc or desc Defaul asc.
- **fie** Optiona a list with a spec

i-
fied
set
of
field

of the resource to be returned.

- **por**
UUI
or
nam
of
a
port-
grou
to
get
only
port
for

that portgroup.

Raises
No-
tAc-
cept
able
HTT
Not-
Foun

get_one
Re-
triev
in-
for-
ma-
tion
abou
the
give
port

Paramet

- **por**
UUI
of
a
port

- **fie** Optiona a list with a specified set of field

of the resource to be returned.

Raises
NotAcceptable HTT NotFoun

invalic

patch (*p*
Updates an existing port

Paramet

- **por** UUI of a port
- **pat** a json PAT docu-

men
to
apply
to
this
port

Raises
NotAcceptable
HTT
Not-
Foun

post (*po*
Create
a
new
port

Paramet
por
a
port
with
the
request
body

Raises
NotAcceptable
HTT
Not-
Foun
Conflict

ironic.

ironic.

`ironic.`

`ironic.`

Remov sensitive and unreques data

Will only keep the field specified in the `fie` parameter.

Paramet

`fie` (*li of str*) list of field to preserv or `Non` to preserve them all

ironic.api.controllers.v1.portgroup module

class i

Base
pec
res
Res

RES
con-
troll
for
port-
grou

delete

Dele
a
port-
grou

Parame

por
UUI
or
log-
i-
cal
nam
of
a
port-
grou

detail

Re-
triev
a
list
of
port-
grou
with
de-
tail.

Parame

- **nod**

UUI or nam of a nod to get only portgrou for

that node.

- **add** MA addres of a portgrou to get the portgrou

which has this MAC address.

- **mar** pagination mark for large data sets.
- **lim** maximum number of resour to

return in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** column to sort results by. Default: id.
- **sor** direction to sort. asc or desc Default: asc.

get_all

Retriev a list of port-grou

Parame

- **nod** UUI or nam of a node

to get only port grou for

that node.

- **add** MAC addres of a port grou to get the port grou

which has this MAC address.

- **mar** pagination mark for large data sets.

- **lim** maximun number of resour to return in

a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **sor** col- umn to sort re- sults by. De- fault id.
- **sor** di- rec- tion to sort. asc or desc De- fault asc.
- **fie** Op- tiona a list with a spec i- fied set of field

of the resource to be returned.

get_one Re- triev in- for- ma- tion abou the give portgrou

Paramet

- **`por`** UUI or logical nam of a portgrou
- **`fie`** Optiona a list with a specified set of field

of the resource to be returned.

`invalic`

`patch` (*p*

Update an existing portgrou

Paramet

- **`por`** UUI or

logical nam of a port grou

- **pat** a json PAT document to apply to this port

group.

post (*po*

Create a new port grou

Parame

por a port grou with the request bod

`ironic.`

Add link to the port

grou

ironic.

ironic.api.controllers.v1.ramdisk module

class i

Base
pec
res
Res

Con
troll
han-
dling
hear
beat
from
de-
ploy
ram

post (*no*
ag

Pro-
cess
a
hear
beat
from
the
de-
ploy
ram

Parame

- **nod**
the
UUI
or
log-
i-
cal

nam of a node

- **cal** the URL to reac back to the ram

- **age** The version of the agen that is hear beating. Non

indicates that the agent that is heartbeating is a version before sending agent_version was introduced so agent v3.0.0 (the last release before sending agent_version was introduced) will be assumed.

- **age** random generated validation token.

- **age** TLS certificate

to use to connect to the agen

- **age** Current status of the hear beating agen Used by

anaconda ramdisk to send status back to Ironic. The valid states are start, end, error

- **age** Optional status message describing current agen

Raises
Nod Not-Foun if node with provide UUI or nam was

not

found.

Raises

In- valic uidC Nam if node is not valic nam or UUI

Raises

No- Valic Hos if RPC topic for node coul not be re- triev

Raises

Not- Four if re- ques API ver- sion does not al- low this

endpoint.

class i

Base pec res

Res

Con troll han- dling node look for a de- ploy ram

get_all

Look up a node by its MA ad- dres and op- tion- ally

UUID.

If the re- stric op- tion is set to True (the de- fault limi

the search to nodes in certain transient states (e.g. deploy wait).

Paramet

- **add** list of

MA
ad-
dres
for
a
nod

- **nod**
  UUI
  of
  a
  node

Raises
Not-
Foun
if
re-
ques
API
ver-
sion
does
not
al-
low
this

endpoint.

Raises
Not-
Foun
if
suit-
able
node
was
not
foun
or
node
pro-
vi-

sion state is not allowed for the lookup.

Raises
In-
com
plete
if
nei-
ther

node
UUI
nor
any
valic
MAC
ad-

dress was provided.

propert

`ironic.`

`ironic.`

ironic.api.controllers.v1.utils module

class i

Base
`obj`

Ob-
ject
to
hold
the
re-
spon
from
a
pass
call

obj
Stor
the
re-
sult
ob-
ject
from
the
view

status_
Stor
an
op-
tiona

sta-
tus_

`ironic.`
Chec
if
agen
to-
ken
is
avai
able

`ironic.`
Chec
if
agen
ver-
sion
is
al-
lowe
to
be
pass
into
hear

beat.

Ver-
sion
1.36
of
the
API
adde
the
abil-
ity
for
agen
to
pass

their version information to Ironic on heartbeat.

`ironic.`
Chec
if
back
fill-
ing
al-
lo-

cations is allowe

Version 1.58 of the API adde support for backfilling al-

locations.

`ironic.`

Chec if allocation own field is allowe

Version 1.60 of the API adde the own field to the allo-

cation object.

`ironic.`

Chec

if updating an existing allocation

is allowed or not.

Version 1.57 of the API adde support for updating an

allocation.

ironic.

Che if accessing allocation endpoin is al-

lowed.

Version 1.52 of the API expose allocation endpoin

and allocation_uuid field for the node.

ironic.

Chec if we shou support bios interface and endpoin

Version 1.40 of the API adde support for bios interface

ironic.

Chec if build ing config drive is allowe

Version

1.56 of the API adde sup- port for buil ing con- fig-

drive.

`ironic.`

Che if con- fig- driv can con- tain a ven- dor_ key.

Ver- sion 1.59 of the API adde sup- port for con- fig- driv ven-

dor_data.

`ironic.`

Che if de- ploy are avai able

`ironic.`

Che
if
accessing
deploy
template
endpoin
is
allowed.

Version
1.55
of
the
API
exposed
deploy
template
endpoin

`ironic.`

Che
if
passing
a
detail=
quer
strin
is
allowe

Version
1.43
allows
a
user
to

pass
the
de-
tail
quer
strin

to list the resource with all the fields.

`ironic.`

Che
if
dy-
nam
driv
API
calls
are
al-
lowe

Ver-
sion
1.30
of
the
API
adde
sup-
port
for
all
of
the
driv

composition related calls in the /v1/drivers API.

`ironic.`

Che
if
dy-
nam
in-
ter-
face
field
are
al-
lowe

Ver-
sion
1.31
of

the API adde sup-port for view-ing and set-ting the fields in `V31_FIELDS` on the node object.

`ironic.`

Chec if ac-cess-ing con-duc-tor end-poin is al-lowe

Ver-sion 1.49 of the API ex-pose con-duc-tor end-poin and conductor field for the node.

`ironic.`

Chec if ac-cess-ing even end-poin is

al-
lowe

Version 1.54 of the API adde the even endpoin

`ironic.`
Chec if a field is allowe in the current version

`ironic.`
Chec if Inject NMI is allowe for the node

Version 1.29 of the API allows Inject

NMI for the node

ironic.

Che if in- spec tion abor is al- lowe

Ver- sion 1.41 of the API adde sup- port for in- spec tion abor

ironic.

Che if in- spec wait is al- lowe for the node

Ver- sion 1.39 of the API adds in- spec

wait state to substi-

tute inspecting state during asynchronous hardware inspection.

`ironic.`

Chec if links are displa

Version 1.14 of the API allows the display of links to

node states and driver properties.

`ironic.`

Chec if network is allowe in ports link_

`ironic.`

`ironic.`

Chec if we shou support node

re-
buil
with
con-
fig-
driv

Ver-
sion
1.35
of
the
API
adde
sup-
port
for
node
re-
buil
with

configdrive.

`ironic.`

Chec
if
we
shou
re-
turn
lo-
cal_
and
pxe_
field

Ver-
sion
1.19
of
the
API
adde
sup-
port
for
these
new
field
in

port object.

`ironic.`

Che if ac- cess ing in- ter- nal_ is al- lowe for the

port.

Ver- sion 1.18 of the API ex- pose in- ter- nal_ read only field

for the port.

ironic.

Che if port is_sı field is al- lowe

Ver- sion 1.53 of the API adde is_sı field to the port

ob-
ject.

`ironic.`

Che
if
port
phys
i-
cal
net-
work
field
is
al-
lowe

Ver-
sion
1.34
of
the
API
adde
the
phys
i-
cal
net-
work
field

to the port object. We also check whether the target version of the Port object supports the physical_network field as this may not be the case during a rolling upgrade.

`ironic.`

Che
if
mod
and
prop
er-
ties
can
be
adde
to/q
from
a

portgroup.

Ver-
sion
1.26

of
the
API
adde
mod
and
prop
er-
ties
field
to

portgroup object.

`ironic.`

Chec
if
we
shou
sup-
port
port-
grou
op-
er-
a-
tions

Ver-
sion
1.23
of
the
API
adde
sup-
port
for
Port
Grou

`ironic.`

Chec
if
port-
grou
can
be
used
as
sub-
con-
troll

Version 1.24 of the API adde sup-port for Port grou as sub-

controllers

ironic.

Chec if RAI configuration is allowe for the

node.

Version 1.12 of the API allows RAI configuration

for the node.

ironic.

Chec if hear beat

and look end-poin are al-lowe

Ver-sion 1.22 of the API in-tro-duce them

ironic.

Chec if chas sis_ can be re-mov from node

Ver-sion 1.25 of the API adde sup-port for chas sis_ re-mov

ironic.

Chec if we shou sup-port

rescue and unrescue operations and interface.

Version 1.38 of the API adde support for rescue and unrescue.

ironic.

Chec if pass ing a reset_ quer strin is al lowe

ironic.

Chec if Soft Pow Off is al lowe for the node

Ver-

sion
1.27
of
the
API
allows
Soft
Pow
Off,
including

Soft Reboot, for the node.

`ironic.`
Chec
if
heartbeat
accept
agen
and
agen

`ironic.`
Chec
if
we
shou
support
storage_
node
and
driv
field

Version
1.33
of
the
API
adde
support
for
storage
in-

interfaces.

`ironic.`
Chec if traits are allowe for the node

Version 1.37 of the API allows traits for the node

`ironic.`
Chec if heartbeat accept agen

`ironic.`
Chec if node can be used

Version 1.28 of the API adde support for

VIF to be attached to Nodes.

`ironic.`

Che if volume connectors and targets are allowe

Version 1.32 of the API adde support for volume connectors and targets

`ironic.`

Apply a JSO patc one operation at a time

If

the patc fails to apply, this allows us to determine which operation failed, making the error message a little less cryptic.

Paramet

- **doc** The JSO document to patc
- **pat** The JSO patc to apply.

Returns
The result of the patc operation.

Raises
Patc ror if the

patc fails to apply.

Raises

exception. if the patc adds a new root attribu

`ironic.`

Chec if the specified policy authorizes reques

on allocation.

Param

policy_ Nam of the policy to chec

Param

al-

location_the UUI or logical nam of a

node.

Raises
HTT Forbidden if the policy forbids access

Raises
AllocationNotFour if the node is not foun

Returns
RPC node identified by node

`ironic.`

`ironic.`

`ironic.`

Chec if de- ploy step are al- lowe

`ironic.`

Chec if get- ting de- taile drive info is al- lowe

Ver- sion 1.30 of the API al- lows this.

`ironic.`

Chec if fil- ter- ing node by con- duc- tor is

al-
lowe

Ver-
sion
1.49
of
the
API
al-
lows
fil-
ter-
ing
node
by
con-

ductor.

`ironic.`

Chec
if
fil-
ter-
ing
node
by
con-
duc-
tor_
is
al-
lowe

Ver-
sion
1.46
of
the
API
al-
lows
fil-
ter-
ing
node
by
con-

ductor_group.

`ironic.`

Chec
if

filtering node by fault is allowe

Version 1.42 of the API allows filtering node by fault

`ironic.`

Chec if filtering node by lesse is allowe

Version 1.62 of the API allows filtering node by lesse

`ironic.`

Chec if filtering node by own is allowe

Version 1.50 of the API allows filtering node by own

`ironic.`

Chec if filtering drive by classic/c is allowe

Version 1.30 of the API allows this.

`ironic.`

`ironic.`

Chec if fil- ter- ing node by drive is al- lowe

Ver- sion 1.16 of the API al- lows fil- ter node by drive

`ironic.`

Chec if fetch ing a sub- set of the re- sour at- tribu

is allowed.

Ver- sion 1.8 of the API al- lows

fetch ing a subset of

the resource attributes, this method checks if the required version is being requested.

`ironic.`

Chec if filtering node by resour is allowe

Version 1.21 of the API allows filtering node by re-

source_class.

`ironic.`

Chec if fetch ing a particular field is allowe

This
meth
chec
if
the
re-
quir
ver-
sion
is
be-
ing
re-
ques

for fields that are only allowed to be fetched in a particular API version.

`ironic.`

Chec
if
fetch
ing
a
par-
tic-
u-
lar
field
of
a
port

group is allowed.

This
meth
chec
if
the
re-
quir
ver-
sion
is
be-
ing
re-
ques

for fields that are only allowed to be fetched in a particular API version.

`ironic.`

Chec
for
re-

ques
non-
exist
field

Chec
if
the
user
re-
ques
non-
exist
field

Paramet

`fie`
A
list
of
field
re-
ques
by
the
user

Object_f
A
list
of
field
sup-
porte
by
the
ob-
ject.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
in-
valid
field
were
re-

quested.

ironic.

Che if filtering node by provision state is al-

lowed.

Version 1.9 of the API allows filter node by provi-

sion state.

ironic.

Che if the list policy authorizes this reques on

an object.

Param

ob-

ject_
type
of
ob-
ject
be-
ing
chec

Param

own
own
fil-
ter
for
list
quer
if
any

Raises

HTT
For-
bid-
den
if
the
pol-
icy
for-
bids
ac-
cess

Returns

own
that
shou
be
used
for
list
quer
if
need

`ironic.`

Chec
if
the
spec

ified policies authorize this re-

quest on a node.

Param
pol-
icy_
List
of
pol-
icy
nam
to
chec

Param
node
the
UUI
or
logical
nam
of
a
node

Param
with
whe
the
RPC
node
shou
include
the
suffix

Raises
HTT
Forbidden

if the policy forbids access

Raises
Nod Not-Foun if the node is not foun

Returns
RPC node identified by node

`ironic.`

Chec if the spec i- fied policy authorizes this request on a node.

Param
policy_ Nam of the

policy to chec

Param
node the UUI or logical nam of a node

Param
with whe the RPC node shou include the suffix

Raises
HTT Forbidden if the policy forbids access

Raises
Nod Not-Foun if the node is

not foun

Returns
RPC node iden tified by node

`ironic.`

Chec if the policy authorizes this reques on an object.

Param
object_ type of object being chec

Param
policy_ Nam of the policy to chec

Param

own
the
own

Param
less
the
less

Param
con-
ceal
the
UUI
of
the
node
IF
we
shou
con-
ceal
the

existence of the node with a 404 Error instead of a 403 Error.

Raises
HTT
Forbidden if the policy forbids access

`ironic.`
Che
if the specified policy is authorised for

this request.

Policy_n

Nam of the policy to chec

Raises

HTT Forbidden if the policy forbids access

ironic.

Chec if the spec ified policy is authorised for

this request.

Policy_n

Nam of the policy to chec

Returns

True if pol-

icy
is
matc
otherwise
false

ironic.

Chec
if
the
specified
policy
authorizes
this
request on a port.

Paramet

- **por**
  Boo
  valu
  default
  false
  indicating
  if
  the
  list
  policy check is for a portgroup as the policy names are different between ports and portgroups.
- **par**
  The
  UUI
  of
  a

node if any, to apply a pol-

icy check to as well before applying other policy check operations.

- **par** The UUI of the parent portgrou if the list of

ports was retrieved via the /v1/portgroups/<uuid>/ports.

Raises
HTT Forbidden if the policy forbids access

Returns
own that shou be used for list quer if need

ironic.

Che if the spec i- fied pol- icy au- tho- rizes this re-

quest on a port.

Param
pol- icy_ Nam of the pol- icy to chec

Param
port The nam uuid or othe valid ID valu to find a port

or portgroup by.

Raises
HTT For- bid- den if the

policy forbids access

Raises

Nod Not-Foun if the node is not foun

Returns

RPC port identified by port associated node

`ironic.`

Chec if the specified policy authorizes this request on a volume.

Paramet

par

The UUI of

a node if any, to apply a pol-

icy check to as well before applying other policy check operations.

Raises
HTT For-bid-den if the pol-icy for-bids ac-cess

Returns
own that shou be used for list quer if need

ironic.

Chec if the spec-i-fied pol-icy au-tho-rizes this

re-

quest on a volume.

Param
pol-
icy_
Nam
of
the
pol-
icy
to
chec

Param
vol_
The
nam
uuid
or
othe
valid
ID
valu
to
find
a
vol-

ume target or connector by.

Param
target: Boo
valu
to
indicate
if
the
chec
is
for

a volume target or connector. Default value is False, implying connector.

Raises
HTT
Forbidden
if
the

policy forbids access

Raises
Volume Connector Not Found if the node is not foun

Raises
Volume get Not Foun if the node is not foun

Returns
RPC port identified by port as sociated node

`ironic.`
Generator

of fiel not al- lowe in the cur- rent

request.

ironic.

Get re- serv nam for a give con- troll

In- spec the con- troll class and re- turn the re- serv nam with

it. Reserved names are names that can not be used as an identifier for a resource because the names are either being used as a custom action or is the name of a nested controller inside the given class.

Paramet

cls

The con- troll class to be in- spec

ironic.

Get the

patc values corre spon ing to the spec ified path.

If there are multiple values spec ified for the sam path, for example

```
↪', 'value': 'abc'},
```

(continues on next page)

```
↪name', 'value': 'bca'}]
```

(continued from previous page)

return all of them in a list (pre-serving order)

Paramet

- **pat** HTT PAT request body
- **pat** the path to get the patc values for.

Returns
list of values for the specified path in the

patc

`ironic.`

Cal-
cu-
late
field
to
re-
turn
from
an
API
re-
ques

The
field
quer
and
de-
tail=
quer
can
not
be
pass
into
a
re-
quest at the same time. To use the detail query we need to be on a version of the API greater than 1.43. This function raises an InvalidParameterValue exception if either of these conditions are not met.

If
these
chec
pass
then
this
func-
tion
will
re-
turn
ei-
ther
the
fields passed in or the default fields provided.

Paramet

- **fie** The field quer pass into the API request.
- **det** The detail quer pass into the API reques
- **def** The default field to return if field and detail=

Raises
In- valid Parameter- Valu if there is an in-

valic

combination of query strings or API version.

Returns
field
pass
in
valu
or
default

`ironic.`
Get
the
RPC
allocation
from
the
allocation

UUID or logical name.

Paramet
`all`
the
UUI
or
logical
nam
of
an
allocation.

Returns
The
RPC
allocation.

Raises
In-

valic
uidC
Nam
if
the
nam
or
uuid
pro-
vide
is
not
valid.

Raises
Al-
lo-
ca-
tion-
Not-
Four
if
the
al-
lo-
ca-
tion
is
not found.

`ironic.`

Get
the
RPC
al-
lo-
ca-
tion
from
the
al-
lo-
ca-
tion
UUID or logical name.

If
HAS
flag
is
set
in
the

peca
en-
vi-
ron-
men
try
also

looking for allocation_ident with .json suffix. Otherwise identical to get_rpc_allocation.

Paramet

`all`

the
UUI
or
log-
i-
cal
nam
of
an
al-
lo-
ca-

tion.

Returns

The
RPC
al-
lo-
ca-
tion.

Raises

In-
valic
uidC
Nam
if
the
nam
or
uuid
pro-
vide
is
not

valid.

Raises

Al-
lo-
ca-

tion-
Not-
Foun
if
the
allocation
is
not found.

`ironic.`

Get
the
RPC
deploy
tem-
plate
from
the
UUI
or
logical name.

Paramet

`tem`
the
UUI
or
logical
nam
of
a
deploy
tem-
plate.

Returns

The
RPC
deploy
tem-
plate

Raises

In-
valid
uidO
Nam
if
the
nam
or
uuid
pro-
vide
is
not

valid.

Raises

De-
ploy
plate
Four
if
the
de-
ploy
tem-
plate
is
not
foun

`ironic.`

Get
the
RPC
de-
ploy
tem-
plate
from
the
UUI
or
log-
i-

cal name.

If
HAS
flag
is
set
in
the

peca en-vi-ron-men try also

looking for template_ident with .json suffix. Otherwise identical to get_rpc_deploy_template.

Paramet

tem the UUI or log-i-cal nam of a de-ploy tem-

plate.

Returns

The RPC de-ploy tem-plate

Raises

In-valic uidC Nam if the nam or uuid pro-vide is not

valid.

Raises

De-ploy plate

Foun if the deploy template is not foun

`ironic.`

Get the RPC node from the node uuid or logical nam

Paramet

nod the UUI or logical nam of a node

Returns

The RPC Nod

Raises

Invalid uidO Nam if the nam or

uuid pro- vide is not

valid.

Raises

Nod Not- Fou if the node is not foun

`ironic.`

Get the RPC node from the node uuid or log- i- cal nam

If HAS flag is set in the peca en- vi- ron- men try also

looking for node_ident with .json suffix. Otherwise identical to get_rpc_node.

Paramet

nod the UUI

or log-ical nam of a node

Returns
The RPC Nod

Raises
In-valid uidO Nam if the nam or uuid pro-vide is not

valid.

Raises
Nod Not-Foun if the node is not foun

`ironic.`
Get the RPC port-grou from the port-grou UUI or

log-
i-

cal name.

Paramet

por
the
UUI
or
log-
i-
cal
nam
of
a
port-
grou

Returns

The
RPC
port-
grou

Raises

In-
valid
uidO
Nam
if
the
nam
or
uuid
pro-
vide
is
not

valid.

Raises

Port-
grou
Not-
Foun
if
the
port-
grou
is
not
foun

`ironic.`

Get the RPC port- grou from the port- grou UUI or log- i-

cal name.

If HAS flag is set in the peca en- vi- ron- men try also

looking for portgroup_ident with .json suffix. Otherwise identical to get_rpc_portgroup.

Paramet

por the UUI or log- i- cal nam of a port- grou

Returns

The RPC port- grou

Raises

In-

valic uidC Nam if the nam or uuid pro- vide is not valid.

Raises

Port grou Not- Four if the port- grou is not foun

ironic.

Re- turn node state to use by de- fault whe cre- at- ing new nodes.

Pre- vi- ousl the de- fault state for new node was AVA ABL Star ing with API 1.11 it is ENROLL.

`ironic.`

Returns whe the patc includ remov of the path (or subpath of).

Paramet

- **pat** HTT PAT request body
- **pat** the path to chec

Returns
True if path or subpath being remov Fals otherer-

wise.

`ironic.`

Returns whe the patc including operation on path

(or its subpath).

Paramet

- **pat** HTT PAT request body
- **pat** the path to chec

Returns
True if path or subpath being patc Fals otherwise

`ironic.`

Determine if the provide name is a valid host name

ironic.

Determine if the provided name is a valid node name

Check to see that the provided node name is valid and isnt a UUID.

Paramet

nam the node name to chec

Returns

True

if the nam is valid Fals otherwise

ironic.

ironic.

Help function to convert RPC objects to RES API dicts

Paramet

- **obj** RPC object to convert to a dict
- **inc** Whe to

in- clud stan- dard base class at- tribu cre- ated

- **inc** Whe to in- clud stan- dard base class at- tribu up- date

- **inc** Whe to in- clud stan- dard base class at- tribu uuid

- **lin** Whe spec i- fied, gen- er- ate a `lin` valu with

a

self and bookmark using this resource name

- **lin** Resour argument to be adde to generated

links. When not specified, the object `uuid` will be used.

- **fie** Key nam for dict values to populate directl

from object attributes

Returns
A dict containing values fron the object

`ironic.`

Update rpc object based on chan field in a dict.

Only field which have a corresponding sche field are update

when changed. Other values can be updated using the id_map.

Paramet

- **fro** Dict containing chan field values
- **rpc** Object to update chan field on

- **fie** Fiel nam on the rpc object
- **sch** json sche to get field nam of the dict
- **id_** Optiona dict map ping object field nam to arbitrary values when there is no matching field in the schema

`ironic.`

Validate that a patc list only modifies al-fields.

lowe

Paramet

- **pat** List of patc dicts to validate
- **all** List of field which are allowe to be patc

Returns
The list of field which will be patc

Raises
exception. if any patc chan a field not in `all`

ironic.

Validate a patc dict object agai a validator

or schema.

This function has the side-effe of dele ing any dict valu whic

is not in the schema. This allows database-loaded objects to be pruned of their internal values before validation.

Paramet

- **pat** dict representation of the object with patc

updates applied

- **sch** Any dict key not in the sche will be dele from the

dict. If no validator is specified then the resulting `patched_dict` will be validated agains the schema

- **val** Optiona validator to use if there is extra

validation required beyond the schema

Raises
exception. if validation fails

`ironic.`
Look up the node refer-

ence in the object and pop-

ulate a dict.

The node is fetch with the object nod at- tribu and the dict

`node_uuid` value is populated with the node uuid

Paramet

- **obj** object to get the node at- tribu
- **to_** dict to pop- u- late with a `nod` valu

Raises

ex- cep-

tion.
if
the
node
is
not
foun

`ironic.`

Re-
plac
`nod`
dict
valu
with
`nod`

`nod`
is
foun
by
fetch-
ing
the
node
by
id
look

Paramet

`to_`
Dict
to
set
`nod`
valu
on

Returns

The
node
ob-
ject
from
the
look

Raises

Nod
Not-
Foun
with
sta-
tus_

set
to
400
BAI
whe
node
is

not found.

`ironic.`

Re-
plac
`nod`
dict
valu
with
`nod`

`nod`
is
foun
by
fetch
ing
the
node
by
uuid
look

Paramet

`to_`
Dict
to
set
`nod`
valu
on

Returns

The
node
ob-
ject
from
the
look

Raises

Nod
Not-
Foun
with
sta-

tus_ set to 400 BAD whe node is not found.

ironic.

Removed sensitive and unrequested data

Will only keep the field specified in the fie parameter (plus the `links` field).

Paramet

- **to_** dict to sanitize
- **fie** (*li of str*

list of field to preserve or Non to

preserve them all

ironic.

ironic.

ironic.

Call a vendor pass API extension

Call the vendor pass API extension and process the meth

response to set the right return code for methods that are asynchronous or synchronous; Attach the return value to the response object if its being served statically.

Paramet

- **ide** The resour identification. For node vendor

passthru this is the nodes UUID, for drivers vendor passthru this is the drivers name.

- **met** The vendor meth nam
- **top** The RPC topi
- **dat** The data pass to the vendor meth Default to Non
- **dri** Boo valu Whe this is a node or drive ven-

dor pass

Defaults to False.

Returns
A WSI re- spon ob- ject to be re- turn by the API

ironic.api.controllers.v1.versions module

ironic.

Re- turns the max i- mum sup- port API ver- sion (as a string).

If the ser- vice is pinn the max i- mum API ver- sion is the pinned version. Otherwise, it is the maximum supported API version.

`ironic.`

Return the minimum support API version (as a

string)

ironic.api.controllers.v1.volume module

class `i`

Base `pec` `res` `Res`

REST controll for volume root

get()

`ironic.`

ironic.api.controllers.v1.volume_connector module

class `i`

Base `pec` `res` `Res`

RES

controll for Volume Connectors.

delete

Dele a volume connector.

Parame

con

UUI of a volume connector.

Raises

OperationNotPermitted if accessed with speci-fying a parent node.

Raises

Nod Lock if node is lock

by another conductor

Raises
Node Not-Fou if the node associated with the con-

nector does not exist

Raises
VolumeCon nectorN Fou if the volume connector

cannot be found

Raises
InvalidState If a node associated with the vol-

ume connector is not powered off.

get_al

Retriev a list of volumes connectors.

Parame

- **nod** UUI or nam of a node to get only volume con-

nectors for that node.

- **mar** pagination mar for large data sets.
- **lim** maximum number of

resour to return in a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.

- **`sor`** column to sort results by. Default id.
- **`sor`** direction to sort. asc or desc Default asc.
- **`fie`** Optiona a list with a specified set of field of the resource to be returned.
-

det Optiona whe to retriev with detail.

Returns
a list of volume connectors, or an emp list if

no volume connector is found.

Raises
InvalidParameterValu if sort_ does not exist

Raises
InvalidParameterValu if sort key

is invali

for sorting.

Raises
Invali Pa-ram-e-ter-Valu if both field and detail

are specified.

`get_one`
Retriev information abou the give volume connec-

tor.

Parame

- **`con`** UUI of a volume connector.
- **`fie`** Op-

tion a list with a spec i- fied set of field

of the resource to be returned.

Returns
API- seria vol- ume con- nec- tor ob- ject.

Raises
Op- er- a- tionl Per- mit- ted if ac- cess with spec i-

fying a parent node.

Raises
Vol- ume Con nec- torN Foun if no vol- ume con- nec-

tor

exists with the specified UUID.

invalic

patch(*c*

Update an existing volume connector.

Paramet

- **con** UUI of a volume connector.
- **pat** a json PAT document to apply to this vol-

ume connector.

Returns API- seria vol- ume

connector object.

Raises
Operation

Permitted if access with specifi-

fying a parent node.

Raises
Patc ror if a give patc can not be applie

Raises
Volume Con nectorN Foun if no volume connector

exists with the specified UUID.

Raises
In-

valic Parameter-Valu if the volume connec-

tors UUID is being changed

Raises
Nod Lock if node is lock by another conductor

Raises
Nod Not-Four if the node associated with the con-

nector does not exist

Raises
Volume Con nec-torT pe-An-dI-

dAl-
read
ists
if
an-

other connector already exists with the same values for type and connector_id fields

Raises

In-
valic
UID
if
in-
valic
node
UUI
is
pass
in
the
patc

Raises

In-
valic
State
If
a
node
as-
so-
ci-
ated
with
the
vol-

ume connector is not powered off.

post (*co*

Cre-
ate
a
new
vol-
ume
con-
nec-
tor.

Paramet

con
a
vol-

ume con- nec- tor with the re- ques body

Returns
API- seria vol- ume con- nec- tor ob- ject.

Raises
Op- er- a- tionl Per- mit- ted if ac- cess with spec i-

fying a parent node.

Raises
Vol- ume Con nec- torT pe- An- dI- dAl- read ists if a

volume connector already exists with the same type and connector_id

Raises

Vol-
ume
Con
nec-
torA
read
ists
if
a
vol-
ume
con-
nec-

tor with the same UUID already exists

`ironic.`

`ironic.`

ironic.api.controllers.v1.volume_target module

class i

Base
pec
res
Res

RES
con-
troll
for
Vol-
ume
gets

delete

Dele
a
vol-

ume tar-get.

Parame

`tar` UUI of a vol-ume tar-get.

Raises

Op-er-a-tionN Per-mit-ted if ac-cess with spec i-

fying a parent node.

Raises

Nod Lock if node is lock by an-othe con-duc-tor

Raises

Nod Not-Foun if the node as-so-ci- ated with the tar-

get does not exist

Raises Vol- ume get- Not- Four if the vol- ume tar- get can- not

be found

Raises In- valic State If a node as- so- ci- ated with the vol-

ume target is not powered off.

`get_all`

Re- triev a list of vol- ume tar- gets

Paramet

- **`nod`** UUI or nam of a node to get only volume targets for that node.
- **`mar`** pagination mark for large data sets.
- **`lim`** maxi mun number of resour to return in a single result. This value cannot be larger than the value of max_limit in the [api] section of the ironic configuration, or only max_limit resources will be returned.
- **`sor`** column to sort results

by. Default id.

- **sor** direction to sort. asc or desc Default asc.
- **fie** Optiona a list with a speci fied set of field

of the resource to be returned.

- **det** Optiona whe to retriev with detail.
- **pro** Optiona an as-

so-
ci-
ated
node
proj
(own
or
lesse

to filter the query upon.

Returns
a
list
of
vol-
ume
tar-
gets
or
an
emp
list
if
no

volume target is found.

Raises
In-
vali
Pa-
ram-
e-
ter-
Valu
if
sort_
does
not
ex-
ist

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
sort
key
is

invalic

for sorting.

Raises
InvalicParameterValu if both field and detail

are specified.

get_one
Retriev information about the give volume target.

Paramet

- **tar** UUI of a volume target.
- **fie** Optiona a list

with a specified set of field

of the resource to be returned.

Returns
API-seria volume target object.

Raises
OperationI Permitted if accessed with speci

fying a parent node.

Raises
VolumegetNotFour if no volume target with this

UUID exists

invalic

patch (*t*

Update an existing volume target.

Paramet

- **tar** UUI of a volume target.
- **pat** a json PAT document to apply to this vol-

ume target.

Returns

API seria volume target object.

Raises

Op-
er-
a-
tionl
Per-
mit-
ted
if
ac-
cess
with
spec
i-

fying a parent node.

Raises

Patc
ror
if
a
give
patc
can
not
be
ap-
plied

Raises

In-
valid
Pa-
ram-
e-
ter-
Valu
if
the
vol-
ume
tar-
gets

UUID is being changed

Raises

Nod
Lock
if
the
node
is
al-
read

lock

Raises
Nod Not-Foun if the node as-so-ci-ated with the vol-

ume target does not exist

Raises
Vol-ume get-Not-Foun if the vol-ume tar-get can-not

be found

Raises
Vol-ume get-Boo dex-Al-read ists if a vol-ume tar-

get already exists with the same node ID and boot index values

Raises
In-valic

UID if invalid node UUI is pass in the patc

Raises
InvalidState If a node associated with the volume target is not powered off.

post (*ta*

Create a new volume target.

Paramet
tar a volume target with the request body

Returns
API

seria volume target object.

Raises
Operationl Permitted if access with spec ifying a parent node.

Raises
Volume get-Boo dex-Alread ists if a volume target already exists with the same node ID and boot index

Raises
Volume getAlread ists if a volume target

with

the same UUID exists

`ironic.`

`ironic.`

Module contents

Version 1 of the Iron API

Specification can be foun at doc/

class i

Base

`obj`

Version 1 API controll root

index()

Submodules

ironic.api.controllers.base module

class i

Base
obj

API
Version
object.

max_str
HTT
response
head

min_str
HTT
response
head

static
Determine
the
API
version
request
base
on
the
head

ers supplied.

Paramet

- **hea**
webob
headers

- **def**
  version to use if not spec- ified in head ers

- **lat**
  version to use if latest is reques

Returns
a tuple of (ma- jor, minor) version num bers

Raises
webob.

string
HTT Hea strin carrying

the requested version

ironic.api.controllers.link module

ironic.

ironic.

Build a dict representing a link

ironic.api.controllers.root module

class i

Base obj

index(

ironic.

ironic.api.controllers.version module

ironic.

ironic.

Return a dict representing the

current defaul

version

id: The ID of the (major) version also acts as the re-

lease number links: A list containing one link that points to the current version of the API

status: Status of the version one of CUR REN SUP POR

DEPRECATED

min_ The current, maximum supported (major.n version of

API.

version Minimum supported (major.m version of API

Module contents

ironic.api.middleware package

Submodules

ironic.api.middleware.auth_public_routes module

class i

Base obj

A wrapper on authentication middleware

Does not perform verification

of authentication tokens for public routes in the API.

ironic.api.middleware.json_ext module

class i

Base

obj

Simplified processing of .json extension

Previousl Iron API used the gues feature. It was neve need as we never allowed non-JSON content types anyway. Now that it is removed, this middleware strips .json extension for backward compatibility.

ironic.api.middleware.parsable_error module

Middleware to replac the plain text message body of an

error response with one formatted so the client can parse it.

Base on peca

class `i`

Base `obj`

Replac error body with som thing the clien can pars

Module contents

class `i`

Base `obj`

A wrapper on authentication middleware

Does not perform verification of authentica-

tion tokens for public routes in the API.

class i

Base

obj

Simplified processing of .json extension

Previously Iron API used the gues feature. It was neve need

as we never allowed non-JSON content types anyway. Now that it is removed, this middleware strips .json extension for backward compatibility.

class i
Base
obj

Replac error body with some thing the clien can pars

Submodules

ironic.api.app module

class i
Base
osl
cor
COR

Iron spec COR clas

Wer addi the Iron spec version head ers to the list of sim-ple headers in order that a request bearing those headers might be accepted by the Ironic REST API.

simple_

class i

Base

obj

ironic.

ironic.

ironic.api.config module

ironic.api.functions module

class i

Base

obj

An argumen definition of an api entry

datatyp

Data type

default

Default valu if argumen is omitted

mandato
True if the argumen is man tory

name
argumen nam

resolve

class i
Base `obj`

An api entry definition

argumen
The func tion argumen (list of *Fun*

body_ty
If the body carry the data of a

single argumen

its type

doc
Func tion documentation

extra_
Dictionary of prot spec options

static
Returns the *Fun* of a meth

get_arg
Returns a *Fun* from its nam

ignore_
True if extra argumen shou

be ignore NOT inserted in the kwargs of the function and not raise UnknownArgument exceptions

name
Func tion nam

resolve

return_
Return type

set_arg

set_opt

status_
Status code

`ironic.`

`ironic.`

`ironic.`
alias of *iro api fun sig*

class i
Base obj

Decorator that spec ify

the argument type of an

exposed function.

Paramet

- **ret** Typ of the valu returne by the func tion
- **arg** Typ of the Nth argumen
- **bod** If the func tion take a final argumen that

is supposed to be the request body by itself, its type.

- **sta** HTT

return status code of the function.

- **ign** Allow extra/u arguments (defaults to Fals

Most of the time this decorator is not supposed to be used directly, unless you are not using WSME on top of another framework.

If an adap is used it will provide either a speciali

version of this decororator, either a new decorator named @wsexpose that takes the same parameters (it will in addition expose the function, hence its name).

`ironic.`

ironic.api.hooks module

class `i`

Base
`pec`
`hoc`
`Pec`

Attach the config object to the reques so controll

can get to it.

before

Ove
ride
this
meth
to
create
a
hool
that
gets
calle
after routing, but before the request gets passed to your controller.

Paramet

sta

The
Peca
`sta`

object for the current reques

class i

Bas

`pec`
`hoc`
`Pec`

Con figures a reques context and attach it to the request.

`after`(*s*

Ove ride this meth to create a hoo that gets calle after the request has been handled by the controller.

Parame

`sta`

The Peca `sta` object for the current reques

before

Override this meth to create a hool that gets calle after routing, but before the request gets passed to your controller.

Parame

sta

The Peca sta object for the current reques

class i

Base pec hoc Pec

Attach the dbap object to the re-

ques
so
con-
troll
can

get to it.

before
Ove
ride
this
meth
to
cre-
ate
a
hool
that
gets
calle
af-

ter routing, but before the request gets passed to your controller.

Paramet
sta
The
Peca
`sta`
ob-
ject
for
the
cur-
rent
re-
ques

class i
Base
`pec`
`hoo`
`Pec`

Wor
rpc.c
de-
se-
ri-
al-
ize_

de-
se-
ri-

alize_ buil rpc exception trace back into error message which is then sent to the client. Such behavior is a security concern so this hook is aimed to cut-off traceback from the error message.

after(*s*

Ove ride this meth to create a hool that gets calle after the request has been handled by the controller.

Paramet

sta

The Peca `sta` object for the current reques

class i

Base `pec` `hoc` `Pec`

Attach the

righ public_u to the reques

Attach the righ public_u to the reques so resour can create links even when the API service is behind a proxy or SSL terminator.

`before`

Ove ride this meth to create a hook that gets calle after routing, but before the request gets passed to your controller.

Paramet

`sta` The Peca `sta` object for the current reques

class i

Base
`pec`
`hoo`
`Pec`

Attach the rp-capi object to the reques so controll can get to it.

before

Override this meth to create a hook that gets calle after routing, but before the request gets passed to your controller.

Paramet

sta

The Peca `sta` object for the current reques

`ironic.`

ironic.api.method module

`ironic.`

Dec-
o-
ra-
tor
whic
plac
HTT
re-
ques
body
JSO
into
a

method argument

Paramet

bod

Nam
of
ar-
gu-
men
to
pop-
u-
late
with
body
JSO

`ironic.`

`ironic.`

Ex-
tract
in-
for-
ma-
tions
that
can
be
sent
to
the
clien

ironic.api.wsgi module

WSG
scrip
for
Iron
API
in-
stall
by
pbr.

`ironic.`

Module contents

ironic.cmd package

Submodules

ironic.cmd.api module

The
Iron
Ser-
vice
API

`ironic.`

ironic.cmd.conductor module

The
Iron
Man
age-
men
Ser-
vice

`ironic.`

`ironic.`

`ironic.`

ironic.cmd.dbsync module

Run storage data migration.

class `i`

Base `obj`

check_c

Chec the version of objects

Chec that the object version are compatible with this

release of ironic. It does this by comparing the objects .version field in the database, with the expected versions of these objects.

Return Non if

com-
pat-
i-
ble;
a
strin
de-
scrib
ing
the

issue otherwise.

create_

online_

revisio

stamp()

upgrade

version

`ironic.`

`ironic.`

ironic.cmd.status module

class i
Base
`osl`
`upg`
`Upg`

Up-
grad
chec
for
the
iron
statu
up-
grad
chec

com man

Up- grad chec shou be adde as sep- a- rate meth ods in this

class and added to _upgrade_checks tuple.

ironic.

Module contents

ironic.common package

Subpackages

ironic.common.glance_service package

Submodules

ironic.common.glance_service.image_service module

class i

Base obj

call (*m*

Call a glan clier meth

If we get

a connection error, retry the reques according to CONF.num_retries.

Paramet

- **met** The meth reques to be calle
- **arg** A list of positiona argumen for the meth called
- **kwa** A dict of keywor argumen for

the
meth
calle

Raises

Glan
Con
nec-
tion-
Faile

`downloa`

Call
out
to
Glan
for
data
and
write
data

Paramet

- **`ima`** The opac im- age iden ti- fier.
- **`dat`** (Op- tiona File ob- ject to write data to.

`show` (*im*

Re- turns a dict with image data for the give opac im-

age id.

Paramet

ima

The opac im- age iden ti- fier.

Returns

A dict con- tain- ing im- age meta data

Raises

Im- ageN Foun

Raises

Imag U- nac- cept able if the im- age sta- tus is not

active

swift_t

Gen er-ate a no-auth Swi temporary URL

This function will generate (or return the cach one if temp URL cache is enabled) the temporary Swift URL using the image id from Glance and the config options: swift_endpoint_url, swift_api_version, swift_account and swift_container. The temporary URL will be valid for swift_temp_url_duration seconds. This allows Ironic to download a Glance image without passing around an auth_token.

Parame

`ima` The return from a GET reques to Glan for a certain image_id. Should be a dictionary, with keys like name and checksum. See https://docs.openstack.org/glance/latest/user/glanceapi.html for examples.

Returns

A sign Swi URL from whic

an image can be dow
load
without authentication.

Raises
Invalic Parameter-Valu if Swif config options
are not set correctly.

Raises
Miss ing-Parameter-Valu if a required parameter is not set.

Raises
Imag U-nac-cept able if the image info from

Glan
does not have an image ID.

class i

Base
`tup`

url
Alia for field number 0

url_exp
Alia for field number 1

`ironic.`
Creates a glan clien if does exists and calls the function.

ironic.common.glance_service.service_utils module

`ironic.`

`ironic.`
Chec the image

sta-
tus.

This
chec
is
need
in
case
the
Glan
im-
age
is
stuck
in
queu

status or pending_delete.

`ironic.`

Chec
im-
age
avai
abil-
ity.

This
chec
is
need
in
case
Nov
and
Glan
are
de-
ploy
with
out

authentication turned on.

`ironic.`

Pars
an
im-
age
id
from
im-

age href

Paramet

ima href of an image

Returns

image id pars from image_

Raises

Inv whe input image href is invali

`ironic.`

Module contents

Submodules

ironic.common.args module

`ironic.`

Validate that every supplied validator pass

The valu re- turn from each val- ida- tor is pass as the valu

to the next one.

Paramet

- **nam** Nam of the ar- gu- men
- **val** A valu

Returns The valu trans form thro ev- ery sup- plied val- ida- tor

Raises The er- ror

from the first faile validator

ironic.

Validate that the valu is a strin representing

a boolean

Paramet

- **nam** Nam of the argumen
- **val** A strin valu

Returns The bool representation of the valu or

Non
if

value is None

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
the
valu
can-
not
be

converted to a boolean

`ironic.`
Re-
turn
a
val-
ida-
tor
func
tion
whic
val-
i-
date
dict

fields

Val-
ida-
tors
will
re-
plac
the
valu
with
the
val-
i-
da-
tion

result. Any dict item which has no validator is ignored. When a key is missing in the value then the corresponding validator will not be run.

Param
validators dict whe the key is a dict key to val-

idate and the value is a validator function to run on that value

Returns
validator func tion whic take nam and valu argumen

`ironic.`

Validate that the valu representa an integer

Paramet

- **nam** Nam of

the argumen

- **val** A valu representing an integer

Returns
The valu as an int, or Non if valu is Non

Raises
InvalidParameterValu if the valu does not represent an integer

`ironic.`
Validate that the

valu rep-re-sent a MA ad-dres

Paramet

- **nam** Nam of the ar-gu-men
- **val** A strin valu rep-re-sent-ing a MA ad-dres

Returns
The valu as a nor-mal-ized MA ad-dres or Non if value is None

Raises
In-vali

Parameter-Valu if the valu is not a

valid MAC address

`ironic.`

Validate that the valu is a logical nam

Paramet

- **nam** Nam of the argumen
- **val** A logical nam strin valu

Returns The valu or Non

if valu is Non

Raises
In-valic Pa-ram-e-ter-Valu if the valu is not a

valid logical name

`ironic.`

Val-i-date if at least one sup-plied val-ida-tor pass

Paramet

- **nam** Nam of the ar-gu-men
- **val** A valu

Returns

The valu re- turn from the first successful validator

Raises
The error from the last validator whe every val-

idation fails

`ironic.`

Val- i- date a patc API operation

`ironic.`
Return a validator

func-
tion
whic
val-
i-
date
the

value with jsonschema

Param

sche
dict
representing
json
sche
to
validate
with

Returns

validator
func-
tion
whic
take
nam
and
valu
arguments

`ironic.`

Validate
that
the
valu
is
a
strin

Paramet

-

nam
Nam
of
the
argumen

- **val**
A
strin
valu

Returns
The
strin
valu
or
Non
if
valu
is
Non

Raises
Invalic
Parameter-
Valu
if
the
valu
is
not
a

string

`ironic.`
Validate
and
convert
com
delimited
strin

to a list.

Paramet

- **nam** Nam of the argu- men
- **val** A com sep- a- ratec strin of val- ues

Returns
A list of uniq val- ues (low case main- tain- ing the same order, or None if value is None

Raises
In- valic Pa- ram- e- ter- Valu if the

valu is not a

string

`ironic.`

Return a validator func tion whic chec the valu is

one of the types

Param

type one or more type to use for the isinstanc test

Returns

validator func tion whic take nam and valu argumen

`ironic.`

Val-

i-
date
that
the
valu
is
a
UUI

Paramet

- **nam**
  Nam
  of
  the
  argument
- **val**
  A
  UUI
  strin
  valu

Returns
The
valu
or
Non
if
valu
is
Non

Raises
Invalid
Param-
eter-
Valu
if
the
valu
is
not
a

valid UUID

ironic.

Validate that the value is a UUI or logical

name

Paramet

- **nam** Nam of the argument
- **val** A UUI or logical nam strin valu

Returns The valu or Non if valu is Non

Raises Invalid Pa-

ram-
e-
ter-
Valu
if
the
valu
is
not
a

valid UUID or logical name

ironic.

Dec-
o-
ra-
tor
whic
val-
i-
date
and
trans
form
func
tion

arguments

ironic.common.boot_devices module

Map
ping
of
boot
de-
vice
used
whe
re-
ques
ing
the
sys-
tem

to boot from an alternate device.

The
op-
tions
pre-
sent

were base on the IPMItool chassis bootdev command. You can find the documentation at: http://linux.die.net/man/1/ipmitool

NOT This module does not includ all the options from ipmitool because they dont make sense in the limited context of Ironic right now.

`ironic.`
Boo into BIO setu

`ironic.`
Boo from CD/

`ironic.`
Boo from default Hard driv

`ironic.`
Boo from a flopp driv

`ironic.`

Boo
from
iSCS
volume

`ironic.`

Boo
from
PXE
boot

`ironic.`

Boo
from
default
Hard
driv
reques
Safe
Moc

`ironic.`

Boo
from
Wid
Area
Networ

ironic.common.boot_modes module

Map
ping
of
boot
mod
used
whe
reques
ing
the
system
to

boot using alternative firmware interfaces.

The
op-

tions presente were base on the Red fish pro-to-col

capabilities, specifically on the BootSourceOverrideMode property.

`ironic.`

Boo over lega PC BIO firm interface

`ironic.`

Boo over Unified Extensible Firm Interface (UE

firmware interface

ironic.common.checksum_utils module

`ironic.`

Com pute chec sum by give image path and algorith

Paramet

- **ima** The path to the file to underg chec sum calculation.
- **alg** The chec sum algorith to utilize. Defaul to md5 due to historical support reasons in Ironic.

Returns
The calculate chec sum valu

Raises
ValueEr

ror whe the chec sum al- go- rithr is not sup-

ported by the system.

`ironic.`

Get and re- turn the im- age chec sum and algo

Paramet

`ins`

The node in- stanc info or new up- date in- stanc valu

Returns

A tu- ple con- tain- ing two val- ues, a chec sum and

algorithm, if available.

ironic.

Gets a checksum value based upon a remote checksum URL

file.

Paramet

- **che** The URL to the checksum URL content.
- **ima** The image source utilized to match with the contents

of the URL payload file.

Raises

Im-

age-
Dow
load
Fail
whe
the
chec
sum
file
can-
not
be

accessed or cannot be parsed.

`ironic.`

Iden
tify
if
chec
sum
is
not
a
url.

Paramet

`che`

The
user
sup-
plied
chec
sum
valu

Returns

True
if
the
chec
sum
is
a
url,
oth-
er-
wise
Fals

Raises

Im-
age
sum

Not-Sup-port should the con-duc-tor have this

support disabled.

`ironic.`

Val-i-date im-age chec sum

Paramet

- `pat` File path in the form of a strin to cal-cu-late

a checksum which is compared to the checksum field.

- `che` The sup-plied chec sum valu a strin whic will be com

pared to the file.

- **che** The chec sum type of the al- go- rithr

Raises Im- age(sum ror if the sup- plied data can- not be pars

or if the supplied value does not match the supplied checksum value.

ironic.common.cinder module

ironic.

At- tach vol- ume to a node

Enu- mer- ate thro the pro- vide list of volume and attach the volumes to the node defined in the task utilizing the provided connector information.

If an attachmen appear to alread exist, we will skip attempting to attach the volume. If use of the volume fails, a user may need to remove any lingering pre-existing/unused attachment records since we have no way to validate if the connector profile data differs from what was provided to cinder.

Paramet

- **`tas`** Task ager instan representing the operation.
- **`vol`** List of volume UUI values

representing volumes.

- **con**
  Dictionary object representing the node sufficiently to attach a volume. This value can vary based upon the nodes configuration, capability, and ultimately the back-end storage driver. As cinder was designed around iSCSI, the ip and initiator keys are generally expected by cinder drivers. For FiberChannel, the key wwpns can be used with a list of port addresses. Some drivers support a multipath boolean key, although it is generally False. The host key is generally used for logging by drivers. Example:

```
↪'],
```

(continues on next page)

(continued from previous page)

Raises

Stor ageE ror If storage subsystem exception is raised.

Returns

List of connect volume including volume that were already connected to desired nodes. The returned list can be relatively consistent depending on the end storage driver that the volume is configured for, however the driver_volume_type key should not be relied upon as it is a free-form value returned by the driver. The accompanying data key contains the actual target details which will indicate either target WWNs and a LUN or a target portal and IQN. It also always contains volume ID in cinder and ironic. Except for these two IDs, each driver may return somewhat different data although the same keys are used if the target is FC or iSCSI, so any logic should be based upon the returned contents. For already attached volumes, the structure contains already_attached: True key-value pair. In such case, connection info for the node is already in the database, data structure contains only basic info of volume ID in cinder and ironic, so any logic based on that should retrieve it from the database. Example:

```
↪'],
```

(continues on next page)

(continued from previous page)

```
↪000002',
```

(continues on next page)

(continued from previous page)

```
↪',
```

(continues on next page)

(continued from previous page)

```
↪',
```

`ironic.`

Detach a list of volumes from a

provide connector detail.

Enumerates through a provided list of volumes and issues detachment requests utilizing the connector information that describes the node.

Paramet

- **`tas`** The Task task representing the reques
- **`vol`** The list of volume id values to detach
-

con

Dictionary object representing the node sufficiently to attach a volume. This value can vary based upon the nodes configuration, capability, and ultimately the back-end storage driver. As cinder was designed around iSCSI, the ip and initiator keys are generally expected. For FiberChannel, the key wwpns can be used with a list of port addresses. Some drivers support a multipath boolean key, although it is generally False. The host key is generally used for logging by drivers. Example:

```
↪']
```

(continues on next page)

(continued from previous page)

- **all** Boo valu governing if errors that are re turn

are treated as warnings instead of exceptions. Default False.

Raises
Stor ageE ror

`ironic.`
Get a cinder clien connection.

Paramet

- **con** request context, instance of ironi

- **aut** (boo When True use auth values from conf parameters

Returns
A cinder clien

ironic.

Chec if a volume is attach to the supplied node

Paramet

- **nod** The object representing the node
- **vol**

The object representing the volume from cin-

der.

Returns

Boo in- di- cat- ing if the vol- ume is at- tach Re-

turns True if cinder shows the volume as presently attached, otherwise returns False.

`ironic.`

Che if a vol- ume is avai able for a con- nec- tion.

Paramet

`vol`

The ob- ject rep- re- sent

ing the vol- ume

Returns

Boo if vol- ume is avai able

ironic.common.components module

Map ping of com mon hard ware com po- nent of a com pute

system.

`ironic.`

Cha sis en- clos- ing one or mor hard ware com po- nent

`ironic.`

Stor age driv

`ironic.`
Network interface

`ironic.`
Pow supply unit

`ironic.`
Com puting system

ironic.common.config module

`ironic.`

ironic.common.context module

class `i`

Base `osl` `con` `Req`

Extend security contexts from the oslo librar

ensure_
Ensure

threa ing contains context

For asyn tasks the context of local threa is miss ing. Set

it with request context and this is useful to log the request_id in log messages.

to_poli

A dictionary of context attribu to enforce pol-

icy with.

oslo enforcemen requir a dictionary of attribu representing the current logged in user on which it applies policy enforcement. This dictionary defines a standard list of attributes that should be available for enforcement across services.

It is expected that services will often have to override this method with either deprecated values or additional attributes used by that service specific policy.

`ironic.`

Create an administrator context.

ironic.common.dhcp_factory module

class `i`

Base `obj`

clean_d

Clea up the DHC BOO options for this node

Paramet

tas

A Tas agen instan

propert

update_

Sen or update the DH BO options for this nod

Paramet

- **tas** A Tas agen instan
- **dhc** this will be a list of dict e.g.

```
```

(continues on next page)

(continued from previous page)

- **por** A dict with keys port and port- grou and dicts as values. Each dict has key/value pairs of the form <ironic UUID>:<neutron port UUID>. e.g.

(continues on next page)

(continued from previous page)

If the valu is Non will get the list of port from the Iron

port/portgroup objects.

ironic.common.driver_factory module

class i

Base obj

Discover load and manage the drive available

This is subclass to

load both main drive and extra interfaces.

get_dri

items()

Iterator over pairs (nan instanc

propert

The list of drive nam avail able

class i

Base *iro com dri Bas*

class i

Base *iro com dri Bas*

class i

Base *iro com dri Bas*

`ironic.`

Get all interfaces for all interface type

Returns

Dictionary mapping interface type to dictionary

mapping interface name to interface object.

`ironic.`

Buil a composable driv for a give task

Star with a *Bare* object, and attach implemen tations

of the various driver interfaces to it. They come from separate driver factories and are configurable via the database.

Paramet

tas

The task containing the node to buil a driv for.

Returns

A driv object for the task

Raises

Driv Not-Foun if node coul not be foun in the iron nam

paces.

Raises

Interface Foun nEntry-poin

if som node in- ter- face

are set to invalid or unsupported values.

Raises

In- com pat- i- bleIr ter- face the re- ques im- ple- men

tation is not compatible with it with the hardware type.

`ironic.`

En- sure that node in- ter- face (e.g. for cre- atior or up-

dating) are valid.

Up- date (but does save to the data hard ware in- ter-face with

calculated defaults, if they are not provided.

This func tion is run on node updating and creation as

well as each time a driver instance is built for a node.

Paramet

- **nod** node object to chec and potentially update
- **hw_** hardware type instan object; will be detecte from

node.driver if missing

Returns

True if any char were mad to the node otherwise Fals

Raises
InterfaceFounEntrypoin on validation failure

Raises
NoVali DefaultForI terface if the default valu cannot be calculated and is not provided in the configuration

Raises
Driv NotFour if the node

hard
ware
type
is
not
foun

ironic.

Cal-
cu-
late
and
re-
turn
the
de-
fault
in-
ter-
face
im-

plementation.

Find
the
first
im-
ple-
men
ta-
tion
that
is
sup-
port
by
the

hardware type and is enabled in the configuration.

Paramet

- **hw_**
hard
ware
type
in-
stan
ob-
ject.

- **int** type of the interface (e.g. boot
- **dri** entrypoint nam of the hw_ object. Is used for

exception message.

- **nod** the identifier of a node If specified, is

used for exception message.

Returns
an entrypoint nam of the calcu-

lated
de-
fault
im-

plementation.

Raises
In-
ter-
face
Foun
nEn-
try-
poin
if
the
en-
try
poin
was

not found.

Raises
No-
Vali
De-
fault
ForI
ter-
face
if
no
de-
fault
in-
ter-

face can be found.

`ironic.`
Get
us-
able
in-
ter-
face
for
a
give
hard
ware
type

For

a give hard ware type find the intersection of en-

abled and supported interfaces for each interface type. This is the set of interfaces that are usable for this hardware type.

Paramet

har

The hard ware type object to sear

Returns

a dict map ping inter face type to a list of en-

abled and supported interface names.

ironic.

Get a hard ware type instance by nam

Paramet

har

the nam of the hard ware type to find

Returns

An in- stan of iron

Raises

Driv Not- Fou if re- ques hard ware type can- not be foun

`ironic.`

Get in- ter- face im- ple- men ta- tion in- stan

For hard ware type also

validate compatibility.

Paramet

- **hw_**
  a hardware type instan
- **int**
  nam of the interface type (e.g. boot
- **int**
  nam of the interface implementation from

an appropriate entry point (ironic.hardware.interfaces.<interface type>).

Returns
instan of the

re-
ques
in-
ter-
face
im-
ple-
men
ta-

tion.

Raises

In-
ter-
face
Foun
nEn-
try-
poin
if
the
en-
try
poin
was

not found.

Raises

In-
com
pat-
i-
bleIn
ter-
face
if
hw_
is
a
hard
ware

type and the requested implementation is not compatible with it.

`ironic.`

Get
all
hard
ware
type

Returns

Dic-
tio-

nary map ping hard ware type nam to hard ware type

object.

`ironic.`

Get all interfaces for a give interface type

Paramet

`int` Strir type of interface to fetcl for.

Returns Dictionary map ping interface nam to interface

object.

ironic.common.exception module

Iron specific exceptions list.

excepti

Base `iro` `exc` `Iro`

excepti

Base `iro` `exc` `Iro`

excepti

Base `iro` `exc` `Iro`

excepti

Base `iro` `exc` `Iro`

excepti

Base *`iro`* *`com`* *`exc`* *`Con`*

excepti

Base *`iro`* *`com`*

exc
Con

excepti

Base
irc
exc
Irc

excepti

Base
irc
com
exc
Not

excepti

Base
irc
com
exc
Con

excepti

Base
irc
com
exc
Not

excepti

Base
irc
com
exc
Not

excepti
Base
irc
exc
Irc

excepti

Base
irc
exc
Irc

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Not

excepti

Base
Run

propert

excepti

Base
iro
exc
Iro

excepti

Base
iro
exc
Iro

excepti

Base
iro
exc
Iro

excepti

Base
iro

com
exc
Not

excepti

Base
iro
exc
Iro

excepti

Base
iro
exc
Iro

excepti
Base
iro
exc
Iro

code =

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Con

excepti

Base
iro
exc
Iro

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Not

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Dri

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Not

excepti

Base

iro
com
exc
Dri

excepti

Base
iro
exc
Iro

excepti
Base
iro
exc
Iro

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Not

excepti

Base
iro
exc
Iro

excepti

Base
iro
exc
Iro

excepti

Base
iro
exc
Iro

excepti

Base
`iro`
`exc`
`Iro`

excepti

Base
`iro`
`exc`
`Iro`

excepti

Base
`iro`
`exc`
`Iro`

excepti

Base
`iro`
`exc`
`Iro`

excepti
Base
`iro`
`exc`
`Iro`

excepti

Base
`iro`
`exc`
`Iro`

excepti

Base
iro
com
exc
Not

excepti

Base
iro
com
exc
Not

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
IBM

excepti

Base
iro
com
exc
Dri

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Dri

excepti

Base
iro
com
exc
Dri

excepti

Base
iro
com
exc
Dri

excepti

Base
iro
com
exc

Dri

excepti

Base *irc* *com* *exc* *Inv*

Cannot load the reques or requir chec sum algorith

excepti

Base *irc* *com* *exc* *Inv*

Exception indicating chec sum faile to matc

excepti

Base *irc* *com* *exc* *Inv*

An

OS- Error was raise whe trying to read the file.

code =

excepti

Base *irc com exc Inv*

Exception indicating we cannot support the remote checksum file.

excepti

Base `irc exc Irc`

excepti

Base `irc exc Irc`

excepti

Base
irc
exc
Irc

excepti

Base
irc
com
exc
Not

excepti

Base
irc
com
exc
Not

excepti

Base
irc
exc
Irc

excepti

Base
irc
exc
Irc

excepti

Base
irc
exc
Irc

excepti

Base
irc
com
exc
Inv

excepti

Base
irc
com

exc
Inv

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Con

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Not

excepti

Base
iro
exc
Iro

excepti

Base
iro
exc
Iro

excepti

Base
iro
exc
Iro

excepti

Base

iro
exc
Iro

excepti

Base
iro
com
exc
Inv

excepti
Base
iro
exc
Iro

code =

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
exc
Iro

excepti

Base

irc
exc
Irc

excepti

Base
irc
exc
Irc

excepti

Base
irc
com
exc
Inv

excepti

Base
irc
com
exc
Ima

excepti

Base
irc
com
exc
Inv

excepti

Base
irc
com
exc
Cli

propert

excepti

Base
irc
com
exc
Inv

excepti

Base

iro
com
exc
Inv

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com

exc
Inv

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
exc
Iro

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Con

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Tem

code =

excepti

Base
iro
com
exc
Tem

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Inv

excepti
Base
iro
com
exc
Not

excepti

Base
iro
com
exc

Con

excepti

Base
iro
com
exc
Inv

excepti

Base
`iro`
`exc`
`Iro`

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Inv

excepti
Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Not

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
HTT

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Not

excepti

Base
iro
exc
Iro

code =

excepti

Base
iro
exc
Iro

code =

excepti

Base
irc
exc
Irc

code =

excepti

Base
irc
exc
Irc

excepti

Base
irc
exc
Irc

excepti

Base
irc
exc
Irc

excepti

Base
irc
com
exc
Not

excepti

Base
irc
exc
Irc

excepti

Base
irc
com
exc
Inv

excepti

Base
`iro`
`exc`
`Iro`

excepti

Base
`iro`
`com`
`exc`
`Con`

excepti

Base
`iro`
`com`
`exc`
`Con`

excepti

Base
`iro`
`com`
`exc`
`Not`

excepti

Base
`iro`
`com`
`exc`
`Con`

excepti

Base
`iro`
`com`
`exc`
`Con`

excepti

Base
`iro`
`com`
`exc`
`Con`

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Not

excepti

Base
iro
exc
Iro

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Rec

excepti

Base
iro
com
exc
Dri

excepti
Base
iro
com
exc
Dri

excepti

Base
iro

`exc`
`Irc`

excepti

Base
`irc`
`exc`
`Irc`

excepti

Base
irc
com
exc
Swi

excepti

Base
`irc`
`exc`
`Irc`

excepti

Base
`irc`
`exc`
`Irc`

code =

excepti

Base
`irc`
`exc`
`Irc`

excepti
Base
irc
com
exc
Cli

propert

excepti

Base

iro com exc Cli

add_fie

Add a field nam to concatenate the full nam

Add a field nam so that the who hierarchy is displayed. Successive calls to this method will prepend `name` to the hierarchy of names.

propert

excepti

Base *iro com exc Inv*

excepti

Base `iro exc Iro`

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Inv

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Not

excepti

Base
iro
com
exc
Con

excepti

Base
iro
com
exc
Con

excepti

Base
irc
com
exc
Cor

excepti

Base
irc
com
exc
Not

excepti

Base
`irc`
`exc`
`Irc`

ironic.common.faults module

Faul definitions

`ironic.`
Nod is mov to main tenanc due to failure of a cleaning operation.

`ironic.`
Nod is mov to main te-

nanc
due
to
pow
syn-
chro
niza

tion failure.

ironic.

Nod
is
mov
to
main
te-
nanc
due
to
fail-
ure
of
clea

ing up during rescue abort.

ironic.common.fsm module

class i

Base
aut
mac
Fin

An
iron
state
mac
class
with
som
iron
spe-
cific
ad-
di-
tions

add_sta

Add
a

give state to the state machin

Paramet

- `sta` Use this to specify that this state is a stable/p state. A state must have been previously defined as stable before it can be used as a target
- `tar` The target state for state to go to. Before a state can be used as a target it must have been previously added and specified as stable

Further argumen are interpret as

for parent meth

`add_state`.

add_tra

Add an allowe transition from start -> end for the given event.

Parame

- **sta** starting state
- **end** ending state
- **eve** even that caus start state to transition to end state
- **rep**

replace existing event instead of raising a

`Duplicate` exception when the transition already exists.

initial

Initialize the FSM

Parame

- **sta** the FSM is initialized to start from this state
- **tar** if specified, the FSM is initialized to this

target state. Otherwise use the default target state

is_stab

Is the state stable?

Parame

sta the state of interest

Raises

Invali Stat if the state is invali

Returns

True if it is a stable state Fals otherwise

process

process the even

Parame

- **eve** the even

to be processed

- **tar** if specified, the final target state for the

event. Otherwise, use the default target state

propert

ironic.common.hash_ring module

class i

Base `obj`

get_rin

classme

propert

ironic.common.i18n module

oslo integration module.

See https

// docs open org/ oslo i18n lates user

ironic.common.image_format_inspector module

This is a pyth implementation of virtual disk format inspection routines gathered from various public specification documents, as well as qemu disk driver code. It attempts to store and parse the minimum amount of data required, and in a streaming-friendly manner to collect metadata about complex-format images.

class i

Base obj

Rep represent a region of a file we wan to capture.

A region

of a file we wan to cap- ture re- quir a

byte offset into the file and a length. This is expected to be used by a data processing loop, calling capture() with the most recently-read chunk. This class handles the task of grabbing the desired region of data across potentially multiple fractional and unaligned reads.

Paramet

- **off** Byte off- set into the file start ing the re- gion
- **len** The leng of the the re- gion

capture

Pro- cess a chun of data

This shou be calle for

each chun in the read loop at least until complete returns True.

Paramet

- **`chu`** A chun of byte in the file
- **`cur`** The position of the file process by the read loop so far. Note that this will be the position in the file *after* the chunk being presented.

`propert`

Returns True whe we have capture the desire data

class i

Base

obj

A stream-based disk image inspector.

This base class works on raw images and is subclassed for more complex types. It is to be presented with the file to be examined one chunk at a time, during read processing and will only store as much data as necessary to determine required attributes of the file.

propert

Returns the total size of the file, usually smal than virtual_size.

NOT this will only be accurate if the entire file is read and processed.

`propert`

Return True if we have all the information need

`propert`

Return info on amo of data held in memory for auditing.

This is a dict of region item that the inspec tor uses

to examine the file.

eat_chu

Call this to pres chun of the file to the inspec tor.

propert

Returns True if the file ap pear to be the ex pect

format.

classme

Read as muc of a file as neces sary to com plete

inspection.

NOT Becaus we only read as muc of the file as nec-es-

sary, the actual_size property will not reflect the size of the file, but the amount of data we read before we satisfied the inspector.

Rais Im-age-For-matl ror if we can-not pars the file.

has_reg

Re-turn True if nam re-gion has been de-fined

new_reg

Add a new Cap-tur-eRe-gion by nam

captured by the defined regions. If any regions are defined by this call, those regions will be presented with the current chunk in case it is within one of the new regions.

post_pr

Post read hool to process wha has beer read so far.

This will be calle after each chun is read and potentiall

region

Get a CaptureRegion by nam

safety_

Perform som chec to determin if this file is safe

Return True if safe Fals otherwise It may raise ImageFormatError if safety cannot be guaranteed because of parsing or other errors.

propert

Return the virtual size of the disk image, or zero if unknown.

class i

Base *iro com ima Fil*

ISO 966 and UDI format

we neec to chec the

first 32K + descrip tor size to look

for the ISO 9660 or UDF signature.

http: // wiki osde org/ ISO 966 http: // wiki osde org/ UDI mk-

isofs help | grep udf

The Universal Disc Format or UDI is the files tem used

on DVDs and Blu-Ray discs.UDF is an extension of ISO 9660 and shares the same header structure and initial layout.

Like the CDI 966 file system, the UDI

file system uses a

2048 byte sector size, and it designates that the first 16 sectors can be used by the OS to store proprietary data or boot logic.

That mea we need to chec the first 32K + descrip tor size

to look for the ISO 9660 or UDF signature. both formats have an extent based layout, so we cant determine ahead of time where the descriptor will be located.

fortunate the ISO 966 and UD formats have a Primar

Volume Descriptor located at the beginning of the image, which contains the volume size.

`propert`

Returns True if the file appear to be the expect

format.

propert

Returns the virtual size of the disk image, or zero

if unknown.

excepti

Base `Exc`

An unrecoverable image format error that abor

the process.

class i

Base `obj`

A file-like object that wrap another and update a format inspector.

This pass chur to the format inspector whil reading. If the inspector fails, it logs the error and stops calling it, but continues proxying data from the source to its user.

close()

read(*siz*

class i

Base *iro com ima Fil*

propert

Return True if the file appear to be the expect format.

safety_

Perform some chec to determine if this file is safe

Returns True if safe Fals otherwise It may raise Image-

FormatError if safety cannot be guaranteed because of parsing or other errors.

class i

Base *irc com ima Fil*

QEM QCC Format

This shou only required abou 32 byte of the be-the file to determine the virtual size, and 104 bytes to perform the safety check.

format.

gin-
ning
of

BF_OFFS

BF_OFFS

I_FEATU

I_FEATU

I_FEATU

I_FEATU

propert

Re-
turns
True
if
the
file
ap-
pear
to
be
the
ex-
pect

propert

propert

propert

propert

safety_

Per-
form
som
chec
to

determin
if
this
file
is
safe

Returns
True
if
safe
Fals
otherwise
It
may
raise
Image-

FormatError if safety cannot be guaranteed because of parsing or other errors.

propert

Returns
the
virtual
size
of
the
disk
image,
or
zero

if unknown.

class i

Base

`obj`

A
logg
like
thing
that
swallows
trac-

ing
whe
we
do
not
wan

it.

debug(*

error(*

info(*a

warning

class i

Base
irc
com
ima
Fil

Vir-
tu-
al-
Box
VDI
for-
mat

This
only
neec
to
store
the
first
512
byte
of
the
image.

propert

Returns
True
if

the file appear to be the expect

format.

propert

Returns the virtual size of the disk image, or zero

if unknown.

class i

Base

irc com ima Fil

Connectix/M VPC VHI Format

This shou only require about 512 byte of the beginning of

the file to determine the virtual size.

propert

Returns True if the file appears to be the expect

format.

propert

Returns the virtual size of the disk image, or zero

if unknown.

class i

Base

irc
com
ima
Fil

MS VHI Format

This requir som com

plex pars ing of the strea The first 256l

of the image is stored to get the header and region information, and then we capture the first metadata region to read those records, find the location of the virtual size data and parse it. This needs to store the metadata table entries up until the VDS record, which may consist of up to 2047 32-byte entries at max. Finally, it must store a chunk of data at the offset of the actual VDS uint64.

METAREC

VHDX_ME

VIRTUAI

propert

Re- turns True if the file ap- pear to be the ex- pect

format.

post_pr

Post read hool to pro- cess wha has been read so far.

This

will be calle after each chun is read and potentiall

captured by the defined regions. If any regions are defined by this call, those regions will be presented with the current chunk in case it is within one of the new regions.

propert

Returns the virtual size of the disk image, or zero

if unknown.

class i

Base *irc com ima Fil*

vmw VMI format (mon lithi Spar and strea timize variants only

This need to store the 512 byte head and the descrip tor region which should be just after that. The descriptor region is some variable number of 512 byte sectors, but is just text defining the layout of the disk.

DESC_MA

DESC_OF

GD_AT_F

propert

Returns True if the file appears to be the expect format.

post_pr

Post read hook to process what has been read so far.

This will be calle after each chun is read and potentiall

captured by the defined regions. If any regions are defined by this call, those regions will be presented with the current chunk in case it is within one of the new regions.

safety_

Perform som chec to determine if this file is safe

Returns True if safe Fals otherwise It may raise Image-

FormatError if safety cannot be guaranteed because of parsing or other errors.

propert

Turn

the virtual size of the disk image, or zero

if unknown.

`ironic.`

`ironic.`

Attempt to detect the format of a file.

This runs through a file one time running all the known inspec

tors in parallel. It stops reading the file once all of them matches or all of them are sure they dont match.

Paramet

`fil`

The path to the file

to in-spec

Returns

A For-math spec tor in-stanc matc ing the file.

Raises

Im-age-For-math ror if mul-ti-ple for-mats are de-tected.

`ironic.`

Re-turns a For-math spec tor class base on the give nam

Paramet

`for`

The nam of

the
disk
(raw
qcov
etc).

Returns
A
For-
matl
spec
tor
or
Non
if
un-
sup-
port

ironic.common.image_service module

class i
Base
`obj`

Pro-
vide
re-
triev
of
disk
im-
ages

abstrac
Dow
load
im-
age
to
spec
i-
fied
lo-
ca-
tion.

Paramet

- **ima**

Image reference

- **`ima`**
  File object to writ data to.

Raises
exception.

Raises
exception.

`abstrac`
Get dictionary of image properties.

Paramet
`ima`
Image reference

Raises
exception.

Returns
dictionary

of image properties. It has three of

them: size, updated_at and properties. updated_at attribute is a naive UTC datetime object.

abstrac

Validate image reference.

Paramet

ima Image reference

Raises exception.

Returns Information needed to further operate with an

image.

class i

Base

irc

com
ima
Bas

Provide retriev of disk images avai able locally on the

conductor.

downloa

Dow load image to specified location.

Parame

- **ima** Image reference
- **ima** File object to writ data to.

Raises
exception.
if
sour
image
file
does
exist.

Raises
exception.
if
exceptions
were
raise
whil
writing
to

file or creating hard link.

show (*im*

Get dictionary
of
image
prop
erties.

Paramet
ima
Image
reference

Raises
exception.

if image file speci- fied does ex- ist.

Returns
dictio- nary of image prop- er- ties. It has three of

them: size, updated_at and properties. updated_at attribute is a naive UTC datetime object.

validat
Val- i- date lo- cal image ref- er- ence

Parame
ima
Im- age ref- er- ence

Raises
ex- cep- tion. if sour

image file does exist.

Returns
Path to image file if it exists.

class i
Base
iro com ima Bas

Provide retriev of disk images using HTT

downloa
Download image to specified location.

Paramet

- **ima** Image reference
- **ima** File object to write data to.

Raises exception. if GET request returned response code not

equal to 200.

Raises exception. if: * IOError happened during file

write; * GET request failed.

static Down load

content and returns the response text.

Parame

ima Image reference

Raises exception. if GET request returned response code not

equal to 200.

Raises exception. if: * IOError happened during file

write; * GET request failed.

show (*im*

Get dic-

tio-
nary
of
im-
age
prop
er-
ties.

Parame
ima
Im-
age
ref-
er-
ence

Raises
ex-
cep-
tion.
if:
*
HEA
re-
ques
faile
*
HEA
re-
ques

returned response code not equal to 200; * Content-Length header not found in response to HEAD request.

Returns
dic-
tio-
nary
of
im-
age
prop
er-
ties.
It
has
three
of

them: size, updated_at and properties. updated_at attribute is a naive UTC datetime object.

validat
Val-
i-

date
HTT
image
reference

Paramet

- **ima**
  Image
  reference
- **sec**
  Specify
  if
  image_
  being
  validate
  shou
  not

be shown in exception message.

Raises
exception.
if
HEA
request
faile
or
return
response

code not equal to 200.

Returns
Respon

to HEA re- ques

`ironic.`

Get im- age ser- vice in- stanc to dow load the im- age.

Paramet

- **ima** Strin con- tain- ing href to get im- age ser- vice for.
- **cli** Glan clien to be used for dow load used only if im-

age_href is Glance href.

- **`con`** reques context, used only if image_ is Glan href

Raises
exception. if no image service can handle spec

ified href.

Returns
Instan of an image service clas that is able to

download specified image.

ironic.common.images module

Han-dling of VM disk im-ages

`ironic.`

Chec im-age for-mat con-sis-tenc

Params

The de-ter-mine im-age for-mat by nam

Params

Op-tion the ex-pect for-mat base upon sup-plied con-fig-uration values.

Params

A node object or Non implying image cach

Raises

InvalidIm age if the request image format is

not permitted by configuration, or the expected_format does not match the determined format.

`ironic.`

Get size of converte raw image.

The size of image converte to raw format can be grow ing up to the virtual size of the image.

Paramet

- **`pat`** path to the image file.
- **`est`** Whe to estimate the size by scaling the original size

Returns

For *estimate* return the size of the raw image file. For *estimate=True*, return the size of the original image scaled by the configuration value *raw_image_growth_factor*.

`ironic.`

Cre-

ates a boot ISO image for a node

Give the href for kernel, ram root partitions UUI and ker-

nel cmdline arguments, this method fetches the kernel and ramdisk, and builds a bootable ISO image that can be used to boot up the baremetal node.

Paramet

- **con** context
- **out** the absolute path of the output ISO file
- **ker** URI or glan

uuid of the kernel to use

- ram URI or glan uuid of the ram to use

- dep URI or glan UUI of the deploy ISO image to

extract EFI system partition image. If not specified, the *esp_image_href* option must be present if UEFI-bootable ISO is desired.

- esp URI or glan UUI of FAT form EFI system parti-

tion image containing the EFI boot loader (e.g. GRUB2) for each hardware architecture to boot. This image will be written onto the ISO image. If not specified, the *deploy_iso_href* option is only required for building UEFI-bootable ISO.

- **`ker`** a strin containing whit pace separate values kernel cmdline arguments of the form K=V or K (optional).
- **`bas`** URI or glan UUI of a to be use as an over ride of what should be retrieved for to use, instead of building an ISO bootable ramdisk.
- **`inj`** Map ping of local sour file path to thei location on the final ISO image.

Boot_m
the boot

mod in whic the deploy is to happen.

Raises
Im- age- Cre- ation Faile if cre- at- ing boot ISO faile

`ironic.`

Cre- ates an ESP im- age on the spec i- fied file.

Cop the pro- vide ker- nel, ram and EFI

system partition image (ESP) to a directory, generates the grub configuration file using kernel parameters and then generates a bootable ISO image for UEFI.

Paramet

- **`out`** the path to the file whe the iso image need to be created.
- **`ker`** the kernel to use.
- **`ram`** the ramd to use.
- **`dep`** deploy ISO image to extract EFI sys-

tem partition image from. If not specified, the *esp_image* option is required.

- **esp** FAT form EFI system partition image containing the EFI boot loader (e.g. GRUB2) for each hardware architecture to boot. This image will be embedded into the ISO image. If not specified, the *deploy_iso* option is required.
- **ker** a list of strin elemen being a strin like K=V or K or combination of them like K1=V1,K2,) to be added as the kernel cmdline.
- **inj** Map ping of local sour file path to thei location on the final ISO image.

Raises
Im-
age-
Cre-
atior
Faile
if
im-
age
cre-
atior
faile
whil
copy

ing files or while running command to generate iso.

`ironic.`

Cre-
ates
an
isoli
im-
age
on
the
spec
i-
fied
file.

Cop
the
pro-
vide
ker-
nel,
ram
to
a
di-
rec-
tory,
gen-
er-

ates the isolinux configuration file using the kernel parameters provided, and then generates a bootable ISO image.

Paramet

- **out** the path to the file whe the iso image need to

be created.

- **ker** the kernel to use.
- **ram** the ram to use.
- **ker** a list of strin elements being a strin like

K=V or K or combination of them like K1=V1,K2,) to be added as the kernel cmdline.

- **inj** Mapping of lo-

cal
sour
file
path
to
their
lo-
ca-

tion on the final ISO image.

Raises

Im-
age-
Cre-
ation
Fail
if
im-
age
cre-
ation
faile
whil
copy

ing files or while running command to generate iso.

`ironic.`

Cre-
ates
the
fat
fs
im-
age
on
the
de-
sired
file.

This
meth
copi
the
give
files
to
a
root
di-

rec-
tory
(op-
tiona

writes the parameters specified to the parameters file within the root directory (optional), and then creates a vfat image of the root directory.

Paramet

- **out**
The
path
to
the
file
whe
the
fat
fs
im-
age
need

to be created.

- **fil**
A
dict
con-
tain-
ing
ab-
so-
lute
path
of
file
to

be copied -> relative path within the vfat image. For example:

(continues on next page)

```
↪root'
```

(continued from previous page)

- **par** A dict containing key-valu pairs of parameters.
- **par** The filenam for the parameters file.
- **fs_** size of the vfat files tem in KiB

Raises
Image-Cre-atio

Faile
if
im-
age
cre-
ation
faile
whil
do-

ing any of filesystem manipulation activities like creating dirs, mounting, creating filesystem, copying files, etc.

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

Com
pata
bil-
ity
help
to
re-
turn
the
im-
age
size
in

MB.

`ironic.`

Re-
turn
the
val-
ues
of
sev-
eral
prop
er-
ties
of

an image

Paramet

- **con** con-text
- **ima** href of the im-age
- **pro** the prop er-ties who val-ues are re-quir This ar-gument is optional, default value is all, so if not specified all properties will be returned.

Returns a dict of the val-ues of the prop er-ties. A prop erty not on the glance metadata will have a value of None.

`ironic.`

`ironic.`

Returns the tmp url for a glan image.

Paramet

- **con** context
- **ima** the UUI of the image in glan

Returns the tmp url for the glan image.

`ironic.`

`ironic.`

`ironic.`

Find out if the image

is a partition image

or a whole disk image.

Paramet

- **ctx** an admin context
- **ins** a node instanc info dict

Returns
True for who disk images and Fals for partition im-

ages and None on no image_source or Error.

`ironic.`
Perform a safe chec on the sup-

plied image.

This method triggers the image format inspectors to both

identify the type of the supplied file and safety check logic to identify if there are any known unsafe features being leveraged, and return the detected file format in the form of a string for the caller.

Paramet

- **ima** A fully qualified path to an image which need

to be evaluated for safety.

- **nod** A Nod object, optional. When supplied logging indicates the node which triggered this issue, but the node is not available in all invocation cases.

Returns
a strin representing the the image type whic is used.

Raises
InvalidI age whe the supplied image is detecte as unsafe, or the image format inspector has failed to parse the supplied images contents.

ironic.common.indicator_states module

Mapping of the indicator LED state

`ironic.`
LED is

blink-
ing

`ironic.`
LED
is
off

`ironic.`
LED
is
on

`ironic.`
LED
state
is
not
know

ironic.common.keystone module

Cen-
tral
plac
for
han-
dling
Key-
ston
au-
tho-
riza-
tion
and
ser-
vice lookup.

`ironic.`
Loa
adap
from
op-
tions
in
a
con-
fig-
u-
ra-
tion
file

section.

The adap will be pass directl to keyston Ada and will override the values loaded from config. Consult keystoneauth1 docs for available adapter options.

Paramet

gro nam of the config section to load adap option from

`ironic.`

Load auth plugin from options in a configuration file section.

The auth will

be pass directl to keyston auth plugin and

will override the values loaded from config. Note that the accepted kwargs will depend on auth plugin type as defined by [group]auth_type option. Consult keystoneauth1 docs for available auth plugins and their options.

Paramet

`gro` nam of the config section to load auth plugin

options from

`ironic.`

Get an endpoin from an adap

The adap will be pass directl to keyston Ada and will over

ride the values loaded from config. Consult keystoneauth1 docs for available adapter options.

Paramet

gro

nam of the config section to load adap options

from

Raises

Catalog-Not-Foun if the endpoin is not foun

`ironic.`

Create auth plugin wrap ping both user and service auth

Whe properly configured and using auth middleware requests with valid service auth will not fail if the user token is expired.

Ideally we wou use the plugin provide by auth middleware however this plugin isnt serialized yet.

Paramet

- `con` The Reques Con text instan from whic the user auth is extracted.
- `end` The reques end-

poin to be utilized

- **ser** The service authentication credentials to be

used.

- **onl** Boo default Fals When set to True the resulting

Service token pair is generated as if it originates from the user itself. Useful to cast admin level operations which are launched by Ironic itself, as opposed to user initiated requests.

Returns
Returns a service token via the Service Toke-

nAuthWrapper class.

`ironic.`

Loa session object from options in a configuration file section.

The session will be pass directl to keyston Session and will override the values loaded from config. Consult keystoneauth1 docs for available options.

Paramet

`gro` nam of the config section to load session options from

`ironic.`

Wra keyston func tions and centralizes exception handling.

ironic.common.molds module

`ironic.`

Gets configuration mold from indicated location.

Paramet

- **tas** A Task agen instan
- **url** URI of the configuration

item to get.

Returns
JSO configuration mol

Raises

- **Iro** If using Swi storage and no authentication token found in tasks context.
- **HTT** If faile to complet HTT reques

`ironic.`
Stor configuration mol to indi-

cate
location.

Paramet

- **tas** A Task ager instanc
- **nam** URL of the configuration item to save to.
- **dat** Con tent of JSO data to save

Raises

- **Iro** If using Swi storage and no

authentica-

tion token found in tasks context.

- **HTT**
  If faile to com- plete HTT re- ques

ironic.common.network module

`ironic.`

Get all VIF ids for a node

This func- tion does not han- dle mult node op- er- a- tions

Paramet
tas a Task agen in- stan

Returns

A dict of Nod neutron port whe keys are port & port grou

and the values are dict of UUIDs and their associated VIFs, e.g.

`ironic.`

Return the set of phys ical networl associated with a portgroup.

Paramet

- **tas** a Task ager instanc
- **por** ID of the portgrou
- **exc** A Port object to exclud fron the determination of the portgroups physical network, or None.

Returns
The set of physical networl associated

with

the portgroup. The set will contain zero or one physical networks.

Raises

Port
grou
Phys
net-
Inco
sis-
tent
if
the
port-
grou
ports
are

not assigned the same physical network.

`ironic.`

Re-
turn
the
set
of
phys
i-
cal
net-
work
for
a
node

Re-
turns
the
set
of
phys
i-
cal
net-
work
as-
so-
ci-
ated

with a nodes ports. The physical network None is excluded from the set.

Paramet

`tas`
a

Task ager instanc

Returns
A set of physi cal networ

`ironic.`
Look a portgrou by ID on a task object.

Paramet

- **tas** a Task ager instanc
- **por** ID of the portgrou

Returns
A Port grou object or

Non

`ironic.`

Look port by their port-grou ID on a task ob- ject.

Paramet

- **tas** a Task agen in- stan
- **por** ID of the port-grou

Returns

A list of Port ob- jects

`ironic.`

Re- mov all vif at- tach men reco from a

node

Paramet

tas

a Task ager instan

ironic.common.neutron module

class i

Base

`obj`

get_cle

get_ins

get_pro

get_res

validat

Validate that the node has required properties for

inspection.

Paramet

tas

A Task ager instance with the

node being chec

Raises
Missing-Param-eter-Valu if node is miss-ing one

or more required parameters

Raises
Unsup-port-ed-Driv-ten-sion

`ironic.`
Nam of the neu-tron net-work API phys-i-cal net-work

parameter.

`ironic.`

Cre-ate neu-tron port

to boot the ram

Create neutron port for each pxe_ port on task to boot the ramdisk.

If the config option neutron is set, neutron port for non-pxe-enabled ports are also created these neutron ports will not have any assigned IP addresses.

Paramet

- **tas** a Task agen instan
- **net** UUI of a neu-

tron net- worl whe port will be cre-

ated.

- **sec** List of Se- cu- rity Grou UUI to be used for net-

work.

Raises
Net- worl Er- ror

Returns
a dic- tio- nary in the form {por neu- tron

ironic.

Re- triev a neu- tron clien connection.

Paramet

- **con** request context, instance of ironi
- **aut** (boo When True use auth values from conf parame-

ters

Returns
A neutron clien

ironic.

Extract the portgrou information.

The information

tion
is
re-
turn
in
the
form
of:

```
```

(continues on next page)

(continued from previous page)

```
↪',
```

Paramet

- **`tas`** a task containing the Nod object.
- **`por`** Iron portgrou object to extract data for.

Returns
port

grou information as a dict

`ironic.`

Gath Neutron port and network configuration

Que Neutron for port and network configuration, return whatever is available.

Paramet

- **por** ironi port ID.
- **vif** Neutron

port ID.

- **cli** Optional a Neutron clien object.
- **con** (`irc com con Req`) request context

Raises
Network Error

Returns
a dict holding network configuration information associated with this ironic or Neutron port.

`ironic.`
Extract the swit port in-

formation for the node

The information is returned in the form of:

```
↪',
```

(continues on next page)

```
↪1',
```

(continued from previous page)

```
↪'val'
```

Paramet

tas

a task containing the Nod object.

Returns

port information as a dict

`ironic.`

Return the set of phys

i-
cal
net-
worl
as-
so-
ci-

ated with a neutron port.

Que
the
net-
worl
to
whic
the
port
is
at-
tach
and
re-
turn

the set of physical networks associated with the segments in that network.

Paramet

- **cli**
  A
  Neu
  tron
  clien
  ob-
  ject.
- **por**
  UUI
  of
  a
  Neu
  tron
  port
  to
  quer

Returns
A
set
of
phys
i-

cal net-worl

Raises
Net-worl Error if the net-worl quer fails

Raises
Invalic Param-e-ter-Valu for miss-ing net-worl

`ironic.`
Chec that the port is Sma NIC port

Paramet
por an instanc of ironi or port data as dict.

Returns

A bool to indicate port as Sma NIC port

`ironic.`

Dele the neutron port matc by para

Paramet

- **tas** a Task agen instanc
- **par** Dict of para to filter port

Raises

Network Error

`ironic.`

Dele the neutron

port cre- ated for boot ing the ram

Paramet

- **tas** a Task ager in- stan
- **net** UUI of a neu- tron net- worl port will be dele from

Raises
Net- worl Er- ror

`ironic.`
At- temp to dele any port cre- ated by clea ing/

Purposefully will not raise any exceptions so error handling can continue.

Paramet

- **tas** a Task ager instanc
- **net** UUI of a neutron network

ironic.

Unbind a neutron port

Remov a neutron port binding

profile and host ID so that it returns to an unbound state.

Paramet

- **por** Neutron port ID.
- **cli** Optional a Neutron clien object.
- **con** (ir com con Req request context
- **res** reset mac address

Raises
Network Error

an unbound state.

ironic.

Un-
date
a
neu-
tron
port

Uses
neu-
tron
clien
from
conf
clien
to
up-
date
a
neu-
tron
clien

Paramet

- **con** re-
ques
con-
text,
in-
stanc
of
iron
- **por** Neu
tron
port
ID.
- **att** The
at-
tribu
to
up-
date

on the port

- **cli** Optiona Neutron clien

ironic.

Updates a porta mac address

Paramet

- **por** Neutron port id.
- **add** new MAC address
- **con** (irc com con Req request context

Raises

Faile ToUpdate Port

`ironic.`

Chec that the give net- work is pres

Paramet

- **uui** net- work UUI or nam
- **net** hum read net- work type for er- ror mes- sage
- **con** (`iro` `com` `con` `Req` re- ques con- text

Returns net- work UUI

Raises Miss ing-

Pa-
ram-
e-
ter-
Valu
if
uuid
is
emp

Raises
Net-
work
Er-
ror
on
fail-
ure
to
con-
tact
Neu
tron

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
for
miss
ing
or
du-
pli-

cated network

`ironic.`
Chec
that
port
con-
tains
enou
in-
for-
ma-
tion
for
de-

ploy

Neu tron net- work in- ter- face re- quir that lo- cal_ field is

filled before we can use this port.

Paramet

- **nod** Iron node ob- ject.
- **por** Iron port ob- ject.

Returns
True if port info is valid Fals oth- er- wise

`ironic.`

Wait for neu- tron agen

to become tar-get state

Paramet

- **cli** A Neutron client object.
- **hos** Age host
- **tar** up: wait for up status, dow wait for dow status

Returns bool indicates the agent state matc para valu target_

Raises

exception.
if
target_
is
not
valic

Raises

exception.
if
host
status
didn
matc
the
requir
sta-

tus after max retry attempts.

`ironic.`

Wai
for
port
status
to
be
the
desired
status

Paramet

- **cli**
  A
  Neu
  tron
  clien
  object.
- **por**

Neu
tron
port

- **sta**
  Port
  target
  status,
  can
  be
  ACTIV
  DOV
  etc.

Returns
bool
indicate
that
the
port
status
matc
the
requir

value passed by param status.

Raises
InvalidParam-
eterValu
if
the
port
does
not
exist.

Raises
excep-

tion. if port sta- tus didn matc the re- quir sta-

tus after max retry attempts.

ironic.common.nova module

ironic.

Cre- ates and send pow state char for the pro- vide serv

Paramet

- **con** re- ques con- text, in- stan of iron
- **ser** The uuid of the node who pow

state char

- **tar** Targete pow state char i.e POW or POW

Returns
A bool whic indicate if the pow update was executed successfully (mainly for testing purposes).

ironic.common.policy module

Policy Engine For Iron

`ironic.`
A shortcut for policy.E

Chec authorization of a rule agai the target and credentials, and raises an exception if the rule is not defined. Always returns true if CONF.auth_strategy is not keystone.

`ironic.`

A shortcut for policy.E

Chec authorization of a rule agai the target and credentials and returns True or False.

`ironic.`

Configuration awar role policy chec wrapper.

Chec

authorization of a rule agai the target and credentials and returns True or False. Always returns true if CONF.auth_strategy is not keystone.

`ironic.`

Provide access to the single instan of Policy enforcer.

`ironic.`

`ironic.`

Synchro initializes the policy enforc

Paramet

- **`pol`** Custom

policy file to use, if none is spec i- fied, *CONF.oslo_policy.policy_file* will be used.

- **rul** Default dictionary / Rule to use. It will be considered just in the first instantiation.
- **def** Default rule to use, CON will be used if none is specified.
- **use** Whe to load rules from config

file.

`ironic.`

ironic.common.profiler module

`ironic.`

Setu OS-profiler notifier and enable profiling.

Paramet

- **`nam`** nam of the service that will be profiled
- **`hos`** host nam or host IP addres that the service will

be running on. By default host will be set to 0.0.0.0, but specifying host name / address usage is highly recommended.

Raises

Typ in case of invalid connection strin for a notifier backend, which is set in osprofiler.initializer.init_from_conf.

`ironic.`

Wra the OSProfiler trace decorator

Wra the OSProfiler trace decorator so that it will not try to patch the class unless OSProfiler is present and enabled in the config

Paramet

- **nam** The nam

of action. For example, wsg rpc, db, etc..

- **kwa** Any othe keyword args used by profiler.

ironic.common.pxe_utils module

class i

Base *iro dri moc ima Ima*

`ironic.`

`ironic.`

`ironic.`

`ironic.`

Buil the kickstart template options for a node

This meth buil the kick start template options for a node give

all the required parameters.

The options shou then be pass to pxe_ to create the ac-

tual config files.

Paramet
`tas`
A Task agen object

Returns
A dictionary of kick start

op- tions to be used in the kickstart template.

`ironic.`

Buil the PXE con- fig op- tions for a node

This meth builc the PXE boot op- tions for a node give all the required parameters.

The op- tions shou then be pass to pxe_ to cre- ate the actual config files.

Paramet

- **`tas`** A Task ager object
- **`pxe`** a dict of values to set on the configuration file
- **`ser`** if True buil service mod pxe config for netb ed user image and skip adding deployment image kernel and ramdisk info to PXE options.
- **`ipx`** Default false bool to indi-

cate
if
ipxe
is
in

use by the caller.

- **ram**
the
parameters
to
be
pass
to
the
ram
as

kernel command-line arguments.

Returns
A
dictionary
of
pxe
options
to
be
used
in
the

pxe bootfile template.

`ironic.`

`ironic.`

Fetc
the
nec-

essary ker- nels and ram for the in- stan

`ironic.`

Clea up the TFT en- vi- ron- men for the task node

Paramet

tas

A Task agen in- stan

`ironic.`

Clea PXE en- vi- ron- men of all the im- ages in im- ages_info.

Clea up the

PXE environment for the mentione images

in images_info.

Paramet

- **tas** a Task agen object
- **ima** A dictionary of images who keys are the im-

age names to be cleaned up (kernel, ramdisk, etc) and values are a tuple of identifier and absolute path.

`ironic.`

Render the iPXE boot script into the HTT root directory

ironic.

Generate PXE configuration file and MAC address links for it.

This method will generate the PXE configuration file for the tasks node under a directory named with the UUID of that node. For each MAC address or DHCP IP address (port) of that node, a symlink for the configuration file will be created under the PXE configuration directory, so regardless of which port boots first theyll get the same PXE configuration. If grub2 bootloader is in use, then its configuration will be created based on DHCP IP address in the form nn.nn.nn.nn.

Paramet

- **tas** A Task ager instan
- **pxe**

A dictionary with the PXE configuration

parameters.

- **tem**

  The PXE configuration template. If no tem-

plate is given the node specific template will be used.

`ironic.`

Retriev the DHC PXE boot options

Paramet

- **tas**

  A Task agen instanc

- **ipx** De- fault false bool that sig- nals if iPX for- mat- ting

should be returned by the method for DHCP server configuration.

- **url** De- fault false bool to in- form the meth if a URI

should be returned to boot the node. If [pxe]ip_version is set to *6*, then this option has no effect as url_boot form is required by DHCPv6 standards.

- **ip_** The IP ver- sion of op- tions to re- turn as val-

ues differ by IP version. Default to [pxe]ip_version. Possible options are integers 4 or 6.

Returns
Dic- tio- nary

to be sent to the networking service describing the DHCP options to be set.

`ironic.`

Generate absolute paths to various images from their name(label)

This method generates absolute file system paths on the conductor where various images need to be placed. For example the kickstart template, file and stage2 squashfs.img needs to be placed in the ipxe_root_dir since they will be transferred by anaconda ramdisk over http(s). The generated paths will be added to the image_info dictionary as values.

Paramet

- **nod** the

UUI of the node

- **roo** Directory in whic the image mus be plac

- **lab** Nam of the image

`ironic.`

Generate http url path to various image ar- tifacts

This meth generates http urls for

various image artifacts int the webserver root. The generated urls will be added to the pxe_options dict and used to render pxe/ipxe configuration templates.

Paramet

- **htt** URI to access the root of the web serv
- **nod** the UUI of the node
- **lab** Nam of the image

`ironic.`

Generate the path for TFT files for deploy or rescue images.

This meth generates the path for the deploy (or rescue) kernel and deploy (or rescue) ramdisk.

Paramet

- `nod` a node object
- `mod` Label indicating a deploy or rescue operation being carried out on the node. Supported values are deploy and rescue. Defaults to deploy, indicating deploy operation is being carried out.
- `ipx` A default Fals bool

valu to tell the meth if the

caller is using iPXE.

Returns
a dic- tio- nary who keys are the nam of the im- ages

(deploy_kernel, deploy_ramdisk, or rescue_kernel, rescue_ramdisk) and values are the absolute paths of them.

Raises
Miss ing- Pa- ram- e- ter- Valu if de- ploy or res- cue_

is missing in nodes driver_info.

`ironic.`

Gen er- ate the path for TFT files for

in-
stan
re-
latec

images.

This
meth
generates
the
path
for
in-
stan
kernel
and
in-

stance ramdisk. This method also updates the node, so caller should already have a non-shared lock on the node.

Paramet

- **tas**
A
Task
ager
in-
stan
containing
node
and
context.
- **ipx**
De-
fault
false
bool
to
indicate
if
ipxe

is in

use by the caller.

Returns
a dictionary whose keys are the names of the images

(kernel, ramdisk) and values are the absolute paths of them. If its a whole disk image or node is configured for localboot, it returns an empty dictionary.

`ironic.`

`ironic.`

Get href and tftp paths for deploy or rescue kernel

and ramdisk.

Paramet

- **nod** UUI of the node
- **dri**

Nod
drive
dict

- `mod`
A
la-
bel
to
in-
di-
cate
whe
path
for
de-
ploy

or rescue ramdisk are being requested. Supported values are deploy rescue. Defaults to deploy, indicating deploy paths will be returned.

- `ipx`
A
de-
fault
Fals
bool
valu
to
tell
the
meth
if
the

caller is using iPXE.

Returns
a
dic-
tio-
nary
who
keys
are
de-
ploy
and
de-
ploy
or

rescue_kernel and rescue_ramdisk and whose values are the absolute paths to them.

Note
drive
shou
be
validate
outside
of
this
meth

ironic.

Return
file
relative
path
to
CON

Paramet

fil
full
file
path
to
be
mad
relative
path

Returns

The
path
relative
to
CON

ironic.

Generate
the
path

for the node PXE configuration file.

Paramet

- **nod** the UU of the node
- **ipx** A default Fals bool valu to tell the meth if the caller is using iPXE.

Returns
The path to the node PXE configuration file.

`ironic.`
Returns

the directory whe the config files and images will live.

ironic.

Add trailing slas (if need necessary for path prefi

Returns

CON ensure to have a trailing slas

ironic.

Identify volume information for iPXE template generation.

`ironic.`

Re- turn true if ipxe is set.

Paramet

`tas`

A Task agen ob- ject

Returns

bool true if `[px` is con- fig- ured or if the task driv

instance is the iPXE driver.

`ironic.`

Gets the driv spe- cific Nod de- ploy men info

This meth val- i- date whe the driv

prop erty of the sup plied

node contains the required information for this driver to deploy images to, or rescue, the node.

Paramet

- **nod** a single Nod
- **mod** Label indicating a deploy or rescue

operation being carried out on the node. Supported values are deploy and rescue. Defaults to deploy, indicating deploy operation is being carried out.

Returns
A dict with the driv values.

Raises
Miss ing-Parameter-Valu

`ironic.`

Prepare to boot ana- conc rame by generating kick

start file

Paramet

- **`tas`** a task from Task ager
- **`ima`** a dict of values of in- stan image meta data

to set on the configuration file.

- **`ana`** if the boot is to

a ana- conc rame con- fig- u-

ration.

ironic.

Pre- pare the con- fig file for PXE boot

Paramet

- **tas** a task from Task ager
- **ima** a dict of val- ues of in- stanc im- age meta data

to set on the configuration file.

- **isc**

if boot is from an iSC vol- ume or not.

- **ram** if the boot is to a ram con- fig- u- ra- tion.

- **ipx** De- fault false bool to in- di- cate if ipxe is in

use by the caller.

- **ana** if the boot is to a ana- conc ram configu-

ration.

Returns

Non

`ironic.`

Chec if boot parameters are valid for trust boot

`ironic.`

Chec if the kickstart file is valid

Paramet

`ks_`

Contents of kickstart file to validate

Raises

Invalid Kickstart File

`ironic.`

Val-

i-
date
the
kick
start
tem-
plate

Paramet

`ks_`

Path
to
the
kick
start
tem-
plate

Raises

In-
valid
Kick
start
Tem
plate

ironic.common.qemu_img module

`ironic.`

Con
vert
im-
age
to
othe
for-
mat.

ironic.common.raid module

`ironic.`

Se-
rial
At-
tach
SCS

`ironic.`

Serial AT Attach men

`ironic.`

Sma Com pute Sys tem Inter face

`ironic.`

Filter the target raid config base on root volume

creation

This meth can be used by any raid interface whic wan to

filter out target raid config based on condition whether the root volume will be created or not.

Paramet

- **nod** a node object
- **cre** A bool de- fault valu True gov- ern- ing if the root

volume is returned else root volumes will be filtered out.

- **cre** A bool de- fault valu True gov- ern- ing if the non

root volume is returned else non-root volumes will be filtered out.

Raises
Miss- ing- Pa- ram- e- ter- Valu if node is miss- ing or

was found to be empty after skipping root volume and/or non-root volumes.

Returns
It will return filtered target_

`ironic.`
Get logical disk properties fron RAI configuration schema.

This meth read the logical properties and thei textual description from the schema that is passed.

Paramet
`rai`
A dictionary whic is the

sche
to
be
used
for

getting properties that may be specified for the logical disk.

Returns
A
dic-
tio-
nary
con-
tain-
ing
the
log-
i-
cal
disk
prop

erties as keys and a textual description for them as values.

`ironic.`

Up-
date
the
node
in-
for-
ma-
tion
base
on
the
RAI
con-

fig.

This
meth
up-
date
the
node
in-
for-
ma-
tion
to
mak
use
of

the configured RAID for scheduling purposes (through properties[capabilities] and properties[local_gb]) and deploying purposes (using properties[root_device]).

Paramet

- **nod** a node object
- **rai** The dictionary containing the current RAI con-

figuration.

Raises

Invalid Parameter Valu if raid has more than one

root volume or if node.properties[capabilities] is malformed.

`ironic.`

Validate the RAI config-

u-
ra-
tion
pass
us-
ing

JSON schema.

This
meth
val-
i-
date
a
RAI
con-
fig-
u-
ra-
tion
agai
a

RAID configuration schema.

Paramet

- **rai**
  A
  dic-
  tio-
  nary
  con-
  tain-
  ing
  RAI
  con-
  fig-
  u-
  ra-

tion information

- **rai**
  A
  dic-
  tio-
  nary
  whic
  is
  the
  sche

to be used for validation.

Raises
In- valid Pa- ram- e- ter- Valu if val- i- da- tion of the RAID configuration fails.

ironic.common.release_mappings module

ironic.

Gets the sup- port ver- sion for all ob- jects

Sup- port ver- sion are from the RE- LEA

Paramet

- **rel** a

list of releas nam if emp versions fron all releases are returned (the default).

- **obj** a list of nam of objects of interest. If empty/None, versions of all objects are returned (the default).

Returns
a dictionary whe the key is the object nam and the value is a set of supported versions.

ironic.common.rpc module

class i
Base
`osl`
`ser`
`Ser`

deseria
Deserialize a dictionary into a reques

context.

Paramet
ctx
Reques context dictionary

Returns
Deserialized form of entity

deseria
Deserialize som

thin
from
prim
i-
tive
form

Parame

- **ctx**
  Re-
  ques
  con-
  text,
  in
  de-
  se-
  ri-
  al-
  ized
  form
- **ent**
  Prim
  i-
  tive
  to
  be
  de-
  se-
  ri-
  al-
  ized

Returns
De-
se-
ri-
al-
ized
form
of
en-
tity

seriali
Se-
ri-
al-
ize
a

request context into a dictionary.

Parame

ctx
Request context

Returns
Serialized form of context

seriali
Serialize something to primitive form

Parame

- **ctx** Request context, in deserial-

ized
form

- **ent**
  Entity
  to
  be
  serialized

Returns
Serialized
form
of
entity

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

ironic.common.rpc_service module

class i

Base
osl
ser
Ser

handle_

Add a signal handler for SIGUS

The handler ensure that the man ager is not dere istere

when it is shutdown.

start()

Star a service

stop()

Stop a service

Parame

gra indicate

whe to wait for all threa to finish

or terminate them instantly

ironic.common.service module

`ironic.`

`ironic.`

ironic.common.states module

Map ping of bare meta node state

Setting the node *pow* is handled by the conductors pow

synchronization thread. Based on the power state retrieved from the driver for the node, the state is set to POWER_ON or POWER_OFF, accordingly. Should this fail, the *power_state* value is left unchanged, and the node is placed into maintenance mode.

The *pow* can also

be set manually via the API. A failure to change the state leaves the current state unchanged. The node is NOT placed into maintenance mode in this case.

`ironic.`

Nod is successfully deploy and associated with an instance.

`ironic.`

Nod faile to com plete the adop tion process.

This state is the resulting state of a node that

faile
to

complete adoption, potentially due to invalid or incompatible information being defined for the node.

`ironic.`

Nod
is
being
adop

This
provision
state
is
intend
for
use
to
mov
a
node

from MANAGEABLE to ACTIVE state to permit designation of nodes as being managed by Ironic, however deployed previously by external means.

`ironic.`

Nod
is
avail
able
for
use
and
sche
ing.

This
state
is
replac
ing
the
NOS
TAT
state
used
prior
to
Kilo

ironic.

Nod faile clea ing. This re- quir op- er- a- tor in- ter- vention to resolve.

ironic.

Nod is be- ing au- to- mat- i- cally clea to pre- pare it for provisioning.

ironic.

Nod is wait ing for a clea step to be fin- ishe

This will be the node *pro- vi-*

sion
whil
the
node
is
wait
ing

for the driver to finish a cleaning step.

`ironic.`

Nod
tear
dow
was
successful.

In
Junc
target_
was
set
to
this
valu
during
node
tear
dow

In
Kilo
this
will
be
a
transitory
valu
of
provision

and never represented in target_provision_state.

`ironic.`

State
in
whic

node
dele-
tion
is
al-
lowe

ironic.
Nod
is
ac-
tivel
be-
ing
torn
dow

ironic.
Nod
was
suc-
cess
fully
de-
ploy

This
is
main
a
tar-
get
pro-
vi-
sion
state
used
dur-
ing
de-

ployment. A successfully deployed node should go to ACTIVE status.

ironic.
Nod
de-
ploy
men
faile

ironic.
Nod
is
read
to

receiv a deploy reques or is currently being deployed.

A nod will have its *provision* set to DEPLO ING brief before it receives its initial deploy request. It will also move to this state from DEPLOYWAIT after the callback is triggered and deployment is continued (disk partitioning and image copying).

`ironic.`

Nod is waiting to be deploy

This will be the node *provision* whil the node is waiting for the driver to finish deployment.

`ironic.`

Nod is en- rolle

This state in- di- cate that Iron is awa of a node but is

not managing it.

`ironic.`

An er- ror oc- curr dur- ing node pro- cess ing.

The *last_* at- tribu of the node de- tails shou con- tain an er-

ror message.

`ironic.`

State

whe API look are permitted with fast trac enable

`ironic.`

Nod inspection faile

`ironic.`

Nod is under inspection.

This is the provision state used whe inspection is start

A successfully inspected node shall transition to MANAGEABLE state. For asynchronous inspection, node shall transition to INSPECTWAIT state.

`ironic.`

Nod is under inspec

tion.

This is the pro- vi- sion state used whe an asyn chro in- spec

tion is in progress. A successfully inspected node shall transition to MANAGEABLE state.

`ironic.`

State whe API look are nor- mall al- lowe for node

`ironic.`

Nod is in a man age- able state

This state in- di- cate that Iron has ver- i- fied, at

least
once

that it had sufficient information to manage the hardware. While in this state, the node is not available for provisioning (it must be in the AVAILABLE state for that).

`ironic.`

No
state
information.

This
state
is
used
with
pow
to
represent
a
lack
of
know

edge of power state, and in target_*_state fields when there is no target.

`ironic.`

Nod
is
powered
off.

`ironic.`

Nod
is
powered
on.

`ironic.`

Nod
is
rebooting.

`ironic.`

Nod
is
to

be rebuilt

This is not used as a state but rather as a verb when chan

ing the nodes provision_state via the REST API.

ironic.

Nod is in rescue mod

ironic.

Nod rescue faile

ironic.

Nod is waiting on an external callback

This will be the node *provi-*

sion
whil
the
node
is
wait
ing

for the driver to finish rescuing the node.

`ironic.`
Nod
is
in
pro-
cess
of
be-
ing
res-
cued

`ironic.`
Nod
is
in
the
pro-
cess
of
soft
pow
off.

`ironic.`
Nod
is
re-
boot
ing
grac
fully

`ironic.`
Stat
that
will
not
tran-
si-
tion
un-
less
re-

ceiv ing a

request.

`ironic.`

State that cannot be resum once a conductor dies

If a node gets stuck with one of these state for som reason

(eg. conductor goes down when executing task), node will be moved to fail state.

`ironic.`

Nod unrescue faile

`ironic.`

Nod is being restore from rescue

mod
(to
active

state).

`ironic.`
State
that
can
be
char
without
external
requests

`ironic.`
Transitional
state
in
whic
we
allow
updating
a

node.

`ironic.`
Mapping
of
state
char
even
that
are
PUT
to
the
RES
API

This is a
PUT

/v1/

{target: active

The dict

{target strin use by the API internal verb

This provide a reference set of supporte actions and in the future may be used to support renaming these actions.

`ironic.`

Nod pow management credentials are being verified.

`ironic.`

Use

to log whe entering a state

`ironic.`

Used to log whe a state is exited.

ironic.common.swift module

class `i`

Base `obj`

API for communicating with Swi

connect

Underlying Swi connection object.

create_

Upload

a give file to Swi

Parame

- **con** The nam of the container for the object.
- **obj** The nam of the object in Swi
- **fil** The file to upload as the object data
- **obj** the headers for the ob-

ject to pass to Swi

Returns
The Swi UUI of the object

Raises
Swi Operationl if any operation with Swi

fails.

delete_
Dele the give Swi object.

Paramet

- **con**
The nam of the container in whic Swi ob-

ject is

placed.

- **obj** The nam of the object in Swi to be dele

Raises
Swi bject-Not-Fou ror, if object is not foun in

Swift.

Raises
Swi Operation if operation with Swi fails

get_ten Returns the

temp url for the give Swi ob- ject.

Parame

- **con** The nam of the con- taine in whic Swi ob- ject is

placed.

- **obj** The nam of the Swi ob- ject.
- **tim** The time out in sec- onds af- ter whic the gen- er-

ated url should expire.

Returns
The temp url for the object.

Raises
Swi Operation if any operation with Swi

fails.

head_ob
Retriev the information about the give Swi object.

Paramet

- **con** The nam of the container in whic

Swi object is placed.

- **obj** The nam of the object in Swi

Returns The information abou the object as return by Swift clients head_object call.

Raises Swi Operation if operation with Swi fails

update_

Update the meta

data of a give Swi ob- ject.

Paramet

- **con** The nam of the con- taine in whic Swi ob- ject is

placed.

- **obj** The nam of the ob- ject in Swi
- **obj** the head ers for the ob- ject to pass to Swi

Raises
Swi

Op-
er-
a-
tionl
if
op-
er-
a-
tion
with
Swi
fails

ironic.

ironic.common.utils module

Util-
i-
ties
and
help
func
tions

ironic.
Chec
a
di-
rec-
tory
is
us-
able

This
func
tion
can
be
used
by
drive
to
chec
that
di-
rec-
to-

ries they need to write to are usable. This should be called from the drivers init function. This function checks that the directory exists and then calls check_dir_writable and check_dir_free_space. If directory_to_check is not provided the default is to use the temp directory.

Paramet

- **dir** the directory to chec
- **req** amo of spac to chec for in MiB

Raises
Path Not-Foun if directory can not be foun

Raises
Directo-ryNo if user is unable to writ to the

directory

Raises

Ins
if
free
spac
is
<
re-
quir
spac

ironic.

ironic.
Con
ve-
nien
wrap
per
arou
os-
los
ex-
e-
cute
meth

Paramet

- **cmd**
  Pass
  to
  pro-
  ces-
  su-
  tils.e
- **use**
  True
  |
  Fals
  De-
  fault
  to
  Fals
  If
  set
  to
  True
  ex-

ecute command with standard locale added to environment variables.

Returns
(stdout, stde
from process execution

Raises
UnknownArgumen
ror

Raises
ProcessExecution
Erro

ironic.
Che
that content of the file is the sam
as provide

reference.

Paramet

- **pat**
path to

file

- **con** reference content to chec agai
- **has** hashing algo from hashlib to use, default is sha2

Returns True if the hash of reference content is the sam

as the hash of files content, False otherwise

`ironic.`
Gets a mim type of the give file.

`ironic.`

Return an update capability strin

This meth update the original (or current) capabilities with the new capabilities. The original capabilities would typically be from a nodes properties[capabilities]. From new_capabilities, any new capabilities are added, and existing capabilities may have their values updated. This updated capabilities string is returned.

Paramet

- **`cur`** Current capability strin
- **`new`** the dictionary of capa-

bil-
i-
ties
to
be

updated.

Returns
An
up-
date
ca-
pa-
bil-
ity
strin
with
new

Raises
Val-
ueEr
ror,
if
cur-
rent
is
mal-
form
or
if
new
is

not a dictionary

`ironic.`
Chec
if
FIPS
mod
is
en-
able
in
the
sys-
tem.

`ironic.`
Old
chec
for
valic

logical node nam

Retaine for com patibility with RES API < 1.10

Nominall

- http: // en. wiki org/ wiki Hos
- http: // tool ietf. org/ html rfc9
- http: // tool ietf. org/ html rfc1

In prac tice, this

chec has several shor com ings and errors

that are more thoroughly documented in bug #1468508.

Paramet

hos

The host nam to be validate

Returns

True if valid Fals if not.

ironic.

Chec available system men ory and hold the deploy men

process.

Evaluates the current

system
men
ory
avai
able
mea
ing

can be allocated to a process by the kernel upon allocation request, and delays the execution until memory has been freed, or until it has timed out.

This
meth
will
issue
a
slee
if
the
amo
of
avai
able
men

ory is insufficent. This is configured using the `[DEFAULT]minimum_memory_wait_time` and the `[DEFAULT]minimum_memory_wait_retries`.

Paramet

rai
Default
Fals
but
if
set
to
true
an
Insuffi-

centMemory exception is raised upon insufficent memory.

Returns

True
if
the
chec
has
time
out.
Oth-

er-
wise
Non
is
re-

turned.

Raises

In-
suf-
fi-
cent
Men
ory
if
the
raise
pa-
ram-
e-
ter

is set to True.

`ironic.`

`ironic.`

Ver-
ify
the
for-
mat
of
an
Ope
Flov
dat-
a-
p-
ath_

Che
if
a
dat-
a-
p-
ath_
is
valid
and
con-
tains

16 hex-

adecimal digits. Datapath ID format: the lower 48-bits are for a MAC address, while the upper 16-bits are implementer-defined.

Paramet

dat

Ope Flov dat-a-p-ath_ to be val-i-date

Returns

True if valic Fals if not.

ironic.

De-ter-minc if a log-i-cal nam is valic

The log-i-cal nam may only con-sist of RFC un-reserv

characters, to wit:

AL-
PHA
/
DIG
/
-

/
.
/
_
/
~

`ironic.`

Che
no_
validity

Che
if
no_
valu
that
will
be
written
to
environmen

variable by ironic-python-agent is valid.

Paramet

`no_`
the
valu
that
re-
quir
validity
chec
Ex-

be a comma-separated list of host names, IP addresses and domain names (with optional :port).

pect to

Returns

True if no_p is vali Fals otherwise

`ironic.`

Mou a device file on spec ified location.

Paramet

- **`src`** the path to the sour file for mou ing
- **`des`** the path whe it need to be mou

- **arg** a tuple containing the arguments to be

passed to mount command.

Raises
processutils.I if it faile to run the process

`ironic.`
Pars the instan capabilities.

One way of having these capabilities set is via Nova, where the capabilities are defined in the Flavor extra_spec and passed to Ironic by the Nova Ironic driver.

NOT Althou our API fully support JSO field to main tain the backward compatibility with Juno the Nova Ironic driver is sending it as a string.

Paramet
`nod` a single Nod

Raises
Invalic ParameterValu if the capabilities string is not a dictionary or is malformed.

Returns
A dictionary with the capa-

bil-
i-
ties
if
foun

otherwise an empty dictionary.

`ironic.`

Pop
a
valu
from
a
dic-
tio-
nary
field
of
a
node

Paramet

- **nod**
  Nod
  ob-
  ject.
- **col**
  Nam
  of
  the
  field
  with
  the
  dic-
  tio-
  nary
- **fie**
  Nest
  field
  nam
- **def**
  The
  de-
  fault

valu to return

Returns
The removed valu or the default

`ironic.`
Removes specific keys from the var, recursing into dicts

and lists.

`ironic.`

Renders Jinja template file with give parameters.

Paramet

- **tem** full path to

the Jinja template file

- **par** dictionary with parameters to use when ren-

dering

- **is_** whe template is file or strin with template itself
- **str** Enables strict template rendering. Default is Fals

Returns

Ren-
dere
tem-
plate

Raises
jinja

`ironic.`

`ironic.`
Re-
mov
trail-
ing
char
ac-
ters
from
a
strin
if
that
does

not make it empty

Paramet

-
  val
  A
  strin
  valu
  that
  will
  be
  strip
-
  cha
  Cha
  ac-
  ters
  to
  re-
  mov

Returns
Strip
valu

`ironic.`
Set

a valu in a dictionary field of a node

Paramet

- **nod** Nod object.
- **col** Nam of the field with the dictionary
- **fie** Nest field nam
- **val** New valu

ironic.

ironic.

Umc a mou location.

passed to the umount command.

Paramet

- **loc** the path to be unmou
- **arg** a tuple containing the arguments to be

Raises

processutils.I if it faile to run the process

`ironic.`

`ironic.`

Validate an Ope Flow data-

p-
ath_
and
re-
turn

normalized form.

Chec
whe
the
sup-
plied
Ope
Flow
dat-
a-
p-
ath_
is
for-
mall

correct and normalize it to all lower case.

Paramet
dat
Ope
Flow
dat-
a-
p-
ath_
to
be
val-
i-
date
and

normalized.

Returns
Nor-
mal-
ized
and
val-
i-
date
Ope
Flow
dat-
a-
p-
ath_

Raises

InvalidDatapathI If an Ope Flow datapath_id is not valid.

`ironic.`

Validate a MA addres and return normalized form.

Chec whe the supplied MA addres is formall correct and normalize it to all lower case.

Paramet

`add` MA addres

to be validate and normalized

Returns
Normalized and validate MAC address

Raises
Invalid MAC If the MAC address is not valid

`ironic.`

`ironic.`
Validate the given port

Parame

- `por` TCP port

-

por
Nam
of
the
port

Returns
An in-te-ger port num-ber.

Raises
In-valic Pa-ram-e-ter-Valu if the port is in-valic

ironic.
Wra the ad-dres in squa brac ets if its an IPv6 ad-

dress.

ironic.

ironic.common.wsgi_service module

class i

Base
osl
ser
Ser

Pro- vide abil- ity to laun iron API from wsg app.

reset()

Re- set serv gree pool size to de- fault

Returns

Non

start()

Start serv ing this ser- vice us- ing load con- fig- u- ra- tion.

Returns

Non

stop()

Stop serv ing this API

Returns

Non

wait()

Wait for the service to stop serv ing this API

Returns

Non

Module contents

ironic.conductor package

Submodules

ironic.conductor.allocations module

Functionality related to allocations

`ironic.`

Assign the

previously allocated node to the node

allocation.

This is not the actual allocation process but mere

backfilling of allocation_uuid for a previously allocated node.

Paramet

- **con** an admin context
- **all** an allocation object associated with

the node

- `nod` An ID of the node

Raises Allocation-Fail if the node does not matc the allocation

Raises Nod Associated if the node is already associated with another instance or allocation.

Raises Instanc Associated if the allo-

ca-
tions
UUI

is already used on another node as instance_uuid.

Raises

Nod
Not-
Four
if
the
node
with
the
pro-
vide
ID
can-
not

be found.

`ironic.`

Pro-
cess
the
al-
lo-
ca-
tion.

This
call
runs
in
a
sep-
a-
rate
threa
on
a
con-
duc-
tor.

It finds suitable nodes for the allocation and reserves one of them.

This
call
does
not
raise
ex-
cep-

tions since its design to work asynchronously.

Paramet

- **con** an admin context
- **all** an allocation object

`ironic.`

Verify that allocation can be removed for the node.

Paramet

- **nod** a node

object

- **all** an allocation object associated with

the node

ironic.conductor.base_manager module

Base conductor manager functionality.

class i

Base obj

del_hos

init_ho

Initialize the conductor host

Paramet

adm the admin context to pass to periodic task

Raises
Run time ror whe conductor is already running

Raises
NoD sLoa whe no driv are enabled on the conductor.

Raises
Driv Not-Four if a driv is enable

that
does
not
ex-
ist.

Raises
Driv
Load
Er-
ror
if
an
en-
able
driv
can-
not
be
load

Raises
Driv
Nam
Con-
flict
if
a
clas-
sic
driv
and
a
dy-
nam
driver are both enabled and have the same name.

iter_no
It-
er-
ate
over
node
map
to
this
con-
duc-
tor.

Re-
ques
node

set from and filters out node that are not map

to this conductor.

Yiel tuples (nod driv conductor_) whe is derive from

fields argument, e.g.: fields=None means yielding (uuid, driver, conductor_group), fields=[foo] means yielding (uuid, driver, conductor_group, foo).

Parame

- **`fie`** list of field to fetch in addition to uuid driv

and conductor_group

- **`kwa`** additiona ar- gu- men to pass to dbap whe look

ing for nodes

Returns

gen- er- a- tor yield ing tu- ples of re- ques field

prepare

Pre- pare host for ini- tial- iza- tion

Re- mov ex- ist- ing data en- tries in- volv with node lock ing

for nodes in a transitory power state and nodes that are presently locked by the hostname of this conductor.

Under normal operation, this is also whe the initial database connectivity is established for the conductors normal operation.

ironic.conductor.cleaning module

Fun tionality related to clea ing.

`ironic.`

Con tinu clea ing after finishing an asyn clea step

This func tion calculates whic

control into do_next_clean_step.

step has to run next and pass

Paramet

`tas`

a Task agen instance with an exclusive lock

`ironic.`

Do clean ing, start ing from the spec i fied clea step

Paramet

- **`tas`**

  a Task agen instance with an exclusive

lock

- **ste** The first clea step in the list to execute. This

is the index (from 0) into the list of clean steps in the nodes driver_internal_info[clean_steps]. Is None if there are no steps to execute.

- **dis** Whe to skip boot ing ram for clea ing.

`ironic.`

Internal RPC meth to perform clea ing of a node

Paramet

- **tas** a Task

agen instan with an exclusive lock on

its node

- **cle** For a manual clea the list of clea step to per-

form. Is None For automated cleaning (default). For more information, see the clean_steps parameter of `ConductorManager.do_node_clean()`.

- **dis** Whe to skip boot ing ram for clea ing.

`ironic.` Internal meth to abor an ongoing op-

er-
a-
tion.

Paramet

- **tas** a Task ager in- stan with an ex- clu- sive lock
- **ste** The nam of the clea step

ironic.conductor.deployments module

Func- tion- al- ity re- latec to de- ploy ing and un- de- ploy ing.

`ironic.`

Con- tinu deploy- men af- ter fin- ish- ing an asyn de-

ploy step.

This func tion cal- cu- lates whic step has to run next and pass

control into do_next_deploy_step. On the first run, deploy steps and templates are also validated.

Paramet

tas a Task agen in- stan with an ex- clu- sive lock

`ironic.`

Do de- ploy men start ing from the spec ified deploy

step.

Paramet

- **tas** a Task ager instance with an exclusive lock
- **ste** The first deploy step in the list to execute

This is the index (from 0) into the list of deploy steps in the nodes driver_internal_info[deploy_steps]. Is None if there are no steps to execute.

`ironic.`

Prepare the environment and deploy a

node

`ironic.`

Start deployment or rebuilding on a node.

This function does not check the node suitability for deployment, its left up to the caller.

Paramet

- **tas** a Task ager instanc
- **man** a Conductor-Man ager to run

task
on.

- **con**
a
con-
fig-
driv
if
re-
ques

- **eve**
even
to
pro-
cess
de-
ploy
or
re-
buil

- **dep**
Op-
tion
de-
ploy
step

`ironic.`
Val-
i-
date
the
de-
ploy
step
af-
ter
the
ram
lear
abou

them.

`ironic.`
Val-
i-
date
that

a node is suitable for deploymen

Paramet

- **tas** a Task agen instanc
- **eve** even to process deploy or rebuil

Raises
Nod Mai tenanc Nod Protecte Invalid State

ironic.conductor.manager module

Con duct all activity relatec to bare meta deploy men

A single instance of *irc con man Con* is created with the *ironic-conductor* process, and is responsible for performing all actions on bare metal resources (Chassis, Nodes, and Ports). Commands are received via RPCs. The conductor service also performs periodic tasks, eg. to monitor the status of active deployments.

Driv are load via entrypoin by the *irc com dri* class. Each driver is instantiated only once, when the ConductorManager service starts. In this way, a single ConductorManager may use multiple drivers, and manage heterogeneous hardware.

Whe multiple *Con* are run on different host they are all active and cooperatively manage all nodes in the deployment. Nodes are locked by each conductor when performing actions which change the state of that node; these locks are represented by the *ironic.conductor.task_manager.TaskManager* class.

A tooz is used to distribu node acro the set of active conductors which support each nodes driver. Rebalancing this ring can trigger various actions by each conductor, such as building or tearing down the TFTP environment for a node, notifying Neutron of a change, etc.

class i

Base *irc con bas Bas*

Iron Conductor man ager main clas

RPC_API

add_no

change_

continu

RPC meth to contin clea ing a node

This is useful for clea ing task that are asyn Whe they com plete, they call back via RPC, a new worker and lock are set up, and cleaning continues. This can also be used to resume cleaning on take_over.

Paramet

- **con** an admin context.
- **nod** the id or uuid of

a node

Raises
In- valic Stat if the node is not in CLE WA state

Raises
NoF duc- tor- Wor whe there is no free worl to start asyn

task

Raises
Nod Locl if node is lock by an- othe con- duc- tor.

Raises
Nod Not- Four if the

nod no long ap- pear in the data

`continu`

RPC meth to continu de- ploy ing a nod

This is useful for de- ploy ing task that are asyn Whe they complete, they call back via RPC, a new worker and lock are set up, and deploying continues. This can also be used to resume deploying on take_over.

Paramet

- **`con`** an admin context.
- **`nod`** the ID or

UUI of a node

Raises
In- valic State if the node is not in DE- PLO WAI state

Raises
NoF duc- tor- Wor whe there is no free work to start asyn

task

Raises
Nod Lock if node is lock by an- othe con- duc- tor.

Raises
Nod Not-

Fou
if
the
node
no
long
ap-
pear
in
the
data

create_

create_

create_

destroy

destroy

destroy

destroy

destroy

destroy

do_node

do_node

do_node

do_node

do_node

do_prov

driver_

get_boo

get_cor

get_dri

get_dri

get_inc

get_noc

get_noc

get_rai

get_sup

get_sup

heartbe

inject_

inspect

object_

Per-
form
an
ac-
tion
on
a
Ver-
sion
dOb
ject
in-
stanc

Paramet

- `con`

The context with whic to perform the action

tion

- **obj** The object instan on whic to perform the ac-
- **obj** The nam of the action meth to call
- **arg** The positiona arguments to the action

meth

- **kwa** The key- word ar- gu- men to the ac- tion meth

Returns A tu- ple with the up- date mad to the ob- ject and

the result of the action method

object_

Per- form a back port of an ob- ject in- stan

The de- fault be- hav- ior of the

base Ver- sion dOb ject- Se-

rializer, upon receiving an object with a version newer than what is in the local registry, is to call this method to request a backport of the object.

Paramet

- **`con`** The con- text with whic to per- form the back port
- **`obj`** An in- stan of a Ver- sion dOb ject to be back

ported

- **`obj`** A dict of {ob- j- nam ver- sion map ping

Returns
The dowgradinstanof objinst

`object_`

Perform an action on a VersiondObject class

Paramet

- `con`
  The context with whic to perform the action
- `obj`
  The registry nam of the object

- **`obj`** The nam of the action meth to call
- **`obj`** A dict of {obj- nam version map ping
- **`arg`** The positiona argumen to the action meth
- **`kwa`** The keywor argumen to the action meth

Returns
The result of the action meth whic may (or may not)

be an instance of the implementing VersionedObject class.

remove_

set_boo

set_cor

set_inc

set_tar

target

update_

update_

update_

update_

update_

validat

vendor_

vif_att

vif_det

vif_lis

`ironic.`

Syn the pow state for this node incremen ing the cour on failure.

Whe the limi of pow is reac the node is put into mai tenance mode and the error recorded.

Paramet

- **tas** a Task agen instanc
- **cou** number of

time
this
node
has
pre-
vi-
ousl
faile
a

sync

Raises

Nod
Loc
if
un-
able
to
up-
grad
task
lock
to
an
ex-

clusive one

Returns

Cou
of
faile
at-
temp
On
suc-
cess
the
coun
is
set
to

0. On failure, the count is incremented by one

`ironic.`

`ironic.`

Han
dles

pow state sync ex- ceed ing the max re- tries

Whe syn- chro niz- ing the pow state be- twee a node and the

DB has exceeded the maximum number of retries, change the DB power state to be the actual node power state and place the node in maintenance.

Paramet

- **tas** a Task ager in- stanc with an ex- clu- sive lock
- **act** the ac- tual pow state of

the node a power state from

ironic.common.states

- **exc** the exception object that caus the sync power state

to fail, if present.

ironic.conductor.notification_utils module

ironic.

Help for conductor sending a set console state notification.

Paramet

- **tas** a

Task ager in- stan

- **act** Ac- tion strin to go in the Ever Typ Mus be ei-

ther console_set or console_restore.

- **sta** One of *iron* END or ER- ROR

`ironic.`

Hel for con- duc- tor send ing a set pow state no- ti-

fication.

Paramet

- **tas** a Task ager instan
- **lev** Notification leve One of *iron*
- **sta** Status to go in the Even Type One of *iron* or

ERROR. ERROR indicates that ironic-conductor couldnt retrieve the power state for this node, or that it couldnt set the power state of the node.

- **to_** the pow state the conductor is attemp ing to

set on the node. This is used instead of the nodes target_power_state attribute since the baremetal.node.power_set.start notification is sent early, before target_power_state is set on the

node.

`ironic.`

Help for conductor send ing a node pow state correcte notification.

Whe ironi detects that the actual pow state on a bare metal hardware is different from the power state on an ironic node (DB), the ironic nodes power state is corrected to be that of the bare metal hardware. A notification is emitted about this after the database is updated to reflect this correction.

Paramet

- **`tas`** a Task agen instanc
- **`fro`** the pow state

of the node be- fore this chan was de- tected

ironic.

Help for con- duc- tor send ing a set pro- vi- sion state notification.

Paramet

- **tas** a Task agen in- stanc
- **lev** One of field
- **sta** One

of field

- **pre** Previous provision state
- **pre** Previous target provision state
- **eve** FSM event that triggered provision state chan

ironic.conductor.rpcapi module

Clie side of the conductor RPC API

class i

Base
`obj`

Client side of the conductor RPC API.

API version history:

1.0 -

Initial version.

Inclu
get_

1.1 -

Add update
and start

1.2 -

Add vendor_

1.3 -

Re-

nam start to char

1.4 -

Add do_r and do_r

1.5 -

Add validate

1.6 -

char do_r and do_r

acce node id instead of node object.

1.7 -

Add topic parameter to RPC

meth
ods.

1.8
-

Add
char

1.9
-

Add
de-
stroy

1.10
-

Re-
mov
get_

1.11
-

Add
get_
set_

1.12
-

val-
i-
date
do_v
re-
plac
by
sin-
gle

venc
meth

1.13
-

Add

update

1.14
-

Add
drive

1.15
-

Add
rebuild
parameter
to
do_r

1.16
-

Add
get_

1.17
-

Add
set_
get_
and

get_

1.18
-

Remov
char

1.19
-

Cha
re-

turn
valu
of
ven-
dor_
and

driv

1.20
-

Add
http_
pa-
ram-
e-
ter
to
ven-
dor_
and

driv

1.21
-

Add
get_
and

get_

1.22
-

Add
con-
fig-
driv
pa-
ram-
e-
ter
to
do_r

1.23
-

Add do_

1.24 -

Add in- spec meth

1.25 -

Add de- stroy

1.26 -

Add con- tinu

1.27 -

Con vert con- tinu to cast

1.28 -

Cha ex- cep- tions raise by de- stroy

1.29 -

Cha
re-
turn
valu
of
ven-
dor_
and

drive
to
a
dic-
tio-
nary

1.30
-

Add
set_
and

get_

1.31
-

Add
Ver-
sion
Ob-
jects
in-
di-
rec-
tion
API

methods:

obje
ob-
ject_
and

obje

1.32
-

Add do_

1.33 -

Add up- date and de- stro port grou

1.34 -

Add hear beat

1.35 -

Add de- stro and up- date

1.36 -

Add cre- ate_

1.37 -

Add de- stro and up- date

change_node_power_state	1.38 - Add vif_ vif_ vif_ 1.39 - Add time out optiona parameter to 1.40 - Add inject_ 1.41 - Add create_ 1.42 - Add optiona agen to hear beat 1.43 -

Add do_r do_r and can_

1.44 -

Add add_ and re- mov

1.45 -

Add con- tinu

1.46 -

Add re- set_ to up- date

1.47 -

Add sup- port for con- duc- tor grou

1.48 -

Add allocation API

1.49 -

Add get_ and agen arguments to heartbeat

1.50 -

Add set_ get_ and

get_

1.51 -

Add agen to heartbeat

1.52 -

Add deploy step arguments to pro-

vi-

sioning

1.53
-

Add
dis-
able
to
do_

1.54
-

Add
op-
tion
agen
and
agen
to
hear
beat

RPC_API

add_noc
Add
or
re-
plac
trait
for
a
node

Parame

- **con**
re-
ques
con-
text.
- **nod**
node
ID

or UUI

- **tra** a list of trait to add to the node
- **rep** True to re- plac all of the node trait
- **top** RPC topic De- fault to self.

Raises
In- valid Pa- ram- e- ter- Valu if addi the trait wou ex-

ceed the per-node traits limit.

Raises
Nod

Lock if node is lock by another conductor.

Raises

Nod NotFour if the node does not exist.

can_ser

Return whe the RPCAF supports the create_ meth

can_ser

Return whe the RPCAF supports node rescue meth ods.

change_

Cha a node pow state

Syn- chro ac- quir lock and start the con- duc- tor back grou task

to change power state of a node.

Paramet

- **con** re- ques con- text.
- **nod** node id or uuid
- **new** one of ironi pow state val- ues
- **tim** time

out (in seconds positive integer (>

0) for any power state. `None` indicates to use default timeout.

- **top** RPC topi De- fault to self.

Raises
NoF duc- tor- Wor whe ther is no free work to start asyn

task.

continu
Signal to conductor service to start the next clea

ing action.

NO this is an RPC cast, ther will be no response or ex-

ception raised by the conductor for this RPC.

Parame

- **con** request context.
- **nod** node id or uuid
- **top** RPC topi Defaul to self.

continu Signal to conductor service to start the next de-

ployment action.

NO this is an RP cast, ther will be no re- spo or ex-

ception raised by the conductor for this RPC.

Parame

- **con** re- ques con- text.
- **nod** nod id or uuid
- **top** RPC topi De- faul to self.

create_

Cre- ate an al- lo- ca- tion.

Paramet

- **`con`** request context.
- **`all`** an allocation object.
- **`top`** RPC topi Defaul to self.

`create_`

Synchro have a conductor validate and create a node.

Create the node information in

a node object.

the data and return

Paramet

- **con** request context.
- **nod** a created (but not save node object.
- **top** RPC topi Default to self.

Returns created node object.

Raises InterfaceFounEntrypoin if

validation fails

for any dynamic interfaces (e.g. network_interface).

Raises

NoValid DefaultForInterface if no default can be

calculated for some interfaces, and explicit values must be provided.

create_

Synchro have a conductor validate and create

a port.

Create the port in formation in the data and return a port object. The conductor will lock related node and trigger specific driver actions if they are needed.

Parame

- **con** request context.
- **por** a created (but not save port object.
- **top** RPC topic Default to self.

Returns created port object.

destroy Dele an allocation.

Parame

- **con** request

context.

- **all** an allocation object.
- **top** RPC topic. Defaults to self.

Raises
Invalid State if the associated node is in the

wrong provision state to perform deallocation.

destroy
Delete a node.

Paramet

- **con** request context.
- **nod**

node id or uuid

- **top** RPC topic. Defaults to self.

Raises
Nod Lock if node is lock by another conductor.

Raises
Nod Associated if the node contains an instan

associated with it.

Raises
Invalid State if the node is in

the wron provision

state to perform deletion.

destroy

Dele a port

Parame

- **con** request context.
- **por** port object
- **top** RPC topic Defaults to self.

Raises

Nod Lock if node is lock by another conductor.

Raises

Nod Not-Foun

if the node associated with the port

does not exist.

destroy

Dele a port grou

Parame

- **con** request context.
- **por** portgrou object
- **top** RPC topic Defaults to self.

Raises

Nod Lock if node is lock by another

conductor.

Raises

Nod Not- Foun if the nod associated with the port-

group does not exist.

Raises

Port grou Notl if port- grou is not emp

destroy

Dele a volume connector.

Dele the volume connector. The conductor will lock related node during this operation.

the

Paramet

- **con** reques context
- **con** volume connector object
- **top** RPC topi Defaul to self.

Raises
Nod Lock if node is lock by anothe conductor

Raises
Nod Not-Four if the node as-

sociated with the con-

nector does not exist

Raises

VolumeCon nectorN Four if the volume connector

cannot be found

destroy

Dele a volume target.

Parame

- **con** request context
- **tar** volume target object
- **top** RPC

topic. Default to self.

Raises
Node Lock if node is lock by another conductor

Raises
Node Not-Foun if the node associated with the target does not exist

Raises
Volume get-Not-Foun if the volume target cannot be found

do_node

Signal to conductor service to perform manual cleaning on a node.

Parame

- **`con`** reques context.
- **`nod`** node ID or UUI
- **`cle`** a list of clea step dictionaries.
- **`dis`** Whe to skip booting ram for clea

ing.

- **top** RPC topi De-fault to self.

Raises
In-valic Pa-ram-e-ter-Valu if val-i-da-tion of

power driver interface failed.

Raises
In-valic State if clea ing can not be per-form

Raises
Nod Mai te-nanc if node is in mai te-nanc mod

Raises
Nod
Lock
if
node
is
lock
by
another
conductor.

Raises
NoF
conductor-
Wor
whe
there
is
no
free
work
to
start
asyn

task.

do_node

Signal to conductor service to perform a deployment.

Parame

- **con**
re-

ques con- text.

- **nod** node id or uuid
- **reb** True if this is a re- build re- ques
- **con** A gzip and base en- code con- fig- drive
- **top** RPC topic De- fault to self.
- **dep** De- ploy steps

Raises
In- stanc ploy

Fail-
ure

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
val-
i-
da-
tion
fails

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing

Raises
NoF
duc-
tor-
Wor
whe
there
is
no
free
work
to
start
asyn

task.

The
node
mus

already be configured and in the appropriate undeployed state before this method is called.

do_node

Signal to conductor service to perform a rescue.

Parame

- **con** request context.
- **nod** node ID or UUI
- **res** A strin representing

the pass word to be set

inside the rescue environment.

- **top** RPC topi De- fault to self.

Raises
In- stan cue- Fail- ure

Raises
NoF duc- tor- Wor whe ther is no free worl to start asyn

task.

The node mus al- read be con- fig- ured and in the ap-

pro-

priate state before this method is called.

do_node

Signal to conductor service to tear dow a deployment.

Paramet

- **con** request context.
- **nod** node id or uuid
- **top** RPC topic Defaul to self.

Raises

In- stan ploy Failure

Raises

In- valid

Pa-
ram-
e-
ter-
Valu
if
val-
i-
da-
tion
fails

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing

Raises
NoF
duc-
tor-
Wor
whe
ther
is
no
free
worl
to
start
asyn

task.

The
node
mus
al-
read
be
con-
fig-
ured

and in the appropriate deployed state before this method is called.

do_node

Signal to conductor service to perform an unrescue.

Paramet

- **con** request context.
- **nod** node ID or UUI
- **top** RPC topi Defaul to self.

Raises

Instanc UnrescueFailure

Raises

NoF ductorWor whe there is no free work to start asyn task.

The node mus already be configured and in the appropriate state before this method is called.

do_prov

Signal to conductor service to perform the give action on a node.

Parame

- **con** reques context.
- **nod** node id or uuid
- **act** an action. One of ironi
- **top** RPC topi Default to self.

Raises
InvalidParameterValu

Raises
NoF ductorWor whe there is no free work

to start asyn

task.

Raises

In- valid State if the re- ques ac- tion can not be per-

formed.

This en- cap- su- lates som pro- vi- sion ing ac- tions in a

single call.

`driver_`

Pass vend spec calls whic dont spec ify a node to a drive

Han dles drive level ven dor pass calls Thes calls dont re quire a

node UUID and are executed on a random conductor with the specified driver. If the method mode is async the conductor will start background worker to perform vendor action.

Parame

- **con** re ques con text.
- **dri** nam of the drive on whic to call the meth
- **dri** nam of the ven dor meth for use by the drive

- **htt** the HTT meth used for the request
- **inf** data to pass thro to the driv
- **top** RPC topic. Default to self.

Raises
In-valid-Param-eter-Valu for param-eter errors.

Raises
Miss-ing-Param-eter-

Valu if a re- quir pa- ram-

eter is missing

Raises

Un- sup- port- ed- Driv ten- sion if the driv does have a

vendor interface, or if the vendor interface does not support the specified driver_method.

Raises

Driv Not- Fou if the sup- plied driv is not load

Raises

NoF duc- tor- Wor whe ther is no free work to start asyn

task.

Raises
In-
ter-
face
Foun
nEn-
try-
poin
if
the
de-
fault
in-
ter-

face for a hardware type is invalid.

Raises
No-
Vali
De-
fault
ForI
ter-
face
if
no
de-
fault
in-
ter-

face implementation can be found for this drivers vendor interface.

Returns

A
dic-
tio-
nary
con-
tain-
ing:

return
The
re-
spon
of
the
in-
voke
ven-
dor

meth

async

Boo valu Whe the meth was invoke asyn chro (Tru or synchronously (False). When invoked asynchronously the response will be always None.

attach

Boo valu Whe to attach the respon of the invoke vendor method to the HTTP response object (True) or return it in the response body (False).

`get_boc`

Get the current boot device

Returns the current boot device of a

node

Paramet

- **con** re- ques con- text.
- **nod** node id or uuid
- **top** RPC topic De- fault to self.

Raises
Nod Lock if node is lock by an- othe con- duc- tor.

Raises
Un- sup- port- ed- Driv ten- sion if the node drive

does support management.

Raises

InvalidParameterValu whe the wron driv info is specified.

Raises

MissingParameterValu if missing supplie info

Returns

a dictionary containing:

boot_

the boot device one of *iro*

com
boo
or
Non
if
it

is unknown.

persis
Whe
the
boot
de-
vice
will
per-
sist
to
all
fu-
ture
boot

or not, None if it is unknown.

get_co
Get
the
con-
duc-
tor
whic
the
node
is
map
to.

Parame
nod
a
node
ob-
ject.

Returns
the
con-
duc-
tor
host
nam

Raises
No-

Vali
Hos

get_cor

Get connection information about the console

Paramet

- **con** request context.
- **nod** node id or uuid
- **top** RPC topi De- fault to self.

Raises

Un- sup- port- ed- Driv ten- sion if the node driv does support console.

Raises
InvaliParameterValu whe the wron driv info is specified.

Raises
MissingParameterValu if a requir parameter is missing

get_cu
Get RP topi nam for the current conductor.

get_dr
Get the

prop erties of the driv

Parame

- **con** request con text.
- **dri** nam of the driv
- **top** RPC topi Default to self.

Returns
a dictionary with <pro erty nam description: entries

Raises
Driv Not-Foun

get_dri
Re-

triev in-for-ma-tion abou ven-dor meth-ods of the given driver.

Parame

- **con** an ad-min con-text.
- **dri** nam of the driv
- **top** RPC topi De-fault to self.

Raises
Un-sup-port-ed-Driv ten-sion if cur-rent driv does

not

have vendor interface.

Raises

Driv
Not-
Foun
if
the
sup-
plie
driv
is
not
load

Raises

In-
ter-
face
Foun
nEn-
try-
poin
if
the
de-
fault
in-
ter-

face for a hardware type is invalid.

Raises

No-
Vali
De-
fault
ForI
ter-
face
if
no
de-
fault
in-
ter-

face implementation can be found for this drivers vendor interface.

Returns

dic-
tio-
nary
of

<me nam meta data entries

get_inc

Get node hardware com ponent indicator state

Parame

- **con** request context.
- **nod** node id or uuid
- **com** The hardware com ponent one of *irc com com*
- **ind**

Indicator IDs, as reported by *get_*

- **top** RPC topic. Defaults to self.

Raises Nod Lock if node is lock by another conductor.

Raises Unsupported- Driv tension if the node drive does support management.

Raises Invalid

Pa-
ram-
e-
ter-
Valu
whe
the
wron
driv
info
is

specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
miss
ing
sup-
plied
info

Returns
In-
di-
ca-
tor
state
one
of
mod

`get_noc`
Re-
triev
in-
for-
ma-
tion
abou
ven-
dor
meth
ods
of
the

given node.

Paramet

- **con** an admin context.
- **nod** the id or uuid of a node
- **top** RPC topic De- fault to self.

Returns dictionary of <me nam meta data entries

get_noc Requests the node from the conductor with an ager to-

ken

Parame

- **con** re- ques con- text.
- **nod** node ID or UUI
- **top** RPC topi De- fault to self.

Raises
Nod Locl if node is lock by an- othe con- duc- tor.

Returns
A Nod ob- ject with ager to- ken.

get_rai

Get the logical disk properties for RAI configuration.

Gets the information abou logical disk properties which can be specified in the input RAID configuration.

Paramet

- **con** reques context.
- **dri** nam of the driv
- **top** RPC topi Default

to self.

Raises
Unsupported-Driv ten-sion if the driv does sup-port

RAID configuration.

Raises
Interface-Four nEntry-poin if the default interface for a hardware type is invalid.

Raises
NoValid Default ForI terface if no default interface implementation can be found for this drivers RAID interface.

Returns
A

dictionary containing the properties that can be mentioned for logical disks and a textual description for them.

get_ran

Get an RPC topic for a random conductor service

get_sup

Get the list of supported devices

Returns the list of supported boot devices of a node

Parame

- **con** request context.
- **nod** node id or uuid
- **top** RPC topic. Defaults to self.

Raises
Nod Lock if node is lock by another conductor.

Raises
Unsupported-Driv tension if the node drive does sup-

port management.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
whe
the
wron
drive
info
is

specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
miss
ing
sup-
plied
info

Returns
A
list
with
the
sup-
porte
boot
de-
vice
de-
fined
in
irc

common.boot_devices.

get_sup
Get
node
hard

ware com- po- nent and thei in- di- ca- tors.

Parame

- **con** re- ques con- text.
- **nod** node id or uuid
- **com** The hard ware com po- nent one of *irc com com*
- **top** RPC topi De- fault to self.

Raises
Nod Locl

if node is lock by another conductor.

Raises
UnsupportedDriv tension if the node driv does sup-

port management.

Raises
InvalidParameterValu whe the wror driv info is

specified.

Raises
MissingParameterValu if miss

ing
sup-
plied
info

Returns

A
dic-
tio-
nary
of
hard
ware
com
po-
nent
(*ir*
com
com
as

keys with indicator IDs as values.

(continues on next page)

(continued from previous page)

```
```

`get_top`

Get the RPC topi for the conductor service the nod

is mapped to.

Paramet

`nod` a nod object.

Returns

an RPC topi strin

Raises

NoVali Hos

`get_top`

Get RPC topi nam for a conductor supporting the

given driver.

The topi is used to rout message to the conductor supporting the specified driver. A conductor is selected at random from the set of qualified conductors.

Parame

`dri` the nam of the driv to rout to.

Returns

an RPC topi strin

Raises

Driv Not- Four

`heartbe`

Process a node hear beat

Parame

- **`con`** reques context.

- **nod** node ID or UUI
- **cal** URI to reac back to the ram
- **top** RPC topi Default to self.
- **age** random generated validation token.
- **age** the version of the ager that is

hear beat ing

- **age** TLS cer- tifi- cate for the agen
- **age** The sta- tus of the agen that is hear beat ing
- **age** Op- tiona mes- sage de- scrib ing the agen sta- tus

Raises
In- valid Pa- ram- e- ter- Valu if an in-

valic
ager
to-

ken is received.

inject_

In-
ject
NMI
for
a
node

In-
ject
NMI
(Nor
Mas
able
In-
ter-
rupt
for
a
node
im-
me-

diately. Be aware that not all drivers support this.

Parame

- `con`
  re-
  ques
  con-
  text.
- `nod`
  node
  id
  or
  uuid
- `top`
  RPC
  topic
  De-
  fault
  to
  self.

Raises
Nod Lock if node is lock by another conductor.

Raises
Unsupported-Driv tension if the node driv does sup-

port management or management.inject_nmi.

Raises
Invalic Parameter-Valu whe the wror driv info is

specified or an invalid boot device is specified.

Raises
Missing-Parameter-

Valu if miss ing sup- plied info

inspect

Sig- nals the con- duc- tor ser- vice to per- form hard ware

introspection.

Paramet

- **con** re- ques con- text.
- **nod** node id or uuid
- **top** RPC topic De- fault to self.

Raises

Nod Lock if

node
is
lock
by
an-
othe
con-
duc-
tor.

Raises
Harc
ware
spec
tion-
Fail-
ure

Raises
NoF
duc-
tor-
Wor
whe
there
is
no
free
worl
to
start
asyn

task.

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion
if
the
node
drive
does
sup-

port inspection.

Raises
In-
valid

Stat if inspec is not a valid action to do in the current state.

object_

Perform an action on a VersiondOb ject instan

We wan any conductor to handle this, so it is intentional that there is no topic argument for this method.

Parame

- **con** The context with

whic to perform the action

- **obj** The object instance on whic to perform the ac-

tion

- **obj** The nam of the action meth to call
- **arg** The positiona arguments to the action meth
- **kwa**

The key-word ar-gu-men to the ac-tion meth

Raises

Noth ple-men ed-Er-ror whe an op-er-a-tor mak

an error during upgrade

Returns

A tu-ple with the up-date mad to the ob-ject and

the result of the action method

object_

Per-form a back port of an object instan

The default behavior of the base Version dObjectSerializer, upon receiving an object with a version newer than what is in the local registry, is to call this method to request a backport of the object.

We wan any conductor to handle this, so it is intentional that there is no topic argument for this method.

Paramet

- `con` The context with whic to perform the backport

- **obj** An instance of a VersionedObject to be back

ported

- **obj** A dict of {objname: version} mappings

Raises
Not implemented Error when an operator mak

an error during upgrade

Returns
The downgraded instance of ob-

jinst

`object_`

Perform an action on a Version dObject clas

We wan any conductor to handle this, so it is intentional that there is no topic argument for this method.

Parame

- `con` The context with whic to perform the action
- `obj` The reg-

istry
nam
of
the
ob-
ject

- **obj**
  The
  nam
  of
  the
  ac-
  tion
  meth
  to
  call
- **obj**
  A
  dict
  of
  {ob-
  j-
  nam
  ver-
  sion
  map
  ping
- **arg**
  The
  po-
  si-
  tiona
  ar-
  gu-
  men
  to
  the
  ac-
  tion
  meth
- **kwa**
  The
  key-
  word
  ar-

gu-
men
to
the
ac-
tion
meth

Raises

Noth
ple-
men
ed-
Er-
ror
whe
an
op-
er-
a-
tor
mak

an error during upgrade

Returns

The
re-
sult
of
the
ac-
tion
meth
whic
may
(or
may
not)

be an instance of the implementing VersionedObject class.

remove_

Re-
mov
som
or
all
trait
from
a
node

Paramet

- **con** re- ques con- text.
- **nod** node ID or UUI
- **tra** a list of trait to re- mov from the node or Non

If None, all traits will be removed from the node.

- **top** RPC topic De- fault to self.

Raises
Nod Lock if node is lock by an- othe con- duc- tor.

Raises

Nod Not-Four if the node does not exist.

Raises

Nod Trai Not-Four if one of the trait is not foun

set_boo

Set the boot device for a node

Set the boot device to use on next reboot of the node

Be aware that not all drivers support this.

Paramet

- **con** reques context.
- **nod** node id or uuid
- **dev** the boot device one of *iro com boo*
- **per** Whe to set next boot or mak the char permanent

Default: False.

- **top** RPC topic Default to self.

Raises
Nod Lock

if node is lock by another conductor.

Raises

Unsupported-Driv tension if the node drive does sup-

port management.

Raises

Invalid Parameter-Valu whe the wron drive info is

specified or an invalid boot device is specified.

Raises

Missing-Parameter-Valu if miss

ing supplied info

set_co

Enable the console

Parame

- **con** request context.
- **nod** node id or uuid
- **top** RPC topic Default to self.
- **ena** Boo valu whe the console is enabled or disable

Raises

Un-

sup-
port-
ed-
Driv
ten-
sion
if
the
node
drive
does
sup-

port console.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
whe
the
wron
drive
info
is

specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quire
pa-
ram-

eter is missing

Raises
NoF
duc-
tor-
Wor
whe
there

is no free worl to start asyn task.

set_inc

Set node hardware component indicator to the desired state.

Paramet

- **con** request context.
- **nod** node id or uuid
- **com** The hardware component one of *irc com*

com

- **ind** Indicator IDs, as reported by *get_*
- **sta** Indicator state one of mod
- **top** RPC topic Defaults to self.

Raises
Nod Lock if node is lock by another conductor.

Raises
Unsupported-

Driv ten- sion if the node drive does sup-

port management.

Raises
In- valid Pa- ram- e- ter- Valu when the wron drive info is

specified or an invalid boot device is specified.

Raises
Miss ing- Pa- ram- e- ter- Valu if miss ing sup- plied info

set_tar
Stor the tar- get RAI con- fig- u- ra- tion

on the node

Stor the tar- get RAI con- fig- u- ra- tion on node

Paramet

- **con** re- ques con- text.
- **nod** node id or uuid
- **tar** Dic- tio- nary con- tain- ing the tar- get RAI con- fig-

uration. It may be an empty dictionary as well.

- **top** RPC topic

Defaul to self.

Raises
UnsupportedDriv tension if the node driv does support RAID configuration.

Raises
InvalidParameterValu if validation of target raid config fails.

Raises
MissingParameterValu if som require parameters are missing.

Raises

Nod Lock if node is lock by another conductor.

update_

Synchro have a conductor update the node information.

Update the node information in the data and return

a node object. The conductor will lock the node while it validates the supplied information. If driver_info is passed, it will be validated by the core drivers. If instance_uuid is passed, it will be set or unset only if the node is properly configured.

Note that pow shou not

be pass via this meth Use char for initiating driver actions.

Parame

- **con** request context.
- **nod** a char (but not save node object.
- **top** RPC topic Defaults to self.
- **res** whe to reset hardware interfaces to their

defaults.

Returns
updated node object, including all field

Raises
NoVali DefaultForInterface if no default can be calculated for some interfaces, and explicit values must be provided.

update_
Synchro have a conductor update the port in- formation.

Updates the port in- formation in the data and return

a port object. The conductor will lock related node and trigger specific driver actions if they are needed.

Paramet

- **con** request context.
- **por** a chan (but not save port object.
- **top** RPC topic Defaults to self.

Returns updated port object, including all field

update_

Synchro have a conductor updat the port grou in-

formation.

Update the port grou information in the data and re-

turn a portgroup object. The conductor will lock related node and trigger specific driver actions if they are needed.

Parame

- **con** request context.
- **por** a char (but not save portgroup object.
-

top
RPC topic. Defaults to self.

Returns
updated portgroup object, including all field

update_
Updates the volume connectors information.

Updates the volume connectors information in the database and return a volume connector object. The conductor will lock the related node during this operation.

Parame

- **con** request context
- **con** a char (but not saved volume connector object
- **top** RPC topic Default to self.

Raises
In-valid Parameter-Valu if the volume connec-

tors UUID is being changed

Raises
Nod Lock if node

is lock by another conductor

Raises
Nod Not-Foun if the node associated with the con-

nector does not exist

Raises
Vol-ume Con nec-torN Foun if the volume connector

cannot be found

Raises
Vol-ume Con nec-torT pe-An-dI-dAl-read ists

if another connector already exists with the same values for type and connector_id fields

Returns
updated volume connector object, including all fields.

`update_`
Updates the volume targets information.

Updates the volume targets information in the data and return a volume target object. The conductor will lock the related node during this operation.

Paramet

- `con`

request context

- **tar** a char (but not save volume target object
- **top** RPC topic Defaults to self.

Raises
In-valid Parameter-Valu if the volume targets

UUID is being changed

Raises
Nod Lock if the node is already

lock

Raises
Nod Not- Fou if the node as- so- ci- ated with the vol-

ume target does not exist

Raises
Vol- ume get- Not- Fou if the vol- ume tar- get can- not

be found

Raises
Vol- ume get- Boo dex- Al- read ists if a vol- ume tar-

get already exists with the same node ID and boot index values

Returns
up- date

volume target object, including all field

validat

Validate the *core* and *standardized* interfaces for drivers.

Parame

- **con** request context.
- **nod** node id or uuid
- **top** RPC topic Defaults to self.

Returns
a dictionary containing the results of each interface validation.

`vendor_`
Receive reques for venc spec actions

Synchro validate drive specific info or get drive status, and if successful invokes the vendor method. If the method mode is async the conductor will start background worker to perform vendor action.

Parame

- `con` request context.

- **nod** node id or uuid
- **dri** nam of meth for driv
- **htt** the HTT meth used for the request
- **inf** info for node driv
- **top** RPC topi Default to self.

Raises
In-valid Parameter-Valu if supplied info

is not

valid.

Raises

Miss ing- Pa- ram- e- ter- Valu if a re- quir pa- ram-

eter is missing

Raises

Un- sup- port- ed- Driv ten- sion if cur- rent driv does not

have vendor interface.

Raises

NoF duc- tor- Wor whe there is no free work to start asyn

task.

Raises

Nod Lock if node is lock by an- othe con- duc- tor.

Returns

A dic- tio- nary con- tain- ing:

return The re- spon of the in- voke ven- dor meth

async Boo valu Whe the meth was in- voke asyn chro (Tru or syn- chronously (False). When invoked asynchronously the response will be always None.

attach Boo

valu Whe to attach the respor of the invoke vendor method to the HTTP response object (True) or return it in the response body (False).

vif_att

Attach VIF to a node

Paramet

- **con** request context.
- **nod** node ID or UUI
- **vif** a dictionary representing VIF object. It must have an id key, whose value is a unique identifier for that VIF.

- **top** RPC topic De- fault to self.

Raises Nod Lock if node has an ex- clu- sive lock held on it

Raises Net- work Er- ror, if an er- ror oc- curs dur- ing at-

taching the VIF.

Raises In- valid Pa- ram- e- ter- Valu if a pa- ram- e-thats required for VIF attach is wrong/missing.

ter

vif_det

De-
tach
VIF
from
a
node

Parame

- **con**
re-
ques
con-
text.
- **nod**
node
ID
or
UUI
- **vif**
an
ID
of
a
VIF.
- **top**
RPC
topic
De-
fault
to
self.

Raises

Nod
Lock
if
node
has
an
ex-
clu-
sive

lock held on it

Raises
Network Error, if an error occurs during de-

taching the VIF.

Raises
Invalid Parameter Value if a parameter

thats required for VIF detach is wrong/missing.

vif_lis
List attach VIF for a node

Paramet

- **con** request context.

- **nod** node ID or UUI
- **top** RPC topi De- fault to self.

Returns
List of VIF dic- tio- nar- ies, each dic- tio- nary will have

an id entry with the ID of the VIF.

Raises
Net- work Er- ror, if an er- ror oc- curs dur- ing list-

ing the VIFs.

Raises
In- valid Pa-

rameterValue if a parameter

thats required for VIF list is wrong/missing.

ironic.conductor.steps module

`ironic.`

Find an identical step in the list of step

`ironic.`

Compare step ignoring their priority.

`ironic.`

Set up the node with clea step information for clea

ing.

For auto-mat clea ing, get the clea step from the driv For

manual cleaning, the users clean steps are known but need to be validated against the drivers clean steps.

Paramet

dis

If *True* only step with re-quir are ac-cept

Raises

In-valic Pa-ram-e-ter-Valu if there is a prob lem

with the users clean steps.

Raises

Nod Clea ing-

Fail-
ure
if
there
was
a
prob-
lem
get-
ting

the clean steps.

`ironic.`

Set
up
the
node
with
de-
ploy
men
step
in-
for-
ma-
tion

for deploying.

Get
the
de-
ploy
steps
from
the
drive

Paramet

res

Whe
to
re-
set
the
cur-
rent
step
to
the
first
one.

Raises

In-
stan
ploy
Fail-
ure
if
there
was
a
prob
lem
get-
ting

the deployment steps.

`ironic.`

Val-
i-
date
the
user
de-
ploy
step
and
the
de-
ploy
tem-

plates for a node.

Paramet

- **tas**
A
Task
ager
ob-
ject
- **dep**
De-
ploy
step
to
val-

idate Optiona If not provided then will check nodes driver internal info.

- **ski** whe skip missing step that are not yet avai able at the time of validation.

Raises
Invalic ParameterValu if the instan has trait that map to deploy steps that are unsupported by the nodes driver interfaces or user deploy steps are unsupported by the nodes driver interfaces

Raises
Instan ploy Failure if ther was a prob

lem getting the deploy steps from the driver.

ironic.conductor.task_manager module

A context manager to perform a series of task on a set of resources.

`Tas` is a context manager created on-dem to allow synchronized access to a node and its resources.

The `Tas` will, by default, acquir an exclusive lock on a node for the duration that the TaskManager instance exists. You may create a TaskManager instance without locking by passing shared=True when creating it, but certain operations on the resources held by such an instance of TaskManager will not be possible. Requiring this exclusive lock guards against parallel operations interfering with each other.

A shar lock is useful whe performing noninter operations, such as validating the driver interfaces.

An exclusive lock is store in the data to coordinate between *ironic.conductor.manager* instances, that are typically deployed on different hosts.

`Tas` methods, as well as driv methods, may be decorated

to determine whether their invocation requires an exclusive lock.

The
Task
ager
instan
expose
certain
node
resour
and
prop

erties as attributes that you may access:

task.cont
The
context
pass
to
Task
ager

task.shar
Fals
if
Nod
is
lock
True
if
it
is
not
lock
(The
shar

kwarg arg of TaskManager())

task.node
The
Nod
object

task.port
Port
belong

ing to the Nod

task.port
Port grou be- long ing to the Nod

task.volu
Stor age con- nec- tors be- long ing to the Nod

task.volu
Stor age tar- gets as- sign to the Nod

task.driv
The Driv for the Nod or the Driv base on the driv kwa

of TaskManager().

Example usage:

```
wit
 ↪t
 ↪m
 ↪a
 ↪␣
 ↪n
 ↪i
 ↪␣
 ↪p
 ↪'
 ↪o
 ↪'
 ↪a
 ↪t
␣
 ↪␣
 ↪␣
 ↪␣
 ↪t
 ↪d
 ↪p
 ↪p
 ↪o
 ↪n
```

If you need to execute task-requ code in a back grou

thread, the TaskManager instance provides an interface to handle this for you, making sure to release resources when the thread finishes (successfully or if an exception occurs). Common use of this is within the Manager like so:

```
wit
 ↪t
 ↪m
 ↪a
 ↪␣
 ↪n
 ↪i
 ↪␣
 ↪p
 ↪'
 ↪w
```

(continues on next page)

(continued from previous page)

```
↪        utils.node_power_action, task, new_state)
```

```
␣
↪␣
↪␣
↪␣
↪
↪<
↪s
↪w
↪
␣
↪␣
↪␣
↪␣
↪t
↪s
↪a
↪_
↪s
↪w
↪
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
```

All exceptions that occur in the current Gree Thre as part of the spawn handling are re-raised. You can specify a hook to execute custom code when such exceptions occur. For example, the hook is a more elegant solution than wrapping the with task_manager.acquire() with a try..exception block. (Note that this hook does not handle exceptions raised in the background thread.):

```
def
↪o
↪e
␣
↪␣
↪␣
↪␣
↪i
↪i
↪␣
↪E
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪.
↪.
↪.
↪

wit
↪t
↪m
↪a
↪␣
↪n
↪i
↪␣
↪p
↪'
↪w
↪'
↪a
↪t
␣
↪␣
↪␣
↪␣
↪
↪<
↪s
↪w
↪
␣
↪␣
↪␣
↪␣
↪t
↪s
↪s
↪e
↪h
↪e
```

(continues on next page)

(continued from previous page)

```
↪        utils.node_power_action, task, new_state)
```

```
␣
↪␣
↪␣
↪␣
↪t
↪s
↪a
↪_
↪s
↪w
↪
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
```

class i

Base
`obj`

Con
text
man
ager
for
task

This
class
wrap
the
lock
ing,
driv
load
ing,
and
ac-

qui-
si-
tion

of related resources (eg, Node and Ports) when beginning a unit of work.

downgra

Dow
grad
the
lock
to
a
shar
one.

load_dr

propert

propert

propert

process

Pro-
cess
the
give
even
for
the
task
cur-
rent
state

Parame

- **eve**
the
nam
of
the
even
to
pro-
cess
-

cal optiona callback to invoke upon even transition

- **cal** optiona args to pass to the callback meth

- **cal** optiona kwa to pass to the callback meth

- **err** optiona error handler to invoke if the callback fails, eg. because there are no workers available (err_handler should accept arguments node, prev_prov_state, and prev_target_state)

- **`tar`** if specified, the target pro- vi- sion state for the node. Otherwise, use the target state from the fsm

Raises
In- valid State if the even is not al- lowe by the associated state machine

`release`
Un- lock a node and re- lease re- sour

If an exclusive lock

is held un- lock the nod Re- set

attributes to make it clear that this instance of TaskManager should no longer be accessed.

set_spa

Cre- ate a hoo to han- dle ex- cep- tion whe spaw ing

a task.

Cre- ate a hoo that gets call upo an ex- cep- tion be- ing

raised from spawning a background thread to do a task.

Parame

- **_on** a call ob- ject, its first

parameter shou ac-

cept the Exception object that was raised.

- **arg** additional args pass to the calla object.
- **kwa** additional kwa pass to the calla object.

spawn_a
Call this to spaw a threa to complete the task

The spec ified meth will

be calle whe the Task ager instan exits.

Paramet

- **_sp** a meth that returns a Gree Thre object
- **arg** args pass to the meth
- **kwa** additiona kwa pass to the meth

upgrade Upgrad a shar lock to an exclusive lock

Also reloa node object from the data If lock is already ex-

clusive only changes the lock purpose when provided with one.

Parame

- **pur** optionally char the purpose of the lock
- **ret** whe to retry lock ing if it fails the class level valu

is used by default

Raises
Nod

Lock if an exclusive lock remain on the node

after node_locked_retry_attempts

propert

propert

`ironic.`

Shortcut for acquiring a lock on a Nod

Paramet

con Request context.

Returns

An instance of *Tas*

`ironic.`

Decorator to re-

quir

an

exclusive

lock

Decorated

func

tions

mus

take

a

Tas

as

the

first

parameter. Decorated class methods should take a *TaskManager* as the first parameter after self.

ironic.conductor.utils module

`ironic.`

Set

node

state

whe

a

task

was

abor

due

to

conductor

take over.

Paramet

tas

a

Task

ager

instanc

`ironic.`

Add

a

se-
cret
to-
ken
to
driv
for
IPA
ver-
i-
fi-

cation.

Paramet

-
  nod
  Nod
  ob-
  ject

-
  pre
  Boo
  valu
  de-
  fault
  Fals
  whic
  in-
  di-
  cate
  if
  the
  to-

ken should be marked as pregenerated in order to facilitate virtual media booting where the token is embedded into the configuration.

`ironic.`

Che
that
the
ager
is
likel
alive

The
meth
then
chec
for
the

last agen hear beat and if it oc-

cured within the timeout set by [deploy]fast_track_timeout, then agent is presumed alive.

Paramet

- **nod** A node ob- ject.
- **tim** Hea beat time out, de- faul to *fast_*

`ironic.`

Buil a con- fig- driv from pro- vide meta net- work and user

If uuid or nam are not pro- vide

in the meta they de- faul

to the nodes uuid and name accordingly.

Paramet

- **nod** an Iron node ob- ject.
- **con** A con- fig- driv as a dict with keys `met` `net` `use`

and `vendor_data` (all optional).

Returns A gzip and base en- code con- fig- driv as a strin

`ironic.`

Put a faile node in CLE FAI and mai te- nanc (if need

Paramet

- **tas** a Task ager in- stan
- **log** Mes sage to be logg
- **err** Mes sage for the user Op- tion if not pro- vide *logn*

is used.

- **tra** Whe to log

a trace back De- fault to Fals

- **tea** Whe to clea up the PXE and DHC files af- ter clea

ing. Default to True.

- **set** Whe to set node to faile state De- fault to True

- **set** Whe to set main te- nanc mod If Non main te- nanc

mode will be set if and only if a clean step is being executed on a node.

`ironic.`

Clea de- ploy task af- ter time out.

Paramet

tas a Task agen in- stan

`ironic.`

Clea a clea ing task af- ter time out.

Paramet

tas a Task agen in- stan

`ironic.`

Clea res- cue task af- ter time out.

Paramet

tas a Task agen in- stan

`ironic.`

Put a faile node in DE- PLO FAI

Paramet

- **tas** the task
- **log** mes- sage to be logg
- **err** mes- sage for the user
- **tra** Boo True to log a trace back
- **cle** Boo True to clear up

`ironic.`

A decorator for failing operation on failure.

`ironic.`

Chec if the operation can be a strea lined deployment sequence.

This is main focuse on ensuring that we are able to quickly sequence through operations if we already have a ramdisk heartbeating through external means.

Paramet

tas

Task agen

object

Returns
True if [deploy is set to True no iSCS boot configuration is present, and no last_error is present for the node indicating that there was a recent failure.

ironic.

Get any attach vif ID for the port

Paramet

por

The port object upon whic to chec for a vif reco

Returns
Returns a tuple of the

vif if foun and the use

of the vif in the form of a string, tenant, cleaning provisioning, rescuing.

Raises

In- valic Stat ex- cep- tion upor find- ing a port with a

transient state vif on the port.

`ironic.`

`ironic.`

`ironic.`

Hasl a sup- plied pass word

Paramet

`pas` pass word to be hash

`ironic.`

De- ter- mine if the to- ken

was generated for out

of band configuration.

Iron sup- port the ability to provide configuration

data to the agent through the a virtual floppy or as part of the virtual media image which is attached to the BMC.

This meth help us iden tify WH we did so as we dont need

to remove records of the token prior to rebooting the token. This is important as tokens provided through out of band means presist in the virtual media image, are loaded as part of the agent ramdisk, and do not require regeneration of the token upon the initial lookup, ultimately making the overall usage of virtual media and pregenerated tokens far more secure.

Paramet

`nod` Nod Object

Returns

True if

the token was pregenerated as indicated by the nodes driver_internal_info field. False in all other cases.

`ironic.`

Determine if an agen token is pres upon a node

Paramet

nod Nod object

Returns

True if an agen valu is pres in a node driv field

`ironic.`

Validate if

a supplie token is vali for the node.

Paramet

- **nod** Nod object
- **tok** A token valu to validate agai the driv field agent_secret_token.

Returns
True if the supplie token matc the token reco in the supplied node object.

`ironic.`
Chec

a fast trac is avai able

This meth first ensure that the nod and conductor configuration is valid to perform a fast track sequence meaning that we already have a ramdisk running through another means like discovery. If not valid, False is returned.

The meth then chec for the last agen hear beat and if it occured within the timeout set by [deploy]fast_track_timeout and the power state for the machine is POWER_ON, then fast track is permitted.

Paramet

`tas` Task agen object

Returns

True if the last

heart beat that was recorded was within the [deploy]fast_track_timeout setting.

ironic.

Generate a random salt with the indicator tag for password type.

Returns

a valid salt for use with cryp

ironic.

Do caching of bios settings if supported by driv

ironic.

Cache the vendor if it can be detecte

ironic.

Reac currently set boot mod from a node

Reac the boot mod for a node If boot mod cant be discov-

ered, *None* is returned.

Paramet

tas

a Task ager instanc

Raises

Driv Operationl or its deriv tive

in
case
of
driv

runtime error.

Raises

Un-
sup-
port
ed-
Driv
ten-
sion
if
cur-
rent
driv
does
not

have management interface or *get_boot_mode()* method is not supported.

Returns

Boo
mod
One
of
`irc`
`com`
`boc`
or
Non
if
boot
mod
cant

be discovered

`ironic.`

Cha
pow
state
or
re-
set
for
a
node

Per-
form
the
re-

ques pow action if the transition is

required.

Paramet

- **tas** a Task ager instanc containing the node to act

on.

- **new** Any pow state from ironi
- **tim** time out (in seconds pos itive integer (>

0) for any power state. `None` indicates to use default timeout.

Raises
In- valic Pa- ram- e- ter- Valu whe the wror state is spec

ified or the wrong driver info is specified.

Raises
Stor ageE ror whe a fail- ure oc- curs up- dat- ing the

nodes storage interface upon setting power on.

Raises
othe ex- cep- tions by the node pow drive if som thing wror

occurred during the power action.

`ironic.`

Set the

boot device for a node

If the node that the boot device char is being reques

for is in ADOPTING state, the boot device will not be set as that change could potentially result in the future running state of an adopted node being modified erroneously.

Paramet

- **tas** a Task ager instan
- **dev** Boo device Values are venc spec
- **per** Whe to set next boot or

mak the char per-ma-nent

Default: False.

Raises
In-valic Pa-ram-e-ter-Valu if the val-i-da-tion

of the ManagementInterface fails.

`ironic.`

Set the boot mod for a node

Sets the boot mod for a node if the node drive in-ter-face

contains a management interface.

If the node that

the boot mod char is being reques for

is in ADOPTING state, the boot mode will not be set as that change could potentially result in the future running state of an adopted node being modified erroneously.

Paramet

- **tas** a Task agen instan
- **mod** Boo mod Values are one of *irc com boo*

Raises
In valic Param eter Valu if the validation

of the ManagementInterface fails.

Raises

Driv Operationl or its derivtive in case of drive runtime error.

Raises
Unsupported-Driv tension if current drive does not have vendor interface or method is unsupported.

`ironic.`

Wait for node to be in new pow state

Paramet

- **tas** a Task agen instanc

- **new** the desired new pow state one of the pow state in

ironic.common.states.

- **tim** number of seconds to wait before giving up.

If not specified, uses the conductor.power_state_change_timeout config value.

Raises
PowerState Failure if time out

`ironic.`

`ironic.`

`ironic.`

Notify the conductor to resum an operation.

Paramet

- **tas** the task
- **ope** the operation, a strin

`ironic.`

Powers on node if it is powered off and has a Smart NIC port

Paramet **tas** A Task agen object

Returns

the previous pow state or Non if no char were mad

Raises
exception. if ager status didn matc the required sta-

tus after max retry attempts.

`ironic.`

Set the node pow state if error occurs

This hool gets calle upon an exception being raise whe spaw

ing the worker thread to change the power state of a node.

Paramet

- **e** the exception object that was raise
- **nod** an Iron node object.
- **pow** the pow state to set on the node

`ironic.`
Handle the pow state for a node reconfiguration.

Powers the node on if and only if it has a Sma NIC

port. Yields for the actual reconfiguration, then restores the power state.

Paramet

`tas` A Task agen object.

`ironic.`

Set the node provisioning state if error occurs

This hook gets calle upon an exception being raise ing the worker to do some provisioning to a node like deployment, tear down, or cleaning.

whe
spav

Paramet

- **e** the exception object that was raise
- **nod** an Iron node object.
- **pro** the provision state to be set on the node
- **tar** the target provision state to be set on

the

node.

ironic.

Help to re- mov the agen reco

ironic.

Help to re- mov res- cue pass word from a node

Re- mov res- cue pass word from node It save node by de- fault

If node should not be saved, then caller needs to explicitly indicate it.

Paramet

- **nod** an Iron node ob- ject.
- **sav**

Boo True (default to save the node Fals otherwise

`ironic.`

Clea rescue task after timeout or failure.

Paramet

- **`tas`** a Task agen instanc
- **`msg`** a message to set into node last_ field
- **`set`** a bool

flag to indicate if node needs to be

transitioned to a failed state. By default node would be transitioned to a failed state.

`ironic.`

Cha the node pow state if pow is not Non

Paramet

- **tas** A Task agen object
- **pow** pow state

`ironic.`

Che if node clear ing need to be skip for an spe-

node.

cific

Paramet

nod the node to consider

ironic.

Handle spaw ing error for node cleaning.

ironic.

Handle spaw ing error for node deploying.

ironic.

Handle spaw ing error for node rescue.

ironic.

Stor certificate receiv from the agen and returr its

path.

`ironic.`

Calculate the next step index and update the node

Paramet

- **tas** A Task agen object
- **ste** The type of steps to process clea or deploy

Returns
Index of

the next step

`ironic.`

Validate traits in instance

All traits in instance must also exist as node traits

Paramet

nod an Ironic node object.

Raises

InvalidParameterValue if the instance traits are

badly formatted, or contain traits that are not set on the node.

`ironic.`

Validate

the consistenc of phys ical networ of ports in a portgroup.

Validate the consistenc of a port phys ical network with other ports in the same portgroup. All ports in a portgroup should have the same value (which may be None) for their physical_network field.

During creatio or update of a port in a portgrou we apply the following validation criteria:

- If the portgrou has exist

ing port with dif- fer- ent

physical networks, we raise PortgroupPhysnetInconsistent. This shouldnt ever happen.

- If the port has a phys i- cal net- wor that is in-

consistent with other ports in the portgroup, we raise exception.Conflict.

If a port phys i- cal net- wor is Non this in- di- cate

that ironics VIF attachment mapping algorithm should operate in a legacy (physical network unaware) mode for this port or portgroup. This allows existing ironic nodes to continue to function after an upgrade to a release including physical network support.

Paramet

- **tas** a Task agen in- stan
-

por
a port object to be validate

Raises
Conflict if the port is a member of a portgrou

which is on a different physical network.

Raises
Portgrou Phys net- Inco sistent if the port portgrou has

ports which are not all assigned the same physical network.

`ironic.`
Che if the time is with the previ-

ous time out sec-

onds from now.

Paramet

- **val** a strin rep- re- sent- ing date and time or Non
- **tim** time out in sec- onds

`ironic.`

Re- mov tem- po- rary clea ing field from driv

`ironic.`

Re- mov tem- po- rary de- ploy men field from

driv

ironic.

Wip in- for- ma- tion that shou not sur- vive re- boot off.

ironic.

Re- mov agen URI and to- ken from the task

Module contents

ironic.conf package

Submodules

ironic.conf.agent module

ironic.

ironic.conf.anaconda module

ironic.

ironic.conf.ansible module

`ironic.`

ironic.conf.api module

`ironic.`

ironic.conf.audit module

`ironic.`

ironic.conf.auth module

`ironic.`

Add auth options to sample config

As these are dynamically registered at runtime this

adds options for most used auth_plugins when generating sample config.

`ironic.`

Register

sess

and auth relat op- tions

Reg- is- ters only ba- sic auth op- tions shar by all auth plu-

gins. The rest are registered at runtime depending on auth plugin used.

ironic.conf.cinder module

`ironic.`

`ironic.`

ironic.conf.conductor module

`ironic.`

ironic.conf.console module

`ironic.`

ironic.conf.database module

ironic.

ironic.conf.default module

ironic.

ironic.

ironic.conf.deploy module

ironic.

ironic.conf.dhcp module

ironic.

ironic.conf.disk_utils module

ironic.

ironic.conf.drac module

ironic.

ironic.conf.glance module

ironic.

ironic.

ironic.conf.healthcheck module

ironic.

ironic.conf.ibmc module

ironic.

ironic.conf.ilo module

ironic.

ironic.conf.inspector module

ironic.

ironic.

ironic.conf.ipmi module

ironic.

ironic.conf.irmc module

ironic.

ironic.conf.iscsi module

ironic.

ironic.conf.metrics module

ironic.

ironic.conf.metrics_statsd module

ironic.

ironic.conf.molds module

ironic.

ironic.conf.neutron module

ironic.

ironic.

ironic.conf.nova module

ironic.

ironic.

ironic.conf.opts module

ironic.

Return a list of oslo options available in Iron code

The returne list includ all oslo options Each elemen of the list is a tuple. The first element is the name of the group, the second element is the options.

The function is discoverable via the ironi entry poin under the oslo.config.opts namespace.

The function is used by Oslo sample config file generator to discover the options.

Returns
a list of (gro

op-
tions
tu-
ples

`ironic.`

ironic.conf.pxe module

`ironic.`

ironic.conf.redfish module

`ironic.`

ironic.conf.service_catalog module

`ironic.`

`ironic.`

ironic.conf.snmp module

`ironic.`

ironic.conf.swift module

`ironic.`

`ironic.`

ironic.conf.xclarity module

`ironic.`

Module contents

ironic.db package

Subpackages

ironic.db.sqlalchemy package

Submodules

ironic.db.sqlalchemy.api module

SQL storage back end.

class i

Base *iro db. api Con*

SqlA connection.

add_nod

Add tag to the node

If the node and tag pair

already exists, this shou still succeed.

Paramet

- **nod** The id of a node

- **tag** A tag strin

Returns the Nod Tag object.

Raises Nod Not-Foun if the node is not foun

add_noc

Add trait to the node

If the node

and trait pair already exists, this shou still succeed.

Parame

- **nod** The id of a node
- **tra** A trait strin
- **ver** the version of the object.

Returns the Nod Trai object.

Raises Invalid Parameter-

Valu if addi the trait wou exceed the per-node traits limit.

Raises

Nod Not-Four if the node is not foun

`check_r`

Chec a list of node iden tities and map it to UUI

This call take a list of node nam and/ UUI and tries to convert them to UUIDs. It fails early if any identities cannot possible be used as names or UUIDs.

Paramet

`ide`

List of identities.

Returns

A mapping from requested identities to node UUI

Raises

Nod Not-Four if som identities were not foun or can-

not be valid names or UUIDs.

check_v

Chec the who data for incompatible objects

This scan all

the tables in sear of objects that are not

supported; i.e., those that are not specified in *ironic.common.release_mappings.RELEASE_MAPPING*. This includes objects that have null version values.

Parame

`ign`

List of mod nam to skip

Returns

A Boo True if all the objects have supported versions

False otherwise.

clear_r

clear_r

create_

Create a new allocation.

with

Paramet

val

Dict of values to create an allocation

Returns

An allocation

Raises

AllocationD pli cate Nam

Raises

Allocation Al read ists

create_

Create a list of BIO Setting reco for a

give
node

Paramet

- **nod** The node id.
- **set** A list of BIO Settings to be created

(continues on next page)

(continued from previous page)

- **ver** the version of the object.

Returns
A list of BIO Setting object.

Raises
Nod Not-Foun if the node is not foun

Raises
BIO

SettingAlread ists if any of the setting reco

already exists.

create_

Create a new chas sis.

Parame

val

Dict of values.

create_

Create a deploy men template

Parame

val

A dict describ ing the deploy men template For example:

Raises
De-
ploy
plat-
eDu
pli-
cate-
Nam
if
a
de-
ploy
tem-
plat

with the same name exists.

Raises
De-
ploy
plat
read
ists
if
a
de-
ploy
tem-
plat
with
the

same UUID exists.

Returns

A deploy template

`create_`

Create a new node

Paramet

`val`

A dict containing several item used to identify

and track the node, and several dicts which are passed into the Drivers when managing this node. For example:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
val-
ues
con-
tains
tags

or traits.

Returns
A
node

create_
Cre-
ate
a
new
port

Parame
val
Dict
of
val-
ues.

create_
Cre-
ate
a
new
port-
grou

Parame
val
Dict
of
val-
ues
with
the

following keys
id
uuid
name node_id address extra created_at updated_at

Returns
A portgrou

Raises
Port
grou
plicate
Nam

Raises
Port
grou
MA
read
ists

Raises
Port
grou
read
ists

`create_`
Create a new volume connector.

Parame
`con`
Dictionary containing infor-

mation abou the connector. Example:

Returns
A volume connector.

Raises
VolumeConnectorTypeAndIdAlreadyExists If a

connector already exists with a matching type and connector_id.

Raises
VolumeConnectorAlreadyExists If a volume connec-

tor with the same UUID already exists.

create_
Create a new volume target.

Parame
tar
Dic-

tio-
nary
con-
tain-
ing
the
in-
for-
ma-
tion
abou

the volume target. Example:

(continues on next page)

(continued from previous page)

Returns
A volume target.

Raises
Volume getBoo dexAlread ists if a volume tar

get already exists with the same boot index and node ID.

Raises
Volume getAlread ists if a volume target with

the same UUID exists.

delete_

Dele a list of BIO settings

Paramet

- **nod** The node id.
- **nam** List of BIO setting nam to be dele

Raises

Nod Not-Four if the node is not foun

Raises

BIO Setting-Not-Four if any of BIO setting nam is

not found.

delete_

Dele spec i- fied tag from the node

Paramet

- **nod** The id of a node
- **tag** A tag strin

Raises

Nod Not- Four if the node is not foun

Raises

Nod Tag- Not- Four if the tag is not foun

delete_

Dele spec i- fied trait from the node

Parame

- **nod** The id of a node
- **tra** A trait strin

Raises

Nod Not- Foun if the node is not foun

Raises

Nod Trai Not- Foun if the trait is not foun

destroy

De- stroy an

allocation.

Paramet

all

Allocation ID or UUI

Raises

Allocation-Not-Foun

destroy

Destroy a chassis.

Paramet

cha

The id or the uuid of a chassis.

destroy

Destroy a deploy-men template

Paramet

tem

ID of the deploy men template to destro

Raises
Deploy plate Four if the deploy template does not exist.

destroy

Destroy a node and its associated resour

Destroy a node including any associ-

ated
port
port

groups, tags, traits, volume connectors, and volume targets.

Paramet

`nod`
The ID or UUI of a node

`destroy`
De- stroy an port

Paramet

`por`
The id or MA of a port

`destroy`
De- stroy a port- grou

Paramet

`por`
The UUI or MA of a port- grou

Raises
Port grou Notl

Raises

Port
grou
Not-
Four

destroy

De-
stroy
a
vol-
ume
con-
nec-
tor.

Parame

ide

The
UUI
or
in-
te-
ger
ID
of
a
vol-
ume
con-

nector.

Raises

Vol-
ume
Con
nec-
torN
Four
If
a
vol-
ume
con-
nec-
tor

with the specified ident does not exist.

destroy

De-
stroy
a
vol-
ume

tar-
get.

Parame
`ide`
The
UUI
or
in-
te-
ger
ID
of
a
vol-
ume
tar-

get.

Raises
Vol-
ume
get-
Not-
Fou
if
a
vol-
ume
tar-
get
with
the

specified ident does not exist.

`get_act`
Re-
triev
hard
ware
type
for
the
reg-
is-
tered
and
ac-
tive

conductors.

Parame
`use`

Whe to factor conductor_ into the keys

Returns

A dict whic map hardware type nam to the set of host whic

support them. For example:

`get_all`

Return an allocation representation.

Parame

all

The id of an allocation.

Returns

An allocation.

Raises

Allocation-Not-Fou

get_all

Return an allocation representation.

Parame

nam

The logical nam

of an allocation.

Returns
An allocation.

Raises
Allocation-Not-Foun

get_all
Return an allocation representation.

Paramet
all
The uuid of an allocation.

Returns
An allocation.

Raises
AllocationNotFou

`get_all`

Return a list of allocations

Parame

- `fil`
  Filters to apply. Default to Non

  node_
  uuid of node

  state
  allocation state

  resour
  reques resour class

-

lim Maximum number of allocations to return.

- **mar** The last item of the previous page we return the next result set.
- **sor** Attribu by whic results shou be sorte
- **sor** Direction in whic results shou

be sorte (asc desc

Returns
A list of allocations

get_bic
Retrieve BIO setting valu

Paramet

- **nod** The node id.
- **nam** String containing nam of BIO setting to be retrieved.

Returns
The BIO Setting object.

Raises
Nod Not- Foun if the node is not foun

Raises
BIO Set- ting- Not- Foun if the BIO set- ting is not foun

`get_bio`
Re- triev BIO set- ting of a give node

Parame
`nod`
The node id.

Returns
A list of BIO Set- ting ob- jects

Raises

Nod Not-Foun if the node is not foun

get_cha

Returns a chassis representation.

Paramet

cha

The id of a chassis.

Returns

A chassis.

get_cha

Returns a chassis representation.

Paramet

cha

The uuid of a

chas
sis.

Returns
A
chas
sis.

get_cha
Re-
turn
a
list
of
chas
sis.

Parame

- **lim**
Max
i-
mum
num
ber
of
chas
sis
to
re-
turn
- **mar**
the
last
item
of
the
pre-
vi-
ous
page
we
re-
turn

the next result set.

- **sor**
At-
tribu

by whic re- sults shou be sorte

- **sor** di- rec- tion in whic re- sults shou be sorte (asc desc

get_con Re- triev a con- duc- tors ser- vice reco from the data

Paramet

- **hos** The host nam of the con- duc- tor ser- vice
-

onl
Specify the filter value on the *online* field whe

querying conductors. The `online` field is ignored if this value is set to None.

Returns
A conductor.

Raises
ConductorNFound if the conductor with given hostnam

does not exist or doesnt meet the specified online expectation.

get_con
Return a list of conductors.

Paramet

- **lim**
Maxi-

mum number of conductors to return

- **mar** the last item of the previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** direction in whic results shou be sorte (asc desc

get_dep

Re-triev a de-ploy men tem-plate by ID.

Paramet

tem

ID of the de-ploy men tem-plate to re-triev

Raises

De-ploy plate Four if the de-ploy tem-plate does not ex-

ist.

Returns

A de-ploy tem-plate

get_dep

Re-triev a

de- ploy men tem- plate by nam

Paramet

`tem` nam of the de- ploy men tem- plate to re- triev

Raises

De- ploy plate Four if the de- ploy tem- plate does not ex- ist.

Returns

A de- ploy tem- plate

`get_dep`

Re- triev a de- ploy men templat by UUI

Paramet

tem

UUI of the deploy men template to retriev

Raises

Deploy plat Four if the deploy template does not exist.

Returns

A deploy template

get_dep

Retriev a list of deploy men template

Paramet

- `lim` Maximum number of deploy templates to re-

turn.

- `mar` The last item of the previous page we return

the next result set.

- `sor` Attribu by whic results shou be sorte
- `sor` Direction in

whic re- sults shou be sorte (asc desc

Returns
A list of de- ploy tem- plate

get_dep
Re- turn a list of de- ploy men tem- plate with one of

a list of names.

Paramet
nam
List of nam to fil- ter by.

Returns
A list of de- ploy tem- plate

get_noc

Return a node

Paramet

`nod` The id of a node

Returns A node

`get_noc`

Return a node

Paramet

`ins` The instanc uuid to searc for.

Returns A node

Raises InstanceNo Foun if the instanc is not foun

Raises Invalid UID if

the instanc uuid is invalic

get_noc
Return a node

Paramet
nod
The logical nam of a node

Returns
A node

get_noc
Find a node by any matching port address

Paramet
add
list of port address (e.g. MAC

Returns
Nod ob-

ject.

Raises

Nod Not-Four if none or several node are foun

get_noc

Return a node

Paramet

nod

The uuid of a node

Returns

A node

get_noc

Return a list of node

Paramet

- **fil** Filters to apply. Default to

Non

associa
True
|
Fals

reserv
True
|
Fals

maint
True
|
Fals

chassi
uuid
of
chas
sis

driver
driv
nam

provis
provisionstate
of
nod

provis
node
with
provisionfield
before
this
interval
in

seconds

- `lim`
Max
i-

mun num ber of node to re- turn

- **mar** the last item of the pre- vi- ous page we re- turn

the next result set.

- **sor** At- tribu by whic re- sults shou be sorte
- **sor** di- rec- tion in whic re- sults shou be sorte (asc desc
-

fie Com separatec field list to return to allow for only specific fields to be returned to have maximum API performance calls where not all columns are needed from the database.

get_noc

Get a node list with specific field

Parame

- **col** A list of colu to retriev from the data and populate into the object.
- **fil** The reques data field

filters in the form of a dictionary with the applicable key, and filter value.

- **`lim`** Lim the number of return node default Non
- **`mar`** Star ing mark to generate a pagi- nate result set for the consumer.
- **`sor`** Sort key to apply to the result set.
-

sor
Sort direction to apply to the result set.

Returns
A list of Nod objects base on the data mod from a

SQLAlchemy result set, which the object layer can use to convert the node into an Node object list.

get_noc
Get node tags base on its id.

Paramet
nod
The id of a node

Returns
A list of Nod Tag

ob-
jects

Raises

Nod
Not-
Fou
if
the
node
is
not
foun

get_noc

Get
node
trait
base
on
its
id.

Paramet

nod

The
id
of
a
node

Returns

A
list
of
Nod
Trai
ob-
jects

Raises

Nod
Not-
Fou
if
the
node
is
not
foun

get_noc

Get

specific colu for matc ing node

Return a list of the speci fied colu for all node that

match the specified filters.

Parame

- `col` List of columns nam to return. Default to id

column when columns == None.

- `fil` Filters to apply. Default to

Non

associ
True
|
Fals

reserv
True
|
Fals

reserv
[con
duc-
tor1
con-
duc-
tor2

maint
True
|
Fals

retired
True
|
Fals

chassi
uuid
of
chas
sis

driver
driv
nam

provis
pro-
vi-
sion
state
of
node

provis
node
with
pro-
vi-
sion
field

before this interval in

seconds

- **lim** Max i- mun num ber of node to return
- **mar** the last item of the previous page we return

the next result set.

- **sor** At- tribu by whic results shou be sorte
- **sor** di-

rec-
tion
in
whic
re-
sults
shou
be
sorte
(asc
desc

Returns
A
list
of
tu-
ples
of
the
spec
i-
fied
colu

get_not
Re-
turn
ob-
jects
with
ver-
sion
that
are
not
the
spec
i-
fied versions.

This
re-
turn
ob-
jects
with
ver-
sion
that
are
not
the

spec i-

fied versions. Objects with null versions (there shouldnt be any) are also returned.

Paramet

- **mod** the nam of the mod (clas of desired objects
- **ver** list of versions of objects not to be returne

Returns
list of the DB objects

Raises
IronicEx ception if there is no class

associated

with the name

get_off

Get a list conductors that are offline (dea

Parame

fie

A field to return hostnam by default

Returns

A list of requested field of offline conductors.

get_onl

Get a list conductor

host nam that are online and

active.

Returns
A list of conductor host nam

get_po
Return a network port representation.

Parame
add
The MA address of a port

Returns
A port

get_po
Return a network port rep-

representation.

Parame

por

The id of a port

Returns

A port

get_por

Return a network port representation.

Parame

por

The nam of a port

Returns

A port

get_por

Return a network port representation.

Paramet

por
The uuid of a port

Returns
A port

get_por

Return a list of ports

Paramet

- **lim** Maximum number of ports to return
- **mar** the last item of the previous page we return
-

the next result set.

sor At-tribu by whic re-sults shou be sorte

- **sor** di-rec-tion in whic re-sults shou be sorte (asc desc

get_por Re-turn a net-work port-grou rep-re-sen-ta-tion.

Paramet

- **add** The MAC ad-dres of a port-grou

- **pro**
  A node owner or lessee to filter by.

Returns
A port-grou

Raises
Port-grou Not-Foun

get_por
Return a network port-grou representation.

Paramet
por
The id of a port-grou

Returns
A port-grou

Raises
Port-grou

Not-Foun

get_por

Return a network portgrou representation.

Paramet

nam The logical nam of a portgrou

Returns A portgrou

Raises Portgrou NotFoun

get_por

Return a network portgrou representation.

Paramet

`por`
The uuid of a port-grou

Returns
A port-grou

Raises
Port-grou Not-Foun

`get_por`

Return a list of port-grou

Paramet

- `lim`
  Maximum number of port-grou to return
- `mar`
  The last item of the

previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** Direction in whic results shou be sorte (asc desc
- **pro** A node own or lesse to filter by.

Returns A list of portgrou

`get_por`

List all the port-grou for a give node

Paramet

- `nod` The in-te-ger node ID.
- `lim` Max i-mun num ber of port-grou to re-turn
- `mar` The last item of the pre-vi-ous page we re-turn

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** Direction in whic results shou be sorte (asc desc
- **pro** A node own or lesse to filter by.

Returns A list of port grou

get_por

List all the port

for a give node

Paramet

- **nod** The integer node ID.
- **lim** Maxi- mun num ber of port to return
- **mar** the last item of the previous page we return

the next result set.

- **sor** Attribu by whic results

shou be sorte

- **sor** direction in whic results shou be sorte (asc desc

Returns
A list of ports

get_por

List all the ports for a give portgrou

Parame

- **por** The integer portgrou ID.
- **lim** Max

i-
mun
num
ber
of
port
to
re-
turn

- **mar**
The
last
item
of
the
pre-
vi-
ous
page
we
re-
turn

the next result set.

- **sor**
At-
tribu
by
whic
re-
sults
shou
be
sorte

- **sor**
Di-
rec-
tion
in
whic
re-
sults
shou
be
sorte
(asc
desc

Returns
A list of port

get_vol
Return a volume connector representation.

Parame
db_
The integer data ID of a volume connec-

tor.

Returns
A volume connector with the specified ID.

Raises
Vol-

ume Con nector N Fou If a volume connector

with the specified ID is not found.

get_vol

Return a volume connector representation.

Parame

con

The UUI of a connector.

Returns

A volume connector with the specified UUI

Raises

Vol- ume Con nec- torN Four If a vol- ume con- nec- tor

with the specified UUID is not found.

get_vol

Re- turn a list of vol- ume con- nec- tors.

Parame

- **lim** Max i- mun num ber of vol- ume con- nec- tors to

return.

- **mar** The last item

of the previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** Direction in whic results shou be sorte (asc desc
- **pro** The associated node proje to searc with

Returns
a

list of `Vol` objects

Returns
A list of volume connectors.

Raises
InvalidParameterValu If sort_ does not exist.

get_vol

List all the volume connectors for a given node

Paramet

- **nod** The integer node ID.

- **lim** Maximum number of volume connectors to

return.

- **mar** The last item of the previous page we return

the next result set.

- **sor** Attribu by which results shou be sorte

- **sor** Direction in whic results shou be sorte (asc desc

- **pro** The associated node proj to sear with

Returns
a list of `Vol` objects

Returns
A list of volume connectors.

Raises
InvalidParameterValu If sort_ does

not exist.

get_vol

Return a volume target representation.

Paramet

db_

The data primary key (integer) ID of a volume target.

Returns

A volume target.

Raises

VolumegetNot-Fou if no volume target

with this

ID exists.

get_vol

Return a volume target representation.

Paramet

uui

The UUI of a volume target.

Returns

A volume target.

Raises

Volume get- Not- Foun if no volume target with this

UUID exists.

get_vol

Return a list of volume targets

Parame

- **lim** Max i- mun num ber of volume targets to re-

turn.

- **mar** the last item of the previous page we return

the next result set.

- **sor** At- tribu by whic re- sults

shou be sorte

- **`sor`** direction in whic results shou be sorte (asc desc
- **`pro`** The associated node proj to searc with

Returns
a list of `Vol` objects

Returns
A list of volume targets

Raises
Invali Param-

e-
ter-
Valu
if
sort_
does
not
ex-
ist.

get_vol

List
all
the
vol-
ume
tar-
gets
for
a
give
node

Parame

- **nod**
  The
  in-
  te-
  ger
  node
  ID.
- **lim**
  Max
  i-
  mum
  num-
  ber
  of
  vol-
  ume
  tar-
  gets
  to
  re-

turn.

-

`mar` the last item of the previous page we return

the next result set.

- `sor` Attribu by whic results shou be sorte
- `sor` direction in whic results shou be sorte (asc desc
- `pro` The associated node proj to sear

with

Returns
a list of `Vol` objects

Returns
A list of volume targets

Raises
InvalidParameterValu if sort_ does not exist.

get_vol

List all the volume targets for a given volume id.

Paramet

- **vol**
  The
  UUI
  of
  the
  vol-
  ume
- **lim**
  Max
  i-
  mun
  num
  ber
  of
  vol-
  ume
  tar-
  gets
  to
  re-

turn.

- **mar**
  the
  last
  item
  of
  the
  pre-
  vi-
  ous
  page
  we
  re-
  turn

the next result set.

- **sor**
  At-
  tribu
  by
  whic
  re-
  sults
  shou
  be
  sorte

- **sor** direction in whic results shou be sorte (asc desc

Returns
A list of volume targets

Raises
InvalicParameterValu if sort_ does not exist.

list_co
List all registerec hardware interface for a con-

ductor.

Paramet

con

Data ID of conductor.

Returns

List of `Con` objects

list_ha

List registered hardware interfaces for given hardware

types.

This is restric to only active conductors. :para hardware

list of hardware types to filter by. :returns: list of `ConductorHardwareInterfaces` objects.

migrate

Trie to mi-

grat
away
from
the
iscsi
de-
ploy
in-
ter-
face

Paramet

- **con** the admin context
- **max** The max imum number of objects to migrat

Must be >= 0. If zero, all the objects will be migrated.

Returns
A 2- tupl 1. the total number of objects that

need to be migrated (at the beginning of this call) and 2. the number of migrated objects.

node_ta

Chec if the spec- i- fied tag ex- ist on the node

Paramet

- **nod** The id of a node
- **tag** A tag strin

Returns

True if the tag ex- ists oth- er- wise Fals

Raises

Nod Not- Foun if the node is not foun

node_tr

Chec if the spec- i- fied trait ex- ists on the node

Paramet

- **nod** The id of a node
- **tra** A trait strin

Returns

True if the trait ex- ists oth- er- wise Fals

Raises

Nod Not- Foun if the node is not foun

registe

Register an active conductor with the cluster.

Paramet

- **val** A dict of values whic mus contain the following:

(continues on next page)

↪Conductor service.

(continued from previous page)

- **upd** Whe false registration will raise an exception when a conflicting online record is found. When true, will overwrite the existing record. Default: False.

Returns
A conductor.

Raises
ConductorA read istered

registe

Registers hard ware interface for a conductor.

Parame

- **con** Data ID of conductor to register for.
- **har** Nam of hard ware type for the interface
- **int** Typ of interface e.g. de-

ploy or boot

- **int** List of interface nam to register.
- **def** Strin the default interface for this hardware type

and interface type.

Raises
ConductorH wareinterfacesAlread istered if at least

one of the interfaces in the combination of all parameters is already registered.

release
Release the

rese vation on a node

Paramet

- `tag` A strin uniq iden tifying the rese vation hold
- `nod` A node id or uuid

Raises
Nod NotFoun if the node is not foun

Raises
Nod Lock if the node is reserv by

another host

Raises

Node Not-Lock if the node was foun to not have a rese

vation at all.

reserve

Reserv a node

To prevent othe Man agem vice from manipulating the

given Node while a Task is performed, mark it reserved by this host.

Parame

- **tag** A strin uniq iden tifying

the reser va- tion hold

- **nod** A node id or uuid

Returns
A Nod ob- ject.

Raises
Nod Not- Foun if the node is not foun

Raises
Nod Lock if the node is al- read re- serv

set_noc
Re- plac all of the node tags with spec i-

fied list of

tags.

This ignore duplicate tags in the specified list.

Parame

- **nod** The id of a node
- **tag** List of tags

Returns
A list of Nod Tag objects

Raises
Nod Not-Foun if the node is not

foun

set_noc

Re-plac all of the node trait with spec i-fied list of

traits.

This ig-nore du-pli-cate trait in the spec i-fied list.

Parame

- **nod** The id of a node
- **tra** List of trait
- **ver** the ver-sion

of the object.

Returns
A list of Nod Trai objects

Raises
InvalidParameterValu if setting the traits wou

exceed the per-node traits limit.

Raises
Nod Not-Four if the node is not foun

take_ov
Do a takeover for an allocation.

The

allocation is only updated if the old conductor matches the provided value, thus guarding against races.

Paramet

- **all** Allocation ID
- **old** The conductor ID we expect to be the current `conductor_affinity` of the allocation.
- **new** The conductor ID of the new `con`

Returns

True if the take over was successful, Fals other wise

Raises

AllocationNotFour

touch_c

Mark a conductor as active by updating its

updated_at property.

Parame

hos

The host nam of this conductor service

Raises

Con
duc-
torN
Foun

touch_r

Mar
the
node
pro-
vi-
sion
ing
as
run-
ning

Mar
the
node
pro-
vi-
sion
ing
as
run-
ning
by
up-
dat-
ing
its provision_updated_at property.

Parame

nod

The
id
of
a
node

Raises

Nod
Not-
Foun

unregis

Re-
mov
this
con-
duc-
tor
from

the service registry immediately.

Parame

`hos`

The host nam of this conductor service

Raises

ConductorNFour

`unregis`

Unregisters all hardware interface for a conductor.

Parame

`con`

Data ID of conductor to unregister for.

unset_r

Remov all tags of the node

Paramet

nod

The id of a node

Raises

Nod Not-Foun if the node is not foun

unset_r

Remov all trait of the node

Paramet

nod

The id of a node

Raises

Nod Not-Foun

if the nod is not foun

update_

Updates prop erties of an allocation.

Parame

- **all** Allocation ID
- **val** Dict of values to update
- **upd** If True and nod is updated updates the nod

with

instance_uuid and traits from the allocation

Returns
An allocation.

Raises
Allocation-Not-Foun

Raises
AllocationDuplicate-Nam

Raises
Instan As-sociated

Raises
Nod As-sociated

update_
Update a list of BIO Setting reco

Paramet

- **nod** The node id.
- **set** A list of BIO Set- tings to be up- date

(continues on next page)

(continued from previous page)

- **ver** the version of the object.

Returns
A list of BIO Setting objects

Raises
Nod NotFour if the node is not foun

Raises
BIO

Setting-Not-Foun if any of the settings is not

found.

update_

Update prop-erties of an chas-sis.

Paramet

- **cha** The id or the uuid of a chas-sis.
- **val** Dict of values to update

Returns

A chas-sis.

`update_`

Update a deploy men templat

Paramet

- **`tem`** ID of the deploy men templat to update
- **`val`** A dict describing the deploy men templat For

example:

(continues on next page)

(continued from previous page)

Raises
Deploy plateDuplicateNam if a deploy template with the same name exists.

Raises
Deploy plate Four if the deploy template does not exist.

Returns
A deploy template

update_
Up-

date properties of a node

Parame

- **nod** The id or uuid of a node
- **val** Dict of values to update May be a partial

list, eg. when setting the properties for a driver. For example:

(continues on next page)

(continued from previous page)

Returns
A node

Raises
Nod Associated

Raises
Nod NotFoun

`update_`
Update properties of an port

Paramet

- **`por`** The

id or MA of a port

- **val** Dict of values to update

Returns
A port

update_
Updates properties of a portgrou

Paramet

- **por** The UUI or MA of a portgrou
- **val** Dict of values to up-

date May contain the following keys: uuid name node_id address extra created_at updated_at

Returns
A portgrou

Raises
InvalidParameterValu

Raises
Port grou NotFoun

Raises
Port grou plicateNam

Raises
Port grou MAC read ists

`update_`
Update objects to their latest know version

This scan all the tables and for objects that are not in

their latest version, updates them to that version.

Parame

- **con** the admin context

- **max** The maximum number of objects to migrate

Must be >= 0. If zero, all the objects will be migrated.

Returns

A 2-tupl 1. the total num ber

of objects that need to be migrated (at the beginning of this call) and 2. the number of migrated objects.

update_

Updates properties of a volume connector.

Parame

- **ide** The UU or integer ID of a volume connector.
- **con** Dictionary containing the information abou

connector to update.

Returns
A volume connector.

Raises
Volume Con nectorT peAndIdAlread ists If another connector already exists with a matching type and connector_id field.

Raises
Volume Con nectorN Four If a volume connector with the specified ident does not exist.

Raises
Invalic Parameter Valu Whe a UUI

is in-clud

in connector_info.

update_

Up-date in-for-ma-tion for a vol-ume tar-get.

Parame

- **ide** The UUI or in-te-ger ID of a vol-ume tar-

get.

- **tar** Dic-tio-nary con-tain-ing the in-for-ma-tion abou

volume target to update.

Returns
A volume target.

Raises
InvalidParameterValu
if a UUI
is includ

in target_info.

Raises
VolumegetBoo
dexAlread
ists if a volume tar-

get already exists with the same boot index and node ID.

Raises
VolumegetNotFou
if no volume target with this

ident exists.

`ironic.`

`ironic.`

`ironic.`

Add an iden tity filter to a quer

Filters results by ID, if supplied valu is a vali integer. Otherwise attempts to filter results by UUID.

Paramet

- **que** Initial quer to add filter to.
- **val** Valu for filter-

ing re-sults by.

Returns

Mod i-fied quer

`ironic.`

`ironic.`

Add a port-spec fil-ter to a quer

Fil-ters re-sults by ad-dres if sup-plied valu is a valid

MAC address. Otherwise attempts to filter results by identity.

Paramet

- **que** Ini-tial quer to add fil-ter to.

- **val** Valu for filtering results by.

Returns Mod i- fied quer

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

Add a port spec filter to a quer

Filters results by address if supplied valu is a valid

MAC address. Otherwise attempts to filter results by identity.

Paramet

- **que** Initial quer to add filter to.
- **val** Valu for filtering results by.

Returns Mod i- fied quer

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

The back end is this mod ule itself.

`ironic.`

Que help for simpler session usage.

Paramet

`ses` if pres the session to use

ironic.db.sqlalchemy.migration module

`ironic.`

Create data sche from mod els descrip tion.

Can be used for initial installation instead of upgrade(head).

`ironic.`

Used

for dow grad ing data

Paramet

ver (*st* De- sired data ver- sion

ironic.

Cre- ates tem- plate for mi- gra- tion.

Paramet

- **mes** (*st* Text that will be used for mi- gra- tion ti- tle
- **aut** (*bo* If True -

  gen- er- ates

diff base on cur-

rent database state

`ironic.`

Stan data with pro- vide re- vi- sion

Don run any mi- gra- tion

Paramet

`rev`

(*st*

Shou matc one from repo i- tory or head -

to stamp database with most recent revision

`ironic.`

Usec for up- grad ing data

Paramet

`ver`

(*st*

De- sirec data ver-

sion

`ironic.`

Cur-
rent
data
ver-
sion

Returns

Data
ver-
sion

Return t

strin

ironic.db.sqlalchemy.models module

SQL
mod-
els
for
bare
data

class i

Base
`sql`
`ext`
`dec`
`api`
`Bas`

Rep-
re-
sent
an
al-
lo-
ca-
tion
of
a
node
for
de-
ploy-
ment.

candida

conduct

createc

extra

id

last_er

name

node_ic

owner

resourc

state

traits

updatec

uuid

versior

class i

Base
`sql`
`ext`
`dec`
`api`
`Bas`

Rep-
re-
sent
a
bios
set-
ting
of
a

bare
meta
node

createc

name

node_ic

updatec

value

versior

class i
Base
`sql`
`ext`
`dec`
`api`
`Bas`

Rep-
re-
sent
a
hard
ware
chas
sis.

createc

descrip

extra

id

updatec

uuid

versior

class i
Base
sql
ext
dec
api
Bas

Rep-
re-
senta
a
con-
duc-
tor
ser-
vice
en-
try.

conduct

createc

drivers

hostnan

id

online

updatec

versior

class i
Base
sql
ext
dec
api
Bas

Internal table used to track wha is load on each con-

ductor.

conduct

createc

default

hardwa

id

interfa

interfa

updatec

versior

class i

Base

sql

ext

dec

api

Bas

Representa deploy

men
tem-
plat

create

extra

id

name

update

uuid

versio

class i
Bas
sql
ext
dec
api
Bas

Rep-
re-
sent
a
de-
ploy
men
step
in
a
de-
ploy
men
tem-

plate.

args

create

deploy_

deploy_

id

interfa

priorit

step

updatec

versior

class i

Base
osl
sql
mod
Tim
osl
sql
mod
Mod

as_dict

metadat

versior

class i

Base
sql
ext
dec
api
Bas

Rep-
re-
sent
a
bare

meta
node

`allocat`

`automat`

`bios_ir`

`boot_ir`

`chassis`

`clean_s`

`conduct`

`conduct`

`console`

`console`

`createc`

`deploy_`

`deploy_`

`descrip`

`driver`

`driver_`

`driver_`

`extra`

`fault`

id

inspect

inspect

inspect

instanc

instanc

last_er

lessee

mainter

mainter

managem

name

network

network

owner

power_i

power_s

propert

protect

protect

provisi

provisi

raid_co

raid_in

rescue_

reserva

resourc

retired

retired

storage

target_

target_

target_

updated

uuid

vendor_

version

class i
Base
sql
ext
dec
api
Bas

Rep-
re-
sent
a
tag
of
a
bare
meta
node

created

node

node_ic

tag

updated

versior

class i
Base
`sql`
`ext`
`dec`
`api`
`Bas`

Rep-
re-
sent
a
trait
of
a
bare
meta
node

created

node

node_i

trait

update

versio

class i

Bas
sql
ext
dec
api
Bas

Represent a network port of a bare meta node

addres

create

extra

id

interna

is_sma

local_

name

node_i

physic

portgr

pxe_en

update

uuid

versio

class i

Bas
sql
ext
dec
api
Bas

Rep-
re-
sent
a
grou
of
net-
wor
port
of
a
bare
meta
node

addres

create

extra

id

interna

mode

name

node_ic

propert

standal

updatec

uuid

versior

class i

Base
`sql`
`ext`
`dec`
`api`
`Bas`

Rep-
re-
senta
a
vol-
ume
con-
nec-
tor
of
a
bare
meta
node

connect

createc

extra

id

node_ic

type

updatec

uuid

versior

class i

Base
sql
ext
dec
api
Bas

Represent a volume target of a bare meta node

boot_ir

createc

extra

id

node_ic

propert

update

uuid

versio

volume_

volume_

`ironic.`

Return the mod clas with the spec ified nam

Paramet

mod the nam of the clas

Returns

the clas with the spec ified nam

Raises

Exception if ther is

no class as- so- ci- ated with

the name

`ironic.`

Module contents

Submodules

ironic.db.api module

Base class for stor- age en- gine

class i

Base `obj`

Base class for stor- age sys- tem con- nec- tions

abstrac

Add tag to the node

If the

node and tag pair already exists, this shou still succeed.

Paramet

- **nod** The id of a node
- **tag** A tag strin

Returns
the Nod Tag object.

Raises
Nod Not-Foun if the node is not foun

abstrac
Add trait to the

node

If the node and trait pair already exists, this shou still succeed.

Parame

- **nod** The id of a node
- **tra** A trait strin
- **ver** the version of the object.

Returns the Nod Trai object.

Raises Invalid

Pa-
ram-
e-
ter-
Valu
if
addi
the
trait
wou
ex-

ceed the per-node traits limit.

Raises

Nod
Not-
Foun
if
the
node
is
not
foun

abstrac

Chec
a
list
of
node
iden
ti-
ties
and
map
it
to
UUI

This
call
take
a
list
of
node
nam
and/
UUI
and
tries
to
con-

vert them to UUIDs. It fails early if any identities cannot possible be used as names or UUIDs.

Paramet
`ide`
List of iden ties.

Returns
A map ping from requesiden ties to node UUI

Raises
Nod Not-Foun if som iden ties were not foun or can-

not be valid names or UUIDs.

abstrac
Chec the who data for incom pat ible ob jects

This scan all the tables in sear of objects that are not supported; i.e., those that are not specified in *ironic.common.release_mappings.RELEASE_MAPPING*.

Parame

ign

List of mod nam to skip

Returns

A Boo True if all the objects have supporte versions False otherwise.

abstrac

Create a new allocation.

Parame

val

Dict of values to create an allocation

with

Returns
An allocation

Raises
AllocationI plicate Nam

Raises
Allocation Already ists

abstrac
Create a list of BIO Setting reco for a give node

Paramet

- **`nod`** The node id.
- **`set`** A list of BIO Settings to be created

(continues on next page)

(continued from previous page)

```
```

- **ver** the version of the object.

Returns
A list of BIO Setting object.

Raises
Nod Not-Four if the node is not four

Raises
BIO

SettingAlread
exists if any of the setting reco

already exists.

abstrac

Create a new chas sis.

Parame

val

Dict of values.

abstrac

Create a deploymen template

Parame

val

A dict describ ing the deploymen template For example:

Raises

De-
ploy
plat-
eDu
pli-
cate-
Nam
if
a
de-
ploy
tem-
plate

with the same name exists.

Raises

De-
ploy
plate
read
ists
if
a
de-
ploy
tem-
plate
with
the

same UUID exists.

Returns
A deploy templat

abstrac
Create a new nod

Paramet
val
A dict containing several item use to iden tify

and track the node, and several dicts which are passed into the Drivers when managing this node. For example:

(continues on next page)

(continued from previous page)

(continues on next page)

(continued from previous page)

```
```

Raises
Invalid Parameter Valu if values contains tags

or traits.

Returns
A node

abstrac
Create a new port

Paramet
val
Dict of values.

abstrac
Create a new portgrou

Paramet
val
Dict of values with the

fol-
low-
ing
keys
id
uuid

name node_id address extra created_at updated_at

Returns
A
port-
grou

Raises
Port
grou
pli-
cate-
Nam

Raises
Port
grou
MA
read
ists

Raises
Port
grou
read
ists

abstrac
Cre-
ate
a
new
vol-
ume
con-
nec-
tor.

Paramet
con
Dic-
tio-
nary
con-
tain-
ing
in-
for-

mation abou the

connector. Example:

Returns
A volume connector.

Raises
VolumeCon nectorT pe-AndIdAlread ists If a

connector already exists with a matching type and connector_id.

Raises
VolumeCon nectorAlread ists If a volume connec-

tor with the same UUID already exists.

abstrac
Create a new volume target.

Parame
tar
Dic-

tionary containing the information abou

the volume target. Example:

(continues on next page)

(continued from previous page)

Returns
A volume target.

Raises
Vol-
ume
get-
Boo
dex-
Al-
read
ists
if
a
volume
tar-

get already exists with the same boot index and node ID.

Raises
Vol-
ume
ge-
tAl-
read
ists
if
a
volume
target
with

the same UUID exists.

abstrac

Dele a list of BIO set- tings

Parame

- **nod** The node id.
- **nam** List of BIO set- ting nam to be dele

Raises

Nod Not- Foun if the node is not foun

Raises

BIO Set- ting- Not- Foun if any of BIO setting nam is

not found.

abstrac

Dele spec i- fied tag from the node

Paramet

- **nod** The id of a node
- **tag** A tag strin

Raises

Nod Not- Foun if the node is not foun

Raises

Nod Tag- Not- Foun if the tag is not foun

abstrac

Dele spec i- fied trait from the node

Paramet

- **nod** The id of a node
- **tra** A trait strin

Raises

Nod Not- Foun if the node is not foun

Raises

Nod Trai Not- Foun if the trait is not foun

abstrac

De- stroy an

allocation.

Paramet

all

Allocation ID

Raises

Allocation-Not-Foun

abstrac

Destroy a chassis.

Paramet

cha

The id or the uuid of a chassis.

abstrac

Destroy a deploymen template

Paramet

tem

ID of

the deploy men template to destroy

Raises

Deploy plate Foun if the deploy template does not exist.

abstrac

Destroy a node and its associated resour

Destroy a node including any associated port

port

groups, tags, traits, volume connectors, and volume targets.

Paramet

nod

The ID or UUI of a node

abstrac

De-stroy an port

Paramet

por

The id or MAC of a port

abstrac

De-stroy a port-grou

Paramet

por

The UUI or MAC of a port-grou

Raises

Port grou Noth

Raises

Port grou

Not-
Foun

abstrac

De-
stroy
a
vol-
ume
con-
nec-
tor.

Paramet

ide

The
UUI
or
in-
te-
ger
ID
of
a
vol-
ume
con-

nector.

Raises

Vol-
ume
Con
nec-
torN
Foun
If
a
vol-
ume
con-
nec-
tor

with the specified ident does not exist.

abstrac

De-
stroy
a
vol-
ume
tar-
get.

Paramet

ide

The UUI or integer ID of a volume target.

Raises

Volume get-Not-Foun if a volume target with the specified ident does not exist.

abstrac

Retriev hardware type for the registered and active conductors.

Paramet

use

Whe to factor conductor_ into the keys

Returns

A dict whic map hardware type nam to the set of host whic

support them. For example:

abstrac

Return an allocation representation.

Paramet

`all` The id of an allocation.

Returns

An allocation.

Raises

AllocationNotFoun

`abstrac`

Return an allocation representation.

Parame

`nam` The logical nam of an allocation.

Returns
An allocation.

Raises
AllocationNot-Foun

abstrac
Return an allocation representation.

Paramet
all
The uuid of an allocation.

Returns
An allocation.

Raises
Allo-

cation-Not-Four

abstrac

Return a list of allocations

Paramet

- **fil** Filters to apply. Defaults to Non

node_u uuid of node

state allocation state

resour requested resource class

- **lim** Maximum number of allocations to re-

turn.

- **mar** The last item of the previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** Direction in whic results shou be sorte (asc

desc

Returns
A list of allocations

abstrac
Retriev BIO setting valu

Paramet

- **nod** The node id.
- **nam** String containing nam of BIO setting to be retrieved.

Returns
The BIO Setting object.

Raises
Nod Not-

Four if the node is not foun

Raises
BIO Setting-Not-Four if the BIO setting is not foun

abstrac
Retriev BIO settings of a give node

Paramet
nod The node id.

Returns
A list of BIO Setting objects

Raises
Nod Not-Four

if the node is not foun

abstrac

Return a chassis representation.

Paramet

cha

The id of a chassis.

Returns

A chassis.

abstrac

Return a chassis representation.

Paramet

cha

The uuid of a chassis.

Returns
A chassis.

abstrac

Return a list of chassis.

Paramet

- **lim** Maximum number of chassis to return.
- **mar** the last item of the previous page; we return

the next result set.

- **sor** Attribu by whic

results shou be sorte

- **sor** direction in whic results shou be sorte (asc desc

abstrac Retriev a conductors services reco from the data

Paramet

- **hos** The host nam of the conductor service
- **onl** Spec

ify the filter valu on the *online* field whe

querying conductors. The `online` field is ignored if this value is set to None.

Returns
A conductor.

Raises
ConductorN Fou if the conductor with give host nam

does not exist or doesnt meet the specified online expectation.

abstrac

Return a list of conductors.

Parame

- **lim** Maxi mum

number of conductors to return

- **mar** the last item of the previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte

- **sor** direction in whic results shou be sorte (asc desc

abstrac

Re-
triev
a
de-
ploy
men
tem-
plate
by
ID.

Parame

tem

ID
of
the
de-
ploy
men
tem-
plate
to
re-
triev

Raises

De-
ploy
plate
Foun
if
the
de-
ploy
tem-
plate
does
not
ex-
ist.

Returns

A
de-
ploy
tem-
plate

abstrac

Re-
triev
a

de-
ploy
men
tem-
plate
by
nam

Paramet
tem
nam
of
the
de-
ploy
men
tem-
plate
to
re-
triev

Raises
De-
ploy
plate
Four
if
the
de-
ploy
tem-
plate
does
not
ex-

ist.

Returns
A
de-
ploy
tem-
plate

abstrac
Re-
triev
a
de-
ploy
men
tem-

plate by UUI

Parame

tem UUI of the deploy men template to retriev

Raises Deploy plate Four if the deploy template does not exist.

Returns A deploy template

abstrac

Retriev a list of deploy men template

Paramet

- **lim** Maximum number of deploy templates to return.
- **mar** The last item of the previous page we return the next result set.
- **sor** Attribu by whic results shou be sorte
- **sor** Direction in

which results should be sorted (asc, desc

Returns
A list of deploy templates

abstrac
Return a list of deployment templates with one of

a list of names.

Paramet
nam
List of names to filter by.

Returns
A list of deploy templates

abstrac

Return a node

Paramet
`nod`
The id of a node

Returns
A node

`abstrac`
Return a node

Paramet
`ins`
The instanc uuid to searc for.

Returns
A node

Raises
InstanceNo Foun if the instanc is not foun

Raises
Invalid UID if

the instanc uuid is invalic

abstrac

Return a node

Paramet

nod

The logical nam of a node

Returns

A node

abstrac

Find a node by any matching port address

Paramet

add

list of port address (e.g. MAC

Returns

Nod ob-

ject.

Raises

Nod Not-Foun if none or several node are foun

`abstrac`

Return a node

Paramet

`nod` The uuid of a node

Returns

A node

`abstrac`

Return a list of node

Paramet

- **`fil`** Filters to apply. Default to

Non

associa
True
|
Fals

reserv
True
|
Fals

maint
True
|
Fals

chassis
uuid
of
chas
sis

driver
driv
nam

provis
pro-
vi-
sion
state
of
node

provis
node
with
pro-
vi-
sion
field
be-
fore
this
in-
ter-
val
in

seconds

- **`lim`**
  Max
  i-

mum num ber of node to return

- **`mar`** the last item of the previous page we return the next result set.
- **`sor`** Attribu by whic results shou be sorte
- **`sor`** direction in whic results shou be sorte (asc desc
-

`fie`

Com separated field list to return to allow for only specific fields to be returned to have maximum API performance calls where not all columns are needed from the database.

`abstrac`

Get node tags base on its id.

Paramet

`nod`

The id of a node

Returns

A list of Nod Tag objects

Raises

Nod NotFoun if the node is not foun

`abstrac`

Get node traits based on its id.

Parame
nod
The id of a node

Returns
A list of Nod Trai objects

Raises
Nod Not-Foun if the node is not foun

abstrac

Get specific colu for matching node

Return a list of the spec

ified colu for all nod that match the specified filters.

Paramet

- **col** List of column nam to return. Defaul to id column when columns == None.
- **fil** Filters to apply. Defaul to Non

associ True | Fals

reserv True | Fals

reserv [conductor1

con-
duc-
tor2

maint
True
|
Fals

retired
True
|
Fals

chassi
uuid
of
chas-
sis

driver
driv
nam

provis
pro-
vi-
sion
state
of
node

provis
node
with
pro-
vi-
sion
field
be-
fore
this
in-
ter-
val
in

seconds

- `lim`
Max-
i-
mun
num-
ber

of node to re- turn

- **mar** the last item of the pre- vi- ous page we re- turn

the next result set.

- **sor** At- tribu by whic re- sults shou be sorte
- **sor** di- rec- tion in whic re- sults shou be sorte (asc desc

Returns
A list of

tuples of the spec i- fied colu

abstrac Returns objects with versions that are not the spec i- fied versions.

Paramet

- **mod** the nam of the mod (clas of desired objects
- **ver** list of versions of objects not to

be returne

Returns
list of the DB objects

Raises
IronicException if there is no class associated

with the name

abstrac
Get a list conductors that are offline (dea

Paramet
fie
A field to return hostnam by default

Returns
A list of requesfield of offline conductors.

abstrac
Get a list conductor hostnam that are online and

active.

Returns
A list of conductor hostnam

abstrac
Return a network port representation.

Paramet

add

The MAC address of a port

Returns

A port

abstrac

Return a network port representation.

Paramet

por

The id of a port

Returns

A port

abstrac

Return a network port representation.

Paramet

por

The nam of a port

Returns
A port

abstrac

Return a networl port representation.

Paramet
por
The uuid of a port

Returns
A port

abstrac

Return a list of port

Paramet

- **lim** Maximum number of

port to re- turn

- **mar** the last item of the pre- vi- ous page we re- turn

the next result set.

- **sor** At- tribu by whic re- sults shou be sorte
- **sor** di- rec- tion in whic re- sults shou be sorte (asc desc

abstrac Re- turn a network portgroup representation.

Parame

- **add** The MA address of a portgrou
- **pro** A node own or lesse to filter by.

Returns
A portgrou

Raises
Portgrou Not-Four

abstrac
Return a network portgrou

representation.

Paramet

`por`

The id of a port-grou

Returns

A port-grou

Raises

Port grou Not-Foun

`abstrac`

Return a network port-grou representation.

Paramet

`nam`

The logical nam of a port-grou

Returns

A

portgrou

Raises

Portgrou NotFoun

abstrac

Return a networl portgrou representation.

Paramet

por

The uuid of a portgrou

Returns

A portgrou

Raises

Portgrou NotFoun

abstrac

Return a list of portgrou

Paramet

- **lim** Maximum num ber of port- grou to return
- **mar** The last item of the previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** Direction in whic results shou be

sorte (asc desc

- **pro** A node own or lesse to filter by.

Returns A list of port-grou

abstrac

List all the port-grou for a give node

Parame

- **nod** The integer node ID.
- **lim** Maximum num

ber of port-grou to return

- **mar** The last item of the previous page we return the next result set.
- **sor** Attribu by whic results shou be sorte
- **sor** Direction in whic results shou be sorte (asc desc
- **pro**

A node own or lesse to filter by.

Returns
A list of port grou

abstrac

List all the port for a give node

Paramet

- **nod** The integer node ID.
- **lim** Maximum number of port to return
-

mar the last item of the previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** direction in whic results shou be sorte (asc desc

Returns A list of port

abstrac

List all

the port for a given portgrou

Parame

- **por** The integer portgrou ID.
- **lim** Max imum number of port to return
- **mar** The last item of the previous page we return

the next result set.

- **sor** Attribu

by whic re- sults shou be sorte

- **sor** Di- rec- tion in whic re- sults shou be sorte (asc desc

Returns A list of port

abstrac Re- turn a vol- ume con- nec- tor rep- re- sen- ta- tion.

Parame **db_** The in- te- ger data ID of a volume connec-

tor.

Returns
A volume connector with the spec i- fied ID.

Raises
Vol- ume Con nec- torN Fou If a volume connector

with the specified ID is not found.

abstrac
Return a volume connector representation.

Parame

con
The UUI of a connector.

Returns
A volume connector with the spec ified UUI

Raises
Volume Con nectorN Foun If a volume connector

with the specified UUID is not found.

abstrac

Return a list of volume connectors.

Paramet

- **lim** Maximum number of volume connectors to

return.

- **mar** The last item of the previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** Direction in

whic results shou be sorte (asc desc

- **pro** The associated node proj to searc with

Returns
a list of `Vol` objects

Returns
A list of volume connectors.

Raises
InvalidParameterValu If sort_ does not ex-

ist.

abstrac

List all the volume connectors for a give node

Parame

- **nod** The integer node ID.
- **lim** Maximum number of volume connectors to return.
- **mar** The last item of

the previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** Direction in whic results shou be sorte (asc desc
- **pro** The associated node proj to searc with

Returns a list

of `Vol` objects

Returns
A list of volume connectors.

Raises
InvalidParameterValu If sort_ does not exist.

abstrac
Return a volume target representation.

Paramet
db_
The data primary key (inte-

ger) ID of a volume target.

Returns
A volume target.

Raises
VolumegetNotFour if no volume target with this ID exists.

abstrac
Return a volume target representation.

Paramet
uui
The UUI of a volume tar-

get.

Returns
A volume target.

Raises
Volume get- Not- Four if no volume target with this

UUID exists.

abstrac

Return a list of volume targets

Parame

- **lim** Max i- mum number of volume targets to

re-

turn.

- **mar** the last item of the previous page we return

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** direction in whic results shou be sorte (asc desc
- **pro** The associated

node proj to sear with

Returns
a list of `Vol` objects

Returns
A list of volume targets

Raises
Invali Parameter-Valu if sort_ does not exist.

abstrac

List all the volume targets for a given node

Paramet

- **nod** The integer node ID.
- **lim** Maximum number of volume targets to re-

turn.

- **mar** the last item of the previous page we return

the next result set.

- **sor** Attribute by which results should be sorte

- **`sor`** direction in whic results shou be sorte (asc desc
- **`pro`** The associated node proj to searc with

Returns
a list of `Vol` objects

Returns
A list of volume targets

Raises
Inval Parameter-

Valu if sort_ does not exist.

abstrac

List all the volume targets for a give volume id.

Paramet

- **vol** The UUI of the volume
- **lim** Maximum number of volume targets to re-

turn.

- **mar** the last item of the previous page we return.

the next result set.

- **sor** Attribu by whic results shou be sorte
- **sor** direction in whic results shou be sorte (asc desc

Returns A list of volume targets

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
sort_
does
not
ex-
ist.

abstrac
List
all
reg-
is-
tered
hard
ware
in-
ter-
face
for
a
con-

ductor.

Paramet
con
Data
ID
of
con-
duc-
tor.

Returns
List
of
`Con`
ob-
jects

abstrac
List
reg-
is-
tered
hard

ware in- ter- face for give hard ware

types.

This is re- stric to only ac- tive con- duc- tors. :para hard ware

list of hardware types to filter by. :returns: list of `ConductorHardwareInterfaces` objects.

abstrac

Trie to mi- grat awa fron the iscsi de- ploy in- ter- face

Paramet

- **con** the ad- min con- text
- **max**

The max- i- mum num- ber of ob- jects to mi- grat

Must be >= 0. If zero, all the objects will be migrated.

Returns

A 2- tupl 1. the to- tal num- ber of ob- jects that

need to be migrated (at the beginning of this call) and 2. the number of migrated objects.

abstrac

Chec if the spec- i- fied tag ex- ist on the node

Paramet

- **nod** The id of a

node

- **tag**
  A tag string

Returns
True if the tag exists otherwise Fals

Raises
Nod Not-Foun if the node is not foun

abstrac
Che if the spec i- fied trait exists on the node

Paramet

- **nod**
  The id of a

node

- `tra`
  A trait strin

Returns
True if the trait exists otherwise Fals

Raises
Nod Not-Foun if the node is not foun

abstrac
Register an active conductor with the cluster.

Paramet

- `val`
  A dict of

values whic mus contain the following:

```
↪Conductor service.
```

(continues on next page)

(continued from previous page)

- **upd** Whe false registration will raise an exception when a conflicting online record is found. When true, will overwrite the existing record. Default: False.

Returns
A conductor.

Raises
Con ductorA read istered

abstrac

Registers hard ware interface for a conductor.

Paramet

- **con** Data ID of conductor to register for.
- **har** Nam of hard ware type for the interface
- **int** Typ of interface e.g. deploy or boot
- **int** List of interface nam to reg-

is-
ter.

-

  def
  Strin
  the
  de-
  fault
  in-
  ter-
  face
  for
  this
  hard
  ware
  type

and interface type.

Raises
Con
duc-
torH
ware
ter-
face
sAl-
read
is-
tered
if
at
least

one of the interfaces in the combination of all parameters is already registered.

abstrac
Re-
lease
the
reser
va-
tion
on
a
node

Paramet

-

  tag
  A
  strin
  uniq

iden ti- fy- ing the reser va- tion hold

- **nod** A node id or uuid

Raises
Nod Not- Foun if the node is not foun

Raises
Nod Lock if the node is re- serv by an- othe host

Raises
Nod Not- Lock if the node was foun to

not have a reser

vation at all.

abstrac

Reserv a node

To prevent othe Manager vice from manipulating the

given Node while a Task is performed, mark it reserved by this host.

Parame

- **tag** A strin uniq identifying the reservation hold
- **nod** A node id or uuid

Returns
A Nod ob- ject.

Raises
Nod Not- Foun if the node is not foun

Raises
Nod Lock if the node is al- read re- serv

abstrac
Re- plac all of the node tags with spec i- fied list of tags.

This ig- nore du- pli- cate tags in

the spec i- fied list.

Parame

- **nod** The id of a node

- **tag** List of tags

Returns
A list of Nod Tag ob- jects

Raises
Nod Not- Foun if the node is not foun

abstrac
Re- plac all of the node trait with spec i-

fied list of traits.

This ignore duplicate trait in the specified list.

Parame

- **`nod`** The id of a nod
- **`tra`** List of trait
- **`ver`** the version of the object.

Returns A list of Nod Trai objects

Raises
Invali Parameter-Valu if setting the trait wou

exceed the per-node traits limit.

Raises
Nod NotFoun if the node is not foun

abstrac

Do a take over for an allocation.

The allocation is only update if the old conductor matches the provided value, thus guarding against races.

Paramet

- **all** Allocation ID
- **old** The conductor ID we expect to be the current `conductor_affinity` of the allocation.
- **new** The conductor ID of the new `con`

Returns True if the take over was successful, Fals oth-

er-
wise

Raises
Al-
lo-
ca-
tion-
Not-
Foun

abstrac
Mar
a
con-
duc-
tor
as
ac-
tive
by
up-
dat-
ing
its

updated_at property.

Paramet
hos
The
host
nam
of
this
con-
duc-
tor
ser-
vice

Raises
Con
duc-
torN
Foun

abstrac
Mar
the
node
pro-
vi-
sion
ing

as running

Mar the nod provisioning as running by updating its provision_updated_at property.

Paramet

`nod`
The id of a nod

Raises
Nod NotFoun

`abstrac`
Remov this conductor fro the service registry immediately.

Paramet

`hos`
The host nam

of this conductor service

Raises
Con duc torN Foun

abstrac
Unregisters all hard ware interfaces for a conductor.

Paramet
con
Data ID of conductor to unregister for.

abstrac
Remov all tags of the node

Parame

nod
The id of a node

Raises
Nod Not-Foun if the node is not foun

abstrac
Remov all trait of the node

Parame

nod
The id of a node

Raises
Nod Not-Foun if the node is not foun

abstrac
Update prop erties of

an allocation.

Parame

- **all** Allocation ID
- **val** Dict of values to update
- **upd** If True and node is updated updates the node with

instance_uuid and traits from the allocation

Returns
An allocation.

Raises
Alloca-

tion-
Not-
Foun

Raises
Allo-
ca-
tion
pli-
cate-
Nam

Raises
In-
stanc
As-
so-
ci-
ated

Raises
Nod
As-
so-
ci-
ated

abstrac
Up-
date
a
list
of
BIO
Set-
ting
reco

Parame

- **nod**
The
node
id.
- **set**
A
list
of
BIO
Set-

tings
to
be
up-
date

(continues on next page)

(continued from previous page)

```
```

- **ver** the version of the object.

Returns
A list of BIO Setting objects

Raises
Nod NotFoun if the node is not foun

Raises
BIO SettingNotFoun if any of the settings is

not found.

abstrac

Update properties of an chassis.

Paramet

- **cha** The id or the uuid of a chassis.
- **val** Dict of values to update.

Returns

A chassis.

abstrac

Update a deploymen template

Paramet

- **tem** ID of the deploy men template to update
- **val** A dict describing the deploy men template For

example:

Raises

De-
ploy
plat-
eDu
pli-
cate
Nam
if
a
de-
ploy
tem-
plate

with the same name exists.

Raises

De-
ploy
plate
Foun
if
the
de-
ploy
tem-
plate
does
not
ex-

ist.

Returns

A
de-
ploy
tem-
plate

abstrac

Up-
date
prop
er-
ties
of
a
node

Paramet

- **nod**
  The

id or uuid of a node

- **`val`** Dict of values to update. May be a partial

list, eg. when setting the properties for a driver. For example:

(continues on next page)

(continued from previous page)

```
```

Returns
A node

Raises
Nod Associated

Raises
Nod Not-Foun

`abstrac`
Update properties of an port

Parame

- `por` The id or MAC of a port
- `val` Dict of val-

ues to update

Returns
A port

abstrac
Update properties of a portgrou

Paramet

- **por** The UUI or MA of a portgrou
- **val** Dict of values to update May contain the fol-

lowing keys: uuid name node_id address extra created_at updated_at

Returns
A portgrou

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu

Raises
Port
grou
Not-
Fou

Raises
Port
grou
pli-
cate
Nam

Raises
Port
grou
MA
read
ists

abstra
Up-
date
ob-
jects
to
thei
lat-
est
kno
ver-
sion

This
scan
all
the
ta-
bles
and
for
ob-
jects
that
are

not in

their latest version, updates them to that version.

Paramet

- **con** the admin context
- **max** The maximum number of objects to migrate

Must be >= 0. If zero, all the objects will be migrated.

Returns

A 2-tupl 1. the total number of objects that

need to be migrated (at the beginning of this call) and 2. the number of migrated objects.

abstrac

Updates prop erties of

a volume connector.

Parame

- **ide** The UUI or integer ID of a volume con

nector.

- **con** Dictionary containing the information abou

connector to update.

Returns
A volume connector.

Raises
Volume Con

nec-
torT
pe-
An-
dI-
dAl-
read
ists
If
an-

other connector already exists with a matching type and connector_id field.

Raises

Vol-
ume
Con
nec-
torN
Four
If
a
vol-
ume
con-
nec-
tor

with the specified ident does not exist.

Raises

In-
valic
Pa-
ram-
e-
ter-
Valu
Whe
a
UUI
is
in-
clud

in connector_info.

abstrac

Up-
date
in-
for-
ma-
tion
for
a

volume target.

Parame

- **`ide`** The UUI or integer ID of a volume tar-

get.

- **`tar`** Dictionary containing the information abou

volume target to update.

Returns
A volume target.

Raises
Invalid Parameter-

Valu
if
a
UUI
is
in-
clud

in target_info.

Raises

Vol-
ume
get-
Boo
dex-
Al-
read
ists
if
a
vol-
ume
tar-

get already exists with the same boot index and node ID.

Raises

Vol-
ume
get-
Not-
Fou
if
no
vol-
ume
tar-
get
with
this

ident exists.

`ironic.`

Re-
turn
a
DB
API
in-
stan

ironic.db.migration module

Data setu and mi-gra-tion com man

`ironic.`

`ironic.`

`ironic.`

`ironic.`

`ironic.`

Mi-grat the data to *ver-sion* or the mos re-cent ver-sion.

`ironic.`

Module contents

ironic.dhcp package

Submodules

ironic.dhcp.base module

Abstrac base class for dhcp prov

class i

Base obj

Base class for DHC prov APIs

clean_c

Clea up the DHC BOO options for all ports in *task*

Parame

tas

A Task agen instanc

Raises

Faile To-Cle-anD HCP

get_ip_

Get

IP ad- dres for all port in *task*

Parame

tas

A Task agen in- stan

Returns

List of IP ad- dres as- so- ci- ated with task port and

portgroups.

abstrac

Send or up- date the DHC BOO op- tions for this node

Parame

- **tas** A Task

ager instanc

- **`opt`** this will be a list of dicts e.g.

(continues on next page)

(continued from previous page)

- **vif**
A
dict
with
keys
port
and
port-
grou
and
dict
as
val-

ues. Each dict has key/value pairs of the form <ironic UUID>:<neutron port UUID>. e.g.

If
the

valu is Non will get the list of port from the Iron

port/portgroup objects.

Raises
Faile ToU dat- eD- HCI tOn- Port

abstrac

Up- date one or more DHC op- tion on the spec i- fied

port.

Parame

- **por** des- ig- nate whic port thes at- tribu will be applie

to.

- **dhc** this will be a list of dict e.g.

(continues on next page)

(continued from previous page)

- **tok** An optiona authen tication token. Dep re-

cated, use context

- **con** (`ir com con Req` request context

Raises
Faile ToU dat- eD- HCF tOn- Port

ironic.dhcp.neutron module

class i

Base *irc dhc bas Bas*

API for communicating to neutron 2.x API

get_ip_

Get IP addres for all port in *task*

Paramet

tas a Task agen instan

Returns

List of IP addres associated with

task
port

update_

Send or up- date the DHC BOO op- tions for this node

Paramet

- **tas** A Task ager in- stan
- **opt** this will be a list of dicts e.g.

(continues on next page)

(continued from previous page)

```
```

- **vif** a dict of Neutron port dicts to update DHC options on. The port/portgroup dict key should be Ironic port UUIDs, and the values should be Neutron port UUIDs, e.g.

update_

Up-
date
a
port
at-
tribu

Updated one or more DHC options on the specified port

For the relevant API spec, see https://docs.openstack.org/api-ref/network/v2/index.html#update-port

Parame

- **por** designated which port these attributes will be applied

to.

- **dhc** this will be a list of dicts e.g.

(continues on next page)

(continued from previous page)

- **`tok`**
  op-

tiona
auth
token.
Dep
recate
use
context.

- **con**
  (iro
  com
  con
  Req
  request
  context

Raises
Faile
ToUdat
eD
HCP
tOn
Port

ironic.dhcp.none module

class i
Base
iro
dhc
bas
Bas

No-op
DHC
API

get_ip_
Get
IP
address
for

all
port
in
task

Paramet
`tas`
A
Task
ager
in-
stan

Returns
List
of
IP
addresses
associated
with
task
port
and

portgroups.

`update_`
Send
or
update
the
DHC
BOO
options
for
this
node

Paramet

- **`tas`**
A
Task
ager
in-
stan

- `opt` this will be a list of dicts e.g.

(continues on next page)

(continued from previous page)

- `vif` A dict with keys port and port- grou and dicts as values. Each dict has key/value pairs of the form <ironic UUID>:<neutron port UUID>. e.g.

If the

valu is Non will get the list of port from the Iron

port/portgroup objects.

Raises

Faile ToU dat- eD- HCP tOn- Port

update_

Up- date one or more DHC op- tions on the spec i- fied

port.

Parame

- **por** des- ig- nate which port these at- tribu will be applie

to.

- **dhc** this will be a list of dicts e.g.

(continues on next page)

(continued from previous page)

```
```

- **tok** An optiona authentication token. Deprecated, use context
- **con** (irc com con Req) request context

Raises
Faile ToU dateD HCE tOnPort

Module contents

ironic.drivers package

Subpackages

ironic.drivers.modules package

Subpackages

ironic.drivers.modules.ansible package

Submodules

ironic.drivers.modules.ansible.deploy module

Ansible deploy interface

class i

Base
iro
dri
mod
age
Hea
iro
dri
mod
age
Age
iro
dri

base.DeployInterface

Interface for depl
relat
ac-

tions

clean_u

Clea up the deploy- men en- vi- ron- men for this node

collect

deploy

Per- form a de- ploy- men to a node

execute

Ex- e- cute a clea step

Paramet

- **tas** a Task agen ob- ject con- tain- ing the

node

- **ste** a clea step dictionary to execute

Returns
Non

get_cle
Get the list of clea steps from the file.

Parame
tas a Task agen object containing the node

Returns
A list of clea step dictionaries

get_pro

Return the prop erties of the interface

has_dec

Whe the driv supports decom pose deploy step

Previousl (sinc Rock driv used a single deploy deploy step on the deploy interface. Some additional steps were added for the direct and iscsi deploy interfaces in the Ussuri cycle, which means that more of the deployment flow is driven by deploy steps.

prepare

Prepare the deploymen environmen

for this node

`prepare`

Boo into the ram to prepare for clean ing.

Parame

`tas` a Task agen object containing the node

Raises

`Noc` if the previous clean ing port cannot be re-

moved or if new cleaning ports cannot be created

Returns

Non or state for asyn prepare

`process`

Star the next clea step if the previous one is complete.

Paramet

- `tas` a Task ager instanc
- `ste` clea or deploy

`take_ov`

Take over man agemen of this task node from a dead conductor.

If conductors

host
mair
tain
a
stati
relation-
ship
to

nodes, this method should be implemented by the driver to allow conductors to perform the necessary work during the remapping of nodes to conductors when a conductor joins or leaves the cluster.

For exar

Neu
tron
mus
forward
DHC
BOC
reques
to
a
conduc-

tor which has prepared the tftpboot environment for the given node. When a conductor goes offline, another conductor must change this setting in Neutron as part of remapping that nodes control to itself. This is performed within the *takeover* method.

Paramet

`tas`

A
Task
ager
instand
contain-
ing
the
node
to
act

on.

`tear_do`

Tear
dow
a
pre-

vi-
ous
de-
ploy
men
on
the
task
node

tear_do

A
de-
ploy
step
to
tear
dow
the
agen

Shut
dow
the
ma-
chin
and
re-
mov
it
from
the
pro-
vi-
sion

ing network.

Parame

tas
a
Task
agen
ob-
ject
con-
tain-
ing
the
node

tear_do

Clea
up

the PXE and DHC files af- ter clea ing.

Parame

tas a Task agen ob- ject con- tain- ing the node

Raises

Noc if the clea ing port can- not be re- mov

validat

Val- i- date the drive spec Nod de- ploy men info

write_i

excepti

Base
`iro`
`exc`
`Iro`

Module contents

ironic.drivers.modules.drac package

Submodules

ironic.drivers.modules.drac.bios module

DRA
BIO
configuration
specific
methods

class i

Base
iro
dri
moc
rec
bic
Rec

iDR
Redfish
interface
for
BIO
setti
relat
actions

Pres
this

class entirel de- fers to its base class a gene venc

independent Redfish interface. Future resolution of Dell EMC- specific incompatibilities and introduction of vendor value added should be implemented by this class.

class i

Base *irc dri bas BIC*

BIO In- ter- face Im- ple- men ta- tion for iDR

apply_c

Ap- ply the BIO con- fig- u- ra- tion to the node

Paramet

-

tas a Task agen instan containing the node to act

on

- **set** List of BIO settings to apply

Raises
DRA C-Operation upon an error from pyth drac

Returns
state (clea ing) or state (deploymen if config-

u-
ra-

tion is in progress asynchronously or None if it is completed.

cache_b

Stor or update the current BIO settings for the node

Get the current BIO settings and store them in the bios data

table.

Parame

tas a Task agen instance containing the node to act

on.

Raises

DracOperationl on an error from pyth drac

factory

Reset the BIO settings of the node to the factory default.

This uses the Lifecycle Controll configuration to perform BIOS configuration reset. Leveraging the python-dracclient methods already available.

Paramet

tas a Task ager instan containing the node to act

on

Raises
DracOperationI on an error from pyth drac

Returns
state (clea ing) or state (deployment if reset is in

progress asynchronously or None if it is completed.

get_pro
Return the properties of the BIO Interface

Returns
dictionary of <pro erty nam <pro erty descrip tion entries

validat
Validate the driv spec information used by the idrac BMC

Paramet
tas a Task agen instance containing the node to act on

Raises
InvalidParam

e-
ter-
Valu
if
som
man
tory
in-
for-

mation is missing on the node or on invalid inputs

`ironic.`

Aba
dons
un-
com
mit-
ted
chan
adde
by
set_

Paramet

`tas`

a
Task
ager
in-
stanc
con-
tain-
ing
the
node
to
act

on.

Raises

Dra-
c-
Op-
er-
a-
tionE
on
an
er-
ror
from
pyth
drac

`ironic.`

Com
mits
pend
ing
char
adde
by
set_

Paramet

- **`tas`**
  a
  Task
  ager
  instanc
  containing
  the
  node
  to
  act
  on.

- **`reb`**
  indicate
  whe
  a
  reboot
  job
  shou
  be
  automatically created with the config job.

Raises

Drac-
Oper-
ationl
on

an error from pyth drac

Returns
the job_ key with the id of the new created config

job.

`ironic.`

Get the BIO configuration.

The BIO set ting look like:

(continues on next page)

(continued from previous page)

```
↪   'pending_value': 'New Value', # could also be None

↪   'read_only': False,

↪   'possible_values': ['Value', 'New Value', 'None']},
```

(continues on next page)

(continued from previous page)

```
↪    'current_value': 'Information',
↪    'pending_value': None,
↪    'read_only': False,
```

(continues on next page)

```
↪    'min_length': 0,
```

(continued from previous page)

```
↪    'max_length': 255,
↪    'pcre_regex': '^[0-9A-Za-z]{0,255}$'},
↪     'current_value': 0,
```

(continues on next page)

(continued from previous page)

```
↪       'pending_value': None,
↪       'read_only': True,
↪       'lower_bound': 0,
```

(continues on next page)

```
↪       'upper_bound': 65535}}
```

(continued from previous page)

Paramet

`nod` an iron node object.

Raises

DracOperationl on an error from pyth drac

Returns

a dictionary containing BIO settings

The abov values are only examples of coun BIO attribu exposed via this API will always be either an enumerated attribute, a string attribute, or an integer attribute. All attributes have the following parameters:

Paramet

- **`nam`** is the nam of the BIO attribu
- **`cur`** is the current valu of the attribu It will always be either an integer or a string.
- **`pen`** is the new valu that we wan the attribu to have None means that there is no pending value.
- **`rea`** indicate

whe this attribu can be chan Trying to change a read-only value will result in an error. The read-only flag can change depending on other attributes. A future version of this call may expose the dependencies that indicate when that may happen.

Enumerable attribu also have the following parameters:

Paramet

pos is an array of values it is permissible to set the attribute to.

Strin attribu also have the following parameters:

Paramet

- **min** is the minimum leng of the strin
- **max** is the maximum leng of the strin
- **pcr** is a PCR compatible regular expres

sion that the string must match. It may be None if the string is read only or if the string does not have to match any particular regular expression.

Integer attribu also have the following parame-

ters:

Paramet

- **low** is the minimun valu the attribu can have
- **upp** is the maximun valu the attribu can have

ironic.

Sets the pend ing_ parameter for

each of the val-

ues passed in.

Paramet

- **tas** a Task ager in- stanc con- tain- ing the node to act

on.

- **kwa** a dic- tio- nary of {At- tribu Nam New Valu

Raises
Dra- c- Op- er- a- tionl on an er- ror from pyth drac

Returns

A dictionary containing the is_c key with a bool

value indicating whether commit_config() needs to be called to make the changes, and the is_reboot_required key which has a value of true or false. This key is used to indicate to the commit_config() call if a reboot should be performed.

ironic.drivers.modules.drac.boot module

class i

Base *iro dri moc rec boo Rec*

iDR Redfish interface for virtual media boot relat actions.

Virtual Media allows boot

ing the system fro virtual CD/DVD drive containing user image that BMC inserts into the drive.

The CD/ images mus be in ISO format and (depen ing on BMC implementation) could be pulled over HTTP, served as iSCSI targets or NFS volumes.

The base line boot wor flow is mos base on the standard Redfish virtual media boot interface, which looks like this:

1. Pull kernel, ram and ESP if UEF boot is re-

ques
(FA

partition image with EFI boot loader) images

2. Create boot ISO out of images (#1) push it to Gla

and pass to the BMC as Swift temporary URL

3. Optionally create flop image with desired system

configuration data, push it to Glance and pass to the BMC as Swift temporary URL

4. Insert CD/ and (optionally) flop images and set prop

boot mode

For

buil ing deploy or rescue ISO redfish boot interface uses *deploy_kernel*/*deploy_ramdisk* or *rescue_kernel*/*rescue_ramdisk* properties from *[instance_info]* or *[driver_info]*.

For buil ing boot (use ISO redfish boot interface seek *kernel_id* and *ramdisk_id* properties in the Glance image metadata found in *[instance_info]image_source* node property.

iDR virtual media boot interface only differs by the way how it sets the node to boot from a virtual media device - this is done via OEM action call implemented in Dell sushy OEM extension package.

ironic.drivers.modules.drac.common module

Com mon func tion al i ties shar be twee dif fer ent DRA

modules.

ironic.

Re turns a DRA Clie ob ject from pyth drac li brar

Paramet

nod
an ironi node ob ject.

Returns
a DRA Clie ob ject.

Raises
In valid Pa ram e-

ter-
Valu
if
man
tory
in-
for-
ma-
tion is missing on the node or on invalid input.

`ironic.`

Pars
a
node
drive
val-
ues.

Pars
the
drive
of
the
node
read
de-
fault
val-
ues
and
re-
turns
a dict containing the combination of both.

Paramet
nod
an
ironi
node
ob-
ject.

Returns
a
dict
con-
tain-
ing
in-
for-
ma-
tion
from

driv and default values.

Raises
Invali ParameterValu if som man tory information is missing on the node or on invalid inputs.

ironic.drivers.modules.drac.inspect module

DR inspec tion interface

class i

Base *iro dri moc dra ins Dra*

Clas alias of class Drac Man Inspec

This class

pro-
vide
on-
go-
ing
sup-
port
of
the
dep-
re-
cate

idrac inspect interface implementation entrypoint.

All
bug
fixes
and
new
fea-
tures
shou
be
im-
ple-
men
in
its

base class, DracWSManInspect. That makes them available to both the deprecated idrac and new idrac-wsman entrypoints. Such changes should not be made to this class.

class i

Base
irc
dri
moc
rec
ins
Rec

iDR
Red-
fish
in-
ter-
face
for
insp
relat
ac-
tions

inspect

In- spec hard ware to get the hard ware prop er- ties.

In- spec hard ware to get the es- sen- tial prop er- ties. It

fails if any of the essential properties are not received from the node.

Parame

tas

a Task agen in- stanc

Raises

Hard ware spec tion- Fail- ure if es- sen- tial prop er- ties

could not be retrieved successfully.

Returns
The resulting state of inspection.

class i

Base
irc
dri
bas
Ins

get_prc
Return the properties of the interface

Returns
dictionary of <property name description entries

inspect
Inspect hardware

In-

spec hard ware to obtain the essential & additional hardware properties.

Paramet

`tas` a Task ager instan containing the node to act on.

Raises

Hard ware spec tion-Fail-ure, if unable to get essential hardware properties.

Returns

state

`validat`

Validate the driv spec info sup- plied

This meth val- i- date whe the driv prop- erty of the sup- plied node contains the required information for this driver to manage the node.

Paramet

tas a Task agen in- stanc con- tain- ing the node to act on.

Raises

In- valid Pa- ram- e- ter- Valu if re- quir driv at-is missing or invalid on the node.

ironic.drivers.modules.drac.job module

tribu

DRA Life cycle job specific meth ods

`ironic.`

Get the details of a Life cycle job of the node

Paramet

- **nod** an ironi node object.
- **job** ID of the Life cycle job.

Returns

a Job object from drac clien

Raises
DracOperation on an error from pyth drac

`ironic.`

List unfinished config jobs of the node

Paramet
`nod` an ironi node object.

Returns
a list of Job objects from drac clien

Raises

DracOperationI on an error from pyth drac

`ironic.`

Validate the job queu on the node

It raise an exception if an unfinished configu-

ration job exists. :param node: an ironic node object. :param name_prefix: A name prefix for jobs to validate. :raises: DracOperationError on an error from python-dracclient.

`ironic.`

Wait for job to complete

It

will wait for the job to com plete for 20 minutes and

raises timeout if job never complete within given interval of time. :param node: an ironic node object. :param retries: no of retries to make conductor wait. :raises: DracOperationError on exception raised from python-dracclient or a timeout while waiting for job completion.

ironic.drivers.modules.drac.management module

DRA man agemen interface

class i

Base *irc dri moc dra mar Dra*

Clas alias of class Drac Mar Mar agemen

This class provide

on-going support of the deprecate

idrac management interface implementation entrypoint.

All bug fixe and new feature shou be implemen in its

base class, DracWSManManagement. That makes them available to both the deprecated idrac and new idrac-wsman entrypoints. Such changes should not be made to this class.

class i

Base
irc
dri
moc
rec
man
Rec

iDR Red-fish interface for man relat actions

EXPORT_

IMPORT_

IMPORT_

clear_

Clea iDR job queu

Parame

tas

a Task agen in-stanc con-tain-ing the node to act on.

Raises

Red-fish-Er-ror on an er-ror.

export_

Ex-port the con-fig-u-ra-tion of the serv

Ex-port

the configuration of the serv agai whic the

step is run and stores it in specific format in indicated location.

Uses Dell Serv Configuration Profile (SC from *sush oem*

idrac library to get ALL configuration for cloning.

Parame

- **tas** A task from Task agen
- **exp** URI of location to save the config-

u-
ra-

tion to.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
miss
ing
con-
fig-
u-

ration name of a file to save the configuration to

Raises
Dra-
c-
Op-
er-
ata-
tion-
Erro
whe
no
man
a-
gage
for

Redfish system found or configuration export from SCP failed

Raises
Red-
fish-
Er-
ror
whe
load
ing
OEM
ex-
ten-
sion
faile

import_
Im-
port

and apply the configuration to the server.

Gets pre-crea configuration from storage by give location and imports that into given server. Uses Dells Server Configuration Profile (SCP).

Parame

- **`tas`** A task from Task agen
- **`imp`** URI of location to fetch desired config-

ration from.

u-

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
miss
ing
con-
fig-
u-

ration name of a file to fetch the configuration from

import_

Im-
port
and
ex-
port
con-
fig-
u-
ra-
tion
in
one
go.

Gets
pre-
crea
con-
fig-
u-
ra-
tion
from
stor-
age
by
give
nam

and imports that into given server. After that exports the configuration of the server against which the step is run and stores it in specific format in indicated storage as configured by Ironic.

Paramet

- **imp** URI of location to fetch desired configu-

ration from.

- **exp** URI of location to save the configura-

tion to.

known_g

Reset iDR to know good state

An iDR is reset to a know good state

by resetting

it and clearing its job queue.

Paramet

tas

a Task ager instanc containing the node to act

on.

Raises

Redfish Error on an error.

reset_i

Reset the iDR

Paramet

tas

a Task ager instanc containing the node to act

on.

Raises
Redfish-Error on an error.

class i

Base
iro
dri
bas
Man

clear_j
Clea the job queu

Paramet
tas
a
Task
agen
instanc
containing the
node
to
act
on.

Returns
Non
if
it
is
completed

Raises
Dra-c-Op-er-

a-
tionl
on
an
er-
ror
from
pyth
drac

get_boc

Get
the
cur-
rent
boot
de-
vice
for
a
node

Re-
turns
the
cur-
rent
boot
de-
vice
of
the
node

Paramet

tas
a
Task
ager
in-
stanc
con-
tain-
ing
the
node
to
act
on.

Raises

Dra-
c-

Op-
er-
a-
tionl
on
an
er-
ror
from
pyth
drac

Returns

a
dic-
tio-
nary
con-
tain-
ing:

boot_d
the
boot
de-
vice
one
of
irc
com
boc
or
Non
if
it

is unknown.

persist
whe
the
boot
de-
vice
will
per-
sist
to
all
fu-
ture
boot

or not, None if it is unknown.

get_pro

Re-turn the prop-er-ties of the in-ter-face

get_ser

Get sen-sors data

Paramet

tas

a Task ager in-stan

Raises

Faile To-Get-Sen-sor-Data whe get-ting the sen-sor data

fails.

Raises

Faile ToPa eSen sor-Data whe pars ing

sensor data fails

Returns
returns a consistent format dict of sensor data

grouped by sensor type, which can be processed by Ceilometer.

get_sup
Get a list of the supported boot devices

Parame
tas
a Task agen instanc containing the node to act

on.

Returns
A list with

the supported boot devices defined in *iro*

common.boot_devices.

known_g

Reset the iDR Clea the job queu

Paramet

tas a Task agen instan containing the node to act

on.

Returns

Non if it is complete

Raises

Drac-Op-er-a-tionl

on an error from pyth drac

reset_i

Reset the iDR

Paramet

tas a Task agen instance containing the node to act on.

Returns

Non if it is complete

Raises

Drac-Op-era-tion on an error from pyth drac

set_boo

Set the boot device for a node.

Set the boot device to use on next reboot of the node.

Paramet

- **`tas`** a Task ager instance containing the node to act on.
- **`dev`** the boot device, one of *`iro`* *`com`*

boo

- **per** Boo valu True if the boot device will persist to all future boots, False if not. Default: False.

Raises Invalid ParameterValu if an invalid boot device is specified.

validat Validate the driv spec info supplied

This meth validate whe the driv

property of the supplied

node contains the required information for this driver to manage the node.

Paramet

tas

a Task ager instanc containing the node to act

on.

Raises

InvalidParameterValu if required drive attribu

is missing or invalid on the node.

`ironic.`

Set the boot device for a node

Set

the boot device to use on next boot of the node

Paramet

- **nod** an ironi node object.
- **dev** the boot device, one of *irc com boc*
- **per** Boo valu True if the boot device will persist to

all future boots, False if not. Default: False.

Raises
Dra-

c-Operation on an error from pyth drac

ironic.drivers.modules.drac.power module

DRA pow interface

class i

Base *iro dri moc dra pow Dra*

Clas alias of class Drac Man Pow

This class provide ongoing support of the depre-

cate

idrac power interface implementation entrypoint.

All bug fixes and new features should be implemented in its

base class, DracWSManPower. That makes them available to both the deprecated idrac and new idrac-wsman entrypoints. Such changes should not be made to this class.

class i

Base *irc dri moc rec pow Rec*

iDR Red-fish interface for pow relat actions

Pres this class entirel defers to its base class a gene vend

independent Redfish interface. Future resolution of Dell EMC- specific incompatibilities and introduction of vendor value added should be implemented by this class.

class i

Bas *iro dri bas Pow*

Interface for pow relat actions

get_pow

Return the pow state of the node

Paramet

tas

a Task ager instan containing the node to act

on.

Returns

the pow state

one of *iro com sta*

Raises
In- valic Pa- ram- e- ter- Valu if re- quire DRA cre- den-

tials are missing.

Raises
Dra- c- Op- er- a- tionl on an er- ror from pyth drac

get_pro
Re- turn the prop- er- ties of the in- ter- face

reboot
Per- form

a reboot of the task node

Parame

- **tas** a Task ager instance containing the node to act on.
- **tim** time out (in seconds Unsupport by this interface.

Raises
In valid Parameter Valu if re-

quir
DR
cre-
den-

tials are missing.

Raises
Dra-
c-
Op-
er-
a-
tion
on
an
er-
ror
from
pyth
drac

set_pow
Set
the
pow
state
of
the
node

Parame

- `tas`
a
Task
agen
in-
stan
con-
tain-
ing
the
node
to
act

on.

- `pow`
a
pow
state

from *iro com sta*

- **tim** Tim to wait for the node to reac the re- ques state

When requested state is reboot, not used as not waiting then.

Raises
In- valic Pa- ram- e- ter- Valu if re- quir DRA cre- den-

tials are missing.

Raises
Dra- c- Op- er- a- tionl on an er- ror from pyth drac

validat

Validate the driv spec Nod pow info

This meth validate whe the driv property of the supplied

node contains the required information for this driver to manage the power state of the node.

Paramet

tas a Task ager instance containing the node to act

on.

Raises

Invalid Parameter Valu if requir

driv
at-
tribu

is missing or invalid on the node.

ironic.drivers.modules.drac.raid module

DRA
RAI
spe-
cific
meth
ods

class i
Base
iro
dri
moc
dra
rai
Dra

Clas
alias
of
class
Drac
Man
RAI

This
class
pro-
vide
on-
go-
ing
sup-
port
of
the
dep-
re-
cate

idrac RAID interface implementation entrypoint.

All
bug
fixes
and

new fea- tures shou be im- ple- men in its

base class, DracWSManRAID. That makes them available to both the deprecated idrac and new idrac-wsman entrypoints. Such changes should not be made to this class.

class i

Base *irc dri moc rec rai Rec*

iDR Red- fish in- ter- face for RAI re- latec ac- tions

In- clud iDR spe- cific ad- just- men for RAI re- latec ac- tions

create_

Create RAI configuration on the node

This meth creates the RAI configuration as read from

node.target_raid_config. This method by default will create all logical disks.

Paramet

- `tas` Task agen object containing the node
- `cre` Setting this to Fals indicate

not to create

root volume that is specified in the nodes target_raid_config. Default value is True.

- **cre** Setting this to Fals indicate not to create

non-root volumes (all except the root volume) in the nodes target_raid_config. Default value is True.

- **del** Setting this to True indicate to dele RAI con-

figuration prior to creating the new configuration. Default is False.

Returns state if RAI configuration is in prog asyn chro

or None if it is complete.

Raises
RedfishError if there is an error creating

the configuration

delete_

Dele RAI configuration on the node

Parame

tas

Task ager object containing the node

Returns

state (clea ing) or state (deploy men if deletion is progress asynchronously or None if it is complete.

in

class i

Base
irc
dri
bas
RAI

apply_c

Applies
RAI
configuration
on
the
give
node

Paramet

- **tas**
  A
  Task
  ager
  instanc
- **rai**
  The
  RAI
  configuration
  to
  apply.
- **cre**
  Setting

this to Fals in- di- cate not to cre- ate

root volume that is specified in raid_config. Default value is True.

- **cre** Set- ting this to Fals in- di- cate not to cre- ate

non-root volumes (all except the root volume) in raid_config. Default value is True.

- **del** Set- ting this to True in- di- cate to dele RAI con-

figuration prior to creating the new configuration.

Raises
In- valic Pa- ram- e- ter- Valu if

the RAI con- fig- u- ration is invalid.

Returns

state if RAI con- fig- u- ra- tion is in prog asyn chro or None if it is complete.

create_

Cre- ate the RAI con- fig- u- ra- tion.

This meth cre- ates the RAI con- fig- u- ra- tion on the give node.

Paramet

- **tas** a Task ager instanc containing the node to act

on.

- **cre** If True a root volume is created during RAI

configuration. Otherwise, no root volume is created. Default is True.

- **cre** If True non-root volume are created. If Fals no

non-root volumes are created. Default is True.

- **del** Setting this

to
Tru
in-
di-
cate
to
dele
RAI
con-

figuration prior to creating the new configuration. Default is False.

Returns
stat
(cle
ing)
or
stat
(de-
ploy
men
if
cre-
atio
is
in

progress asynchronously or None if it is completed.

Raises
Mis
ing-
Pa-
ram-
e-
ter-
Val
if
nod
is
mis
ing
or

empty.

Raises
Dra-
c-
Op-
er-
a-
tion
on
an
er-

ror
fron
pyth
drac

delete_

Dele
the
RAI
configuration.

Parame

tas

a
Task
ager
instanc
containing
the
node
to
act

on.

Returns

state
(clea
ing)
or
state
(deploy
men
if
dele
tion
is
in

progress asynchronously or None if it is completed.

Raises

Dra-c-Op-er-ation on an error from pyth drac

get_log

Get the RAI configuration of the node

Paramet

tas

a Task agen instan containing the node to act

on.

Returns

A dictionary of prop erties.

Raises

DracOpera-tionl on an error from pyth drac

get_pro

Return the properties of the interface

ironic.

Dele all pending chan on a RAI controll

Paramet

- **nod** an ironi node object.
- **rai** id of the RAI controll

Raises

Dra-
c-
Op-
er-
a-
tion
on
an
er-
ror
from
pyth
drac

`ironic.`

Con
vert
disk
RAI
sta-
tus

This
meth
con-
verts
the
re-
ques
phys
i-
cal
disk
from
RAI
to

JBOD or vice versa. It does this by only converting the disks that are not already in the correct state.

Paramet

- **nod**
an
iron
node
ob-
ject.
-

mod the mod to char the disk ei- ther to RAI or JBO

- **con** Dic- tio- nary of con- troll and cor- re- spon ing disk

ids to convert to the requested mode.

Returns
a dic- tio- nary con- tain- ing: - con- ver- sion a

dictionary that maps controller ids to the conversion results for that controller. The conversion results are a dict that contains: - The is_commit_required key with the value always set to True indicating that a config job must be created to complete disk conversion. - The is_reboot_required key with a RebootRequired enumerated value indicating whether the server must be rebooted to complete disk conversion.

Raises
DR C-

Op-
er-
a-
tionl
on
an
er-
ror
from
pyth
drac

`ironic.`

Free
up
the
for-
eign
driv

Paramet

- **nod**
  an
  ironi
  node
  ob-
  ject.
- **rai**
  id
  of
  the
  RAI
  con-
  troll

Returns
a
dic-
tio-
nary
con-
tain-
ing
-

The
is_c
need

with a boolean value indicating whether a config job must be created for the values to be applied. - The is_reboot_required key with a RebootRequired enumerated value indicating whether the server must be rebooted to clear foreign configuration.

key

Raises

Dra- c- Op- er- a- tionI on an er- ror from pyth drac

`ironic.`

Ap- ply all pend ing char on a RAI con- troll

Paramet

- **nod** an ironi node ob- ject.
- **rai** id of the RAI con- troll

- **reb** indicate whe a reboot job shou be automatically created with the config job. (optional, defaults to False)
- **rea** indicate RAI controll support realtime (optional, defaults to False)

Returns
id of the created job

Raises
DracOperation on an error from pyth

drac

`ironic.`

Create a single virtual disk on a RAI controll

The created virtual disk will be in pend ing state The DRA card will do the actual configuration once the changes are applied by calling the `commit_config` method.

Paramet

- **nod** an ironi node object.
- **rai**

id of the RAI con- troll

- **phy** ids of the phys i- cal disk
- **rai** RAI leve of the vir- tual disk
- **siz** size of the vir- tual disk
- **dis** nam of the vir- tual disk (op- tiona
- **spa** Num ber of span in

virtual disk (optiona

- **spa** Number of disk per span (optiona

Returns
a dictionary containing the commit_key with a

boolean value indicating whether a config job must be created for the values to be applied.

Raises
Dra-c-Op-er-a-tion on an error from pyth drac

`ironic.`

Dele a single virtual disk on a RAI controll

The dele virtual disk will be in pend ing state The DRA card will do the actual configuration once the changes are applied by calling the `commit_config` method.

Paramet

- **nod** an ironi node object.
- **vir** id of the virtual disk

Returns
a dictionary containing

the com mit_ key with a

boolean value indicating whether a config job must be created for the values to be applied.

Raises
Dra- c- Op- er- a- tion on an er- ror from pyth drac

`ironic.`
List the phys i- cal disk of the node

Paramet
nod
an ironi node ob- ject.

Returns
a list of Phys i- calD isk ob- jects from

drac clien

Raises
DracOperation on an error from pyth drac

`ironic.`
List the RAI controll of the node

Paramet
nod an ironi node object.

Returns
a list of RAI Controll objects from drac clien

Raises
DracOper-

a-
tionl
on
an
er-
ror
from
pyth
drac

ironic.

List
the
RAI
con-
fig-
u-
ra-
tion
set-
tings

Paramet

nod

an
ironi
node
ob-
ject.

Returns

a
dic-
tio-
nary
with
the
RAI
set-
tings
us-
ing
In-
stan-

ceID as the key. The attributes are RAIDEnumerableAttribute, RAIDStringAttribute and RAIDIntegerAttribute objects.

Raises

DRA
C-
Op-
er-
a-

tionl on error reporte back by

the DRAC interface

ironic.

List the virtual disk of the node

Paramet

nod an ironi node object.

Returns

a list of VirtualD isk objects from drac client

Raises

DracOperationl on an error from

pyth
drac

`ironic.`

Sets the RAI con- fig- u- ra- tion

It sets the penc ing_ pa- ram- e- ter for each of the at- tributes passed in. For the values to be applied, a config job must be created.

Paramet

- **nod** an ironi node ob- ject.
- **con** the ID of the RAI con- troll
- **set**

a dictionary containing the propose values,

with each key being the name of attribute and the value being the proposed value.

Returns

a dictionary containing: - The is_c key with

a boolean value indicating whether a config job must be created for the values to be applied. - The is_reboot_required key with a RebootRequired enumerated value indicating whether the server must be rebooted for the values to be applied. Possible values are true and false.

Raises

DRAC-Operation on error report back by

the DRAC interface

ironic.drivers.modules.drac.utils module

`ironic.`

Loa OEM man ager and ex- e- cute pass meth on it.

Kno iDR Red- fish sys- tem has only one man ager but as Red-fish schema allows a list this method iterates through all values in case this changes in future. If there are several managers, this will try starting from the first in the list until the first success.

Paramet

- **tas** a Task ager in- stan
- **pro** user frier nam

of meth to be execute Usec in

exception and log messages.

- **lam** meth to execute as lamb func tion with input

parameter OEM extension manager. Example: lambda m: m.reset_idrac() For older versions also support second input parameter Redfish manager itself when pass_manager set to True.

- **pas** whe to pass man ager itself to execute OEM

extension method. This is for backward compability, new functions must not pass manager, but acquire it internally. Will be removed in future.

Returns

Returne valu of lamb

Raises

Redfish-Error if cant execute OEM function either because there are no managers to the system, failed to load OEM extension or execution of the OEM method failed itself.

ironic.drivers.modules.drac.vendor_passthru module

DRA vend pass interface

class i

Base iro dri moc rec ver Rec

iDR Redfish interface for vendor_

Use the Redfish implemen ta- tion for ven- dor pass

class i

Base *irc* *dri* *moc* *dra* *ven* *Dra*

Clas alias of class Drac Man Ven- dor- Pass

This class pro- vide on- go- ing sup- port of the dep- re- catec

idrac vendor passthru interface implementation entrypoint.

All bug fixes and new fea- tures shou be implemented in its

base class, DracWSManVendorPassthru. That makes them available to both the deprecated idrac and new idrac-wsman entrypoints. Such changes should not be made to this class.

class i

Base *iro* *dri* *bas* *Ven*

Interface for DRA specific meth ods.

abandon

Aban don a BIO configuration job.

This meth is used to aban don a BIO configuration previously submitted through set_bios_config().

Paramet

- **tas** a Task ager in- stanc con- tain- ing the node to act

on.

- **kwa** not used

Raises
Dra- c- Op- er- a- tionI on an er- ror from pyth drac

commit_
Com mit a BIO con- fig- u- ra- tion job.

This

meth is used to commit a BIO configuration

job. submitted through set_bios_config().

Paramet

- **tas** a Task ager instance containing the node to act

on.

- **reb** indicates whe a reboot job shou be automatically created with the config job.
- **kwa** not

used

Raises
Drac-Operation on an error from pyth drac

Returns
A dictionary containing the `job` key with the id

of the newly created config job, and the `reboot_required` key indicating whether the node needs to be rebooted to start the config job.

get_bio
Get the BIO configuration.

This meth is used to retriev the BIO

settings from a node

Paramet

- **`tas`** a Task ager instanc containing the node to act on.
- **`kwa`** not used

Raises
DracOperation on an error from pyth drac

Returns
a dictionary containing BIO

settings

get_pro

Return the properties of the interface

list_un

List unfinished config jobs of the node

Parame

- **tas** a Task ager instan containing the node to act

on.

- **kwa** not used

Returns

a

dictionary containing the unf key; this key poin

to a list of dicts, with each dict representing a Job object.

Raises

DracOperationi on an error from pyth drac

set_bio

Cha BIO settings

This meth is used to chan the BIO settings on a node

Paramet

- **tas**

a Task ager in-stanc con-tain-ing the node to act

on.

- **kwa** a dic-tio-nary of {At-tribu Nam New Valu

Raises
Dra-c-Op-er-a-tionI on an er-ror from pyth drac

Returns
A dic-tio-nary con-tain-ing the `is_` key

with a Boo

value indicating whether commit_bios_config() needs to be called to make the changes, and the `is_reboot_required` key with a value of true or false. This key is used to indicate to the commit_bios_config() call if a reboot should be performed.

validat

Validate the drive spec info supplied

This meth validate whe the drive property of the supplied

node contains the required information for this driver to manage the power state of the node.

Parame

- **tas** a Task agen instance containing the node to act

on.

-

kwa
not
used

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
re-
quir
driv
at-
tribu

is missing or invalid on the node.

Module contents

ironic.drivers.modules.ibmc package

Submodules

ironic.drivers.modules.ibmc.management module

iBM
Man-
age-
men
In-
ter-
face

class i

Base
irc
dri
bas
Man

get_boo
Get
the
cur-
rent

boot device for a node

Parame
tas
A task from Task agen

Raises
Invalid Parameter Valu on malformed parameter(s)

Raises
Missing Parameter Valu on missing parameter(s)

Raises
IBM C-Con tion-Erro

whe it fails to connect to iBM

Raises

IBM CError when iBM responds an error informa-

tion

Returns

a dictionary containing:

boot_d

the boot device, one of *irc com boo* or Non if it

is unknown.

persist

Boo

valu or Non True if the boot de- vice per- sists Fals

otherwise. None if its disabled.

get_boo

Get the cur- rent boot mod for a node

Pro- vide the cur- rent boot mod of the node

Paramet

tas A task from Task agen

Raises

In- valid Pa- ram- e- ter- Valu on

malformed parameter(s)

Raises
MissingParameterValue on missing parameter(s)

Raises
IBMC-Connection-Erro when it fails to connect to iBM

Raises
IBMCError when iBM respon an error information

Returns

The boot mod one of `irc` `com` `boo` or Non if it is unknown.

get_pro

Return the properties of the interface

Returns

dictionary of <pro erty nam description entries

get_ser

Get sensors data

Not implemen

for this driv

Raises
Notl ple- men ed- Er- ror

get_sup
Get a list of the sup- port boot de- vice

Paramet
tas a task from Task agen

Raises
In- valic Pa- ram- e- ter- Valu on mal- form pa- ram- e-

ter(s)

Raises
Miss ing- Pa- ram- e-

ter- Valu on miss ing pa- ram- e-

ter(s)

Raises
IBM C- Con tion- Erro whe it fails to con- nect to iBM

Raises
IBM CEr- ror whe iBM re- spor an er- ror in- for- ma-

tion

Returns
A list with the sup- porte boot de- vice de- fined

in *irc*

common.boot_devices.

get_sup

Get a list of the support boot mod

Paramet

tas

A task from Task ager

Returns

A list with the support boot mod defined in *irc* *com*

boot_modes. If boot mode support cant be determined, empty list is returned.

inject_

Inject NM Non Mas able Interrupt

Inject NM (Nor

Mas
able
In-
ter-
rupt
for
a
node
im-
me-
diately.

Parame
tas
A
Task
ager
in-
stan
con-
tain-
ing
the
node
to
act
on.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
on
mal-
form
pa-
ram-
e-
ter(s)

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
on

miss
ing
parameter(s)

Raises
IBM
C-
Con
tion-
Erro
whe
it
fails
to
connect
to
iBM

Raises
IBM
CEr-
ror
whe
iBM
re-
spon
an
error
information

set_boo
Set
the
boot
device
for
a
node

Paramet

- **tas**
A

task
fron
Task
ager

- **dev**
  The
  boot
  de-
  vice
  one
  of
  `irc`
  `com`
  `boc`

- **per**
  Boo
  valu
  True
  if
  the
  boot
  de-
  vice
  will
  per-
  sist
  to

all future boots, False if not. Default: False.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
on
mal-
form
pa-
ram-
e-

ter(s)

Raises
Miss
ing-
Pa-

ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-

ter(s)

Raises
IBM
C-
Con
tion-
Erro
whe
it
fails
to
con-
nect
to
iBM

Raises
IBM
CEr-
ror
whe
iBM
re-
spon
an
er-
ror
in-
for-
ma-

tion

set_boo
Set
the
boot
mod
for
a
node

Set

the boot mod to use on next re-boot of the node

Parame

- **tas** A task from Task agen
- **mod** The boot mod one of *iro com boo*

Raises In-valid Pa-ram-e-ter-Valu on mal-form pa-ram-e-

ter(s)

Raises Miss ing-

Pa- ram- e- ter- Valu on miss ing pa- ram- e-

ter(s)

Raises

IBM C- Con tion- Erro whe it fails to con- nect to iBM

Raises

IBM CEr- ror whe iBM re- spon an er- ror in- for- ma-

tion

validat

Val- i- date the drive in- for- mation need by the iBM driver.

Parame

tas

A Task ager instanc containing the node to act on.

Raises

Invalid ParameterValu on malform parameter(s)

Raises

Missing-ParameterValu on miss ing parame-

ter(s)

ironic.drivers.modules.ibmc.mappings module

iBM
and
Iron
con-
stan
map
ping

ironic.drivers.modules.ibmc.power module

iBM
Pow
In-
ter-
face

class i
Base
irc
dri
bas
Pow

get_pow
Get
the
cur-
rent
pow
state
of
the
task
node

Parame
tas
A
Task
ager
in-
stand
con-
tain-
ing

the node to act on.

Returns
A pow state One of *irc com sta*

Raises
InvalidParameterValue on malformed parameter(s)

Raises
MissingParameterValue on missing parameter(s)

Raises
IBM C-Con tion-

Erro whe it fails to connect to iBM

Raises
IBM CError whe iBM respon an error information

get_pro
Return the properties of the interface

Returns
dictionary of <pro erty nam descrip tion entries

get_sup

Get a list of the supporte pow state

Paramet

tas

A Task agen instanc containing the node to act

on. Not used by this driver at the moment.

Returns

A list with the supporte pow state defined in *iro* *com*

states.

reboot

Perform a hard reboot of the task

node

Paramet

- **tas** A Task ager in- stanc con- tain- ing the node to act

on.

- **tim** Time to wait for the node to be- com pow- ered on.

Raises
In- valic Pa- ram- e- ter- Valu on mal- form pa- ram- e-

ter(s)

Raises
Miss

ingParameterValu if a requir parameter is missing.

Raises
IBM C-Con tion-Erro whe it fails to connect to iBM

Raises
IBM CError whe iBM respor an error information

set_pow
Set the pow state of the task

nod

Paramet

- **tas** A Task agen instanc containing the node to act on.
- **pow** Any pow state from *irc com sta*
- **tim** Time to wait for the node to reac the reques state

Raises InvalidParameterValu

on mal-form pa-ram-e-

ter(s)

Raises
Miss ing-Pa-ram-e-ter-Valu if a re-quir pa-ram-

eter is missing.

Raises
IBM C-Con tion-Erro whe it fails to con-nect to iBM

Raises
IBM CEr-ror whe iBM re-spon an er-ror in-for-ma-

tion

validat

Validate the driv information need by the iBM driver.

Parame

tas

A Task ager instanc containing the node to act on.

Raises

Invalic Parameter-Valu on malform parame- ter(s)

Raises

Miss ing-

Parameter-Valu
on
miss
ing
parameter(s)

ironic.drivers.modules.ibmc.raid module

iBM
RAI
configuration
spe-
cific
meth
ods

class i
Base
irc
dri
bas
RAI

Implementation
of
RAI
Interface
for
iBM

RAID_AE

apply_c

Applies RAI configuration on the given node

Paramet

- **tas** A Task agen instanc
- **rai** The RAI configuration to apply.
- **cre** Setting this to Fals indicates not to create

root volume that is specified in raid_config. Default value is True.

- **cre** Setting this to Fals indicate not to create non-root volumes (all except the root volume) in raid_config. Default value is True.
- **del** Setting this to True indicate to dele RAI configuration prior to creating the new configuration.

Raises
InvalidParameterValu if the RAI configuration is invalid.

Returns
state if

RAI configuration is in prog asyn chro or None if it is complete.

`create_`

Create a RAI configuration.

This meth creates a RAI configuration on the give node.

Paramet

- **`tas`** a Task agen instanc
- **`cre`**

If True a root volume is created during RAI

configuration. Otherwise, no root volume is created. Default is True.

- `cre` If True non-root volumes are created. If Fals no

non-root volumes are created. Default is True.

- `del` Setting this to True indicates to dele RAI configuration prior to creating the new configuration. Default is False.

Raises
MissingParameter-

Valu if node is missing or

empty after skipping root volume and/or non-root volumes.

Raises

IBM CError, on failure to execute step

delete_

Dele the RAI configuration.

Paramet

tas

a Task ager instanc containing the node to act

on.

Returns

state if clear ing

operation in prog asynchro or

states.DEPLOYWAIT if deploy operation in progress synchronously or None if it is completed.

Raises
IBM CEr ror, on failure to execute step

get_pro
Return the prop erties of the interface

Returns
dictionary of <pro erty nam description entries

ironic.drivers.modules.ibmc.utils module

iBM
Driv
com
mon
utils

ironic.
Dec-
o-
ra-
tor
to
han-
dle
iBM
clien
ex-
cep-
tion.

Dec-
o-
rated
func
tions
mus
take
a
`Tas`
as
the
first
pa-
ram-
eter.

ironic.
Pars
the
in-
for-
ma-
tion
re-
quire
for
Iron
to
con-

nect

to iBMC.

Paramet

nod

an Iron node object

Returns

dictionary of parameters

Raises

Invalid ParameterValu on malform parame-

ter(s)

Raises

MissingParameterValu on missing parame-

ter(s)

`ironic.`

ironic.drivers.modules.ibmc.vendor module

iBM
Ven-
dor
In-
ter-
face

class i
Base
irc
dri
bas
Ven

boot_up
List
boot
type
or-
der
of
the
node

Parame

- **tas**
A
Task
ager
in-
stanc
con-
tain-
ing
the
node
to
act

on.

- **kwa**
Not
used

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
kwa
does
not
con-
tain

method.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu

Raises
IBM
C-
Con
tion-
Erro
whe
it
fails
to
con-
nect
to
iBM

Raises
IBM
CEr-
ror
whe
iBM
re-
spon
an
er-
ror
in-
for-

ma-

tion

Returns

A dic- tio- nary con- tain- ing node boot up se- quen in

ascending order.

get_pro

Re- turn the prop er- ties of the in- ter- face

Returns

dic- tio- nary of <pro erty nam de- scrip tion en- tries

get_rai

List RAI con- troll sum mary

info of the node

on.

Parame

- **tas** A Task ager in-stan con-tain-ing the node to act
- **kwa** Not used

Raises
IBM C-Con tion-Erro whe it fails to con-nect to iBM

Raises
IBM CEr-ror whe iBM re-spon an er-

ror information

Returns

A list of dictionaries, every dictionary represents a RAID controller summary of node.

validat

Validate vend spec actions

If invalid raise an exception; otherwise returns Non

Paramet

- **tas** A task from

Task
agen

- **met**
  Metl
  to
  be
  val-
  i-
  date

- **kwa**
  Info
  for
  ac-
  tion.

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion
if
metl
can
not
be
map

to the supported interfaces.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
kwa
does
not
con-
tain

method.

Raises
Miss
ing-

Pa-
ram-
e-
ter-
Valu

Module contents

ironic.drivers.modules.ilo package

Submodules

ironic.drivers.modules.ilo.bios module

iLO
BIO
In-
ter-
face

class i
Base
irc
dri
bas
BIC

apply_c
Ap-
plies
the
pro-
vide
con-
fig-
u-
ra-
tion
on
the
node

Paramet

- **tas**
a
Task

agen in- stanc

- **set** Set- tings in- tend to be ap- plied on the node

Raises
Nod Clea ing- Fail- ure, on fail- ure to ex- e- cute of

clean step.

Raises
In- stanc ploy Fail- ure, on fail- ure to ex- e- cute of

deploy step.

cache_b
Stor the

BIO set- tings in the data

Parame

tas a Task agen in- stan

Raises

Nod Clea ing- Fail- ure, on fail- ure to ex- e- cute of

clean step.

Raises

In- stan ploy Fail- ure, on fail- ure to ex- e- cute of

deploy step.

factory

Re- set the BIO settings to factory configuration.

Parame

tas a Task ager instanc

Raises Nod Clea ing-Failure, on failure to execute of clean step.

Raises Instanc ploy Failure, on failure to execute of deploy step.

get_pro

Return

the prop erties of the interface

Returns
dictionary of <pro erty nam description entries

validat
Chec that drive contains required ILO credentials

Validates whe the drive property of the supplied tasks node

contains the required credentials information.

Paramet

tas
a task from Task agen

Raises
In- valid Pa- ram- e- ter- Valu if re- quire iLO pa- ram-

eters are not valid.

Raises
Miss ing- Pa- ram- e- ter- Valu if a re- quire pa- ram-

eter is missing.

ironic.drivers.modules.ilo.boot module

Boo In- ter- face for iLO drive and its

supporting meth ods.

class i

Base *iro dri moc pxe PXE*

clean_u

Clea up the boot of instanc

This meth clea up the PXE environmen that was setu for booting the instance. It unlinks the instance kernel/ramdisk in the nodes directory in tftproot and removes its PXE config. In case of UEFI iSCSI booting, it cleans up iSCSI target information from the node.

Parame

tas a task from Task agen

Returns

Non

Raises
IloOperationI if som operation on iLO failed.

prepare

Prepare the boot of instan

This meth prepare the boot of the instanc after read ing relevant information from the nodes instance_info. In case of netboot, it updates the dhcp entries and switches the PXE config. In case of localboot, it cleans up the PXE config. In case of boot from volume, it updates the iSCSI info onto iLO and sets the node to boot from UefiTarget boot device.

Parame
tas a task from Task agen

Returns
Non

Raises
Ilo- Op- er- a- tionl if som op- er- a- tion on iLO

failed.

prepare

Pre- pare the boot of Iron ram us- ing PXE

This meth pre- pare the boot of the de- ploy or res- cue ram

after reading relevant information from the nodes driver_info and instance_info.

Paramet

- **tas** a task from Task

ager

- **ram** the parameters to be pass to the ram

Returns
Non

Raises
Missing-Parameter-Valu if som information

is missing in nodes driver_info or instance_info.

Raises
Invalid-Parameter-Valu if som information

provided is invalid.

Raises
IronicEx

cep- tion, if som pow or set boot boot de- vice

operation failed on the node.

Raises
Ilo- Op- er- a- tionI if som op- er- a- tion on iLO

failed.

class i

Base *iro dri bas Boo*

capabil

clean_u
Clea up the boot of in- stan

This meth clea up

the en- vi- ron- men that was setu for boot

ing the instance.

Parame

tas

A task from Task agen

Returns

Non

clean_

Clea up the boot of iron ramd

This meth clea up the en- vi- ron- men that was setu for boot

ing the deploy ramdisk.

Parame

tas

A task from Task

ager

Returns
Non

get_pro
Return the properties of the interface

Returns
dictionary of <pro erty nam description entries

prepare
Prepare the boot of instanc

This meth prepare the boot of the instanc after read

ing

relevant information from the nodes instance_info. It does the following depending on boot_option for deploy:

- If the boot reques for this deploy is local or image is a whole disk image, then it sets the node to boot from disk.
- Otherwise it finds the boot ISO sets the node boot option to UEFIHTTP and sets the URL as the boot ISO to boot the instance image.

Parame
: **`tas`** a task from Task ager

Returns
: Non

Raises
: IloOperation

if som op-er-a-tion on iLO failed.

Raises
In-stan ploy Fail-ure, if its try to boot iSCS vol-ume in BIOS boot mode.

prepare
Pre-pare the boot of de-ploy ramd us-ing UEF HTT boot

This meth pre-pare the boot of the de-ploy or rescue ram

after reading relevant information from the nodes driver_info and instance_info.

Paramet

- **tas** a task from Task ager
- **ram** the parameters to be pass to the ram

Returns
Non

Raises
MissingParameterValu if some information

is missing in nodes driver_info or instance_info.

Raises
InvalidParam

e-
ter-
Valu
if
som
in-
for-
ma-
tion

provided is invalid.

Raises

Iron
icEx
cep-
tion,
if
som
pow
or
set
boot
boot
de-
vice

operation failed on the node.

Raises

Ilo-
Op-
er-
a-
tionl
if
som
op-
er-
a-
tion
on
iLO

failed.

validat

Val-
i-
date
the
de-
ploy
men
in-
for-

mation for the tasks node.

This meth validate whe the driv and/ instan properties of the tasks node contains the required information for this interface to function.

Parame

`tas` A Task ager instan containing the node to act on.

Raises

InvalidParameterValu on malformparameter(s)

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-

ter(s)

validat

Val-
i-
date
that
the
node
has
re-
quir
prop
er-
ties
for

inspection.

Parame

tas

A
Task
agen
in-
stan
with
the
node
be-
ing
chec

Raises
Miss
ing-
Pa-
ram-
e-
ter-

Valu if node is missing one

or more required parameters

Raises

Unsupported-Driv ten-sion

validat

Validate that the node has required prop erties for

rescue.

Paramet

tas

a Task ager instanc with the node being chec

Raises

Missing-Param-

e-
ter-
Valu
if
node
is
miss
ing
one

or more required parameters

class i

Base
irc
dri
bas
Boc

capabil

clean_u

Clea
up
the
boot
of
in-
stan

This
meth
clea
up
the
en-
vi-
ron-
men
that
was
setu
for
boot

ing the instance. It ejects virtual media. In case of UEFI iSCSI booting, it cleans up iSCSI target information from the node.

Parame

tas

a
task
from

Task
ager

Returns
Non

Raises
Ilo-
Op-
er-
a-
tionl
if
som
op-
er-
a-
tion
on
iLO

failed.

clean_u

Clea
up
the
boot
of
iron
ram

This
meth
clea
up
vir-
tual
me-
dia
de-
vice
setu
for
the
de-

ploy or rescue ramdisk.

Parame
tas
a
task
from
Task
ager

Returns
Non

Raises
IloOperationtionI if som operation on iLO

failed.

get_pro
Return the properties of the interface

Returns
dictionary of <pro erty nam description entries

prepare
Prepare the boot of in-

stan

This meth prepare the boot of the instan after read ing

relevant information from the nodes instance_info. It does the following depending on boot_option for deploy:

- If the boot mod is uefi and its boot ing from volume

then it sets the iSCSI target info and node to boot from UefiTarget boot device.

- If not boot from volume and the boot reques for this

deploy is local or image is a whole disk image, then it sets the node to boot from disk.

- Oth-

er-
wise
it
finds
the
boot
ISO
to
boot
the
in-
stan

image, attaches the boot ISO to the bare metal and then sets the node to boot from CDROM.

Parame

tas
a
task
from
Task
ager

Returns
Non

Raises
Ilo-
Op-
er-
a-
tion
if
som
op-
er-
a-
tion
on
iLO

failed.

Raises
In-
stan
ploy
Fail-
ure,
if
its
try
to
boot
iSC

vol-
ume

in BIOS boot mode.

prepare

Pre-
pare
the
boot
of
de-
ploy
ram
us-
ing
vir-
tual
me-

dia.

This
meth
pre-
pare
the
boot
of
the
de-
ploy
or
res-
cue
ram

after reading relevant information from the nodes driver_info and instance_info.

Parame

- **tas**
  a
  task
  from
  Task
  agen
- **ram**
  the
  pa-
  ram-
  e-

ters to be pass to the ram

Returns
Non

Raises
Miss ing-Pa-ram-e-ter-Valu if som in-for-ma-tion

is missing in nodes driver_info or instance_info.

Raises
In-valid Pa-ram-e-ter-Valu if som in-for-ma-tion

provided is invalid.

Raises
Iron icEx cep-tion, if som pow or set

boot
boot
device

operation failed on the node.

Raises
IloOperation if som operation on iLO failed.

validat
Validate the deploy men information for the tasks node.

Paramet
`tas` a Task agen instance containing the node to act on.

Raises
Invalic Parameter-Valu if som information is invalid.

Raises
Missing-Parameter-Valu if kernel_ and ramc are missing in the Glance image or kernel and ramdisk not provided in instance_info for non-Glance image.

validat
Validate that the node has require properties for inspection.

Paramet
`tas`
A Task agen

in-
stan
with
the
node
be-
ing
chec

Raises
Miss-
ing-
Pa-
ram-
e-
ter-
Valu
if
node
is
miss
ing
one

or more required parameters

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion

validat
Val-
i-
date
that
the
node
has
re-
quire
prop
er-
ties
for

rescue.

Paramet
tas
a

Task agen in- stanc with the node be- ing chec

Raises
Miss ing- Pa- ram- e- ter- Valu if node is miss ing one

or more required parameters

class i

Base *irc* *dri* *moc* *ipx* *iPX*

clean_u

Clea up the boot of in- stanc

This meth clea up the PXE en- vi-

ron-
men
that
was
setu
for

booting the instance. It unlinks the instance kernel/ramdisk in the nodes directory in tftproot and removes its PXE config. In case of UEFI iSCSI booting, it cleans up iSCSI target information from the node.

Parame

tas
a task from Task ager

Returns
Non

Raises
IloOperation if som operation on iLO

failed.

prepare

Prepare the boot of instan

This meth prepare the boot of the

instance after reading

relevant information from the nodes instance_info. In case of netboot, it updates the dhcp entries and switches the PXE config. In case of localboot, it cleans up the PXE config. In case of boot from volume, it updates the iSCSI info onto iLO and sets the node to boot from UefiTarget boot device.

Paramet

`tas`

a task from Task ager

Returns

Non

Raises

Ilo-Operation if some operation on iLO

failed.

`prepare`

Prepare the boot of Iron ram using PXE

This meth prepare the

boot of the deploy or rescue ram

after reading relevant information from the nodes driver_info and instance_info.

Parame

- **tas** a task from Task agen
- **ram** the parameters to be pass to the ram

Returns Non

Raises MissingParameterValu if som information

is missing in nodes driver_info or instance_info.

Raises
In- valic Pa- ram- e- ter- Valu if som in- for- ma- tion

provided is invalid.

Raises
Iron icEx cep- tion, if som pow or set boot boot de- vice

operation failed on the node.

Raises
Ilo- Op- er- a- tionl if som op- er- a- tion on iLO

failed.

`ironic.`

Gets the driv

specific Nod deploy men info

This meth validate whe the driv prop erty of the supplied node contains the required information for this driver to deploy images to the node.

Paramet

- **`nod`** a single Nod
- **`mod`** Label indicating a deploy or rescue operation being carried out on the node. Supported values are deploy and rescue. Defaults to deploy, indicating deploy operation is being carried out.

Returns
A dict

with
the
driv
val-
ues.

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
any
of
the
re-
quir

parameters are missing.

`ironic.`

Com
mon
prep
tory
step
for
all
iLO
driv

This
meth
per-
form
com
mon
prep
tory
step
re-
quir
for
all
driv

1. Power off node 2. Disables secure boot, if it is in enabled state. 3. Updates boot_mode capability to uefi if secure boot is requested. 4. Changes boot mode of the node if secure boot is disabled currently.

Paramet

`tas`

a

Task ager in- stanc con- tain- ing the node to act on.

Raises
Ilo- Op- er- a- tionE if som op- er- a- tion on iLO failed.

ironic.drivers.modules.ilo.common module

Com mon func tion- al- i- ties shar be- twee dif- fer- ent iLO modules.

`ironic.`
Nod is in

Finished Post post state

`ironic.`

Node is in In-Post Discovery post state

`ironic.`

Node is in In-Post post state

`ironic.`

Node is in Null post state

`ironic.`

Node is in Pow post state

`ironic.`

Node is in Reset post state

`ironic.`

Nod

is in Un- know post state

`ironic.`

Nod sup- port both lega BIO and UEF boot mod

`ironic.`

Nod sup- port only lega BIO boot mod

`ironic.`

Nod sup- port only UEF boot mod

`ironic.`

Add cer- tifi- cate to the node

Add cer- tifi- cate to the

nod base on the driv info provide

Paramet

- **tas** a Task ager instanc containing the nod to act on.
- **cer** List of certificate to be adde to the nod If None, certificates from path configured in webserver_verify_ca will be added to the node.

Raises

IloOperation on an error from Ilo- Clie li-

brary.

Raises
Ilo- Op- er- a- tion Sup- port if re- triev ing post state

is not supported on the server.

Raises
In- valic Pa- ram- e- ter- Valu if any of the re- quir

parameters are invalid.

`ironic.`

At- tach the give url as vir- tual me- dia on the node device

Paramet

- **nod** an iron node object.
- **dev** the virtual media device to attach
- **url** the http url to attach as the virtual media

Raises
Ilo-Operation if insert virtual media

faile

ironic.

Clea a node af- ter a vir- tual me- dia boot

This meth clea up a node af- ter a vir- tual me- dia boot

It deletes the floppy image if it exists in CONF.ilo.swift_ilo_container or web server. It also ejects both virtual media cdrom and virtual media floppy.

Paramet

tas

a Task ager in- stanc con- tain- ing the node to act on.

ironic.

Clea any certificate added to the node

Clear the certificate added to the node as part of any Iron

operation

Paramet

- **tas** a Taskager instance containing the node to act

on.

- **cer** List of certificates to be removed

from
node
If

None, all the certificates present on the node will be removed.

Raises

Ilo-
Op-
er-
a-
tion
on
an
er-
ror
from
Ilo-
Clie
li-

brary.

Raises

Ilo-
Op-
er-
a-
tion
Sup-
port
if
re-
triev
ing
post
state

is not supported on the server.

`ironic.`

Up-
load
the
give
im-
age
to
swif

This
meth
copi
the

give image to swif

Paramet

- **sou** The absolute path of the image file whic need to be copied to swift.
- **des** The nam of the object that will contain the copi image.

Raises
Swi Operationl if any operation

with
Swi

fails.

Returns
temp
url
from
swif
after
the
sour
image
is
up-
load

`ironic.`

Cop
the
give
image
to
the
http
web
serv

This
meth
copi
the
give
image
to
the
http_
location.
It
enables read-write access to the image else the deploy fails as the image file at the web_server url is inaccessible.

Paramet

-

sou The absolute path of the image file whic need to be copied to the web server root.

- **des** The nam of the file that will contain the copi image.

Raises
Imag Uploa Faile exception if copy ing the sour file to the web server fails.

Returns
image url after

the sour im- age is up- load

`ironic.`

Re- mov the tem- po- rary flop im- age.

It re- mov the flop im- age cre- ated for de- ploy :para node

an ironic node object.

`ironic.`

Ejec vir- tual me- dia de- vice

This meth ejec vir- tual me- dia flop and cdro

Paramet

tas

a Task ager instan containing the node to act

on.

Returns

Non

Raises

IloOperationl if som error was encoun terec

while trying to eject virtual media floppy or cdrom.

`ironic.`

Get the current boot mod for a node

Paramet

nod

an ironi node ob-

ject.

Raises
IloOperationE if failed to fetch boot mode

Raises
IloOperationNSupported if node does not support

getting pending boot mode.

`ironic.`

Gets an IloClient object from proliantutils library

Given an ironic node object, this meth

give back a IloClie object to do operations on the iLO.

Paramet

`nod` an ironi node object.

Returns

an IloClie object.

Raises

InvalicParameterValu on invalic inputs.

Raises

MissingParameterValu if som man tory information is missing on the node

`ironic.`

Retriev cur-rent en-able state of UEF se-cure boot on

the node

Re-turn the cur-rent en-able state of UEF se-cure boot on

the node.

Paramet

tas a task from Task ager

Raises

Miss ing-Pa-ram-e-ter-Valu if a re-quir iLO parameter is missing.

Raises
Ilo- Op- er- a- tionI on an er- ror from Ilo- Clie li-

brary.

Raises
Ilo- Op- er- a- tionI Sup- port if UEF se- cure boot is

not supported.

Returns
Boo valu in- di- cat- ing cur- rent state of UEF se- cure

boot on the node.

`ironic.`
Get the

current state of system POS

Paramet

nod

an ironi node object.

Returns

POS state of the serv The validation state are:-

null, Un-

known, Reset, PowerOff, InPost, InPostDiscoveryComplete and FinishedPost.

Raises

IloOperation on an error from IloClie li-

brary.

Raises

IloOperationl Supporte if retrieving post state

is not supported on the server.

`ironic.`

Gets the drive specific Nod info

This meth validate whe the drive property of the supplied

node contains the required information for this driver.

Paramet

nod an ironi Nod object.

Returns

a dict containing information

tion from driv (or whe

applicable, config values).

Raises
In- valic Pa- ram- e- ter- Valu if any pa- ram- e- ters

are incorrect

Raises
Miss ing- Pa- ram- e- ter- Valu if som man tory in- for-

mation is missing on the node

`ironic.`

Re- mov the give im- age from swif

This meth

removes the given image name from swift. It deletes the image if it exists in CONF.ilo.swift_ilo_container

Paramet

- **obj** The name of the object which needs to be removed from swift.
- **ass** string to depict the component this object is associated to.

`ironic.`

Removes the

give im- age from the con- fig- ured web serv

This meth re- mov the give im- age from the http_ lo- ca- tion,

if the image exists.

Paramet

obj The nam of the im- age file whic need to be re-

moved from the web server root.

`ironic.`

Re- mov (dele the file or list of files

This meth only accept single or list of files to dele If single file is passed, this method removes (deletes) the file. If list of files is passed, this method removes (deletes) each of the files iteratively.

Paramet

fil a single or a list of file path

`ironic.`

Sets the node to boot using boot for the next boot

Paramet

- **nod** an ironi node object.

- **boo** Nex boot mod

Raises
Ilo-Op-er-a-tionI if set-ting boot mod faile

ironic.
En-able or dis-able UEF Se-cure Boo for the next boot

En-able or dis-able UEF Se-cure Boo for the next boot

Paramet

- **tas** a

task from Task agen

- **fla** Boo valu True if the se-cure boot to be en-able

in next boot.

Raises
Ilo-Op-er-a-tion on an er-ror from Ilo-Clie li-

brary.

Raises
Ilo-Op-er-a-tion Sup-port if UEF se-cure boot is

not supported.

`ironic.`

Sets up system to boot from UEFI-HTT boot device

Sets the one-time boot device to UEFI-HTT base on the argument supplied.

Paramet

- **`tas`** a Task agen instan containing the node to act on.
- **`isc`**

ISO URI to be set to boot from

- **per** Indicates whe the system shou be set to boot

from the given device one-time or each time.

Raises
IloOperation on an error from IloClie li-

brary.

Raises
IloOperation Support if retriev ing

post
state

is not supported on the server.

`ironic.`

Attach virtual media and sets it as boot device

This meth attach the give boot ISO as virtual media, prepares the arguments for ramdisk in virtual media floppy.

Paramet

- **`tas`** a Task ager instan containing the node to act

on.

either of below:

It is assumed that the image object is present in CONF.ilo.swift_ilo_container;

`<glance-image-uuid>` or just `<glance-image-uuid>`;

- **`isc`** a boot ISO image href to attach to. Shou be
  - A Swi object - It shou be of format `swi`
  - A Glar image - It shou be format `gla` /
  - An HTT

URI

- **ram** the options to be pass to the ram in virtual

media floppy.

Raises
Image-Creation Fail if it faile whil creating the

floppy image.

Raises
Ilo-Operation if som operation on iLO

failed.

ironic.

Sets up the node to boot from the given ISO image.

This method attaches the given boot on the node and passes the required parameters to it via virtual floppy image.

Paramet

- **tas** a Task instance containing the node to act on.
- **boo** a boot ISO

image to attach to. Shou be either of below:

- A Swi object - It shou be of format swi It is assumed that the image object is present in CONF.ilo.swift_ilo_container;
- A Gla image - It shou be format gla / `<glance-image-uuid>` or just `<glance-image-uuid>`;
- An HTT URI

- **par** the param-

e-
ters
to
pass
in
the
vir-
tual
flop

image in a dictionary. This is optional.

Raises
Im-
age-
Cre-
atio
Fail
if
it
faile
whil
cre-
at-
ing
the

floppy image.

Raises
Swi
Op-
er-
a-
tion
if
any
op-
er-
a-
tion
with
Swi

fails.

Raises
Ilo-
Op-
er-
a-
tion
if
at-
tach
ing

virtual media

failed.

`ironic.`

Update instan with boot mod to be used for deploy

This meth updates instan with boot mod to be used for de-

ploy if node properties[capabilities] do not have boot_mode. It sets the boot mode on the node.

Paramet

tas

Task object.

Raises

IloOperationl if setting boot

mod
faile

`ironic.`

Updated
ipmi
properties
to
node
driv

Paramet
tas
a
task
from
Task
ager

`ironic.`

Verifies
chec
sum
(md
of
image
file
agai
the
expected one.

This
meth
generates
the
chec
sum
of
the
image
file

on the fly and verifies it against the expected checksum provided as argument.

Paramet

- **ima** location of image file who chec sum is verified.
- **exp** chec sum to be chec agai

Raises
ImageR fValidationFail if invali file path or verification fails.

ironic.drivers.modules.ilo.console module

iLO Deploy Driv and supporting meth ods.

class i

Base *iro dri moc ipm IPM*

A Con sole terface that uses ipmitool and shellinab

get_pro

Return the prop erties of the interface.

Returns dictionary of <pro erty nam description en tries

validat

Validate the Nod console info

Paramet

tas a task from Task agen

Raises

Invalid Parameter Valu

Raises

Missing Parameter Valu whe a required param eter is missing

ironic.drivers.modules.ilo.firmware_processor module

Firm file processor

class i

Base obj

Firm image location class

This class acts as a wrap per class for the firm image location. It primarily helps in removing the firmware files from their respective locations, made available for firmware update operation.

remove

Expose meth to remov the wrap firm file

This meth gets over rid- den by the re- mov meth for the re- spective type of firmware file location it wraps.

class i

Base obj

Firm file pro- ces- sor

This class help in dow load ing the firm file from url, ex- tract ing the firmware file (if its in compact format) and makes it ready for firmware update operation. In future, methods can be added as and when required to extend functionality for different firmware file types.

process

Pro- cess the firm file from the url

This is the template meth whic dow load the firm file from url, verifies checksum and extracts the firmware and makes it ready for firmware update operation. `_download_fw_to` method is set in the firmware processor object creation factory method, `get_fw_processor()`, based on the url type. :param node: a single Node. :param expected_checksum: checksum to be checked against. :returns: wrapper object of raw firmware image location :raises: IloOperationError, on failure to process firmware file. :raises: ImageDownloadFailed, on failure to download the original file. :raises: ImageRefValidationFailed, on failure to verify the checksum. :raises: SwiftOperationError, if upload to Swift fails. :raises: ImageUploadFailed, if upload to web server fails.

`ironic.`

Validate the firm image info and returns the retrieved values.

Paramet

`fir` dict object containing the firm image info

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
for
miss
ing
field
(or
val-

ues) in image info.

Raises

In-
valid
Pa-
ram-
e-
ter-
Valu
for
un-
sup-
port
firm
com

ponent

Returns

tu-
ple
of
firm
url,
chec
sum
com
po-
nent
whe
the
firm

update is ilo based.

`ironic.`

Gets
swif

temp
url.

It gen-er-ates a temp url for the swif base firm url to

the target file. Expecting url as swift://containername/objectname.

Paramet

par
Pars url ob-ject.

Raises

Swi Op-er-a-tionl on fail-ure to get url from swif

`ironic.`

Ver-i-fies the firm up-date ar-gu-men

ironic.drivers.modules.ilo.inspect module

iLO Inspect Interface

class i

Base

iro

dri

bas

Ins

get_pro

Return the properties of the interface

Returns

dictionary of <property name: description> entries

inspect

Inspect hardware to get the

hard-
ware
prop-
er-
ties.

In-
spec
hard-
ware
to
get
the
es-
sen-
tial
and
ad-
di-
tion

hardware properties. It fails if any of the essential properties are not received from the node. It doesnt fail if node fails to return any capabilities as the capabilities differ from hardware to hardware mostly.

Parame

`tas`
a
Task
agen
in-
stan

Raises

Hard-
ware
spec
tion-
Fail-
ure
if
es-
sen-
tial
prop-
er-
ties

could not be retrieved successfully.

Raises

Ilo-
Op-
er-
a-
tion

if system fails to get pow state

Returns
The resulting state of inspection.

validat
Che that driv contains required ILO credentials.

Validates whe the driv property of the supplied task node

contains the required credentials information.

Paramet
tas
a task

fron
Task
agen

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
re-
quir
iLO
pa-
ram-

eters are not valid.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing.

ironic.drivers.modules.ilo.management module

iLO
Man
age-
men
In-
ter-
face

class i

Base
irc
dri

moc ilc man Ilc

erase_c

Eras all the driv on the node

This meth perform out-of-banc sanitize disk erase on all the supported physical drives in the node. This erase cannot be performed on logical drives.

Paramet

tas a Task ager instanc

Raises

Invalic Parameter-Valu if any of the arguments are invalid.

Raises
IloError on an error from iLO

`one_but`
Eras the who system secure

The One butt secure eras process resets iLO and dele all licenses stored there, resets BIOS settings, and deletes all Active Health System (AHS) and warranty data stored on the system. It also erases supported non-volatile storage data and deletes any deployment setting profiles.

Paramet
`tas` a Task ager instanc

Raises
IloError on an

error from iLO

class i

Base
iro
dri
bas
Man

activat

Activate iLO Advanc license

Paramet

tas a Task agen object.

Raises

Invalid Parameter-Valu if any of the arguments are invalid.

Raises

Nod Clearing-Failure,

on failure to execute of

clean step.

clear_i

Unset iSCS details of the system in UEF boot mod

Paramet

tas

a task from Task agen

Raises

Ilo-Com man Not-Supported-In-Bios ror if sys-

tem in BIOS boot mode.

Raises

IloError

on an error from iLO

clear_s

Clea all secure boot keys

Clea all the secure boot keys This operation is sup-

ported only on HP Proliant Gen9 and above servers.

Parame

tas

a task from Task agen

Raises

Nod Clea ing-Fail-ure, on fail-ure to ex-e-cute of clean step.

Raises
Instan ploy Failure, on failure to execute of

deploy step.

`flash_f`
Deploy step to Update the firm using Sma Update

Manager (SUM).

Paramet
`tas` a Task ager object.

Raises
Instan ploy Failure, on failure to ex-

e- cute of

deploy step.

Returns

state to signify the step will be com- plete asyn

get_boo

Get the current boot device for a node

Returns the current boot device of the node

Parame

tas

a task from Task agen

Raises

Miss ing- Pa-

ram-
e-
ter-
Valu
if
a
re-
quir
iLO
pa-

rameter is missing.

Raises

Ilo-
Op-
er-
a-
tion
on
an
er-
ror
from
Ilo-
Clie
li-

brary.

Returns

a
dic-
tio-
nary
con-
tain-
ing:

boot_d

the
boot
de-
vice
one
of
the
sup-
port
de-
vice
liste
in

ironic.common.boot_devices or None if it is unknown.

persist

Whe the boot device will persist to all future boot

or not, None if it is unknown.

get_boo

Get the current boot mod for a node

Provide the current boot mod of the node

Paramet

tas

A task from Task ager

Raises

IloOperation

on an error from Ilo- Clie li- brary.

Returns
The boot mod one of `iro com boo` or Non if it is unknown.

get_pro
Return the properties of the interface

Returns
dictionary of <pro erty nam description> entries

`get_sec`

Get the current secure boot state for the node

Paramet

`tas` A task from Task ager

Raises

Missing-Parameter-Valu if a requir parameter is missing

Raises

Ilo-Oper-ation on an error from Ilo-Clie library.

Raises

Un-
sup-
port-
ed-
Driv
ten-
sion
if
se-
cure
boot
is
not

supported by the hardware

Returns

Boo

get_ser

Get
sen-
sors
data

Paramet

tas

a
Task
ager
in-
stanc

Raises

Fail
To-
Get-
Sen-
sor-
Data
whe
get-
ting
the
sen-
sor
data

fails.

Raises

Fail
ToPa
eSen
sor-

Data whe pars ing sensor data fails

Raises
Invalic ParameterValu if requir ipmi parameters are missing.

Raises
Miss ingParameterValu if a requir parameter is missing.

Returns
return a dict of sensor data grou by sen-

sor type

get_sup

Get a list of the supported boot devices

Parame

tas

a task from Task agen

Returns

A list with the supported boot devices defined in *iro*

common.boot_devices.

get_sup

Get a list of the supported boot devices

Parame

tas

a

task from Task agen

Raises

Ilo- Op- er- a- tionl if any ex- cep- tion hap- pens in

proliantutils

Returns

A list with the sup- porte boot de- vice de- fined in *iro*

common.boot_devices.

inject_

In- ject NMI Non Mas able In- ter- rupt

In- ject NMI (Non Mas

able Interrupt for a node immediately.

Paramet

`tas` A Task ager instance containing the node to act on.

Raises

IloCom man NotSupportedError if system does not support NMI injection.

Raises

IloError on an error from iLO.

Returns
Non

reset_b
Resets the BIO settings to default values.

Resets BIO to default settings. This operation is

currently supported only on HP Proliant Gen9 and above servers.

Parame
tas
a task from Task agen

Raises
Nod Cleaning-Failure, on failure to execute

of

clean step.

Raises
In-
stan
ploy
Fail-
ure,
on
fail-
ure
to
ex-
e-
cute
of

deploy step.

reset_i

Re-
sets
the
iLO

Paramet
tas
a
task
from
Task
agen

Raises
Nod
Clea
ing-
Fail-
ure,
on
fail-
ure
to
ex-
e-
cute
of

clean step.

Raises
In-
stan
ploy
Fail-

ure, on failure to execute of

deploy step.

reset_i

Resets the iLO pass word

Parame

- **tas** a task from Task agen
- **cha** Valu for pass word to update on iLO

Raises

Nod Clea ing-Fail ure, on failure to execute of

clean step.

Raises

In- stan ploy Failure, on failure to execute of

deploy step.

reset_s

Reset secure boot keys to man ufacturing de-

faults.

Resets the secure boot keys to man ufacturing de-

faults. This operation is supported only on HP Proliant Gen9 and above servers.

Paramet

tas

a task from Task agen

Raises

Nod Clea ing-Fail-ure, on fail-ure to ex-e-cute of

clean step.

Raises

In-stan ploy Fail-ure, on fail-ure to ex-e-cute of

deploy step.

set_boo

Set the boot de-vice for a node

Set the boot device to use on next reboot of the node

Paramet

- **tas** a task from Task agen
- **dev** the boot device one of the support device liste in `ironic.common.boot_devices`.
- **per** Boo valu True if the boot device will persist to all future boots, False if not. Default: False.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
an
in-
valic
boot
de-

vice is specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing.

Raises
Ilo-
Op-
er-
a-
tionl
on
an
er-
ror
from
Ilo-
Clie
li-

brary.

set_boo
Set
the
boot
mod

for a node

Set the boot mode to use on next reboot of the node

Parame

- **tas** A task from Task agen
- **mod** The boot mode one of *irc com boc*

Raises

Invalid Parameter Value if an invalid boot mod is specified.

Raises
IloOperation if setting boot mod faile

set_isc
Set iSCS details of the system in UEF boot mod

The initiator is set with the target details like IQN LUN, IP, Port etc. :param task: a task from TaskManager. :raises: MissingParameterValue if a required parameter is missing. :raises: IloCommandNotSupportedInBiosError if system in BIOS boot mode. :raises: IloError on an error from iLO.

set_sec
Set the current se-

cure boot state for the node

Parame

- **`tas`** A task from Task ager
- **`sta`** A new state as a bool

Raises
Missing-Param-e-ter-Valu if a re-quire pa-ram-

eter is missing

Raises
Ilo-Op-er-a-tion on an er-ror from

Ilo-
Clie
li-

brary.

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion
if
se-
cure
boot
is
not

supported by the hardware

update_
Up-
date
the
firm

Parame
tas
a
Task
ager
ob-
ject.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
up-
date
firm
mod
is

not ilo. Even applicable for invalid input cases.

Raises
Nod
Clea

ing-Fail-ure, on fail-ure to ex-e-cute of clean step.

Raises
In-stan ploy Fail-ure, on fail-ure to ex-e-cute of deploy step.

update_
Clea step to up-date the firm us-ing Sma Up-date Man ager (SUM)

Parame
tas
a Task ager ob-ject.

Raises

Nod Clea ing- Fail- ure, on fail- ure to ex- e- cute of

clean step.

Returns

state to sig- nify the step will be com plete asyn

validat

Che that driv con- tains re- quir ILO cre- den- tials

Val- i- date whe the driv prop erty of the sup-

plied
task
node

contains the required credentials information.

Parame

`tas`
a
task
from
Task
agen

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
re-
quir
iLO
pa-
ram-

eters are not valid.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing.

ironic.drivers.modules.ilo.power module

iLO
Pow
Driv

class i
Base
iro
dri
bas
Pow

get_pow
Gets
the
current
pow
state

Parame

- **tas**
a
Task
ager
instan
- **nod**
The
Nod

Returns
one
of
iro
com
sta
POW
POW
or
ER-
ROR

Raises
Invalid

Parameter-Valu if required iLO credentials are missing.

Raises

IloOperation on an error from IloClie library.

get_pro

Return the properties of the interface

Returns

dictionary of <property name: description:

entries

get_sup

Get a list of the supported power state

Parame

tas

A Task agen instance containing the node to act

on. currently not used.

Returns

A list with the supported power state defined in *iro* *com*

states.

reboot

Reboot the node

Parame

- **tas** a Task ager instan
- **tim** time out (in seconds Unsupport by this interface.

Raises
Power State Failure if the final state of the node is not POWER_ON.

Raises
Ilo Operation on an error from Ilo Clie

library.

set_pow

Turn the current power state on or off.

Parame

- **tas** a Task agent instance
- **pow** The desired power state POW or REBOO from *ir* *com* *states*.
- **tim** timeout (in seconds) Unsupported by this in-

ter-

face.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
an
in-
valic
pow
state

was specified.

Raises
Ilo-
Op-
er-
a-
tionl
on
an
er-
ror
from
Ilo-
Clie
li-

brary.

Raises
Pow
er-
State
Fail-
ure
if
the
pow
coul
be
set
to
pow

validat
Chec
if

node contains the required iLO credentials

Paramet

- **tas** a Task ager instan

- **nod** Single node object.

Raises
Invalid Param- e- ter- Valu if required iLO credentials are missing.

ironic.drivers.modules.ilo.raid module

iLO
RAI
spe-
cific
meth
ods

class i
Base
irc
dri
bas
RAI

Im-
ple-
men
ta-
tion
of
OOI
RAI
In-
ter-
face
for
iLO

apply_c

Ap-
plies
RAI
con-
fig-
u-
ra-
tion
on
the
give
node

Parame

- **tas**
  A
  Task

agen
instanc

- **rai**
  The RAI configuration to apply.

- **cre**
  Setting this to Fals indicate not to create

root volume that is specified in raid_config. Default value is True.

- **cre**
  Setting this to Fals indicate not to create

non-root volumes (all except the root volume) in raid_config. Default value is True.

- **del**
  Setting

this to True indicate to dele RAI configuration prior to creating the new configuration.

Raises
Invali ParameterValu if the RAI configuration is invalid.

Returns
state if RAI configuration is in prog asyn chro or None if it is complete.

create_

Create a RAI configuration on a bare meta

using agent ramdisk.

This meth creates a RAI configuration on the give

node.

Parame

- **tas** a Task agen instan
- **cre** If True a root volume is created during RAI

configuration. Otherwise, no root volume is created. Default is True.

- **cre** If

True non-root volumes are created. If Fals no

non-root volumes are created. Default is True.

Raises
Miss ing-Parameter-Valu if node is miss ing or

was found to be empty after skipping root volume and/or non-root volumes.

Raises
Nod Clea ing-Failure, on failure to execute clea

step.

Raises
Instan ploy Failure, on fail-

ure to execute deploy step.

delete_

Dele the RAI configuration.

Paramet

tas a Task ager instan containing the node to act on.

Raises

Nod Clea ing-Failure, on failure to execute clea step.

Raises

Instan

ploy Fail- ure, on fail- ure to ex- e- cute de- ploy step.

get_pro

Re- turn the prop er- ties of the in- ter- face

ironic.drivers.modules.ilo.vendor module

Ven- dor In- ter- face for iLO driv and its sup- port- ing meth ods.

class i

Base *iro* *dri* *bas* *Ven*

Vend spec interface for iLO deploy drive

boot_ir

Attach an ISO image in glan and reboot bare meta

This meth accept an ISO image href (a Glan UUI or an HTTP(S) URL) attaches it as virtual media and then reboots the node. This is useful for debugging purposes. This can be invoked only when the node is in manage state.

Paramet

- **tas** A Task agen ob-

ject.

- **kwa** The argumen sent with vendor pass The expect

kwargs are:

```
↪can be
```

get_pro

Return the properties of the

interface

Returns
dictionary of <pro erty nam description entries

validat
Validate vend spec actions

Chec if a valid vendor pass meth was pass and validate

the parameters for the vendor passthru method.

Paramet

- **tas** a Task agen instance containing the node to act on.

- **met** meth to be validated

- **kwa** kwa containing the vendor pass methods parameters.

Raises
Missing-Parameter-Valu if som required parameters were not passed.

Raises
Invalid Pa-

ram-e-ter-Valu if any of the pa-ram-

eters have invalid value.

Module contents

ironic.drivers.modules.intel_ipmi package

Submodules

ironic.drivers.modules.intel_ipmi.management module

In-tel IPM Hard-ware

Sup-port In-tel Spee Se-lect Per-for-man Pro-file.

class i

Base
iro dri mod ipm IPM

configu

Module contents

ironic.drivers.modules.irmc package

Submodules

ironic.drivers.modules.irmc.bios module

iRM BIO configuration specific methods

class i

Base

iro dri bas BIO

apply_c

Applies BIO configuration on the given node

This method takes the BIO

setting from the setting para and applies BIOS configuration on the given node. After the BIOS configuration is done, self.cache_bios_settings() may be called to sync the nodes BIOS-related information with the BIOS configuration applied on the node. It will also validate the given settings before applying any settings and manage failures when setting an invalid BIOS config. In the case of needing password to update the BIOS config, it will be taken from the driver_info properties.

Paramet

- `tas` a Task ager instan
- `set` Dictionary containing the BIO configuration. It may be an empty dictionary as well.

Raises
IRM C-Operation apply bios set-

tings
faile

`cache_b`

Stor or up- date BIO set- tings on the give node

This meth store BIO prop er- ties to the bios set- tings db

Parame

`tas` a Task agen in- stan

Raises

IRM C- Op- er- a- tionl get bios set- tings faile

Returns

Non if

it is com plete

factory

Re-set BIO con-fig-u-ra-tion to fac-tory de-fault

on the given node.

Parame

tas

a Task agen in-stan

Raises

Un-sup-port-ed-Driv ten-sion if the node drive does sup-

port BIOS reset.

get_pro

Re-turn the prop er-ties of

the interface

validat

Validate the driv spec Nod info

This meth validate whe the driv property of the supplied

node contains the required information for this driver to manage the BIOS settings of the node.

Paramet

tas a Task ager instance containing the node to act

on.

Raises

Invalid ParameterValu if requir driv attribu

is missing or invalid on the node.

Raises
Miss ingParameterValu if a requir parameter is missing in the driver_info property.

ironic.drivers.modules.irmc.boot module

iRM Boo Driv

class i

Base *iro dri moo pxe PXE*

iRM PXE boot

prepare

Prepare the boot of

Iron ram us- ing PXE

This meth pre- pare the boot of the de- ploy ker- nel/r af- ter

reading relevant information from the nodes driver_info and instance_info.

Paramet

- **tas** a task from Task agen
- **ram** the pa- ram- e- ters to be pass to the ramd pxe

driver passes these parameters as kernel command-line arguments.

Returns
Non

Raises
Miss

ing-Parameter-Valu if som information

is missing in nodes driver_info or instance_info.

Raises
In-valic Parameter-Valu if som information

provided is invalid.

Raises
IronicEx ception if som pow or set boot device op-

eration failed on the node.

class i

Base
irc
dri
bas
Boc

iro dri moc irm boc IRM

iRM Virtual Media boot relat actions

capabil

clean_u

Clea up the boot of instanc

This meth clea up the environmen that was setu for boot

ing the instance.

Paramet

tas a task from Task ager

Returns
Non

Raises
IRMC-OperationE if iRM operation faile

clean_u
Clea up the boot of ironi ramd

This meth clea up the environmen that was setu for boot

ing the deploy or rescue ramdisk.

Paramet
tas
a task from Task agen

Returns
Non

Raises

IRMC-Operation if iRMC operation faile

get_pro

Return the proper-ties of the interface

Returns

dictionary of <pro erty nam description entries

prepare

Prepare the boot of instan

This meth

prepare the boot of the instanc after reading

relevant information from the nodes database.

Paramet

`tas` a task from Task agen

Returns Non

`prepare`

Prepare the deploy or rescue rame using virtual

media.

Prepare the options for the deploy or rescue the node to boot from virtual media cdrom.

on.

vice can handle specified href.

ram
sets

Paramet

- **tas** a Task ager instanc containing the node to act
- **ram** the options to be pass to the ram

Raises
ImageF f-Validation-Fail if no image ser-

Raises
Image-

Cre-
atio
Fail
if
it
faile
whil
cre-
at-
ing
the

floppy image.

Raises
In-
vali
Pa-
ram-
e-
ter-
Valu
if
the
val-
i-
da-
tion

of the PowerInterface or ManagementInterface fails.

Raises
IRM
C-
Op-
er-
a-
tion
if
som
op-
er-
a-
tion
on

iRMC fails.

validat

Val-
i-
date
the
de-
ploy
men

information for the

tasks node.

Parame

`tas`

a Task ager instanc containing the node to act

on.

Raises

Invalid Parameter-Valu if config option has

invalid value.

Raises

IRMC-Shar FileS tem-Not-Mou if shar file system

is

not mounted.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
som
in-
for-
ma-
tion

is invalid.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
ker-
nel_
and
ramc
are

missing in the Glance image, or if kernel and ramdisk are missing in the Non Glance image.

validat
Val-
i-
date
that
the
node
has
re-
quire
prop
er-
ties
for

rescue.

Paramet
tas

a Task ager in- stand with the node be- ing chec

Raises

Miss ing- Pa- ram- e- ter- Valu if node is miss ing one

or more required parameters

Raises

In- valic Pa- ram- e- ter- Valu if any of the pa- ram-

eters have invalid value.

class i

Base obj

Mix in class for vol-

ume
boot
configuration
to
iRM

iRM
has
a
feature
to
set
up
remote
boot
to
a
serv

This feature can be used by VIOM (Virtual I/O Manager) library of SCCI client.

`ironic.`

At-
tach
boot
ISO
for
a
deploy
node
if
it
exists.

This
meth
chec
the
instanc
info
of
the
bare
meta
node

for a

boot ISO. If the instance info has a value of key irmc_boot_iso, it indicates that boot_option is netboot. Threfore it attaches the boot ISO on the bare metal node and then sets the node to boot from virtual media cdrom.

Paramet

tas

a Task ager instan containing the node to act

on.

Raises

IRMC-Operation if attaching virtual me-

dia failed.

Raises

Invalid Parameter-Valu if the validation

of the ManagementInterface fails.

`ironic.`

Chec if Shar File System (NF or CIF is mou

Raises
In-valic Param-e-ter-Valu if config option has

invalid value.

Raises
IRM C-Shar FileS tem-Not-Mou if shar file system is

not mounted.

ironic.drivers.modules.irmc.common module

Common functionalities shar betwee different iRM modules.

`ironic.`

Gets an iRM SCC clien

Give an iron node object, this meth give back a iRM SCC clien to do operations on the iRMC.

Paramet

nod An iron node object.

Returns

scci_ partial

func tion whic take a SCC com man para

Raises
In- valid Pa- ram- e- ter- Valu on in- valid in- puts

Raises
Miss ing- Pa- ram- e- ter- Valu if som man tory in- for-

mation is missing on the node

Raises
IRM C- Op- er- a- tionl if iRM op- er- a- tion

faile

ironic.

Gets iRM SCC re- port

Give an iron node ob- ject, this meth give back a iRM SCC re-

port.

Paramet

nod

An iron node ob- ject.

Returns

A xml. ob- ject.

Raises

In- valid Pa- ram- e- ter- Valu on in- valid in- puts

Raises

Miss-
ing-
Pa-
ram-
e-
ter-
Valu
if
som
man
tory
in-
for-

mation is missing on the node.

Raises
scci.
if
re-
quir
pa-
ram-
e-
ters
are
in-
valid

Raises
scci.
if
SCC
faile

`ironic.`
Get
the
cur-
rent
se-
cure
boot
mod

Paramet
nod
An
ironi
node
ob-
ject.

Raises
Un-

supported Driv tension if secure boot is not

present.

Raises
IRM C-Operation if the operation fails

`ironic.`

Gets the specific Nod drive info

This meth validate whe the drive property of the supplied

node contains the required information for this driver.

Paramet

nod

An ironi node object.

Returns

A dict containing information from drive and default values.

Raises

InvalidParameterValu if invalid valu is contained in the driver_info property.

Raises

MissingParameterValu if som mandatory key

is

missing in the driver_info property.

`ironic.`

En-
able
or
dis-
able
UEF
Se-
cure
Boo

Paramet

- `nod`
  An
  ironi
  node
  ob-
  ject.
- `ena`
  Boo
  valu
  True
  if
  the
  se-
  cure
  boot
  to
  be
  en-
  able

Raises
IRM
C-
Op-
er-
a-
tionI
if
the
op-
er-
a-
tion

fails

ironic.

Updated ipmi properties to node driv

Paramet

tas

A task from Task agen

ironic.drivers.modules.irmc.inspect module

iRM Inspect Interface

class i

Base

iro

dri

bas

Ins

Interface for out of band inspection.

get_pro

Re-

turn the prop er- ties of the in- ter- face

Returns

dic- tio- nary of <pro erty nam de- scrip tion: en- tries

inspect

In- spec hard ware

In- spec hard ware to ob- tain the es- sen- tial hard ware prop

erties and mac addresses.

Paramet

tas

a task from Task

ager

Raises

Hard ware spec tion- Fail- ure, if hard ware in- spec tion faile

Returns

state if hard ware in- spec tion suc- ceed

validat

Val- i- date the drive spec in- spec tion in- for- ma- tion.

This meth val- i- date whe the drive prop erty of the supplie

node contains the required information for this driver.

Parame

tas a Task ager instan containing the node to act

on.

Raises

Invali Parameter Valu if requir drive attribu

is missing or invalid on the node.

Raises

Missing Parameter Valu if a requir param-

eter is missing.

`ironic.`

SC2 sc2U return NIC type

sc2UnitN

SYN TAX INTEGE { unknow primar seconda man

blade(4), secondary-remote(5), secondary-remote-backup(6), baseboard-controller(7) } ACCESS read-only STATUS mandatory DESCRIPTION Management node class: primary: local operating system interface secondary: local management controller LAN interface management-blade: management blade interface (in a blade server chassis) secondary-remote: remote management controller (in an RSB concentrator environment) secondary-remote-backup: backup remote management controller baseboard-controller: local baseboard management controller (BMC) ::= { sc2ManagementNodes 8 }

`ironic.`

SC2 sc2U return NIC MA addres

sc2UnitN

SYN TAX Phys dres ACCES read only STATU man tory DE-

SCRIPTION Management node hardware (MAC) address ::= { sc2ManagementNodes 9 }

ironic.drivers.modules.irmc.management module

iRM
Man
agemen
Driv

class i

Base
irc
dri
moc
ipm
IPM

get_prc

Return the prop erties of the interface

Returns

Dictionary of <prc erty nam descrip tion entries

get_sec

Get the current

secure
boot
state
for
the
node

NOTE:
may
not
implement
that.

Parame
`tas`
A
task
from
Task
agen

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing

Raises
Driv
Op-
er-
a-
tion
or
its
deriv
tive
in
case
of

driv

runtime error.

Raises

Unsupport-ed-Driv ten-sion if secure boot is not

supported by the driver or the hardware

Returns

Boo

get_ser

Get sensors data meth

It gets sensor data from the task node via SCC and convert

the data from XML to the dict format.

Parame

tas

A Task agen instanc

Raises

Faile
To-
Get-
Sen-
sor-
Data
whe
get-
ting
the
sen-
sor
data

fails.

Raises
Faile
ToPa
eSen
sor-
Data
whe
pars
ing
sen-
sor
data
fails

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
re-
quire
pa-
ram-
e-

ters are invalid.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu

if a requir param-eter is missing.

Returns

Return a consistent formatted dict of sensor data grouped by sensor type, which can be processed by Ceilometer. Example:

(continues on next page)

(continued from previous page)

```
↪',
```

(continues on next page)

↪

(continued from previous page)

```
↪',
```

(continues on next page)

(continued from previous page)

```
↪

↪',
```

(continues on next page)

(continued from previous page)

```
↪



















↪',
```

inject_

Inject NMI Non Mas able Interrupt

Inject NMI (Nor Mas able Interrupt) for a node immediately.

Paramet

`tas` A Task ager instance containing the node to act on.

Raises

IRM C-Operation on an error

from
SCO

Returns
Non

restore
Re-
store
BIO
con-
fig
for
a
node

Paramet
tas
a
task
from
Task
agen

Raises
Nod
Clea
ing-
Fail-
ure,
on
fail-
ure
to
ex-
e-
cute
step

Returns
Non

set_boo
Set
the
boot
de-
vice
for
a
node

Set
the
boot

device to use on next reboot of the node

Parame

- **tas** A task from Task agen
- **dev** The boot device, one of the supported devices liste in *ironic.common.boot_devices*.
- **per** Boo valu True if the boot device will persist to

all future boots, False if not. Default: False.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
an
in-
valic
boot
de-

vice is specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing.

Raises
IP-
MI-
Fail-
ure
on
an
er-
ror
from
ip-
mi-
tool.

set_sec
Set
the
cur-
rent

secure boot state for the node

NOTE: may not implement that.

Parame

- **tas** A task from Task agen
- **sta** A new state as a bool

Raises
MissingParameterValu if a required param-

eter is missing

Raises
Driv

Operation or its derivative in case of driv

runtime error.

Raises

Unsupported- Driv tension if secure boot is not

supported by the driver or the hardware

validat

Validate the drive spec management information.

This meth validate whe the

driv
prop
erty
of
the
sup-
plie

node contains the required information for this driver.

Paramet

`tas`

A
Tas
ager
in-
stan
con-
tain-
ing
the
node
to
act

on.

Raises

In-
vali
Pa-
ram-
e-
ter-
Valu
if
re-
quir
pa-
ram-
e-

ters are invalid.

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir

pa- ram-

eter is missing.

`ironic.`

Bac BIO con- fig from a node

Paramet

`tas`

a Task agen in- stanc con- tain- ing the node to act

on.

Raises

IRM C- Op- er- a- tion on fail- ure.

ironic.drivers.modules.irmc.packaging_version module

`excepti`

Base `Val`

Rais whe a ver- sion strin

is not a valid version

`ironic.`

A strin containing the regular expression used to match a valid version.

The pattern

is not anchor at either end, and is intended for embedding in larger expressions (for example, matching a version number as part of a file name). The regular expression should be compiled with the `re.VERBOSE` and `re.IGNORECASE` flags set.

class i

Base
`iro`
`dri`
`mod`
`irm`
`pac`
`_Ba`

This
class
ab-
strac
han-
dlin
of
a
proj
ver-
sion

A
Ver
in-
stan
is
com
par-
i-
son
awa
and
can
be
com

pared and sorted using the standard Python interfaces.

(continues on next page)

(continued from previous page)

propert

The base version of the version

(continues on next page)

(continued from previous page)

The base version is the public version of the proj with

out any pre or post release markers.

propert

The developmen num ber of the version

`propert`

The epoc of the version

(continues on next page)

(continued from previous page)

propert

Whe this version is a developmen releas

propert

Whe this version is a post relea

propert

Whe this version is a pre-relea

(continues on next page)

(continued from previous page)

propert

The local version segmen of the version

(continues on next page)

(continued from previous page)

propert

The first item of *rel* or `0` if un- avai able

propert

The third item of *rel* or `0` if un- avai able

propert

The second item of *rel* or `0` if unavai able

propert

The post relea num ber of the version

propert

The pre-relea segmen of the version

(continues on next page)

(continued from previous page)

propert

The public portion of the

version

propert

The component of the

release segmen of the version.

Includ trailing zeroes but not the epoc or any pre-relea / development / post-release suffixes.

`ironic.`

Pars the give version strin

Paramet

`ver` The version strin to pars

Raises

`Inv` Whe

the version string is not a valid version

ironic.drivers.modules.irmc.power module

iRM Pow Driv using the Base Serv Profile

class i
Base
iro dri bas Pow

Interface for pow relat actions

get_pow
Return the pow state of the task node

Paramet

tas

a Task ager in- stanc con- tain- ing the node to act

on.

Returns

a pow state One- of *iro com sta*

Raises

In- valid Pa- ram- e- ter- Valu if re- quire ipmi pa- ram-

eters are missing.

Raises

Miss ing- Pa- ram- e- ter- Valu if a re-

quir-
pa-
ram-

eter is missing.

Raises
IPMIFailure on an error from ipmitool (from

_power_status call).

get_pro
Return the prop-
erties of the interface

Returns
dictionary of <pro-
erty nam
description
entries

get_sup
Get a list of

the sup- port pow state

Parame

tas

A Task ager in- stanc con- tain- ing the node to act

on. currently not used.

Returns

A list with the sup- port pow state de- fined in *irc com*

states.

reboot

Per- form a hard re- boot of the task node

Parame

-

tas a Task agen in- stan con- tain- ing the node to act

on.

- **tim** time out (in sec- onds pos- i- tive in- te- ger (>

0) for any power state. `None` indicates default timeout.

Raises
In- valic Pa- ram- e- ter- Valu if an in- valic pow state

was specified.

Raises
IRM C- Op- er- a-

tionl if faile to set the pow state

set_pow Set the pow state of the task node

Paramet

- **tas** a Task agen in-stanc con-tain-ing the node to act

on.

- **pow** Any pow state from *irc com sta*
- **tim** time out (in sec-

onds positive integer (> 0) for any power state. `None` indicates default timeout.

Raises
Invali ParameterValu if an invali pow state was specified.

Raises
MissingParameterValu if som man tory information is missing on the node

Raises
IRM COperationE if faile to set

the pow state

validat

Val- i- date the driv spec Nod pow info

This meth val- i- date whe the driv prop- erty of the sup- plied node contains the required information for this driver to manage the power state of the node.

Parame

tas a Task agen in- stan con- tain- ing the node to act on.

Raises

In- valid Pa- ram- e-

ter-
Valu
if
re-
quir
driv
at-
tribu

is missing or invalid on the node.

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing.

`ironic.`

SC2
sc2s
re-
turn
sta-
tus
of
the
cur-
rent
boot

ironic.drivers.modules.irmc.raid module

Irmc
RAI
spe-
cific
meth
ods

class i

Base
irc

dri
bas
RAI

create_

Create the RAI configuration.

This meth creates the RAI configuration on the give node.

Paramet

- **tas** a Task ager instanc containing the node to act on.
- **cre**

If True a root volume is created during RAI

configuration. Otherwise, no root volume is created. Default is True.

- **cre**

  If True non-root volumes are created. If Fals no

non-root volumes are created. Default is True.

Returns

state if RAI configuration is in prog asyn chro

Raises

Missing-Parameter-Valu

if node is missing or empty.

Raises
IRMC-OperationError on an error from scci-client

delete_
Delete the RAID configuration.

Paramet
tas a TaskManager instance containing the node to act on.

Returns
states if

dele-
tion
is
in
prog
asyn
chro
or
Non
if
it
is complete.

get_pro
Re-
turn
the
prop
er-
ties
of
the
in-
ter-
face

Module contents

ironic.drivers.modules.network package

Submodules

ironic.drivers.modules.network.common module

class i
Base
iro
dri
mod
net
com
VIF

VIF
port
ID
mixi
class
for
neu-

tron network interface

Mix class that provides VIF-relat network interface methods

for neutron network interfaces. On VIF attach/detach, the associated neutron port will be updated.

get_noc

Get network configuration data for node ports

Pull network data from ironi node object if pres otherwise

collect it for Neutron VIFs.

Paramet

tas

A Task ager in- stanc

Raises

In- valid Pa- ram- e- ter- Valu if the net- work in- ter-

face configuration is invalid.

Raises

Miss ing- Pa- ram- e- ter- Valu if som pa- ram- e- ters

are missing.

Returns

a dict hold ing net- work con- fig- u- ra- tion in-

for-

mation adhearing Nova network metadata layout (*network_data.json*).

port_ch

Han- dle any ac- tions re- quir whe a port char

Parame

- **tas** a Task ager in- stan
- **por** a char Port ob- ject from the API be- fore it is

saved to database.

Raises

Faile ToU dat- eD- HCF tOn- Port Con- flict

portgro

Handle any actions requir whe a port- grou char

Paramet

- **tas** a Task agen instan
- **por** a char Port grou object from the API before it

is saved to database.

Raises

Faile ToU dat- eD- HCE tOn- Port Con flict

vif_att

Attach a virtual networl interface to a node

Attach a virtual interface to a node Whe selecting a port or portgroup to attach the virtual interface to, the following ordered criteria are applied:

- Requir port or portgrou to have a physical network that is either None or one of the VIFs allowed physical networks.
- Prefer port or portgrou with

a phys i- cal net- work

field which is not None.

- Pre- fer port- grou to port

- Pre- fer port with PXE en- able

Paramet

- **tas** A Task agen in- stan

- **vif** a dic- tio- nary of in- for- ma- tion abou a VIF.

It must have an id key, whose value is a unique identifier for that VIF.

Raises
Net-

wor

Error,

VifAlread

Attach

NoF

hysical-

Ports

Raises

Port

grou

Phys

net-

Inconsistent if one of the node

port-

groups has ports which are not all assigned the same physical network.

vif_det

Detach a virtual network interface from a node

Paramet

- **tas** A Task

ager

in-
stan

- **vif** A VIF ID to de-
tach

Raises
VifN tAt-
tach if VIF not at-
tach

Raises
Net-
worl Er-
ror if un-
bind Neu tron port faile

class i
Base obj

VIF port ID mixi class for non- neut net-
worl in-
ter-
face

Mix

class that provide VIF relat network interface methods

for non-neutron network interfaces. There are no effects due to VIF attach/detach that are external to ironic.

NOT This does not yet support the full set of VIF methods,

as it does not provide vif_attach, vif_detach, port_changed, or portgroup_changed.

get_cu

Returns the currentl used VIF associated with port

or portgroup

We are booting the

node only in one networ at a time

and presence of cleaning_vif_port_id means were doing cleaning, of provisioning_vif_port_id - provisioning, of rescuing_vif_port_id - rescuing. Otherwise its a tenant network

Paramet

- `tas` A Task ager instanc
- `p_o` Iron port or portgrou object.

Returns
VIF ID associated with p_ol or Non

`vif_lis`
List attach VIF IDs for a node

Paramet

tas

A Task ager in- stanc

Returns

List of VIF dictionaries, each dictionary will have

an id entry with the ID of the VIF.

`ironic.`

Find free portlike object (por grou or port VIF will be

attached to.

Ensure that the VIF is not al- read

attach to this nod

When selecting a port or portgroup to attach the virtual interface to, the following ordered criteria are applied:

- Requir port or portgrou to have a phys ical network that is either None or one of the VIFs allowed physical networks.
- Prefer port or portgrou with a phys ical network field which is not None.
- Prefer portgrou to port
- Prefer port with

PXE enable

Paramet

- **tas** a Task agen instan
- **vif** Nam or UUI of a VIF.
- **phy** Set of phys ical networ on whic the VIF may

be attached. This is governed by the segments of the VIFs network. An empty set indicates that the ports physical networks should be ignored.

- **vif** dict that may contain extra informa-

tion,
such

as port_uuid

Raises
Vi-
fAl-
read
At-
tach
if
VIF
is
al-
read
at-
tach
to

the node.

Raises
NoF
hys-
i-
cal-
Port
if
there
is
no
port-
like
ob-
ject

VIF can be attached to.

Raises
Port
grou
Phys
net-
Inco
sis-
tent
if
one
of
the
node
port-

groups has ports which are not all assigned the same physical network.

Returns

port-like object VIF will be attached to.

`ironic.`

Plug port-like object to tenant network.

Paramet

- **`tas`** A Task ager instanc
- **`por`** port-like object to plug
- **`cli`** Neutron clien instanc

Raises

Net-
worl
Er-
ror
if
faile
to
up-
date
Neu
tron
port

Raises

VifN
tAt-
tach
if
ten-
ant
VIF
is
not
as-
so-
ci-
ated

with port_like_obj.

ironic.drivers.modules.network.flat module

Flat
net-
worl
in-
ter-
face
Use-
ful
for
shar
flat
net-
worl

class i

Base
irc
dri
moc

net
com
Neu
irc
com
neu
Neu
irc
dri

base.NetworkInterface

Flat net- work in- ter- face

add_cle
Add the clea ing net- work to a node

Parame
tas
A Task agen in- stanc

Returns
a dic- tio- nary in the form {por neu- tron

Raises
Net- work Er-

ror, Invalid Parameter Valu

add_ins

Add the inspection network to the node

Parame

tas A Task ager instanc

Returns a dictionary in the form {por neutron

Raises NetworkError

Raises Invalid Parameter-

Valu if the network interface configuration is invalid.

add_pro

Add the provisioning network to a node

Paramet

tas

A Task manager instance

Raises

NetworkError when failed to set binding:l

add_res

Add the rescuing network to a node

Flat network does not use the rescuing network. Bind the

port again since unconfigure_tenant_network() unbound it.

Parame

`tas`

A Task ager instan

Returns

a dictionary in the form {por neutron

Raises

NetworkError, InvalidParameterValu

`configu`

Configure

tenant network for a node

Paramet

tas

A Task manager instance

remove_

Remove the cleaning network from a node

Paramet

tas

A Task manager instance

Raises

NetworkError

remove_

Remove the inspection network from a node

Paramet

`tas`

A Task ager instanc

Raises

Invalid Parameter Valu if the network interface configuration is invalid.

Raises

Missing Parameter Valu if som parameters are missing.

`remove_`

Remov the provisioning network from a node

Paramet

tas

A Task agen instanc

remove_

Removing the rescuing network from a node

Flat network does not use the rescuing network. Unbind the port again since add_rescuing_network() bound it.

Paramet

tas

A Task agen instanc

Raises

NetworkError

unconfi

Unconfigure tenant network for a node

Unbind the port here to avoid the possibility of the ironic port being bound to the tenant and cleaning networks at the same time.

Paramet

`tas`

A Task ager instan

Raises

Network Error

`validat`

Validate the network interface

Paramet

`tas`

a

Task agen instanc

Raises Invali ParameterValu if the network interface configuration is invalid.

Raises MissingParameterValu if som parameters are missing.

ironic.drivers.modules.network.neutron module

class i

Base *iro dri moc net com Neu iro com neu Neu*

iro
dri

base.NetworkInterface

Neu tron v2 network interface

add_cle

Create neutron ports for each port on task to boot the

ramdisk.

Parame

tas

a Task agen instanc

Raises

Network Error

Returns

a dictionary in the form {por neutron

add_ins

Add the in- spec tion net- work to the node

Paramet

tas

A Task agen in- stanc

Returns

a dic- tio- nary in the form {por neu- tron

Raises

Net- work Er- ror

Raises

In- valid Pa- ram- e- ter- Valu if the net- work in- ter-face configuration is invalid.

add_pro

Add the provisioning network to a node

Paramet

tas

A Task agen instan

Raises

NetworkError

add_res

Create neutron ports for each port to boot the rescue

ramdisk.

Paramet

tas

a Task agen instan

Returns

a dictionary in the form {por neutron

configu

Configure tenant network for a node

Paramet

tas

A Task agen instanc

Raises

Network Error

need_po

Chec if the node has any Sma NIC ports

Paramet

tas

A Task agen instanc

Returns
A
bool
to
in-
di-
cate
Sma
NIC
port
pres
ence

remove_
Dele
the
neu-
tron
port
cre-
ated
for
boot
ing
the
ram

Parame
tas
a
Task
ager
in-
stanc

Raises
Net-
worl
Er-
ror

remove_
Re-
mov
the
in-
spec
tion
net-
worl
from
a

node

Paramet

`tas`

A Task ager instanc

Raises

Invalic Parameter-Valu if the network inter-

face configuration is invalid.

Raises

Missing-Parameter-Valu if som parameters

are missing.

`remove_`

Remov the provisioning network from

a node

Paramet

`tas`

A Task ager instan

Raises

Network Error

`remove_`

Dele neutron port created for booting the rescue ram

Paramet

`tas`

a Task ager instan

Raises

Network Error

`unconfi`

Unconfigure tenant network for a node

Nov take care of port re- mov from ten- ant net- work we un-

bind it here/now to avoid the possibility of the ironic port being bound to the tenant and cleaning networks at the same time.

Parame

`tas`

A Task agen in- stan

Raises

Net- work Er- ror

validat

Val- i- date the net- work in- ter- face

Parame

`tas`

a Task agen in- stan

Raises

Invalic ParameterValu if the networl interface configuration is invalid.

Raises

Miss ingParameterValu if som parameters are missing.

`validat`

Validate the networl interface for rescue operation.

Paramet

`tas` a Task agen

in-
stand

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
the
net-
work
in-
ter-

face configuration is invalid.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
som
pa-
ram-
e-
ters

are missing.

ironic.drivers.modules.network.noop module

class i

Base
iro
dri
bas
Net

Noo
net-
work
in-
ter-
face

add_cle

Add the clea ing networl to a node

Paramet

tas A Task ager instanc

add_pro

Add the provision ing networl to a node

Paramet

tas A Task ager instanc

configu

Configure tenant networl for a node

Paramet

tas

A Task agen in- stan

get_cu

Re- turn the cur- rent used VIF as- so- ci- ated with port

or portgroup

We are boot ing the node only in one net- work at a time

and presence of cleaning_vif_port_id means were doing cleaning, of provisioning_vif_port_id - provisioning of rescuing_vif_port_id - rescuing. Otherwise its a tenant network

Parame

- **tas** A Task agen in- stan
- **p_c** Iron port

or port-grou object.

Returns
VIF ID associated with p_ob or Non

port_ch
Handle any actions required whe a port char

Paramet

- **tas** a Task agen instanc
- **por** a char Port object.

Raises
Conflict, Faile

ToU dat- eD- HCI tOn- Port

portgro

Han dle any ac- tions re- quir whe a port- grou char

Paramet

- **tas** a Task agen in- stan
- **por** a char Port grou ob- ject.

Raises

Con flict, Faile ToU dat- eD- HCI tOn- Port

remove_

Re-

mov the clean ing net- work from a node

Parame

tas

A Task agen in- stanc

remove_

Re- mov the pro- vi- sion ing net- work from a node

Parame

tas

A Task agen in- stanc

unconfi

Un- con- fig- ure ten- ant net- work for a node

Parame

tas
A Task ager in- stan

validat
Val- i- date that the node has re- quire prop- er- ties for

inspection.

Paramet
tas
A Task ager in- stan with the node be- ing chec

vif_att
At- tach a vir- tual net- work in- ter- face to a node

Paramet

- **tas** A Task ager in- stanc
- **vif** a dic- tio- nary of in- for- ma- tion abou a VIF.

It must have an id key, whose value is a unique identifier for that VIF.

Raises
Net- work Er- ror, Vi- fAl- read At- tach NoP hys- i- cal-

Ports

vif_det
De- tach a vir- tual net- work in- ter- face

from a node

Paramet

- **`tas`** A Task agen instan
- **`vif`** A VIF ID to detach

Raises
Networl Error, VifN tAttach

`vif_lis`
List attach VIF IDs for a node

Paramet
`tas` A Task agen instan

Returns
List of VIF

dictionaries, each dictionary will have an id entry with the ID of the VIF.

Module contents

ironic.drivers.modules.redfish package

Submodules

ironic.drivers.modules.redfish.bios module

class i

Base *iro* *dri* *bas* *BIO*

apply_c

Apply the BIO settings to the node

Paramet

- **tas** a Task agen instan containing the node to act

on.

- **set**
a list of BIO settings to be updated

Raises
Redfish-Connection-Error when it fails to connect to

Redfish

Raises
Redfish-Error on an error from the Sush library

cache_b

Stor or update the current BIO settings for the node

Get the current BIO settings and store them in the bios data table.

Parame

tas a Task agen instance containing the node to act on.

Raises

Redfish Connection Erro

whe
it
fails
to
connect
to

Redfish

Raises
Redfish-Error on an error from the Sush library

Raises
Unsupported-Driv tension if the system does not

support BIOS settings

factory
Reset the BIO settings of the node to the fac-

tory default.

Paramet

`tas` a Task ager instand containing the node to act on.

Raises

RedfishCon nectionErro whe it fails to connect to Redfish

Raises

RedfishError on an error from the Sush librar

`get_pro`

Return

the prop er- ties of the in- ter- face

Returns

dic- tio- nary of <pro erty nam de- scrip tion> en- tries

post_co

Per- form post con- fig- u- ra- tion ac- tion to store the

BIOS settings.

Ex- ten- sion poin to al- low ven- dor im- ple- men ta-

tion

to extend this class and override this method to perform a custom action to write the BIOS settings to the Redfish service. The default implementation performs a reboot.

Parame

- **tas** a Task ager instan containing the nod to act on.
- **set** a list of BIO setting to be update

post_re

Perform post reset action to apply the BIO factory reset.

Extension poin to allow vendor imple men tations

to extend this class and override this method to perform a custom action to apply the BIOS factory reset to the Redfish service. The default implementation performs a reboot.

Paramet

tas a Task agen instanc containing the node to act

on.

validat

Validate the drive information need by the red-

fish driver.

Paramet

tas a Task agen instan con-tain-ing the nod to act on.

Raises
In-vali Pa-ram-e-ter-Valu on mal-form pa-ram-e-ter(s)

Raises
Miss-ing-Pa-ram-e-ter-Valu on miss-ing pa-ram-e-ter(s)

ironic.drivers.modules.redfish.boot module

class i

Base
iro dri bas Boo

Virtual media boot interface over Redfish.

Virtual Media allows booting the system from the virtual CD/DVD drive containing the user image that BMC inserts into the drive.

The CD/ images mus be in ISO format and (depending on BMC implementation) could be pulled over HTTP, served as iSCSI targets or NFS volumes.

The baseline boot worl flow look like this:

1. Pull kernel, ram and ESP (FA partition image with

EFI boot loader) images (ESP is only needed for UEFI boot)

2. Create boot ISO out of images (#1) push it to Gla

and pass to the BMC as Swift temporary URL

3. Optionally create flop image with desired system

configuration data, push it to Glance and pass to the BMC as Swift temporary URL

4. Insert CD/ and

(optionally) flopp images and set prop

boot mode

For buil ing deploy or rescue ISO redfish boot interface uses *deploy_kernel*/*deploy_ramdisk* or *rescue_kernel*/*rescue_ramdisk* properties from *[instance_info]* or *[driver_info]*.

For buil ing boot (use ISO redfish boot interface seek *kernel_id* and *ramdisk_id* properties in the Glance image metadata found in *[instance_info]image_source* node property.

`capabil`

`clean_u`

Clea up the boot

of instanc

This meth clea up the environmen that was setu for boot

ing the instance.

Paramet

`tas` A task from Task ager

Returns

Non

`clean_u`

Clea up the boot of iron ramc

This meth clea up the environmen that was setu for boot ing the deploy ramdisk.

Paramet

`tas`

A task from Task agen

Returns

Non

`get_pro`

Return the properties of the interface

Returns

dictionary of <prop erty nam description entries

`prepare`

Prepare the boot of instanc over virtual media.

This meth prepare the boot of the instanc after reading relevant information from the nodes instance_info.

The internal logi is as follows

- If *boo* reques for this deploy is local, then set the node to boot from disk.
- Unless *boo* reques for this deploy is

ram
pass
root

disk/partition ID to virtual media boot image

- Otherwise build boot image, insert it into virtual

media device and set node to boot from CD.

Parame

`tas`

a task from Task agen

Returns

Non

Raises

Instan ploy Failure, if its try to boot iSCS volume

in BIOS boot mode.

`prepare`

Prepare the boot

of de- ploy or res- cue ram over vir-

tual media.

This meth pre- pare the boot of the de- ploy or res- cue ram

after reading relevant information from the nodes driver_info and instance_info.

Parame

- **tas** A task from Task ager
- **ram** the pa- ram- e- ters to be pass to the ram

Returns Non

Raises
Miss-ing-Pa-ram-e-ter-Valu if som in-for-ma-tion

is missing in nodes driver_info or instance_info.

Raises
In-valic Pa-ram-e-ter-Valu if som in-for-ma-tion

provided is invalid.

Raises
Iron icEx cep-tion, if som pow or set boot boot de-vice

operation failed on the node.

validat
Val-i-date the

deployment information for the tasks node.

This meth validates whe the driv and/ instance properties of the tasks node contains the required information for this interface to function.

Parame

tas

A Task agen instance containing the node to act on.

Raises

Invalid Parameter Valu on malform pa- ram- e- ter(s)

Raises
Miss ing- Pa- ram- e- ter- Valu on miss ing pa- ram- e- ter(s)

validat
Val- i- date that the node has re- quire prop er- ties for inspection.

Paramet
tas
A Task agen in- stan with the node be- ing chec

Raises

Missing-Parameter-Valu if node is missing one

or more required parameters

Raises

Unsupported-Driv tension

`ironic.`

Ejec virtual CDs and DVI

Paramet

- **tas** A task from Task ager
- **boo** sush boot device e.g. *VIR-TUA VIR-*

TUA
or
VIR-
TUA

or *None* to eject everything (default).

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
no
suit-
able
vir-
tual

CD or DVD is found on the node.

ironic.drivers.modules.redfish.inspect module

Red-
fish
In-
spec
In-
ter-
face

class i

Base
irc
dri
bas
Ins

get_prc
Re-
turn
the
prop
er-
ties
of
the
in-

ter- face

Returns
dic- tio- nary of <pro erty nam de- scrip tion: en- tries

inspect
In- spec hard ware to get the hard ware prop er- ties.

In- spec hard ware to get the es- sen- tial prop er- ties. It

fails if any of the essential properties are not received from the node.

Parame
tas
a Task agen in-

stan

Raises

Hard ware spec tion Fail ure if es sen tial prop er ties

could not be retrieved successfully.

Returns

The re sult ing state of in spec tion.

validat

Val i date the driv spec Nod de ploy men info

This meth val i date whe the driv prop er ties of the task

node contains the required information for this interface to function.

This meth is often executed synchro in API reques

so it should not conduct long-running checks.

Paramet

tas

A Task agen instan containing the node to act

on.

Raises

InvalidParameterValu on malform parame-

ter(s)

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-

ter(s)

ironic.drivers.modules.redfish.management module

`class i`

Base
irc
dri
bas
Man

`clear_s`
Clea
all
se-
cure
boot
keys

Parame
`tas`
a
task
from
Task
ager

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion
if

secure boot is now

supported.

Raises
Redfish-Error on runtime driv er- ror.

detect_
Detects and returns the hardware vendor.

Uses the Systems Man u- facturer field

Parame
tas
A task from Task agen

Raises
In- valid Pa- ram-

e-
ter-
Valu
if
an
in-
valic
com
po-

nent, indicator or state is specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quire
pa-
ram-

eter is missing

Raises
Red-
fish-
Er-
ror
on
drive
spec
prob
lems

Returns
Strin
rep-
re-
sent-
ing
the
BMC
re-
porte
Ven-
dor
or
Man

ufacturer, otherwise returns None.

`get_boo`

Get the current boot device for a node

Paramet

`tas` a task from Task agen

Raises Invalid Parameter Valu on malformed parameter(s)

Raises Missing Parameter Valu on missing parameter(s)

Raises

Redfish

Redfish-Connection-Erro when it fails to connect to

Raises

Redfish-Error on an error from the Sush library

Returns

a dictionary containing:

boot_d

the boot device one of *irc com boo* or Non if it

is unknown.

persist
Boo valu or Non True if the boot device persists Fals

otherwise. None if its unknown.

get_boo
Get the current boot mod for a node

Provide the current boot mod of the node

Parame
tas
A task from Task agen

Raises
Missing-Pa-

ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing

Raises

Driv
Op-
er-
a-
tionl
or
its
deriv
tive
in
case
of
driv

runtime error.

Returns

The
boot
mod
one
of
`iro`
`com`
`boo`
or
Non
if
it
is

unknown.

`get_inc`

Get
cur-
rent
state
of
the
in-
di-

cator of the hard

ware component.

Parame

- **tas** A task from Task agen
- **com** The hardware component; one of *iro com com*
- **ind** Indicator ID (as reported by *get_*

Raises
Missing-Parameter-Valu if

a required param-

eter is missing

Raises
Red-fish-Error on an error from the Sush li-bra

Returns
Current state of the indicator, one of *iro com*

indicator_states.

get_mac
Get MAC address information for the node

Parame
tas

A Task agen instan containing the node to act on.

Raises
Redfish-Con nection-Erro whe it fails to con-nect to Redfish

Raises
Redfish-Error on an error from the Sush library

Returns
a dictio-nary containing

MAC addres of enable

interfaces in a {mac: state} format

get_pro

Return the proper ties of the interface

Returns

dictionary of <pro erty nam description entries

get_sec

Get the current secure boot state for the node

Paramet

tas

A task from

Task agen

Raises
Missing-Param-e-ter-Valu if a re-quir pa-ram-

eter is missing

Raises
Red-fish-Er-ror or its deriv tive in case of a driv

runtime error.

Raises
Un-sup-port-ed-Driv ten-sion if se-cure boot is not

supported by the hardware.

Returns
Boo

get_se

Get sensors data

Parame

tas

a Task ager instan

Raises

Fail ToGetSensorData whe getting the sensor data

fails.

Raises

Fail ToPa eSe sorData whe parsing sensor data fails

Raises

Invali ParameterValu if

required parameters are missing.

Raises
MissingParameterValue if a required parameter is missing.

Returns
returns a dict of sensor data grouped by sensor type.

get_sup
Get a list of the supported boot devices.

Parame
tas
a

task from Task ager

Returns
A list with the supported boot devices defined in *irc*

common.boot_devices.

get_sup
Get a list of the supported boot mod

Parame
tas
A task from Task ager

Returns
A list with the supported boot mod defined in *irc*

com

boot_modes. If boot mode support cant be determined, empty list is returned.

get_sup

Get a map of the supported indicators (e.g. LED

Parame

- **tas** A task from Task agen
- **com** If not *Non* return indicator information for just this component, otherwise return indicators for all existing components.

Returns

A dictionary of hard

ware
com
po-
nent
(*ir*
com
com
as

keys with values being dictionaries having indicator IDs as keys and indicator properties as values.

```
↪": true,
```

(continues on next page)

(continued from previous page)

```
↪    "OFF",
```

```
↪    "ON"
```

(continues on next page)

(continued from previous page)

```
↪": true,
↪": [
```

(continues on next page)

(continued from previous page)

```
↪    "ON"
```

(continues on next page)

(continued from previous page)

```
↪": true,
↪": [
↪   "OFF",
↪"ON"
```

(continues on next page)

(continued from previous page)

```
```

inject_

Inject NM Non Mas able Interrupt

Inject NM (Non Mas able Interrupt for a

node immediately.

Paramet

tas A Tas agen instan containing the node to act on.

Raises Invali ParameterValu on malform parameter(s)

Raises Miss ingParameterValu on missing parameter(s)

Raises
Redfish Con nection Erro whe it fails to connect to Redfish

Raises
Redfish Error on an error from the Sush library

reset_s
Reset secure boot keys to manufacturing defaults.

Paramet
tas
a task from Task

ager

Raises

Unsupported-Driv tension if secure boot is now

supported.

Raises

Redfish-Error on runtime driv error.

restore

Restore boot device if need

Chec the redfish_ internal flag and sets the one-time boot device accordingly. A warning is issued if it fails.

This meth is suppose to be calle from the Redfish pow interface and should be considered private to the Redfish hardware type.

Parame

- **`tas`** a task from Task agen
- **`sys`** a Redfish System object.

`set_boo`

Set the boot device for a node

Set the boot device

to use on next re-boot of the node

Parame

- **tas** a task from Task agen
- **dev** the boot device one of *iro com boo*
- **per** Boo valu True if the boot device will persist to

all future boots, False if not. Default: False.

Raises
In-valid Pa-ram-

e-
ter-
Valu
on
mal-
form
pa-
ram-
e-

ter(s)

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-

ter(s)

Raises
Red-
fish-
Con
nec-
tion-
Erro
whe
it
fails
to
con-
nect
to

Redfish

Raises
Red-
fish-
Er-
ror
on
an
er-
ror
from

the Sush li- brar

set_boo

Set the boot mod for a node

Set the boot mod to use on next re- boot of the node

Paramet

- **tas** A task from Task ager
- **mod** The boot mod one of *iro com boo*

Raises

In- valid Pa-

ram-
e-
ter-
Valu
if
an
in-
valic
boot
mod

is specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing

Raises
Red-
fish-
Con
nec-
tion-
Erro
whe
it
fails
to
con-
nect
to

Redfish

Raises
Red-
fish-
Er-
ror
on
an
er-
ror

from the Sush library

set_inc

Set indicator on the hardware component to

the desired state.

Paramet

- **tas** A task from Task agen
- **com** The hardware component one of *iro com com*
- **ind** Indicator ID

(as re-
port
by
*get_*

- **sta**
  De-
  sire
  state
  of
  the
  in-
  di-
  ca-
  tor,
  one
  of
  iro

common.indicator_states.

Raises
In-
vali
Pa-
ram-
e-
ter-
Valu
if
an
in-
vali
com
po-

nent, indicator or state is specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing

Raises
Redfish-Error on an error from the Sush library

set_sec
Set the current secure boot state for the node

Paramet

- **tas** A task from Task agen
- **sta** A new state as a bool

Raises
Missing-Parame-

ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing

Raises
Red-
fish-
Er-
ror
or
its
deri
tive
in
case
of
a
driv

runtime error.

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion
if
se-
cure
boot
is
not

supported by the hardware.

update_
Up-
date
the
firm
on
the
node

Parame

- **tas** a Task ager in- stan con- tain- ing the node to act

on.

- **fir** A list of firm im- ages are to ap- ply.

Returns
Non if it is com plete

Raises
Red- fish- Er- ror on an er- ror from the Sush li- brar

validat

Validate the driv information need by the redfish driver.

Paramet

`tas` a Task ager instanc containing the node to act on.

Raises

Invali ParameterValu on malform parameter(s)

Raises

Miss ingParameter-Valu on missing parameter(s)

ironic.drivers.modules.redfish.power module

class i

Base *irc dri bas Pow*

get_pow

Get the current pow state of the task node

Paramet **tas** a Task agen instance containing the node to act on.

Returns a pow

state One of *iro com sta*

Raises
In- valic Pa- ram- e- ter- Valu on mal- form pa- ram- e-

ter(s)

Raises
Miss ing- Pa- ram- e- ter- Valu on miss ing pa- ram- e-

ter(s)

Raises
Red- fish- Con nec- tion- Erro whe it fails to con- nect to

Redfish

Raises
Red- fish- Error on an error from the Sush li- brar

`get_pro`
Return the prop- erties of the interface

Returns
dictionary of <pro erty nam description en- tries

`get_sup`
Get a list of the sup- port pow

state

Paramet

tas

A Task ager instance containing the node to act

on. Not used by this driver at the moment.

Returns

A list with the supported power state defined in *irc* *com*

states.

reboot

Perform a hard reboot of the task node

Paramet

- **tas** a Task ager

in- stan con- tain- ing the node to act

on.

- **tim** Tim to wait for the node to be- com pow ered on.

Raises
Miss ing- Pa- ram- e- ter- Valu if a re- quir pa- ram-

eter is missing.

Raises
Red- fish- Con nec- tion- Erro whe it fails to

connect to

Redfish

Raises
RedfishError on an error from the Sush li- brar

set_pow
Set the pow state of the task node

Paramet

- **tas** a Task agen instanc containing the node to act

on.

- **pow** Any pow state

from
iro
com
sta

- **`tim`**
  Tim to wait for the node to reac the reques state

Raises
Miss ingParameterValu if a requir parameter is missing.

Raises
RedfishCon nectionErro whe it fails to connect to Redfish

Raises

RedfishError on an error from the Sush library

validat

Validate the driver information needed by the redfish driver.

Paramet

tas a Task ager instanc containing the node to act on.

Raises

Invalid Parameter-

Valu on mal-form pa-ram-e-

ter(s)

Raises

Miss ing-Pa-ram-e-ter-Valu on miss ing pa-ram-e-

ter(s)

ironic.drivers.modules.redfish.raid module

class i

Base *irc dri bas RAI*

apply_c

Ap-plies RAI con-fig-u-ra-tion on the give node

Paramet

- **`tas`** A Task agen instan
- **`rai`** The RAI configuration to apply.
- **`cre`** Setting this to Fals indicate not to create

root volume that is specified in raid_config. Default value is True.

- **`cre`** Setting this to Fals indicate not to create non-root volumes (all except the root volume) in raid_config. Default value is True.

- `del` Setting this to True indicate to dele RAI configuration prior to creating the new configuration.

Raises
Invali Parameter Valu if the RAI configuration is invalid.

Returns
state if RAI configuration is in prog asyn chro or None if it is complete.

`create_`

Cre-

ate RAI configuration on the nod

This meth creates the RAI configuration as read from

node.target_raid_config. This method by default will create all logical disks.

Paramet

- **`tas`** Task agen object containing the nod
- **`cre`** Setting this to Fals indicate not

to create root volume that is specified in the nodes target_raid_config. Default value is True.

- **cre** Setting this to Fals indicate not to create non-root volumes (all except the root volume) in the nodes target_raid_config. Default value is True.
- **del** Setting this to True indicate to dele RAI configuration prior to creating the new configuration. Default is False.

Returns state if RAI configuration is in prog asyn chro or None if it is complete.

Raises
Redfish-Error if there is an error creating

the configuration

delete_

Dele RAI configuration on the node

Paramet

tas

Task agen object containing the node

Returns

state (clea ing) or state (deploy men if dele tion is progress asynchronously or None if it is complete.

in

get_pro

Return the prop erties of the interface

Returns

dictionary of <pro erty nam description entries

post_cr

Perform post create_ action to commit the config.

Extension poin to allow vendor implementations

to extend this class and override this method to perform a custom action to commit the RAID create configuration to the Redfish service.

Parame

- **tas** a Taskager instan containing the node to act

on.

- **rai** a list of dictionaries containing the RAI

configuration operation details.

- **ret** state to return base on operation being invoked

post_de

Perform post delete action to commit the config.

Extension point to allow vendor implementations to extend this class and override this method to perform a custom action to commit the RAID delete configuration to the Redfish service.

Parame

- **tas** a Task ager instan containing the node to act on.

- **rai** a list of dictionaries containing the RAI configuration operation details.
- **ret** state to return base on operation being invoked

pre_cre

Perform requir actions before creating config.

Extension poin to allow vendor implemen tations

to extend this class and override this method to perform custom actions prior to creating the RAID configuration on the Redfish service.

Parame

- **tas** a Task agen instanc containing the node to act

on.

- **log** list of logical disk to create.

pre_del Perform required actions

before dele ing config.

Extension poin to allow vendor implemen tation

to extend this class and override this method to perform custom actions prior to deleting the RAID configuration on the Redfish service.

Parame

- **`tas`** a Task ager instan containing the node to act on.
- **`vol`** list of volume to dele

`validat`

Val-

i-
date
the
give
RAI
con-
fig-
u-
ra-
tion.

Paramet

- **tas**
  A
  Task
  ager
  in-
  stan
- **rai**
  The
  RAI
  con-
  fig-
  u-
  ra-
  tion
  to
  val-
  i-
  date

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
the
RAI
con-
fig-
u-

ration is invalid.

`volume_`
Han

dle error from faile Volume lection.

Extension poin to allow vendor implemen tations

to extend this class and override this method to perform a custom action if the call to VolumeCollection.create() fails.

Paramet

- **tas** a Task ager instanc containing the node to act

on.

- **exc** the exception raise

by Volume lec- tion.

- **vol** the sush Volume lec- tion in- stan
- **pay** the pay- load pass to the faile cre- ate()

Returns
New cre- ated Volume re- sour or Task i- tor if asyn

task.

Raises
Red- fish- Er- ror if ther is

an error creating

the virtual disk.

`ironic.`

Convert size in logical_ from gb to byte

`ironic.`

Create a single virtual disk on a RAI controll

Paramet

- **tas** Task ager

object containing the node

- **`rai`** id of the RAI controll
- **`phy`** ids of the phys ical disk
- **`rai`** RAI level of the virtual disk
- **`siz`** size of the virtual disk
- **`dis`** nam of the virtual disk

(optiona

- **spa** Number of span in virtual disk (optiona
- **spa** Number of disk per span (optiona
- **err** func tion to call if volume create fails (optiona

Returns
New created Volume resour or Task i-

tor if asyn

task.

Raises

Red- fish- Con nection- Erro whe it fails to connect to

Redfish.

Raises

Red- fish- Error if there is an error creating

the virtual disk.

`ironic.`

Get the phys i- cal driv of the node for RAI controll

Paramet

nod
an
iron
node
ob-
ject.

Returns
a
list
of
Driv
ob-
jects
from
sush

Raises
Red-
fish-
Con
nec-
tion-
Erro
whe
it
fails
to
con-
nect
to

Redfish

Raises
Red-
fish-
Er-
ror
if
there
is
an
er-
ror
get-
ting
the

drives via Redfish

`ironic.`
Up-
date
node

raid_ field with current logical disk

Paramet

nod node for whic to update the raid_ field

ironic.drivers.modules.redfish.utils module

class i

Base obj

Cach of HTT sessions credentials

AUTH_CI

ironic.

Get information on MAC address of

en-
able
port

using Redfish.

Paramet

- **tas** a Task ager in- stan con- tain- ing the node to act

on.

- **sys** a Red- fish Sys- tem ob- ject

Returns a dic- tio- nary con- tain- ing MA ad- dres of en- able

interfaces in a {mac: state} format

`ironic.`

Get a Red

fish System that represents a node

Paramet

nod an Iron node object

Raises

Redfish Connection Error when it fails to connect to

Redfish

Raises

Redfish Error if the System is not registered

in Redfish

`ironic.`

Get a

Redfish

Task-i-tor for a nod

Paramet

- **nod** an Iron node ob-ject
- **uri** the URI of a Task-i-tor

Raises
Red-fish-Con-nec-tion-Erro whe it fails to con-nect to

Raises
Red-fish-Er-ror whe the Task-i-tor

is not avai able

in Redfish

`ironic.`

Get a node update service

Paramet

`nod` an Iron node object

Raises

Redfish-Con nection-Erro whe it fails to connect to

Redfish

Raises

Redfish-Error whe the Update Service is not registered in Redfish

ironic.

Pars the information requir for Iron to connect

to Redfish.

Paramet

nod an Iron node object

Returns

dictionary of parameters

Raises

InvalidParameterValu on malformed parame-

ter(s)

Raises

Miss

ingParameterValu on miss ing parame-

ter(s)

`ironic.`

Wai until Redfish system is read

Paramet

`nod` an Iron node object

Raises

RedfishConnectionErro on time out.

ironic.drivers.modules.redfish.vendor module

Vendor Interface for Redfish drive and its supporting methods.

class i

Base
iro
dri
bas
Ven

Vend spec interface for Redfish driv

eject_v

Ejec a virtual media device

Parame

- **tas** A Task

ager ob- ject.

- **kwa** The ar- gu- men sent with ven- dor pass The op- tion

kwargs are:: boot_device: the boot device to eject

get_pro Re- turn the prop er- ties of the in- ter- face

Returns dic- tio- nary of <pro erty nam de- scrip tion en- tries

validat Val- i- date vend spec

actions

Chec if a valic vendor pass meth was pass and validate the parameters for the vendor passthru method.

Paramet

- **tas** a Task agen instanc containing the node to act on.
- **met** meth to be validate
- **kwa** kwa containing the

vendor pass meth ods parameters.

Raises
Invalic ParameterValu if any of the parameters have invalid value.

Module contents

ironic.drivers.modules.storage package

Submodules

ironic.drivers.modules.storage.cinder module

class i

Base *irc dri bas Stc*

A storage_ drive supporting Cinder.

attach_

In-form the stor-age sub-sys-tem to at-tach all vol-

umes for the node.

Paramet

tas

The task ob-ject.

Raises

Stor ageE ror If an un-der-ly-ing ex-cep-tion or

failure is detected.

detach_

In-form the stor-age sub-sys-tem to de-tach all volumes for the node.

This ac- tion is re- tried in case of fail- ure.

Parame

- `tas` The task ob- ject.
- `con` The dic- tio- nary rep- re- sent- ing a node con- nec- tivity as defined by _generate_connector(). Generated if not passed.
- `abo` Boo rep- re- sent- ing if this de- tach men was re-

quested to handle aborting a failed attachment

Raises
Stor ageE ror If an underlying exception or

failure is detected.

get_pr
Return the prop erties of the interface

Returns
dictionary of <pro erty nam descrip tion: entries

should_
Deter min if deploy shou

perform the image writ

out.

Paramet

tas

The task object.

Returns

True if the deployment writ out process shou be ex-

ecuted.

validat

Validate stor- age_ configuration for Cinder

usage.

In order to provide

fail fast functionality prio to nodes being requested to enter the active state, this method performs basic checks of the volume connectors, volume targets, and operator defined capabilities. These checks are to help ensure that we should have a compatible configuration prior to activating the node.

Parame

`tas`

The task object.

Raises

InvalicParameterValu If a misconfiguration or mismatch exists that would prevent storage the cinder storage driver from initializing attachments.

ironic.drivers.modules.storage.external module

class `i`

Base *iro dri bas Sto*

Externally driv Storage

Interface

attach_

Inform the storage subsystem to attach all vol-

umes for the node.

Parame

tas

A Task ager instan

Raises

Unsupport ed-Driv tension

detach_

Inform the storage subsystem to detach all vol-

umes for the node.

Paramet

tas

A Task ager instan

Raises

Unsupported Driv tension

get_pro

Return the properties of the interface

Returns

dictionary of <pro erty nam description entries

should_

Determine if deploy shou perform the image writ out.

This enable the user to define a volume and Iron understand that the image may already exist and we may be booting to that volume.

Paramet

`tas` The task object.

Returns

True if the deploy men writ out process shou be executed.

`validat`

Validate the driv spec

Nod deploy men info

This meth validate whe the driv and/ instan prop erties

of the tasks node contains the required information for this interface to function.

This meth is often execute synchro in API reques

so it should not conduct long-running checks.

Paramet

`tas` A Task ager instan containing the node to act

on.

Raises
InvaliParameterValu on malform parame-

ter(s)

Raises
MissingParameterValu on missing parame-

ter(s)

ironic.drivers.modules.storage.noop module

class i

Base
irc
dri
bas
Stc

No-op Storage Interface

attach_

Inform the storage subsystem to attach all volumes for the node.

Paramet

`tas`

A Task ager instanc

Raises

Unsupported-Driv tension

`detach_`

Inform the storage subsystem to detach all volumes for the node.

Paramet

`tas`

A Task ager

instanc

Raises

Unsupport-ed-Driv tension

get_prc

Return the prop erties of the interface

Returns

dictionary of <prc erty nam description entries

should_

Determine if deploy shou perform the image writ out.

Paramet

tas

A Task agen in- stan

Returns

Boo valu to in- di- cate if the in- ter- face ex- pect

the image to be written by Ironic.

Raises

Un- sup- port- ed- Driv ten- sion

validat

Val- i- date the drive spec Nod de- ploy men info

This meth val- i- date whe

the driv and/ instan properties of the tasks node contains the required information for this interface to function.

This meth is often execute synchro in API reques so it should not conduct long-running checks.

Paramet

`tas` A Task agen instan containing the node to act on.

Raises

Invalid ParameterValu on malform

pa-
ram-
e-

ter(s)

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-

ter(s)

Module contents

ironic.drivers.modules.xclarity package

Submodules

ironic.drivers.modules.xclarity.common module

`ironic.`

`ironic.`

Val-
i-
date
node
con-
fig-
u-
ra-
tion
and
re-
turn
xcla

ity hardware id.

Val-
i-
date

whe node configu tion is consistent with XCl

ity and returns the XClarity Hardware ID for a specific node. :param node: node object to get information from :returns: the XClarity Hardware ID for a specific node :raises: MissingParameterValue if unable to validate XClarity Hardware ID

`ironic.`

Generates an instanc of the XCl ity clien

Generates an instanc of the XCl ity clien using the

imported xclarity_client library.

Paramet

`nod` an ironi node object.

Returns

an instanc of the XCl ity clien

Raises
XCl i- ty- Er- ror if cant get to the XCl ity clien

`ironic.`

Pars a node drive val- ues.

Pars the drive of the node read de- fault val- ues and re- turns

a dict containing the combination of both.

Paramet
nod
an ironi node

object to get information from

Returns
a dict containing information pars from driv

Raises
Invalid Parameter Valu if som required infor-

mation is missing on the node or inputs is invalid.

`ironic.`

Tran lates ironics pow action strin to XCl itys format.

Paramet

pow
pow
action
strin
to
be
trans
late

Returns
the
pow
action
trans
late

`ironic.`

Tran
lates
XCl
itys
pow
state
strin
to
be
consistent

with Ironic.

Paramet

pow
pow
state
strin
to
be
trans
late

Returns
the
trans
late
pow
state

ironic.drivers.modules.xclarity.management module

class i

Base
irc
dri
bas
Man

get_boc
Get
the
current
boot
device
for
the
task
node

Parame
tas
a
task
from
Task
ager

Returns
a
dictionary
containing:
:boo
the
boot
device
one

of [PXE, DISK, CDROM, BIOS] :persistent: Whether the boot device will persist or not It returns None if boot device is unknown.

Raises
Invalid
Pa-

rameterValu if the boot device is

unknown

Raises

XCl ityError if the com munication with

XClarity fails

get_pro

Return the prop erties of the interface.

Returns

dictionary of <pro erty nam description en-

tries

get_ser

Get sensors data

Paramet

tas

a Task agen instanc

Raises

Noth plemented- Error

get_sup

Gets a list of the supporte boot devices

Paramet

tas

a task from Task agen

Returns

A list with the supporte boot devices

de-
fined
in
iro

common.boot_devices.

set_boo

Sets
the
boot
de-
vice
for
a
node

Paramet

- **tas**
  a
  task
  from
  Task
  ager
- **dev**
  the
  boot
  de-
  vice
  one
  of
  the
  sup-
  porte
  de-
  vice
  liste

in *ironic.common.boot_devices*.

- **per**
  Boo
  valu
  True
  if
  the
  boot
  de-
  vice
  will

per- sist to

all future boots, False if not. Default: False.

Raises
In- valic Pa- ram- e- ter- Valu if an in- valic boot de-

vice is specified.

Raises
XCl i- ty- Er- ror if the com mu- ni- ca- tion with

XClarity fails

validat
Val- i- date the drive spec info sup- plied

This meth val- i- date

if the drive prop- erty of the sup- plied

tasks node contains the required information for this driver to manage the node.

Parame

tas

a task from Task ager

ironic.drivers.modules.xclarity.power module

class i

Base *iro dri bas Pow*

get_pow

Gets the cur- rent pow state

Parame

tas

a Task ager in- stand

Returns

one of *iro com sta* POW

POW or ERROR

Raises
XCl ityError if fails to retriev pow state of

XClarity resource

get_pro

Return the properties of the interface.

Returns
dictionary of <pro erty nam description entries

reboot

Soft reboot the

node

Parame

- **tas** a Task agen instan
- **tim** time out (in seconds Unsupport by this interface.

set_pow

Turn the current pow state on or off.

Parame

- **tas** a Task agen instan
- **pow** The de-

sired pow state POW POW or RE-BOO from iro

common.states.

- tim time out (in sec-onds Un-sup-port by this in-ter-

face.

Raises
In-valid Pa-ram-e-ter-Valu if an in-valid pow state

was specified.

Raises
XCl i-ty-Er-ror if XCl ity

fails setting the pow

state.

validat

Validate the driv spec info supplied

This meth validate if the driv property of the supplied

tasks node contains the required information for this driver to manage the power state of the node.

Parame

tas a task from Task agen

Module contents

Submodules

ironic.drivers.modules.agent module

class i

Base

irc

dri
moc
age
Age
irc
dri
moc
age
Age
irc
dri

base.DeployInterface

Interface for depl relat actions

clean_u

Clea up the deploy men environ men for this node

If prep ra tion of the deploy men environ men ahea

of time is possible, this method should be implemented by the driver. It should erase anything cached by the *prepare* method.

If implemen this meth mus be iden potent. It may be

called multiple times for the same node on the same conductor, and it may be called by multiple conductors in parallel. Therefore, it must not require an exclusive lock.

This meth is calle before *tear*

Parame

`tas` a Task agen instan

`deploy`

Perform a deploymen to a node

Perform the necessary work to

deploy an image onto

the specified node. This method will be called after prepare(), which may have already performed any preparatory steps, such as pre-caching some data for the node.

Parame

`tas`

a Task ager instan

Returns

status of the deploy One of iron

`get_pro`

Return the properties of the interface

Returns

dictionary of <property name description: en-

tries

prepare

Prepare the deployment environment for this node

Paramet

tas

a Task ager instand

Raises

Network Error: if the previous clear ing port can-

not be removed or if new cleaning ports cannot be created.

Raises

Invalid Parameter Valu when the wron pow

state
is

specified or the wrong driver info is specified for power management.

Raises
Stor
ageE
ror
If
the
storage
driv
is
unable
to
at-

tach the configured volumes.

Raises
othe
exceptions
by
the
node
pow
driv
if
som
thin
wro

occurred during the power action.

Raises
exception.
if
image_
is
not
Glan
href
and
is
not

HTTP(S) URL.

Raises

exception. if network validation fails

Raises
any boot interface prepare exceptions

should_
When agent boot is managed by ironi

validat
Validate the drive spec Nod deployment info

This meth validate whe

the prop erties of the supplie

node contain the required information for this driver to deploy images to the node.

Paramet

tas a Task ager instanc

Raises

MissingParameterValu if any of the requir

parameters are missing.

Raises

InvalidParameterValu if any of the param-

eters have invalid value.

class i

Base

irc

dri
moc
age
Age

has_dec

Whe
the
driv
sup-
port
de-
com
pose
de-
ploy
step

Pre-
vi-
ousl
(sinc
Roc
driv
used
a
sin-
gle
de-
ploy
de-
ploy

step on the deploy interface. Some additional steps were added for the direct and iscsi deploy interfaces in the Ussuri cycle, which means that more of the deployment flow is driven by deploy steps.

prepare

write_i

class i

Base
iro
dri
bas
RAI

Im-
ple-
men
ta-
tion

of RAI Interface whic uses agen ram

apply_c

Applies RAI configuration on the give node

Paramet

- **`tas`** A Task agen instan
- **`rai`** The RAI configuration to apply.
- **`del`** Setting this

to True indicates to dele RAI con-

figuration prior to creating the new configuration.

Raises

Invalid Parameter Valu if the RAI configu-

ration is invalid.

Returns

state if RAI configuration is in prog asynchro

or None if it is complete.

create_

Create a RAI configuration on a bare meta using agent ramdisk.

This meth creates a RAI configuration on the give node.

Parame

- **tas** a Task ager instan
- **cre** If True a root volume is created during RAI configuration. Otherwise, no root volume is created. Default is True.
- **cre** If True

non-root volumes are created. If False, no

non-root volumes are created. Default is True.

Returns

state if operation was successfully invoke

Raises

Missing-ParameterValu if node is missing or

was found to be empty after skipping root volume and/or non-root volumes.

delete_

Dele RAI configuration on the give

node

Paramet

tas
a Task agen in- stand

Returns
state if op- er- a- tion was suc- cess fully in- voke

get_cle

Get the list of clea steps from the agen

Paramet

tas
a Task agen ob- ject con- tain- ing the node

Raises
Noc
if the clea steps

are not yet avai able (cac for ex-

ample, when a node has just been enrolled and has not been cleaned yet.

Returns
A list of clea step dic- tio- nar- ies

get_dep
Get the list of de- ploy step from the agen

Parame
tas a Task agen ob- ject con- tain- ing the node

Raises
Ins if the de- ploy step

are not yet avai able (cac for example, when a node has just been enrolled and has not been deployed yet.

Returns
A list of deploy step dictionaries

get_pro
Return the prop erties of the interface

class i
Base
iro dri bas Res

Implementation of RescueI terface whic uses

ager
ram

clean_

Clea up af- ter RES CUI WA time out/ or fin- ish- ing

rescue.

Res- cue pass wor shou be re- mov from the node and ram boot

environment should be cleaned if Ironic is managing the ramdisk boot.

Parame

tas a Task ager in- stan with the node

Raises

Net- worl Er- ror if

the rescue port cannot be removed.

get_pro

Return the properties of the interface

rescue

Boo a rescue ram on the node

Parame

tas

a Task ager instanc

Raises

NetworkError if the tenant port cannot

be removed.

Raises
In- valic Pa- ram- e- ter- Valu whe the wror pow state is

specified or the wrong driver info is specified for power management.

Raises
othe ex- cep- tions by the node pow drive if som thing wror

occurred during the power action.

Raises
any boot in- ter- face pre- pare ex- cep- tions

Returns
Re- turns state

unrescu

Attemp to mov a rescued node back to active state

Paramet

tas a Task ager instanc

Raises

NetworkError if the rescue ports cannot be removed.

Raises

InvalidParameterValu whe the wron pow state is specified or the wrong driver info is specified for power management.

Raises

othe ex- cep- tion by the node pow driv if som thin wro

occurred during the power action.

Raises

any boot in- ter- face pre- pare ex- cep- tion

Returns

Re- turn state

validat

Val- i- date that the node has re- quir prop er- ties for

agent rescue.

Paramet

tas

a Task

agen
in-
stan
with
the
node
be-
ing
chec

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
in-
stan
has
emp
pass

word or rescuing network UUID config option has an invalid value.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
node
is
miss
ing
one

or more required parameters

`ironic.`

Chec
if
the
re-
ques
im-
age
is
large
than

the ram size.

Paramet

tas

a Task agen instance containing the node to act on.

Raises

Invalid Parameter Valu if size of the image is greater than the available ram size.

`ironic.`

Validates configuration options require to perform HTTP provisioning.

Paramet

nod
an
ironi
node
ob-
ject

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
re-
quire
op-
tion(
is

not set.

`ironic.`
Chec
that
the
pro-
vide
prox
pa-
ram-
e-
ters
are
valid

Paramet
nod
an
Ironi
node

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
any

of the pro- vide

proxy parameters are incorrect.

ironic.drivers.modules.agent_base module

class i

Base

obj

Mix with base meth ods not re- ly- ing on any de- ploy step

clean_u

Clea up the de- ploy men en- vi- ron- men for the task

node.

Un- link TFT and in- stan im- ages

and triggers image cach

cleanup. Removes the TFTP configuration files for this node.

Paramet

`tas` a Task ager instanc containing the nod to act

on.

`prepare`

Boo into the ager to prepare for clea ing.

Paramet

`tas` a Task ager object containing the nod

Raises

Nod Clea ing-

Fail-
ure,
Net-
worl
Er-
ror
if
the
pre-
vi-

ous cleaning ports cannot be removed or if new cleaning ports cannot be created.

Raises

In-
valid
Pa-
ram-
e-
ter-
Valu
if
clean
ing
net-
worl
UUI

config option has an invalid value.

Returns

state
to
sig-
nify
an
asyn
chro
pre-
pare

refresh

Re-
fresh
the
node
cach
step

Paramet

- **tas**
  a
  Task

ager in- stanc

- **ste** clea or de- ploy

should_ Whe ager boot is man aged by iron

take_ov Take over man age- men of this node from a dead con- duc-

tor.

Parame **tas** a Task ager in- stanc

tear_do Tear dow a pre- vi- ous de- ploy men on the task node

Pow off the node All actual clea up is done in the clea

method which should be called separately.

Parame

`tas` a Task agen instance containing the node to act

on.

Returns

deploy state DEL

Raises

Network Error if the clea ing

port can-not be re-

moved.

Raises
In-valic Pa-ram-e-ter-Valu whe the wron state is spec

ified or the wrong driver info is specified.

Raises
Stor ageE ror whe vol-ume de-tach men fails

Raises
othe ex-cep-tions by the node pow drive if som thing wron

occurred during the power action.

tear_do
Clea

up the PXE and DHC files af- ter clear ing.

Parame

tas

a Task agen ob- ject con- tain- ing the node

Raises

Nod Clea ing- Fail- ure, Net- work Er- ror if the clear ing

ports cannot be removed

class i

Base

irc dri moc age Hea irc dri moc age Age

Mix with deploy meth ods.

`configu`

Help meth to configure local boot on the node

This meth triggers boot load installation on the node On successful installation of bootloader, this method sets the node to boot from disk.

Parame

- **`tas`** a Task agen object containing the

node

- **`roo`** The UUI of the root partition. This is used for identifying the partition which contains the image deployed or None in case of whole disk images which we expect to already have a bootloader installed.
- **`efi`** The UUI of the efi system partition. This is used only in uefi boot mode.
- **`pre`** The UUI of the PRe Boo partition. This is used only for booting ppc64* hardware.

Raises

In-

stan ploy Failure if boot load installation faile

or on encountering error while setting the boot device on the node.

execute

Execute a clea step asyn chro on the agen

Parame

- **tas** a Task agen object containing the node
- **ste** a clea step dictionary to ex-

command status

e- cute

Raises
Nod Clea ing- Fail- ure if the agen does not re- turn a

Returns
state to sig- nify the step will be com plete asyn

execute

Ex- e- cute a de- ploy step

Wer try- ing to find a step amo both out- of- banc and in-band steps. In case of duplicates, out-of-band steps take priority. This property allows having an out-of-band deploy step that calls into a corresponding in-band step after some preparation (e.g. with additional input).

Paramet

- **tas** a Task ager object containing the node
- **ste** a deploy step dictionary to execute

Raises
Instanc ploy Failure if the ager does not return a command status

Returns
state

to signify the step will be com plete asyn

get_cle

Get the list of clea steps from the agen

Parame

tas

a Task agen object containing the node

Raises

Noc

if the clea steps are not yet avai able (cac for example, when a node has just been enrolled and has not been cleaned yet.

Returns

A

list of clea step dic-tio-nar-ies

get_dep

Get the list of de-ploy step from the agen

Parame

tas

a Task agen ob-ject con-tain-ing the node

Raises

Ins if the de-ploy step are not yet avai able (cac for

example, when a node has just been enrolled and has not been deployed yet.

Returns

A list

of deploy step dictionaries

prepare

Prepare instance to boot

Paramet

- **tas** a Task agen object containing the node
- **roo** the UUI for root partition
- **efi** the UUI for the efi partition

Raises

Invalid State if fails to prepare instan

process

Start the next clea step if the previous one is com

plete.

In order to avoi errors and mak agen upgrad painless,

the agent compares the version of all hardware managers at the start of the process (the agents get_clean|deploy_steps() call) and before executing each step. If the version has changed between steps, the agent is unable to tell if an ordering change will cause an issue so it returns VERSION_MISMATCH. For automated cleaning, we restart the entire cleaning cycle. For manual cleaning or deploy, we dont.

Additionally, if a

step includ the reboot prop erty

set to True, this method will coordinate the reboot once the step is completed.

reboot_

Help meth to trigger reboot on the node and finish

deploy.

This meth initiates a reboot on the node On success

it marks the deploy as complete. On failure, it logs the error and marks deploy as failure.

Parame

tas a Task ager object containing the

nod

Raises

Instan ploy Failure, if nod reboot faile

`refres`

Refres the nod cach clea step from the boot agen

Gets the nod step from the boot agen and cach them The step are cached to make get_clean_steps() calls synchronous, and should be refreshed as soon as the agent boots to start cleaning/deploy or if cleaning is restarted because of a hardware manager version mismatch.

Paramet

- **`tas`** a Task agen instan

- **ste** clea or de- ploy

Raises
Nod Clea ing- Fail- ure or In- stan ploy Fail- ure if the

agent returns invalid results

tear_do
A de- ploy step to tear dow the agen

Paramet
tas a Task agen ob- ject con- tain- ing the node

class i
Base `obj`

Mix with out-of-banc deploy steps

boot_in

Deploy step to boot the final instan

Paramet

tas a Task agen object containing the node

switch_

Deploy step to swit the node to the tenant network

Paramet

tas a Task agen object containing the nod

class i

Bas

obj

Mix clas implemen ing hear beat processing.

collect

continu

Star the next clea ing step if the previous one is complete.

Parame

tas

a Task agen instan

continu

Continue-ues the deploy men of bare node

This meth continues the deploy men of the bare node after the ramdisk have been booted.

Parame

`tas` a Task agen instan

`deploy_`

Chec if the deploy men has start already

Returns

True if the deploy has start

Fals otherwise

deploy_

Chec if the deployment is already com plete

Returns

True if the deployment is complete Fals otherwise

has_dec

Whe the drive supports decomposed deploy steps

Previously (sinc Rock drive used

a single de- ploy de- ploy

step on the deploy interface. Some additional steps were added for the direct and iscsi deploy interfaces in the Ussuri cycle, which means that more of the deployment flow is driven by deploy steps.

`heartbe`

Pro- cess a hear beat

Paramet

- `tas` task to worl with
- `cal` agen HTT API URI
- `age` The ver- sion of the agen that is hear beat ing
- `age` TLS cer- tifi-

cate for the agen

- **age** Status of the heart beating agen
- **age** Status message that describ the agen

propert Define node state whe heart beating is allowe

in_core Chec if we are in the deploy deploy step

As-
sum
that
we
are
in
the
DE-
PLO
WA
state

Parame
tas
a
Task
agen
in-
stan

Returns
True
if
the
cur-
rent
de-
ploy
step
is
de-
ploy

process
Star
the
next
clea
step
if
the
pre-
vi-
ous
one
is
com

plete.

Parame

- **tas**

a Task agen in- stanc

- ste clea or de- ploy

reboot_

Met in- voke af- ter the de- ploy men is com plete

Parame

tas a Task agen in- stanc

refresh

Re- fresh the node cach clea step

Parame

tas a Task agen in- stanc

refresh

Re-

fres the nod cach clea step

Parame

- **tas** a Task agen instan
- **ste** clea or deploy

`ironic.`

`ironic.`

Execute a clea or deploy step asyn chro on the agent.

Paramet

- **tas** a Task agen object

containing the node

- `ste` a step dictionary to execute
- `ste` clea or deploy
- `cli` agen clien (if avai able

Raises
Nod Clea ing-Failure (clea step) or Instan ploy Failure

(deploy step) if the agent does not return a command status.

Returns
state to signify

the step will be com plet asyn

`ironic.`

Find the give in-banc step

`ironic.`

Get the list of cach clea or de-ploy step from the agen

The step cach is up-date at the be-gin-ning of clea ing or deploy.

Paramet

-

- **`tas`** a Task agen object containing the node
- **`ste`** clea or deploy
- **`int`** The interface for whic clea step are to be returned. If this is not provided, it returns the steps for all interfaces.
- **`ove`** a dictionary with keys being step nam and values being new priorities for them. If a step isnt in this dictionary, the steps original priority is used.

Returns
A

list of clea step dic- tio- nar- ies

`ironic.`

Hel meth to log the er- ror and rais ex- cep- tion

Paramet

- **tas** a Tas agen in- stan con- tain- ing the nod to act

on.

- **msg** the mes- sage to set

in last_ of the node

- **col** Boo indicating whe to attempt to collect

logs from IPA-based ramdisk. Defaults to True. Actual log collection is also affected by CONF.agent.deploy_logs_collect config option.

- **exc** Exception that caus the failure.

`ironic.`

Decorator meth for addi a post clea step hool

This is a mec

anism for addi a post clea step hook for a particular clean step. The hook will get executed after the clean step gets executed successfully. The hook is not invoked on failure of the clean step.

Any meth to be mad as a hook may be decorate with @post_clean_step_hook mentioning the interface and step after which the hook should be executed. A TaskManager instance and the object for the last completed command (provided by agent) will be passed to the hook method. The return value of this method will be ignored. Any exception raised by this method will be treated as a failure of the clean step and the node will be moved to CLEANFAIL state.

Paramet

- **`int`** nam of the interface
- **`ste`** The nam of the step af-

ter
whic
it
shou
be
ex-
ecuted.

Returns
A
metl
whic
reg-
is-
ters
the
give
metl
as
a
post
clea
step hook.

`ironic.`

Dec-
o-
ra-
tor
metl
for
addi
a
post
de-
ploy
step
hool

This
is
a
mec
a-
nism
for
addi
a
post
de-
ploy
step

hool

for a particular deploy step. The hook will get executed after the deploy step gets executed successfully. The hook is not invoked on failure of the deploy step.

Any metl to be mad as a hool may be decoratec with

@post_deploy_step_hook mentioning the interface and step after which the hook should be executed. A TaskManager instance and the object for the last completed command (provided by agent) will be passed to the hook method. The return value of this method will be ignored. Any exception raised by this method will be treated as a failure of the deploy step and the node will be moved to DEPLOYFAIL state.

Paramet

- **int** nam of the interface
- **ste** The nam of the step after whic it shou be executed.

Returns
A

meth
whic
registers
the
give
meth
as
a
post
de-

ploy step hook.

ironic.drivers.modules.agent_client module

class i
Base
obj

Clie
for
interacting
with
node
via
a
RES
API

collect
Collect
and
pack
age
diagnostic
and
support
data

from the ramdisk.

Parame
nod

A Nod ob- ject.

Raises
Iron icEx cep- tion whe faile to is- sue the re- ques or

there was a malformed response from the agent.

Raises
Age tAPI ror whe agen faile to ex- e- cute spec i- fied

command.

Raises
Age Prog whe the com man fails to ex- e- cute as the

agent is presently executing the prior command.

Returns
A dict containing command response from agent. See *get*

for a command result sample.

execute
Execute specified clean step

Parame

- **ste** A clean step dictionary to execute
- **nod** A Node object.
- **por** Port associated

ciated with the nod

Raises
Iron icEx ception whe faile to issue the request or

there was a malformed response from the agent.

Raises
Age tAPI ror whe agen faile to execute spec ified

command.

Raises
Age Prog whe the command fails to execute as the

agent is presently executing the prior command.

Returns

A dict containing command response from agent. See *get* for

a command result sample. The value of key command_result is in the form of:

```
↪,
```

execute

Execute spec ified deploy step

Parame

- **ste** A deploy step dictionary to execute
- **nod** A Nod object.
- **por** Port associated with the node

Raises
Iron icEx ception whe faile to issue the

re-
ques
or

there was a malformed response from the agent.

Raises
Age
tAPI
ror
whe
agen
faile
to
ex-
e-
cute
spec
i-
fied

command.

Raises
Age
Prog
whe
the
com
man
fails
to
ex-
e-
cute
as
the

agent is presently executing the prior command.

Returns

A
dict
con-
tain-
ing
com
man
re-
spon
from
agen
See
get

for

a command result sample. The value of key command_result is in the form of:

```
↪,
```

`finaliz`

In-struc the rame to fi-nal-ize en-ter-ing of res-cue mode.

Paramet

`nod`

A

Nod ob- ject.

Raises

Iron icEx cep- tion if res- cue_ is miss ing, or whe faile

to issue the request, or there was a malformed response from the agent.

Raises

Age tAPI ror whe agen faile to ex- e- cute spec i- fied

command.

Raises

Age Prog whe the com man fails to ex- e- cute as the

agent is presently executing the prior command.

Raises

In-
stan
cue-
Fail-
ure
whe
the
agen
ram
is
too
old
to

support transmission of the rescue password.

Returns
A
dict
con-
tain-
ing
com
man
re-
spon
from
agen
See
get

for a command result sample.

get_cle
Get
clea
step
from
agen

Parame

- **nod**
A
node
ob-
ject.
- **por**
Port
as-
so-
ci-

ated with the node

Raises

Iron icEx cep- tion whe faile to is- sue the re- ques or

there was a malformed response from the agent.

Raises

Age tAPI ror whe agen faile to ex- e- cute spec i- fied

command.

Raises

Age Prog whe the com man fails to ex- e- cute as the

agent is presently executing the prior command.

Returns

A
dict
containing
com
man
re-
spor
from
ager
See
get
for

a command result sample. The value of key command_result is in the form of:

get_con

Get
com
man
status

from
agen

Parame

- **nod**
  A
  Nod
  ob-
  ject.
- **ret**
  Whe
  to
  retry
  con-
  nec-
  tion
  prob
  lems
- **exp**
  If
  True
  do
  not
  log
  con-
  nec-
  tion
  prob
  lems
  as
  er-

rors.

Returns

A
list
of
com
man
re-
sults
each
re-
sult
is

related to

a command been issued to agent. A typical result can be:

```
↪
```

(continues on next page)

```
↪       e.g. 'RUNNING', 'SUCCEEDED', 'FAILED'>,
```

(continued from previous page)

```
↪     failed>,

↪      succeeded, the value is command specific,
```

(continues on next page)

(continued from previous page)

```
↪       * a dictionary containing keys clean_result

↪         and clean_step for the command

↪         clean.execute_clean_step;

↪       * a dictionary containing keys deploy_result
```

(continues on next page)

(continued from previous page)

```
↪         and deploy_step for the command
↪         deploy.execute_deploy_step;
↪       * a string representing result message for
↪         the command standby.cache_image;
```

(continues on next page)

(continued from previous page)

```
↪       * None for the command standby.sync.>
```

get_dep

Get de- ploy step fron ager

Parame

- **nod** A node ob- ject.
- **por** Port as- so- ci- ated with the node

Raises

Iron icEx cep- tion whe faile

to issue the reques or

there was a malformed response from the agent.

Raises

Age tAPI ror whe agen faile to execute speci fied

command.

Returns

A dict containing com man response fron agen See *get* for

a command result sample. The value of key command_result is in the form of:

(continues on next page)

(continued from previous page)

get_las

Get the last status for the given com man

Parame

- **nod** A Nod object.
- **met** Com man nam

Returns

A dict containing com man status from agen or Non

if the command was not found.

get_pa

Get deploy step from agen

Parame

nod

A nod object.

Raises

Ironic IcEx cep tion whe faile to issue the request or

there was a malformed response from the agent.

Raises

Age tAP ror whe agen faile to execute spec ified

command.

Raises

Age Prog

whe the com man fails to execute as the

agent is presently executing the prior command.

Returns
A dict containing com man response from agen

install

Installs a boot load on the image.

Parame

- **nod** A node object.
- **roo** The UUI of

the root partition.

- **tar** The target deployment boot mod
- **efi** The UUI of the efi system partition whe the

bootloader will be installed to, only used for uefi boot mode.

- **pre** The UUI of the PRe Boo partition whe the boot

loader will be installed to when local booting a partition image on a ppc64* system.

Raises
Iron icEx cep-

tion whe faile to issue the reques or

there was a malformed response from the agent.

Raises

Age tAPI ror whe agen faile to execute spec ified

command.

Raises

Age Prog whe the com man fails to execute as the

agent is presently executing the prior command.

Returns

A dict containing com man re-

spor
from
ager
See
get

for a command result sample.

power_o

Soft
pow
ers
off
the
bare
meta
node
by
shut
ting
dow
ramo

OS.

Paramet

nod

A
Nod
ob-
ject.

Raises

Iron
icEx
cep-
tion
whe
faile
to
is-
sue
the
re-
ques
or

there was a malformed response from the agent.

Raises

Age
tAPI
ror
whe
agen
faile

to execute spec i- fied

command.

Raises

Age Prog whe the com man fails to execute as the

agent is presently executing the prior command.

Returns

A dict containing com man response from agen See *get*

for a command result sample.

prepare

Call the *prepare* meth on the node

Parame

- **nod** A Nod object.
- **ima** A dictionary containing various image

related information.

- **wai** True to wait for the command to finish exe-

cuting, False otherwise.

Raises
Iron icEx ception whe faile to issue the request or

there was a malformed response from the agent.

Raises

Age tAPI ror whe agen faile to execute spec i- fied

command.

Raises

Age Prog whe the com man fails to execute as the

agent is presently executing the prior command.

Returns

A dict containing com man status from agen See *get*

for a command result sample.

reboot

Soft re-

boot
the
bare
meta
node
by
shut
ting
dow
ram
OS.

Parame
nod
A
Nod
ob-
ject.

Raises
Iron
icEx
cep-
tion
whe
faile
to
is-
sue
the
re-
ques
or

there was a malformed response from the agent.

Raises
Age
tAPI
ror
whe
agen
faile
to
ex-
e-
cute
spec
i-
fied

command.

Raises
Age

Prog whe the com man fails to ex- e- cute as the

agent is presently executing the prior command.

Returns

A dict con- tain- ing com man re- spon from agen See `get`

for a command result sample.

start_i

Ex- pose the node disk as an ISCS tar- get.

Paramet

- **nod** an Iron node ob- ject
-

iqn iSCS target IQN

- **por** iSCS portal port

- **wip** True if the agen shou wipe first the disk mag strin like

the partition table, RAID or filesystem signature.

Raises
Iron icEx ception whe faile to issue the request or

there was a malformed response from the agent.

Raises
Age tAPI ror whe agen faile to

execute spec ified command.

Raises
Age Prog whe the com man fails to execute as the agent is presently executing the prior command.

Returns
A dict containing com man respor from agen See *get* for a command result sample.

sync (*no*
Flus file system buff forcing char bloc to disk

Parame

`nod`

A Nod object.

Raises

Iron icEx ception whe faile to issue the reques or there was a malformed response from the agent.

Raises

Age tAP ror whe agen faile to execute spec ified command.

Raises

Age Prog whe the com man fails to execute as the

agent is presently executing the prior command.

Returns
A dict containing command response from agen See
get

for a command result sample.

`ironic.`
Get clien for this node

`ironic.`
Extract an error strin from the command result.

Paramet
`com`
Command information from the agen

Returns
Error

strin

ironic.drivers.modules.agent_power module

The agen pow in- ter- face

class i

Base *irc dri bas Pow*

Pow in- ter- face us- ing the run- ning agen for pow ac- tions

get_pow

Re- turn the pow state of the task node

Es- sen- tially the only know

state is POW ON, everything else

is an error (or more precisely `None`).

Paramet

`tas`

A Task agen instance containing the node to act

on.

Returns

A pow state One of *`iro com sta`*

`get_pro`

Return the properties of the interface

Returns

dictionary

of <pro erty nam description: entries

get_sup

Get a list of the supporte pow state

Only contains REBOO

Paramet

tas A Task agen instance containing the node to act

on.

Returns

A list with the supporte pow state defined in *iro* *com*

states.

reboot

Perform a reboot of the task node

Only soft reboot is implemen

Paramet

- **tas** A Task agen instanc containing the node to act

on.

- **tim** timeout (in seconds positive integer (>

0) for any power state. `None` indicates to use default timeout.

set_pow

Set the pow state of the task node

Paramet

- **tas** A Task agen instan containing the node to act

on.

- **pow** Pow state from *irc com sta* Only RE-BOO and SOF are

supported and are synonymous.

- **tim**

time out (in seconds pos- i- tive in- te- ger (>

0) for any power state. `None` indicates to use default timeout.

Raises

Pow- er- Stat Fail- ure on non- supp pow state

support

Che if pow sync is sup- port for the give node

Not sup- port for the agen pow sinc it is not pos- si- ble to power on/off nodes.

Paramet

tas

A Task ager in- stanc con- tain- ing the node to act

on with a **shared** lock.

Returns

bool whe pow sync is sup- porte

validat

Val- i- date the drive spec Nod de- ploy men info

Paramet

tas

A Task ager in- stanc con- tain- ing the node to act on.

Raises
In- valic Pa- ram- e- ter- Valu on mal- form pa- ram- e- ter(s)

ironic.drivers.modules.boot_mode_utils module

`ironic.`
Con fig- ures se- cure boot if it has been re- ques for the node.

`ironic.`
De- con- fig- ures se- cure boot if it has been re- ques

for the node.

`ironic.`

Re-
turn
the
boot
mod

Paramet
nod
an
iron
node
ob-
ject.

Returns
bios
or
uefi

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
the
node
boot
mod
dis-

agrees with the boot mode set to node properties/capabilities

`ironic.`

Re-
turn
the
boot
mod
that
wou
be
used
for
de-
ploy

This
meth

returns boot mod to be used for deploy It returns uefi if secure_boot is set to true or returns bios if trusted_boot is set to true in instance_info/capabilities of node. Otherwise it returns value of boot_mode in properties/capabilities of node if set. If that is not set, it returns boot mode in internal_driver_info/deploy_boot_mode for the node. If that is not set, it returns boot mode in instance_info/deploy_boot_mode for the node. It would return None if boot mode is present neither in capabilities of node properties nor in nodes internal_driver_info nor in nodes instance_info (which could also be None).

Paramet

`nod` an ironi node object.

Returns

bios uefi or Non

Raises

InvalidParameterValu if the node boot mod disagrees with the boot mode set to node properties/capabilities

`ironic.`

Returns True

if se- cure is re- ques for de- ploy

This meth chec node prop- erty for se- cure and re- turns True if

it is requested.

Paramet

`nod` a single Nod

Raises

Invalid Parameter-Valu if the capabili-

ties string is not a dictionary or is malformed.

Returns

True if secure is requested.

`ironic.`

Returns True if trust is requested for deploy

This method checks instance property for trust and returns True if

it is requested.

Paramet

nod a single Nod

Raises

InvalidParameterValu if the capa-

bil-
i-

ties string is not a dictionary or is malformed.

Returns

True
if
trust
is
re-
ques

`ironic.`

Set
node
boot
mod
from
bare
meta
con-
fig-
u-
ra-
tion

At-
temp
to
read
cur-
rentl
set
boot
mod
off
the
bare
meta
ma-

chine. Also read nodes boot mode configuration:

- If
BM
drive
does
not
im-
ple-
men
get-
ting

boot mod assume BM boot mode is not set and apply the logic that follows

- If Iron node boot mod is not set and BM node boot mod is not set - set Ironic boot mode to *[deploy]/default_boot_mode*
- If Iron node boot mod is not set and BM node boot mod is set - set BM node boot mode on the Ironic node
- If Iron node boot mod is set and BM node boot mod is not set - set Ironic boot mode to BM boot mode

- If both Iron and BM node boot mod are set but they differ - try to set Ironic boot mode to BM boot mode and fail hard if underlying hardware type does not support setting boot mode

  In the end, the new boot mod may be set in driv

Paramet
: **tas** a task object

ironic.drivers.modules.console_utils module

Iron console utilities.

`ironic.`
: Returns a free

TCP port on current host

Find and returns a free TCP port in the range of CON

ironic.

Get a url to access the console via shellinab

Paramet

por

the terminal port for the node

ironic.

Get a URL to access

the console via socat.

Paramet

por

the terminal port (integer) for the node

Returns

an access URI to the socat console of the node

`ironic.`

Writ a file containing a pass word until dele

`ironic.`

Re-
lease
spec
i-
fied
TCP
port

`ironic.`

Ope
the
se-
rial
con-
sole
for
a
node

Paramet

- **nod**
the
uuid
for
the
node
- **por**
the
ter-
mi-
nal
port
for
the
node
- **con**
the
shell
com
man
that
gets
the
con-

sole

Raises

Con sole ror if the directory for the PID file can-

not be created or an old process cannot be stopped.

Raises

Con sole-Subprocess Fail whe invoking the subpro-

cess failed.

`ironic.`

Ope the serial console for a node

Paramet

- **nod** the uuid

of the node

- **por** the terminal port for the node
- **con** the shell command that will be executed by so-

cat to establish console to the node

Raises

- *Con* if the directory for the PID file or the PID

file cannot be created

- *Con* whe invoking the subprocess faile

`ironic.`

Clos the serial console for a node

Paramet

`nod` the UUI of the node

Raises

Con soleE ror if unable to stop the console process

`ironic.`

Clos the serial console for a node

Paramet

nod the UUI of the node

Raises

Con if unable to stop the console process

ironic.drivers.modules.deploy_utils module

class i

Base *iro dri moo ima Ima*

ironic.

Add require config parameters to node drive

Add the require conf options to

node
driv
It is Requir

to pass the information to IPA.

Paramet

`tas`
a Task agen instan

`ironic.`

Buil the options to be pass to the agen ram

Paramet

`nod`
an ironi node object

Returns
a dictionary containing the parameters to

be passed to agent ramdisk.

`ironic.`

Buil in- stan nec- es- sary for de- ploy ing to a node

Paramet

`tas` a Task agen ob- ject con- tain- ing the node

Returns

a dic- tio- nary con- tain- ing the prop er- ties to be

updated in instance_info

Raises

ex- cep- tion. if im- age_ is not Glan

href and is not

HTTP(S) URL.

ironic.

Fetc the in- stan im- age from Glan

This meth pulls the disk im- age and write them to the ap- pro-

priate place on local disk.

Paramet

- **ctx** con- text
- **nod** an iron node ob- ject

- **for** whe con- vert im- age to raw for- mat
- **exp** The ex- pect for- mat of the disk im- age con- tents
- **exp** The ex- pect im- age chec sum to be used if we

need to convert the image to raw prior to deploying.

- **exp** The chec sum algo in use, if sep-

a-
ratel
set.

Returns
a
tu-
ple
con-
tain-
ing
the
uuid
of
the
im-
age
and

the path in the filesystem where image is cached.

Raises
In-
va-
lidIr
age
if
the
re-
ques
im-
age
is
in-
valic

and cannot be used for deployed based upon contents of the image or the metadata surrounding the image not matching the configured image.

`ironic.`

Chec
for
emp
para
in
the
pro-
vide
dic-
tio-
nary

Paramet

- **inf** The dictionary to inspec
- **err** The error message to prefix before print ing

the information about missing parameters.

- **par** Add this prefix to each parameter for er-

ror messages

Raises

Miss ing-Parameter-Valu

if one or more param-

eters are empty in the provided dictionary.

`ironic.`

Evaluate interface to determine if capa-

bility is present.

Paramet

- `int` The interface object to chec
- `cap` The value representing the capa-

bility that

the caller wishes to check if present.

Returns

True if capability foun otherwise Fals

`ironic.`

Com puted checsum by give image path and algorithm

`ironic.`

Dele instanc image file and symbolic link refer to it.

`ironic.`

Dele in-

stan
im-
age
file.

Paramet

nod
the
uuid
of
the
iron
node

`ironic.`

Whe
con-
vert
im-
age
to
raw
for-
mat
for
spec
i-
fied

node.

Paramet

nod
iron
node
ob-
ject

Returns

Boo
whe
the
di-
rect
de-
ploy
in-
ter-
face
shou
con-
vert

image to raw.

ironic.

Che
for
avai
able
disk
spac
and
fetcl
im-
ages
us-
ing
Im-
ageCache.

Paramet

- **ctx**
  con-
  text
- **cac**
  Im-
  age-
  Cacl
  in-
  stan
  to
  use
  for
  fetcl
  ing
- **ima**
  list
  of
  tu-
  ples
  (im-
  age
  href

destination path

- **for** bool value whether to convert the image to raw format

mat

- **exp** The expect format of the image.
- **exp** The expect image chec sum to be used if we

need to convert the image to raw prior to deploying.

- **exp** The

checsum algo in use, if separately set.

Raises
Instanc ploy Failure if unable to find enou disk spac

Raises
InvalidIr age if the supplied image meta data or

contents are deemed to be invalid, unsafe, or not matching the expectations asserted by configuration supplied or set.

`ironic.`
Returns state base on operation

(clea ing/ being

invoked

Paramet
nod an ironi node object.

Returns
state if clean ing operation in prog or state if

deploy operation in progress.

`ironic.`

Gets the boot option.

Paramet
nod A single Nod

Raises
InvalidParameterValu if the

capabili-

ties string is not a dict or is malformed.

Returns

A strin rep-re-sent-ing the boot op-tion type De-fault

to netboot.

`ironic.`

Gets the de-fault boot op-tion.

`ironic.`

Re-turn the disk la-bel re-ques for de-ploy if any.

Paramet

`nod`

a sin-gle Nod

Raises

In-
valic
Pa-
ram-
e-
ter-
Valu
if
the
ca-
pa-
bil-
i-
ties string is not a dictionary or is malformed.

Returns
the
disk
la-
bel
or
Non
if
no
disk
la-
bel
was
spec
ified.

`ironic.`
Get
the
ef-
fec-
tive
valu
of
im-
age_
for
the
node

`ironic.`
Gets
the
im-
age
in-
for-
ma-

tion from the node

Get image information for the given node instance from

its instance_info property.

Paramet

`nod`

a single Nod

Returns

A dict with required image properties retrieved from

nodes instance_info.

Raises

Missing-Parameter-Valu if

image_
is
miss
ing

in nodes instance_info. Also raises same exception if kernel/ramdisk is missing in instance_info for non-glance images.

`ironic.`

Return the iPXE boot file name requested for deploy

This method returns iPXE boot file name to be used for deploy

Architecture specific boot file is searched first. BIOS/UEFI boot file is used if no valid architecture specific file found.

If no valid value is found the default reverts to the `get` method and thus the `[pxe]pxe_bootfile_name` and `[pxe]uefi_ipxe_bootfile_name` settings.

Paramet

nod
A single Nod

Returns
The iPXI boot file nam

`ironic.`

Return the iPXI config template file nam request of

deploy.

This meth returns the iPXI configuration template file.

Paramet

nod
A single Nod

Returns

The iPXI config template file nam

`ironic.`

Resolv Iron API endpoin

either from config of from Keystone catalog.

`ironic.`

Return the PXE boot file nam requests for deploy

This meth returns PXE boot file nam

to be used for deploy

Architecture specific boot file is searched first. BIOS/UEFI boot file is used if no valid architecture specific file found.

Paramet

nod

A single Nod

Returns

The PXE boot file nam

`ironic.`

Return the PXE config template file nam requested for

deploy.

This meth returns PXE config template file to be used for deploy. First specific pxe template is searched in the node. After that architecture specific template file is searched. BIOS/UEFI template file is used if no valid architecture specific file found.

Paramet

nod

A single Nod

Returns

The PXE config template file nam

`ironic.`

Identify a boot volume from any configured volume

Returns

Non or the volume target representing the volume.

`ironic.`

Get a

root de- vice re- ques for de- ploy men or Non

Raises

In- valic Pa- ram- e- ter- Valu on in- valic hints

Returns

Pars root de- vice hints or Non if no hints were pro- vide

`ironic.`

Re- turn the MA(ad- dres of a port whic has a

VIF

port id.

Paramet

tas

a Task ager instanc containing the port to act

on.

Returns

MA address of the port connect to deployment net-

work. None if it cannot find any port with vif id.

`ironic.`

Determine if Ana conc deploy interface is in

use for the deployment.

Paramet

nod

A single Nod

Returns

A bool valu of True whe Ana cond deploy interface

is in use otherwise False

ironic.

Return true if boot ing from an iscsi volume

ironic.

ironic.

Determine if software raid is in use for the deployment.

Paramet

nod
A single Nod

Returns
A bool valu of True whe software raid is in use, otherwise False

ironic.

Gets the instanc specific Nod deployment info

This meth validates whe the instanc property of the supplied node contains the required information for this driver to deploy images to the node.

Paramet
nod

a single Nod

Returns
A dict with the instanc values.

Raises
Missing-Param-eter-Valu if any of the require

parameters are missing.

Raises
In-valid Param-eter-Valu if any of the param-

eters have invalid value.

`ironic.`

Set node drive for boot

from volume parameters.

Paramet

tas

a Task agen object containing the node

Raises

Stor ageE ror whe a node has an iSCS or FibreC nel

boot volume defined but is not capable to support it.

`ironic.`

Prepare booting the agen on the node

Paramet

tas

a Task agen

in-
stan

ironic.

Pre-
pare
the
node
to
boot
into
agen
for
in-
banc
clea
ing.

This
meth
does
the
fol-
low-
ing:
1.
Pre-
pare
the
clea
ing
port

for the bare metal node and updates the clean parameters in nodes driver_internal_info. 2. If manage_boot parameter is set to true, it also calls the prepare_ramdisk method of boot interface to boot the agent ramdisk. 3. Reboots the bare metal node.

Paramet

- **tas**
a
Task
agen
ob-
ject
con-
tain-
ing
the
node

- **`man`** If this is set to True this meth calls the prepare

method of boot interface to boot the agent ramdisk. If False, it skips preparing the boot agent ramdisk using boot interface, and assumes that the environment is setup to automatically boot agent ramdisk every time bare metal node is rebooted.

Returns
state to signify an asyn chro prepare

Raises
Network Error, Nod Clea ing-Fail ure if the previous cleaning ports cannot be removed or if new cleaning ports cannot be created.

Raises
Invali Parameter-Valu

if clear ing net-worl UUI

config option has an invalid value.

`ironic.`

Re-boot the node into IPA to fin-ish a de-ploy step

Paramet

`tas` a Task agen in-stan

Returns

state if clear ing op-er-a-tion in prog or state if

deploy operation in progress.

`ironic.`

`ironic.`

ironic.

Sets appropriate reboot flags in drive base on operation

Paramet

- **nod** an ironi node object.
- **reb** Boo valu to set for node drive flag clear ing_ or deployment_reboot based on cleaning or deployment operation in progress. If it is None, corresponding reboot flag is not set in nodes driver_internal_info.
- **ski** Boo valu to set

for node driv flag skip or skip base

on cleaning or deployment operation in progress. If it is None, corresponding skip step flag is not set in nodes driver_internal_info.

- **pol** Boo valu to set for node driv flag deploy men or

cleaning_polling. If it is None, the corresponding polling flag is not set in the nodes driver_internal_info.

`ironic.`

Sets the deploy status as faile with relevant messages.

This meth sets the deploy men as fail

with the give mes- sage

It sets nodes provision_state to DEPLOYFAIL and updates last_error with the given error message. It also powers off the baremetal node.

Paramet

- **tas** a Task ager in- stanc con- tain- ing the node to act

on.

- **msg** the mes- sage to set in logs and last_ of the node
- **col** Boo in- di- cat- ing whe to at- temp

to col- lect

logs from IPA-based ramdisk. Defaults to True. Actual log collection is also affected by CONF.agent.deploy_logs_collect config option.

`ironic.`

Swi a pxe con- fig from de- ploy men mod to ser- vice

mode.

Paramet

- **pat** path to the pxe con- fig file in tftp- boot
- **roo** root uuid in case

of partition image or disk

in case of whole disk image.

- **boo**
if boot mod is uefi or bios

- **is_**
if the image is a who disk image or not.

- **tru**
if boot with trust or not. The usage of is_w and

trusted_boot are mutually exclusive. You can have one or neither, but not both.

- **isc**

if boot is from an iSCS vol- ume or not.

- **ram** if the boot is to be to a ramd con- fig- u- ration.

- **ipx** A de- fault Fals bool valu to tell the meth if the caller is using iPXE.

- **ana** if the boot is to be to an

ana-
conc
con-
fig-

uration.

`ironic.`

Tear
dow
the
en-
vi-
ron-
men
setu
for
in-
banc
clea
ing.

This
meth
does
the
fol-
low-
ing:
1.
Pow
ers
off
the
bare
meta

node (unless the node is fast tracked or there was a cleaning failure). 2. If manage_boot parameter is set to true, it also calls the clean_up_ramdisk method of boot interface to clean up the environment that was set for booting agent ramdisk. 3. Deletes the cleaning ports which were setup as part of cleaning.

Paramet

- **tas**
a
Task
ager
ob-
ject
con-
tain-

ing the node

- **man** If this is set to True this meth calls the clea meth

of boot interface to boot the agent ramdisk. If False, it skips this step.

Raises
Network Error, Nod Clea ing Fail ure if the clea ing

ports cannot be removed.

`ironic.`
Clea up storage configuration.

Remov entries from drive

for storage and dele the volume

targets from the database. This is done to ensure a clean state for the next boot of the machine.

`ironic.`

Trie to set the boot device on the node

This meth tries to set the boot device on the node to the

given boot device. Under uefi boot mode, setting of boot device may differ between different machines. IPMI does not work for setting boot devices in uefi mode for certain machines. This method ignores the expected IPMI failure for uefi boot mode and just logs a message. In error cases, it is expected the operator has to manually set the node to boot from the correct device.

Paramet

- **`tas`** a Task ager object

containing the node

- **dev** the boot device
- **per** When to set the boot device persistent

Raises
Any exception from set_ exception IPMIFailure (setting of boot device using ipmi is expected to fail).

`ironic.`
Validate that specified supported ca-

pa-
bil-
i-

ties have valid value

This
meth
chec
if
the
any
of
the
sup-
port
ca-
pa-
bil-
ity

is present in Node capabilities. For all supported capabilities specified for a Node, it validates that it has a valid value. The node can have capability as part of the properties or instance_info or both. Note that the actual value of a capability does not need to be the same in the nodes properties and instance_info.

Paramet
nod
an
iron
node
ob-
ject.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
the
ca-
pa-
bil-
ity

is not set to a valid value.

`ironic.`

Validate the image.

For Glan images it chec that the image exists in Glan and its properties or deployment info contain the properties passed. If its not a Glance image, it checks that deployment info contains needed properties.

Paramet

- **`ctx`** security context
- **`dep`** the deploy to be validated
- **`pro`** the list of image meta

prop to be val- i- date

Raises
In- valid Pa- ram- e- ter- Valu if: * con- nec- tion to

glance failed; * authorization for accessing image failed; * HEAD request to image URL failed or returned response code != 200; * HEAD request response does not contain Content-Length header; * the protocol specified in image URL is not supported.

Raises
Miss ing- Pa- ram- e- ter- Valu if the im- age does con-

tain the mentioned properties.

ironic.drivers.modules.fake module

Fake driv in- ter- face used in testing.

This is also an example of som kind of thin whic can be done within drivers. For instance, the MultipleVendorInterface class demonstrates how to load more than one interface and wrap them in some logic to route incoming vendor_passthru requests appropriately. This can be useful eg. when mixing functionality between a power interface and a deploy interface, when both rely on separate vendor_passthru methods.

class i

Base *irc dri bas BIC*

Fake implemen tation of simple BIO Interface

apply_c

Validate & apply BIO settings

on the give node

This meth take the BIO settings from the settings para and applies BIOS settings on the given node. It may also validate the given bios settings before applying any settings and manage failures when setting an invalid BIOS config. In the case of needing password to update the BIOS config, it will be taken from the driver_info properties. After the BIOS configuration is done, cache_bios_settings will be called to update the nodes BIOS setting table with the BIOS configuration applied on the node.

Parame

- **tas** a Task agen instan
- **set** Dictona containing the BIO configuration.

Raises
Unsup-

portedDriv tension if the node driv does support BIOS configuration.

Raises
InvalidParameterValu if validation of settings fails.

Raises
MissingParameterValu if som required parameters are missing.

Returns
state if BIO configura-

tion is in prog asyn chro

or None if it is complete.

cache_b

Stor or up-date BIO prop-er-ties on the give node

This meth store BIO prop-er-ties to the bios ta-ble dur-ing

cleaning operation and updates bios_settings table when apply_configuration() and factory_reset() are called to set new BIOS configurations. It will also update the timestamp of each bios setting.

Paramet

tas a Task agen in-stan

Raises

Un-sup-port-ed-Driv

tension if the node drive does support getting BIOS properties from bare metal.

Returns
Non

`factory`
Reset BIO configuration to factory default on the given node.

This meth resets BIO configuration to factory default on the given node. After the BIOS reset action is done, cache_bios_settings will be called to update the nodes BIOS settings table with default bios settings.

Paramet
`tas` a Task ager instan

Raises
Unsupported-Driv ten-sion if the node driv does sup-

port BIOS reset.

Returns
state if BIO con-fig-u-ra-tion is in prog asyn chro

or None if it is complete.

get_pro
Re-turn the prop er-ties of the in-ter-face

Returns
dic-tio-nary of <pro erty nam

description: entries

validat

Validate the driver-specific Node deployment info.

This method validates whether the driver and/or instance properties of the tasks node contains the required information for this interface to function.

This method is often executed synchronously in API requests, so it should not conduct long-running checks.

Paramet

`tas`

A Task
ager
in-
stan
con-
tain-
ing
the
node
to
act
on.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
on
mal-
form
pa-
ram-
e-
ter(s)

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-
ter(s)

class i
Base
iro
dri
bas
Boc

Example implementation of a simple boot interface.

capabil

clean_u

Clea up the boot of instanc

This meth clea up the environmen that was setu for booting the instance.

Parame

tas A task from Task agen

Returns Non

clean_u

Clea up the boot of ironi ramd

This meth clea up the environmen that was setu for boot

ing the deploy or rescue ramdisk.

Parame

tas

A task from Task agen

Returns

Non

get_pro

Return the prop erties of the interface

Returns

dictionary of

<pro erty nam description entries

prepare

Prepare the boot of instanc

This meth prepares the boot of the instanc after reading relevant information from the nodes database.

Paramet

tas A task from Task agen

Returns

Non

prepare

Prepare the boot of Iron rame

This meth prepare the boot of the deploy or rescue ram after reading relevant information from the nodes database.

Paramet

- `tas` A task from Task agen
- `ram` The options to be pass to the ironi ram Different implementations might want to boot the ramdisk in different ways by passing parameters to them. For example,

Whe Age ram is boot to deploy a

node
it
take
the
pa-

rameters ipa-api-url, etc.

Othe
im-
ple-
men
ta-
tions
can
mak
use
of
ram
to
pass
such

information. Different implementations of boot interface will have different ways of passing parameters to the ramdisk.

Returns
Non

`validat`

Val-
i-
date
the
drive
spec
Nod
de-
ploy
men
info

This
meth
val-
i-
date
whe
the
drive
and/
in-
stan
prop
er-

ties of the tasks node contains the required information for this interface to function.

This meth is often execute synchro in API reques so it should not conduct long-running checks.

Parame

`tas` A Task agen instan containing the node to act on.

Raises

Invali Parameter Valu on malform parameter(s)

Raises

Miss

ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-

ter(s)

class i
Base
irc
dri
bas
Con

Ex-
am-
ple
im-
ple-
men
ta-
tion
of
a
sim-
ple
con-
sole

interface.

get_con
Get
con-
nec-
tion
in-
for-
ma-
tion
abou
the
con-
sole

This
meth

shou re- turn the nec- es- sary in- for- ma- tion for

the client to access the console.

Parame

`tas`

A Task ager in- stan con- tain- ing the node to act

on.

Returns

the con- sole con- nec- tion in- for- ma- tion.

`get_pro`

Re- turn the prop er- ties of the in- ter-

face

Returns
dictionary of <pro erty nam description en tries

start_c
Star a remote console for the task node

This meth shou not raise an exception if console already

started.

Parame
tas
A Task agen instanc contain-

ing the node to act on.

stop_co

Stop the remote console session for the task node

Parame

tas

A Task agen instance containing the node to act on.

validat

Validate the drive spec Node deployment info

This meth val-

i-
date
whe
the
driv
and/
in-
stan
prop
er-
ties

of the tasks node contains the required information for this interface to function.

This
meth
is
of-
ten
ex-
e-
cute
syn-
chro
in
API
re-
ques

so it should not conduct long-running checks.

Paramet

`tas`

A
Task
ager
in-
stan
con-
tain-
ing
the
node
to
act

on.

Raises

In-
valid
Pa-
ram-
e-
ter-
Valu

on malform parameter(s)

Raises

Missing-Parameter-Valu on missing parameter(s)

class i

Base

iro

dri

bas

Dep

Clas for a fake deploy men driv

Example implementation of a deploy in-

ter-

face that uses a separate power interface.

clean_u

Clea up the de- ploy men en- vi- ron- men for the task

node.

If prep ra- tion of the de- ploy men en- vi- ron- men ahea

of time is possible, this method should be implemented by the driver. It should erase anything cached by the *prepare* method.

If im- ple- men this meth mus be iden po- tent. It may be

called multiple times for the same node on the same conductor, and it may be called by multiple conductors in parallel. Therefore, it must not require an exclusive lock.

This meth is calle before *tear*

Parame

`tas`

A Task ager instan containing the node to act on.

`deploy`

Perform a deploy men to the task node

Perform the necessary work to deploy an image onto the specified node. This method will be called after prepare(), which may have already performed any preparatory steps, such as pre-caching some data for the node.

Paramet

tas

A Task ager in- stanc con- tain- ing the node to act

on.

Returns

sta- tus of the de- ploy One of ironi

get_pro

Re- turn the prop er- ties of the in- ter- face

Returns

dic- tio- nary of <pro erty nam de- scrip tion: en- tries

prepare

Prepare the deployment environment for the task's node.

If preparation of the deployment environment ahead of time is possible, this method should be implemented by the driver.

If implemented, this method must be idempotent. It may be called multiple times for the same node on the same conductor.

This method is called before *deploy*

Paramet

`tas`

A Task ager instan containing the node to act on.

`take_ov`

Take over man agemen of this task node from a dead conductor.

If conductors host mai tain a stati relationship to nodes, this method should be implemented by the driver to allow conductors to perform the necessary work during the remapping of nodes to conductors when a conductor joins or leaves the cluster.

For exar

Neu tron mus forwarc DHC BOC requesl to a conductor which has prepared the tftpboot environment for the given node. When a conductor goes offline, another conductor must change this setting in Neutron as part of remapping that nodes control to itself. This is performed within the *takeover* method.

Paramet

`tas`

A Task ager instanc containing the node to act on.

`tear_dc`

Tear dow a previous deploy men on the task node

Give a node

that has been previously deployed to, do all cleanup and tear down necessary to un-deploy that node.

Paramet

tas

A Task ager instance containing the node to act on.

Returns

status of the deploy. One of ironi

validat

Validate the driver spec Node deployment info

This meth val- i- date whe the driv and/ in- stan prop er- ties

of the tasks node contains the required information for this interface to function.

This meth is of- ten ex- e- cute syn- chro in API re- ques

so it should not conduct long-running checks.

Parame

tas

A Task agen in- stan con- tain- ing the node to act

on.

Raises

In- valid Pa- rameterValu on malform parameter(s)

Raises

MissingParameterValu on missing parameter(s)

class i

Base

iro

dri

bas

Ins

Example implementation of a simple inspec interface.

get_pro

Return

the prop- erties of the interface

Returns
dictionary of <pro erty nam description entries

inspect
Inspec hard ware

Inspec hard ware to obtain the essential & additional hardware properties.

Paramet
tas
A task from Task agen

Raises

Hard-
ware-
spec-
tion-
Fail-
ure,
if
un-
able
to
get
es-
sen-

tial hardware properties.

Returns

Re-
sult-
ing
state
of
the
in-
spec-
tion
i.e.
state
or
Non

validat

Val-
i-
date
the
driv
spec
Nod
de-
ploy
men
info

This
meth
val-
i-
date
whe
the
driv
and/

instance properties of the tasks node contains the required information for this interface to function.

This method is often executed synchronously in API requests so it should not conduct long-running checks.

Parame

`tas` A TaskManager instance containing the node to act on.

Raises

InvalidParameterValu on malformed parame-

ter(s)

Raises

Miss ing-Pa-ram-e-ter-Valu on miss ing pa-ram-e-

ter(s)

class i

Base

irc

dri

bas

Man

Ex-am-ple im-ple-men ta-tion of a sim-ple man age-

ment interface.

get_boo

Get the cur-rent boot de-vice for a node

Pro-

vide
the
current
boot
device
of
the
node
Be
awa
that

not all drivers support this.

Parame

`tas`

A
task
from
Task
ager

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing

Returns

A
dic-
tio-
nary
con-
tain-
ing:

boot_

Ahe
boot
de-

vice one of *iro com boo* or Non if it

is unknown.

persist

Whe the boot de- vice will per- sist to all fu- ture boot

or not, None if it is unknown.

`get_inc`

Get cur- rent state of the in- di- ca- tor of the hard

ware component.

Paramet

- **`tas`** A task from Task

ager

- **com** The hardware component, one of *iro com com*
- **ind** Indicator ID (as reported by *get_*

Raises Invalid Parameter Valu if an invalid compo-

nent or indicator is specified.

Raises Missing Parameter Valu

if a required param-

eter is missing

Returns
Current state of the indicator, one of *iro com*

indicator_states.

get_pro
Return the properties of the interface

Returns
dictionary of <pro erty nam description entries

get_ser
Get

sen-
sors
data
meth

Parame
tas
A
Task
agen
in-
stan

Raises
Faile
To-
Get-
Sen-
sor-
Data
whe
get-
ting
the
sen-
sor
data

fails.

Raises
Faile
ToPa
eSen
sor-
Data
whe
pars
ing
sen-
sor
data
fails

Returns

Re-
turns
a
con-
sis-
tent
for-
mat

dict
of
sen-
sor
data
grou

by sensor type, which can be processed by Ceilometer. eg,

```
↪
```

(continues on next page)

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

(continues on next page)

↪

(continued from previous page)

```
```

(continues on next page)

↪

(continued from previous page)

get_sup

Get a list of the supporte boot device

Paramet

tas

A task from

Task ager

Returns
A list with the sup- port boot de- vice de- fined in *iro*

common.boot_devices.

get_sup
Get a map of the sup- port in- di- ca- tors (e.g. LED

Paramet

- **tas** A task from Task ager
- **com** If not *Non* re- turn in- di-

cator information for just this component, otherwise return indicators for all existing components.

Returns

A dictionary of hard ware com ponent (*ir com com* as keys with values being dictionaries having indicator IDs as keys and indicator properties as values.

(continues on next page)

(continued from previous page)

```
↪": [
↪   "off",
↪   "on"
```

(continues on next page)

(continued from previous page)

```
↪": true,
```

(continues on next page)

(continued from previous page)

```
↪   "pff",







↪   "on"
```

(continues on next page)

(continued from previous page)

```
↪": true,

↪": [
```

(continues on next page)

(continued from previous page)

```
↪  "on"
```

set_boo

Set the boot de- vice for a

node

Set the boot device to use on next reboot of the node

Paramet

- `tas` A task from Task agen
- `dev` The boot device one of *`iro com boo`*
- `per` Boo valu True if the boot device will persist to

all future boots, False if not. Default: False.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
an
in-
valid
boot
de-

vice is specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quire
pa-
ram-

eter is missing

validat
Val-
i-
date
the
drive
spec
Nod
de-
ploy
men
info

This
meth
val-
i-
date
whe

the driv and/ instan prop erties of the tasks node contains the required information for this interface to function.

This meth is often execute synchro in API reques so it should not conduct long-running checks.

Parame

`tas` A Task ager instan containing the node to act on.

Raises

Invalic Parameter-Valu on malform

parameter(s)

Raises

Miss ing-Parameter-Valu on miss ing parameter(s)

class i

Base

irc dri bas Pow

Example implementation of a simple pow interface.

get_pow

Return the pow state of the task

node

Paramet

tas

A Task ager in- stand con- tain- ing the node to act

on.

Raises

Miss ing- Pa- ram- e- ter- Valu if a re- quire pa- ram-

eter is missing.

Returns

A pow state One of *iro com sta*

get_pro

Re- turn the prop er- ties of the

interface

Returns

dictionary of <property name: description> entries

get_sup

Get a list of the supported power state

Paramet

tas

A Task instance containing the node to act

on.

Returns

A list with the supported pow

state de- fined in *iro com*

states.

reboot

Per- form a hard re- boot of the task node

Driv are ex- pect to prop erly han- dle case whe node is pow

ered off by powering it on.

Parame

- **tas** A Task agen in- stanc con- tain- ing the node to act

on.

- **tim** timeout (in seconds positive integer (>

0) for any power state. `None` indicates to use default timeout.

Raises
Missing-ParameterValu if a require param-

eter is missing.

set_pow
Set the pow state of the task node

Paramet

- **tas** A Taskager instanc

containing the node to act on.

- **pow** Any power state from *iro com sta*
- **tim** timeout (in seconds) positive integer (> 0) for any power state. `None` indicates to use default timeout.

Raises
MissingParameterValue if a required parameter is missing.

validat
Val-

i-
date
the
driv
spec
Nod
de-
ploy
men
info

This
meth
val-
i-
date
whe
the
driv
and/
in-
stan
prop
er-
ties

of the tasks node contains the required information for this interface to function.

This
meth
is
of-
ten
ex-
e-
cute
syn-
chro
in
API
re-
ques

so it should not conduct long-running checks.

Parame

`tas`

A
Task
agen
in-
stan
con-
tain-
ing

the node to act on.

Raises
In- valid Pa- ram- e- ter- Valu on mal- form pa- ram- e- ter(s)

Raises
Miss ing- Pa- ram- e- ter- Valu on miss ing pa- ram- e- ter(s)

class i
Base
iro dri bas RAI

Ex- am- ple im- ple- men ta- tion

of simple RAI In- ter-

face.

create_

Cre- ates RAI con- fig- u- ra- tion on the give node

This meth cre- ates a RAI con- fig- u- ra- tion on the give

node. It assumes that the target RAID configuration is already available in node.target_raid_config. Implementations of this interface are supposed to read the RAID configuration from node.target_raid_config. After the RAID configuration is done (either in this method OR in a call-back method), ironic.common.raid.update_raid_info() may be called to sync the nodes RAID-related information with the RAID configuration applied on the node.

Parame

- **tas** A Task agen in- stan

- **`cre`** Setting this to Fals indicate not to create root volume that is specified in the nodes target_raid_config. Default value is True.
- **`cre`** Setting this to Fals indicate not to create non-root volumes (all except the root volume) in the nodes target_raid_config. Default value is True.
- **`del`** Setting this to True indicate to dele RAI configuration prior to creating the new configuration.

Returns state (clea ing) or

state (deploy men if RAI configuration is in progress asynchronously, or None if it is complete.

delete_

Dele RAI configuration on the give node

This meth dele the RAI configuration on the give node After RAID configuration is deleted, node.raid_config should be cleared by the implementation.

Paramet

tas A Task ager instan

Returns

state (clea ing) or

state (deployment if deletion is in

progress asynchronously, or None if it is complete.

get_pro

Return the properties of the interface

Returns

dictionary of <pro erty nam description: entries

class i

Base

iro dri bas Res

Example implementation

of a simple rescue

interface.

get_pro

Return the prop erties of the interface

Returns

dictionary of <pro erty nam description entries

rescue

Boo the task node into a rescue environmen

Paramet

tas

A Task

ager in- stan con- tain- ing the node to act

on.

Raises

In- stan cue- Fail- ure if node val- i- da- tion or res-

cue operation fails.

Returns

state if res- cue is in prog asyn chro or state if it

is complete.

unrescu

Tear dow the res- cue en- vi- ronmen and return to normal.

Parame

`tas` A Task ager instan containing the node to act on.

Raises

InstanUnrescueFailure if node validation or unrescue operation fails.

Returns

state if it is successful.

`validat`

Validate the

driv spec Nod deployment info

This meth validate whe the driv and/ instance prop erties

of the tasks node contains the required information for this interface to function.

This meth is often executed synchro in API reques

so it should not conduct long-running checks.

Parame

tas A Task agen instance containing the node to

act

on.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
on
mal-
form
pa-
ram-
e-

ter(s)

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-

ter(s)

class i
Base
iro
dri
bas
Sto

Ex-
am-
ple
im-
ple-
men
ta-
tion
of
sim-
ple

storage Interface.

attach_

Inform the storage subsystem to attach all volumes for the node.

Parame

tas

A Task ager instan

Raises

Unsupported-DrivTension

detach_

Inform the storage subsystem to detach all volumes for the node.

Paramet

tas

A Task ager in- stan

Raises

Un- sup- port- ed- Driv ten- sion

get_pro

Re- turn the prop er- ties of the in- ter- face

Returns

dic- tio- nary of <pro erty nam de- scrip tion: en- tries

should_

De- ter- mine if de- ploy shou per-

form
the
image
write

out.

Parame
tas
A
Task
ager
instan

Returns
Boo
valu
to
indicate
if
the
interface
expect

the image to be written by Ironic.

Raises
Unsupported-
Driv
tension

validat
Validate
the
drive
spec
Nod
deploy
men
info

This

meth validate whe the driv and/ instan properties of the tasks node contains the required information for this interface to function.

This meth is often exe cute synchro in API reques so it should not conduct long-running checks.

Paramet

tas A Task agen instan containing the node to act on.

Raises

Invalid Parame-

ter-
Valu
on
mal-
form
pa-
ram-
e-
ter(s)

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-
ter(s)

class i
Base
irc
dri
bas
Ver

Ex-
am-
ple
im-
ple-
men
ta-
tion
of
a
ven-
dor
pass
in-
terface.

first_r

get_prc
Re-

turn the prop- er- ties of the in- ter- face

Returns
dic- tio- nary of <pro erty nam de- scrip tion: en- tries

validat

Val- i- date vend spec ac- tions

If in- valid raise an ex- cep- tion: oth- er- wise re- turns Non

Paramet

- **tas**

A task from Task ager

- **met** Met to be val- i- date
- **kwa** Info for ac- tion.

Raises
Un- sup- port- ed- Driv ten- sion if meth can not be map

to the supported interfaces.

Raises
In- valid Pa- ram- e- ter- Valu if kwa does not con- tain

method.

Raises
Missing-Parameter-Valu

class i
Base
irc dri bas Ven

Example implementation of a secondary vendor passthru.

fourth_

get_prc
Return the properties of the interface

Returns
dictionary of <prc

erty nam descrip tion entries

second_

third_

validat

Validate venc spec actions

If invalid raise an exception otherwise returns Non

Parame

- **tas** A task from Task agen
- **met** Met to be val-

i-
date

- **kwa**
Info
for
ac-
tion.

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion
if
meth
can
not
be
map

to the supported interfaces.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
kwa
does
not
con-
tain

method.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu

ironic.drivers.modules.image_cache module

Utility for cach mas ter images

class i

Base obj

Clas handling access to cach for master images

clean_

Clea up directory with images keeping cach of the latest images.

Files with link coun >1

are neve dele Protecte by glob lock so

that no one messes with master images after we get listing and before we actually delete files.

Paramet

amo if pres amo of spac to reclair in byte clear ing

will stop, if this goal was reached, even if it is possible to clean up more files

`fetch_i`

Fetc image by give href to the destination path

Doe nothing if destination path exists and is up

to date with cache and href contents. Only creates a hard link (dest_path) to cached image if requested image is already in cache and up to date with href contents. Otherwise downloads an image, stores it in cache and creates a hard link (dest_path) to it.

Parame

- **hre** image UUI or href to fetcl
- **des** destination file path
- **ctx** context
- **for** bool valu whe to convert the image to raw format
- **exp** The

expect image format.

- **exp** The expect image chec sum

- **exp** The expect image chec sum algorithm if need

`ironic.`

Explicitly clea cach base on their priority (if required).

This clea up the

cach to free up the amo of spac requir

for the images in images_info. The caches are cleaned up one after the other in the order of their priority. If we still cannot free up enough space after trying all the caches, this method throws exception.

Paramet

- **`ctx`** context
- **`dir`** the directory (of the cach to be free up.
- **`ima`** a list of tuples of the form (image_ for whic

space is to be created in cache.

Raises

In- sufficien Disk exception, if we cannot free up enough space after trying all the caches.

`ironic.`

Decorator meth for addi clea priority to a class.

ironic.drivers.modules.image_utils module

class `i`

Base `obj`

publish

Mak image file dow load able

Depend ing on iron settings

push give file into Swi or copi it over to local HTTP servers document root and returns publicly accessible URL leading to the given file.

Parame

- **`ima`** path to file to publish
- **`obj`** nam of the published file

Returns
a URI to dow loaded published file

`unpubli`
With draw the image previously mad dow load able

De-
pend
ing
on
ironi
set-
tings
re-
mov
pre-
vi-
ousl
pub-
lishe

file from where it has been published - Swift or local HTTP servers document root.

Paramet

obj
nam
of
the
pub-
lishe
file
(op-
tiona

classme

With
draw
the
im-
age
pre-
vi-
ousl
mad
dow
load
able

De-
pend
ing
on
ironi
set-
tings
re-
mov
pre-
vi-
ousl

file from where it has been published - Swift or local HTTP servers document root.

publishe

Paramet

- **nod** the node for whic image was publishe
- **pre** object nam prefix.
- **suf** object nam suffix.

update_

`ironic.`

Dele the image if it was created for the node

Paramet

- **tas** an iron node object.
- **pre** Prefix to use for the object nam

ironic.

Dele the flopp image if it was created for the node

Paramet

tas an iron node object.

ironic.

Dele the ISO if it was created for

the instan

Paramet

`tas`

A task from Task agen

`ironic.`

`ironic.`

Prepare boot ISO image

Buil boot ISO out of *[instan* *[instan* and *[dri* if pres Otherwise, read *kernel_id* and *ramdisk_id* from *[instance_info]/image_source* Glance image metadata.

Push produce ISO image up to Glan and return temporary Swift URL to the image.

on.

Paramet

- `tas` a Task ager in- stan con- tain- ing the node to act
- `d_i` De- ploy men in- for- ma- tion of the node
- `roo` Roo UUI

Returns
boot ISO HTT URI

Raises
Miss- ing- Pa- ram- e- ter- Valu if

any of the requir

parameters are missing.

Raises
InvaliParameterValu if any of the parameters have invalid value.

Raises
ImageCreatio Fail if creating ISO image faile

`ironic.`

Prepare an image with configdriv

Decode base

contents and writ it into a disk image that can be attached e.g. to a virtual USB device. Images stored in Swift are downloaded first.

Paramet

- **`tas`** a Task agen instan containing the node to act on.
- **`con`** Config driv as a base enco strin

Raises

ImageCreationFail if it faile whil

creating the image.

Raises
Swi Operation if any operation with Swi fails.

Returns
image URL for the image.

`ironic.`

Prepare deploy or rescue ISO image

Buil boot ISO out of *[dri* or *[dri* and *[dri* then push built image up to Glance and return temporary Swift URL to the image.

If networl interface supplies networl configuration (*network_data*), a *network_data.json* will be written into an appropriate location on the final ISO.

Paramet

- **tas** a Task ager instan containing the node to act on.
- **par** a dictionary containing param-

e-
ter
nam

>value mapping to be passed to kernel command line.

- `mod`
ei-
ther
de-
ploy
or
res-
cue.
- `d_i`
De-
ploy
men
in-
for-
ma-
tion
of
the
node

Returns
boot
ISO
HTT
URL

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
any
of
the
re-
quir

parameters are missing.

Raises
In-
valid
Pa-

ram-
e-
ter-
Valu
if
any
of
the
pa-
ram-

eters have invalid value.

Raises

Im-
age-
Cre-
atio
Fail
if
cre-
at-
ing
ISO
im-
age
faile

`ironic.`

Pre-
pare
an
im-
age
with
the
give
con-
tent.

If
con-
tent
is
al-
read
an
HTT
URI
re-
turn
it
un-

char

Paramet

- `tas` a Task ager in- stan containing the node to act on.
- `con` Con tent as a strin with a file nam or byte with contents.
- `pre` Prefix to use for the object nam

Raises

Image- Creation

Faile if it faile whil creating the image.

Raises
Swi Operation if any operation with Swi fails.

Returns
image UR for the image.

`ironic.`

Prepare the flopp image for passing the parameters.

This meth prepare a temporary VFA files tem image and adds a file into the image which contains parameters to be passed to the ramdisk. Then this method uploads built image to Swift [redfish]swift_container, setting it to auto expire after [redfish]swift_object_expiry_timeout seconds. Finally, a temporary Swift URL is returned addressing Swift object just created.

Paramet

- **tas** a Task agen instan containing the node to act on.
- **par** a dictionary containing parameter nam >value mapping to be passed to deploy or rescue image via floppy image.

Raises
Im- age- Cre- ation Fail if it faile whil cre- at- ing the

floppy image.

Raises
Swi Op- er- a- tionI if any op- er- a- tion with Swi

fails.

Returns
im- age URI for the flop im- age.

ironic.drivers.modules.inspect_utils module

`ironic.`

Cre- ate ironi port

from
MA
ad-
dres
data
dict.

Cre-
ates
iron
port
from
MA
ad-
dres
data
re-
turn
with
in-
spec

tion or as requested by operator. Helper argument to detect the MAC address `get_mac_address` defaults to value part of MAC address dict key-value pair.

Paramet

- **`tas`**
  A
  Task
  agen
  in-
  stanc
- **`mac`**
  A
  dic-
  tio-
  nary
  of
  MA
  ad-
  dres
  re-
  turn
  by
  node

inspection.

- **`get`**
  a

func
tion
to
get
the
MAC
ad-
dres
from
mac
item

A mac item is the dict key-value pair of the previous `macs` argument.

ironic.drivers.modules.inspector module

Modules r
https
//
pypi
org/
proj
ironi

class i
Base
*iro
dri
bas
Ins*

In-
band
in-
spec
tion
via
ironi
insp
proj

abort (*t*
Abo
hard
ware
in-
spec
tion.

Paramet
tas
a

task from Task agen

get_pro

Return the properties of the interface

Returns
dictionary of <pro erty nam description: entries

inspect

Inspect hardware to obtain the hardware properties.

This particular implementation only start inspec

tion using ironic-inspector. Results will be checked in a periodic task.

Paramet

tas a task from Task ager

Returns

state

Raises

Hardware spection-Failure on failure

validat

Validate the drive spec inspection information.

If invalid raise an exception; oth- er- wise re- turn Non

Paramet

tas

a task from Task ager

Raises

Un- sup- port- ed- Driv ten- sion

ironic.drivers.modules.ipmitool module

IPM pow man ager driv

Uses the ip- mi- tool com man (http // ipmi sour net/) to re- motely manage hardware. This includes setting the boot device, getting a serial-over-LAN console, and controlling the power state of the machine.

NOT THA

CER
TAI
DIS-
TRO
MAY
IN-
STA
oper
BY
DE-
FAU
IN-

STEAD OF ipmitool, WHICH PROVIDES DIFFERENT COMMAND-LINE OPTIONS AND *IS NOT SUPPORTED* BY THIS DRIVER.

class i

Base
irc
dri
bas
Con

A base Con solel ter- face that uses ip- mi- tool.

get_pro

Re- turn the prop er- ties of the in- ter- face

Returns
dic- tio- nary of

<pro erty nam de- scrip tion: en- tries

validat

Val- i- date the Nod con- sole info

Parame

tas

a task from Task ager

Raises

In- valid Pa- ram- e- ter- Valu

Raises

Miss ing- Pa- ram- e- ter- Valu whe a re- quire pa- ram-

eter is missing

class i

Base
irc
dri
bas
Mar

detect_

Detects and returns the hardware vendor.

Parame

tas

A task from Task agen

Raises

Invalid Parameter-Valu if an invalid compo-

nent, indicator or state is specified.

Raises

Missing-Parameter-Valu if a

re-
quir
pa-
ram-

eter is missing

Returns

Strin
rep-
re-
sent-
ing
the
BM
re-
port
Ven-
dor
or
Man

ufacturer, otherwise returns None.

get_boo

Get
the
cur-
rent
boot
de-
vice
for
the
task
node

Re-
turn
the
cur-
rent
boot
de-
vice
of
the
node

Paramet

tas
a
task
from
Task

ager

Raises
InvalidParameterValu if requir IPM parameters are missing.

Raises
IPMIFailure on an error from ipmitool.

Raises
MissingParameterValu if a requir parameter is missing.

Returns
a dictio

nary containing:

boot_d
the boot device, one of *irc com boo* or Non if it

is unknown.

persist
Whe the boot device will persist to all future boot

or not, None if it is unknown.

get_pro
Return the prop erties of the interface

Returns
dictio-

nary
of
<pro
erty
nam
de-
scrip
tion
en-
tries

get_ser
Get
sen-
sors
data

Paramet
tas
a
Task
agen
in-
stanc

Raises
Faile
To-
Get-
Sen-
sor-
Data
whe
get-
ting
the
sen-
sor
data

fails.

Raises
Faile
ToPa
eSen
sor-
Data
whe
pars
ing
sen-
sor
data

fails

Raises
In- valic Pa- ram- e- ter- Valu if re- quir ipmi pa- ram-

eters are missing

Raises
Miss ing- Pa- ram- e- ter- Valu if a re- quir pa- ram-

eter is missing.

Returns
re- turns a dict of sen- sor data grou by sen- sor type

get_sup
Get a list

of the supported boot devices

Parame

tas

a task from Task agen

Returns

A list with the supported boot devices defined in *irc*

common.boot_devices.

inject_

Inject NMI Non Maskable Interrupt

Inject NMI (Nor Maskable Interrupt) for

a node immediately.

Parame

tas

A TaskManager instance containing the node to act on.

Raises

IPMIFailure on an error from ipmitool.

Returns

Non

set_boo

Set the boot device for the task's node.

Set the boot device to use on next reboot of the node

Parame

- **`tas`** a task from Task agen
- **`dev`** the boot device one of *`iro com boo`*
- **`per`** Boo valu True if the boot device will persist to

all future boots, False if not. Default: False.

Raises
Invalid Pa-

ram-
e-
ter-
Valu
if
an
in-
valic
boot
de-

vice is specified

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
re-
quire
ipmi
pa-
ram-

eters are missing.

Raises
IP-
MI-
Fail-
ure
on
an
er-
ror
from
ip-
mi-
tool.

validat
Chec
that
drive
con-
tains
IPM
cre-
den-
tials

Validate whe the driv property of the supplied task node

contains the required credentials information.

Parame

`tas`

a task from Task ager

Raises

Invalic Parameter-Valu if requir IPM param

eters are missing.

Raises

Missing-Parameter-Valu if a requir parameter is missing.

class i

Base *irc dri bas Pow*

get_pow

Get the current pow state of the task node

Paramet

tas a Task agen instanc containing the node to act on.

Returns

one of ironi POW POW or ERROR

Raises

Invali ParameterValu if requir ipmi parameters are missing.

Raises

MissingParameterValu if a requir parameter is missing.

Raises

IPMIFailure on an error from ipmitool (fro _power_status call).

get_pro

Return the properties of the interface

Returns
dictionary of <pro erty nam descrip tion entries

get_sup
Get a list of the support pow state

Paramet
tas
A Task ager instanc containing the node to act

on. currently not used.

Returns
A list with the support pow state

defined in *iro* *com*

states.

reboot

Cycles the pow to the task node

Paramet

- **tas** a Task agen instanc containing the node to act on.
- **tim** time out (in seconds posi- tive integer (> 0) for any power state. The timeout is counted once during power off and once during power on for reboots. `None` indicates that the default timeout will be used.

Raises

Miss-
ing-
Pa-
ram-
e-
ter-
Valu
if
re-
quir
ipmi
pa-
ram-

eters are missing.

Raises

In-
valid
Pa-
ram-
e-
ter-
Valu
if
an
in-
valid
pow
state

was specified.

Raises

Pow
er-
State
Fail-
ure
if
the
fi-
nal
state
of
the
node

is not POWER_ON or the intermediate state of the node is not POWER_OFF.

set_pow

Turn
the
pow
on,
off,

soft reboot or soft pow off.

on.

Paramet

- **tas** a Task agen instanc containing the node to act
- **pow** desired pow state one of ironi POW POW SOF or SOF
- **tim** time out (in seconds pos- i- tive in-

te-
ger
(>

0) for any power state. The timeout is counted once during power off and once during power on for reboots. `None` indicates that the default timeout will be used.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
an
in-
valic
pow
state

was specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
re-
quire
ipmi
pa-
ram-

eters are missing

Raises
Pow
er-
State
Fail-
ure
if
the
pow
coul
be
set
to
psta

validat

Validates driv for ipmitool driv

Chec that nod contains IPM credentials

Paramet

tas a Task agen instanc containing the nod to act on.

Raises

Invalid Parameter Valu if required ipmi parameters are missing.

Raises

Miss ing-Pa- ram- e- ter- Valu if a re- quir pa- ram-

eter is missing.

class i

Base
irc
dri
moc
ipm
IPM

A Con sole ter- face that uses ip- mi- tool and shel- linab

get_cor

Get the type and con- nec- tion in- for- ma- tion abou the console.

start_c

Star a remot console for the nod

Paramet

tas a task from Task ager

Raises

Invalic Parameter-Valu if requir ipmi param-eters are missing

Raises

Pass word File-Faile ToC ate if unable to create a file containing the password

Raises
Con sole ror if the directory for the PID file cannot be created

Raises
Con sole Subprocess Faile whe invoking the subprocess failed

stop_co
Stop the remote console session for the nod

Paramet
tas a task from Task ager

Raises

Con sole ror if unable to stop the console

`class` i

Base

irc dri moc ipm IPM

A Con sole ter face that uses ipmitool and socat.

get_cor

Get the type and connection information about the

console.

Paramet

tas

a task from Task agen

start_c

Star a re- mote con- sole for the node

Paramet

tas

a task from Task agen

Raises

In- valid Pa- ram- e- ter- Valu if re- quire ipmi pa- ram-

eters are missing

Raises

Pass word File- Faile ToC ate if un- able to create a file containing the password

Raises
Con sole ror if the directory for the PID file can not be created

Raises
Con sole Sub process Fail whe invoking the sub process failed

stop_co
Stop the remote console session for the node

Parame
tas

a task from Task ager

Raises
Con sole ror if unable to stop the console

class i

Base *irc dri bas Ver*

bmc_res
Reset BM with IPM com man bmc reset (war

Paramet

- **tas** a Task ager instanc
- **htt**

the HTT meth used on the request

- **war** bool parameter to decide on warm or cold

reset.

Raises
IPMIFailure on an error from ipmitool.

Raises
MissingParameterValu if a required param

eter is missing.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
whe
an
in-
valic
valu
is

specified

get_pro
Re-
turn
the
prop
er-
ties
of
the
in-
ter-
face

Returns
dic-
tio-
nary
of
<pro
erty
nam
de-
scrip
tion
en-
tries

send_ra
Send
raw
byte
to
the
BMC
Byte

shou be a strin of byte

Parame

- **tas** a Task agen in- stanc
- **htt** the HTT meth used on the re- ques
- **raw** a strin of raw byte to send e.g. 0x00 0x01

Raises
IP- MI- Fail- ure on an er- ror from ip- mi-

tool.

Raises

Miss ing- Pa- ram- e- ter- Valu if a re- quir pa- ram-

eter is missing.

Raises

In- valic Pa- ram- e- ter- Valu whe an in- valic valu is

specified.

validat

Val- i- date venc spec ac- tions

If in- valic raise an ex- cep- tion oth- erwise re- turn Non

Valid me

- send
- bmc

Paramet

- **tas** a task from Task agen
- **met** meth to be val- i- date
- **kwa** info for ac- tion.

Raises
In- valic Pa- ram- e- ter- Valu whe an in- valic pa-

rameter value is specified.

Raises

Miss ingParameterValu if a requir parameter is missing.

`ironic.`

Chec if the com man stder contains cipher suite errors.

Paramet

`cmd`

The com man stder

Returns

True if the cmd contains a cipher suite error, Fals

otherwise.

ironic.

Give the possible next avai cipher suite version

Base on CON and the last cipher suite version used that faile

This function is only called if the node doesnt have cipher_suite set. Starts using the last element of the list and decreasing the index.

Paramet

act latest cipher suite used in the ipmi call.

Returns the next possible ciphersuite or None in case of empty configuration.

`ironic.`

Dum SDR data to a file.

Paramet

- **`tas`** a Task ager instance
- **`fil`** the path to SDR dum file.

Raises

IPMIFailure on an error from ipmitool.

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing.

Raises

In-
valid
Pa-
ram-
e-
ter-
Valu
whe
an
in-
valid
valu
is

specified.

`ironic.`

Send
raw
byte
to
the
BMC
Byte
shou
be
a
strin
of
byte

Paramet

- **tas**
  a
  Task

agen instanc

- **raw** a strin of raw byte to send e.g. 0x00 0x01

Returns

a tuple with stdout and stde

Raises

IPMIFailure on an error from ipmitool.

Raises

MissingParameterValu if a requir

pa-
ram-

eter is missing.

Raises

In-
valic
Pa-
ram-
e-
ter-
Valu
whe
an
in-
valic
valu
is

specified.

`ironic.`

Up-
date
vari-
able
and
the
ci-
pher
suite
cmd

This
func
tion
up-
date
the
val-
ues
for
all
pa-
ram-
e-
ters

so they can be used in the next retry of _exec_ipmitool.

Paramet

- **act**

a strin that represent the cipher suite that was used in the command.

- **arg** a list that contains the ipmitool com man that was executed, it will be modified in-place.

Returns the next actual_

ironic.drivers.modules.ipxe module

iPXI Boo Interface

class i
Base *iro dri moc pxe PXE*

iro
dri
bas
Boo

capabil

ipxe_en

ironic.drivers.modules.iscsi_deploy module

class i

Base
iro
dri
mod
age
Age
iro
dri
mod
age
Age
iro
dri

base.DeployInterface

iSCS De- ploy In- ter- face for depl relat ac- tions

clean_u

Clea up the de- ploy men en- vironmen for the task node.

Unlinks TFT and instan images and triggers image cach cleanup. Removes the TFTP configuration files for this node.

Paramet

tas a Task ager instan containing the node to act on.

deploy

Star deploymen of the task node

Fetc instan image, update the DHC port options for next boot, and issues a reboot request to the power driver. This causes the node to boot into the deployment ramdisk and triggers the next phase of PXE-based deployment via agent heartbeats.

Parame

`tas` a Task agen instanc containing the node to act on.

Returns

deploy state DEPLO WA

`get_pro`

Return the properties of the interface

Returns

dictio-

nary of <pro erty nam de- scrip tion: en- tries

has_dec

Whe the driv sup- ports de- com pose de- ploy steps

Pre- vi- ousl (sinc Rock driv used a sin- gle de- ploy de- ploy

step on the deploy interface. Some additional steps were added for the direct and iscsi deploy interfaces in the Ussuri cycle, which means that more of the deployment flow is driven by deploy steps.

prepare

Pre- pare the de- ploy men en- vi- ron- men for this task

node.

Gen- er- ates the TFT con- fig- u- ra- tion for PXE boot both

the deployment and user images, fetches the TFTP image from Glance and add it to the local cache.

Paramet

`tas` a Task ager in- stan con- tain- ing the node to act

on.

Raises

Net- worl Er- ror: if the pre- vi- ous clea ing port can-

not be removed or if new cleaning ports cannot be created.

Raises

In-
vali
Pa-
ram-
e-
ter-
Valu
whe
the
wro
pow
state
is

specified or the wrong driver info is specified for power management.

Raises

Stor
ageE
ror
If
the
stor-
age
driv
is
un-
able
to
at-

tach the configured volumes.

Raises

othe
ex-
cep-
tions
by
the
node
pow
driv
if
som
thing
wro

occurred during the power action.

Raises

any
boot
in-
ter-
face

prepare exceptions

prepare

support

Indicate if an interface is supporte

This will be set to Fals for interface whic are unte in first- or third-party CI, or in the process of being deprecated.

validat

Validate the deploy men information for the tasks node.

Paramet

tas

a Task agen in- stanc con- tain- ing the node to act

on.

Raises

In- valic Pa- ram- e- ter- Valu

Raises

Miss ing- Pa- ram- e- ter- Valu

write_i

Meth in- voke whe de- ploy us- ing iSCS

This meth is in- voke dur- ing a

heart
beat
from
an
agen
whe

the node is in wait-call-back state. This deploys the image on the node and then configures the node to boot according to the desired boot option (netboot or localboot).

Parame

- **tas** a Task agen ob- ject con- tain- ing the node
- **kwa** the kwa pass from the heart beat meth

Raises

In- stan ploy Fail- ure, if it en- coun ters som er- ror

during the deploy.

ironic.

En- sure the file sys- tem sees the iSCS bloc de- vice

ironic.

Chec if the re- ques im- age is large than the root par-

tition size.

Doe noth ing for who disk im- ages

Paramet

tas a Task agen in- stanc con- tain- ing the node to act

on.

Raises

In- stand ploy Fail- ure if size of the im- age is grea

than root partition.

`ironic.`

Re- sum a de- ploy men upon get- ting POS data from de-

ploy ramdisk.

This meth raise no ex- cep- tions be- caus it is in- tend to

be invoked asynchronously as a callback from the deploy ramdisk.

Paramet

- **`tas`** a Task ager instan containing the node to act on.
- **`kwa`** the kwa to be pass to deploy

Raises
Invali Stat if the even is not allowe by the associated state machine.

Returns
a dictionary con-

taining the following keys

For partition image:

- root uuid UUI of root partition
- efi system partition uuid UUI of the uefi system

partition (if boot mode is uefi).

Not
If key exists but valu is Non it mea

partition

doesnt exist.

For whole disk image:

- disk identifier: ID of the disk to whic image was

deployed.

`ironic.`

Dele the iSCS target.

`ironic.`

All-in-one function to deploy a who disk im-to a node.

age

Paramet

- **add** The iSC IP ad- dres
- **por** The iSC port num- ber.
- **iqn** The iSC qual i- fied nam
- **lun** The iSC log- i- cal unit num- ber.
- **ima** Path for the in- stan disk im- age.
- **nod**

node uuid

- con Optiona Base encode Gzip configdrive content or

configdrive HTTP URL.

- con Optiona Add a flag that will modify the behavi

of the image copy to disk.

Returns
a dictionary containing the key disk identifier

to identify the disk which was used for deployment.

ironic.

All-in-one function to deploy a partition image to a node.

Paramet

- **add** The iSCS IP address
- **por** The iSCS port number.
-

`iqn` The iSCS qualified nam

- `lun` The iSCS logical unit number.
- `ima` Path for the instan disk image.
- `roo` Size of the root partition in meg
- `swa` Size of the swap partition in meg

- **eph** Size of the ephe par- ti- tion in meg If 0, no

ephemeral partition will be created.

- **eph** The type of file sys- tem to for- mat the ephe par-

tition.

- **nod** node uuid Used for log- ging

- **pre** If True no files tem is writ- ten to the device, preserving whatever content it had (if the partition table has not changed).

configdrive HTTP URL.

ephe bloc

- **con** Optiona Base encode Gzip configdriv content or
- **boo** Can be local or netboot netboot by default
- **boo** Can be bios or uefi. bios by default
- **dis** The disk label

to be used when creating the

partition table. Valid values are: msdos, gpt or None; If None ironic will figure it out according to the boot_mode parameter.

- **cpu** Architecture of the node being deployed to.

Raises

Instan ploy Failure if image virtual size is bigger than root partition size.

Returns

a dictionary containing the following

ing keys root

uuid: UUID of root partition efi system partition uuid: UUID of the uefi system partition (if boot mode is uefi). NOTE: If key exists but value is None, it means partition doesnt exist.

`ironic.`

Do iSCS dis-cov-ery on por-tal.

`ironic.`

Met in-voke whe de-ploy with the agen ram

This meth is in-voke by driv for do-ing iSCS de-ploy us-ing agent ramdisk. This method assumes that the agent is booted up on the node and is heartbeating.

Paramet

- **tas** a Task agen

object containing the node

- **age** an instance of agen which will be used during iscsi

deploy (for exposing nodes target disk via iSCSI, for install boot loader, etc).

Returns

a dictionary containing the following keys

For partition image:

- root uuid UUI of root par-

ti-
tion

- efi sys- tem par- ti- tion uuid UUI of the uefi sys- tem

partition (if boot mode is uefi).

Not
If key ex- ists but valu is Non it mea par- ti- tion

doesnt exist.

For who disk im- age:

- disk iden ti- fier: ID of the disk to

which image was

deployed.

Raises

Instanc ployFailure if it encounters som error

during the deploy.

`ironic.`

force iSCS initiator to reread luns

`ironic.`

Returns the information require for doing iSCS

deploy in a dictionary.

Paramet

- **nod** ironi node ob- ject
- **add** iSCS ad- dres
- **iqn** iSCS iqn for the tar- get disk
- **por** iSCS port de- fault to one spec i- fied in the con-

figuration

- **lun** iSCS lun, de- fault to 1
- **con** flag that

will mod ify the be havi of the im age

copy to disk.

Raises
Miss ing- Pa- ram e- ter- Valu if som re- quire pa- ram

eters were not passed.

Raises
In- valid Pa- ram e- ter- Valu if any of the pa- ram

eters have invalid value.

`ironic.`

Lo- gin to an iSCS tar- get.

`ironic.`

Lo-
gout
from
an
iSCS
tar-
get.

`ironic.`

Val-
i-
date
the
pre-
requ
for
iSCS
de-
ploy

Val-
i-
date
whe
node
in
the
task
pro-
vide
has
som
port
en-

rolled. This method validates whether conductor url is available either from CONF file or from keystone.

Paramet

`tas`
a
Task
ager
in-
stanc
con-
tain-
ing
the
node
to
act

on.

Raises
In-valid Pa-ram-e-ter-Valu if the URI of the Iron

API service is not configured in config file and is not accessible via Keystone catalog.

Raises
Miss ing-Pa-ram-e-ter-Valu if no port are en-rolle

for the given node.

`ironic.`

Ver-ify iscsi con-nec-tion.

ironic.drivers.modules.noop module

Dun in-ter-face im-ple-men ta-tion

for use as default with optional interfaces.

Note that unlike fake implemen tations thes do not pass validation and raise exceptions for user-accessible actions.

class i

Base `obj`

Mix to add to an interface to mak it fail validation.

get_pro

validat

class i

Base *iro* *dri* *moc*

noc
Fai
irc
dri
bas
BIC

BIO interface implementation that raise errors on all requests.

apply_c

Validate & apply BIO settings on the give node

This meth take the BIO settings from the settings para and applies BIOS settings on the given node. It may also validate the given bios settings before applying any settings and manage failures when setting an invalid BIOS config. In the case of needing password to update the BIOS config, it will be taken from the driver_info properties. After the BIOS configuration is done, cache_bios_settings will be called to update the nodes BIOS setting table with the BIOS configuration applied on the node.

Paramet

- **tas**
  a
  Task
  ager
  in-
  stan
- **set**
  Dic-
  tona
  con-
  tain-
  ing
  the
  BIO
  con-
  fig-
  u-
  ra-
  tion.

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion
if
the
node
driv
does
sup-

port BIOS configuration.

Raises
In-
valid
Pa-
ram-
e-
ter-

Valu if val- i- da- tion of settings fails.

Raises
Miss ing- Pa- ram- e- ter- Valu if som re- quir pa- ram- eters are missing.

Returns
state if BIO con- fig- u- ra- tion is in prog asyn chro or None if it is complete.

cache_b
Stor or up- date BIO prop er- ties on the give

node

This meth store BIO prop erties to the bios table during cleaning operation and updates bios_settings table when apply_configuration() and factory_reset() are called to set new BIOS configurations. It will also update the timestamp of each bios setting.

Parame

`tas`

a Task agen instanc

Raises

Unsupport ed-Driv tension if the node driv does support getting BIOS properties from bare metal.

Returns

Non

`factory`

Reset BIO configuration to factory default

on the given node.

This method resets BIOS configuration to factory de-

fault on the given node. After the BIOS reset action is done, cache_bios_settings will be called to update the nodes BIOS settings table with default bios settings.

Parame

tas

a Task

ager

instan

Raises

Unsupported-

Driv

tension

if the node

driv

does

sup-

port BIOS reset.

Returns

state

if BIO

configuration is in prog asyn chro

or None if it is complete.

class i

Base *irc dri moc noc Fai irc dri bas Con*

Con sole interface implemen tation that raise errors

on all requests.

get_cor

Get connection information abou the console

This meth shou return the necessary information for the client to access the console.

Paramet

tas A Task agen instan containing the node to act on.

Returns the console connection information.

start_c Star a remote console for the

task
node

This
meth
shou
not
raise
an
ex-
cep-
tion
if
con-
sole
al-
read

started.

Parame

`tas`

A
Task
agen
in-
stan
con-
tain-
ing
the
node
to
act

on.

`stop_co`

Stop
the
re-
mote
con-
sole
ses-
sion
for
the
task
node

Parame

`tas`

A
Task

ager in- stan con- tain- ing the node to act

on.

class i

Base

iro
dri
moc
noc
Fai
iro
dri
bas
Ins

In- spec in- ter- face im- ple- men ta- tion that raise er- rors

on all requests.

inspect

In- spec hard ware

In- spec hard ware to ob- tain

the essential & additional hardware properties.

Parame

tas

A task from Task agen

Raises

Hard ware spec tion-Fail ure, if unable to get essential hardware properties.

Returns

Resulting state of the inspection i.e. state or Non

class i

Base

irc

dri

moc

noc
Fai
irc
dri
bas
RAI

RAI in- ter- face im- ple- men ta- tion that raise er- rors on all requests.

create_

Cre- ates RAI con- fig- u- ra- tion on the give node

This meth cre- ates a RAI con- fig- u- ra- tion on the give node. It assumes that the target RAID configuration is already available in node.target_raid_config. Implementations of this interface are supposed to read the RAID configuration from node.target_raid_config. After the RAID configuration is done (either in this method OR in a call-back method), ironic.common.raid.update_raid_info() may be called to sync the nodes RAID-related information with the RAID configuration applied on the node.

Parame

- **tas**
  A Task ager instan
- **cre**
  Setting this to Fals indicate not to create

root volume that is specified in the nodes target_raid_config. Default value is True.

- **cre**
  Setting this to Fals indicate not to create

non-root volumes (all except the root volume) in the nodes target_raid_config. Default value is True.

- **del**
  Setting this to True

indicate to dele RAI configuration prior to creating the new configuration.

Returns
state (clea ing) or state (deploymen if RAI configuration is in progress asynchronously, or None if it is complete.

delete_

Dele RAI configuration on the give node

This meth dele the RAI configuration on the give node

After RAID configuration is deleted, node.raid_config should be cleared by the implementation.

Paramet

tas

A Task ager instan

Returns

state (clea ing) or state (deploy men if dele tion is in

progress asynchronously, or None if it is complete.

validat

Validate the give RAI configuration.

This meth validate the give RAI configuration. Driv

implementations of this interface can override this method to support custom parameters for RAID configuration.

Paramet

- **tas** A Task agen in- stanc
- **rai** The RAI con- fig- u- ra- tion to val- i- date

Raises
In- valic Pa- ram- e- ter- Valu if the RAI con- fig- u- ration is invalid.

class i
Base
irc
dri
moc
noc
Fai
irc
dri
bas
Res

Rescue interface implementation that raise errors on all requests.

rescue

Boo the task node into a rescue environment

Paramet

tas

A Task agen instanc containing the node to act on.

Raises

Instan cue-Failure if node

validation or rescue operation fails.

Returns
state if rescue is in prog asynchro or state if it is complete.

unrescu
Tear dow the rescue environmen and return to normal.

Paramet
tas
A Task agen instanc containing the node to

act on.

Raises
In- stan Un- res- cue- Fail- ure if node val- i- da- tion or unrescue operation fails.

Returns
state if it is suc- cess ful.

class i
Base
irc
dri
moc
noc
Fai
irc
dri
bas
Ver

Ven- dor in- ter- face im- ple- men ta- tion that raise errors

on all requests.

driver_

Validate driv venc pass actions

If invalid raise an exception; otherwise returns Non

Paramet

- **met** meth to be validate
- **kwa** info for action.

Raises

Missing-Parameter-

Valu if kwa does not contain

certain parameter.

Raises

Invalic ParameterValu if parameter does

not match.

ironic.drivers.modules.noop_mgmt module

No-op man agement interface implementation.

`class i`

Base *iro dri bas Mar*

No-op

man agemen interface implemen tation.

Using this implemen tation requir the boot order

to be preconfigured to first try PXE booting, then fall back to hard drives.

`get_boo`

Get the current boot device for a node

Provide the current boot device of the node Be awa

that

not all drivers support this.

Parame

tas

A task from Task ager

Raises

Miss ing-Pa-ram-e-ter-Valu if a re-quir pa-ram-

eter is missing

Returns

A dic-tio-nary con-tain-ing:

boot_d

Ahe boot de-vice one of *irc com boc* or Non if it

is unknown.

persist
When the boot device will persist to all future boot

or not, None if it is unknown.

get_pro
Return the prop erties of the interface

Returns
dictionary of <pro erty nam description: entries

get_ser
Get sensors data meth

Parame
tas
A

Task
ager
in-
stan

Raises
Fail
To-
Get-
Sen-
sor-
Data
whe
get-
ting
the
sen-
sor
data

fails.

Raises
Fail
ToPa
eSen
sor-
Data
whe
pars
ing
sen-
sor
data
fails

Returns

Re-
turns
a
con-
sis-
tent
for-
mat
dict
of
sen-
sor
data
grou

by sensor type, which can be processed by Ceilometer. eg,

```
↪
```

(continues on next page)

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

get_sup

Get a list of the support boot device

Paramet

tas

A task from Task agen

Returns

A list with the support boot device de- fined in *iro*

common.boot_devices.

set_boo

Set the boot de- vice for a node

Set the boot de- vice to use on next re- boot of the node

Parame

- **tas** A task from Task agen
- **dev** The boot de- vice one of *iro com boo*

- **per** Boo valu True if the boot device will persist to

all future boots, False if not. Default: False.

Raises
In valic Param eter Valu if an in valic boot de

vice is specified.

Raises
Miss ing Param eter Valu if a re quir param

eter is missing

validat
Val i date the

drive
spec
Nod
deployment
info

This
meth
validate
whe
the
drive
and/
instance
prop
erties

of the tasks node contains the required information for this interface to function.

This
meth
is
often
executed
synchro
in
API
reques

so it should not conduct long-running checks.

Parame

tas

A
Task
agen
instanc
containing
the
node
to

act

on.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
on
mal-
form
pa-
ram-
e-

ter(s)

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing
pa-
ram-
e-

ter(s)

ironic.drivers.modules.pxe module

PXE
Boo
In-
ter-
face

class i
Base
irc
dri
moc
age
Age
irc

dri
moc
age
Hea
irc
dri

base.DeployInterface

clean_u

Clea up the de- ploy men en- vi- ron- men for the task

node.

Un- link TFT and in- stan im- ages and trig- gers im- age cach

cleanup. Removes the TFTP configuration files for this node.

Parame

tas
a Task agen in- stan con- tain- ing the node to

act

on.

deploy

Perform a deployment to the task node

Perform the necessary work to deploy an image onto the specified node. This method will be called after prepare(), which may have already performed any preparatory steps, such as pre-caching some data for the node.

Parame

tas

A Task ager instance containing the node to act on.

Returns

status of the

de-
ploy
One
of
ironi

deploy_

Chec
if
the
de-
ploy
men
has
start
al-
read

Returns

True
if
the
de-
ploy
has
start
Fals
oth-
er-
wise

deploy_

Chec
if
the
de-
ploy
men
is
al-
read
com
plete

Returns

True
if
the
de-
ploy
men
is
com

plete
Fals
otherwise

get_pro

Return the properties of the interface.

Returns
dictionary of <pro
erty
nam
description
entries

prepare

Prepare the deploymen
environmen
for the task

node.

If preparation

of the deploy men environmen ahea

of time is possible, this method should be implemented by the driver.

If implemen this meth mus be iden potent. It may be

called multiple times for the same node on the same conductor.

This meth is calle before *deploy*

Parame

tas

A Task ager instanc containing the node to act

on.

reboot_

Met
in-
voke
af-
ter
the
de-
ploy
men
is
com
plete

Parame

tas
a
Task
agen
in-
stan

should_

Whe
agen
boot
is
man
aged
by
iron

validat

Val-
i-
date
the
drive
spec
Nod
de-
ploy
men
info

This
meth
val-
i-
date
whe
the
drive

and/
in-
stan
prop
er-
ties

of the tasks node contains the required information for this interface to function.

This
meth
is
of-
ten
ex-
e-
cute
syn-
chro
in
API
re-
ques

so it should not conduct long-running checks.

Parame

`tas`

A
Task
ager
in-
stan
con-
tain-
ing
the
node
to
act

on.

Raises

In-
valid
Pa-
ram-
e-
ter-
Valu
on
mal-
form
pa-
ram-

e-

ter(s)

Raises

Miss ing- Pa- ram- e- ter- Valu on miss ing pa- ram- e-

ter(s)

class i

Base

iro

dri

moc

pxe

PXE

iro

dri

bas

Boc

capabil

class i

Base

iro

dri

moc

age

Age

iro

dri

moc

age

Hea

iro

dri

base.DeployInterface

deploy

Per- form

a deploy ment to the task node

Perform the necessary work to deploy an image onto the specified node. This method will be called after prepare(), which may have already performed any preparatory steps, such as pre-caching some data for the node.

Paramet

tas

A Task ager instanc containing the node to act on.

Returns

status of the deploy One of ironi

get_pro

Return the prop- er- ties of the in- ter- face

Returns

dic- tio- nary of <pro erty nam de- scrip tion entries

prepare

Pre- pare the de- ploy men en- vi- ron- men for the task

node.

If prep- ra- tion of the de- ploy men environmen ahea

of time is possible, this method should be implemented by the driver.

If implemen this meth mus be iden potent. It may be

called multiple times for the same node on the same conductor.

This meth is calle before *deploy*

Paramet

`tas` A Task ager instanc containing the node to act

on.

`validat`

Validate the drive

spec
Nod
deploy
men
info

This
meth
validate
whe
the
driv
and/
instan
properties

of the tasks node contains the required information for this interface to function.

This
meth
is
often
executed
synchro
in
API
reques

so it should not conduct long-running checks.

Paramet

tas
A
Task
agen
instan
containing
the
node
to
act

on.

Raises

In-
vali
Pa-
ram
e-
ter-
Valu
on
mal-
form
pa-
ram
e-
ter(s)

Raises

Miss
ing-
Pa-
ram
e-
ter-
Valu
on
miss
ing
pa-
ram
e-
ter(s)

ironic.drivers.modules.pxe_base module

Base
PXE
In-
ter-
face
Met
ods

class i

Base
`obj`

clean_u

Clea

up the boot of instance

This method clean up the environment that was setup for boot

ing the instance. It unlinks the instance kernel/ramdisk in nodes directory in tftproot and removes the PXE config.

Parame

`tas`

a task from Task agen

Returns

Non

`clean_`

Clea up the boot of ironi ram

This meth clean up the PXE environment

that was setu for booting the deploy or rescue ramdisk. It unlinks the deploy/rescue kernel/ramdisk in the nodes directory in tftproot and removes its PXE config.

Paramet

- **tas** a task from Task agen
- **mod** Label indicating a deploy or rescue operation was carried out on the node. Supported values are deploy and rescue. Defaults to deploy, indicating deploy operation was carried out.

Returns
Non

get_pro
Return the properties of the interface

Returns
dictio-

nary
of
<pro
erty
nam
de-
scrip
tion
entries

ipxe_en

prepare

Prepare
the
boot
of
instan

This
meth
prepare
the
boot
of
the
instan
after
read
ing

relevant information from the nodes instance_info. In case of netboot, it updates the dhcp entries and switches the PXE config. In case of localboot, it cleans up the PXE config.

Paramet

tas
a
task
from
Task
agen

Returns
Non

prepare

Prepare

the boot of Iron ram us- ing PXE

This meth pre- pare the boot of the de- ploy or res- cue ker- nel/ramdisk after reading relevant information from the nodes driver_info and instance_info.

Paramet

- **tas** a task from Task ager
- **ram** the pa- ram- e- ters to be pass to the ram pxe driver passes these parameters as kernel command-line arguments.

Returns
Non

Raises
MissingParameterValu if som information

is missing in nodes driver_info or instance_info.

Raises
InvalidParameterValu if som information

provided is invalid.

Raises
IronicException, if som pow or set boot boot device

operation failed on the node.

validat
Validate the

PXE spec info for boot ing deploy images.

This meth validate the PXE spec info for boot ing the ram and instance on the node. If invalid, raises an exception; otherwise returns None.

Parame

tas a task from Task agen

Returns

Non

Raises

Invali ParameterValu if som parameters are invalid.

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
som
re-
quir
pa-
ram-

eters are missing.

validat

Val-
i-
date
that
the
node
has
re-
quir
prop
er-
ties
for

inspection.

Parame

tas

A
Task
ager
in-
stan
with
the
node
be-
ing
chec

Raises

Un-
sup-
port-
ed-
Driv
ten-

sion

validat

Validate that the node has requir properties for rescue.

Paramet

tas

a Task agen instan with the node being chec

Raises

MissingParameterValu if node is missing one or more required parameters

ironic.drivers.modules.snmp module

Iron
SNM
pow
man
agen

Pro-
vide
ba-
sic
pow
con-
trol
us-
ing
an
SNM
enab
sma
pow

controller. Uses a pluggable driver model to support devices with different SNMP object models.

class i

Base
obj

SNM
clien
ob-
ject.

Per-
form
low
leve
SNM
get
and
set
op-
er-
a-

tions
En-
cap-

sulates all interaction with PySNMP to simplify dynamic importing and unit testing.

get (*oid*

Use
PyS-
NM
to
per-
form
an
SNM
GET
op-
er-
a-
tion

on a single object.

Parame

`oid`
The
OID
of
the
ob-
ject
to
get.

Raises

SN-
MP-
Fail-
ure
if
an
SNM
re-
ques
fails

Returns

The
valu
of
the
re-
ques
ob-
ject.

`get_nex`

Use PyS NM to perform an SNM GET NEX operation on a table object.

Paramet

`oid` The OID of the object to get.

Raises

SNMPFail ure if an SNM request fails

Returns

A list of values of the reques table object.

set (*oid,*

Use PyS-NM to perform an SNM SET operation

on a single object.

Parame

- **oid** The OID of the object to set.
- **val** The valu of the object to set.

Raises

SNMP-Failure if an SNM request fails

class i

Base
iro
dri
moc
snm
SNM

SNM
driv
clas
for
APC
Mas
ter-
Swi
PDU
de-
vice

SNM
ob-
jects
for
APC
SN-
M-
P-
Driv
APC
Mas
ter-
Swi
PDU

1.3.6.1.4.1.318.1.1.4.4.2.1.3 sPDUOutletCtl Values: 1=On, 2=Off, 3=PowerCycle, [more options follow]

oid_dev

system_

value_p

value_p

class i

Base
iro
dri

moc
snm
SNM

SNM
drive
class
for
APC
Mas
ter-
Swit
Plus
PDU
de-
vice

SNM
ob-
jects
for
APC
SN-
M-
P-
Driv
APC
Mas
ter-
Swit
Plus

PDU: 1.3.6.1.4.1.318.1.1.6.5.1.1.5 sPDUOutletControlMSPOutletCommand Values: 1=On, 3=Off, [more options follow]

oid_dev

system_

value_p

value_p

class i

Base
irc
dri
moc
snm
SNM

SNM
driv
clas
for
APC
Rac
PDU
device

SNM
objects
for
APC
SNMP
Driv
APC
PDU
PDU
#
1.3.

rPDUOutletControlOutletCommand Values: 1=On, 2=Off, 3=PowerCycle, [more options follow]

oid_dev

system_

value_p

value_p

class i

Bas
irc
dri
moc
snm
SNM

SNM
driv
clas
for
Ater
PDU
device

SNM objects for Aten PDU 1.3.6 Outlet Pow Values: 1=O 2=O 3=Pending, 4=Reset

oid_dev

system_

value_p

value_p

class i

Base *iro dri moc snm SNM*

SYS_OBJ

class i

Base `obj`

SNM pow driv base class

The SNMP-Driv class

hierarchy implemen man

specific MIB actions over SNMP to interface with different smart power controller products.

oid_ent

power_c

Set the pow state to this node to OFF

Raises
SNMPFailure if an SNM request fails

Returns
pow state One of *iro com sta*

power_c

Set the pow state to this node to

ON.

Raises
SNMPFailure if an SNM request fails

Returns
pow state One of *irc com sta*

power_r

Reset the pow to this node

Raises
SNMPFailure if an SNM request fails

Returns
pow state One of *irc com sta*

power_s

Returns a node's current power state

Raises
SNMPFailure if an SNM request fails

Returns
power state One of *iro com sta*

retry_i

class i

Base *iro dri moc snm SNM*

SNM driv class for Bay MRI PDU devices

SNM ob-

jects
for
Bay
MRI
PDU
4779
1,
3,
5,
3,
1,
3,

{unit_id} Outlet Power Values: 0=Off, 1=On, 2=Reboot

oid_dev

unit_ic

value_p

value_p

class i

Base
irc
dri
moc
snm
SNM

SNM
drive
class
for
Cy-
ber-
Pow
PDU
de-
vice

SNM
ob-
jects
for
Cy-
ber-
Pow
PDU
1.3.6

eP-
DU-
Out-
let-
Con

trolOutletCommand Values: 1=On, 2=Off, 3=PowerCycle, [more options follow]

oid_dev

system_

value_p

value_p

class i

Base
irc
dri
moc
snm
SNM

SNM
driv
class
for
Eato
Pow
PDU

The
Eato
pow
PDU
does
not
fol-
low
the
mod
of
SN-
M-
P-

DriverSimple as it uses multiple SNMP objects.

SNM
ob-
jects

for
Eato
Pow
PDU
1.3.6
let
ID>
out-
let-
Con
trol-

Status Read 0=off, 1=on, 2=pending off, 3=pending on 1.3.6.1.4.1.534.6.6.7.6.6.1.3.<outlet ID> outletControlOffCmd Write 0 for immediate power off 1.3.6.1.4.1.534.6.6.7.6.6.1.4.<outlet ID> outletControlOnCmd Write 0 for immediate power on

oid_dev

oid_pow

oid_pow

oid_sta

status_

status_

status_

status_

system_

value_p

value_p

class i
Base
iro
dri
moo
snm
SNM

SNM
driv

base class for simple PDU device

Here simple refer to device whic provide a single SNM

object for controlling the power state of an outlet.

The default OID of the pow state object is of the form

<enterprise OID>.<device OID>.<outlet ID>. A different OID may be specified by overriding the _snmp_oid method in a subclass.

abstrac

Device-dependent portion of the pow

OID.

state object

abstrac
Valu representing pow off state

abstrac
Valu representing pow on state

class i

Base *irc dri moc snm SNM*

SNM drive class for Teltron PDU device

SNM objects for Teltron PDU 1.3.6 Outlet

Pow Values: 1=O

2=On

oid_dev

system_

value_p

value_p

class i

Base

iro
dri
bas
Pow

SNM Pow Interface

This Pow erInterface class provide a mec anism for

controlling the power state of a physical device using an SNMP-enabled smart power controller.

get_pow

Get the current pow state

Poll

the SNM de- vice for the cur- rent pow state of the node

Paramet

tas

An in- stan of *iron*

Raises

Miss ing- Pa- ram- e- ter- Valu if re- quire SNM pa- ram-

eters are missing.

Raises

In- valid Pa- ram- e- ter- Valu if SNM pa- ram- e- ters

are invalid.

Raises
SNMPFailure if an SNM request fails

Returns
pow state One of *iro com sta*

get_pro
Return the properties of the interface

Returns
dictionary of <pro erty nam description: entries

reboot
Cycles the pow

to a nod

Parame

- **`tas`** An instan of *iron*
- **`tim`** timeout (in seconds Unsupport by this interface.

Raises
MissingParameterValu if requir SNM parameters are missing.

Raises
InvalidParameter-

Valu if SNM parameters

are invalid.

Raises

Power-State-Fail-ure if the final pow state of the

node is not POWER_ON after the timeout.

Raises

SN-MP-Fail-ure if an SNM request fails

set_pow

Turn the pow on or off.

Set the pow state of a node

Paramet

- **tas**
  An in- stan of *iron*
- **pst**
  Ei- ther POW or POW from :clas *iron*
- **tim**
  time out (in sec- onds Un- sup- port by this in- ter-

face.

Raises
Miss ing- Pa- ram- e- ter- Valu if re- quir SNM pa- ram-

eters are missing.

Raises

In-
valic
Pa-
ram-
e-
ter-
Valu
if
SNM
pa-
ram-
e-
ters

are invalid or *pstate* is invalid.

Raises

Pow
er-
Stat
Fail-
ure
if
the
fi-
nal
pow
state
of
the

node is not as requested after the timeout.

Raises

SN-
MP-
Fail-
ure
if
an
SNM
re-
ques
fails

validat

Chec
that
node
con-
tains
the
req-
ui-
site

field

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
re-
quir
SNM
pa-
ram-

eters are missing.

Raises

In-
valic
Pa-
ram-
e-
ter-
Valu
if
SNM
pa-
ram-
e-
ters

are invalid.

`ironic.`

`ironic.`

Module contents

Submodules

ironic.drivers.base module

Ab-
strac
base
class
for

drive

`ironic.`

Con stan hold ing all know interface

class `i`

Base *iro dri bas Bas*

abstrac

Validate & apply BIO settings on the give node

This meth take the BIO settings from the settings para and ap-

plies BIOS settings on the given node. It may also validate the given bios settings before applying any settings and manage failures when setting an invalid BIOS config. In the case of needing password to update the BIOS config, it will be taken from the driver_info properties. After the BIOS configuration is done, cache_bios_settings will be called to update the nodes BIOS setting table with the BIOS configuration applied on the node.

Paramet

- **tas** a Task ager in- stan

- **set** Dic- tona con- tain- ing the BIO con- fig- u- ra- tion.

Raises

Un- sup- port ed- Driv ten- sion if the node drive does sup-

port BIOS configuration.

Raises

In- valid Pa- ram- e- ter- Valu

if validation of settings fails.

Raises
Miss ing-Parameter-Valu if some required parameters are missing.

Returns
state if BIO configuration is in prog asyn chro or None if it is complete.

abstrac
Stor or update BIO properties on the give node

This meth stor BIO prop erties to the bios table during cleaning operation and updates bios_settings table when apply_configuration() and factory_reset() are called to set new BIOS configurations. It will also update the timestamp of each bios setting.

Parame

`tas` a Task agen instan

Raises

Unsup port ed-Driv tension if the node driv does support getting BIOS properties from bare metal.

Returns

Non

`abstrac`

Reset BIO configuration

to factory default

on the given node.

This method resets BIOS configuration to factory de-

fault on the given node. After the BIOS reset action is done, cache_bios_settings will be called to update the nodes BIOS settings table with default bios settings.

Parame

tas

a Task

ager

instan

Raises

Unsupport-

ed-

Driv

tension

if the node

drive

does

sup-

port BIOS reset.

Returns

state

if BIO

con-

fig-
u-
ra-
tion
is
in
prog
asyn
chro

or None if it is complete.

interfa

In-
ter-
face
type
used
for
clea
step
and
log-
ging

class i

Base
obj

A
bare
driv
ob-
ject
whic
will
have
in-
ter-
face
at-
tach
later

Any
com
pos-
able
in-
ter-
face
shou
be
adde

as class at- tribu

of this class, as well as appended to core_interfaces or standard_interfaces here.

propert

bios =
Stan dard at- tribu for BIO re- lated fea- tures

A ref- er- ence to an in- stan of :clas

boot =
Stan dard at- tribu for boot re- lated fea- tures

A ref- er- ence to an in- stan of :clas

console

Standard attribute for managing console access.

A reference to an instance of :clas

propert

Interfaces that are required to be implemen

deploy

Core attribute for managing deploymen

A ref-

er-
ence
to
an
in-
stan
of
:clas

get_pro

Get
the
prop
er-
ties
of
the
driv

Returns

dic-
tio-
nary
of
<pro
erty
nam
de-
scrip
tion
en-
tries

inspect

Stan
dard
at-
tribu
for
in-
spec
tion
re-
lated
fea-
tures

A
ref-
er-
ence
to
an

in- stanc of :clas

managem

Stan dard at- tribu for man age- men re- latec fea- tures

A ref- er- ence to an in- stanc of :clas

network

Core at- tribu for net- work con- nec- tiv- ity.

A ref- er- ence to an in- stanc of :clas

propert

propert

Interface that can be no-op.

power =

Cor attribu for managing pow state

A reference to an instanc of :clas

raid =

Standard attribu for RAI related features

A reference to an instanc

of :clas

rescue

Stan dard attribu for accessing rescue features

A reference to an instance of :clas

storage

Stan dard attribu for (remote storage interface

A reference to an instance of :clas

vendor

At-
tribu
for
ac-
cess
ing
any
venc
spec
ex-
ten-
sion

A
ref-
er-
ence
to
an
in-
stanc
of
:clas

class i
Base
obj

A
base
in-
ter-
face
im-
ple-
men
ing
com
mon
func
tions
for

Driver Interfaces.

execute
Ex-
e-
cute
the
clear
step
on
task

A clea step mus take a single positiona argumen

a TaskManager object. It may take one or more keyword variable arguments (for use with manual cleaning only.)

A step can be executed synchro or asyn chro A step

should return None if the method has completed synchronously or states.CLEANWAIT if the step will continue to execute asynchronously. If the step executes asynchronously, it should issue a call to the continue_node_clean RPC, so the conductor can begin the next clean step.

Paramet

- **tas** A Task ager object
- **ste** The clea step dictio-

nary rep- re- sent- ing the step

to execute

Returns

Non if this meth has com- plete syn- chro or state if the

step will continue to execute asynchronously.

execute

Ex- e- cute the de- ploy step on task

A de- ploy step mus take a sin- gle po- si- tion ar- gu-

ment: a TaskManager object. It may take one or more keyword variable arguments (for use in the future, when deploy steps can be specified via the API).

A step can be executed synchronously or asynchronously. A step

should return None if the method has completed synchronously or states.DEPLOYWAIT if the step will continue to execute asynchronously. If the step executes asynchronously, it should issue a call to the continue_node_deploy RPC, so the conductor can begin the next deploy step.

Parame

- **tas** A Task agen object
- **ste** The deploy step dictionary representing the

step to execute

Returns
Non if this meth has com plete

syn-
chro
or
state
if
the

step will continue to execute asynchronously.

get_cle

Get
a
list
of
(en-
able
and
dis-
able
clea
step
for
the

interface.

This
func
tion
will
re-
turn
all
clea
step
(bot
en-
able
and
dis-

abled) for the interface, in an unordered list.

Parame

tas

A
Task
agen
ob-
ject,
use-
ful
for
in-
ter-
face

over

riding this function

Raises

Noc

if there is a prob lem get-ting the step from the

driver. For example, when a node (using an agent driver) has just been enrolled and the agent isnt alive yet to be queried for the available clean steps.

Returns

A list of clea step dic-tio-nar-ies

get_dep

Get a list of (en-able and dis-able de-ploy step for

the interface.

This func-tion will re-turn

all deploy step (bot enable and disabled) for the interface, in an unordered list.

Paramet

`tas` A Task ager object, useful for interface overriding this function

Raises

`Ins` if ther is a problem getting the step from the driver. For example, when a node (using an agent driver) has just been enrolled and the agent isnt alive yet to be queried for the available deploy steps.

Returns

A list of deploy step dictionaries

abstrac
Returnthe proper-ties of the in-terface

Returns
dic-tio-nary of <pro erty nam de-scrip tion: en-tries

interfa
In-ter-face type used for clea steps and log-ging

support
In-di-cates if an in-ter-face is sup-

port

This will be set to Fals for interface whic are unte in first- or third-party CI, or in the process of being deprecated.

abstrac

Validate the drive spec Nod deployment info

This meth validate whe the drive and/ instance prop erties of the tasks node contains the required information for this interface to function.

This meth is often ex-

e-
cute
syn-
chro
in
API
re-
ques

so it should not conduct long-running checks.

Paramet
`tas`
A
Task
agen
in-
stanc
con-
tain-
ing
the
node
to
act

on.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
on
mal-
form
pa-
ram-
e-

ter(s)

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
on
miss
ing

parameter(s)

class i

Base
iro
dri
bas
Bas

Interface for boot relat actions

capabil

abstrac

Clea up the boot of instan

This meth clea up the environmen that was setu for booting the instance.

Paramet

tas
A task

from
Task
agen

Returns
Non

abstrac
Clea
up
the
boot
of
ironi
ramd

This
meth
clear
up
the
environ-
men
that
was
setu
for
boot

ing the deploy or rescue ramdisk.

Parame
tas
A
task
from
Task
agen

Returns
Non

interfa
Inter-
face
type
used
for
clea
steps
and
log-

ging

abstrac

Pre-
pare
the
boot
of
in-
stan

This
meth
pre-
pare
the
boot
of
the
in-
stan
af-
ter
read
ing

relevant information from the nodes database.

Paramet

tas

A
task
from
Task
ager

Returns

Non

abstrac

Pre-
pare
the
boot
of
Iron
ram

This
meth
pre-
pare
the
boot
of

the deploy or rescue ram after reading relevant information from the nodes database.

Parame

- **`tas`** A task from Task ager
- **`ram`** The options to be pass to the ironi ramd Different implementations might want to boot the ramdisk in different ways by passing parameters to them. For example,

When Age ram is boot to deploy a node it take the parameters ipa-api-url, etc.

Othe

implementations can mak use of ram to pass such

information. Different implementations of boot interface will have different ways of passing parameters to the ramdisk.

Returns

Non

validat

Validate that the node has required properties for

inspection.

Parame

tas

A Task ager instance with the node being chec

Raises

Missing-Pa-

rameterValu if node is miss ing one or more required parameters

Raises
UnsupportedDriv tension

`validat`
Validate that the node has requir prop erties for rescue.

Paramet
`tas` A Task agen instanc with the node being chec

Raises
Miss

ingParameterValu if node is missing one

or more required parameters

Raises

Unsupported-Driv tension

class i

Base *iro dri bas Bas*

Interface for cons relat actions

abstrac

Get connection information about the console

This meth shou return the necessary information for the client to access the console.

Paramet

`tas` A Task ager instanc containing the node to act on.

Returns

the console connection information.

`interfa`

Interface type used for clea steps

and logging

abstrac

Star a remote console for the task node

This meth shou not raise an exception if console alread started.

Parame

tas A Task agen instan containing the node to act on.

abstrac

Stop the remot

con-
sole
ses-
sion
for
the
task
node

Paramet
tas
A
Task
ager
in-
stan
con-
tain-
ing
the
node
to
act

on.

class i
Base
irc
dri
bas
Bas

In-
ter-
face
for
depl
relat
ac-
tions

abstrac
Clea
up
the
de-
ploy
men
en-
vi-
ron-
men

for the task

node.

If prep ration of the deploy men environ men ahea

of time is possible, this method should be implemented by the driver. It should erase anything cached by the *prepare* method.

If implemen this meth mus be iden potent. It may be

called multiple times for the same node on the same conductor, and it may be called by multiple conductors in parallel. Therefore, it must not require an exclusive lock.

This meth is calle before *tear*

Parame

`tas`

A Task ager in-

stan
con-
tain-
ing
the
node
to
act

on.

abstrac

Per-
form
a
de-
ploy
men
to
the
task
node

Per-
form
the
nec-
es-
sary
work
to
de-
ploy
an
im-
age
onto

the specified node. This method will be called after prepare(), which may have already performed any preparatory steps, such as pre-caching some data for the node.

Paramet

tas

A
Task
ager
in-
stan
con-
tain-
ing
the
node
to
act

on.

Returns
sta- tus of the de- ploy One of iron

heartbe

Rec a hear beat for the nod

Parame

- **tas** A Task agen in- stan con- tain- ing the nod to act

on.

- **cal** a URI to use to call to the ram

- **age** The version of the agen that is heartbeating
- **age** TLS certificate for the agen
- **age** Status of the heartbeating agen
- **age** Message describing the agen status

Returns Non

interfa Interface

type used for clean steps and logging

abstrac

Prepare the deployment environment for the task's node.

If preparation of the deployment environment ahead of time is possible, this method should be implemented by the driver.

If implemented this meth must be idempotent. It may be called multiple times for the same node on the same conductor.

This meth is calle be- fore *de- ploy*

Paramet

tas A Task agen in- stan con- tain- ing the node to act on.

prepare

Pre- pare the node for clea ing task

For ex- am- ple, node that use the Iron Pyth Age will need to boot the ramdisk in order to do in-band cleaning tasks.

If the func tion is asyn chro the driv will need to handle

settings node.driver_internal_info[clean_steps] and node.clean_step, as they would be set in ironic.conductor.manager._do_node_clean, but cannot be set when this is asynchronous. After, the interface should make an RPC call to continue_node_cleaning to start cleaning.

NOT this shou be mov to Boo Interface whe it gets implemented.

Parame

`tas` A Task ager instanc containing the node to act on.

Returns

If this func tion is going to be asyn chro shou return *states.CLEANWAIT*. Otherwise, should return *None*. The interface will need to call _get_cleaning_steps and then RPC to continue_node_cleaning

abstrac

Take over man agemen of this task node from a dead conductor.

If conductors host main tain a stati relationship to nodes, this method should be implemented by the driver to allow conductors to perform the necessary work during the remapping of nodes to conductors when a conductor joins or leaves the cluster.

For exar

Neu tron mus

for-
ward
DHC
BOO
re-
ques
to
a
con-
duc-

tor which has prepared the tftpboot environment for the given node. When a conductor goes offline, another conductor must change this setting in Neutron as part of remapping that nodes control to itself. This is performed within the *takeover* method.

Parame
`tas`
A
Task
agen
in-
stanc
con-
tain-
ing
the
node
to
act

on.

`abstrac`
Tear
dow
a
pre-
vi-
ous
de-
ploy
men
on
the
task
node

Give
a
node
that
has
been
pre-

vi-
ousl
de-
ploy
to,
do
all

cleanup and tear down necessary to un-deploy that node.

Paramet

`tas`

A
Task
ager
in-
stan
con-
tain-
ing
the
node
to
act

on.

Returns

sta-
tus
of
the
de-
ploy
One
of
iron

`tear_do`

Tear
dow
af-
ter
clea
ing
is
com
plete

Give
that
clea
ing
is
com

plet do all clea and tear dow necessary to allow the node to be deployed to again.

NOT this shou be mov to Boo Interface whe it gets implemented.

Parame

tas

A Task ager instan containing the node to act on.

class i

Base

irc dri bas Bas

Interface for

insp relat actions

ESSENTI

The prop erties requir by sche uler/

abort (*t*

Abo asyn chro nize hard ware inspec tion.

Abo an ongoing hard ware introspec tion, this is only used for asynchronize based inspect interface.

NOT This interface is calle with node

exclusive lock held

the interface implementation is expected to be a quick processing.

Parame

`tas`

a task from Task agen

Raises

Unsupported-Driv tension if the meth is not im-

plemented by specific inspect interface.

`abstrac`

Inspec hard ware

Inspec hard ware to obtain the essential & additional hardware properties.

Parame

tas

A task from Task agen

Raises

Hard ware spec tion-Fail-ure, if unable to get essen-

tial hardware properties.

Returns

Resulting state of the inspection i.e. state or Non

interfa

Interface type used for clea step and logging

class i

Base

iro dri bas Bas

Interface for management related actions

detect_

Detects stores and returns the hardware vendor.

If the Nod object `pro` field does not already contain a

`vendor` field, then this method is intended to query Detects the BMC hardware vendor and stores the returned value with-in the Node object `properties` field if detected.

Parame

tas

A task from

Task
agen

Raises
In-
vali
Pa-
ram-
e-
ter-
Valu
if
an
in-
vali
com
po-

nent, indicator or state is specified.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-

eter is missing

Returns
Strin
rep-
re-
sent-
ing
the
BMC
re-
port
Ven-
dor
or
Man

ufacturer, otherwise returns None.

abstrac
Get

the current boot device for a node

Provide the current boot device of the node. Be aware that not all drivers support this.

Paramet

tas

A task from Task ager

Raises

MissingParameterValue if a required parameter is missing

Returns

A

dictionary containing:

boot_d
Ahe boot device one of *iro com boo* or Non if it

is unknown.

persist
Whe the boot device will persist to all future boot

or not, None if it is unknown.

get_boo
Get the current boot mod for a node

Provide the

current boot mod of the node

NOTE: I may not implement that.

Paramet

tas

A task from Task agen

Raises

Missing-Param-e-ter-Valu if a re-quir pa-ram-

eter is missing

Raises

Driv Op-er-a-tionl or its deriv tive in case of

driv

runtime error.

Raises
Unsupported-Driv tension if request operation is not supported by the driver

Returns
The boot mod one of `irc` `com` `boc` or Non if it is unknown.

get_inc
Get current state of the indicator of the hardware component.

Paramet

- **tas** A task from Task agen
- **com** The hardware component, one of *iro com com*
- **ind** Indicator ID (as reported by *get_*

Raises
Invalid Parameter Valu if an invalid component or indicator is specified.

Raises
Miss

ing-Parameter-Valu if a re-quir pa-ram-

eter is missing

Returns

Cur-rent state of the in-di-ca-tor, one of *iro com*

indicator_states.

get_mac

Get MA ad-dres in-for-ma-tion for the node

Parame

tas

A Task agen in-stan con-tain-ing

the node to act on.

Raises
Unsupported-Driv tension

Returns
A list of MAC addres for the node

get_sec
Get the current secure boot state for the node

NOTE: may not implement that.

Paramet
tas
A task from Task

ager

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
a
re-
quir
pa-
ram-
eter is missing

Raises
Driv
Op-
er-
a-
tionl
or
its
deriv
tive
in
case
of
drive
runtime error.

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion
if
se-
cure
boot
is
not
supported by the driver or the hardware

Returns
Boo

abstrac

Get sensors data meth

Paramet

tas

A Task agen instanc

Raises

Fail To- Get- Sen- sor- Data whe get- ting the sen- sor data

fails.

Raises

Fail ToPa eSen sor- Data whe pars ing sen- sor data fails

Returns

Re- turns a con- sis- tent

format dict of sensor data grou

by sensor type, which can be processed by Ceilometer. eg,

```
↪
```

(continues on next page)

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

```
```

(continues on next page)

↪

(continued from previous page)

```
↪
```

(continues on next page)

(continued from previous page)

abstrac

Get a list of the sup- porte boot de- vice

Parame

tas

A task from

Task ager

Returns
A list with the sup- port boot de- vice de- fined in *irc*

common.boot_devices.

get_sup
Get a list of the sup- port boot mod

NOTE: may not im- ple- men that.

Paramet
tas
A task from Task ager

Raises
Un- sup- port- ed- Driv ten-

sion if reques operation is not supported by the driver

Raises
Driv OperationI or its deriv tive in case of drive runtime error.

Raises
Miss ingParameterValu if a requir parameter is missing

Returns
A list with the supporte boot mod defined in

irc
com

boot_modes. If boot mode support cant be determined, empty list is returned.

get_sup

Get a map of the sup- port in- di- ca- tors (e.g. LED

Paramet

- **tas** A task from Task agen
- **com** If not *Non* re- turn in- di- ca- tor in- for- ma-

tion for just this component, otherwise return indicators for all existing components.

Returns

A dic- tio- nary of

hard
ware
com
po-
nent
(`ir`
`com`
`com`
as

keys with values being dictionaries having indicator IDs as keys and indicator properties as values.

```
↪": true,
```

(continues on next page)

(continued from previous page)

```
↪   "off",
↪   "on"
```

(continues on next page)

(continued from previous page)

```
↪": true,






↪": [
```

(continues on next page)

(continued from previous page)

```
↪   "on"
```

(continues on next page)

(continued from previous page)

```
↪": true,

↪": [

↪  "off",
```

(continues on next page)

```
↪ "on"
```

(continued from previous page)

inject_

Inject NM Non Mas able Interrupt

Inject NM (Nor Mas able Interrupt for a

node immediately.

Parame

tas

A Task agen instance containing the node to act on.

Raises

Unsupported-Driv tension

interfa

Interface type used for clean step and logging

abstrac

Set the boot device for a node

Set

the boot device to use on next reboot of the node

Parame

- **tas** A task from Task agen
- **dev** The boot device, one of *irc com boc*
- **per** Boo valu True if the boot device will persist to

all future boots, False if not. Default: False.

Raises

InvalidParameterValu if an invalid boot device is specified.

Raises

MissingParameterValu if a requir parameter is missing

set_boo

Set the boot mod for a node

Set the boot mod to use on next reboot of the node

Driv
implemen
ing
this
meth
are
required
to
implemen

the *get_supported_boot_modes* method as well.

NOTE: I
one
boot
mod
may
not
implemen
that.

Paramet

- **tas**
A
task
from
Task
ager
- **mod**
The
boot
mod
one
of
iro
com
boo

Raises
Invalid
Pa-

rameterValu if an invalid boot mod

is specified.

Raises
MissingParameterValu if a requir parameter is missing

Raises
UnsupportedDrivtension if request operation is not supported by the driver

Raises
Driv Operation or its deriv

tive in case of driv

runtime error.

set_inc

Set indicator on the hardware component to the desired state.

Parame

- **tas** A task from Task ager
- **com** The hardware component, one of *irc com com*
- **ind** Indica-

tor ID (as reported by *get_*

State
Desired state of the indicator, one of *iro com*

indicator_states.

Raises
InvalidParameterValu if an invalid compo-

nent, indicator or state is specified.

Raises
MissingParameterValu if a required pa-

ram-

eter is missing

set_sec

Set the current secure boot state for the node

NOTE: may not implement that.

Parame

- **tas** A task from Task agen
- **sta** A new state as a bool

Raises

Missing-Parameter-Valu if a

re-
quir
pa-
ram-

eter is missing

Raises
Driv
Op-
er-
a-
tion
or
its
deriv
tive
in
case
of
driv

runtime error.

Raises
Un-
sup-
port-
ed-
Driv
ten-
sion
if
se-
cure
boot
is
not

supported by the driver or the hardware

class i

Base
iro
dri
bas
Bas

Base
class
for
net-
work
in-
ter-
face

`abstrac`

Add the clea ing networ to a node

Paramet

`tas` A Task ager instan

Returns

a dictionary in the form {por neutron

Raises

Networ Error

`add_ins`

Add the inspec tion networ to the node

Paramet

`tas` A

Task ager in- stanc

Returns

a dic- tio- nary in the form {por neu- tron

Raises

Net- work Er- ror

Raises

In- valic Pa- ram- e- ter- Valu if the net- work in- ter-

face configuration is invalid.

abstrac

Add the pro- vi- sion ing net- work to a node

Parame

tas

A Task ager in- stan

Raises

Net- worl Er- ror

add_res

Add the res- cu- ing net- worl to the node

Paramet

tas

A Task ager in- stan

Returns

a dic- tio- nary in the form {por neu- tron

Raises

Net- worl Er- ror

Raises

In- valic Pa-

ram-
e-
ter-
Valu
if
the
net-
work
in-
ter-

face configuration is invalid.

abstrac

Con
fig-
ure
ten-
ant
net-
work
for
a
node

Paramet

tas

A
Task
ager
in-
stanc

Raises

Net-
work
Er-
ror

abstrac

Re-
turns
the
cur-
rentl
used
VIF
as-
so-
ci-
ated
with
port

or portgroup

We are booting the node only in one network at a time

and presence of cleaning_vif_port_id means were doing cleaning, of provisioning_vif_port_id - provisioning, of rescuing_vif_port_id - rescuing. Otherwise its a tenant network.

Parame

- `tas` A Task agen instanc
- `p_o` Iron port or port grou object.

Returns VIF ID associated with p_o or Non

get_noc Return network configuration for node NIC

Gath L2 and L3 network settings from ironi port ob- jects and

underlying network provider, then put together collected data in form of Nova network metadata (*network_data.json*) dict.

Iron wou even- tually pass network configuration to

the node being managed out-of-band.

Parame

tas

A Task agen instan

Raises

In-

valic Parameter-Valu if the network interface configuration is invalid.

Raises
Missing-Parameter-Valu if some parameters are missing.

Returns
a dict holding network configuration information adhearing Nova network metadata layout (*network_data.json*).

get_pro
Return the properties

of the interface

Returns

dictionary of <property name

description

entries

interfa

Interface type used for clear steps and logging

need_p

Chec if ironi node mus be powered on before applying network changes

Paramet

tas

A Task

ager in- stanc

Returns
Boo

abstrac
Handle any actions required when a port char

Parame

- **tas** A Task ager instanc
- **por** a char Port object.

Raises
Conflict, Faile ToU dateD- HCI tOn- Port

abstrac
Handle any ac-

tions
re-
quire
whe
a
port
char

Parame

- **tas**
  A
  Task
  ager
  in-
  stanc
- **por**
  a
  char
  Port
  ob-
  ject.

Raises
Con
flict,
Faile
ToU
dat-
eD-
HCF
tOn-
Port

abstrac
Re-
mov
the
clear
ing
net-
work
from
a
node

Parame
tas
A
Task

ager in- stan

Raises
Net- worl Er- ror

remove_
Re- mov the in- spec tion net- worl from a node

Paramet
tas
A Task ager in- stan

Raises
Net- worl Er- ror

Raises
In- valic Pa- ram- e- ter- Valu if the net- worl in- ter-

face configuration is invalid.

Raises

MissingParameterValu if som parameters are missing.

abstrac

Remov the provision ing networl from a node

Paramet

tas

A Task agen instan

remove_

Remov the rescuing networl from a node

Paramet

tas

A

Task agen in- stanc

Raises
Net- worl Er- ror

Raises
In- valic Pa- ram- e- ter- Valu if the net- worl in- ter-

face configuration is invalid.

Raises
Miss ing- Pa- ram- e- ter- Valu if som pa- ram- e- ters

are missing.

abstrac
Un- con- fig- ure ten- ant net- worl for

a node

Parame

tas

A Task agen instanc

validat

Validate the network interface

Parame

tas

A Task agen instanc

Raises

InvalidParameterValu if the network inter-

face configuration is invalid.

Raises

MissingParameterValu

if som parameters

are missing.

validat

Validate that the node has requir proper ties for

inspection.

Paramet

tas

A Task agen instan with the node being chec

Raises

Missing-Parameter-Valu if node is missing one

or more required parameters

Raises
Unsupported-Driv ten-sion

validat
Validate the network interface for rescue operation.

Paramet
tas
A Task agen instanc

Raises
InvalidParameterValu if the network interface configuration is invalid.

Raises
Missing-Pa-

ram- e- ter- Valu if som pa- ram- e- ters

are missing.

abstrac

At- tach a vir- tual net- worl in- ter- face to a node

Parame

- **tas** A Task agen in- stan
- **vif** a dic- tio- nary of in- for- ma- tion abou a VIF.

It must have an id key, whose value is a unique identifier for that VIF.

Raises
Network Error, VifAlready Attach NoF hysical-

Ports

abstrac

Detach a virtual network interface from a node

Parame

- **tas** A Task agen instanc
- **vif** A VIF ID to detach

Raises
Network

Error, VifN tAttach

abstrac

List attach VIF IDs for a node

Paramet

tas

A Task agen instanc

Returns

List of VIF dictionaries, each dictionary will have

an id entry with the ID of the VIF.

class i

Base *iro dri bas Bas*

Interface for pow relat ac-

tions

abstrac

Re- turn the pow state of the tasks node

Paramet

tas

A Task ager in- stanc con- tain- ing the node to act

on.

Raises

Miss ing- Pa- ram- e- ter- Valu if a re- quir pa- ram-

eter is missing.

Returns

A pow state One of *irc*

com
sta

get_sup

Get a list of the sup-port pow state

Parame

tas

A Task agen in-stan con-tain-ing the node to act on.

Returns

A list with the sup-port pow state de-fined in iro com states.

interfa

In-ter-face type used for

clea steps and logging

abstrac

Perform a hard reboot of the task node

Driv are expect to properly handle case whe node is pow

ered off by powering it on.

Paramet

- **tas** A Task agen instanc containing the node to act

on.

-

tim
time-out (in sec-onds pos-i-tive in-te-ger (>

0) for any power state. `None` indicates to use default timeout.

Raises
Miss-ing-Pa-ram-e-ter-Valu if a re-quir pa-ram-

eter is missing.

abstrac
Set the pow state of the task node

Paramet

- **tas** A Task agen in-stan con-tain-ing the node to act

on.

- **pow** Any power state from *irc com sta*

- **tim** time out (in seconds pos-itive integer (>

0) for any power state. `None` indicates to use default timeout.

Raises
Missing-Parameter-Valu if a required param-

eter is missing.

support
Che if pow

sync is sup- port for the give node

If Fal the con- duc- tor will sim- ply store wha ever get re- turns in the database instead of trying to force the expected power state.

Paramet

tas A Task agen in- stan con- tain- ing the node to act on.

Returns bool whe pow sync is sup- port

class i

Base *irc*

dri
bas
Bas

apply_c

Applies RAI configuration on the given node

Paramet

- **tas** A Task agen instanc
- **rai** The RAI configuration to apply.
- **cre** Setting this to Fals indicates

not to create root volume that is specified in raid_config. Default value is True.

- **`cre`** Setting this to Fals indicate not to create non-root volumes (all except the root volume) in raid_config. Default value is True.
- **`del`** Setting this to True indicate to dele RAI configuration prior to creating the new configuration.

Raises
Invali ParameterValu if the RAI configuration is invalid.

Returns

state if RAI configuration is in prog asyn chro

or None if it is complete.

abstrac

Creates RAI configuration on the give node

This meth creates a RAI configuration on the give

node. It assumes that the target RAID configuration is already available in node.target_raid_config. Implementations of this interface are supposed to read the RAID configuration from node.target_raid_config. After the RAID configuration is done (either in this method OR in a call-back method), ironic.common.raid.update_raid_info() may be called to sync the nodes RAID-related information with the RAID configuration applied on the node.

Paramet

- **`tas`** A Task ager instan
- **`cre`** Setting this to Fals indicate not to create

root volume that is specified in the nodes target_raid_config. Default value is True.

- **`cre`** Setting this to Fals indicate not to create

non-root volumes (all except the root volume) in the nodes target_raid_config. Default value is True.

- **`del`** Setting this to True indicate to dele RAI configuration prior to creating the new configuration.

Returns
state (clea ing) or state (deploy men if RAI configuration is in progress asynchronously, or None if it is complete.

abstrac

Dele RAI configuration on the give node

This meth dele the RAI configuration on the give node After RAID configuration is deleted, node.raid_config should be cleared by the implementation.

Paramet
tas
A

Task ager instand

Returns
state (clea ing) or state (deploy men if dele tion is in progress asynchronously, or None if it is complete.

get_log

Get the prop erties that can be spec ified for logical disks.

This meth returns a dictionary containing the prop erties that can be specified for logical disks and a textual description for them.

Returns
A dictionary containing properties that can be

mentioned for logical disks and a textual description for them.

get_pro
Return the prop- erties of the interface

Returns
dictionary of <pro erty nam description: entries

interfa
Interface type used for clea step and

logging

validat

Validate the RAI Interface

This meth validate the properties define by Iron for RAID configuration. Driver implementations of this interface can override this method for doing more validations (such as BMCs credentials).

Paramet

`tas` A Task agen instan

Raises

InvalidParameterValu if the RAI configuration is invalid.

Raises

Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
som
pa-
ram-
e-
ters

are missing.

validat

Val-
i-
date
the
give
RAI
con-
fig-
u-
ra-
tion.

This
meth
val-
i-
date
the
give
RAI
con-
fig-
u-
ra-
tion.
Driv

implementations of this interface can override this method to support custom parameters for RAID configuration.

Paramet

- **tas**
  A
  Task

agen in- stanc

- **rai** The RAI con- fig- u- ra- tion to val- i- date

Raises
In- valid Pa- ram- e- ter- Valu if the RAI con- fig- u- ration is invalid.

ironic.
This may be used as the de- ploy args ar- gu- men for RAID interfaces implementing an apply_configuration deploy step.

class i
Base
irc

dri bas Bas

In- ter- face for resc relat ac- tions

clean_u

Clea up the res- cue en- vi- ron- men for the task node

This is par- tic- u- larly use- ful for node whe res- cu- ing

is asynchronous and a timeout occurs.

Parame

tas

A Task agen in- stan containing the node to act on.

Returns
Non

interfa
Interface type used for clea step and logging

abstrac
Boo the task node into a rescue environmen

Paramet
tas
A Task agen instance containing the node to act on.

Raises
In-
stan
cue-
Fail-
ure
if
node
val-
i-
da-
tion
or
res-

cue operation fails.

Returns
state
if
res-
cue
is
in
prog
asyn
chro
or
state
if
it

is complete.

abstrac
Tear
dow
the
res-
cue
en-
vi-
ron-
men
and
re-
turn
to

normal.

Paramet
tas
A
Task
ager

in-
stanc
con-
tain-
ing
the
node
to
act

on.

Raises

In-
stanc
Un-
res-
cue-
Fail-
ure
if
node
val-
i-
da-
tion

or unrescue operation fails.

Returns

state
if
it
is
suc-
cess
ful.

class i

Base
irc
dri
bas
Bas

Base
class
for
stor-
age
in-
ter-
face

abstrac

Inform the storage subsystem to attach all volumes for the node.

Paramet

tas

A Task ager instanc

Raises

Unsupported-Driv ten- sion

abstrac

Inform the storage subsystem to detach all volumes for the node.

Paramet

tas

A Task ager

in-
stan

Raises
Un-
sup-
port
ed-
Driv
ten-
sion

interfa
In-
ter-
face
type
used
for
clea
step
and
log-
ging

abstrac
De-
ter-
min
if
de-
ploy
shou
per-
form
the
im-
age
writ

out.

Parame
tas
A
Task
agen
in-
stan

Returns
Boo
valu
to
in-

di- cate if the in- ter- face ex- pect

the image to be written by Ironic.

Raises

Un- sup- port- ed- Driv ten- sion

class i

Base

irc

dri

bas

Bas

In- ter- face for all ven- dor pass func tion- al- ity.

Ad- di- tiona venc

or drive spec ca- pa- bil- i- ties

shou

be implemented as a method in the class inheriting from this class and use the @passthru or @driver_passthru decorators.

Meth ods dec-o-rated with @dr shou be shor lived be-caus it

is a blocking call.

driver_

Val-i-date driv vend pass ac-tions

If in-valic raise an ex-cep-tion; oth-er-wise re-turn Non

Paramet

- **met** meth to be val-

i-
date

-
kwa
info
for
ac-
tion.

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
kwa
does
not
con-
tain

certain parameter.

Raises
In-
valid
Pa-
ram-
e-
ter-
Valu
if
pa-
ram-
e-
ter
does

not match.

interfa
In-
ter-
face
type
used
for
clea
steps
and
log-

ging

abstrac

Validate venc spec actions

If invalic raise an exception otherwise returns Non

Paramet

- **tas** A task from Task ager
- **met** Met to be validate
- **kwa** Info for action.

Raises

Un-

supportedDriv tension if meth can not be map

to the supported interfaces.

Raises

InvaliParameterValu if kwa does not contain

method.

Raises

MissingParameterValu

class i

Base tup

metadat

Alia for field number 1

method

Alia

for field num ber 0

`ironic.`

A decorator to cach bios settings after run-

ning the function.

Paramet

`fun`

Function or meth to wra

`ironic.`

Decorator for clea ing step

Clea ing step may be used in man ual or automate clea ing.

For automate clea ing, only step with priorities grea than 0 are used. These steps are ordered by priority from highest value to lowest value. For steps with the same priority, they are ordered by driver interface priority (see conductor.steps.CLEANING_INTERFACE_PRIORITY). execute_clean_step() will be called on each step.

For manual clea ing, the clea step will be executed in a similar fashion to automated cleaning, but the steps and order of execution must be explicitly specified by the user when invoking the cleaning API.

Decorate clea step mus take as the only positiona argument, a TaskManager object. Clean steps used in manual cleaning may also take keyword variable arguments (as described in argsinfo).

Clea step can be either syn chro or asyn chro If the step is synchronous, it should return *None* when finished, and the conductor will continue on to the next step. While the clean step is executing, the node will be in *states.CLEANING* provision state. If the step is asynchronous, the step should return *states.CLEANWAIT* to the conductor before it starts the asynchronous work. When the step is complete, the step should make an RPC call to *continue_node_clean* to move to the next step in cleaning. The node will be in *states.CLEANWAIT* provision state during the asynchronous work.

Examples

```
↪CONF option
```

(continues on next page)

(continued from previous page)

```
↪       {'size': {'description': 'size of widget (MB)',
```

(continues on next page)

(continued from previous page)

Paramet

- **`pri`** an integer priority, shou be a CON option
- **`abo`** Boo valu Whe the clea step

is
abor
or
not;
default

to False.

- **arg**
a
dictionary
of
keyword
arguments
whe
key

is the name of the argument and value is a dictionary as follows:

```
↪ possible values.
```

(continues on next page)

(continued from previous page)

```
↪required.  If so, it must be specified in

↪request; false if it is optional.
```

- **req**
  Whe
  this
  step
  re-
  quir
  the
  ram
  to
  be
  run-
  ning
  Shou

be set to False for purely out-of-band steps.

Raises
Inv
if
any
of
the
ar-

gu-
men
are
in-
valic

ironic.

Dec-
o-
ra-
tor
for
de-
ploy
men
step

Only
step
with
pri-
or-
i-
ties
grea
than
0
are
used
Thes
step

are ordered by priority from highest value to lowest value. For steps with the same priority, they are ordered by driver interface priority (see conductor.steps.DEPLOYING_INTERFACE_PRIORITY). execute_deploy_step() will be called on each step.

Dec-
o-
rated
de-
ploy
step
mus
take
as
the
only
po-
si-
tiona

argument, a TaskManager object.

De-
ploy

step can be either synchro or asyn chro If the

step is synchronous, it should return *None* when finished, and the conductor will continue on to the next step. While the deploy step is executing, the node will be in *states.DEPLOYING* provision state. If the step is asynchronous, the step should return *states.DEPLOYWAIT* to the conductor before it starts the asynchronous work. When the step is complete, the step should make an RPC call to *continue_node_deploy* to move to the next step in deployment. The node will be in *states.DEPLOYWAIT* provision state during the asynchronous work.

Examples

Paramet

- **`pri`** an integer (>=(priority; used for determining the order in which the step is run in the deployment process.
- **`arg`** a dictionary of keywor argumen whe key is the name of the argument and value is a dictionary as follows:

(continues on next page)

(continued from previous page)

```
↪required.  If so, it must be specified in
↪request; false if it is optional.
```

Raises

Inv

if any of the argumen are

in-
valic

`ironic.`

`ironic.`

ironic.drivers.drac module

DRA
Driv
for
remote
system
man
age-
men
using
Dell
Re-
mote Access Card.

class i
Base
iro
dri
gen
Gen

in-
te-
grat
Dell
Re-
mote
Ac-
cess
Con
troll
hard
ware
type

propert

List of sup-porte bios in-ter-face

propert
List of sup-porte boot in-ter-face

propert
List of sup-porte in-spec in-ter-face

propert
List of sup-porte man age-men in-ter-face

propert
List of sup-porte pow in-ter-face

propert
List

of sup- port raid in- ter- face

propert

List of sup- port ven- dor in- ter- face

ironic.drivers.fake_hardware module

Fake hard ware type

class i

Base *iro dri har Abs*

Fake hard ware type

This hard ware type is spec case in the drive fac- tory to bypass compatibility verification. Thus, supported_* methods here are only for calculating the defaults, not for actual check.

All fake implemen tation are still expect to be enabled in the configuration.

propert
List of clas of support bios interface

propert
List of clas of support boot interface

propert
List of clas of support console

interface

propert
List of clas of supporte deploy interface

propert
List of clas of supporte inspec interface

propert
List of clas of supporte managemen interface

propert
List of supporte networl inter-

face

propert

List of class of supporte pow interface

propert

List of class of supporte raid interface

propert

List of class of supporte rescue interface

propert

List of class of supporte storage interface

propert

List

of clas of sup-port res-cue in-ter-face

ironic.drivers.generic module

Gen hard ware type

class i

Base

irc dri har Abs

Ab-strac base clas rep-re-sent-ing gene hard ware

This clas pro-vide rea-son-able de-fault for all of the

in-

terfaces.

propert

List of supporte boot interface

propert

List of supporte deploy interface

propert

List of supporte inspec interface

propert

List of supporte network interface

propert

List of supporte raid inter-

face

propert

List of supported rescue interface

propert

List of supported storage interface

class i

Base

iro
dri
gen
Gen

Hard ware type that uses man ual pow and boot man age men

Using this hard ware type as sum that

an operator manages reboot and setting boot devices manually. This hardware type should only be used when no suitable hardware type exists in ironic, or the existing hardware type misbehaves for any reason.

propert

List of suppor man agemen interface

propert

List of suppor pow interface

propert

List of suppor vendor interface

ironic.drivers.hardware_type module

Abstrac base class for all hard ware type

class i

Base obj

Abstrac base clas for all hard ware type

Hard ware type is a family of hard ware supporting the same set of interfaces from the ironic standpoint. This can be as wide as all hardware supporting the IPMI protocol or as narrow as several hardware models supporting some specific interfaces.

A hard ware type defines an ordere list of supporte implementations for each driver interface (power, deploy, etc).

get_pro

Get the prop erties of the hard ware type

Note that this returns prop erties for the default inter-

face of each type, for this hardware type. Since this is not node-aware, interface overrides cant be detected.

Returns

dictionary of <pro erty nam description> entries

support

When hard ware is supported by the com munity.

propert

List

of sup- port bios in- ter- face

abstrac
List of sup- port boot in- ter- face

propert
List of sup- port con- sole in- ter- face

abstrac
List of sup- port de- ploy in- ter- face

propert
List of sup- port in- spec in- ter- face

abstrac
List of

sup-
port
man
age-
men
in-
ter-
face

propert

List
of
sup-
port
net-
work
in-
ter-
face

abstrac

List
of
sup-
port
pow
in-
ter-
face

propert

List
of
sup-
port
raid
in-
ter-
face

propert

List
of
sup-
port
res-
cue
in-
ter-
face

propert

List
of

supporte storage interface

propert

List of supporte vendor interface

ironic.drivers.ibmc module

iBM Driv for managing HUA V5 series rack serv such as

2288H V5, CH121 V5.

class i

Base *iro dri gen Gen*

Hua iBM hardware type

propert

List of supported manage- ment in- ter- face

propert
List of supported pow in- ter- face

propert
List of supported raid in- ter- face

propert
List of supported ven- dor in- ter- face

ironic.drivers.ilo module

iLO Driv for managing HP Pro-

liant
Gen
and
abov
serv

class i

Base
irc
dri
ilc
Ilc

iLO
hard
ware
type

iLO
hard
ware
type
is
targete
for
iLO
base
Proliant
Gen
serv

propert

List
of
supporte
boot
interface

propert

List
of
supporte
man
age-
men
in-

terface

propert

List of supporte raid interface

class i

Base

iro
dri
gen
Gen

iLO hard ware type

iLO hard ware type is targete for iLO 4 base Proliant Gen

and Gen9 servers.

propert

List of supporte bios interface

propert

List of

sup-
port
boot
in-
ter-
face

propert

List
of
sup-
port
con-
sole
in-
ter-
face

propert

List
of
sup-
port
in-
spec
in-
ter-
face

propert

List
of
sup-
port
man
age-
men
in-
ter-
face

propert

List
of
sup-
port
pow
in-
ter-
face

propert

List
of

sup-
port
pow
in-
ter-
face

ironic.drivers.intel_ipmi module

class i
Base
irc
dri
ipm
IPM

In-
tel
IPM
hard
ware
type

Uses
`ipm`
to
im-
ple-
men
pow
and
man
age-
men
Pro-
vide
se-
rial console implementations via `shellinabox` or `socat`. Supports Intel SST-PP feature.

propert
List
of
sup-
port
man
age-
men
in-
ter-
face

ironic.drivers.ipmi module

Hard- ware type for IPM (us- ing ip- mi- tool)

class i

Base

iro dri gen Gen

IPM hard ware type

Uses `ipm` to im- ple- men pow and man age- men Pro- vide se-

rial console implementations via `shellinabox` or `socat`.

propert

List of sup- port con- sole in- ter- face

propert

List of supporte man agemen interface

propert

List of supporte pow interface

propert

List of supporte vendor interface

ironic.drivers.irmc module

iRM Driv for man aging FUJITS PRI BX S4 or RX S8 generation of FUJITSU PRIMERGY servers, and above servers.

class i

Base
irc
dri
gen
Gen

iRM
hard
ware
type

iRM
hard
ware
type
is
targete
for
FU-
JITS
PRI
serv
whic
have

iRMC S4 management system.

propert
List
of
sup-
port
bios
interface

propert
List
of
sup-
port
boot
interface

propert
List
of
sup-
port
console

in-
ter-
face

propert
List
of
sup-
port
in-
spec
in-
ter-
face

propert
List
of
sup-
port
man
age-
men
in-
ter-
face

propert
List
of
sup-
port
pow
in-
ter-
face

propert
List
of
sup-
port
raid
in-
ter-
face

ironic.drivers.redfish module

class i

Base
iro
dri
gen
Gen

Red-
fish
hard
ware
type

propert

List
of
sup-
port
bios
in-
ter-
face

propert

List
of
sup-
port
boot
in-
ter-
face

propert

List
of
sup-
port
pow
in-
ter-
face

propert

List
of
sup-
port
man
age-

men-
in-
ter-
face

propert
List
of
sup-
porte
pow
in-
ter-
face

propert
List
of
sup-
porte
raid
in-
ter-
face

propert
List
of
sup-
porte
ven-
dor
in-
ter-
face

ironic.drivers.snmp module

SNM
hard
ware
type

class i
Base
iro
dri
gen
Gen

SNM
Harc

ware type

propert List of supporte management interface

propert List of supporte power interface

ironic.drivers.utils module

class i Base *iro dri bas Ven*

Wrapper around multiple VendorInterface

get_pro Return the

properties from all the VendorInterfaces.

Returns

a dictionary of <pro erty_ entries

validat

Call validate on the appropriate interface only.

Raises

UnsupportedDriv tension if meth can not be map to the supported interfaces.

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
meth
is
in-
valic

Raises
Miss
ing-
Pa-
ram-
e-
ter-
Valu
if
miss
ing
meth
or
pa-

rameters in kwargs.

`ironic.`

Add
ca-
pa-
bil-
ity
to
node
ca-
pa-
bil-
i-
ties
prop

erty.

If
ca-
pa-
bil-
ity

is already pres then a duplicate

entry will be added.

Paramet

- **tas** Task object.
- **cap** Capability key.
- **val** Capability valu

`ironic.`

Pars the capabilities strin into a dictionary

Paramet

cap the

capabilities of the node as a formatted string.

Raises

InvalidParameterValu if capabilities is not an string or has a malformed value

`ironic.`

Collect and store the system logs from the IPA ramo

Collect and store the system logs from the

IPA
ram
This
meth

makes a call to the IPA ramdisk to collect the logs and store it according to the configured storage backend.

Paramet

- **nod**
  A
  node
  ob-
  ject.
- **lab**
  A
  strin
  to
  la-
  bel
  the
  log
  file
  such
  as
  a
  clea

step name.

`ironic.`

En-
sure
boot
from
cor-
rect
de-
vice
if
per-
sis-
tent
is

True

If
ipmi
is
True
and

is_n set to boot from correct device

else unset is_next_boot_persistent field.

Paramet

- **tas** Nod object.
- **dri** Nod driv

ironic.

Set persistent boot device to driv

If persistent is True set persistent_ field to the boot

device and reset persistent to False, else set is_next_boot_persistent to False.

Paramet

- **tas** Task object.
- **dev** Boo device
- **per** Whe next boot is persistent or not.

`ironic.` Returns capability valu from node capabili-

ties property.

Paramet

- **nod** Nod object.
- **cap** Capabil-

ity key.

Returns
Capability valu

If capability is not pres

then return None

ironic.

Get all MA

addres for the port

belonging to this

tasks node.

Paramet

tas

a Task

ager

instan

containing the

node

to act

on.

Returns

A

list of MA ad- dres in the for- mat xx:x

`ironic.`

Con struc the log file nam

Paramet

- **nod** A node ob- ject.
- **lab** A strin to la- bel the log file such as a clea

step name.

Returns

The log file nam

`ironic.`

Chec if

node need prep ing ram

Paramet

nod

Nod to chec for

Returns

True if need to prepare ram otherwise Fals

`ironic.`

Remov - and : characters and lowercase

the MAC string.

Paramet

mac

MA address to normalize.

Returns

Nor-

mal-
ized
MA
ad-
dres
strin

`ironic.`

Stor
the
ram
logs

This
meth
stor
the
ram
logs
ac-
cord
ing
to
the
con-
fig-
ured

storage backend.

Paramet

- **nod**
A
nod
ob-
ject.
- **log**
A
gzip
and
base
en-
code
strin
con-
tain-
ing
the
logs

archive.

- **lab** A strin to label the log file such as a clea step name.

Raises
OSError if the directory to save the logs cannot be created.

Raises
IOError whe the logs cant be save to the local file system.

Raises
Swi Opera-

tionl if any op- er- a- tion with Swi

fails.

ironic.drivers.xclarity module

XCl ity Driv and sup- port- ing meta clas

class i

Base *irc dri gen Gen*

XCl ity hard ware type

propert

List of sup- port man age- men in- ter- face

propert

List

of sup- port pow in- ter- face

Module contents

ironic.objects package

Submodules

ironic.objects.allocation module

class i

Base

irc

obj

bas

Irc

osl

bas

Ver

VERSION

propert

propert

create

Cre- ate a Al- lo- ca- tion reco in the DB.

Parame

con

Security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Allocation(context)

Raises

Allocation pli- cate- Nam Allocation- Alread ists

propert

dbapi =

destro

Dele the Allocation from the DB.

Paramet

con

Security context. NOT This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Allocation(context)

Raises

Allocation-Not-Foun

`propert`

`fields`

`classme`

Find an allocation by its ID, UUI or nam

Paramet

- `all` The ID, UUI or nam of an alloca-

tion.

- **con**
  Security context

Returns
An *All* object.

Raises
InvalidIdentity

classme
Find an allocation by its integer ID.

Paramet

- **cls** the *All*
- **con** Security context
-

all
The ID of an allocation.

Returns
An *All* object.

Raises
AllocationNotFoun

classme
Find an allocation base by its nam

Paramet

- **cls** the *All*
- **con** Security context
- **nam** The

nam of an allocation.

Returns
An *All* object.

Raises
Allocation-NotFour

classme
Find an allocation by its UUI

Parame

- **cls** the *All*
- **con** Security context
- **uui** The UUI of an

allocation.

Returns
An *All* object.

Raises
Allocation-Not-Foun

propert

propert

classme

Return a list of Allocation objects

Paramet

- **cls** the *All*
- **con** Security context.
-

fil Filters to apply.

- **lim** Maxi-mun num-ber of re-sour to re-turn in

a single result.

- **mar** Pagi-na-tion marl for large data sets.
- **sor** Col-umn to sort re-sults by.
- **sor** Di-rec-tion to sort. asc

or desc

Returns
A list of *All* object.

Raises
Invalid Parameter Valu

propert

propert

propert

refresh
Load updates for this Allocation.

Load an allocation with the same uuid from the data and checks for updated attributes. Updates are applied from the loaded allocation column by column, if there are any updates.

Parame

`con`

Security context. NO This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Allocation(context)

Raises

AllocationNotFoun

propert

save (*co*

Save update to this Allocation.

Update will be mad column by column base on the result of self.what_changed().

Paramet

con

Security context. NOT This shou only be usec internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Allocation(context)

Raises

AllocationNotFour AllocationI plicateNam

propert

propert

propert

propert

class i

Base *irc obj not Not*

No- ti- fi- ca- tion whe ironi cre- ates, up- date or dele an allocation.

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Base *iro obj not Not*

SCHEMA

VERSION

propert

propert

propert

fields

propert

propert

propert

propert

propert

propert

propert

propert

propert

ironic.objects.base module

Iron
com
mon
internal
object
mod

class i

Base
`osl`
`bas`
`Ver`

Base
class

and object factory.

This form the base of all objects that can be remot or

instantiated via RPC. Simply defining a class that inherits from this base class will make it remotely instantiatable. Objects should implement the necessary get classmethod routines as well as save object methods as appropriate.

OBJ_PRO

OBJ_SEI

as_dict

Return the object represent as a dict.

The returned object is JSO seria

convert

Con

vert this object to the target version

Convert the object to the target version. The target version may be the same, older, or newer than the version of the object. This is used for DB interactions as well as for serialization/deserialization.

The remov flag is used to distinguish these two case

1) For serialization/ we need to re-

mov the unavailable fields, because the service receiving the object may not know about these fields. remove_unavailable_fields is set to True in this case.

2) For DB interactions we need to set the unavailable fields to their appropriate values so that these fields are saved in the DB. (If they are not set, the VersionedObject magic will not know to save/update them to the DB.) remove_unavailable_fields is set to False in this case.

convert does the actual work

Parame

- **`tar`** the desired version of the object
- **`rem`** True to remove field

that are unavai able in the

target version; set this to True when (de)serializing. False to set the unavailable fields to appropriate values; set this to False for DB interactions.

do_vers

Cha the object to the version need for the data

If need this chan the object (mo ifies object field to

be in the correct version for saving to the database.

The version used to save the object in the DB is determined as follows:

- If the object is pinn we save the object in the pinned version. Since it is pinned, we must not save in a newer version, in case a rolling upgrade is happening and some services are still using the older version of ironic, with no knowledge of this newer version.
- If the object isnt pinn we save the object in the latest version.

Becaus the object may be converte to a different object version, this method must only be called just before saving the object to the DB.

Returns
a dictionary of char field and their new values (cou

be an empty dictionary). These are the fields/values of the object that would be saved to the DB.

`fields`

`classme`
Returns the target version for this object.

This is the version in whic the object shou be manip-

ulated, e.g. sent over the wire via RPC or saved in the DB.

Returns
if pinn returns

the version of this object corre-

sponding to the pin. Otherwise, returns the version of the object.

Raises

ovo_

obj_ref

Applies updates for objects that inherit from base

Chec for updates attribu in an object. Updates are ap-

plied from the loaded object column by column in comparison with the current object.

classme

Returns whe this object supports

a particular

version.

Che the requesversion agai the objects target version

The target version may not be the latest version during an upgrade, when object versions are pinned.

Paramet

ver

A tuple representing the version to chec

Returns

Whe the version is supporte

Raises

ovo_

class i

Base

osl

bas

`Obj`

as_dict

Return the object represented as a dict.

The returned object is JSO seria

class i

Base

`osl`

`bas`

`Ver`

registr

class i

Base

`osl`

`bas`

`Ver`

OBJ_BAS

alias of

`iro`

`obj`

`bas`

`Iro`

seriali

Serialize

the entity.

This serializes the entity so that it can be sent over e.g. RPC. A serialized entity for an IronicObject is a dictionary with keys: ironic_object.namespace, ironic_object.data, ironic_object.name, ironic_object.version, and ironic_object.changes.

We assum that the clien (iron API is alway talking to a server (ironic-conductor) that is running the same or a newer release than the client. The client doesnt need to downgrade any IronicObjects when sending them over RPC. The server, on the other hand, will need to do so if the server is pinned and the target version of an IronicObject is older than the latest version of that Object.

(Internall the service deal with the latest version of

objects so we know that these objects are always in the latest versions.)

Paramet

- **con** security context
- **ent** the entity to be serialized, may be an

IronicObject

Returns
the serialized entity

Raises
ovo_ (via .get_

`ironic.`
Return the maximum version

in the list.

Paramet

ver

a list of (stri ver- sion as- sum to have at leas

one entry

Returns

the max i- mun ver- sion (stri

ironic.objects.bios module

class i

Base

iro obj bas Iro

VERSION

create

Cre- ate a BIO Set- ting reco in DB.

Paramet

`con`

Security context. NOT This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: BIOSSetting(context)

Raises

Nod Not- Foun if the node id is not foun

Raises

BIO Setting Al- read ists if the setting reco al- read

exists.

`propert`

`dbapi =`

`classme`

Dele a

BIO Setting base on its node and nam

Paramet

- **con** Security context.
- **nod** The node id.
- **nam** BIO setting nam to be dele

Raises
Nod Not-Foun if the node id is not foun

Raises
BIO Setting-Not-

Foun if the bios set- ting nam is not

found.

fields

classme

Get a BIO Set- ting base on its node and nam

Parame

- **con** Se- cu- rity con- text.
- **nod** The node id.
- **nam** BIO set- ting nam to be re- triev

Raises
Nod
Not-
Foun
if
the
node
id
is
not
foun

Raises
BIO
Set-
ting-
Not-
Foun
if
the
bios
set-
ting
nam
is
not

found.

Returns
A
:clas
ob-
ject.

propert

propert

save (*co*
Save
BIO
Set-
ting
up-
date
in
DB.

Paramet
con
Se-
cu-

rity
con-
text.
NOT
This
shou
only
be
used
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: BIOSSetting(context)

Raises
Nod
Not-
Foun
if
the
node
id
is
not
foun

Raises
BIO
Set-
ting-
Not-
Foun
if
the
bios
set-
ting
nam
is
not

found.

propert

propert

class i
Base
irc
obj
bas
Irc
irc

obj
bas
Iro

VERSION

classme

Create a list of BIO Setting reco in DB.

Parame

- **con**
  Security context. NO This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: BIOSSetting(context)
- **nod**
  The node id.
- **set**
  A list of bios set-

tings

Raises
Nod Not- Foun if the node id is not foun

Raises
BIO Set- tin- gAl- read ists if any of the set- ting reco

already exists.

Returns
A list of BIO Set- ting ob- jects

propert

dbapi =

classme
Dele BIO Set- tings base on node

and nam

Paramet

- **con** Security context.
- **nod** The node id.
- **nam** List of BIO setting nam to be dele

Raises Nod Not-Foun if the node id is not foun

Raises BIO Setting-Not-Foun if any of BIO

set-
ting
fails
to

delete.

fields

classme

Get
BIO
Set-
ting
base
on
node

Parame

- **con**
Se-
cu-
rity
con-
text.
- **nod**
The
node
id.

Raises

Nod
Not-
Foun
if
the
node
id
is
not
foun

Returns

A
list
of
BIO
Set-
ting

objects

`propert`

`classme`

Save a list of BIO Setting update in DB.

Parame

- **`con`** Security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: BIOSSetting(context)
- **`nod`** The node id.
- **`set`** A list of bios settings

Raises

Nod
Not-
Foun
if
the
node
id
is
not
foun

Raises
BIO
Set-
ting-
Not-
Foun
if
any
of
the
bios
set-
ting
nam

is not found.

Returns
A
list
of
BIO
Set-
ting
ob-
jects

classme
Re-
turns
lists
of
cre-
ate/u
set-
tings

This
meth
sync
with
bios
data

table and sorts out four lists of

create/update/delete/unchanged settings.

Parame

- **con** Security context.
- **nod** The node id.
- **set** BIO settings to be sync

Returns
A 4-tuple of lists of BIO settings to be created

updated, deleted and unchanged.

propert

ironic.objects.chassis module

class i

Base

iro

obj

bas

Iro

osl

bas

Ver

VERSION

create

Create a Chas sis reco in the DB.

Colu wise up-date will be mad base on the re-sult of self.

If target_power_state is provided, it will be checked against the in-database copy of the chassis before updates are made.

Paramet

con

Se-cu-rity con-text. NOT This

shou
only
be
used
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Chassis(context)

propert

dbapi =

propert

destroy
Dele
the
Cha
sis
from
the
DB.

Paramet
con
Se-
cu-
rity
con-
text.
NOT
This
shou
only
be
used
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Chassis(context)

propert

fields

classme
Find
a
chas
sis
base

on its id or uuid and return

a Chassis object.

Parame

- **con** Security context
- **cha** the id *or* uuid of a chassis.

Returns a *Cha* object.

classme

Find a chassis based on its integer ID and re-

turn a Chassis object.

Paramet

- **cls** the *Cha*
- **con** Security context
- **cha** the ID of a chas sis.

Returns a *Cha* object.

classme Find a chassis base on UUI and return a *Cha* ob-

ject.

Paramet

- **cls** the *Cha*

- **con**
  Security context
- **uui**
  the UUI of a chassis.

Returns
a *Cha* object.

propert

classme

Return a list of Chassis objects

Paramet

- **cls**
  the *Cha*
- **con**
  Security context.
-

lim max- i- mum num- ber of re- sour to re- turn in

a single result.

- **mar** pag- i- na- tion mark for larg data sets.
- **sor** col- umn to sort re- sults by.
- **sor** di- rec- tion to sort. asc or desc

Returns
a list of *Cha*

object.

refresh

Loa and applies updates for this Chas sis.

Loa a *Cha* with the sam uuid from the data and chec for updated attributes. Updates are applied from the loaded chassis column by column, if there are any updates.

Paramet

con Security context. NO This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Chassis(context)

save(*co*

Save updat to

this Cha sis.

Up- date will be mad col- umn by col- umn base on the re-

sult of self.what_changed().

Paramet

con

Se- cu- rity con- text. NO This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Chassis(context)

propert

propert

class i

Base *irc obj not Not*

No- ti- fi-

ca-
tion
emit
ted
whe
iron
cre-
ates,
up-
date
dele

a chassis.

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

SCHEMA

VERSION

propert

propert

propert

fields

propert

propert

ironic.objects.conductor module

class i

Base
iro
obj
bas
Iro
osl
bas
Ver

VERSION

propert

propert

dbapi =

propert

fields

classme

Get
a
Con
duc-
tor
reco
by
its
host
nam

Paramet

- **cls** the *Con*
- **con** Security context
- **hos** the host nam on whic a Conductor is running
- **onl** Spec ify the expect `onl` field valu for the conductor to be retrieved. The `online` field is ignored if this value is set to None.

Returns a *Con* object.

propert

propert

classme

Return a list of Conductor objects

Paramet

- **cls** the *Con*
- **con** Security context.
- **lim** maximum number of resour to return in a single result.
- **mar** pagina-

tion mark for large data sets.

- **sor** column to sort results by.
- **sor** direction to sort. asc or desc

Returns a list of *Con* object.

refresh Load and applies updates for this Conductor.

Load a *Con* with the

sam uuid from the data and chec for updated attributes. Updates are applied from the loaded chassis column by column, if there are any updates.

Parame

`con` Security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Conductor(context)

`classme`

Register an active conductor with the cluster.

Parame

- **`cls`** the *`Con`*
- **`con`**

Security context

- **hos** the host nam on whic the conductor will run
- **dri** the list of driv enable in the conductor
- **con** conductor grou to join, used for node affin ity.
- **upd** Whe false reg-

istration will raise an exception

when a conflicting online record is found. When true, will overwrite the existing record. Default: False.

Raises
Con ductorA read istere

Returns
a *Con* object.

registe
Register hard ware interface with the conductor.

Paramet
`int`
List of interface to register, each en-

try

should be a dictionary conaining hardware_type, interface_type, interface_name and default, e.g. {hardware_type: hardware-type, interface_type: deploy, interface_name: iscsi, default: True}

save(*co*

Save is not supported by Con ductor objects

touch(*c*

Touc this conductors DB reco mark ing it as up-to-

date.

unregis

Remov this conductor from the service reg- istry

unregis

Unregister all

hardware interface for this conductor.

propert

ironic.objects.deploy_template module

class i

Base *iro obj bas Iro* osl bas Ver

VERSION

create

Create a Deploy plate reco in the DB.

Parame

con security context. NOT This shou only

be
used
internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: DeployTemplate(context).

Raises
De-
ploy
plat-
eDu
pli-
cate-
Nam
if
a
de-
ploy
tem-
plate
with the same name exists.

Raises
De-
ploy
plate
read
ists
if
a
de-
ploy
tem-
plate
with
the
same UUID exists.

propert

dbapi =

destro
Dele
the
De-
ploy
plate
from
the
DB.

Paramet

con

security context. NO This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: DeployTemplate(context).

Raises

Deploy plate Four if the deploy template no long appears in the database.

propert

fields

classme

Find a deploy template base on its integer ID.

Paramet

- **`con`** security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: DeployTemplate(context).
- **`tem`** The ID of a deploy template

Raises

Deploy plate Foun if the deploy template no long appears in the database.

Returns

a *Dep* object.

`classme`

Find a deploy template based on its nam

Paramet

- **con** security context. NOT This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: DeployTemplate(context).

- **nam** The nam of a deploy template

Raises

Deploy plate Foun if the deploy template no long ap-

pears in the database.

Returns
a *Dep* ob- ject.

classme
Find a de- ploy tem- plate base on its UUI

Paramet

- **con** se- cu- rity con- text. NOT This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: DeployTemplate(context).

- **uui** The UUI of a de- ploy tem- plate

Raises

De- ploy plate Four if the de- ploy tem- plate no long ap-

pears in the database.

Returns

a *Dep* ob- ject.

propert

classme

Re- turn a list of De- ploy plate ob- jects

Paramet

- **con** se- cu- rity con- text. NOT This shou only be used

in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: DeployTemplate(context).

- **lim** maximum number of resources to return in

a single result.

- **mar** pagination mark for large data sets.
- **sor** column to sort results by.
- **sor** direction to sort. asc or desc

Returns

a list of *Dep* ob- jects

classme

Re- turn a list of De- ploy plate ob- jects matc ing a set of names.

Paramet

- **con** se- cu- rity con- text. NOT This shou only be used in- ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: DeployTemplate(context).
- **nam** a list of nam to fil- ter by.

Returns
a list of *Dep* objects

`propert`

`refresh`
Loa update for this deploy template

Loa a deploy template with the same uuid from the data and checks for updated attributes. Updates are applied from the loaded template column by column, if there are any updates.

Parame
`con` Security context. NOT This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Port(context)

Raises

De-
ploy
plate
Four
if
the
de-
ploy
tem-
plate
no
long
ap-

pears in the database.

save (*co*

Save
up-
date
to
this
De-
ploy
plate

Colu
wise
up-
date
will
be
mad
base
on
the
re-
sult
of
self.

Parame

con

Se-
cu-
rity
con-
text.
NOT
This

shou
only
be
use
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: DeployTemplate(context)

Raises
De-
ploy
plat-
eDu
pli-
cate
Nam
if
a
de-
ploy
tem-
plat

with the same name exists.

Raises
De-
ploy
plat
Fou
if
the
de-
ploy
tem-
plat
does
not
ex-

ist.

propert

propert

propert

class i

Bas
irc
obj

not
Not

Notification emit ted on de ploy tem plate API op erations.

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

SCHEMA

VERSION

propert

propert

fields

propert

propert

propert

propert

ironic.objects.deployment module

class i

Base
irc
obj
bas
Irc
`osl`
`bas`
`Ver`

VERSION

create

Create
a
De-
ploy
men

Up-
date
the
cor-
re-
spon
ing
node
un-
der

the hood

Paramet

- **con** Security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Deployment(context)

- **nod** Nod object for deploy men

Raises
Instanc Associated Nod Associated Nod Not-Found

propert

dbapi =

destroy

Dele the De- ploy men

Up- date the cor- re- spon ing node un- der the hood

Parame

- **con** Se- cu- rity con- text. NO This shou only be used in- ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Node(context)
- **nod** Nod ob- ject for de- ploy men

fields

classme

Find a deploymen base by its node UUI

Paramet

- **cls** the *Dep*
- **con** Security context
- **nod** The UUI of a corresponding node

Returns
An *Dep* object.

Raises
Nod Not-Foun

classme
Find a deploy

men
by
its
UUI

Parame

- **cls**
  the
  Dep
- **con**
  Se-
  cu-
  rity
  con-
  text
- **uui**
  The
  UUI
  of
  a
  de-
  ploy
  men

Returns
An
Dep
ob-
ject.

Raises
In-
stan-
ceN
Foun

propert

propert

instanc

instanc

propert

classme

Return a list of Deploymen objects

Parame

- **cls** the *Dep*
- **con** Security context.
- **fil** Filters to apply.
- **lim** Maximum number of resour to return in

a single result.

-

mar Pagination mark for large data sets.

- **sor** Column to sort results by.
- **sor** Direction to sort. asc or desc

Returns A list of *Dep* object.

Raises InvalidParameterValu

node_ma

propert

propert

refresh

Re-
fresh
the
ob-
ject
by
re-
fetch
from
the
DB.

Parame

con

Se-
cu-
rity
con-
text.
NOT
This
shou
only
be
used
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Node(context)

propert

propert

propert

propert

propert

propert

ironic.objects.fields module

class i

Base
osl
fie
Boo

class i

Base
osl
fie
Dat

class i

Base
osl
fie
Enu

class i

Base
osl
fie
Fie

static

This is calle to coerce (if possible) a valu on assignment.

This meth shou convert the valu give into the

des-
ig-
nate
type

or throw an exception if this is not possible.

Param:
The
Ver-
sion
dOb
ject
on
whic
an
at-
tribu
is
be-
ing

set

Param:
The
nam
of
the
at-
tribu
be-
ing
set

Param:
The
valu
be-
ing
set

Returns
A
prop
type
valu

class i
Base
`osl`
`fie`
`Aut`

AUTO_TY

class i

Base
osl
fie
Int

class i

Base
osl
fie
Aut

AUTO_T

class i

Base
osl
fie
Lis

class i

Base
osl
fie
Lis

class i

Base
osl
fie
Fie

static

This is calle to coerce (if possible) a valu on assignment.

This

meth shou con- vert the valu give into the des- ig- nate type

or throw an exception if this is not possible.

Param:c
The Ver- sion dOb ject on whic an at- tribu is be- ing

set

Param:a
The nam of the at- tribu be- ing set

Param:v
The valu be- ing set

Returns
A prop type

valu

class i

Base

osl

fie

Aut

AUTO_TY

class i

Base

osl

fie

Enu

ALL =

CRITICA

DEBUG =

ERROR =

INFO =

WARNING

class i

Base

osl

fie

Bas

AUTO_TY

class i

Base

osl

fie

Enu

ALL =

END =

ERROR =

START =

SUCCES

class i
Base
`osl`
`fie`
`Bas`

AUTO_TY

class i

Base
`osl`
`fie`
`Obj`

class i
Base
`osl`
`fie`
`Str`

static
This is calle to coerce (if possible) a valu on assignment.

This meth shou convert the valu give into the designate type

or throw an exception if this is not possible.

Param: The VersiondOb ject on whic an attribu is being

set

Param: The nam of the attribu being set

Param: The valu being set

Returns A prop type valu

class i
Base

`osl`
`fie`
`Str`

class `i`
Base
`osl`
`fie`
`Str`

Custom Strin Fiel object that allows for func tions as default

In som case we need to allow for dynam default base on configuration options, this StringField object allows for a function to be passed as a default, and will only process it at the point the field is coerced

AUTO_TY

class `i`
Base
`osl`
`fie`
`UUI`

ironic.objects.indirection module

class i

Base

`osl`

`bas`

`Ver`

object_

Perform an action on a Version dOb ject instan

Whe indirection_ is set on a Version dOb ject (to a class implementing this interface), method calls on remotable methods will cause this to be executed to actually make the desired call. This often involves performing RPC.

Parame

- **con** The context with whic to perform the action

tion

- **obj** The object instance on which to perform the ac-
- **obj** The name of the action method to call
- **arg** The positional arguments to the action meth
- **kwa** The keyword

ar-
gu-
men
to
the
ac-
tion
meth

Returns

The
re-
sult
of
the
ac-
tion
meth

object_

Per-
form
a
back
port
of
an
ob-
ject
in-
stan

This
meth
is
ba-
si-
cally
just
like
ob-
ject_
but
in-
stead
of

providing a specific target version for the toplevel object and relying on the service-side mapping to handle sub-objects, this sends a mapping of all the dependent objects and their client-supported versions. The server will backport objects within the tree starting at objinst to the versions specified in object_versions, removing objects that have no entry. Use obj_tree_get_versions() to generate this mapping.

NOT

This was not in the initial spec for this interface

so the base class raises NotImplementedError if you dont implement it. For backports, this method will be tried first, and if unimplemented, will fall back to object_backport().

Parame

- `con` The context with whic to perform the back port
- `obj` An instance of a VersiondObject to be back

ported

- `obj` A dict

of {ob- j- nam ver- sion map ping

object_

Dep re- cate sinc ver- sion 0.10

Use *obj* in- stea

Per- form an ac- tion on a Ver- sion dOb ject class

Whe in- di- rec- tion_ is set on a Ver- sion dOb ject (to

a class implementing this interface), classmethod calls on remotable_classmethod methods will cause this to be executed to actually make the desired call. This usually involves performing RPC.

Paramet

- **con** The context with whic to perform the action
- **obj** The registry nam of the object
- **obj** The nam of the action meth to call
- **obj** The (remote version of the object on whic the

action is being taken

- **arg**
  The positional arguments to the action meth
- **kwa**
  The keyword arguments to the action meth

Returns
The result of the action meth whic may (or may not)

be an instance of the implementing VersionedObject class.

object_

Perform an action on a VersiondObject clas

Whe indirection is set on a VersiondObject (to

a class implementing this interface), classmethod calls on remotable_classmethod methods will cause this to be executed to actually make the desired call. This usually involves performing RPC.

This differs from object in that it is provide with ob-

ject_versions, a manifest of client-side object versions for easier nested backports. The manifest is the result of calling obj_tree_get_versions().

NO This was not in the initial spec for

this interface

so the base class raises NotImplementedError if you dont implement it. For backports, this method will be tried first, and if unimplemented, will fall back to object_class_action(). New implementations should provide this method instead of object_class_action()

Parame

- `con` The context with whic to perform the action
- `obj` The registry nam of the object
- `obj` The nam of the action meth to call
- `obj` A dict of {ob-

jnam version mapping

- **arg** The positiona argumen to the action meth
- **kwa** The keyword argumen to the action meth

Returns The result of the action meth whic may (or may not)

be an instance of the implementing VersionedObject class.

ironic.objects.node module

class i

Base
iro
obj
bas
Iro
osl
bas
Ver

VERSION

propert

as_dict

Return the object represent as a dict.

The return object is JSO seria

propert

propert

propert

propert

propert

propert

propert

propert

propert

create

Create a Nod reco in the DB.

Colu wise update will be mad base on the result of self.

If target_power_state is provided, it will be checked against the in-database copy of the node before updates are made.

Paramet

con

Security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Node(context)

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
som
prop
erty
val-
ues

are invalid.

propert

dbapi =

propert

propert

propert

destroy
Dele
the
Nod
from
the
DB.

Parame
con
Se-
cu-
rity
con-
text.
NOT
This
shou
only
be
used
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Node(context)

propert

propert

propert

propert

propert

fields

classme

Find a node base on its id or uuid and re- turn a Node object.

Parame

- **con** Se- cu- rity con- text
- **nod** the id *or* uuid of

a node

Returns a *Noc* object.

classme Find a node based on its integer ID and return

a Node object.

Paramet

- **cls** the *Noc*
- **con** Security context
- **nod** the ID of a node

Returns a *Noc* object.

classme

Find a node based on the instance UUI and return a

Node object.

Paramet

- **cls** the *Noc*
- **con** Security context
- **uui** the UUI of the instanc

Returns a *Noc* object.

classme

Find a node based on nam

and return a Nod object.

Paramet

- **cls** the *Noc*
- **con** Security context
- **nam** the logical nam of a node

Returns
a *Noc* object.

classme
Get a node by associated port address

Paramet

- **cls** the *Noc*
- **con** Security context.
- **add** A list of port addres

Raises
Nod Not-Four if the node is not foun

Returns
a *Noc* object.

classme
Find a node base on UUI and return a Nod ob-

ject.

Paramet

- **cls** the *Noc*
- **con** Security context
- **uui** the UUI of a node

Returns
a *Noc* object.

propert

propert

propert

propert

propert

propert

propert

propert

classme

Return a list of Nod objects

Parame

- **cls** the *Noc*
- **con** Security context.
- **lim** maximum number of resources to return in

a single result.

- **mar** pagination mark for large data sets.
-

sor columns to sort results by.

- **sor** direction to sort. asc or desc
- **fil** Filters to apply.
- **fie** Requested fields to be returned. Please note some fields are

mandatory for the data model and are automatically included. These are: id, version, updated_at, created_at, owner, and lessee.

Returns
a list of *Noc* object.

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

refresh

Re-
fresh
the
ob-
ject
by
re-
fetch
from
the
DB.

Paramet

con

Se-
cu-
rity
con-
text.
NOT
This
shou
only
be
used
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Node(context)

classme

Re-
lease
the
reser
va-
tion
on
a
node

Paramet

- **con**
  Se-
  cu-
  rity
  con-
  text.
- **tag**
  A
  strin
  uniq
  iden
  ti-
  fy-
  ing
  the
  reser
  va-
  tion
  hold

- **nod** A node id or uuid

Raises
Nod Not- Foun if the node is not foun

propert

propert

classme
Get and re- serv a node

To pre- vent othe Man agen vice from ma- nip- u- lat- ing the given Node while a Task is performed, mark it reserved by this host.

Paramet

- **cls** the

Noc

- **con** Security context.
- **tag** A strin uniq iden tifying the reser vation hold
- **nod** A node ID or UUI

Raises Nod NotFour if the node is not foun

Returns a *Noc* object.

propert

propert

propert

save(*co*

Save updates to this Nod

Colu wise updates will be mad base on the result of self.

If target_power_state is provided, it will be checked against the in-database copy of the node before updates are made.

Paramet

con

Security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Node(context)

Raises

InvaliParameterValu

if
som
prop
erty
val-
ues

are invalid.

propert

propert

propert

propert

touch_p

Touc
the
data
reco
to
mark
the
pro-
vi-
sion
ing
as
alive

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

No-
ti-
fi-

ca-
tion
emit
ted
whe
iron
cre-
ates,
up-
date
or

deletes a node.

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Base
irc
obj
noc
Noc

Pay-
load
sche
for
whe
iron
cre-
ates,
up-
date
or
dele

a
node

SCHEMA

VERSION

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

fields

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

No-
ti-
fi-
ca-
tion
emit
ted
whe
node
con-
sole
state
chan

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

No-
ti-
fi-
ca-
tion
for
whe
a
node
pow
state
is
cor-
recte

in the database.

This
no-
ti-
fi-
ca-
tion
is
emit
ted
whe
iron

detects
that

the actual power state on a bare metal hardware is different from the power state on an ironic node (DB). This notification is emitted after the database is updated to reflect this correction.

VERSIO

propert

propert

fields

propert

propert

propert

propert

class i

Base
irc
obj
noc
Noc

Notification payload
sche
for
whe
a
node
pow
state

is corrected.

from
in-
di-

cate
the
pre-
vi-
ous
pow
state
on
the
ironi
node

before the node was updated.

VERSION

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

fields

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

Notification
emit
ted
whe
mair
tenanc
state
char

via

API.

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Base

irc

obj

not

Not

Base class used for all notification payload about a

Node object.

SCHEMA

VERSION

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

fields

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

class i

Base
iro
obj
not
Not

Notification
emit
ted
whe
iron
char
a
node
pow
state

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Base

irc

obj

noc

Noc

Pay-
load
sche
for
whe
ironi
char
a
node
pow
state

VERSION

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

fields

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

No-
ti-
fi-
ca-
tion

emitted whe iron char a node provision state.

VERSIO

propert

propert

fields

propert

propert

propert

propert

class i

Base *irc obj noc Noc*

Payload sche for whe iron char a node provision

state

SCHEMA

VERSION

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

fields

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

propert

ironic.objects.notification module

class i

Base *iro obj bas Iro*

De-fines the even to be sent on

the wire

An Even Type mus spec ify the ob- ject be- ing actec on, a

string describing the action being taken on the notification, and the status of the action.

VERSION

propert

propert

fields

propert

propert

to_even

Con struc strin for even to be sent on the wire

The strin is in the

format: bare

Raises

ValueEr ror if self. is not one of `fie` `Not`

Returns

ever strin

`propert`

`class` `i`

Base *`iro`* *`obj`* *`bas`* *`Iro`*

Base class for version notifications

Subclass mus define the payload field whic mus

be
a
sub-

class of NotificationPayloadBase.

VERSION

propert

emit (*co*

Senc
the
notification.

Raises

NotificationPayloadError

Raises

oslo

propert

fields

propert

propert

propert

class i

Base
iro
obj
bas
Iro

Base class for the payload of versioned notifications

SCHEMA

VERSION

propert

fields

populat

Populate the object based on the SCH and the sour

objects

Paramet

kwa

A dict contains the sour object

and
the
keys
de-

fined in the SCHEMA

Raises

No-
ti-
fi-
ca-
tion-
Sche
jectE
ror

Raises

No-
ti-
fi-
ca-
tion-
Sche
Er-
ror

propert

class i

Base
irc
obj
bas
Irc

VERSION

propert

fields

propert

propert

propert

ironic.

Re-
mov
se-
crets
from
pay-
load
ob-
ject.

ironic.objects.port module

class i
Base
irc
obj
bas
Irc
osl
bas
Ver

VERSION

propert

create

Cre-
ate
a
Port
reco
in
the
DB.

Paramet
con
Se-
cu-
rity
con-
text.
NOT
This
shou
only
be
used

in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Port(context)

Raises
MA
read
ists
if
ad-
dres
col-
umn
is
not
uniq

Raises
Por-
tAl-
read
ists
if
uuid
col-
umn
is
not
uniq

propert

dbapi =

destroy
Dele
the
Port
from
the
DB.

Paramet
con
Se-
cu-
rity
con-
text.
NOT
This
shou

only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Port(context)

Raises

Port Not- Four

propert

fields

classme

Find a port

Find a port base on its id or uuid or nam or MA ad-

dress and return a Port object.

Parame

- **con** Se- cu- rity con- text
- **por** the id *or*

uuid *or* nam *or* MA address of a port.

Returns
a *Por* object.

Raises
InvalidIdentity

classme
Find a port base on address and return a *Por* object.

Paramet

- **cls** the *Por*
- **con** Security

context

- **add** the address of a port
- **own** DEPRECATED a node owner to match against
- **pro** a node owner or lessee to match against

Returns
a *Por* object.

Raises
Port Not-Foun

classme
Find a port based on its in-

teger ID and return

a Port object.

Paramet

- **cls** the *Por*
- **con** Security context
- **por** the ID of a port

Returns a *Por* object.

Raises Port NotFour

classme

Find a port base on nam and return a *Por*

object.

Parame

- **cls** the *Por*
- **con** Security context
- **nam** the nam of a port

Returns
a *Por* object.

Raises
Port Not-Foun

classme
Find a port base on UUI and return a *Por* object.

Parame

- **cls** the *Por*
- **con** Security context
- **uui** the UUI of a port

Returns
a *Por* object.

Raises
Port Not-Foun

propert

propert

propert

classme

Return a list of Port objects

Paramet

-

con Security context.

- **lim** max imun num ber of resour to return in a single result.
- **mar** pagination marl for large data sets.
- **sor** column to sort results by.
- **sor** direction to sort. asc

or desc

- **own** DEP RE- CAT a node own to matc agai
- **pro** a node own or lesse to matc agai

Returns
a list of *Por* ob- ject.

Raises
In- valid Pa- ram- e- ter- Valu

classme

Re- turn a list of Port ob- jects

associated with

a given node ID.

Parame

- **con** Security context.
- **nod** the ID of the node
- **lim** maximum number of resour to return in

a single result.

- **mar** pagination mark for large data sets.

- **sor** column to sort results by.
- **sor** direction to sort. asc or desc
- **own** DEPRECATED a node owner to matc agai
- **pro** a node owner or lesse to matc agai

Returns a list of *Por* object.

classme

Return a list of Port objects associated with

a given portgroup ID.

Paramet

- **con** Security context.
- **por** the ID of the portgrou
- **lim** maximum number of resour to return in

a single result.

- **mar** pagination mar for larg data sets.
- **sor** column to sort results by.
- **sor** direction to sort. asc or desc
- **own** DEPRECAT a node own to matc agai
- **pro** a node own or less

to mat agai

Returns
a list of *Por* object.

propert

propert

propert

propert

propert

propert

refresh

Loa updat for this Port

Loa a port with the sam uuid from the data and chec for updated attributes. Updates are applied from the loaded port column by column, if there are any updates.

Parame
`con`

Security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Port(context)

Raises

Port Not-Fou

save (*co*

Save updates to this Port

Updates will be mad column by column based on the result of self.what_changed().

Paramet

`con`

Security context. NOT This

shou
only
be
used
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Port(context)

Raises
Port
Not-
Foun

Raises
MA
read
ists
if
ad-
dres
col-
umn
is
not
uniq

classme
Re-
turn
whe
is_s
field
is
sup-
port

Returns
Whe
is_s
field
is
sup-
port

Raises
ovo_

classme
Re-
turn
whe
the
phys
i-

cal_
field
is
sup-
port

Returns
Whe
the
phys
i-
cal_
field
is
sup-
port

Raises
ovo_

propert

propert

class i
Base
iro
obj
not
Not

No-
ti-
fi-
ca-
tion
emit
ted
whe
iron
cre-
ates,
up-
date
or

deletes a port.

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

SCHEMA

VERSIO

propert

propert

propert

fields

propert

propert

propert

propert

propert

propert

propert

propert

propert

ironic.objects.portgroup module

class i

Base
iro
obj
bas
Iro
`osl`
`bas`
`Ver`

VERSION

propert

create

Create a Port grou reco in the DB.

Paramet

con
Security context. NOT This shou only be

used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Portgroup(context)

Raises

Duplicate-Nam MA read ists, Port grou read ists

propert

dbapi =

destroy

Dele the Port grou from the DB.

Parame

con

Security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Portgroup(context)

Raises

Port grou Notl

Port
grou
Not-
Four

propert

fields

classme
Find
a
port-
grou
base
on
its
id,
uuid
nam
or
ad-
dres

Parame

- **por**
The
id,
uuid
nam
or
ad-
dres
of
a
port-
grou
- **con**
Se-
cu-
rity
con-
text

Returns
A
Por

object.

Raises

InvalidIdentity

classme

Find portgrou by address and return a *Por* object.

Paramet

- **cls** the *Por*
- **con** Security context
- **add** The MA address of a portgrou
- **pro**

a node owner or lesse to matc agai

Returns
A *Por* object.

Raises
Portgrou Not-Fou

classme
Find a portgrou by its integer ID and return

a Portgroup object.

Paramet

- **cls** the *Por*
- **con** Security context
- **por**

The ID of a port- grou

Returns
A *Por* ob- ject.

Raises
Port grou Not- Foun

classme
Find port- grou base on nam and re- turn a *Por* ob- ject.

Paramet

- **cls** the *Por*
- **con** Se- cu- rity con- text
- **nam** The nam of

a port-grou

Returns
A *Por* object.

Raises
Portgrou Not-Foun

classme
Find a port-grou by UUI and return a *Por* object.

Parame

- **cls** the *Por*
- **con** Security context
- **uui** The UUI of a port-grou

Returns
A *Por*
object.

Raises
Port
grou
Not-
Foun

propert

propert

classme

Return a list of Port
grou
objects

Paramet

- **cls**
  the *Por*
- **con**
  Security context.
- **lim**
  Maxi-
  mun
  number of re-

sour to return in

a single result.

- **mar** Pagination mark for large data sets.
- **sor** Column to sort results by.
- **sor** Direction to sort. asc or desc
- **pro** a node own or lesse to matc agai

Returns
A list

of *Por* object.

Raises
InvalidParameterValu

classme

Return a list of Port grou objects associated with a given node ID.

Paramet

- **cls** the *Por*
- **con** Security context.
- **nod** The ID of the node

- **lim** Maximum num ber of resour to return in

a single result.

- **mar** Pagination mark for large data sets.
- **sor** Column to sort results by.
- **sor** Direction to sort. asc or desc
- **pro** a node

own or lesse to matc agai

Returns
A list of *Por* ob- ject.

Raises
In- valic Pa- ram- e- ter- Valu

propert

propert

propert

propert

refresh
Loac up- date for this Port grou

Loac a port- grou with the same uuid from the data and chec for updated attributes. Updates are applied from the loaded portgroup column by column, if there are any updates.

Paramet

con Security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Portgroup(context)

Raises

Port grou NotFou

save (*co*

Save update to this Port grou

Update will be mad column by column base on the result of self.what_changed().

Paramet

con

Security context. NOT This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Portgroup(context)

Raises

Port grou Not-Foun Duplicate-Nam MAC read ists

propert

propert

propert

class i

Base *irc obj not Not*

Notification whe

ironi creates, update or dele a portgroup.

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Bas

irc

obj

not

Not

SCHEMA

VERSION

propert

propert

propert

fields

propert

propert

propert

propert

propert

propert

propert

ironic.objects.trait module

class i

Base

irc

obj

bas

Irc

VERSION

create

Cre-

ate

a

Trai

reco

in

the

DB.

Paramet

con

se-

cu-

rity

con-

text.

NOT

This
shou
only
be
usec
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Trait(context).

Raises

In-
valic
Pa-
ram-
e-
ter-
Valu
if
addi
the
trait
wou
ex-

ceed the per-node traits limit.

Raises

Nod
Not-
Four
if
the
node
no
long
ap-
pear
in
the
data

propert

dbapi =

classme

Dele
the
Trai
from
the
DB.

Parame

- **con** se-curity con-text. NO This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Trait(context).

- **nod** The id of a node
- **tra** A trait strin

Raises
Nod Not-Foun if the node no long ap-pear in the data

Raises
Nod Trai Not-Foun

if the trait is not foun

classme

Chec whe a Trai ex- ists in the DB.

Paramet

- **con** se- cu- rity con- text. NOT This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Trait(context).

- **nod** The id of a node
- **tra** A trait strin

Returns

True

if the trait ex- ists oth- er- wise Fals

Raises

Nod Not- Foun if the node no long ap- pear in the data

fields

propert

propert

propert

class i

Base *irc obj bas Irc irc obj bas Irc*

VERSION

classme

Re- plac

all existing trait
with the spec
ified list.

Parame

- **con**
  security context.
  NO
  This
  shou
  only
  be
  use
  internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Trait(context).
- **nod**
  The id of a node
- **tra**
  List of Strin
  trait
  to set.

Raises
  In-
  vali
  Pa-
  ram-

e-
ter-
Valu
if
addi
the
trait
wou
ex-

ceed the per-node traits limit.

Raises

Nod
Not-
Foun
if
the
node
no
long
ap-
pear
in
the
data

propert

dbapi =

classme

Dele
all
trait
for
the
spec
i-
fied
node

Parame

- **con**
  se-
  cu-
  rity
  con-
  text.
  NOT

This shou only be use internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Trait(context).

- `nod` The id of a node

Raises
Nod Not-Foun if the node no long appear in the data

`fields`

`classme`
Return all trait for the specified node

Paramet

- `con` secu-

rity
con-
text.
NOT
This
shou
only
be
used
in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: Trait(context).

- **nod**
The
id
of
a
node

Raises
Nod
Not-
Foun
if
the
node
no
long
ap-
pear
in
the
data

get_tra
Re-
turn
a
list
of
nam
of
the
trait
in
this
list.

propert

propert

ironic.objects.volume_connector module

class i

Base
iro
obj
bas
Iro
`osl`
`bas`
`Ver`

VERSION

propert

create

Create a VolumeConnector record in the DB.

Parame

con

security context. NOT This should only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: VolumeConnector(context).

Raises
Vol-
ume
Con
nec-
torT
pe-
An-
dI-
dAl-
read
ists
if
a

volume connector already exists with the same type and connector_id

Raises
Vol-
ume
Con
nec-
torA
read
ists
if
a
vol-
ume
con-
nec-

tor with the same UUID already exists

propert

dbapi =

destroy
Dele
the
Vol-
ume
Con
nec-
tor
from
the
DB.

Parame
con
se-
cu-

rity context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: VolumeConnector(context).

Raises

Vol- ume Con nec- torN Four if the volume connector cannot be found

propert

fields

classme

Find a volume connector base on its ID or UUI

Paramet

-

con security context

- **ide** the data primary key ID *or* the UUI of a volume connector

Returns
a *Vol* object

Raises
InvalidIdentity if iden is neither an integer ID nor a UUID

Raises
VolumeConnectorN Foun if

no volume connector exists with the specified ident

classme

Find a volume connector base on its integer ID.

Paramet

- **cls** the *Vol*
- **con** Security context.
- **db_** The integer (dat primary mar key) ID of a vol-

ume connector.

Returns
A *Vol* object.

Raises
Volume Con nectorN Fou if no volume connector

exists with the specified ID.

classme
Find a volume connector base on its UU

Parame

- **cls** the *Vol*
- **con** security context
- **uui**

the UUI of a volume connector

Returns
a *Vol* object

Raises
VolumeConnectorNFoun if no volume connector

exists with the specified UUID

propert

classme

Return a list of VolumeConnector objects

Paramet

-

`con` security context

- **`lim`** max i- mun num ber of re- sour to return in

a single result

- **`mar`** pagi- nation mar for larg data sets
- **`sor`** column to sort results by
- **`sor`** direction to sort. asc

or desc

- **pro** The associated node proj to sear with

Returns a list of *Vol* objects

Raises InvalidParameterValu if sort_ does not exist

classme

Return a list of VolumeConnector objects

re-

lated to a given node ID.

Parame

- **con** security context
- **nod** the integer ID of the node
- **lim** maxi- mum number of resour to return in

a single result

- **mar** pagination mark for large data sets
-

sor column to sort results by

- **sor** direction to sort. asc or desc
- **pro** The associated node proj to searc with

Returns
a list of *Vol* objects

Returns
a list of *Vol* objects

Raises
Invalid Param-

e-
ter-
Valu
if
sort_
does
not
ex-
ist

propert

refresh

Load
up-
date
for
this
Vol-
ume
Con
nec-
tor.

Load
a
vol-
ume
con-
nec-
tor
with
the
same
UUI
from
the
data

and check for updated attributes. If there are any updates, they are applied from the loaded volume connector, column by column.

Parame

con

se-
cu-
rity
con-
text.
NOT
This
shou
only

be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: VolumeConnector(context).

save (*co*

Save updates to this VolumeConnector.

Updates will be made column by column based on the re-

sult of self.do_version_changes_for_db().

Parame

con

security context. NOT This should only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: VolumeConnector(context).

Raises

Vol-

ume Con nec- torN Foun if the vol- ume con- nec- tor

cannot be found

Raises
Vol- ume Con nec- torT pe- An- dI- dAl- read ists if an-

other connector already exists with the same values for type and connector_id fields

Raises
In- valid Pa- ram- e- ter- Valu whe the UUI is be- ing

changed

propert

propert

propert

class i

Base
irc
obj
not
Not

Notification emitted at CRU of a volume connector.

VERSIO

propert

propert

fields

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

Pay-

load
sche
for
CRU
of
a
vol-
ume
con-
nec-
tor.

SCHEMA

VERSION

propert

propert

propert

fields

propert

propert

propert

propert

ironic.objects.volume_target module

class i

Base
irc
obj
bas
Irc
osl
bas
Ver

`VERSIO`

`propert`

`create`

Create a Volume get reco in the DB.

Parame

`con` security context. NO This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: VolumeTarget(context).

Raises

Volume get- Boo dex- Al- read ists if a volume target already exists with the same node ID and boot index

Raises

Volume

getAlreadyExists if a volume target with

the same UUID exists

propert

dbapi =

destroy

Delete the VolumeTarget from the DB.

Parame

con

security context. NOT This should only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: VolumeTarget(context).

Raises

VolumeTargetNotFound if the

volume target cannot be found

propert

fields

classme

Find a volume target base on its ID or UUI

Parame

- **con** security context
- **ide** the data primary key ID *or* the UUI of a volume target

Returns
a *Vol* object

Raises
InvalidIdentity if iden is neither an in-

teger ID nor a UUID

Raises
VolumegetNotFour if no volume target with this

ident exists

classme
Find a volume target base on its data ID.

Paramet

- **cls** the *Vol*
- **con** security context
- **db_** the data primary key (integer) ID of a vol-

ume target

Returns a *Vol* object

Raises VolumeTarget-NotFour if no volume target with this

ID exists

classme

Find a

volume target base on its UUI

Paramet

- **cls** the *Vol*
- **con** security context
- **uui** the UUI of a volume target

Returns
a *Vol* object

Raises
Volume getNotFoun if no volume target

with this

UUID exists

propert

classme

Return a list of Volume get objects

Paramet

- con security context
- lim maximum number of resources to return in

a single result

- mar pagination mark

for large data sets

- **sor** column to sort results by
- **sor** direction to sort. asc or desc
- **pro** The associated node proj to sear with

Returns a list of `Vol` objects

Returns a list of *`Vol`* objects

Raises
Invalid Parameter Valu if sort_ does not exist

classme

Return a list of Volume get objects related to

a given node ID.

Parame

- **con** security context
- **nod** the integer ID of the node

a single result

- **`lim`**
  max
  i-
  mun
  num
  ber
  of
  re-
  sour
  to
  re-
  turn
  in
- **`mar`**
  pag-
  i-
  na-
  tion
  marl
  for
  large
  data
  sets
- **`sor`**
  col-
  umn
  to
  sort
  re-
  sults
  by
- **`sor`**
  di-
  rec-
  tion
  to
  sort.
  asc
  or
  desc
- **`pro`**
  The
  as-

so-
ci-
ated
node
proj
to
sear
with

Returns
a
list
of
`Vol`
ob-
jects

Returns
a
list
of
`Vol`
ob-
jects

Raises
In-
valic
Pa-
ram-
e-
ter-
Valu
if
sort_
does
not
ex-
ist

`classme`

Re-
turn
a
list
of
Vol-
ume
get
ob-
jects

related to

a given volume ID.

Paramet

- **con** security context
- **vol** the UUI of the volume
- **lim** maximum number of volume targets to re-

turn in a single result

- **mar** pagination mark for large data sets
-

`sor` column to sort results by

- **`sor`** direction to sort. asc or desc

Returns
a list of *`Vol`* objects

Raises
Invali Parameter Valu if sort_ does not exist

`propert`

`propert`

`refresh`
Loa update for

this Volume get.

Loa a volume target with the same UUI from the data and

check for updated attributes. If there are any updates, they are applied from the loaded volume target, column by column.

Parame

con security context. NOT This shou only be used in-

ternally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: VolumeTarget(context).

Raises

Volume get-Not-Foun if the volume target cannot be found

`save` (*co

Save update to this Volume get.

Update will be mad column by column base on the result of self.do_version_changes_for_db().

Paramet

con security context. NOT This shou only be used internally by the indirection_api. Unfortunately, RPC requires context as the first argument, even though we dont use it. A context should be set when instantiating the object, e.g.: VolumeTarget(context).

Raises

Invali Parameter Valu

if
the
UUI
is
being

changed

Raises
Vol-
ume
get-
Boo
dex-
Al-
read
ists
if
a
vol-
ume
tar-

get already exists with the same node ID and boot index values

Raises
Vol-
ume
get-
Not-
Fou
if
the
vol-
ume
tar-
get
can-
not

be found

propert

propert

propert

propert

class i

Base

irc
obj
not
Not

Notification emitted at CRU of a volume target.

VERSION

propert

propert

fields

propert

propert

propert

propert

class i

Base
irc
obj
not
Not

SCHEMA

VERSIO

propert

propert

propert

fields

propert

propert

propert

propert

propert

propert

Module contents

ironic.

Submodules

ironic.version module

Module contents

9.1. the Iror ics CI

Its im-por-tant

to understand the role of each job in the CI, how to add new jobs and how to debug failures that may arise. To facilitate that, we have created the documentation below.

Jobs description

The description of each jobs that runs in the CI whe you submit a patch for *openstack/ironic* is visible in *Table. OpenStack Ironic CI jobs description*.

Job name	Description
ironic-tox-unit-with-driver-libs-python3	Runs Ironic unit tests with the driver dependencies installed under Python3
ironic-standalone	Deploys Ironic in standalone mode and runs tempest tests that match the regex *ironic_standalone*.
ironic-tempest-functional-python3	Deploys Ironic in standalone mode and runs tempest functional tests that matches the regex *ironic_tempest_plugin.tests.api* under Python3
ironic-grenade	Deploys Ironic in a DevStack and runs upgrade for all enabled services.
ironic-grenade-dsvm-multinode-multitenant	Deploys Ironic in a multinode DevStack and runs upgrade for all enabled services.
ironic-tempest-ipa-partition-pxe_ipmitool	Deploys Ironic in DevStack under Python3, configured to use dib ramdisk partition image with *pxe* boot and *ipmi* driver. Runs tempest tests that match the regex *ironic_tempest_plugin.tests.scenario* and deploy 1 virtual baremetal.
ironic-tempest-partition-bios-redfish-pxe	Deploys Ironic in DevStack, configured to use dib ramdisk partition image with *pxe* boot and *redfish* driver. Runs tempest tests that match the regex *ironic_tempest_plugin.tests.scenario*, also deploys 1 virtual baremetal.
ironic-tempest-ipa-partition-uefi-pxe_ipmitool	Deploys Ironic in DevStack, configured to use dib ramdisk partition image with *uefi* boot and *ipmi* driver. Runs tempest tests that match the regex *ironic_tempest_plugin.tests.scenario*, also deploys 1 virtual baremetal.
ironic-tempest-ipa-wholedisk-direct-tinyipa-multinode	Deploys Ironic in a multinode DevStack, configured to use a pre-build tinyipa ramdisk wholedisk image that is downloaded from a Swift temporary url, *pxe* boot and *ipmi* driver. Runs tempest tests that match the regex *(ironic_tempest_plugin.tests.scenario|test_schedule_to_all_nod* and deploys 7 virtual baremetal.
ironic-tempest-ipa-wholedisk-bios-agent_ipmitool-tinyipa	Deploys Ironic in DevStack, configured to use a pre-build tinyipa ramdisk wholedisk image that is downloaded from a Swift temporary url, *pxe* boot and *ipmi* driver. Runs tempest tests that match the regex *ironic_tempest_plugin.tests.scenario* and deploys 1 virtual baremetal.
ironic-tempest-ipa-wholedisk-bios-agent_ipmitool-indirect	Deploys Ironic in DevStack, configured to use a pre-built dib ramdisk wholedisk image that is downloaded from http url, *pxe* boot and *ipmi* driver. Runs tempest tests that match the regex *ironic_tempest_plugin.tests.scenario* and deploys 1 virtual baremetal.
ironic-tempest-ipa-partition-bios-agent_ipmitool-indirect	Deploys Ironic in DevStack, configured to use a pre-built dib ramdisk partition image that is downloaded from http url, *pxe* boot and *ipmi* driver. Runs tempest tests that match the regex

Adding a new Job

Are you familiar with Zuul?

Before start trying to figure out how Zuul work take som time and read about Zuul Config and the Zuul Best Practices.

Where can I find the existing jobs?

The jobs for the Iron proj are defined under the zuul fold in the root directory, that contains three files, whose function is described below.

- iron jobs Contains the configuration of each

Iron

Job converted to Zuul v3.

- lega iron jobs Contains the configuration of each

Ironic Job that havent been converted to Zuul v3 yet.

- proj Contains the jobs that will run during chec and gate

phase.

Create a new Job

Identify amo the existing jobs the one that mos clos resembles the scenario you want to test, the existing job will be used as *parent* in your job definition. Now you will only need to either overwrite or add variables to your job definition under the *vars* section to represent the desired scenario.

The code block below shows the minimal structure of a new job definition that you need to add to ironic-jobs.yaml.

```
-
 ↪␣
 ↪j
␣
 ↪␣
 ↪␣
 ↪␣
 ↪n
 ↪
 ↪<
 ↪o
 ↪t
 ↪n
 ↪j
 ↪
␣
 ↪␣
 ↪␣
 ↪␣
 ↪d
 ↪
 ↪<
 ↪y
 ↪j
 ↪d
 ↪
␣
 ↪␣
 ↪␣
 ↪␣
 ↪p
 ↪
 ↪<
 ↪t
 ↪a
 ↪e
 ↪
```

(continues on next page)

(continued from previous page)

```
␣
 ↪␣
 ↪␣
 ↪␣
 ↪v
␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪
 ↪<
 ↪:
 ↪
 ↪<
 ↪v
 ↪
```

After having the definition of your new job you just need to add the job name to the project.yaml under *check* and *gate*. Only jobs that are voting should be in the *gate* section.

```
-
 ↪␣
 ↪p
␣
 ↪␣
 ↪␣
 ↪␣
 ↪c
␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪j
␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪␣
 ↪-
 ↪
```

(continues on next page)

(continued from previous page)

```
↪new job>
```

```
␣
↪␣
↪␣
↪␣
↪g
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪q
↪i
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪j
␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪␣
↪-
↪␣
↪
↪<
↪o
↪t
```

Debugging CI failures

If you see *FAIL URE* in one or mor jobs for your patc plea

dont panic. This guide may help you to find the initial reason for the failure. When clicking in the failed job you will be redirect to the Zuul web page that contains all the information about the job build.

Zuul Web Page

The pag has thre tabs *Sum* *mar* *Log* and *Con* *sole*

- Sum mar Con tain over all information abou the buil

of the job, if the job build failed it will contain a general output of the failure.

- Log Con tain all configuration and log files abou

all services that were used in the job. This will give you an overall idea of the failures and you can identify services that may be involved. The *job-output* file can give an overall idea of the failures and what services may be involved.

- Con

sole Con tains all the play book that were execute

by clicking in the arrow before each playbook name you can find the roles and commands that were executed.

9.1. pol icy for stable bra

Stable bran that are on Extend Mai tenanc and have re-

ceived backports in a while, can be tagged as `Unmaintained`, after discussions within the ironic community. If such a decision is taken, an email will be sent to the OpenStack mailing list.

Wha does `Unm` mea The bran still exists, but the community will not actively backport patches from maintained branches. Fixes can still be merged, though, if pushed into review by operators or other downstream developers. It also means that branchless projects (e.g.: ironic-tempest-plugin), may not have configurations that are compatible with those branches.

ironi upstrea

As of 09 Mar 2020 the list of `Unm` bran includ

- Oca (Las com mit - Jun 28, 201
- Pike (Las com mit - Oct 2, 201

CHAPTER

TEN

RELEASE NOTES

Release Notes

PYTHON MODULE INDEX

a

ironic.api, 2397
ironic.api.app, 2382
ironic.api.config, 2383
ironic.api.controllers, 2378
ironic.api.controllers.base, 2374
ironic.api.controllers.link, 2376
ironic.api.controllers.root, 2376
ironic.api.controllers.v1, 2373
ironic.api.controllers.v1.allocation, 2148
ironic.api.controllers.v1.bios, 2155
ironic.api.controllers.v1.chassis, 2156
ironic.api.controllers.v1.collection, 2162
ironic.api.controllers.v1.conductor, 2165
ironic.api.controllers.v1.deploy_template, 2168
ironic.api.controllers.v1.driver, 2175
ironic.api.controllers.v1.event, 2183
ironic.api.controllers.v1.node, 2184
ironic.api.controllers.v1.notification_utils, 2237
ironic.api.controllers.v1.port, 2240
ironic.api.controllers.v1.portgroup, 2251
ironic.api.controllers.v1.ramdisk, 2258
ironic.api.controllers.v1.utils, 2264
ironic.api.controllers.v1.versions, 2346
ironic.api.controllers.v1.volume, 2347
ironic.api.controllers.v1.volume_connector, 2347
ironic.api.controllers.v1.volume_target, 2360
ironic.api.controllers.version, 2376
ironic.api.functions, 2383
ironic.api.hooks, 2389
ironic.api.method, 2396
ironic.api.middleware, 2380
ironic.api.middleware.auth_public_routes, 2378
ironic.api.middleware.json_ext, 2379
ironic.api.middleware.parsable_error, 2380
ironic.api.wsgi, 2397

c

ironic.cmd, 2400
ironic.cmd.api, 2397
ironic.cmd.conductor, 2397
ironic.cmd.dbsync, 2398
ironic.cmd.status, 2399
ironic.common, 2697
ironic.common.args, 2408
ironic.common.boot_devices, 2424
ironic.common.boot_modes, 2426
ironic.common.checksum_utils, 2427
ironic.common.cinder, 2433
ironic.common.components, 2446
ironic.common.config, 2447
ironic.common.context, 2447
ironic.common.dhcp_factory, 2449
ironic.common.driver_factory, 2452
ironic.common.exception, 2467
ironic.common.faults, 2492
ironic.common.fsm, 2493
ironic.common.glance_service, 2408
ironic.common.glance_service.image_service, 2400

ironic.common.glance_service.service_utils, 2406
ironic.common.hash_ring, 2498
ironic.common.i18n, 2498
ironic.common.image_format_inspector, 2499
ironic.common.image_service, 2524
ironic.common.images, 2537
ironic.common.indicator_states, 2556
ironic.common.keystone, 2557
ironic.common.molds, 2564
ironic.common.network, 2567
ironic.common.neutron, 2573
ironic.common.nova, 2598
ironic.common.policy, 2599
ironic.common.profiler, 2603
ironic.common.pxe_utils, 2605
ironic.common.qemu_img, 2632
ironic.common.raid, 2632
ironic.common.release_mappings, 2639
ironic.common.rpc, 2641
ironic.common.rpc_service, 2645
ironic.common.service, 2646
ironic.common.states, 2646
ironic.common.swift, 2662
ironic.common.utils, 2670
ironic.common.wsgi_service, 2696
ironic.conductor, 2918
ironic.conductor.allocations, 2697
ironic.conductor.base_manager, 2702
ironic.conductor.cleaning, 2707
ironic.conductor.deployments, 2711
ironic.conductor.manager, 2717
ironic.conductor.notification_utils, 2734
ironic.conductor.rpcapi, 2739
ironic.conductor.steps, 2857
ironic.conductor.task_manager, 2862
ironic.conductor.utils, 2878
ironic.conf, 2926
ironic.conf.agent, 2918
ironic.conf.anaconda, 2918
ironic.conf.ansible, 2919
ironic.conf.api, 2919
ironic.conf.audit, 2919
ironic.conf.auth, 2919
ironic.conf.cinder, 2920
ironic.conf.conductor, 2920
ironic.conf.console, 2920
ironic.conf.database, 2921
ironic.conf.default, 2921
ironic.conf.deploy, 2921
ironic.conf.dhcp, 2921
ironic.conf.disk_utils, 2921
ironic.conf.drac, 2921
ironic.conf.glance, 2921
ironic.conf.healthcheck, 2922
ironic.conf.ibmc, 2922
ironic.conf.ilo, 2922
ironic.conf.inspector, 2922
ironic.conf.ipmi, 2922
ironic.conf.irmc, 2922
ironic.conf.iscsi, 2922
ironic.conf.metrics, 2923
ironic.conf.metrics_statsd, 2923
ironic.conf.molds, 2923
ironic.conf.neutron, 2923
ironic.conf.nova, 2923
ironic.conf.opts, 2923
ironic.conf.pxe, 2925
ironic.conf.redfish, 2925
ironic.conf.service_catalog, 2925
ironic.conf.snmp, 2925
ironic.conf.swift, 2925
ironic.conf.xclarity, 2926

d

ironic.db, 3193
ironic.db.api, 3072
ironic.db.migration, 3193
ironic.db.sqlalchemy, 3072
ironic.db.sqlalchemy.api, 2926
ironic.db.sqlalchemy.migration, 3053
ironic.db.sqlalchemy.models, 3056
ironic.dhcp, 3214
ironic.dhcp.base, 3194
ironic.dhcp.neutron, 3201
ironic.dhcp.none, 3207
ironic.drivers, 4341
ironic.drivers.base, 4163
ironic.drivers.drac, 4301
ironic.drivers.fake_hardware, 4303
ironic.drivers.generic, 4307
ironic.drivers.hardware_type, 4310
ironic.drivers.ibmc, 4315
ironic.drivers.ilo, 4316
ironic.drivers.intel_ipmi, 4320
ironic.drivers.ipmi, 4321
ironic.drivers.irmc, 4322

ironic.drivers.modules, 4163
ironic.drivers.modules.agent, 3750
ironic.drivers.modules.agent_base, 3774
ironic.drivers.modules.agent_client, 3810
ironic.drivers.modules.agent_power, 3844
ironic.drivers.modules.ansible, 3223
ironic.drivers.modules.ansible.deploy, 3214
ironic.drivers.modules.boot_mode_utils, 3851
ironic.drivers.modules.console_utils, 3858
ironic.drivers.modules.deploy_utils, 3865
ironic.drivers.modules.drac, 3327
ironic.drivers.modules.drac.bios, 3223
ironic.drivers.modules.drac.boot, 3243
ironic.drivers.modules.drac.common, 3247
ironic.drivers.modules.drac.inspect, 3249
ironic.drivers.modules.drac.job, 3255
ironic.drivers.modules.drac.management, 3258
ironic.drivers.modules.drac.power, 3278
ironic.drivers.modules.drac.raid, 3286
ironic.drivers.modules.drac.utils, 3314
ironic.drivers.modules.drac.vendor_passthru, 3316
ironic.drivers.modules.fake, 3908
ironic.drivers.modules.ibmc, 3367
ironic.drivers.modules.ibmc.management, 3327
ironic.drivers.modules.ibmc.mappings, 3343
ironic.drivers.modules.ibmc.power, 3343
ironic.drivers.modules.ibmc.raid, 3352
ironic.drivers.modules.ibmc.utils, 3359
ironic.drivers.modules.ibmc.vendor, 3361
ironic.drivers.modules.ilo, 3500
ironic.drivers.modules.ilo.bios, 3367
ironic.drivers.modules.ilo.boot, 3372
ironic.drivers.modules.ilo.common, 3404
ironic.drivers.modules.ilo.console, 3441
ironic.drivers.modules.ilo.firmware_processor 3443
ironic.drivers.modules.ilo.inspect, 3448
ironic.drivers.modules.ilo.management, 3451
ironic.drivers.modules.ilo.power, 3482
ironic.drivers.modules.ilo.raid, 3489
ironic.drivers.modules.ilo.vendor, 3495
ironic.drivers.modules.image_cache, 3989
ironic.drivers.modules.image_utils, 3994
ironic.drivers.modules.inspect_utils, 4010
ironic.drivers.modules.inspector, 4012
ironic.drivers.modules.intel_ipmi, 3501
ironic.drivers.modules.intel_ipmi.management, 3500
ironic.drivers.modules.ipmitool, 4015
ironic.drivers.modules.ipxe, 4055
ironic.drivers.modules.irmc, 3578
ironic.drivers.modules.irmc.bios, 3501
ironic.drivers.modules.irmc.boot, 3506
ironic.drivers.modules.irmc.common, 3521
ironic.drivers.modules.irmc.inspect, 3528
ironic.drivers.modules.irmc.management, 3533
ironic.drivers.modules.irmc.packaging_versio 3549

ironic.drivers.modules.irmc.power, 3566
ironic.drivers.modules.irmc.raid, 3574
ironic.drivers.modules.iscsi_deploy, 4056
ironic.drivers.modules.network, 3622
ironic.drivers.modules.network.common, 3578
ironic.drivers.modules.network.flat, 3594
ironic.drivers.modules.network.neutron, 3603
ironic.drivers.modules.network.noop, 3613
ironic.drivers.modules.noop, 4084
ironic.drivers.modules.noop_mgmt, 4105
ironic.drivers.modules.pxe, 4118
ironic.drivers.modules.pxe_base, 4131
ironic.drivers.modules.redfish, 3721
ironic.drivers.modules.redfish.bios, 3622
ironic.drivers.modules.redfish.boot, 3630
ironic.drivers.modules.redfish.inspect, 3643
ironic.drivers.modules.redfish.management, 3647
ironic.drivers.modules.redfish.power, 3683
ironic.drivers.modules.redfish.raid, 3691
ironic.drivers.modules.redfish.utils, 3711
ironic.drivers.modules.redfish.vendor, 3718
ironic.drivers.modules.snmp, 4140
ironic.drivers.modules.storage, 3736
ironic.drivers.modules.storage.cinder, 3721
ironic.drivers.modules.storage.external, 3726
ironic.drivers.modules.storage.noop, 3731
ironic.drivers.modules.xclarity, 3750
ironic.drivers.modules.xclarity.common, 3736
ironic.drivers.modules.xclarity.management, 3741
ironic.drivers.modules.xclarity.power, 3746
ironic.drivers.redfish, 4325
ironic.drivers.snmp, 4326
ironic.drivers.utils, 4327
ironic.drivers.xclarity, 4340

i

ironic, 4552

o

ironic.objects, 4552
ironic.objects.allocation, 4341
ironic.objects.base, 4353
ironic.objects.bios, 4365
ironic.objects.chassis, 4378
ironic.objects.conductor, 4387
ironic.objects.deploy_template, 4395
ironic.objects.deployment, 4408
ironic.objects.fields, 4416
ironic.objects.indirection, 4424
ironic.objects.node, 4434
ironic.objects.notification, 4470
ironic.objects.port, 4476
ironic.objects.portgroup, 4495
ironic.objects.trait, 4510
ironic.objects.volume_connector, 4519
ironic.objects.volume_target, 4534

v

ironic.version, 4552

INDEX

Symbols

--autogenerate
  revision command line option, 1921
--config-dir <DIR>
  ironic-dbsync command line option, 1919
--config-file <PATH>
  ironic-dbsync command line option, 1919
--debug
  ironic-dbsync command line option, 1919
--help
  create_schema command line option, 1920
  ironic-dbsync command line option, 1919
  online_data_migrations command line option, 1920
  revision command line option, 1921
  stamp command line option, 1921
  upgrade command line option, 1922
  version command line option, 1922
--max-count <NUMBER>
  online_data_migrations command line option, 1920
--message <MESSAGE>
  revision command line option, 1921
--option <MIGRATION.KEY=VALUE>
  online_data_migrations command line option, 1920
--revision <REVISION>
  stamp command line option, 1921
  upgrade command line option, 1922
--version
  ironic-dbsync command line option, 1919
-d
  ironic-dbsync command line option, 1919
-h
  create_schema command line option, 1920
  ironic-dbsync command line option, 1919
  online_data_migrations command line option, 1920
  revision command line option, 1921
  stamp command line option, 1921
  upgrade command line option, 1922
  version command line option, 1922
-m <MESSAGE>
  revision command line option, 1921

A

abandon_bios_config() (*ironic.drivers.modules.drac.vendor_passthru.DracWSMa* *method*), 3318
abandon_config() (*in module ironic.drivers.modules.drac.bios*), 3230
abandon_config() (*in module ironic.drivers.modules.drac.raid*), 3298
abort() (*ironic.drivers.base.InspectInterface method*), 4207
abort() (*ironic.drivers.modules.inspector.Inspector method*), 4012
abort_on_conductor_take_over() (*in module ironic.conductor.utils*), 2878
AbstractHardwareType (*class in ironic.drivers.hardware_type*), 4311
acquire() (*in module ironic.conductor.task_manager*), 2877

`acquire_port()` (*in module ironic.drivers.modules.console_utils*), 3858

`action()` (*ironic.objects.notification.EventType property*), 4471

`activate_license()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3454

`ACTIVE` (*in module ironic.common.states*), 2647

`actual_size()` (*ironic.common.image_format_inspector.FileInspector property*), 2502

`add_allocation_filter_by_conductor()` (*in module ironic.db.sqlalchemy.api*), 3049

`add_allocation_filter_by_node()` (*in module ironic.db.sqlalchemy.api*), 3049

`add_auth_opts()` (*in module ironic.conf.auth*), 2919

`add_certificates()` (*in module ironic.drivers.modules.ilo.common*), 3406

`add_cleaning_network()` (*ironic.drivers.base.NetworkInterface method*), 4244

`add_cleaning_network()` (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3595

`add_cleaning_network()` (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3604

`add_cleaning_network()` (*ironic.drivers.modules.network.noop.NoopNetwork method*), 3613

`add_command_parsers()` (*in module ironic.cmd.dbsync*), 2399

`add_fieldname()` (*ironic.common.exception.UnknownAttribute method*), 2490

`add_identity_filter()` (*in module ironic.db.sqlalchemy.api*), 3049

`add_inspection_network()` (*ironic.drivers.base.NetworkInterface method*), 4244

`add_inspection_network()` (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3596

`add_inspection_network()` (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3605

`add_node_capability()` (*in module ironic.drivers.utils*), 4329

`add_node_filter_by_chassis()` (*in module ironic.db.sqlalchemy.api*), 3050

`add_node_tag()` (*ironic.db.api.Connection method*), 3072

`add_node_tag()` (*ironic.db.sqlalchemy.api.Connection method*), 2926

`add_node_trait()` (*ironic.db.api.Connection method*), 3073

`add_node_trait()` (*ironic.db.sqlalchemy.api.Connection method*), 2927

`add_node_traits()` (*ironic.conductor.manager.ConductorManager method*), 2719

`add_node_traits()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2751

`add_port_filter()` (*in module ironic.db.sqlalchemy.api*), 3050

`add_port_filter_by_node()` (*in module ironic.db.sqlalchemy.api*), 3051

`add_port_filter_by_node_owner()` (*in module ironic.db.sqlalchemy.api*), 3051

`add_port_filter_by_node_project()` (*in module ironic.db.sqlalchemy.api*), 3051

`add_port_filter_by_portgroup()` (*in module ironic.db.sqlalchemy.api*), 3051

`add_portgroup_filter()` (*in module ironic.db.sqlalchemy.api*), 3051

`add_portgroup_filter_by_node()` (*in module ironic.db.sqlalchemy.api*), 3052

`add_portgroup_filter_by_node_project()` (*in module ironic.db.sqlalchemy.api*), 3052

`add_ports_to_network()` (*in module ironic.common.neutron*), 2574

`add_provisioning_network()` (*ironic.drivers.base.NetworkInterface method*), 4245

`add_provisioning_network()` (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3597

`add_provisioning_network()` (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3606

`add_provisioning_network()` (*ironic.drivers.modules.network.noop.NoopNetwork method*), 3614 add_rescuing_network() (*ironic.drivers.base.NetworkInterface method*), 4246

add_rescuing_network() (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3597

add_rescuing_network() (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3606

add_secret_token() (*in module ironic.conductor.utils*), 2878

add_state() (*ironic.common.fsm.FSM method*), 2493

add_transition() (*ironic.common.fsm.FSM method*), 2495

add_volume_conn_filter_by_node_project() (*in module ironic.db.sqlalchemy.api*), 3052

add_volume_target_filter_by_node_project() (*in module ironic.db.sqlalchemy.api*), 3052

address (*ironic.db.sqlalchemy.models.Port attribute*), 3067

address (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3068

address() (*ironic.objects.port.Port property*), 4476

address() (*ironic.objects.port.PortCRUDPayload property*), 4494

address() (*ironic.objects.portgroup.Portgroup property*), 4495

address() (*ironic.objects.portgroup.PortgroupCRUDPayload property*), 4509

ADOPTFAIL (*in module ironic.common.states*), 2647

ADOPTING (*in module ironic.common.states*), 2648

advanced_net_fields (*ironic.api.controllers.v1.port.PortsController attribute*), 2240

after() (*ironic.api.hooks.ContextHook method*), 2390

after() (*ironic.api.hooks.NoExceptionTracebackHook method*), 2393

agent_add_clean_params() (*in module ironic.drivers.modules.deploy_utils*), 3865

agent_is_alive() (*in module ironic.conductor.utils*), 2879

AgentAPIError, 2467

AgentBaseMixin (*class in ironic.drivers.modules.agent_base*), 3774

AgentClient (*class in ironic.drivers.modules.agent_client*), 3810

AgentCommandTimeout, 2467

AgentConnectionFailed, 2467

AgentDeploy (*class in ironic.drivers.modules.agent*), 3750

AgentDeployMixin (*class in ironic.drivers.modules.agent*), 3757

AgentDeployMixin (*class in ironic.drivers.modules.agent_base*), 3780

AgentInProgress, 2467

AgentOobStepsMixin (*class in ironic.drivers.modules.agent_base*), 3792

AgentPower (*class in ironic.drivers.modules.agent_power*), 3844

AgentRAID (*class in ironic.drivers.modules.agent*), 3758

AgentRescue (*class in ironic.drivers.modules.agent*), 3765

ALL (*ironic.objects.fields.NotificationLevel attribute*), 4420

ALL (*ironic.objects.fields.NotificationStatus attribute*), 4420

ALL_INTERFACES (*in module ironic.drivers.base*), 4164

all_interfaces() (*in module ironic.common.driver_factory*), 2453

all_interfaces() (*ironic.drivers.base.BareDriver property*), 4170

all_versions() (*in module ironic.api.controllers.version*), 2376

Allocation (*class in ironic.db.sqlalchemy.models*), 3056

Allocation (*class in ironic.objects.allocation*), 4341

allocation_id (*ironic.db.sqlalchemy.models.Node attribute*), 3063

allocation_id() (*ironic.objects.node.Node property*), 4434

allocation_sanitize() (*in module ironic.api.controllers.v1.allocation*), 2154

AllocationAlreadyExists, 2467

`AllocationCRUDNotification` (*class in ironic.objects.allocation*), 4351

`AllocationCRUDPayload` (*class in ironic.objects.allocation*), 4352

`AllocationDuplicateName`, 2467

`AllocationFailed`, 2468

`AllocationNotFound`, 2468

`AllocationsController` (*class in ironic.api.controllers.v1.allocation*), 2148

`allow_agent_token()` (*in module ironic.api.controllers.v1.utils*), 2265

`allow_agent_version_in_heartbeat()` (*in module ironic.api.controllers.v1.utils*), 2265

`allow_allocation_backfill()` (*in module ironic.api.controllers.v1.utils*), 2265

`allow_allocation_owner()` (*in module ironic.api.controllers.v1.utils*), 2266

`allow_allocation_update()` (*in module ironic.api.controllers.v1.utils*), 2266

`allow_allocations()` (*in module ironic.api.controllers.v1.utils*), 2267

`allow_bios_interface()` (*in module ironic.api.controllers.v1.utils*), 2268

`allow_build_configdrive()` (*in module ironic.api.controllers.v1.utils*), 2268

`allow_configdrive_vendor_data()` (*in module ironic.api.controllers.v1.utils*), 2269

`allow_deploy_steps()` (*in module ironic.api.controllers.v1.utils*), 2269

`allow_deploy_templates()` (*in module ironic.api.controllers.v1.utils*), 2269

`allow_detail_query()` (*in module ironic.api.controllers.v1.utils*), 2270

`allow_dynamic_drivers()` (*in module ironic.api.controllers.v1.utils*), 2271

`allow_dynamic_interfaces()` (*in module ironic.api.controllers.v1.utils*), 2271

`allow_expose_conductors()` (*in module ironic.api.controllers.v1.utils*), 2272

`allow_expose_events()` (*in module ironic.api.controllers.v1.utils*), 2272

`allow_field()` (*in module ironic.api.controllers.v1.utils*), 2273

`allow_inject_nmi()` (*in module ironic.api.controllers.v1.utils*), 2273

`allow_inspect_abort()` (*in module ironic.api.controllers.v1.utils*), 2274

`allow_inspect_wait_state()` (*in module ironic.api.controllers.v1.utils*), 2274

`allow_links_node_states_and_driver_properties` (*in module ironic.api.controllers.v1.utils*), 2275

`allow_local_link_connection_network_type()` (*in module ironic.api.controllers.v1.utils*), 2275

`allow_node_logical_names()` (*in module ironic.api.controllers.v1.utils*), 2275

`allow_node_rebuild_with_configdrive()` (*in module ironic.api.controllers.v1.utils*), 2275

`allow_port_advanced_net_fields()` (*in module ironic.api.controllers.v1.utils*), 2276

`allow_port_internal_info()` (*in module ironic.api.controllers.v1.utils*), 2276

`allow_port_is_smartnic()` (*in module ironic.api.controllers.v1.utils*), 2277

`allow_port_physical_network()` (*in module ironic.api.controllers.v1.utils*), 2278

`allow_portgroup_mode_properties()` (*in module ironic.api.controllers.v1.utils*), 2278

`allow_portgroups()` (*in module ironic.api.controllers.v1.utils*), 2279

`allow_portgroups_subcontrollers()` (*in module ironic.api.controllers.v1.utils*), 2279

`allow_raid_config()` (*in module ironic.api.controllers.v1.utils*), 2280

`allow_ramdisk_endpoints()` (*in module ironic.api.controllers.v1.utils*), 2280

`allow_remove_chassis_uuid()` (*in module ironic.api.controllers.v1.utils*), 2281

`allow_rescue_interface()` (*in module ironic.api.controllers.v1.utils*), 2281

`allow_reset_interfaces()` (*in module ironic.api.controllers.v1.utils*), 2282

`allow_soft_power_off()` (*in module ironic.api.controllers.v1.utils*), 2282

`allow_status_in_heartbeat()` (*in module ironic.api.controllers.v1.utils*), 2283

`allow_storage_interface()` (*in module ironic.api.controllers.v1.utils*), 2283

`allow_traits()` (*in module ironic.api.controllers.v1.utils*), 2284

`allow_verify_ca_in_heartbeat()` (*in module ironic.api.controllers.v1.utils*), 2284 allow_vifs_subcontroller() (*in module ironic.api.controllers.v1.utils*), 2284

allow_volume() (*in module ironic.api.controllers.v1.utils*), 2285

and_valid() (*in module ironic.common.args*), 2408

AnsibleDeploy (*class in ironic.drivers.modules.ansible.deploy*), 3214

apply_configuration() (*ironic.drivers.base.BIOSInterface method*), 4164

apply_configuration() (*ironic.drivers.base.RAIDInterface method*), 4269

apply_configuration() (*ironic.drivers.modules.agent.AgentRAID method*), 3759

apply_configuration() (*ironic.drivers.modules.drac.bios.DracWSManBIOS method*), 3224

apply_configuration() (*ironic.drivers.modules.drac.raid.DracWSManRAID method*), 3291

apply_configuration() (*ironic.drivers.modules.fake.FakeBIOS method*), 3909

apply_configuration() (*ironic.drivers.modules.ibmc.raid.IbmcRAID method*), 3352

apply_configuration() (*ironic.drivers.modules.ilo.bios.IloBIOS method*), 3367

apply_configuration() (*ironic.drivers.modules.ilo.raid.Ilo5RAID method*), 3489

apply_configuration() (*ironic.drivers.modules.irmc.bios.IRMCBIOS method*), 3501

apply_configuration() (*ironic.drivers.modules.noop.NoBIOS method*), 4086

apply_configuration() (*ironic.drivers.modules.redfish.bios.RedfishBIOS method*), 3622

apply_configuration() (*ironic.drivers.modules.redfish.raid.RedfishRAID method*), 3691

apply_jsonpatch() (*in module ironic.api.controllers.v1.utils*), 2285

args (*ironic.db.sqlalchemy.models.DeployTemplateStep attribute*), 3061

arguments (*ironic.api.functions.FunctionDefinition attribute*), 2384

as_dict() (*ironic.db.sqlalchemy.models.IronicBase method*), 3062

as_dict() (*ironic.objects.base.IronicObject method*), 4354

as_dict() (*ironic.objects.base.IronicObjectListBase method*), 4362

as_dict() (*ironic.objects.node.Node method*), 4434

attach_boot_iso_if_needed() (*in module ironic.drivers.modules.irmc.boot*), 3518

attach_vmedia() (*in module ironic.drivers.modules.ilo.common*), 3408

attach_volumes() (*in module ironic.common.cinder*), 2433

attach_volumes() (*ironic.drivers.base.StorageInterface method*), 4282

attach_volumes() (*ironic.drivers.modules.fake.FakeStorage method*), 3979

attach_volumes() (*ironic.drivers.modules.storage.cinder.CinderStorage method*), 3721

attach_volumes() (*ironic.drivers.modules.storage.external.ExternalStorage method*), 3727

attach_volumes() (*ironic.drivers.modules.storage.noop.NoopStorage method*), 3731

AUTH_CLASSES (*ironic.drivers.modules.redfish.utils.SessionCach attribute*), 3711

authorize() (*in module ironic.common.policy*), 2599

AuthPublicRoutes (*class in ironic.api.middleware*), 2380

AuthPublicRoutes (*class in ironic.api.middleware.auth_public_routes*), 2378

AUTO_TYPE (*ironic.objects.fields.FlexibleDictField attribute*), 4417

AUTO_TYPE (*ironic.objects.fields.ListOfFlexibleDictsField attribute*), 4418

AUTO_TYPE (*ironic.objects.fields.MACAddressField attribute*), 4420

AUTO_TYPE (*ironic.objects.fields.NotificationLevelField attribute*), 4420

AUTO_TYPE (*ironic.objects.fields.NotificationStatusField attribute*), 4421
AUTO_TYPE (*ironic.objects.fields.StringFieldThatAcceptsCallable attribute*), 4423
automated_clean (*ironic.db.sqlalchemy.models.Node attribute*), 3063
automated_clean() (*ironic.objects.node.Node property*), 4434
AVAILABLE (*in module ironic.common.states*), 2648

B

backfill_allocation() (*in module ironic.conductor.allocations*), 2697
backup_bios_config() (*in module ironic.drivers.modules.irmc.management*), 3549
BadRequest, 2468
BareDriver (*class in ironic.drivers.base*), 4169
base_version() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3553
BaseConductorManager (*class in ironic.conductor.base_manager*), 2702
BaseDHCP (*class in ironic.dhcp.base*), 3194
BaseDriverFactory (*class in ironic.common.driver_factory*), 2452
BaseImageService (*class in ironic.common.image_service*), 2524
BaseInterface (*class in ironic.drivers.base*), 4176
before() (*ironic.api.hooks.ConfigHook method*), 2389
before() (*ironic.api.hooks.ContextHook method*), 2391
before() (*ironic.api.hooks.DBHook method*), 2392
before() (*ironic.api.hooks.PublicUrlHook method*), 2394
before() (*ironic.api.hooks.RPCHook method*), 2395
BF_OFFSET (*ironic.common.image_format_inspector.QcowInspector attribute*), 2513
BF_OFFSET_LEN (*ironic.common.image_format_inspector.QcowInspector attribute*), 2513
BIOS (*in module ironic.common.boot_devices*), 2425
bios (*ironic.drivers.base.BareDriver attribute*), 4170
bios_interface (*ironic.db.sqlalchemy.models.Node attribute*), 3063
bios_interface() (*ironic.objects.node.Node property*), 4434
bios_interface() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457
bios_interface() (*ironic.objects.node.NodeCRUDPayload property*), 4452
bios_interface() (*ironic.objects.node.NodePayload property*), 4460
bios_interface() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464
bios_interface() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
BIOSInterface (*class in ironic.drivers.base*), 4164
BIOSSetting (*class in ironic.db.sqlalchemy.models*), 3057
BIOSSetting (*class in ironic.objects.bios*), 4365
BIOSSettingAlreadyExists, 2468
BIOSSettingList (*class in ironic.objects.bios*), 4370
BIOSSettingListNotFound, 2468
BIOSSettingNotFound, 2468
BLINKING (*in module ironic.common.indicator_states*), 2556
bmc_reset() (*ironic.drivers.modules.ipmitool.VendorPassthru method*), 4043
body() (*in module ironic.api.method*), 2396
body_type (*ironic.api.functions.FunctionDefinition attribute*), 2384
boolean() (*in module ironic.common.args*), 2410
BooleanField (*class in ironic.objects.fields*), 4416
boot (*ironic.drivers.base.BareDriver attribute*), 4170
boot_device (*ironic.api.controllers.v1.node.NodeManagementC attribute*), 2196
boot_index (*ironic.db.sqlalchemy.models.VolumeTarget attribute*), 3070
boot_index() (*ironic.objects.volume_target.VolumeTarget property*), 4535

boot_index() (ironic.objects.volume_target.VolumeTargetCRUDPayload property), 4552
boot_instance() (ironic.drivers.modules.agent_base.AgentOobStepsMixin method), 3793
boot_interface (ironic.db.sqlalchemy.models.Node attribute), 3063
boot_interface() (ironic.objects.node.Node property), 4434
boot_interface() (ironic.objects.node.NodeCorrectedPowerStatePayload property), 4457
boot_interface() (ironic.objects.node.NodeCRUDPayload property), 4452
boot_interface() (ironic.objects.node.NodePayload property), 4460
boot_interface() (ironic.objects.node.NodeSetPowerStatePayload property), 4464
boot_interface() (ironic.objects.node.NodeSetProvisionStatePayload property), 4468
boot_into_iso() (ironic.drivers.modules.ilo.vendor.VendorPassthru method), 3496
boot_up_seq() (ironic.drivers.modules.ibmc.vendor.IBMCVendor method), 3361
BootDeviceController (class in ironic.api.controllers.v1.node), 2184
BootInterface (class in ironic.drivers.base), 4186
build_agent_options() (in module ironic.drivers.modules.deploy_utils), 3866
build_configdrive() (in module ironic.conductor.utils), 2880
build_deploy_pxe_options() (in module ironic.common.pxe_utils), 2605
build_driver_for_task() (in module ironic.common.driver_factory), 2454
build_extra_pxe_options() (in module ironic.common.pxe_utils), 2605
build_instance_info_for_deploy() (in module ironic.drivers.modules.deploy_utils), 3866
build_instance_pxe_options() (in module ironic.common.pxe_utils), 2605
build_kickstart_config_options() (in module ironic.common.pxe_utils), 2605
build_pxe_config_options() (in module ironic.common.pxe_utils), 2607
build_service_pxe_config() (in module ironic.common.pxe_utils), 2609
build_url() (in module ironic.api.controllers.link), 2376

C

cache_bios_settings() (in module ironic.drivers.base), 4290
cache_bios_settings() (ironic.drivers.base.BIOSInterface method), 4166
cache_bios_settings() (ironic.drivers.modules.drac.bios.DracWSManBIOS method), 3226
cache_bios_settings() (ironic.drivers.modules.fake.FakeBIOS method), 3912
cache_bios_settings() (ironic.drivers.modules.ilo.bios.IloBIOS method), 3368
cache_bios_settings() (ironic.drivers.modules.irmc.bios.IRMCBIOS method), 3503
cache_bios_settings() (ironic.drivers.modules.noop.NoBIOS method), 4088
cache_bios_settings() (ironic.drivers.modules.redfish.bios.RedfishBIOS method), 3623
cache_instance_image() (in module ironic.drivers.modules.deploy_utils), 3868
cache_ramdisk_kernel() (in module ironic.common.pxe_utils), 2609
call() (ironic.common.glance_service.image_service.GlanceIma method), 2400
can_send_create_port() (ironic.conductor.rpcapi.ConductorAPI method), 2753
can_send_rescue() (ironic.conductor.rpcapi.ConductorAPI method), 2753
candidate_nodes (ironic.db.sqlalchemy.models.Allocation attribute), 3056

candidate_nodes() (*ironic.objects.allocation.Allocation property*), 4341
candidate_nodes() (*ironic.objects.allocation.AllocationCRUDPayload property*), 4352
capabilities (*ironic.drivers.base.BootInterface attribute*), 4186
capabilities (*ironic.drivers.modules.fake.FakeBoot attribute*), 3917
capabilities (*ironic.drivers.modules.ilo.boot.IloUefiHttpsBoot attribute*), 3377
capabilities (*ironic.drivers.modules.ilo.boot.IloVirtualMediaBoot attribute*), 3387
capabilities (*ironic.drivers.modules.ipxe.iPXEBoot attribute*), 4056
capabilities (*ironic.drivers.modules.irmc.boot.IRMCVirtualMediaBoot attribute*), 3509
capabilities (*ironic.drivers.modules.pxe.PXEBoot attribute*), 4126
capabilities (*ironic.drivers.modules.redfish.boot.RedfishVirtualMediaBoot attribute*), 3633
capabilities_to_dict() (*in module ironic.drivers.utils*), 4330
capture() (*ironic.common.image_format_inspector.CaptureRegion method*), 2500
CaptureRegion (*class in ironic.common.image_format_inspector*), 2499
CatalogNotFound, 2468
CDROM (*in module ironic.common.boot_devices*), 2425
change_node_power_state() (*ironic.conductor.manager.ConductorManager method*), 2719
change_node_power_state() (*ironic.conductor.rpcapi.ConductorAPI method*), 2753
change_physical_disk_state() (*in module ironic.drivers.modules.drac.raid*), 3299
Chassis (*class in ironic.db.sqlalchemy.models*), 3058
Chassis (*class in ironic.objects.chassis*), 4378
CHASSIS (*in module ironic.common.components*), 2446
chassis_id (*ironic.db.sqlalchemy.models.Node attribute*), 3063
chassis_id() (*ironic.objects.node.Node property*), 4434
chassis_uuid() (*ironic.objects.node.NodeCRUDPayload property*), 4452
ChassisAlreadyExists, 2468
ChassisController (*class in ironic.api.controllers.v1.chassis*), 2156
ChassisCRUDNotification (*class in ironic.objects.chassis*), 4385
ChassisCRUDPayload (*class in ironic.objects.chassis*), 4386
ChassisNotEmpty, 2469
ChassisNotFound, 2469
check() (*in module ironic.common.policy*), 2600
check_allocation_policy_and_retrieve() (*in module ironic.api.controllers.v1.utils*), 2287
check_allow_clean_disable_ramdisk() (*in module ironic.api.controllers.v1.utils*), 2288
check_allow_configdrive() (*in module ironic.api.controllers.v1.utils*), 2289
check_allow_deploy_steps() (*in module ironic.api.controllers.v1.utils*), 2289
check_allow_driver_detail() (*in module ironic.api.controllers.v1.utils*), 2289
check_allow_filter_by_conductor() (*in module ironic.api.controllers.v1.utils*), 2289
check_allow_filter_by_conductor_group() (*in module ironic.api.controllers.v1.utils*), 2290
check_allow_filter_by_fault() (*in module ironic.api.controllers.v1.utils*), 2290
check_allow_filter_by_lessee() (*in module ironic.api.controllers.v1.utils*), 2291
check_allow_filter_by_owner() (*in module ironic.api.controllers.v1.utils*), 2291
check_allow_filter_driver_type() (*in module ironic.api.controllers.v1.utils*), 2292
check_allow_management_verbs() (*in module ironic.api.controllers.v1.utils*), 2292
check_allow_specify_driver() (*in module ironic.api.controllers.v1.utils*), 2293
check_allow_specify_fields() (*in module ironic.api.controllers.v1.utils*),

2293
check_allow_specify_resource_class() (*in module ironic.api.controllers.v1.utils*), 2294
check_allowed_fields() (*in module ironic.api.controllers.v1.utils*), 2294
check_allowed_portgroup_fields() (*in module ironic.api.controllers.v1.utils*), 2295
check_and_update_node_interfaces() (*in module ironic.common.driver_factory*), 2456
check_cipher_suite_errors() (*in module ironic.drivers.modules.ipmitool*), 4049
check_dir() (*in module ironic.common.utils*), 2670
check_file_system_for_iscsi_device() (*in module ironic.drivers.modules.iscsi_deploy*), 4064
check_for_invalid_fields() (*in module ironic.api.controllers.v1.utils*), 2295
check_for_invalid_state_and_allow_filter() (*in module ironic.api.controllers.v1.utils*), 2297
check_for_missing_params() (*in module ironic.drivers.modules.deploy_utils*), 3870
check_if_image_format_is_permitted() (*in module ironic.common.images*), 2537
check_image_service() (*in module ironic.common.glance_service.image_service*), 2406
check_image_size() (*in module ironic.drivers.modules.agent*), 3771
check_image_size() (*in module ironic.drivers.modules.iscsi_deploy*), 4065
check_interface_capability() (*in module ironic.drivers.modules.deploy_utils*), 3872
check_list_policy() (*in module ironic.api.controllers.v1.utils*), 2297
check_multiple_node_policies_and_retrieve() (*in module ironic.api.controllers.v1.utils*), 2298
check_node_list() (*ironic.db.api.Connection method*), 3075
check_node_list() (*ironic.db.sqlalchemy.api.Connection method*), 2929
check_node_policy_and_retrieve() (*in module ironic.api.controllers.v1.utils*), 2300
check_obj_versions() (*ironic.cmd.dbsync.DBCommand method*), 2398
check_owner_policy() (*in module ironic.api.controllers.v1.utils*), 2302
check_policy() (*in module ironic.api.controllers.v1.utils*), 2303
check_policy() (*in module ironic.common.policy*), 2600
check_policy_true() (*in module ironic.api.controllers.v1.utils*), 2304
check_port_list_policy() (*in module ironic.api.controllers.v1.utils*), 2305
check_port_policy_and_retrieve() (*in module ironic.api.controllers.v1.utils*), 2306
check_share_fs_mounted() (*in module ironic.drivers.modules.irmc.boot*), 3519
check_versions() (*ironic.db.api.Connection method*), 3076
check_versions() (*ironic.db.sqlalchemy.api.Connection method*), 2930
check_volume_list_policy() (*in module ironic.api.controllers.v1.utils*), 2308
check_volume_policy_and_retrieve() (*in module ironic.api.controllers.v1.utils*), 2309
Checks (*class in ironic.cmd.status*), 2399
choose_cipher_suite() (*in module ironic.drivers.modules.ipmitool*), 4050
chunked_reader() (*in module ironic.common.image_format_inspector*), 2522
CinderStorage (*class in ironic.drivers.modules.storage.cinder*), 3721
clean_dhcp() (*ironic.common.dhcp_factory.DHCPFactory method*), 2449
clean_dhcp_opts() (*ironic.dhcp.base.BaseDHCP method*), 3194
CLEAN_FAILURE (*in module ironic.common.faults*), 2492
clean_step (*ironic.db.sqlalchemy.models.Node

attribute), 3063

`clean_step()` (*in module ironic.drivers.base*), 4290

`clean_step()` (*ironic.objects.node.Node property*), 4434

`clean_step()` (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457

`clean_step()` (*ironic.objects.node.NodeCRUDPayload property*), 4452

`clean_step()` (*ironic.objects.node.NodePayload property*), 4461

`clean_step()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464

`clean_step()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468

`clean_up()` (*ironic.drivers.base.DeployInterface method*), 4195

`clean_up()` (*ironic.drivers.base.RescueInterface method*), 4279

`clean_up()` (*ironic.drivers.modules.agent.AgentDeploy method*), 3751

`clean_up()` (*ironic.drivers.modules.agent.AgentRescue method*), 3766

`clean_up()` (*ironic.drivers.modules.agent_base.AgentBaseMixin method*), 3774

`clean_up()` (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3215

`clean_up()` (*ironic.drivers.modules.fake.FakeDeploy method*), 3929

`clean_up()` (*ironic.drivers.modules.image_cache.ImageCache method*), 3989

`clean_up()` (*ironic.drivers.modules.iscsi_deploy.ISCSIDeploy method*), 4056

`clean_up()` (*ironic.drivers.modules.pxe.PXEAnacondaDeploy method*), 4119

`clean_up_caches()` (*in module ironic.drivers.modules.image_cache*), 3992

`clean_up_instance()` (*ironic.drivers.base.BootInterface method*), 4186

`clean_up_instance()` (*ironic.drivers.modules.fake.FakeBoot method*), 3917

`clean_up_instance()` (*ironic.drivers.modules.ilo.boot.IloiPXEBoot method*), 3397

`clean_up_instance()` (*ironic.drivers.modules.ilo.boot.IloPXEBoot method*), 3373

`clean_up_instance()` (*ironic.drivers.modules.ilo.boot.IloUefiHttpsBoot method*), 3377

`clean_up_instance()` (*ironic.drivers.modules.ilo.boot.IloVirtualMediaBoot method*), 3387

`clean_up_instance()` (*ironic.drivers.modules.irmc.boot.IRMCVirtualMediaBoo method*), 3509

`clean_up_instance()` (*ironic.drivers.modules.pxe_base.PXEBaseMixin method*), 4131

`clean_up_instance()` (*ironic.drivers.modules.redfish.boot.RedfishVirtualMediaB method*), 3633

`clean_up_pxe_config()` (*in module ironic.common.pxe_utils*), 2610

`clean_up_pxe_env()` (*in module ironic.common.pxe_utils*), 2610

`clean_up_ramdisk()` (*ironic.drivers.base.BootInterface method*), 4187

`clean_up_ramdisk()` (*ironic.drivers.modules.fake.FakeBoot method*), 3917

`clean_up_ramdisk()` (*ironic.drivers.modules.ilo.boot.IloUefiHttpsBoot method*), 3378

`clean_up_ramdisk()` (*ironic.drivers.modules.ilo.boot.IloVirtualMediaBoot method*), 3388

`clean_up_ramdisk()` (*ironic.drivers.modules.irmc.boot.IRMCVirtualMediaBoo method*), 3510

`clean_up_ramdisk()` (*ironic.drivers.modules.pxe_base.PXEBaseMixin method*), 4132

`clean_up_ramdisk()` (*ironic.drivers.modules.redfish.boot.RedfishVirtualMediaB method*), 3634

`CLEANFAIL` (*in module ironic.common.states*), 2648

`CLEANING` (*in module ironic.common.states*), 2649

`cleaning_error_handler()` (*in module ironic.conductor.utils*), 2881

`cleanup()` (*in module ironic.common.rpc*), 2644

`cleanup()` (*in module ironic.drivers.modules.image_cache*), 3994

`cleanup_after_timeout()` (*in module*

ironic.conductor.utils), 2883
`cleanup_cleanwait_timeout()` (*in module ironic.conductor.utils*), 2884
`cleanup_disk_image()` (*in module ironic.drivers.modules.image_utils*), 3997
`cleanup_floppy_image()` (*in module ironic.drivers.modules.image_utils*), 3998
`cleanup_iso_image()` (*in module ironic.drivers.modules.image_utils*), 3998
`cleanup_rescuewait_timeout()` (*in module ironic.conductor.utils*), 2884
`cleanup_vmedia_boot()` (*in module ironic.drivers.modules.ilo.common*), 3410
`CLEANWAIT` (*in module ironic.common.states*), 2649
`clear_certificates()` (*in module ironic.drivers.modules.ilo.common*), 3410
`clear_foreign_config()` (*in module ironic.drivers.modules.drac.raid*), 3301
`clear_iscsi_boot_target()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3455
`clear_job_queue()` (*ironic.drivers.modules.drac.management.DracRedfishManagement method*), 3260
`clear_job_queue()` (*ironic.drivers.modules.drac.management.DracWSManManagement method*), 3267
`clear_node_reservations_for_conductor()` (*ironic.db.sqlalchemy.api.Connection method*), 2931
`clear_node_target_power_state()` (*ironic.db.sqlalchemy.api.Connection method*), 2931
`clear_secure_boot_keys()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3456
`clear_secure_boot_keys()` (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3647
`ClientSideError`, 2469
`close()` (*ironic.common.image_format_inspector.InfoWrapper method*), 2511
`code` (*ironic.common.exception.Conflict attribute*), 2470
`code` (*ironic.common.exception.ImageChecksumFileReadFailure attribute*), 2476
`code` (*ironic.common.exception.Invalid attribute*), 2479
`code` (*ironic.common.exception.NoFreeConductorWorker attribute*), 2483
`code` (*ironic.common.exception.NotAcceptable attribute*), 2485
`code` (*ironic.common.exception.NotAuthorized attribute*), 2486
`code` (*ironic.common.exception.NotFound attribute*), 2486
`code` (*ironic.common.exception.TemporaryFailure attribute*), 2489
`coerce()` (*ironic.objects.fields.FlexibleDict static method*), 4416
`coerce()` (*ironic.objects.fields.MACAddress static method*), 4418
`coerce()` (*ironic.objects.fields.StringAcceptsCallable static method*), 4421
`collect_deploy_logs` (*ironic.drivers.modules.agent_base.HeartbeatMixin attribute*), 3794
`collect_deploy_logs` (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy attribute*), 3215
`collect_ramdisk_logs()` (*in module ironic.drivers.utils*), 4331
`collect_system_logs()` (*ironic.drivers.modules.agent_client.AgentClient method*), 3810
`collection_from_list()` (*in module ironic.api.controllers.v1.bios*), 2155
`commit_bios_config()` (*ironic.drivers.modules.drac.vendor_passthru.DracWSMa method*), 3319
`commit_config()` (*in module ironic.drivers.modules.drac.bios*), 3230
`commit_config()` (*in module ironic.drivers.modules.drac.raid*), 3302
`CommunicationError`, 2469
`complete()` (*ironic.common.image_format_inspector.CaptureRe property*), 2501
`complete()` (*ironic.common.image_format_inspector.FileInspec property*), 2503
`compute_image_checksum()` (*in module ironic.common.checksum_utils*), 2427
`compute_image_checksum()` (*in module ironic.drivers.modules.deploy_utils*), 3873
`Conductor` (*class in ironic.db.sqlalchemy.models*), 3059

Conductor (*class in ironic.objects.conductor*), 4387
conductor_affinity (*ironic.db.sqlalchemy.models.Allocation attribute*), 3056
conductor_affinity (*ironic.db.sqlalchemy.models.Node attribute*), 3063
conductor_affinity() (*ironic.objects.allocation.Allocation property*), 4341
conductor_affinity() (*ironic.objects.node.Node property*), 4434
conductor_group (*ironic.db.sqlalchemy.models.Conductor attribute*), 3059
conductor_group (*ironic.db.sqlalchemy.models.Node attribute*), 3063
conductor_group() (*ironic.objects.conductor.Conductor property*), 4387
conductor_group() (*ironic.objects.node.Node property*), 4435
conductor_group() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457
conductor_group() (*ironic.objects.node.NodeCRUDPayload property*), 4452
conductor_group() (*ironic.objects.node.NodePayload property*), 4461
conductor_group() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464
conductor_group() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
conductor_id (*ironic.db.sqlalchemy.models.ConductorHardwareInterfaces attribute*), 3060
ConductorAlreadyRegistered, 2469
ConductorAPI (*class in ironic.conductor.rpcapi*), 2739
ConductorHardwareInterfaces (*class in ironic.db.sqlalchemy.models*), 3059
ConductorHardwareInterfacesAlreadyRegistered, 2469
ConductorManager (*class in ironic.conductor.manager*), 2718
ConductorNotFound, 2469
ConductorsController (*class in ironic.api.controllers.v1.conductor*), 2165
config() (*in module ironic.api.controllers.v1.ramdisk*), 2264
ConfigHook (*class in ironic.api.hooks*), 2389
ConfigInvalid, 2470
ConfigNotFound, 2470
configure_intel_speedselect() (*ironic.drivers.modules.intel_ipmi.management.IntelIPMI method*), 3500
configure_local_boot() (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3781
configure_secure_boot_if_needed() (*in module ironic.drivers.modules.boot_mode_utils*), 3851
configure_tenant_networks() (*ironic.drivers.base.NetworkInterface method*), 4247
configure_tenant_networks() (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3598
configure_tenant_networks() (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3607
configure_tenant_networks() (*ironic.drivers.modules.network.noop.NoopNetwork method*), 3614
Conflict, 2470
Connection (*class in ironic.db.api*), 3072
Connection (*class in ironic.db.sqlalchemy.api*), 2926
connection (*ironic.common.swift.SwiftAPI attribute*), 2662
connector_id (*ironic.db.sqlalchemy.models.VolumeConnector attribute*), 3069
connector_id() (*ironic.objects.volume_connector.VolumeConnector property*), 4519
connector_id() (*ironic.objects.volume_connector.VolumeConnectorCRUD property*), 4534
console (*ironic.api.controllers.v1.node.NodeStatesController attribute*), 2197
console (*ironic.drivers.base.BareDriver attribute*), 4170

console_enabled (*ironic.db.sqlalchemy.models.Node attribute*), 3063
console_enabled() (*ironic.objects.node.Node property*), 4435
console_enabled() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457
console_enabled() (*ironic.objects.node.NodeCRUDPayload property*), 4452
console_enabled() (*ironic.objects.node.NodePayload property*), 4461
console_enabled() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464
console_enabled() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
console_interface (*ironic.db.sqlalchemy.models.Node attribute*), 3063
console_interface() (*ironic.objects.node.Node property*), 4435
console_interface() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457
console_interface() (*ironic.objects.node.NodeCRUDPayload property*), 4452
console_interface() (*ironic.objects.node.NodePayload property*), 4461
console_interface() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464
console_interface() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
ConsoleError, 2470
ConsoleInterface (*class in ironic.drivers.base*), 4192
ConsoleSubprocessFailed, 2470
context_info() (*ironic.common.image_format_inspector.FileInspector property*), 2503
ContextHook (*class in ironic.api.hooks*), 2390
continue_cleaning() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3794
continue_deploy() (*in module ironic.drivers.modules.iscsi_deploy*), 4066
continue_deploy() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3794
continue_node_clean() (*in module ironic.conductor.cleaning*), 2707
continue_node_clean() (*ironic.conductor.manager.ConductorManager method*), 2719
continue_node_clean() (*ironic.conductor.rpcapi.ConductorAPI method*), 2755
continue_node_deploy() (*in module ironic.conductor.deployments*), 2711
continue_node_deploy() (*ironic.conductor.manager.ConductorManager method*), 2721
continue_node_deploy() (*ironic.conductor.rpcapi.ConductorAPI method*), 2756
Controller (*class in ironic.api.controllers.v1*), 2373
convert() (*in module ironic.api.controllers.v1.volume*), 2347
convert_drive_units() (*in module ironic.drivers.modules.redfish.raid*), 3706
convert_image() (*in module ironic.common.qemu_img*), 2632
convert_steps() (*in module ironic.api.controllers.v1.deploy_template*), 2173
convert_to_version() (*ironic.objects.base.IronicObject method*), 4354
convert_with_links() (*in module ironic.api.controllers.v1.allocation*), 2154
convert_with_links() (*in module ironic.api.controllers.v1.bios*), 2155
convert_with_links() (*in module ironic.api.controllers.v1.chassis*), 2161
convert_with_links() (*in module ironic.api.controllers.v1.conductor*), 2168
convert_with_links() (*in module ironic.api.controllers.v1.deploy_template*),

2173
convert_with_links() (*in module ironic.api.controllers.v1.driver*), 2180
convert_with_links() (*in module ironic.api.controllers.v1.port*), 2249
convert_with_links() (*in module ironic.api.controllers.v1.portgroup*), 2257
convert_with_links() (*in module ironic.api.controllers.v1.ramdisk*), 2264
convert_with_links() (*in module ironic.api.controllers.v1.volume_connector*), 2360
convert_with_links() (*in module ironic.api.controllers.v1.volume_target*), 2373
converted_size() (*in module ironic.common.images*), 2538
copy_image_to_swift() (*in module ironic.drivers.modules.ilo.common*), 3412
copy_image_to_web_server() (*in module ironic.drivers.modules.ilo.common*), 3414
core_interfaces() (*ironic.drivers.base.BareDriver property*), 4171
create() (*ironic.objects.allocation.Allocation method*), 4341
create() (*ironic.objects.bios.BIOSSetting method*), 4365
create() (*ironic.objects.bios.BIOSSettingList class method*), 4371
create() (*ironic.objects.chassis.Chassis method*), 4378
create() (*ironic.objects.deploy_template.DeployTemplate method*), 4395
create() (*ironic.objects.deployment.Deployment method*), 4408
create() (*ironic.objects.node.Node method*), 4435
create() (*ironic.objects.port.Port method*), 4476
create() (*ironic.objects.portgroup.Portgroup method*), 4495
create() (*ironic.objects.trait.Trait method*), 4510
create() (*ironic.objects.trait.TraitList class method*), 4514
create() (*ironic.objects.volume_connector.VolumeConnector method*), 4519
create() (*ironic.objects.volume_target.VolumeTarget method*), 4535
create_allocation() (*ironic.conductor.manager.ConductorManager method*), 2723
create_allocation() (*ironic.conductor.rpcapi.ConductorAPI method*), 2757
create_allocation() (*ironic.db.api.Connection method*), 3077
create_allocation() (*ironic.db.sqlalchemy.api.Connection method*), 2931
create_bios_setting_list() (*ironic.db.api.Connection method*), 3078
create_bios_setting_list() (*ironic.db.sqlalchemy.api.Connection method*), 2932
create_boot_iso() (*in module ironic.common.images*), 2539
create_chassis() (*ironic.db.api.Connection method*), 3081
create_chassis() (*ironic.db.sqlalchemy.api.Connection method*), 2935
create_configuration() (*ironic.drivers.base.RAIDInterface method*), 4271
create_configuration() (*ironic.drivers.modules.agent.AgentRAID method*), 3760
create_configuration() (*ironic.drivers.modules.drac.raid.DracRedfishRAID method*), 3287
create_configuration() (*ironic.drivers.modules.drac.raid.DracWSManRAID method*), 3293
create_configuration() (*ironic.drivers.modules.fake.FakeRAID method*), 3970
create_configuration() (*ironic.drivers.modules.ibmc.raid.IbmcRAID method*), 3355
create_configuration() (*ironic.drivers.modules.ilo.raid.Ilo5RAID method*), 3491
create_configuration() (*ironic.drivers.modules.irmc.raid.IRMCRAID*

method), 3575

`create_configuration()` (*ironic.drivers.modules.noop.NoRAID method*), 4096

`create_configuration()` (*ironic.drivers.modules.redfish.raid.RedfishRAID method*), 3693

`create_deploy_template()` (*ironic.db.api.Connection method*), 3081

`create_deploy_template()` (*ironic.db.sqlalchemy.api.Connection method*), 2935

`create_esp_image_for_uefi()` (*in module ironic.common.images*), 2543

`create_ipxe_boot_script()` (*in module ironic.common.pxe_utils*), 2611

`create_isolinux_image_for_bios()` (*in module ironic.common.images*), 2546

`create_link_without_raise()` (*in module ironic.common.utils*), 2672

`create_node()` (*ironic.conductor.manager.ConductorManager method*), 2723

`create_node()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2758

`create_node()` (*ironic.db.api.Connection method*), 3083

`create_node()` (*ironic.db.sqlalchemy.api.Connection method*), 2937

`create_object()` (*ironic.common.swift.SwiftAPI method*), 2662

`create_port()` (*ironic.conductor.manager.ConductorManager method*), 2723

`create_port()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2760

`create_port()` (*ironic.db.api.Connection method*), 3085

`create_port()` (*ironic.db.sqlalchemy.api.Connection method*), 2939

`create_portgroup()` (*ironic.db.api.Connection method*), 3085

`create_portgroup()` (*ironic.db.sqlalchemy.api.Connection method*), 2939

`create_ports_if_not_exist()` (*in module ironic.drivers.modules.inspect_utils*), 4010

`create_pxe_config()` (*in module ironic.common.pxe_utils*), 2612

`create_schema command line option`
  `--help`, 1920
  `-h`, 1920

`create_schema()` (*in module ironic.db.migration*), 3193

`create_schema()` (*in module ironic.db.sqlalchemy.migration*), 3053

`create_schema()` (*ironic.cmd.dbsync.DBCommand method*), 2399

`create_schema,`
  `ironic-dbsync command line option`, 1919

`create_vfat_image()` (*in module ironic.common.images*), 2548

`create_virtual_disk()` (*in module ironic.drivers.modules.drac.raid*), 3304

`create_virtual_disk()` (*in module ironic.drivers.modules.redfish.raid*), 3706

`create_volume_connector()` (*ironic.db.api.Connection method*), 3086

`create_volume_connector()` (*ironic.db.sqlalchemy.api.Connection method*), 2940

`create_volume_target()` (*ironic.db.api.Connection method*), 3088

`create_volume_target()` (*ironic.db.sqlalchemy.api.Connection method*), 2942

`created_at` (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057

`created_at` (*ironic.db.sqlalchemy.models.BIOSSetting attribute*), 3058

`created_at` (*ironic.db.sqlalchemy.models.Chassis attribute*), 3058

`created_at` (*ironic.db.sqlalchemy.models.Conductor attribute*), 3059

`created_at` (*ironic.db.sqlalchemy.models.ConductorHardwareI attribute*), 3060

`created_at` (*ironic.db.sqlalchemy.models.DeployTemplate attribute*), 3061

`created_at` (*ironic.db.sqlalchemy.models.DeployTemplateStep*

attribute), 3061

created_at (*ironic.db.sqlalchemy.models.Node attribute*), 3063

created_at (*ironic.db.sqlalchemy.models.NodeTag attribute*), 3066

created_at (*ironic.db.sqlalchemy.models.NodeTrait attribute*), 3066

created_at (*ironic.db.sqlalchemy.models.Port attribute*), 3067

created_at (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3068

created_at (*ironic.db.sqlalchemy.models.VolumeConnector attribute*), 3069

created_at (*ironic.db.sqlalchemy.models.VolumeTarget attribute*), 3070

created_at() (*ironic.objects.allocation.Allocation property*), 4342

created_at() (*ironic.objects.allocation.AllocationCRUDNotification property*), 4352

created_at() (*ironic.objects.allocation.AllocationCRUDPayload property*), 4352

created_at() (*ironic.objects.bios.BIOSSetting property*), 4366

created_at() (*ironic.objects.bios.BIOSSettingList property*), 4372

created_at() (*ironic.objects.chassis.Chassis property*), 4379

created_at() (*ironic.objects.chassis.ChassisCRUDNotification property*), 4386

created_at() (*ironic.objects.chassis.ChassisCRUDPayload property*), 4386

created_at() (*ironic.objects.conductor.Conductor property*), 4387

created_at() (*ironic.objects.deploy_template.DeployTemplate property*), 4396

created_at() (*ironic.objects.deploy_template.DeployTemplateCRUDNotification property*), 4407

created_at() (*ironic.objects.deploy_template.DeployTemplateCRUDPayload property*), 4407

created_at() (*ironic.objects.deployment.Deployment property*), 4409

created_at() (*ironic.objects.node.Node property*), 4436

created_at() (*ironic.objects.node.NodeConsoleNotification property*), 4454

created_at() (*ironic.objects.node.NodeCorrectedPowerStateNotification property*), 4456

created_at() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457

created_at() (*ironic.objects.node.NodeCRUDNotification property*), 4451

created_at() (*ironic.objects.node.NodeCRUDPayload property*), 4452

created_at() (*ironic.objects.node.NodeMaintenanceNotificatio property*), 4460

created_at() (*ironic.objects.node.NodePayload property*), 4461

created_at() (*ironic.objects.node.NodeSetPowerStateNotificat property*), 4463

created_at() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464

created_at() (*ironic.objects.node.NodeSetProvisionStateNotif property*), 4467

created_at() (*ironic.objects.node.NodeSetProvisionStatePaylo property*), 4468

created_at() (*ironic.objects.notification.EventType property*), 4471

created_at() (*ironic.objects.notification.NotificationBase property*), 4473

created_at() (*ironic.objects.notification.NotificationPayloadB property*), 4474

created_at() (*ironic.objects.notification.NotificationPublisher property*), 4475

created_at() (*ironic.objects.port.Port property*), 4477

created_at() (*ironic.objects.port.PortCRUDNotification property*), 4493

created_at() (*ironic.objects.port.PortCRUDPayload property*), 4494

created_at() (*ironic.objects.portgroup.Portgroup property*), 4496

created_at() (*ironic.objects.portgroup.PortgroupCRUDNotific property*), 4509

created_at() (*ironic.objects.portgroup.PortgroupCRUDPaylo property*), 4509

created_at() (*ironic.objects.trait.Trait property*), 4511

created_at() (*ironic.objects.trait.TraitList property*), 4516

created_at() (*ironic.objects.volume_connector.VolumeConnec property*), 4520

created_at() (*ironic.objects.volume_connector.VolumeConnec property*), 4533

created_at() (*ironic.objects.volume_connector.VolumeConnec property*), 4534

created_at() (*ironic.objects.volume_target.VolumeTarget property*), 4536

created_at() (*ironic.objects.volume_target.VolumeTargetCRU property*), 4551

created_at() (*ironic.objects.volume_target.VolumeTargetCRU property*), 4552

CRITICAL (*ironic.objects.fields.NotificationLevel*

attribute), 4420

D

DatabaseVersionTooOld, 2470
datatype (*ironic.api.functions.FunctionArgument attribute*), 2383
DateTimeField (*class in ironic.objects.fields*), 4416
dbapi (*ironic.objects.allocation.Allocation attribute*), 4342
dbapi (*ironic.objects.bios.BIOSSetting attribute*), 4366
dbapi (*ironic.objects.bios.BIOSSettingList attribute*), 4372
dbapi (*ironic.objects.chassis.Chassis attribute*), 4379
dbapi (*ironic.objects.conductor.Conductor attribute*), 4387
dbapi (*ironic.objects.deploy_template.DeployTemplate attribute*), 4396
dbapi (*ironic.objects.deployment.Deployment attribute*), 4409
dbapi (*ironic.objects.node.Node attribute*), 4436
dbapi (*ironic.objects.port.Port attribute*), 4477
dbapi (*ironic.objects.portgroup.Portgroup attribute*), 4496
dbapi (*ironic.objects.trait.Trait attribute*), 4511
dbapi (*ironic.objects.trait.TraitList attribute*), 4516
dbapi (*ironic.objects.volume_connector.VolumeConnector attribute*), 4520
dbapi (*ironic.objects.volume_target.VolumeTarget attribute*), 4536
DBCommand (*class in ironic.cmd.dbsync*), 2398
DBHook (*class in ironic.api.hooks*), 2391
DEBUG (*ironic.objects.fields.NotificationLevel attribute*), 4420
debug() (*ironic.common.image_format_inspector.TraceDisabled method*), 2515
deconfigure_secure_boot_if_needed() (*in module ironic.drivers.modules.boot_mode_utils*), 3851
default (*ironic.api.functions.FunctionArgument attribute*), 2383
default (*ironic.db.sqlalchemy.models.ConductorHardwareInterfaces attribute*), 3060
default_interface() (*in module ironic.common.driver_factory*), 2459
default_version() (*in module ironic.api.controllers.version*), 2376
del_host() (*ironic.conductor.base_manager.BaseConductorMa method*), 2702
delete() (*ironic.api.controllers.v1.allocation.AllocationsContro method*), 2148
delete() (*ironic.api.controllers.v1.allocation.NodeAllocationCo method*), 2154
delete() (*ironic.api.controllers.v1.chassis.ChassisController method*), 2156
delete() (*ironic.api.controllers.v1.deploy_template.DeployTemp method*), 2169
delete() (*ironic.api.controllers.v1.node.NodeMaintenanceCont method*), 2195
delete() (*ironic.api.controllers.v1.node.NodesController method*), 2215
delete() (*ironic.api.controllers.v1.node.NodeTraitsController method*), 2210
delete() (*ironic.api.controllers.v1.node.NodeVIFController method*), 2212
delete() (*ironic.api.controllers.v1.port.PortsController method*), 2240
delete() (*ironic.api.controllers.v1.portgroup.PortgroupsContro method*), 2251
delete() (*ironic.api.controllers.v1.volume_connector.VolumeCo method*), 2348
delete() (*ironic.api.controllers.v1.volume_target.VolumeTarget method*), 2360
delete() (*ironic.objects.bios.BIOSSetting class method*), 4366
delete() (*ironic.objects.bios.BIOSSettingList class method*), 4372
DELETE_ALLOWED_STATES (*in module ironic.common.states*), 2650
delete_bios_setting_list() (*ironic.db.api.Connection method*), 3091
delete_bios_setting_list() (*ironic.db.sqlalchemy.api.Connection method*), 2945
delete_configuration() (*ironic.drivers.base.RAIDInterface method*), 4273
delete_configuration() (*ironic.drivers.modules.agent.AgentRAID method*), 3762
delete_configuration() (*ironic.drivers.modules.drac.raid.DracRedfishRAID method*), 3290
delete_configuration() (*ironic.drivers.modules.drac.raid.DracWSManRAID method*), 3296
delete_configuration()

(*ironic.drivers.modules.fake.FakeRAID method*), 3972
delete_configuration() (*ironic.drivers.modules.ibmc.raid.IbmcRAID method*), 3357
delete_configuration() (*ironic.drivers.modules.ilo.raid.Ilo5RAID method*), 3494
delete_configuration() (*ironic.drivers.modules.irmc.raid.IRMCRAID method*), 3577
delete_configuration() (*ironic.drivers.modules.noop.NoRAID method*), 4098
delete_configuration() (*ironic.drivers.modules.redfish.raid.RedfishRAID method*), 3696
delete_iscsi() (*in module ironic.drivers.modules.iscsi_deploy*), 4069
delete_node_tag() (*ironic.db.api.Connection method*), 3092
delete_node_tag() (*ironic.db.sqlalchemy.api.Connection method*), 2946
delete_node_trait() (*ironic.db.api.Connection method*), 3092
delete_node_trait() (*ironic.db.sqlalchemy.api.Connection method*), 2946
delete_object() (*ironic.common.swift.SwiftAPI method*), 2664
delete_virtual_disk() (*in module ironic.drivers.modules.drac.raid*), 3306
DELETED (*in module ironic.common.states*), 2650
DELETING (*in module ironic.common.states*), 2651
deploy (*ironic.drivers.base.BareDriver attribute*), 4171
deploy() (*ironic.drivers.base.DeployInterface method*), 4197
deploy() (*ironic.drivers.modules.agent.AgentDeploy method*), 3752
deploy() (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3215
deploy() (*ironic.drivers.modules.fake.FakeDeploy method*), 3930
deploy() (*ironic.drivers.modules.iscsi_deploy.ISCSIDeploy method*), 4057
deploy() (*ironic.drivers.modules.pxe.PXEAnacondaDeploy method*), 4120
deploy() (*ironic.drivers.modules.pxe.PXERamdiskDeploy method*), 4126
deploy_disk_image() (*in module ironic.drivers.modules.iscsi_deploy*), 4069
deploy_has_started() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3795
deploy_has_started() (*ironic.drivers.modules.pxe.PXEAnacondaDeploy method*), 4121
deploy_interface (*ironic.db.sqlalchemy.models.Node attribute*), 3063
deploy_interface() (*ironic.objects.node.Node property*), 4436
deploy_interface() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457
deploy_interface() (*ironic.objects.node.NodeCRUDPayload property*), 4452
deploy_interface() (*ironic.objects.node.NodePayload property*), 4461
deploy_interface() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464
deploy_interface() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
deploy_is_done() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3796
deploy_is_done() (*ironic.drivers.modules.pxe.PXEAnacondaDeploy method*), 4121
deploy_partition_image() (*in module ironic.drivers.modules.iscsi_deploy*), 4071
deploy_step (*ironic.db.sqlalchemy.models.Node attribute*), 3063
deploy_step() (*in module ironic.drivers.base*), 4297
deploy_step() (*ironic.objects.node.Node property*), 4436
deploy_step()

(*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457
deploy_step() (*ironic.objects.node.NodeCRUDPayload property*), 4452
deploy_step() (*ironic.objects.node.NodePayload property*), 4461
deploy_step() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464
deploy_step() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
deploy_template (*ironic.db.sqlalchemy.models.DeployTemplateStep attribute*), 3061
deploy_template_id (*ironic.db.sqlalchemy.models.DeployTemplateStep attribute*), 3062
DEPLOYDONE (*in module ironic.common.states*), 2651
DEPLOYFAIL (*in module ironic.common.states*), 2651
DEPLOYING (*in module ironic.common.states*), 2651
deploying_error_handler() (*in module ironic.conductor.utils*), 2884
DeployInterface (*class in ironic.drivers.base*), 4195
Deployment (*class in ironic.objects.deployment*), 4408
DeployTemplate (*class in ironic.db.sqlalchemy.models*), 3060
DeployTemplate (*class in ironic.objects.deploy_template*), 4395
DeployTemplateAlreadyExists, 2470
DeployTemplateCRUDNotification (*class in ironic.objects.deploy_template*), 4406
DeployTemplateCRUDPayload (*class in ironic.objects.deploy_template*), 4407
DeployTemplateDuplicateName, 2471
DeployTemplateNotFound, 2471
DeployTemplatesController (*class in ironic.api.controllers.v1.deploy_template*), 2168
DeployTemplateStep (*class in ironic.db.sqlalchemy.models*), 3061
DEPLOYWAIT (*in module ironic.common.states*), 2652
DESC_MAX_SIZE (*ironic.common.image_format_inspector.VMDKInspector attribute*), 2520
DESC_OFFSET (*ironic.common.image_format_inspector.VMDKIn attribute*), 2520
description (*ironic.db.sqlalchemy.models.Chassis attribute*), 3058
description (*ironic.db.sqlalchemy.models.Node attribute*), 3063
description() (*ironic.objects.chassis.Chassis property*), 4379
description() (*ironic.objects.chassis.ChassisCRUDPayload property*), 4386
description() (*ironic.objects.node.Node property*), 4436
description() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457
description() (*ironic.objects.node.NodeCRUDPayload property*), 4452
description() (*ironic.objects.node.NodePayload property*), 4461
description() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464
description() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
deserialize_context() (*ironic.common.rpc.RequestContextSerializer method*), 2641
deserialize_entity() (*ironic.common.rpc.RequestContextSerializer method*), 2641
destroy() (*ironic.objects.allocation.Allocation method*), 4342
destroy() (*ironic.objects.chassis.Chassis method*), 4379
destroy() (*ironic.objects.deploy_template.DeployTemplate method*), 4396
destroy() (*ironic.objects.deployment.Deployment method*), 4409
destroy() (*ironic.objects.node.Node method*), 4436
destroy() (*ironic.objects.port.Port method*), 4477
destroy() (*ironic.objects.portgroup.Portgroup

method), 4496

`destroy()` (*ironic.objects.trait.Trait class method*), 4511

`destroy()` (*ironic.objects.trait.TraitList class method*), 4516

`destroy()` (*ironic.objects.volume_connector.VolumeConnector method*), 4520

`destroy()` (*ironic.objects.volume_target.VolumeTarget method*), 4536

`destroy_allocation()` (*ironic.conductor.manager.ConductorManager method*), 2723

`destroy_allocation()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2761

`destroy_allocation()` (*ironic.db.api.Connection method*), 3093

`destroy_allocation()` (*ironic.db.sqlalchemy.api.Connection method*), 2947

`destroy_chassis()` (*ironic.db.api.Connection method*), 3094

`destroy_chassis()` (*ironic.db.sqlalchemy.api.Connection method*), 2948

`destroy_deploy_template()` (*ironic.db.api.Connection method*), 3094

`destroy_deploy_template()` (*ironic.db.sqlalchemy.api.Connection method*), 2948

`destroy_floppy_image_from_web_server()` (*in module ironic.drivers.modules.ilo.common*), 3416

`destroy_http_instance_images()` (*in module ironic.drivers.modules.deploy_utils*), 3873

`destroy_images()` (*in module ironic.drivers.modules.deploy_utils*), 3873

`destroy_node()` (*ironic.conductor.manager.ConductorManager method*), 2723

`destroy_node()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2762

`destroy_node()` (*ironic.db.api.Connection method*), 3095

`destroy_node()` (*ironic.db.sqlalchemy.api.Connection method*), 2949

`destroy_port()` (*ironic.conductor.manager.ConductorManager method*), 2723

`destroy_port()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2764

`destroy_port()` (*ironic.db.api.Connection method*), 3096

`destroy_port()` (*ironic.db.sqlalchemy.api.Connection method*), 2950

`destroy_portgroup()` (*ironic.conductor.manager.ConductorManager method*), 2723

`destroy_portgroup()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2765

`destroy_portgroup()` (*ironic.db.api.Connection method*), 3096

`destroy_portgroup()` (*ironic.db.sqlalchemy.api.Connection method*), 2950

`destroy_volume_connector()` (*ironic.conductor.manager.ConductorManager method*), 2723

`destroy_volume_connector()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2766

`destroy_volume_connector()` (*ironic.db.api.Connection method*), 3097

`destroy_volume_connector()` (*ironic.db.sqlalchemy.api.Connection method*), 2951

`destroy_volume_target()` (*ironic.conductor.manager.ConductorManager method*), 2723

`destroy_volume_target()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2768

`destroy_volume_target()` (*ironic.db.api.Connection method*), 3097

`destroy_volume_target()` (*ironic.db.sqlalchemy.api.Connection method*), 2951

detach_volumes() (in module ironic.common.cinder), 2440
detach_volumes() (ironic.drivers.base.StorageInterface method), 4283
detach_volumes() (ironic.drivers.modules.fake.FakeStorage method), 3979
detach_volumes() (ironic.drivers.modules.storage.cinder.CinderStorage method), 3722
detach_volumes() (ironic.drivers.modules.storage.external.ExternalStorage method), 3727
detach_volumes() (ironic.drivers.modules.storage.noop.NoopStorage method), 3732
detail() (ironic.api.controllers.v1.chassis.ChassisController method), 2156
detail() (ironic.api.controllers.v1.node.NodesController method), 2215
detail() (ironic.api.controllers.v1.port.PortsController method), 2241
detail() (ironic.api.controllers.v1.portgroup.PortgroupsController method), 2251
detect_file_format() (in module ironic.common.image_format_inspector), 2522
detect_vendor() (ironic.drivers.base.ManagementInterface method), 4210
detect_vendor() (ironic.drivers.modules.ipmitool.IPMIManagement method), 4018
detect_vendor() (ironic.drivers.modules.redfish.management.RedfishManagement method), 3648
dev() (ironic.drivers.modules.irmc.packaging_version.Version property), 3554
dhcp_options_for_instance() (in module ironic.common.pxe_utils), 2613
DHCPFactory (class in ironic.common.dhcp_factory), 2449
DHCPLoadError, 2470
dict_valid() (in module ironic.common.args), 2411
direct_deploy_should_convert_raw_image() (in module ironic.drivers.modules.deploy_utils), 3874
DirectoryNotWritable, 2471
disallowed_fields() (in module ironic.api.controllers.v1.utils), 2311
discovery() (in module ironic.drivers.modules.iscsi_deploy), 4077
DISK (in module ironic.common.boot_devices), 2425
DISK (in module ironic.common.components), 2446
do_agent_iscsi_deploy() (in module ironic.drivers.modules.iscsi_deploy), 4077
do_allocate() (in module ironic.conductor.allocations), 2700
do_next_clean_step() (in module ironic.conductor.cleaning), 2708
do_next_deploy_step() (in module ironic.conductor.deployments), 2712
do_node_clean() (in module ironic.conductor.cleaning), 2709
do_node_clean() (ironic.conductor.manager.ConductorManager method), 2723
do_node_clean() (ironic.conductor.rpcapi.ConductorAPI method), 2769
do_node_clean_abort() (in module ironic.conductor.cleaning), 2710
do_node_deploy() (in module ironic.conductor.deployments), 2713
do_node_deploy() (ironic.conductor.manager.ConductorManager method), 2723
do_node_deploy() (ironic.conductor.rpcapi.ConductorAPI method), 2772
do_node_rescue() (ironic.conductor.manager.ConductorManager method), 2723
do_node_rescue() (ironic.conductor.rpcapi.ConductorAPI method), 2775
do_node_tear_down() (ironic.conductor.manager.ConductorManager method), 2723
do_node_tear_down() (ironic.conductor.rpcapi.ConductorAPI method), 2777
do_node_unrescue() (ironic.conductor.manager.ConductorManager method), 2723

`do_node_unrescue()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2779

`do_provisioning_action()` (*ironic.conductor.manager.ConductorManager method*), 2723

`do_provisioning_action()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2780

`do_sync_power_state()` (*in module ironic.conductor.manager*), 2731

`do_version_changes_for_db()` (*ironic.objects.base.IronicObject method*), 4357

`doc` (*ironic.api.functions.FunctionDefinition attribute*), 2385

`downgrade()` (*in module ironic.db.sqlalchemy.migration*), 3053

`downgrade_lock()` (*ironic.conductor.task_manager.TaskManager method*), 2870

`download()` (*ironic.common.glance_service.image_service.GlanceImageService method*), 2402

`download()` (*ironic.common.image_service.BaseImageService method*), 2524

`download()` (*ironic.common.image_service.FileImageService method*), 2527

`download()` (*ironic.common.image_service.HttpImageService method*), 2530

`download_size()` (*in module ironic.common.images*), 2551

`DracInspect` (*class in ironic.drivers.modules.drac.inspect*), 3249

`DracManagement` (*class in ironic.drivers.modules.drac.management*), 3258

`DracOperationError`, 2471

`DracPower` (*class in ironic.drivers.modules.drac.power*), 3278

`DracRAID` (*class in ironic.drivers.modules.drac.raid*), 3286

`DracRedfishBIOS` (*class in ironic.drivers.modules.drac.bios*), 3223

`DracRedfishInspect` (*class in ironic.drivers.modules.drac.inspect*), 3250

`DracRedfishManagement` (*class in ironic.drivers.modules.drac.management*), 3259

`DracRedfishPower` (*class in ironic.drivers.modules.drac.power*), 3279

`DracRedfishRAID` (*class in ironic.drivers.modules.drac.raid*), 3287

`DracRedfishVendorPassthru` (*class in ironic.drivers.modules.drac.vendor_passthru*), 3316

`DracRedfishVirtualMediaBoot` (*class in ironic.drivers.modules.drac.boot*), 3243

`DracVendorPassthru` (*class in ironic.drivers.modules.drac.vendor_passthru*), 3317

`DracWSManBIOS` (*class in ironic.drivers.modules.drac.bios*), 3224

`DracWSManInspect` (*class in ironic.drivers.modules.drac.inspect*), 3252

`DracWSManManagement` (*class in ironic.drivers.modules.drac.management*), 3267

`DracWSManPower` (*class in ironic.drivers.modules.drac.power*), 3280

`DracWSManRAID` (*class in ironic.drivers.modules.drac.raid*), 3291

`DracWSManVendorPassthru` (*class in ironic.drivers.modules.drac.vendor_passthru*), 3318

`driver` (*ironic.db.sqlalchemy.models.Node attribute*), 3063

`driver()` (*ironic.objects.node.Node property*), 4437

`driver()` (*ironic.objects.node.NodeCorrectedPowerStatePayloa property*), 4457

`driver()` (*ironic.objects.node.NodeCRUDPayload property*), 4452

`driver()` (*ironic.objects.node.NodePayload property*), 4461

`driver()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464

`driver()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468

`driver_info` (*ironic.db.sqlalchemy.models.Node attribute*), 3063

`driver_info()` (*ironic.objects.node.Node property*), 4437

`driver_info()` (*ironic.objects.node.NodeCRUDPayload property*), 4452

`driver_internal_info`

(*ironic.db.sqlalchemy.models.Node attribute*), 3063
driver_internal_info() (*ironic.objects.node.Node property*), 4437
driver_internal_info() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
driver_passthru() (*in module ironic.drivers.base*), 4301
driver_validate() (*ironic.drivers.base.VendorInterface method*), 4286
driver_validate() (*ironic.drivers.modules.noop.NoVendor method*), 4104
driver_vendor_passthru() (*ironic.conductor.manager.ConductorManager method*), 2723
driver_vendor_passthru() (*ironic.conductor.rpcapi.ConductorAPI method*), 2782
DriverLoadError, 2471
DriverNotFound, 2471
DriverNotFoundInEntrypoint, 2471
DriverOperationError, 2472
DriverPassthruController (*class in ironic.api.controllers.v1.driver*), 2175
DriverRaidController (*class in ironic.api.controllers.v1.driver*), 2176
drivers (*ironic.db.sqlalchemy.models.Conductor attribute*), 3059
drivers() (*ironic.objects.conductor.Conductor property*), 4387
DriversController (*class in ironic.api.controllers.v1.driver*), 2178
dump_sdr() (*in module ironic.drivers.modules.ipmitool*), 4051
Duplicate, 2472
duplicate_steps() (*in module ironic.api.controllers.v1.deploy_template*), 2173
DuplicateName, 2472

E

eat_chunk() (*ironic.common.image_format_inspector.FileInspector method*), 2504
eject_vmedia() (*in module ironic.drivers.modules.redfish.boot*), 3642
eject_vmedia() (*ironic.drivers.modules.redfish.vendor.RedfishVendorPass method*), 3718
eject_vmedia_devices() (*in module ironic.drivers.modules.ilo.common*), 3416
emit() (*ironic.objects.notification.NotificationBase method*), 4473
emit_console_notification() (*in module ironic.conductor.notification_utils*), 2734
emit_end_notification() (*in module ironic.api.controllers.v1.notification_utils*), 2237
emit_power_set_notification() (*in module ironic.conductor.notification_utils*), 2735
emit_power_state_corrected_notification() (*in module ironic.conductor.notification_utils*), 2737
emit_provision_set_notification() (*in module ironic.conductor.notification_utils*), 2738
emit_start_notification() (*in module ironic.api.controllers.v1.notification_utils*), 2238
enabled_supported_interfaces() (*in module ironic.common.driver_factory*), 2461
END (*ironic.objects.fields.NotificationStatus attribute*), 4420
ENROLL (*in module ironic.common.states*), 2652
ensure_next_boot_device() (*in module ironic.drivers.utils*), 4332
ensure_thread_contain_context() (*ironic.common.context.RequestContext method*), 2447
EnumField (*class in ironic.objects.fields*), 4416
epoch() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3555
erase_devices() (*ironic.drivers.modules.ilo.management.Ilo5Management method*), 3452
ERROR (*in module ironic.common.states*), 2653
ERROR (*ironic.objects.fields.NotificationLevel attribute*), 4420
ERROR (*ironic.objects.fields.NotificationStatus attribute*), 4421
error() (*ironic.common.image_format_inspector.TraceDisabled

method), 2515

ESSENTIAL_PROPERTIES (*ironic.drivers.base.InspectInterface attribute*), 4207

event() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468

event_type() (*ironic.objects.allocation.AllocationCRUDNotification property*), 4352

event_type() (*ironic.objects.chassis.ChassisCRUDNotification property*), 4386

event_type() (*ironic.objects.deploy_template.DeployTemplateCRUDNotification property*), 4407

event_type() (*ironic.objects.node.NodeConsoleNotification property*), 4455

event_type() (*ironic.objects.node.NodeCorrectedPowerStateNotification property*), 4456

event_type() (*ironic.objects.node.NodeCRUDNotification property*), 4451

event_type() (*ironic.objects.node.NodeMaintenanceNotification property*), 4460

event_type() (*ironic.objects.node.NodeSetPowerStateNotification property*), 4463

event_type() (*ironic.objects.node.NodeSetProvisionStateNotification property*), 4467

event_type() (*ironic.objects.notification.NotificationBase property*), 4473

event_type() (*ironic.objects.port.PortCRUDNotification property*), 4493

event_type() (*ironic.objects.portgroup.PortgroupCRUDNotification property*), 4509

event_type() (*ironic.objects.volume_connector.VolumeConnectorCRUDNotification property*), 4533

event_type() (*ironic.objects.volume_target.VolumeTargetCRUDNotification property*), 4551

events_valid() (*in module ironic.api.controllers.v1.event*), 2184

EventsController (*class in ironic.api.controllers.v1.event*), 2183

EventType (*class in ironic.objects.notification*), 4470

ExclusiveLockRequired, 2472

execute() (*in module ironic.common.utils*), 2672

execute_clean_step() (*in module ironic.drivers.modules.agent_base*), 3801

execute_clean_step() (*ironic.drivers.base.BaseInterface method*), 4176

execute_clean_step() (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3783

execute_clean_step() (*ironic.drivers.modules.agent_client.AgentClient method*), 3812

execute_clean_step() (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3215

execute_deploy_step() (*ironic.drivers.base.BaseInterface method*), 4178

execute_deploy_step() (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3784

execute_deploy_step() (*ironic.drivers.modules.agent_client.AgentClient method*), 3814

execute_oem_manager_method() (*in module ironic.drivers.modules.drac.utils*), 3314

execute_step() (*in module ironic.drivers.modules.agent_base*), 3801

exists() (*ironic.objects.trait.Trait class method*), 4513

export_configuration() (*ironic.drivers.modules.drac.management.DracRedfishMa method*), 3260

EXPORT_CONFIGURATION_ARGSINFO (*ironic.drivers.modules.drac.management.DracRedfishMa attribute*), 3259

expose() (*in module ironic.api.method*), 2396

ExternalStorage (*class in ironic.drivers.modules.storage.external*), 3726

extra (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057

extra (*ironic.db.sqlalchemy.models.Chassis attribute*), 3058

extra (*ironic.db.sqlalchemy.models.DeployTemplate attribute*), 3061

extra (*ironic.db.sqlalchemy.models.Node attribute*), 3063

extra (*ironic.db.sqlalchemy.models.Port attribute*), 3067

extra (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3068

extra (*ironic.db.sqlalchemy.models.VolumeConnector attribute*), 3069

extra (*ironic.db.sqlalchemy.models.VolumeTarget attribute*), 3070

extra (*ironic.objects.allocation.Allocation*

property), 4343
`extra()` (*ironic.objects.allocation.AllocationCRUDPayload property*), 4353
`extra()` (*ironic.objects.chassis.Chassis property*), 4379
`extra()` (*ironic.objects.chassis.ChassisCRUDPayload property*), 4386
`extra()` (*ironic.objects.deploy_template.DeployTemplate property*), 4397
`extra()` (*ironic.objects.deploy_template.DeployTemplateCRUDPayload property*), 4408
`extra()` (*ironic.objects.node.Node property*), 4437
`extra()` (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457
`extra()` (*ironic.objects.node.NodeCRUDPayload property*), 4452
`extra()` (*ironic.objects.node.NodePayload property*), 4461
`extra()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4464
`extra()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
`extra()` (*ironic.objects.port.Port property*), 4478
`extra()` (*ironic.objects.port.PortCRUDPayload property*), 4494
`extra()` (*ironic.objects.portgroup.Portgroup property*), 4497
`extra()` (*ironic.objects.portgroup.PortgroupCRUDPayload property*), 4509
`extra()` (*ironic.objects.volume_connector.VolumeConnector property*), 4521
`extra()` (*ironic.objects.volume_connector.VolumeConnectorCRUDPayload property*), 4534
`extra()` (*ironic.objects.volume_target.VolumeTarget property*), 4537
`extra()` (*ironic.objects.volume_target.VolumeTargetCRUDPayload property*), 4552
`extra_options` (*ironic.api.functions.FunctionDefinition attribute*), 2385

F

`factory_reset()` (*ironic.drivers.base.BIOSInterface method*), 4167
`factory_reset()` (*ironic.drivers.modules.drac.bios.DracWSManBIOS method*), 3227
`factory_reset()` (*ironic.drivers.modules.fake.FakeBIOS method*), 3913
`factory_reset()` (*ironic.drivers.modules.ilo.bios.IloBIOS method*), 3369
`factory_reset()` (*ironic.drivers.modules.irmc.bios.IRMCBIOS method*), 3504
`factory_reset()` (*ironic.drivers.modules.noop.NoBIOS method*), 4089
`factory_reset()` (*ironic.drivers.modules.redfish.bios.RedfishBIOS method*), 3625
`fail_on_error()` (*in module ironic.conductor.utils*), 2885
`FailedToCleanDHCPOpts`, 2472
`FailedToGetIPAddressOnPort`, 2472
`FailedToGetSensorData`, 2472
`FailedToParseSensorData`, 2472
`FailedToUpdateDHCPOptOnPort`, 2473
`FailedToUpdateMacOnPort`, 2473
`FailMixin` (*class in ironic.drivers.modules.noop*), 4085
`FakeBIOS` (*class in ironic.drivers.modules.fake*), 3909
`FakeBoot` (*class in ironic.drivers.modules.fake*), 3916
`FakeConsole` (*class in ironic.drivers.modules.fake*), 3923
`FakeDeploy` (*class in ironic.drivers.modules.fake*), 3928
`FakeHardware` (*class in ironic.drivers.fake_hardware*), 4303
`FakeInspect` (*class in ironic.drivers.modules.fake*), 3937
`FakeManagement` (*class in ironic.drivers.modules.fake*), 3941
`FakePower` (*class in ironic.drivers.modules.fake*), 3962
`FakeRAID` (*class in ironic.drivers.modules.fake*), 3969
`FakeRescue` (*class in ironic.drivers.modules.fake*), 3973
`FakeStorage` (*class in ironic.drivers.modules.fake*), 3978
`FakeVendorA` (*class in ironic.drivers.modules.fake*), 3983
`FakeVendorB` (*class in ironic.drivers.modules.fake*), 3986
`fast_track_able()` (*in module*

ironic.conductor.utils), 2886

`FASTTRACK_LOOKUP_ALLOWED_STATES` (*in module ironic.common.states*), 2653

`fault` (*ironic.db.sqlalchemy.models.Node attribute*), 3063

`fault()` (*ironic.objects.node.Node property*), 4437

`fault()` (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457

`fault()` (*ironic.objects.node.NodeCRUDPayload property*), 4452

`fault()` (*ironic.objects.node.NodePayload property*), 4461

`fault()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465

`fault()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468

`faultstring()` (*ironic.common.exception.ClientSideError property*), 2469

`faultstring()` (*ironic.common.exception.InvalidInput property*), 2480

`faultstring()` (*ironic.common.exception.UnknownArgument property*), 2489

`faultstring()` (*ironic.common.exception.UnknownAttribute property*), 2490

`fetch()` (*in module ironic.common.images*), 2551

`fetch_image()` (*ironic.drivers.modules.image_cache.ImageCache method*), 3990

`fetch_images()` (*in module ironic.drivers.modules.deploy_utils*), 3874

`fetch_into()` (*in module ironic.common.images*), 2551

`fields` (*ironic.objects.allocation.Allocation attribute*), 4343

`fields` (*ironic.objects.allocation.AllocationCRUDNotification attribute*), 4352

`fields` (*ironic.objects.allocation.AllocationCRUDPayload attribute*), 4353

`fields` (*ironic.objects.base.IronicObject attribute*), 4359

`fields` (*ironic.objects.bios.BIOSSetting attribute*), 4368

`fields` (*ironic.objects.bios.BIOSSettingList attribute*), 4374

`fields` (*ironic.objects.chassis.Chassis attribute*), 4379

`fields` (*ironic.objects.chassis.ChassisCRUDNotification attribute*), 4386

`fields` (*ironic.objects.chassis.ChassisCRUDPayload attribute*), 4387

`fields` (*ironic.objects.conductor.Conductor attribute*), 4387

`fields` (*ironic.objects.deploy_template.DeployTemplate attribute*), 4397

`fields` (*ironic.objects.deploy_template.DeployTemplateCRUDNo attribute*), 4407

`fields` (*ironic.objects.deploy_template.DeployTemplateCRUDPa attribute*), 4408

`fields` (*ironic.objects.deployment.Deployment attribute*), 4410

`fields` (*ironic.objects.node.Node attribute*), 4437

`fields` (*ironic.objects.node.NodeConsoleNotification attribute*), 4455

`fields` (*ironic.objects.node.NodeCorrectedPowerStateNotificatio attribute*), 4456

`fields` (*ironic.objects.node.NodeCorrectedPowerStatePayload attribute*), 4457

`fields` (*ironic.objects.node.NodeCRUDNotification attribute*), 4451

`fields` (*ironic.objects.node.NodeCRUDPayload attribute*), 4452

`fields` (*ironic.objects.node.NodeMaintenanceNotification attribute*), 4460

`fields` (*ironic.objects.node.NodePayload attribute*), 4461

`fields` (*ironic.objects.node.NodeSetPowerStateNotification attribute*), 4463

`fields` (*ironic.objects.node.NodeSetPowerStatePayload attribute*), 4465

`fields` (*ironic.objects.node.NodeSetProvisionStateNotification attribute*), 4467

`fields` (*ironic.objects.node.NodeSetProvisionStatePayload attribute*), 4468

`fields` (*ironic.objects.notification.EventType attribute*), 4471

`fields` (*ironic.objects.notification.NotificationBase attribute*), 4473

`fields` (*ironic.objects.notification.NotificationPayloadBase attribute*), 4474

`fields` (*ironic.objects.notification.NotificationPublisher attribute*), 4475

`fields` (*ironic.objects.port.Port attribute*), 4478

`fields` (*ironic.objects.port.PortCRUDNotification attribute*), 4494

fields (*ironic.objects.port.PortCRUDPayload attribute*), 4494
fields (*ironic.objects.portgroup.Portgroup attribute*), 4497
fields (*ironic.objects.portgroup.PortgroupCRUDNotification attribute*), 4509
fields (*ironic.objects.portgroup.PortgroupCRUDPayload attribute*), 4509
fields (*ironic.objects.trait.Trait attribute*), 4514
fields (*ironic.objects.trait.TraitList attribute*), 4517
fields (*ironic.objects.volume_connector.VolumeConnector attribute*), 4521
fields (*ironic.objects.volume_connector.VolumeConnectorCRUDNotification attribute*), 4533
fields (*ironic.objects.volume_connector.VolumeConnectorCRUDPayload attribute*), 4534
fields (*ironic.objects.volume_target.VolumeTarget attribute*), 4537
fields (*ironic.objects.volume_target.VolumeTargetCRUDNotification attribute*), 4551
fields (*ironic.objects.volume_target.VolumeTargetCRUDPayload attribute*), 4552
file_has_content() (*in module ironic.common.utils*), 2673
file_mime_type() (*in module ironic.common.utils*), 2674
FileImageService (*class in ironic.common.image_service*), 2526
FileInspector (*class in ironic.common.image_format_inspector*), 2501
FileSystemNotSupported, 2473
filter_target_raid_config() (*in module ironic.common.raid*), 2633
finalize_rescue() (*ironic.drivers.modules.agent_client.AgentClient method*), 3817
find_step() (*in module ironic.conductor.steps*), 2857
find_step() (*in module ironic.drivers.modules.agent_base*), 3803
FirmwareImageLocation (*class in ironic.drivers.modules.ilo.firmware_processor*), 3443
FirmwareProcessor (*class in ironic.drivers.modules.ilo.firmware_processor*), 3444
first_method() (*ironic.drivers.modules.fake.FakeVendorA method*), 3983
flash_firmware_sum() (*ironic.drivers.modules.ilo.management.IloManagement method*), 3457
FlatNetwork (*class in ironic.drivers.modules.network.flat*), 3594
FlexibleDict (*class in ironic.objects.fields*), 4416
FlexibleDictField (*class in ironic.objects.fields*), 4417
FLOPPY (*in module ironic.common.boot_devices*), 2425
ForbiddenPort, 2473
force_iscsi_lun_update() (*in module ironic.drivers.modules.iscsi_deploy*), 4080
force_persistent_boot() (*in module ironic.drivers.utils*), 4333
force_raw_will_convert() (*in module ironic.common.images*), 2551
format_exception() (*in module ironic.api.method*), 2396
format_match() (*ironic.common.image_format_inspector.FileInspector property*), 2504
format_match() (*ironic.common.image_format_inspector.ISOInspector property*), 2509
format_match() (*ironic.common.image_format_inspector.QcowInspector property*), 2513
format_match() (*ironic.common.image_format_inspector.QEDInspector property*), 2511
format_match() (*ironic.common.image_format_inspector.VDIInspector property*), 2515
format_match() (*ironic.common.image_format_inspector.VHDInspector property*), 2517
format_match() (*ironic.common.image_format_inspector.VHDXInspector property*), 2518
format_match() (*ironic.common.image_format_inspector.VMDKInspector property*), 2520
fourth_method_shared_lock() (*ironic.drivers.modules.fake.FakeVendorB method*), 3986
from_chassis (*ironic.api.controllers.v1.node.NodesController*

attribute), 2220
from_file() (*ironic.common.image_format_inspector.FileInspector class method*), 2504
from_power() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4457
FSM (*class in ironic.common.fsm*), 2493
FunctionArgument (*class in ironic.api.functions*), 2383
FunctionDefinition (*class in ironic.api.functions*), 2384

G

GD_AT_END (*ironic.common.image_format_inspector.VMDKInspector attribute*), 2520
GenericHardware (*class in ironic.drivers.generic*), 4307
get() (*ironic.api.controllers.v1.node.BootDeviceController method*), 2184
get() (*ironic.api.controllers.v1.node.NodeConsoleController method*), 2193
get() (*ironic.api.controllers.v1.node.NodeStatesController method*), 2197
get() (*ironic.api.controllers.v1.volume.VolumeController method*), 2347
get() (*ironic.api.functions.FunctionDefinition static method*), 2385
get() (*ironic.common.image_service.HttpImageService static method*), 2531
get() (*ironic.drivers.modules.snmp.SNMPClient method*), 4141
get() (*ironic.objects.allocation.Allocation class method*), 4343
get() (*ironic.objects.bios.BIOSSetting class method*), 4368
get() (*ironic.objects.chassis.Chassis class method*), 4379
get() (*ironic.objects.node.Node class method*), 4437
get() (*ironic.objects.port.Port class method*), 4478
get() (*ironic.objects.portgroup.Portgroup class method*), 4497
get() (*ironic.objects.volume_connector.VolumeConnector class method*), 4521
get() (*ironic.objects.volume_target.VolumeTarget class method*), 4537
get_active_hardware_type_dict() (*ironic.db.api.Connection method*), 3098
get_active_hardware_type_dict() (*ironic.db.sqlalchemy.api.Connection method*), 2952
get_adapter() (*in module ironic.common.keystone*), 2557
get_admin_context() (*in module ironic.common.context*), 2449
get_all() (*ironic.api.controllers.v1.allocation.AllocationsContr method*), 2149
get_all() (*ironic.api.controllers.v1.allocation.NodeAllocationC method*), 2154
get_all() (*ironic.api.controllers.v1.bios.NodeBiosController method*), 2155
get_all() (*ironic.api.controllers.v1.chassis.ChassisController method*), 2158
get_all() (*ironic.api.controllers.v1.conductor.ConductorsContr method*), 2165
get_all() (*ironic.api.controllers.v1.deploy_template.DeployTe method*), 2169
get_all() (*ironic.api.controllers.v1.driver.DriversController method*), 2178
get_all() (*ironic.api.controllers.v1.node.IndicatorController method*), 2187
get_all() (*ironic.api.controllers.v1.node.NodesController method*), 2221
get_all() (*ironic.api.controllers.v1.node.NodeTraitsController method*), 2211
get_all() (*ironic.api.controllers.v1.node.NodeVIFController method*), 2212
get_all() (*ironic.api.controllers.v1.port.PortsController method*), 2244
get_all() (*ironic.api.controllers.v1.portgroup.PortgroupsContr method*), 2253
get_all() (*ironic.api.controllers.v1.ramdisk.LookupController method*), 2262
get_all() (*ironic.api.controllers.v1.volume_connector.VolumeC method*), 2350
get_all() (*ironic.api.controllers.v1.volume_target.VolumeTarge method*), 2362
get_allocation_by_id() (*ironic.db.api.Connection method*), 3099
get_allocation_by_id() (*ironic.db.sqlalchemy.api.Connection method*), 2953
get_allocation_by_name() (*ironic.db.api.Connection method*), 3100
get_allocation_by_name() (*ironic.db.sqlalchemy.api.Connection method*), 2954
get_allocation_by_uuid() (*ironic.db.api.Connection method*), 3101

get_allocation_by_uuid() (ironic.db.sqlalchemy.api.Connection method), 2955
get_allocation_list() (ironic.db.api.Connection method), 3102
get_allocation_list() (ironic.db.sqlalchemy.api.Connection method), 2956
get_allowed_exmods() (in module ironic.common.rpc), 2644
get_and_validate_firmware_image_info() (in module ironic.drivers.modules.ilo.firmware_processor), 3445
get_arg() (ironic.api.functions.FunctionDefinition method), 2385
get_async_step_return_state() (in module ironic.drivers.modules.deploy_utils), 3877
get_attached_vif() (in module ironic.conductor.utils), 2887
get_auth() (in module ironic.common.keystone), 2558
get_backend() (in module ironic.db.migration), 3193
get_backend() (in module ironic.db.sqlalchemy.api), 3052
get_bios_config() (ironic.drivers.modules.drac.vendor_passthru.DracWSManVendorPassthru method), 3321
get_bios_setting() (ironic.db.api.Connection method), 3104
get_bios_setting() (ironic.db.sqlalchemy.api.Connection method), 2958
get_bios_setting_list() (ironic.db.api.Connection method), 3105
get_bios_setting_list() (ironic.db.sqlalchemy.api.Connection method), 2959
get_boot_device() (ironic.conductor.manager.ConductorManager method), 2723
get_boot_device() (ironic.conductor.rpcapi.ConductorAPI method), 2787
get_boot_device() (ironic.drivers.base.ManagementInterface method), 4211
get_boot_device() (ironic.drivers.modules.drac.management.DracWSManM
method), 3268
get_boot_device() (ironic.drivers.modules.fake.FakeManagement method), 3941
get_boot_device() (ironic.drivers.modules.ibmc.management.IBMCManagen
method), 3327
get_boot_device() (ironic.drivers.modules.ilo.management.IloManagement method), 3458
get_boot_device() (ironic.drivers.modules.ipmitool.IPMIManagement method), 4019
get_boot_device() (ironic.drivers.modules.noop_mgmt.NoopManagement method), 4106
get_boot_device() (ironic.drivers.modules.redfish.management.RedfishMana
method), 3649
get_boot_device() (ironic.drivers.modules.xclarity.management.XClarityMar
method), 3741
get_boot_mode() (in module ironic.drivers.modules.boot_mode_utils), 3852
get_boot_mode() (ironic.drivers.base.ManagementInterface method), 4213
get_boot_mode() (ironic.drivers.modules.ibmc.management.IBMCManagen
method), 3330
get_boot_mode() (ironic.drivers.modules.ilo.management.IloManagement method), 3460
get_boot_mode() (ironic.drivers.modules.redfish.management.RedfishMana
method), 3652
get_boot_mode_for_deploy() (in module ironic.drivers.modules.boot_mode_utils), 3852
get_boot_option() (in module ironic.drivers.modules.deploy_utils), 3878
get_by_address() (ironic.objects.port.Port class method), 4479
get_by_address() (ironic.objects.portgroup.Portgroup

class method), 4498
get_by_hostname() (*ironic.objects.conductor.Conductor class method*), 4387
get_by_id() (*ironic.objects.allocation.Allocation class method*), 4344
get_by_id() (*ironic.objects.chassis.Chassis class method*), 4380
get_by_id() (*ironic.objects.deploy_template.DeployTemplate class method*), 4397
get_by_id() (*ironic.objects.node.Node class method*), 4438
get_by_id() (*ironic.objects.port.Port class method*), 4480
get_by_id() (*ironic.objects.portgroup.Portgroup class method*), 4499
get_by_id() (*ironic.objects.volume_connector.VolumeConnector class method*), 4523
get_by_id() (*ironic.objects.volume_target.VolumeTarget class method*), 4538
get_by_instance_uuid() (*ironic.objects.node.Node class method*), 4438
get_by_name() (*ironic.objects.allocation.Allocation class method*), 4345
get_by_name() (*ironic.objects.deploy_template.DeployTemplate class method*), 4398
get_by_name() (*ironic.objects.node.Node class method*), 4439
get_by_name() (*ironic.objects.port.Port class method*), 4481
get_by_name() (*ironic.objects.portgroup.Portgroup class method*), 4500
get_by_node_id() (*ironic.objects.bios.BIOSSettingList class method*), 4374
get_by_node_id() (*ironic.objects.trait.TraitList class method*), 4517
get_by_node_uuid() (*ironic.objects.deployment.Deployment class method*), 4410
get_by_port_addresses() (*ironic.objects.node.Node class method*), 4440
get_by_uuid() (*ironic.objects.allocation.Allocation class method*), 4346
get_by_uuid() (*ironic.objects.chassis.Chassis class method*), 4381
get_by_uuid() (*ironic.objects.deploy_template.DeployTemplate class method*), 4400
get_by_uuid() (*ironic.objects.deployment.Deployment class method*), 4411
get_by_uuid() (*ironic.objects.node.Node class method*), 4441
get_by_uuid() (*ironic.objects.port.Port class method*), 4482
get_by_uuid() (*ironic.objects.portgroup.Portgroup class method*), 4501
get_by_uuid() (*ironic.objects.volume_connector.VolumeConnector class method*), 4524
get_by_uuid() (*ironic.objects.volume_target.VolumeTarget class method*), 4539
get_chassis_by_id() (*ironic.db.api.Connection method*), 3106
get_chassis_by_id() (*ironic.db.sqlalchemy.api.Connection method*), 2960
get_chassis_by_uuid() (*ironic.db.api.Connection method*), 3106
get_chassis_by_uuid() (*ironic.db.sqlalchemy.api.Connection method*), 2960
get_chassis_list() (*ironic.db.api.Connection method*), 3107
get_chassis_list() (*ironic.db.sqlalchemy.api.Connection method*), 2961
get_checksum_and_algo() (*in module ironic.common.checksum_utils*), 2429
get_checksum_from_url() (*in module ironic.common.checksum_utils*), 2430
get_class() (*in module ironic.db.sqlalchemy.models*), 3071
get_clean_steps() (*ironic.drivers.base.BaseInterface method*), 4180
get_clean_steps() (*ironic.drivers.modules.agent.AgentRAID

method), 3763
`get_clean_steps()` (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3786
`get_clean_steps()` (*ironic.drivers.modules.agent_client.AgentClient method*), 3819
`get_clean_steps()` (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3216
`get_cleaning_network_uuid()` (*ironic.common.neutron.NeutronNetworkInterfaceMixin method*), 2573
`get_client()` (*in module ironic.common.cinder*), 2443
`get_client()` (*in module ironic.common.neutron*), 2576
`get_client()` (*in module ironic.common.rpc*), 2644
`get_client()` (*in module ironic.drivers.modules.agent_client*), 3843
`get_command_error()` (*in module ironic.drivers.modules.agent_client*), 3843
`get_commands_status()` (*ironic.drivers.modules.agent_client.AgentClient method*), 3821
`get_conductor()` (*ironic.db.api.Connection method*), 3108
`get_conductor()` (*ironic.db.sqlalchemy.api.Connection method*), 2962
`get_conductor_for()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2790
`get_conductor_list()` (*ironic.db.api.Connection method*), 3109
`get_conductor_list()` (*ironic.db.sqlalchemy.api.Connection method*), 2963
`get_config()` (*in module ironic.drivers.modules.drac.bios*), 3232
`get_configuration()` (*in module ironic.common.molds*), 2564
`get_console()` (*ironic.drivers.base.ConsoleInterface method*), 4192
`get_console()` (*ironic.drivers.modules.fake.FakeConsole method*), 3923
`get_console()` (*ironic.drivers.modules.ipmitool.IPMIShellinaboxConsole method*), 4037
`get_console()` (*ironic.drivers.modules.ipmitool.IPMISocatConsole method*), 4040
`get_console()` (*ironic.drivers.modules.noop.NoConsole method*), 4091
`get_console_information()` (*ironic.conductor.manager.ConductorManager method*), 2724
`get_console_information()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2791
`get_controller_reserved_names()` (*in module ironic.api.controllers.v1.utils*), 2312
`get_current_boot_mode()` (*in module ironic.drivers.modules.ilo.common*), 3417
`get_current_topic()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2792
`get_current_vif()` (*ironic.drivers.base.NetworkInterface method*), 4247
`get_current_vif()` (*ironic.drivers.modules.network.common.VIFPortIDMixin method*), 3587
`get_current_vif()` (*ironic.drivers.modules.network.noop.NoopNetwork method*), 3615
`get_default_boot_option()` (*in module ironic.drivers.modules.deploy_utils*), 3879
`get_deploy_info()` (*in module ironic.drivers.modules.iscsi_deploy*), 4080
`get_deploy_steps()` (*ironic.drivers.base.BaseInterface method*), 4181
`get_deploy_steps()` (*ironic.drivers.modules.agent.AgentRAID method*), 3764
`get_deploy_steps()` (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3787
`get_deploy_steps()` (*ironic.drivers.modules.agent_client.AgentClient*

method), 3827
get_deploy_template_by_id() (*ironic.db.api.Connection method*), 3110
get_deploy_template_by_id() (*ironic.db.sqlalchemy.api.Connection method*), 2964
get_deploy_template_by_name() (*ironic.db.api.Connection method*), 3111
get_deploy_template_by_name() (*ironic.db.sqlalchemy.api.Connection method*), 2965
get_deploy_template_by_uuid() (*ironic.db.api.Connection method*), 3112
get_deploy_template_by_uuid() (*ironic.db.sqlalchemy.api.Connection method*), 2966
get_deploy_template_list() (*ironic.db.api.Connection method*), 3113
get_deploy_template_list() (*ironic.db.sqlalchemy.api.Connection method*), 2967
get_deploy_template_list_by_names() (*ironic.db.api.Connection method*), 3115
get_deploy_template_list_by_names() (*ironic.db.sqlalchemy.api.Connection method*), 2969
get_disk_label() (*in module ironic.drivers.modules.deploy_utils*), 3879
get_drac_client() (*in module ironic.drivers.modules.drac.common*), 3247
get_driver() (*ironic.common.driver_factory.BaseDriverFactory method*), 2453
get_driver_properties() (*ironic.conductor.manager.ConductorManager method*), 2724
get_driver_properties() (*ironic.conductor.rpcapi.ConductorAPI method*), 2792
get_driver_vendor_passthru_methods() (*ironic.conductor.manager.ConductorManager method*), 2724
get_driver_vendor_passthru_methods() (*ironic.conductor.rpcapi.ConductorAPI method*), 2793
get_enabled_macs() (*in module ironic.drivers.modules.redfish.utils*), 3711
get_endpoint() (*in module ironic.common.keystone*), 2559
get_enforcer() (*in module ironic.common.policy*), 2601
get_file_path_from_label() (*in module ironic.common.pxe_utils*), 2615
get_free_port_like_object() (*in module ironic.drivers.modules.network.common*), 3589
get_hardware_type() (*in module ironic.common.driver_factory*), 2462
get_http_url_path_from_label() (*in module ironic.common.pxe_utils*), 2616
get_ilo_object() (*in module ironic.drivers.modules.ilo.common*), 3418
get_image_download_source() (*in module ironic.drivers.modules.deploy_utils*), 3880
get_image_info() (*in module ironic.common.pxe_utils*), 2617
get_image_instance_info() (*in module ironic.drivers.modules.deploy_utils*), 3880
get_image_mb() (*in module ironic.common.images*), 2551
get_image_properties() (*in module ironic.common.images*), 2551
get_image_service() (*in module ironic.common.image_service*), 2535
get_indicator_state() (*ironic.conductor.manager.ConductorManager method*), 2724
get_indicator_state() (*ironic.conductor.rpcapi.ConductorAPI method*), 2796
get_indicator_state() (*ironic.drivers.base.ManagementInterface method*), 4215
get_indicator_state() (*ironic.drivers.modules.fake.FakeManagement method*), 3943
get_indicator_state() (*ironic.drivers.modules.redfish.management.RedfishMana method*), 3653
get_inspection_network_uuid() (*ironic.common.neutron.NeutronNetworkInterfaceMixin method*), 2573
get_inspector() (*in module ironic.common.image_format_inspector*),

2523
get_instance() (*in module ironic.db.api*), 3192
get_instance_image_info() (*in module ironic.common.pxe_utils*), 2619
get_interface() (*in module ironic.common.driver_factory*), 2463
get_ip_addresses() (*ironic.dhcp.base.BaseDHCP method*), 3194
get_ip_addresses() (*ironic.dhcp.neutron.NeutronDHCPApi method*), 3201
get_ip_addresses() (*ironic.dhcp.none.NoneDHCPApi method*), 3207
get_ipxe_boot_file() (*in module ironic.drivers.modules.deploy_utils*), 3882
get_ipxe_config_template() (*in module ironic.drivers.modules.deploy_utils*), 3883
get_ipxe_root_dir() (*in module ironic.common.pxe_utils*), 2621
get_irmc_client() (*in module ironic.drivers.modules.irmc.common*), 3521
get_irmc_report() (*in module ironic.drivers.modules.irmc.common*), 3523
get_ironic_api_url() (*in module ironic.drivers.modules.deploy_utils*), 3884
get_job() (*in module ironic.drivers.modules.drac.job*), 3255
get_kernel_ramdisk_info() (*in module ironic.common.pxe_utils*), 2621
get_last_command_status() (*ironic.drivers.modules.agent_client.AgentClient method*), 3829
get_local_group_information() (*in module ironic.common.neutron*), 2577
get_logical_disk_properties() (*in module ironic.common.raid*), 2635
get_logical_disk_properties() (*ironic.drivers.base.RAIDInterface method*), 4274
get_logical_disks() (*ironic.drivers.modules.drac.raid.DracWSManRAID method*), 3297
get_mac_addresses() (*ironic.drivers.base.ManagementInterface method*), 4217
get_mac_addresses() (*ironic.drivers.modules.redfish.management.RedfishMana method*), 3655
get_neutron_port_data() (*in module ironic.common.neutron*), 2580
get_next() (*in module ironic.api.controllers.v1.collection*), 2162
get_next() (*ironic.drivers.modules.snmp.SNMPClient method*), 4141
get_node_by_id() (*ironic.db.api.Connection method*), 3115
get_node_by_id() (*ironic.db.sqlalchemy.api.Connection method*), 2969
get_node_by_instance() (*ironic.db.api.Connection method*), 3116
get_node_by_instance() (*ironic.db.sqlalchemy.api.Connection method*), 2970
get_node_by_name() (*ironic.db.api.Connection method*), 3117
get_node_by_name() (*ironic.db.sqlalchemy.api.Connection method*), 2971
get_node_by_port_addresses() (*ironic.db.api.Connection method*), 3117
get_node_by_port_addresses() (*ironic.db.sqlalchemy.api.Connection method*), 2971
get_node_by_uuid() (*ironic.db.api.Connection method*), 3118
get_node_by_uuid() (*ironic.db.sqlalchemy.api.Connection method*), 2972
get_node_capability() (*in module ironic.drivers.utils*), 4334
get_node_list() (*ironic.db.api.Connection method*), 3118
get_node_list() (*ironic.db.sqlalchemy.api.Connection method*), 2972
get_node_list_columns() (*ironic.db.sqlalchemy.api.Connection method*), 2975

`get_node_mac_addresses()` (*in module ironic.drivers.utils*), 4335

`get_node_network_data()` (*ironic.drivers.base.NetworkInterface method*), 4248

`get_node_network_data()` (*ironic.drivers.modules.network.common.NeutronVIFPortIDMixin method*), 3579

`get_node_next_clean_steps()` (*in module ironic.conductor.utils*), 2888

`get_node_next_deploy_steps()` (*in module ironic.conductor.utils*), 2888

`get_node_portmap()` (*in module ironic.common.neutron*), 2581

`get_node_tags_by_node_id()` (*ironic.db.api.Connection method*), 3121

`get_node_tags_by_node_id()` (*ironic.db.sqlalchemy.api.Connection method*), 2977

`get_node_traits_by_node_id()` (*ironic.db.api.Connection method*), 3121

`get_node_traits_by_node_id()` (*ironic.db.sqlalchemy.api.Connection method*), 2978

`get_node_vendor_passthru_methods()` (*ironic.conductor.manager.ConductorManager method*), 2724

`get_node_vendor_passthru_methods()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2798

`get_node_vif_ids()` (*in module ironic.common.network*), 2567

`get_node_with_token()` (*ironic.conductor.manager.ConductorManager method*), 2724

`get_node_with_token()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2799

`get_nodeinfo_list()` (*ironic.db.api.Connection method*), 3122

`get_nodeinfo_list()` (*ironic.db.sqlalchemy.api.Connection method*), 2978

`get_nodes_controller_reserved_names()` (*in module ironic.api.controllers.v1.node*), 2231

`get_not_versions()` (*ironic.db.api.Connection method*), 3126

`get_not_versions()` (*ironic.db.sqlalchemy.api.Connection method*), 2982

`get_object_versions()` (*in module ironic.common.release_mappings*), 2639

`get_offline_conductors()` (*ironic.db.api.Connection method*), 3127

`get_offline_conductors()` (*ironic.db.sqlalchemy.api.Connection method*), 2984

`get_one()` (*ironic.api.controllers.v1.allocation.AllocationsContr method*), 2151

`get_one()` (*ironic.api.controllers.v1.bios.NodeBiosController method*), 2155

`get_one()` (*ironic.api.controllers.v1.chassis.ChassisController method*), 2159

`get_one()` (*ironic.api.controllers.v1.conductor.ConductorsContr method*), 2167

`get_one()` (*ironic.api.controllers.v1.deploy_template.DeployTen method*), 2171

`get_one()` (*ironic.api.controllers.v1.driver.DriversController method*), 2179

`get_one()` (*ironic.api.controllers.v1.node.IndicatorController method*), 2188

`get_one()` (*ironic.api.controllers.v1.node.NodesController method*), 2227

`get_one()` (*ironic.api.controllers.v1.port.PortsController method*), 2247

`get_one()` (*ironic.api.controllers.v1.portgroup.PortgroupsContr method*), 2255

`get_one()` (*ironic.api.controllers.v1.volume_connector.VolumeC method*), 2353

`get_one()` (*ironic.api.controllers.v1.volume_target.VolumeTarge method*), 2366

`get_online_conductors()` (*ironic.db.api.Connection method*), 3128

`get_online_conductors()` (*ironic.db.sqlalchemy.api.Connection method*), 2984

`get_oslo_policy_enforcer()` (*in module ironic.common.policy*), 2601

`get_partition_uuids()` (*ironic.drivers.modules.agent_client.AgentClient method*), 3830

`get_patch_values()` (*in module ironic.api.controllers.v1.utils*), 2312

`get_path_relative_to_tftp_root()` (*in module ironic.common.pxe_utils*),

2623
get_pecan_config() (*in module ironic.api.app*), 2383
get_physical_disks() (*in module ironic.drivers.modules.redfish.raid*), 3709
get_physnets_by_port_uuid() (*in module ironic.common.neutron*), 2583
get_physnets_by_portgroup_id() (*in module ironic.common.network*), 2568
get_physnets_for_node() (*in module ironic.common.network*), 2570
get_port_by_address() (*ironic.db.api.Connection method*), 3128
get_port_by_address() (*ironic.db.sqlalchemy.api.Connection method*), 2985
get_port_by_id() (*ironic.db.api.Connection method*), 3129
get_port_by_id() (*ironic.db.sqlalchemy.api.Connection method*), 2985
get_port_by_name() (*ironic.db.api.Connection method*), 3129
get_port_by_name() (*ironic.db.sqlalchemy.api.Connection method*), 2986
get_port_by_uuid() (*ironic.db.api.Connection method*), 3130
get_port_by_uuid() (*ironic.db.sqlalchemy.api.Connection method*), 2986
get_port_list() (*ironic.db.api.Connection method*), 3130
get_port_list() (*ironic.db.sqlalchemy.api.Connection method*), 2987
get_portgroup_by_address() (*ironic.db.api.Connection method*), 3131
get_portgroup_by_address() (*ironic.db.sqlalchemy.api.Connection method*), 2988
get_portgroup_by_id() (*in module ironic.common.network*), 2571
get_portgroup_by_id() (*ironic.db.api.Connection method*), 3132
get_portgroup_by_id() (*ironic.db.sqlalchemy.api.Connection method*), 2989
get_portgroup_by_name() (*ironic.db.api.Connection method*), 3133
get_portgroup_by_name() (*ironic.db.sqlalchemy.api.Connection method*), 2990
get_portgroup_by_uuid() (*ironic.db.api.Connection method*), 3134
get_portgroup_by_uuid() (*ironic.db.sqlalchemy.api.Connection method*), 2990
get_portgroup_list() (*ironic.db.api.Connection method*), 3134
get_portgroup_list() (*ironic.db.sqlalchemy.api.Connection method*), 2991
get_portgroups_by_node_id() (*ironic.db.api.Connection method*), 3136
get_portgroups_by_node_id() (*ironic.db.sqlalchemy.api.Connection method*), 2993
get_ports_by_node_id() (*ironic.db.api.Connection method*), 3138
get_ports_by_node_id() (*ironic.db.sqlalchemy.api.Connection method*), 2994
get_ports_by_portgroup_id() (*in module ironic.common.network*), 2572
get_ports_by_portgroup_id() (*ironic.db.api.Connection method*), 3139
get_ports_by_portgroup_id() (*ironic.db.sqlalchemy.api.Connection method*), 2996
get_power_state() (*ironic.drivers.base.PowerInterface method*), 4263
get_power_state() (*ironic.drivers.modules.agent_power.AgentPower method*), 3844
get_power_state() (*ironic.drivers.modules.drac.power.DracWSManPower method*), 3280
get_power_state()

(*ironic.drivers.modules.fake.FakePower method*), 3962
get_power_state() (*ironic.drivers.modules.ibmc.power.IBMCPower method*), 3343
get_power_state() (*ironic.drivers.modules.ilo.power.IloPower method*), 3482
get_power_state() (*ironic.drivers.modules.ipmitool.IPMIPower method*), 4029
get_power_state() (*ironic.drivers.modules.irmc.power.IRMCPower method*), 3566
get_power_state() (*ironic.drivers.modules.redfish.power.RedfishPower method*), 3683
get_power_state() (*ironic.drivers.modules.snmp.SNMPPower method*), 4156
get_power_state() (*ironic.drivers.modules.xclarity.power.XClarityPower method*), 3746
get_properties() (*in module ironic.drivers.modules.xclarity.common*), 3736
get_properties() (*ironic.drivers.base.BareDriver method*), 4172
get_properties() (*ironic.drivers.base.BaseInterface method*), 4183
get_properties() (*ironic.drivers.base.NetworkInterface method*), 4250
get_properties() (*ironic.drivers.base.RAIDInterface method*), 4275
get_properties() (*ironic.drivers.hardware_type.AbstractHardwareType method*), 4311
get_properties() (*ironic.drivers.modules.agent.AgentDeploy method*), 3753
get_properties() (*ironic.drivers.modules.agent.AgentRAID method*), 3765
get_properties() (*ironic.drivers.modules.agent.AgentRescue method*), 3767
get_properties() (*ironic.drivers.modules.agent_power.AgentPower method*), 3845
get_properties() (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3216
get_properties() (*ironic.drivers.modules.drac.bios.DracWSManBIOS method*), 3228
get_properties() (*ironic.drivers.modules.drac.inspect.DracWSManInspect method*), 3252
get_properties() (*ironic.drivers.modules.drac.management.DracWSManM method*), 3270
get_properties() (*ironic.drivers.modules.drac.power.DracWSManPower method*), 3281
get_properties() (*ironic.drivers.modules.drac.raid.DracWSManRAID method*), 3298
get_properties() (*ironic.drivers.modules.drac.vendor_passthru.DracWSMa method*), 3323
get_properties() (*ironic.drivers.modules.fake.FakeBIOS method*), 3914
get_properties() (*ironic.drivers.modules.fake.FakeBoot method*), 3918
get_properties() (*ironic.drivers.modules.fake.FakeConsole method*), 3924
get_properties() (*ironic.drivers.modules.fake.FakeDeploy method*), 3931
get_properties() (*ironic.drivers.modules.fake.FakeInspect method*), 3937
get_properties() (*ironic.drivers.modules.fake.FakeManagement method*), 3945
get_properties() (*ironic.drivers.modules.fake.FakePower method*), 3963
get_properties() (*ironic.drivers.modules.fake.FakeRAID method*), 3973
get_properties() (*ironic.drivers.modules.fake.FakeRescue method*), 3974
get_properties()

(*ironic.drivers.modules.fake.FakeStorage method*), 3980
`get_properties()` (*ironic.drivers.modules.fake.FakeVendorA method*), 3983
`get_properties()` (*ironic.drivers.modules.fake.FakeVendorB method*), 3986
`get_properties()` (*ironic.drivers.modules.ibmc.management.IBMCManagement method*), 3332
`get_properties()` (*ironic.drivers.modules.ibmc.power.IBMCPower method*), 3345
`get_properties()` (*ironic.drivers.modules.ibmc.raid.IbmcRAID method*), 3358
`get_properties()` (*ironic.drivers.modules.ibmc.vendor.IBMCVendor method*), 3363
`get_properties()` (*ironic.drivers.modules.ilo.bios.IloBIOS method*), 3370
`get_properties()` (*ironic.drivers.modules.ilo.boot.IloUefiHttpsBoot method*), 3379
`get_properties()` (*ironic.drivers.modules.ilo.boot.IloVirtualMediaBoot method*), 3389
`get_properties()` (*ironic.drivers.modules.ilo.console.IloConsoleInterface method*), 3441
`get_properties()` (*ironic.drivers.modules.ilo.inspect.IloInspect method*), 3448
`get_properties()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3461
`get_properties()` (*ironic.drivers.modules.ilo.power.IloPower method*), 3483
`get_properties()` (*ironic.drivers.modules.ilo.raid.Ilo5RAID method*), 3495
`get_properties()` (*ironic.drivers.modules.ilo.vendor.VendorPassthru method*), 3497
`get_properties()` (*ironic.drivers.modules.inspector.Inspector method*), 4013
`get_properties()` (*ironic.drivers.modules.ipmitool.IPMIConsole method*), 4016
`get_properties()` (*ironic.drivers.modules.ipmitool.IPMIManagement method*), 4021
`get_properties()` (*ironic.drivers.modules.ipmitool.IPMIPower method*), 4030
`get_properties()` (*ironic.drivers.modules.ipmitool.VendorPassthru method*), 4045
`get_properties()` (*ironic.drivers.modules.irmc.bios.IRMCBIOS method*), 3504
`get_properties()` (*ironic.drivers.modules.irmc.boot.IRMCVirtualMediaBoo method*), 3511
`get_properties()` (*ironic.drivers.modules.irmc.inspect.IRMCInspect method*), 3528
`get_properties()` (*ironic.drivers.modules.irmc.management.IRMCManage method*), 3533
`get_properties()` (*ironic.drivers.modules.irmc.power.IRMCPower method*), 3568
`get_properties()` (*ironic.drivers.modules.irmc.raid.IRMCRAID method*), 3578
`get_properties()` (*ironic.drivers.modules.iscsi_deploy.ISCSIDeploy method*), 4058
`get_properties()` (*ironic.drivers.modules.noop.FailMixin method*), 4085
`get_properties()` (*ironic.drivers.modules.noop_mgmt.NoopManagement method*), 4108
`get_properties()` (*ironic.drivers.modules.pxe.PXEAnacondaDeploy method*), 4122
`get_properties()` (*ironic.drivers.modules.pxe.PXERamdiskDeploy method*), 4127
`get_properties()` (*ironic.drivers.modules.pxe_base.PXEBaseMixin method*), 4133
`get_properties()` (*ironic.drivers.modules.redfish.bios.RedfishBIOS method*), 3626
`get_properties()`

(*ironic.drivers.modules.redfish.boot.RedfishVirtualMediaBoot method*), 3635
get_properties() (*ironic.drivers.modules.redfish.inspect.RedfishInspect method*), 3643
get_properties() (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3657
get_properties() (*ironic.drivers.modules.redfish.power.RedfishPower method*), 3685
get_properties() (*ironic.drivers.modules.redfish.raid.RedfishRAID method*), 3697
get_properties() (*ironic.drivers.modules.redfish.vendor.RedfishVendorPassthru method*), 3719
get_properties() (*ironic.drivers.modules.snmp.SNMPPower method*), 4158
get_properties() (*ironic.drivers.modules.storage.cinder.CinderStorage method*), 3724
get_properties() (*ironic.drivers.modules.storage.external.ExternalStorage method*), 3728
get_properties() (*ironic.drivers.modules.storage.noop.NoopStorage method*), 3733
get_properties() (*ironic.drivers.modules.xclarity.management.XClarityManagement method*), 3742
get_properties() (*ironic.drivers.modules.xclarity.power.XClarityPower method*), 3747
get_properties() (*ironic.drivers.utils.MixinVendorInterface method*), 4327
get_provisioning_network_uuid() (*ironic.common.neutron.NeutronNetworkInterfaceMixin method*), 2573
get_pxe_boot_file() (*in module ironic.drivers.modules.deploy_utils*), 3884
get_pxe_config_file_path() (*in module ironic.common.pxe_utils*), 2623
get_pxe_config_template() (*in module ironic.drivers.modules.deploy_utils*), 3885
get_raid_controller_list() (*ironic.drivers.modules.ibmc.vendor.IBMCVendor method*), 3363
get_raid_logical_disk_properties() (*ironic.conductor.manager.ConductorManager method*), 2724
get_raid_logical_disk_properties() (*ironic.conductor.rpcapi.ConductorAPI method*), 2800
get_ramdisk_logs_file_name() (*in module ironic.drivers.utils*), 4336
get_random_topic() (*ironic.conductor.rpcapi.ConductorAPI method*), 2803
get_remote_boot_volume() (*in module ironic.drivers.modules.deploy_utils*), 3886
get_request_return_fields() (*in module ironic.api.controllers.v1.utils*), 2315
get_rescuing_network_uuid() (*ironic.common.neutron.NeutronNetworkInterfaceMixin method*), 2573
get_ring() (*ironic.common.hash_ring.HashRingManager method*), 2498
get_root_device_for_deploy() (*in module ironic.drivers.modules.deploy_utils*), 3886
get_root_dir() (*in module ironic.common.pxe_utils*), 2624
get_rpc_allocation() (*in module ironic.api.controllers.v1.utils*), 2317
get_rpc_allocation_with_suffix() (*in module ironic.api.controllers.v1.utils*), 2318
get_rpc_deploy_template() (*in module ironic.api.controllers.v1.utils*), 2320
get_rpc_deploy_template_with_suffix() (*in module ironic.api.controllers.v1.utils*), 2321
get_rpc_node() (*in module ironic.api.controllers.v1.utils*), 2323
get_rpc_node_with_suffix() (*in module ironic.api.controllers.v1.utils*), 2324
get_rpc_portgroup() (*in module ironic.api.controllers.v1.utils*), 2325
get_rpc_portgroup_with_suffix() (*in module ironic.api.controllers.v1.utils*), 2326
get_secure_boot_mode() (*in module ironic.drivers.modules.ilo.common*), 3419
get_secure_boot_mode() (*in module*

ironic.drivers.modules.irmc.common), 3524
get_secure_boot_state() (*ironic.drivers.base.ManagementInterface method*), 4218
get_secure_boot_state() (*ironic.drivers.modules.ilo.management.IloManagement method*), 3461
get_secure_boot_state() (*ironic.drivers.modules.irmc.management.IRMCManagement method*), 3533
get_secure_boot_state() (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3657
get_sensors_data() (*ironic.drivers.base.ManagementInterface method*), 4219
get_sensors_data() (*ironic.drivers.modules.drac.management.DracWSManManagement method*), 3270
get_sensors_data() (*ironic.drivers.modules.fake.FakeManagement method*), 3945
get_sensors_data() (*ironic.drivers.modules.ibmc.management.IBMCManagement method*), 3332
get_sensors_data() (*ironic.drivers.modules.ilo.management.IloManagement method*), 3463
get_sensors_data() (*ironic.drivers.modules.ipmitool.IPMIManagement method*), 4022
get_sensors_data() (*ironic.drivers.modules.irmc.management.IRMCManagement method*), 3535
get_sensors_data() (*ironic.drivers.modules.noop_mgmt.NoopManagement method*), 4108
get_sensors_data() (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3658
get_sensors_data() (*ironic.drivers.modules.xclarity.management.XClarityManagement method*), 3743
get_sensors_notifier() (*in module ironic.common.rpc*), 2644
get_server() (*in module ironic.common.rpc*), 2644
get_server_hardware_id() (*in module ironic.drivers.modules.xclarity.common*), 3736
get_server_post_state() (*in module ironic.drivers.modules.ilo.common*), 3421
get_service_auth() (*in module ironic.common.keystone*), 2560
get_session() (*in module ironic.common.keystone*), 2563
get_shellinabox_console_url() (*in module ironic.drivers.modules.console_utils*), 3859
get_single_nic_with_vif_port_id() (*in module ironic.drivers.modules.deploy_utils*), 3887
get_socat_console_url() (*in module ironic.drivers.modules.console_utils*), 3859
get_source_format() (*in module ironic.common.images*), 2552
get_steps() (*in module ironic.drivers.modules.agent_base*), 3803
get_supported_boot_devices() (*ironic.conductor.manager.ConductorManager method*), 2724
get_supported_boot_devices() (*ironic.conductor.rpcapi.ConductorAPI method*), 2803
get_supported_boot_devices() (*ironic.drivers.base.ManagementInterface method*), 4225
get_supported_boot_devices() (*ironic.drivers.modules.drac.management.DracWSManMa*
method), 3271
get_supported_boot_devices() (*ironic.drivers.modules.fake.FakeManagement method*), 3951
get_supported_boot_devices() (*ironic.drivers.modules.ibmc.management.IBMCManagem*
method), 3333
get_supported_boot_devices() (*ironic.drivers.modules.ilo.management.IloManagement method*), 3465
get_supported_boot_devices() (*ironic.drivers.modules.ipmitool.IPMIManagement method*), 4023
get_supported_boot_devices() (*ironic.drivers.modules.noop_mgmt.NoopManagement method*), 4114
get_supported_boot_devices()

(*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3660

get_supported_boot_devices() (*ironic.drivers.modules.xclarity.management.XClarityManagement method*), 3743

get_supported_boot_modes() (*ironic.drivers.base.ManagementInterface method*), 4226

get_supported_boot_modes() (*ironic.drivers.modules.ibmc.management.IBMCManagement method*), 3335

get_supported_boot_modes() (*ironic.drivers.modules.ilo.management.IloManagement method*), 3465

get_supported_boot_modes() (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3661

get_supported_indicators() (*ironic.conductor.manager.ConductorManager method*), 2724

get_supported_indicators() (*ironic.conductor.rpcapi.ConductorAPI method*), 2805

get_supported_indicators() (*ironic.drivers.base.ManagementInterface method*), 4228

get_supported_indicators() (*ironic.drivers.modules.fake.FakeManagement method*), 3952

get_supported_indicators() (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3662

get_supported_power_states() (*ironic.drivers.base.PowerInterface method*), 4264

get_supported_power_states() (*ironic.drivers.modules.agent_power.AgentPower method*), 3846

get_supported_power_states() (*ironic.drivers.modules.fake.FakePower method*), 3964

get_supported_power_states() (*ironic.drivers.modules.ibmc.power.IBMCPower method*), 3345

get_supported_power_states() (*ironic.drivers.modules.ilo.power.IloPower method*), 3484

get_supported_power_states() (*ironic.drivers.modules.ipmitool.IPMIPower method*), 4031

get_supported_power_states() (*ironic.drivers.modules.irmc.power.IRMCPower method*), 3568

get_supported_power_states() (*ironic.drivers.modules.redfish.power.RedfishPower method*), 3685

get_swift_session() (*in module ironic.common.swift*), 2670

get_swift_url() (*in module ironic.drivers.modules.ilo.firmware_processor*), 3446

get_system() (*in module ironic.drivers.modules.redfish.utils*), 3712

get_target_version() (*ironic.objects.base.IronicObject class method*), 4259

get_task_monitor() (*in module ironic.drivers.modules.redfish.utils*), 3713

get_temp_url() (*ironic.common.swift.SwiftAPI method*), 2665

get_temp_url_for_glance_image() (*in module ironic.common.images*), 2552

get_tftp_path_prefix() (*in module ironic.common.pxe_utils*), 2625

get_topic_for() (*ironic.conductor.rpcapi.ConductorAPI method*), 2809

get_topic_for_driver() (*ironic.conductor.rpcapi.ConductorAPI method*), 2809

get_trait_names() (*ironic.objects.trait.TraitList method*), 4518

get_transport_url() (*in module ironic.common.rpc*), 2644

get_update_service() (*in module ironic.drivers.modules.redfish.utils*), 3715

get_updated_capabilities() (*in module ironic.common.utils*), 2674

get_vendor_passthru_metadata() (*in module ironic.conductor.manager*), 2732

get_versioned_notifier() (*in module ironic.common.rpc*), 2644

get_volume_connector_by_id() (*ironic.db.api.Connection method*), 3141

get_volume_connector_by_id() (*ironic.db.sqlalchemy.api.Connection*

method), 2998

`get_volume_connector_by_uuid()` (*ironic.db.api.Connection method*), 3142

`get_volume_connector_by_uuid()` (*ironic.db.sqlalchemy.api.Connection method*), 2999

`get_volume_connector_list()` (*ironic.db.api.Connection method*), 3143

`get_volume_connector_list()` (*ironic.db.sqlalchemy.api.Connection method*), 3000

`get_volume_connectors_by_node_id()` (*ironic.db.api.Connection method*), 3146

`get_volume_connectors_by_node_id()` (*ironic.db.sqlalchemy.api.Connection method*), 3002

`get_volume_pxe_options()` (*in module ironic.common.pxe_utils*), 2625

`get_volume_target_by_id()` (*ironic.db.api.Connection method*), 3148

`get_volume_target_by_id()` (*ironic.db.sqlalchemy.api.Connection method*), 3005

`get_volume_target_by_uuid()` (*ironic.db.api.Connection method*), 3149

`get_volume_target_by_uuid()` (*ironic.db.sqlalchemy.api.Connection method*), 3006

`get_volume_target_list()` (*ironic.db.api.Connection method*), 3150

`get_volume_target_list()` (*ironic.db.sqlalchemy.api.Connection method*), 3006

`get_volume_targets_by_node_id()` (*ironic.db.api.Connection method*), 3152

`get_volume_targets_by_node_id()` (*ironic.db.sqlalchemy.api.Connection method*), 3009

`get_volume_targets_by_volume_id()` (*ironic.db.api.Connection method*), 3155

`get_volume_targets_by_volume_id()` (*ironic.db.sqlalchemy.api.Connection method*), 3011

`get_xclarity_client()` (*in module ironic.drivers.modules.xclarity.common*), 3737

`getargspec()` (*in module ironic.api.functions*), 2386

`GlanceConnectionFailed`, 2473

`GlanceImageService` (*class in ironic.common.glance_service.image_service*), 2400

H

`handle_error_notification()` (*in module ironic.api.controllers.v1.notification_utils*), 2239

`handle_ibmc_exception()` (*in module ironic.drivers.modules.ibmc.utils*), 3359

`handle_signal()` (*ironic.common.rpc_service.RPCService method*), 2645

`handle_sync_power_state_max_retries_exceeded(` (*in module ironic.conductor.manager*), 2732

`hardware_type` (*ironic.db.sqlalchemy.models.ConductorHardwareInterfac attribute*), 3060

`hardware_types()` (*in module ironic.common.driver_factory*), 2465

`HardwareInspectionFailure`, 2473

`HardwareTypesFactory` (*class in ironic.common.driver_factory*), 2453

`has_backing_file()` (*ironic.common.image_format_inspector.QcowInspector property*), 2513

`has_data_file()` (*ironic.common.image_format_inspector.QcowInspector property*), 2513

`has_decomposed_deploy_steps` (*ironic.drivers.modules.agent.AgentDeployMixin attribute*), 3758

`has_decomposed_deploy_steps` (*ironic.drivers.modules.agent_base.HeartbeatMixin attribute*), 3796

`has_decomposed_deploy_steps` (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy attribute*), 3217

`has_decomposed_deploy_steps` (*ironic.drivers.modules.iscsi_deploy.ISCSIDeploy attribute*), 4059

`has_header()` (*ironic.common.image_format_inspector.QcowIn property*), 2513

`has_next()` (*in module ironic.api.controllers.v1.collection*), 2162

`has_region()` (*ironic.common.image_format_inspector.FileInsp*

method), 2505
has_unknown_features() (*ironic.common.image_format_inspector.QcowInspector property*), 2513
hash_password() (*in module ironic.conductor.utils*), 2888
HashRingManager (*class in ironic.common.hash_ring*), 2498
head_object() (*ironic.common.swift.SwiftAPI method*), 2667
heartbeat() (*ironic.conductor.manager.ConductorManager method*), 2724
heartbeat() (*ironic.conductor.rpcapi.ConductorAPI method*), 2810
heartbeat() (*ironic.drivers.base.DeployInterface method*), 4198
heartbeat() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3797
heartbeat_allowed_states() (*ironic.drivers.modules.agent_base.HeartbeatMixin property*), 3798
HeartbeatController (*class in ironic.api.controllers.v1.ramdisk*), 2258
HeartbeatMixin (*class in ironic.drivers.modules.agent_base*), 3794
hide_fields_in_newer_versions() (*in module ironic.api.controllers.v1.allocation*), 2154
hide_fields_in_newer_versions() (*in module ironic.api.controllers.v1.driver*), 2181
hide_fields_in_newer_versions() (*in module ironic.api.controllers.v1.node*), 2231
hide_fields_in_newer_versions() (*in module ironic.api.controllers.v1.port*), 2249
host() (*ironic.objects.notification.NotificationPublisher property*), 4475
hostname (*ironic.db.sqlalchemy.models.Conductor attribute*), 3059
hostname() (*ironic.objects.conductor.Conductor property*), 4388
HTTPForbidden, 2473
HttpImageService (*class in ironic.common.image_service*), 2530
HTTPNotFound, 2473

I

I_FEATURES (*ironic.common.image_format_inspector.QcowInsp attribute*), 2513
I_FEATURES_DATAFILE_BIT (*ironic.common.image_format_inspector.QcowInspector attribute*), 2513
I_FEATURES_LEN (*ironic.common.image_format_inspector.QcowInspector attribute*), 2513
I_FEATURES_MAX_BIT (*ironic.common.image_format_inspector.QcowInspector attribute*), 2513
IBMCConnectionError, 2474
IBMCError, 2474
IBMCHardware (*class in ironic.drivers.ibmc*), 4315
IBMCManagement (*class in ironic.drivers.modules.ibmc.management*), 3327
IBMCPower (*class in ironic.drivers.modules.ibmc.power*), 3343
IbmcRAID (*class in ironic.drivers.modules.ibmc.raid*), 3352
IBMCVendor (*class in ironic.drivers.modules.ibmc.vendor*), 3361
id (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057
id (*ironic.db.sqlalchemy.models.Chassis attribute*), 3058
id (*ironic.db.sqlalchemy.models.Conductor attribute*), 3059
id (*ironic.db.sqlalchemy.models.ConductorHardwareInterfaces attribute*), 3060
id (*ironic.db.sqlalchemy.models.DeployTemplate attribute*), 3061
id (*ironic.db.sqlalchemy.models.DeployTemplateStep attribute*), 3062
id (*ironic.db.sqlalchemy.models.Node attribute*), 3063
id (*ironic.db.sqlalchemy.models.Port attribute*), 3067
id (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3068
id (*ironic.db.sqlalchemy.models.VolumeConnector attribute*), 3070
id (*ironic.db.sqlalchemy.models.VolumeTarget attribute*), 3070
id() (*ironic.objects.allocation.Allocation prop-*

erty), 4347
id() (*ironic.objects.chassis.Chassis property*), 4382
id() (*ironic.objects.conductor.Conductor property*), 4389
id() (*ironic.objects.deploy_template.DeployTemplate property*), 4401
id() (*ironic.objects.node.Node property*), 4442
id() (*ironic.objects.port.Port property*), 4483
id() (*ironic.objects.portgroup.Portgroup property*), 4502
id() (*ironic.objects.volume_connector.VolumeConnector property*), 4525
id() (*ironic.objects.volume_target.VolumeTarget property*), 4541
IDRACHardware (*class in ironic.drivers.drac*), 4301
ignore_extra_args (*ironic.api.functions.FunctionDefinition attribute*), 2385
Ilo5Hardware (*class in ironic.drivers.ilo*), 4317
Ilo5Management (*class in ironic.drivers.modules.ilo.management*), 3451
Ilo5RAID (*class in ironic.drivers.modules.ilo.raid*), 3489
IloBIOS (*class in ironic.drivers.modules.ilo.bios*), 3367
IloConsoleInterface (*class in ironic.drivers.modules.ilo.console*), 3441
IloHardware (*class in ironic.drivers.ilo*), 4318
IloInspect (*class in ironic.drivers.modules.ilo.inspect*), 3448
IloiPXEBoot (*class in ironic.drivers.modules.ilo.boot*), 3397
IloManagement (*class in ironic.drivers.modules.ilo.management*), 3454
IloOperationError, 2474
IloOperationNotSupported, 2474
IloPower (*class in ironic.drivers.modules.ilo.power*), 3482
IloPXEBoot (*class in ironic.drivers.modules.ilo.boot*), 3373
IloUefiHttpsBoot (*class in ironic.drivers.modules.ilo.boot*), 3377
IloVirtualMediaBoot (*class in ironic.drivers.modules.ilo.boot*), 3387
image_checksum() (*ironic.objects.deployment.Deployment property*), 4412
image_ref() (*ironic.objects.deployment.Deployment property*), 4412
image_show() (*in module ironic.common.images*), 2553
image_to_raw() (*in module ironic.common.images*), 2553
ImageCache (*class in ironic.drivers.modules.image_cache*), 3989
ImageChecksumAlgorithmFailure, 2475
ImageChecksumError, 2475
ImageChecksumFileReadFailure, 2475
ImageChecksumURLNotSupported, 2476
ImageConvertFailed, 2476
ImageCreationFailed, 2476
ImageDownloadFailed, 2476
ImageFormatError, 2510
ImageHandler (*class in ironic.drivers.modules.image_utils*), 3994
ImageNotAuthorized, 2477
ImageNotFound, 2477
ImageRefValidationFailed, 2477
ImageUnacceptable, 2477
ImageUploadFailed, 2477
import_configuration() (*ironic.drivers.modules.drac.management.DracRedfishMa method*), 3262
IMPORT_CONFIGURATION_ARGSINFO (*ironic.drivers.modules.drac.management.DracRedfishMa attribute*), 3260
import_export_configuration() (*ironic.drivers.modules.drac.management.DracRedfishMa method*), 3264
IMPORT_EXPORT_CONFIGURATION_ARGSINFO (*ironic.drivers.modules.drac.management.DracRedfishMa attribute*), 3260
in_core_deploy_step() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3798
IncompatibleInterface, 2477
IncompleteLookup, 2477
index() (*ironic.api.controllers.root.RootController method*), 2376
index() (*ironic.api.controllers.v1.Controller method*), 2373
indicator_convert_with_links() (*in module ironic.api.controllers.v1.node*),

2232
indicator_list_from_dict() (*in module ironic.api.controllers.v1.node*), 2232
IndicatorAtComponent (*class in ironic.api.controllers.v1.node*), 2187
IndicatorController (*class in ironic.api.controllers.v1.node*), 2187
indicators (*ironic.api.controllers.v1.node.NodeManagementController attribute*), 2196
INFO (*ironic.objects.fields.NotificationLevel attribute*), 4420
info() (*ironic.common.image_format_inspector.TraceDisabled method*), 2515
InfoWrapper (*class in ironic.common.image_format_inspector*), 2510
init() (*in module ironic.common.rpc*), 2644
init_enforcer() (*in module ironic.common.policy*), 2601
init_host() (*ironic.conductor.base_manager.BaseConductorManager method*), 2702
initial_node_provision_state() (*in module ironic.api.controllers.v1.utils*), 2328
initialize() (*ironic.common.fsm.FSM method*), 2496
initialize_wsgi_app() (*in module ironic.api.wsgi*), 2397
inject_nmi (*ironic.api.controllers.v1.node.NodeManagementController attribute*), 2197
inject_nmi() (*ironic.conductor.manager.ConductorManager method*), 2724
inject_nmi() (*ironic.conductor.rpcapi.ConductorAPI method*), 2813
inject_nmi() (*ironic.drivers.base.ManagementInterface method*), 4234
inject_nmi() (*ironic.drivers.modules.ibmc.management.IBMCManagement method*), 3335
inject_nmi() (*ironic.drivers.modules.ilo.management.IloManagement method*), 3466
inject_nmi() (*ironic.drivers.modules.ipmitool.IPMIManagement method*), 4024
inject_nmi() (*ironic.drivers.modules.irmc.management.IRMCManagement method*), 3541
inject_nmi() (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3668
InjectNmiController (*class in ironic.api.controllers.v1.node*), 2190
InputFileError, 2478
inspect (*ironic.drivers.base.BareDriver attribute*), 4172
inspect_hardware() (*ironic.conductor.manager.ConductorManager method*), 2724
inspect_hardware() (*ironic.conductor.rpcapi.ConductorAPI method*), 2815
inspect_hardware() (*ironic.drivers.base.InspectInterface method*), 4208
inspect_hardware() (*ironic.drivers.modules.drac.inspect.DracRedfishInspect method*), 3250
inspect_hardware() (*ironic.drivers.modules.drac.inspect.DracWSManInspect method*), 3252
inspect_hardware() (*ironic.drivers.modules.fake.FakeInspect method*), 3938
inspect_hardware() (*ironic.drivers.modules.ilo.inspect.IloInspect method*), 3448
inspect_hardware() (*ironic.drivers.modules.inspector.Inspector method*), 4013
inspect_hardware() (*ironic.drivers.modules.irmc.inspect.IRMCInspect method*), 3529
inspect_hardware() (*ironic.drivers.modules.noop.NoInspect method*), 4094
inspect_hardware() (*ironic.drivers.modules.redfish.inspect.RedfishInspect method*), 3644
inspect_interface (*ironic.db.sqlalchemy.models.Node attribute*), 3064
inspect_interface() (*ironic.objects.node.Node property*), 4442
inspect_interface() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
inspect_interface() (*ironic.objects.node.NodeCRUDPayload property*), 4452
inspect_interface() (*ironic.objects.node.NodePayload property*), 4461
inspect_interface() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465

inspect_interface() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
INSPECTFAIL (*in module ironic.common.states*), 2654
INSPECTING (*in module ironic.common.states*), 2654
InspectInterface (*class in ironic.drivers.base*), 4206
inspection_finished_at (*ironic.db.sqlalchemy.models.Node attribute*), 3064
inspection_finished_at() (*ironic.objects.node.Node property*), 4442
inspection_finished_at() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
inspection_finished_at() (*ironic.objects.node.NodeCRUDPayload property*), 4452
inspection_finished_at() (*ironic.objects.node.NodePayload property*), 4461
inspection_finished_at() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
inspection_finished_at() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4468
inspection_started_at (*ironic.db.sqlalchemy.models.Node attribute*), 3064
inspection_started_at() (*ironic.objects.node.Node property*), 4442
inspection_started_at() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
inspection_started_at() (*ironic.objects.node.NodeCRUDPayload property*), 4453
inspection_started_at() (*ironic.objects.node.NodePayload property*), 4461
inspection_started_at() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
inspection_started_at() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
Inspector (*class in ironic.drivers.modules.inspector*), 4012
INSPECTWAIT (*in module ironic.common.states*), 2654
install_bootloader() (*ironic.drivers.modules.agent_client.AgentClient method*), 3831
instance_info (*ironic.db.sqlalchemy.models.Node attribute*), 3064
instance_info() (*ironic.objects.node.Node property*), 4442
instance_info() (*ironic.objects.node.NodeCRUDPayload property*), 4453
instance_info() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
instance_info_mapping (*ironic.objects.deployment.Deployment attribute*), 4412
instance_info_mapping_rev (*ironic.objects.deployment.Deployment attribute*), 4412
instance_uuid (*ironic.db.sqlalchemy.models.Node attribute*), 3064
instance_uuid() (*ironic.objects.node.Node property*), 4442
instance_uuid() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
instance_uuid() (*ironic.objects.node.NodeCRUDPayload property*), 4453
instance_uuid() (*ironic.objects.node.NodePayload property*), 4461
instance_uuid() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
instance_uuid() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
InstanceAssociated, 2478
InstanceDeployFailure, 2478
InstanceImageCache (*class in ironic.drivers.modules.deploy_utils*), 3865
InstanceNotFound, 2478
InstanceRescueFailure, 2478

InstanceUnrescueFailure, 2478

InsufficentMemory, 2478

InsufficientDiskSpace, 2478

integer() (*in module ironic.common.args*), 2412

IntegerField (*class in ironic.objects.fields*), 4418

IntelIPMIHardware (*class in ironic.drivers.intel_ipmi*), 4320

IntelIPMIManagement (*class in ironic.drivers.modules.intel_ipmi.management*), 3500

interface (*ironic.db.sqlalchemy.models.DeployTemplateStep attribute*), 3062

interface_name (*ironic.db.sqlalchemy.models.ConductorHardwareInterfaces attribute*), 3060

interface_type (*ironic.db.sqlalchemy.models.ConductorHardwareInterfaces attribute*), 3060

interface_type (*ironic.drivers.base.BaseInterface attribute*), 4183

interface_type (*ironic.drivers.base.BIOSInterface attribute*), 4169

interface_type (*ironic.drivers.base.BootInterface attribute*), 4187

interface_type (*ironic.drivers.base.ConsoleInterface attribute*), 4193

interface_type (*ironic.drivers.base.DeployInterface attribute*), 4199

interface_type (*ironic.drivers.base.InspectInterface attribute*), 4209

interface_type (*ironic.drivers.base.ManagementInterface attribute*), 4235

interface_type (*ironic.drivers.base.NetworkInterface attribute*), 4251

interface_type (*ironic.drivers.base.PowerInterface attribute*), 4264

interface_type (*ironic.drivers.base.RAIDInterface attribute*), 4275

interface_type (*ironic.drivers.base.RescueInterface attribute*), 4280

interface_type (*ironic.drivers.base.StorageInterface attribute*), 4284

interface_type (*ironic.drivers.base.VendorInterface attribute*), 4287

InterfaceNotFoundInEntrypoint, 2479

interfaces() (*in module ironic.common.driver_factory*), 2466

internal_info (*ironic.db.sqlalchemy.models.Port attribute*), 3067

internal_info (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3068

internal_info() (*ironic.objects.port.Port property*), 4483

internal_info() (*ironic.objects.portgroup.Portgroup property*), 4502

Invalid, 2479

invalid_sort_key_list (*ironic.api.controllers.v1.allocation.AllocationsController attribute*), 2152

invalid_sort_key_list (*ironic.api.controllers.v1.allocation.NodeAllocationContr attribute*), 2154

invalid_sort_key_list (*ironic.api.controllers.v1.chassis.ChassisController attribute*), 2160

invalid_sort_key_list (*ironic.api.controllers.v1.conductor.ConductorsController attribute*), 2168

invalid_sort_key_list (*ironic.api.controllers.v1.deploy_template.DeployTemplat attribute*), 2172

invalid_sort_key_list (*ironic.api.controllers.v1.node.NodesController attribute*), 2228

invalid_sort_key_list (*ironic.api.controllers.v1.port.PortsController attribute*), 2248

invalid_sort_key_list (*ironic.api.controllers.v1.portgroup.PortgroupsController attribute*), 2256

invalid_sort_key_list (*ironic.api.controllers.v1.volume_connector.VolumeConne attribute*), 2355

invalid_sort_key_list

(*ironic.api.controllers.v1.volume_target.VolumeTargetsController attribute*), 2367
InvalidConductorGroup, 2479
InvalidDatapathID, 2479
InvalidDeployTemplate, 2479
InvalidEndpoint, 2479
InvalidIdentity, 2480
InvalidImage, 2480
InvalidImageRef, 2480
InvalidInput, 2480
InvalidIPAddress, 2479
InvalidIPv4Address, 2480
InvalidKickstartFile, 2480
InvalidKickstartTemplate, 2480
InvalidMAC, 2481
InvalidName, 2481
InvalidParameterValue, 2481
InvalidState, 2481
InvalidStateRequested, 2481
InvalidSwitchID, 2481
InvalidUUID, 2481
InvalidUuidOrName, 2482
InvalidVersion, 3549
IPMIConsole (*class in ironic.drivers.modules.ipmitool*), 4016
IPMIFailure, 2474
IPMIHardware (*class in ironic.drivers.ipmi*), 4321
IPMIManagement (*class in ironic.drivers.modules.ipmitool*), 4017
IPMIPower (*class in ironic.drivers.modules.ipmitool*), 4029
IPMIShellinaboxConsole (*class in ironic.drivers.modules.ipmitool*), 4037
IPMISocatConsole (*class in ironic.drivers.modules.ipmitool*), 4040
ipxe_enabled (*ironic.drivers.modules.ipxe.iPXEBoot attribute*), 4056
ipxe_enabled (*ironic.drivers.modules.pxe_base.PXEBaseMixin attribute*), 4134
iPXEBoot (*class in ironic.drivers.modules.ipxe*), 4055
IRMCBIOS (*class in ironic.drivers.modules.irmc.bios*), 3501
IRMCHardware (*class in ironic.drivers.irmc*), 4322
IRMCInspect (*class in ironic.drivers.modules.irmc.inspect*), 3528
IRMCManagement (*class in ironic.drivers.modules.irmc.management*), 3563
IRMCOperationError, 2474
IRMCPower (*class in ironic.drivers.modules.irmc.power*), 3566
IRMCPXEBoot (*class in ironic.drivers.modules.irmc.boot*), 3506
IRMCRAID (*class in ironic.drivers.modules.irmc.raid*), 3574
IRMCSharedFileSystemNotMounted, 2474
IRMCVirtualMediaBoot (*class in ironic.drivers.modules.irmc.boot*), 3508
IRMCVolumeBootMixIn (*class in ironic.drivers.modules.irmc.boot*), 3517
ironic
  module, 4552
ironic.api
  module, 2397
ironic.api.app
  module, 2382
ironic.api.config
  module, 2383
ironic.api.controllers
  module, 2378
ironic.api.controllers.base
  module, 2374
ironic.api.controllers.link
  module, 2376
ironic.api.controllers.root
  module, 2376
ironic.api.controllers.v1
  module, 2373
ironic.api.controllers.v1.allocation
  module, 2148
ironic.api.controllers.v1.bios
  module, 2155
ironic.api.controllers.v1.chassis
  module, 2156
ironic.api.controllers.v1.collection
  module, 2162
ironic.api.controllers.v1.conductor
  module, 2165
ironic.api.controllers.v1.deploy_template
  module, 2168
ironic.api.controllers.v1.driver
  module, 2175
ironic.api.controllers.v1.event

module, 2183
ironic.api.controllers.v1.node
module, 2184
ironic.api.controllers.v1.notification_utils
module, 2237
ironic.api.controllers.v1.port
module, 2240
ironic.api.controllers.v1.portgroup
module, 2251
ironic.api.controllers.v1.ramdisk
module, 2258
ironic.api.controllers.v1.utils
module, 2264
ironic.api.controllers.v1.versions
module, 2346
ironic.api.controllers.v1.volume
module, 2347
ironic.api.controllers.v1.volume_connector
module, 2347
ironic.api.controllers.v1.volume_target
module, 2360
ironic.api.controllers.version
module, 2376
ironic.api.functions
module, 2383
ironic.api.hooks
module, 2389
ironic.api.method
module, 2396
ironic.api.middleware
module, 2380
ironic.api.middleware.auth_public_routes
module, 2378
ironic.api.middleware.json_ext
module, 2379
ironic.api.middleware.parsable_error
module, 2380
ironic.api.wsgi
module, 2397
ironic.cmd
module, 2400
ironic.cmd.api
module, 2397
ironic.cmd.conductor
module, 2397
ironic.cmd.dbsync
module, 2398
ironic.cmd.status
module, 2399
ironic.common
module, 2697
ironic.common.args
module, 2408
ironic.common.boot_devices
module, 2424
ironic.common.boot_modes
module, 2426
ironic.common.checksum_utils
module, 2427
ironic.common.cinder
module, 2433
ironic.common.components
module, 2446
ironic.common.config
module, 2447
ironic.common.context
module, 2447
ironic.common.dhcp_factory
module, 2449
ironic.common.driver_factory
module, 2452
ironic.common.exception
module, 2467
ironic.common.faults
module, 2492
ironic.common.fsm
module, 2493
ironic.common.glance_service
module, 2408
ironic.common.glance_service.image_service
module, 2400
ironic.common.glance_service.service_utils
module, 2406
ironic.common.hash_ring
module, 2498
ironic.common.i18n
module, 2498
ironic.common.image_format_inspector
module, 2499
ironic.common.image_service
module, 2524
ironic.common.images
module, 2537
ironic.common.indicator_states
module, 2556
ironic.common.keystone
module, 2557
ironic.common.molds
module, 2564
ironic.common.network
module, 2567
ironic.common.neutron

module, 2573
ironic.common.nova
module, 2598
ironic.common.policy
module, 2599
ironic.common.profiler
module, 2603
ironic.common.pxe_utils
module, 2605
ironic.common.qemu_img
module, 2632
ironic.common.raid
module, 2632
ironic.common.release_mappings
module, 2639
ironic.common.rpc
module, 2641
ironic.common.rpc_service
module, 2645
ironic.common.service
module, 2646
ironic.common.states
module, 2646
ironic.common.swift
module, 2662
ironic.common.utils
module, 2670
ironic.common.wsgi_service
module, 2696
ironic.conductor
module, 2918
ironic.conductor.allocations
module, 2697
ironic.conductor.base_manager
module, 2702
ironic.conductor.cleaning
module, 2707
ironic.conductor.deployments
module, 2711
ironic.conductor.manager
module, 2717
ironic.conductor.notification_utils
module, 2734
ironic.conductor.rpcapi
module, 2739
ironic.conductor.steps
module, 2857
ironic.conductor.task_manager
module, 2862
ironic.conductor.utils
module, 2878
ironic.conf
module, 2926
ironic.conf.agent
module, 2918
ironic.conf.anaconda
module, 2918
ironic.conf.ansible
module, 2919
ironic.conf.api
module, 2919
ironic.conf.audit
module, 2919
ironic.conf.auth
module, 2919
ironic.conf.cinder
module, 2920
ironic.conf.conductor
module, 2920
ironic.conf.console
module, 2920
ironic.conf.database
module, 2921
ironic.conf.default
module, 2921
ironic.conf.deploy
module, 2921
ironic.conf.dhcp
module, 2921
ironic.conf.disk_utils
module, 2921
ironic.conf.drac
module, 2921
ironic.conf.glance
module, 2921
ironic.conf.healthcheck
module, 2922
ironic.conf.ibmc
module, 2922
ironic.conf.ilo
module, 2922
ironic.conf.inspector
module, 2922
ironic.conf.ipmi
module, 2922
ironic.conf.irmc
module, 2922
ironic.conf.iscsi
module, 2922
ironic.conf.metrics
module, 2923
ironic.conf.metrics_statsd

module, 2923
ironic.conf.molds
module, 2923
ironic.conf.neutron
module, 2923
ironic.conf.nova
module, 2923
ironic.conf.opts
module, 2923
ironic.conf.pxe
module, 2925
ironic.conf.redfish
module, 2925
ironic.conf.service_catalog
module, 2925
ironic.conf.snmp
module, 2925
ironic.conf.swift
module, 2925
ironic.conf.xclarity
module, 2926
ironic.db
module, 3193
ironic.db.api
module, 3072
ironic.db.migration
module, 3193
ironic.db.sqlalchemy
module, 3072
ironic.db.sqlalchemy.api
module, 2926
ironic.db.sqlalchemy.migration
module, 3053
ironic.db.sqlalchemy.models
module, 3056
ironic.dhcp
module, 3214
ironic.dhcp.base
module, 3194
ironic.dhcp.neutron
module, 3201
ironic.dhcp.none
module, 3207
ironic.drivers
module, 4341
ironic.drivers.base
module, 4163
ironic.drivers.drac
module, 4301
ironic.drivers.fake_hardware
module, 4303
ironic.drivers.generic
module, 4307
ironic.drivers.hardware_type
module, 4310
ironic.drivers.ibmc
module, 4315
ironic.drivers.ilo
module, 4316
ironic.drivers.intel_ipmi
module, 4320
ironic.drivers.ipmi
module, 4321
ironic.drivers.irmc
module, 4322
ironic.drivers.modules
module, 4163
ironic.drivers.modules.agent
module, 3750
ironic.drivers.modules.agent_base
module, 3774
ironic.drivers.modules.agent_client
module, 3810
ironic.drivers.modules.agent_power
module, 3844
ironic.drivers.modules.ansible
module, 3223
ironic.drivers.modules.ansible.deploy
module, 3214
ironic.drivers.modules.boot_mode_utils
module, 3851
ironic.drivers.modules.console_utils
module, 3858
ironic.drivers.modules.deploy_utils
module, 3865
ironic.drivers.modules.drac
module, 3327
ironic.drivers.modules.drac.bios
module, 3223
ironic.drivers.modules.drac.boot
module, 3243
ironic.drivers.modules.drac.common
module, 3247
ironic.drivers.modules.drac.inspect
module, 3249
ironic.drivers.modules.drac.job
module, 3255
ironic.drivers.modules.drac.management
module, 3258
ironic.drivers.modules.drac.power
module, 3278
ironic.drivers.modules.drac.raid

module, 3286
ironic.drivers.modules.drac.utils
module, 3314
ironic.drivers.modules.drac.vendor_passthru
module, 3316
ironic.drivers.modules.fake
module, 3908
ironic.drivers.modules.ibmc
module, 3367
ironic.drivers.modules.ibmc.management
module, 3327
ironic.drivers.modules.ibmc.mappings
module, 3343
ironic.drivers.modules.ibmc.power
module, 3343
ironic.drivers.modules.ibmc.raid
module, 3352
ironic.drivers.modules.ibmc.utils
module, 3359
ironic.drivers.modules.ibmc.vendor
module, 3361
ironic.drivers.modules.ilo
module, 3500
ironic.drivers.modules.ilo.bios
module, 3367
ironic.drivers.modules.ilo.boot
module, 3372
ironic.drivers.modules.ilo.common
module, 3404
ironic.drivers.modules.ilo.console
module, 3441
ironic.drivers.modules.ilo.firmware_processor
module, 3443
ironic.drivers.modules.ilo.inspect
module, 3448
ironic.drivers.modules.ilo.management
module, 3451
ironic.drivers.modules.ilo.power
module, 3482
ironic.drivers.modules.ilo.raid
module, 3489
ironic.drivers.modules.ilo.vendor
module, 3495
ironic.drivers.modules.image_cache
module, 3989
ironic.drivers.modules.image_utils
module, 3994
ironic.drivers.modules.inspect_utils
module, 4010
ironic.drivers.modules.inspector
module, 4012
ironic.drivers.modules.intel_ipmi
module, 3501
ironic.drivers.modules.intel_ipmi.management
module, 3500
ironic.drivers.modules.ipmitool
module, 4015
ironic.drivers.modules.ipxe
module, 4055
ironic.drivers.modules.irmc
module, 3578
ironic.drivers.modules.irmc.bios
module, 3501
ironic.drivers.modules.irmc.boot
module, 3506
ironic.drivers.modules.irmc.common
module, 3521
ironic.drivers.modules.irmc.inspect
module, 3528
ironic.drivers.modules.irmc.management
module, 3533
ironic.drivers.modules.irmc.packaging_versio
module, 3549
ironic.drivers.modules.irmc.power
module, 3566
ironic.drivers.modules.irmc.raid
module, 3574
ironic.drivers.modules.iscsi_deploy
module, 4056
ironic.drivers.modules.network
module, 3622
ironic.drivers.modules.network.common
module, 3578
ironic.drivers.modules.network.flat
module, 3594
ironic.drivers.modules.network.neutron
module, 3603
ironic.drivers.modules.network.noop
module, 3613
ironic.drivers.modules.noop
module, 4084
ironic.drivers.modules.noop_mgmt
module, 4105
ironic.drivers.modules.pxe
module, 4118
ironic.drivers.modules.pxe_base
module, 4131
ironic.drivers.modules.redfish
module, 3721
ironic.drivers.modules.redfish.bios
module, 3622
ironic.drivers.modules.redfish.boot

module, 3630
ironic.drivers.modules.redfish.inspect
module, 3643
ironic.drivers.modules.redfish.management
module, 3647
ironic.drivers.modules.redfish.power
module, 3683
ironic.drivers.modules.redfish.raid
module, 3691
ironic.drivers.modules.redfish.utils
module, 3711
ironic.drivers.modules.redfish.vendor
module, 3718
ironic.drivers.modules.snmp
module, 4140
ironic.drivers.modules.storage
module, 3736
ironic.drivers.modules.storage.cinder
module, 3721
ironic.drivers.modules.storage.external
module, 3726
ironic.drivers.modules.storage.noop
module, 3731
ironic.drivers.modules.xclarity
module, 3750
ironic.drivers.modules.xclarity.common
module, 3736
ironic.drivers.modules.xclarity.management
module, 3741
ironic.drivers.modules.xclarity.power
module, 3746
ironic.drivers.redfish
module, 4325
ironic.drivers.snmp
module, 4326
ironic.drivers.utils
module, 4327
ironic.drivers.xclarity
module, 4340
ironic.objects
module, 4552
ironic.objects.allocation
module, 4341
ironic.objects.base
module, 4353
ironic.objects.bios
module, 4365
ironic.objects.chassis
module, 4378
ironic.objects.conductor
module, 4387
ironic.objects.deploy_template
module, 4395
ironic.objects.deployment
module, 4408
ironic.objects.fields
module, 4416
ironic.objects.indirection
module, 4424
ironic.objects.node
module, 4434
ironic.objects.notification
module, 4470
ironic.objects.port
module, 4476
ironic.objects.portgroup
module, 4495
ironic.objects.trait
module, 4510
ironic.objects.volume_connector
module, 4519
ironic.objects.volume_target
module, 4534
ironic.version
module, 4552
ironic-dbsync command line option
--config-dir <DIR>, 1919
--config-file <PATH>, 1919
--debug, 1919
--help, 1919
--version, 1919
-d, 1919
-h, 1919
create_schema,, 1919
online_data_migrations, 1919
revision, 1919
stamp, 1919
upgrade, 1919
version, 1919
IronicBase (*class in ironic.db.sqlalchemy.models*), 3062
IronicCORS (*class in ironic.api.app*), 2382
IronicObject (*class in ironic.objects.base*), 4353
IronicObjectIndirectionAPI (*class in ironic.objects.indirection*), 4424
IronicObjectListBase (*class in ironic.objects.base*), 4361
IronicObjectRegistry (*class in ironic.objects.base*), 4362
IronicObjectSerializer (*class in ironic.objects.base*), 4362

is_agent_token_pregenerated() (*in module ironic.conductor.utils*), 2888
is_agent_token_present() (*in module ironic.conductor.utils*), 2890
is_agent_token_valid() (*in module ironic.conductor.utils*), 2890
is_anaconda_deploy() (*in module ironic.drivers.modules.deploy_utils*), 3888
is_checksum_url() (*in module ironic.common.checksum_utils*), 2431
is_devrelease() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3556
is_equivalent() (*in module ironic.conductor.steps*), 2857
is_fast_track() (*in module ironic.conductor.utils*), 2891
is_fips_enabled() (*in module ironic.common.utils*), 2676
is_glance_image() (*in module ironic.common.glance_service.service_utils*), 2406
is_hostname_safe() (*in module ironic.common.utils*), 2676
is_image_active() (*in module ironic.common.glance_service.service_utils*), 2406
is_image_available() (*in module ironic.common.glance_service.service_utils*), 2407
is_ipxe_enabled() (*in module ironic.common.pxe_utils*), 2625
is_iscsi_boot() (*in module ironic.drivers.modules.deploy_utils*), 3889
is_memory_insuffcient() (*in module ironic.common.utils*), 2678
is_path_removed() (*in module ironic.api.controllers.v1.utils*), 2329
is_path_updated() (*in module ironic.api.controllers.v1.utils*), 2330
is_postrelease() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3556
is_prerelease() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3557
is_ramdisk_deploy() (*in module ironic.drivers.modules.deploy_utils*), 3889
is_regex_string_in_file() (*in module ironic.common.utils*), 2680
is_secure_boot_requested() (*in module ironic.drivers.modules.boot_mode_utils*), 3853
is_smartnic (*ironic.db.sqlalchemy.models.Port attribute*), 3067
is_smartnic() (*ironic.objects.port.Port property*), 4483
is_smartnic() (*ironic.objects.port.PortCRUDPayload property*), 4494
is_smartnic_port() (*in module ironic.common.neutron*), 2585
is_software_raid() (*in module ironic.drivers.modules.deploy_utils*), 3889
is_stable() (*ironic.common.fsm.FSM method*), 2496
is_trusted_boot_requested() (*in module ironic.drivers.modules.boot_mode_utils*), 3855
is_valid_datapath_id() (*in module ironic.common.utils*), 2680
is_valid_logical_name() (*in module ironic.api.controllers.v1.utils*), 2330
is_valid_logical_name() (*in module ironic.common.utils*), 2681
is_valid_no_proxy() (*in module ironic.common.utils*), 2682
is_valid_node_name() (*in module ironic.api.controllers.v1.utils*), 2331
is_volume_attached() (*in module ironic.common.cinder*), 2444
is_volume_available() (*in module ironic.common.cinder*), 2445
is_whole_disk_image() (*in module ironic.common.images*), 2553
ISCSIBOOT (*in module ironic.common.boot_devices*), 2425
ISCSIDeploy (*class in ironic.drivers.modules.iscsi_deploy*), 4056
ISOInspector (*class in ironic.common.image_format_inspector*), 2507
issue_startup_warnings() (*in module ironic.cmd.conductor*), 2397
iswsmefunction() (*in module ironic.api.functions*), 2386

items() (*ironic.common.driver_factory.BaseDriverFactory method*), 2453
iter_nodes() (*ironic.conductor.base_manager.BaseConductorManager method*), 2704

J

JsonExtensionMiddleware (*class in ironic.api.middleware*), 2381
JsonExtensionMiddleware (*class in ironic.api.middleware.json_ext*), 2379

K

kernel_ref() (*ironic.objects.deployment.Deployment property*), 4412
KeystoneFailure, 2482
KeystoneUnauthorized, 2482
known_good_state() (*ironic.drivers.modules.drac.management.DracRedfishManagement method*), 3265
known_good_state() (*ironic.drivers.modules.drac.management.DracWSManManagement method*), 3272
ks_exceptions() (*in module ironic.common.keystone*), 2563

L

last_error (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057
last_error (*ironic.db.sqlalchemy.models.Node attribute*), 3064
last_error() (*ironic.objects.allocation.Allocation property*), 4347
last_error() (*ironic.objects.allocation.AllocationCRUDPayload property*), 4353
last_error() (*ironic.objects.node.Node property*), 4442
last_error() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
last_error() (*ironic.objects.node.NodeCRUDPayload property*), 4453
last_error() (*ironic.objects.node.NodePayload property*), 4461
last_error() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
last_error() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
LEGACY_BIOS (*in module ironic.common.boot_modes*), 2427
lessee (*ironic.db.sqlalchemy.models.Node attribute*), 3064
lessee() (*ironic.objects.node.Node property*), 4442
lessee() (*ironic.objects.node.NodeCorrectedPowerStatePayloa property*), 4458
lessee() (*ironic.objects.node.NodeCRUDPayload property*), 4453
lessee() (*ironic.objects.node.NodePayload property*), 4461
lessee() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
lessee() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
level() (*ironic.objects.allocation.AllocationCRUDNotification property*), 4352
level() (*ironic.objects.chassis.ChassisCRUDNotification property*), 4386
level() (*ironic.objects.deploy_template.DeployTemplateCRUDN property*), 4407
level() (*ironic.objects.node.NodeConsoleNotification property*), 4455
level() (*ironic.objects.node.NodeCorrectedPowerStateNotificati property*), 4456
level() (*ironic.objects.node.NodeCRUDNotification property*), 4451
level() (*ironic.objects.node.NodeMaintenanceNotification property*), 4460
level() (*ironic.objects.node.NodeSetPowerStateNotification property*), 4463
level() (*ironic.objects.node.NodeSetProvisionStateNotification property*), 4467
level() (*ironic.objects.notification.NotificationBase property*), 4473
level() (*ironic.objects.port.PortCRUDNotification property*), 4494
level() (*ironic.objects.portgroup.PortgroupCRUDNotification property*), 4509
level() (*ironic.objects.volume_connector.VolumeConnectorCRU property*), 4533
level() (*ironic.objects.volume_target.VolumeTargetCRUDNotifi property*), 4551
list() (*ironic.objects.allocation.Allocation class method*), 4347
list() (*ironic.objects.chassis.Chassis class method*), 4382
list() (*ironic.objects.conductor.Conductor class method*), 4389
list() (*ironic.objects.deploy_template.DeployTemplate class method*), 4401
list() (*ironic.objects.deployment.Deployment class method*), 4412
list() (*ironic.objects.node.Node class method*), 4442
list() (*ironic.objects.port.Port class method*),

4483

list() (*ironic.objects.portgroup.Portgroup class method*), 4502

list() (*ironic.objects.volume_connector.VolumeConnector class method*), 4525

list() (*ironic.objects.volume_target.VolumeTarget class method*), 4541

list_by_names() (*ironic.objects.deploy_template.DeployTemplate class method*), 4403

list_by_node_id() (*ironic.objects.port.Port class method*), 4485

list_by_node_id() (*ironic.objects.portgroup.Portgroup class method*), 4504

list_by_node_id() (*ironic.objects.volume_connector.VolumeConnector class method*), 4527

list_by_node_id() (*ironic.objects.volume_target.VolumeTarget class method*), 4543

list_by_portgroup_id() (*ironic.objects.port.Port class method*), 4487

list_by_volume_id() (*ironic.objects.volume_target.VolumeTarget class method*), 4545

list_conductor_hardware_interfaces() (*ironic.db.api.Connection method*), 3157

list_conductor_hardware_interfaces() (*ironic.db.sqlalchemy.api.Connection method*), 3013

list_convert_with_links() (*in module ironic.api.controllers.v1.allocation*), 2154

list_convert_with_links() (*in module ironic.api.controllers.v1.chassis*), 2161

list_convert_with_links() (*in module ironic.api.controllers.v1.collection*), 2162

list_convert_with_links() (*in module ironic.api.controllers.v1.conductor*), 2168

list_convert_with_links() (*in module ironic.api.controllers.v1.deploy_template*), 2174

list_convert_with_links() (*in module ironic.api.controllers.v1.driver*), 2182

list_convert_with_links() (*in module ironic.api.controllers.v1.port*), 2249

list_convert_with_links() (*in module ironic.api.controllers.v1.portgroup*), 2258

list_convert_with_links() (*in module ironic.api.controllers.v1.volume_connector*), 2360

list_convert_with_links() (*in module ironic.api.controllers.v1.volume_target*), 2373

list_hardware_type_interfaces() (*ironic.db.api.Connection method*), 3157

list_hardware_type_interfaces() (*ironic.db.sqlalchemy.api.Connection method*), 3014

list_opts() (*in module ironic.conf.cinder*), 2920

list_opts() (*in module ironic.conf.default*), 2921

list_opts() (*in module ironic.conf.glance*), 2921

list_opts() (*in module ironic.conf.inspector*), 2922

list_opts() (*in module ironic.conf.neutron*), 2923

list_opts() (*in module ironic.conf.nova*), 2923

list_opts() (*in module ironic.conf.opts*), 2923

list_opts() (*in module ironic.conf.service_catalog*), 2925

list_opts() (*in module ironic.conf.swift*), 2925

list_physical_disks() (*in module ironic.drivers.modules.drac.raid*), 3308

list_policies() (*in module ironic.common.policy*), 2603

list_raid_controllers() (*in module ironic.drivers.modules.drac.raid*), 3309

list_raid_settings() (*in module ironic.drivers.modules.drac.raid*), 3310

list_unfinished_jobs() (*in module ironic.drivers.modules.drac.job*), 3256

list_unfinished_jobs() (*ironic.drivers.modules.drac.vendor_passthru.DracWSMa method*), 3323

list_virtual_disks() (*in module ironic.drivers.modules.drac.raid*), 3311

ListOfFlexibleDictsField (*class in ironic.objects.fields*), 4418

ListOfObjectsField (*class in ironic.objects.fields*), 4418

ListOfStringsField (*class in*

ironic.objects.fields), 4418
load_driver() (*ironic.conductor.task_manager.TaskManager method*), 2870
local() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3558
local_link_connection (*ironic.db.sqlalchemy.models.Port attribute*), 3067
local_link_connection() (*ironic.objects.port.Port property*), 4490
local_link_connection() (*ironic.objects.port.PortCRUDPayload property*), 4494
local_link_normalize() (*in module ironic.api.controllers.v1.utils*), 2332
log_and_raise_deployment_error() (*in module ironic.drivers.modules.agent_base*), 3805
logical_disk_properties() (*ironic.api.controllers.v1.driver.DriverRaidController method*), 2177
login_iscsi() (*in module ironic.drivers.modules.iscsi_deploy*), 4082
logout_iscsi() (*in module ironic.drivers.modules.iscsi_deploy*), 4082
LOOKUP_ALLOWED_STATES (*in module ironic.common.states*), 2655
lookup_allowed_states() (*ironic.api.controllers.v1.ramdisk.LookupController property*), 2264
LookupController (*class in ironic.api.controllers.v1.ramdisk*), 2261

M

mac_address() (*in module ironic.common.args*), 2413
MACAddress (*class in ironic.objects.fields*), 4418
MACAddressField (*class in ironic.objects.fields*), 4420
MACAlreadyExists, 2482
main() (*in module ironic.cmd.api*), 2397
main() (*in module ironic.cmd.conductor*), 2397
main() (*in module ironic.cmd.dbsync*), 2399
main() (*in module ironic.cmd.status*), 2400
maintenance (*ironic.api.controllers.v1.node.NodesController attribute*), 2228
maintenance (*ironic.db.sqlalchemy.models.Node attribute*), 3064
maintenance() (*ironic.objects.node.Node property*), 4444
maintenance() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
maintenance() (*ironic.objects.node.NodeCRUDPayload property*), 4453
maintenance() (*ironic.objects.node.NodePayload property*), 4461
maintenance() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
maintenance() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
maintenance_reason (*ironic.db.sqlalchemy.models.Node attribute*), 3064
maintenance_reason() (*ironic.objects.node.Node property*), 4445
maintenance_reason() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
maintenance_reason() (*ironic.objects.node.NodeCRUDPayload property*), 4453
maintenance_reason() (*ironic.objects.node.NodePayload property*), 4461
maintenance_reason() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
maintenance_reason() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
major() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3559
make_link() (*in module ironic.api.controllers.link*), 2376
make_persistent_password_file() (*in module ironic.drivers.modules.console_utils*), 3860
make_salt() (*in module*

ironic.conductor.utils), 2893
MANAGEABLE (in module ironic.common.states), 2655
management (ironic.api.controllers.v1.node.NodesController attribute), 2228
management (ironic.drivers.base.BareDriver attribute), 4173
management_interface (ironic.db.sqlalchemy.models.Node attribute), 3064
management_interface() (ironic.objects.node.Node property), 4445
management_interface() (ironic.objects.node.NodeCorrectedPowerStatePayload property), 4458
management_interface() (ironic.objects.node.NodeCRUDPayload property), 4453
management_interface() (ironic.objects.node.NodePayload property), 4462
management_interface() (ironic.objects.node.NodeSetPowerStatePayload property), 4465
management_interface() (ironic.objects.node.NodeSetProvisionStatePayload property), 4469
ManagementInterface (class in ironic.drivers.base), 4209
mandatory (ironic.api.functions.FunctionArgument attribute), 2383
ManualManagementHardware (class in ironic.drivers.generic), 4309
mask_secrets() (in module ironic.objects.notification), 4475
max_string (ironic.api.controllers.base.Version attribute), 2374
max_version() (in module ironic.objects.base), 4364
max_version_string() (in module ironic.api.controllers.v1.versions), 2346
memoize() (in module ironic.drivers.modules.snmp), 4163
metadata (ironic.db.sqlalchemy.models.IronicBase attribute), 3062
metadata (ironic.drivers.base.VendorMetadata attribute), 4289
METAREGION (ironic.common.image_format_inspector.VHDXInspector attribute), 2518
method (ironic.drivers.base.VendorMetadata attribute), 4289
methods() (ironic.api.controllers.v1.driver.DriverPassthruContr method), 2175
methods() (ironic.api.controllers.v1.node.NodeVendorPassthruC method), 2213
METRICS (in module ironic.drivers.modules.irmc.inspect), 3531
METRICS (in module ironic.drivers.modules.irmc.power), 3574
micro() (ironic.drivers.modules.irmc.packaging_version.Version property), 3559
migrate_from_iscsi_deploy() (ironic.db.api.Connection method), 3158
migrate_from_iscsi_deploy() (ironic.db.sqlalchemy.api.Connection method), 3014
min_string (ironic.api.controllers.base.Version attribute), 2374
min_version_string() (in module ironic.api.controllers.v1.versions), 2347
minor() (ironic.drivers.modules.irmc.packaging_version.Version property), 3560
MissingParameterValue, 2482
MixinVendorInterface (class in ironic.drivers.utils), 4327
mode (ironic.db.sqlalchemy.models.Portgroup attribute), 3069
mode() (ironic.objects.portgroup.Portgroup property), 4506
mode() (ironic.objects.portgroup.PortgroupCRUDPayload property), 4510
model_query() (in module ironic.db.sqlalchemy.api), 3052
module
    ironic, 4552
    ironic.api, 2397
    ironic.api.app, 2382
    ironic.api.config, 2383
    ironic.api.controllers, 2378
    ironic.api.controllers.base, 2374
    ironic.api.controllers.link, 2376
    ironic.api.controllers.root, 2376
    ironic.api.controllers.v1, 2373

ironic.api.controllers.v1.allocation, 2148
ironic.api.controllers.v1.bios, 2155
ironic.api.controllers.v1.chassis, 2156
ironic.api.controllers.v1.collection, 2162
ironic.api.controllers.v1.conductor, 2165
ironic.api.controllers.v1.deploy_template, 2168
ironic.api.controllers.v1.driver, 2175
ironic.api.controllers.v1.event, 2183
ironic.api.controllers.v1.node, 2184
ironic.api.controllers.v1.notification_utils, 2237
ironic.api.controllers.v1.port, 2240
ironic.api.controllers.v1.portgroup, 2251
ironic.api.controllers.v1.ramdisk, 2258
ironic.api.controllers.v1.utils, 2264
ironic.api.controllers.v1.versions, 2346
ironic.api.controllers.v1.volume, 2347
ironic.api.controllers.v1.volume_connector, 2347
ironic.api.controllers.v1.volume_target, 2360
ironic.api.controllers.version, 2376
ironic.api.functions, 2383
ironic.api.hooks, 2389
ironic.api.method, 2396
ironic.api.middleware, 2380
ironic.api.middleware.auth_public_routes, 2378
ironic.api.middleware.json_ext, 2379
ironic.api.middleware.parsable_error, 2380
ironic.api.wsgi, 2397
ironic.cmd, 2400
ironic.cmd.api, 2397
ironic.cmd.conductor, 2397
ironic.cmd.dbsync, 2398
ironic.cmd.status, 2399
ironic.common, 2697
ironic.common.args, 2408
ironic.common.boot_devices, 2424
ironic.common.boot_modes, 2426
ironic.common.checksum_utils, 2427
ironic.common.cinder, 2433
ironic.common.components, 2446
ironic.common.config, 2447
ironic.common.context, 2447
ironic.common.dhcp_factory, 2449
ironic.common.driver_factory, 2452
ironic.common.exception, 2467
ironic.common.faults, 2492
ironic.common.fsm, 2493
ironic.common.glance_service, 2408
ironic.common.glance_service.image_service
2400
ironic.common.glance_service.service_utils
2406
ironic.common.hash_ring, 2498
ironic.common.i18n, 2498
ironic.common.image_format_inspector, 2499
ironic.common.image_service, 2524
ironic.common.images, 2537
ironic.common.indicator_states, 2556
ironic.common.keystone, 2557
ironic.common.molds, 2564
ironic.common.network, 2567
ironic.common.neutron, 2573
ironic.common.nova, 2598
ironic.common.policy, 2599
ironic.common.profiler, 2603
ironic.common.pxe_utils, 2605
ironic.common.qemu_img, 2632
ironic.common.raid, 2632
ironic.common.release_mappings, 2639
ironic.common.rpc, 2641
ironic.common.rpc_service, 2645
ironic.common.service, 2646
ironic.common.states, 2646
ironic.common.swift, 2662

ironic.common.utils, 2670
ironic.common.wsgi_service, 2696
ironic.conductor, 2918
ironic.conductor.allocations, 2697
ironic.conductor.base_manager, 2702
ironic.conductor.cleaning, 2707
ironic.conductor.deployments, 2711
ironic.conductor.manager, 2717
ironic.conductor.notification_utils, 2734
ironic.conductor.rpcapi, 2739
ironic.conductor.steps, 2857
ironic.conductor.task_manager, 2862
ironic.conductor.utils, 2878
ironic.conf, 2926
ironic.conf.agent, 2918
ironic.conf.anaconda, 2918
ironic.conf.ansible, 2919
ironic.conf.api, 2919
ironic.conf.audit, 2919
ironic.conf.auth, 2919
ironic.conf.cinder, 2920
ironic.conf.conductor, 2920
ironic.conf.console, 2920
ironic.conf.database, 2921
ironic.conf.default, 2921
ironic.conf.deploy, 2921
ironic.conf.dhcp, 2921
ironic.conf.disk_utils, 2921
ironic.conf.drac, 2921
ironic.conf.glance, 2921
ironic.conf.healthcheck, 2922
ironic.conf.ibmc, 2922
ironic.conf.ilo, 2922
ironic.conf.inspector, 2922
ironic.conf.ipmi, 2922
ironic.conf.irmc, 2922
ironic.conf.iscsi, 2922
ironic.conf.metrics, 2923
ironic.conf.metrics_statsd, 2923
ironic.conf.molds, 2923
ironic.conf.neutron, 2923
ironic.conf.nova, 2923
ironic.conf.opts, 2923
ironic.conf.pxe, 2925
ironic.conf.redfish, 2925
ironic.conf.service_catalog, 2925
ironic.conf.snmp, 2925
ironic.conf.swift, 2925
ironic.conf.xclarity, 2926
ironic.db, 3193
ironic.db.api, 3072
ironic.db.migration, 3193
ironic.db.sqlalchemy, 3072
ironic.db.sqlalchemy.api, 2926
ironic.db.sqlalchemy.migration, 3053
ironic.db.sqlalchemy.models, 3056
ironic.dhcp, 3214
ironic.dhcp.base, 3194
ironic.dhcp.neutron, 3201
ironic.dhcp.none, 3207
ironic.drivers, 4341
ironic.drivers.base, 4163
ironic.drivers.drac, 4301
ironic.drivers.fake_hardware, 4303
ironic.drivers.generic, 4307
ironic.drivers.hardware_type, 4310
ironic.drivers.ibmc, 4315
ironic.drivers.ilo, 4316
ironic.drivers.intel_ipmi, 4320
ironic.drivers.ipmi, 4321
ironic.drivers.irmc, 4322
ironic.drivers.modules, 4163
ironic.drivers.modules.agent, 3750
ironic.drivers.modules.agent_base, 3774
ironic.drivers.modules.agent_client, 3810
ironic.drivers.modules.agent_power, 3844
ironic.drivers.modules.ansible, 3223
ironic.drivers.modules.ansible.deploy, 3214
ironic.drivers.modules.boot_mode_utils, 3851
ironic.drivers.modules.console_utils, 3858
ironic.drivers.modules.deploy_utils, 3865
ironic.drivers.modules.drac, 3327

ironic.drivers.modules.drac.bios, 3223
ironic.drivers.modules.drac.boot, 3243
ironic.drivers.modules.drac.common, 3247
ironic.drivers.modules.drac.inspect, 3249
ironic.drivers.modules.drac.job, 3255
ironic.drivers.modules.drac.management, 3258
ironic.drivers.modules.drac.power, 3278
ironic.drivers.modules.drac.raid, 3286
ironic.drivers.modules.drac.utils, 3314
ironic.drivers.modules.drac.vendor_passthru, 3316
ironic.drivers.modules.fake, 3908
ironic.drivers.modules.ibmc, 3367
ironic.drivers.modules.ibmc.management, 3327
ironic.drivers.modules.ibmc.mappings, 3343
ironic.drivers.modules.ibmc.power, 3343
ironic.drivers.modules.ibmc.raid, 3352
ironic.drivers.modules.ibmc.utils, 3359
ironic.drivers.modules.ibmc.vendor, 3361
ironic.drivers.modules.ilo, 3500
ironic.drivers.modules.ilo.bios, 3367
ironic.drivers.modules.ilo.boot, 3372
ironic.drivers.modules.ilo.common, 3404
ironic.drivers.modules.ilo.console, 3441
ironic.drivers.modules.ilo.firmware_processor, 3443
ironic.drivers.modules.ilo.inspect, 3448
ironic.drivers.modules.ilo.management, 3451
ironic.drivers.modules.ilo.power, 3482
ironic.drivers.modules.ilo.raid, 3489
ironic.drivers.modules.ilo.vendor, 3495
ironic.drivers.modules.image_cache, 3989
ironic.drivers.modules.image_utils, 3994
ironic.drivers.modules.inspect_utils, 4010
ironic.drivers.modules.inspector, 4012
ironic.drivers.modules.intel_ipmi, 3501
ironic.drivers.modules.intel_ipmi.manageme
3500
ironic.drivers.modules.ipmitool, 4015
ironic.drivers.modules.ipxe, 4055
ironic.drivers.modules.irmc, 3578
ironic.drivers.modules.irmc.bios, 3501
ironic.drivers.modules.irmc.boot, 3506
ironic.drivers.modules.irmc.common, 3521
ironic.drivers.modules.irmc.inspect, 3528
ironic.drivers.modules.irmc.management, 3533
ironic.drivers.modules.irmc.packaging_vers
3549
ironic.drivers.modules.irmc.power, 3566
ironic.drivers.modules.irmc.raid, 3574
ironic.drivers.modules.iscsi_deploy, 4056
ironic.drivers.modules.network, 3622
ironic.drivers.modules.network.common, 3578
ironic.drivers.modules.network.flat, 3594
ironic.drivers.modules.network.neutron, 3603
ironic.drivers.modules.network.noop,

3613
`ironic.drivers.modules.noop`, 4084
`ironic.drivers.modules.noop_mgmt`, 4105
`ironic.drivers.modules.pxe`, 4118
`ironic.drivers.modules.pxe_base`, 4131
`ironic.drivers.modules.redfish`, 3721
`ironic.drivers.modules.redfish.bios`, 3622
`ironic.drivers.modules.redfish.boot`, 3630
`ironic.drivers.modules.redfish.inspect`, 3643
`ironic.drivers.modules.redfish.management`, 3647
`ironic.drivers.modules.redfish.power`, 3683
`ironic.drivers.modules.redfish.raid`, 3691
`ironic.drivers.modules.redfish.utils`, 3711
`ironic.drivers.modules.redfish.vendor`, 3718
`ironic.drivers.modules.snmp`, 4140
`ironic.drivers.modules.storage`, 3736
`ironic.drivers.modules.storage.cinder`, 3721
`ironic.drivers.modules.storage.external`, 3726
`ironic.drivers.modules.storage.noop`, 3731
`ironic.drivers.modules.xclarity`, 3750
`ironic.drivers.modules.xclarity.common`, 3736
`ironic.drivers.modules.xclarity.management`, 3741
`ironic.drivers.modules.xclarity.power`, 3746
`ironic.drivers.redfish`, 4325
`ironic.drivers.snmp`, 4326
`ironic.drivers.utils`, 4327
`ironic.drivers.xclarity`, 4340
`ironic.objects`, 4552
`ironic.objects.allocation`, 4341
`ironic.objects.base`, 4353
`ironic.objects.bios`, 4365
`ironic.objects.chassis`, 4378
`ironic.objects.conductor`, 4387
`ironic.objects.deploy_template`, 4395
`ironic.objects.deployment`, 4408
`ironic.objects.fields`, 4416
`ironic.objects.indirection`, 4424
`ironic.objects.node`, 4434
`ironic.objects.notification`, 4470
`ironic.objects.port`, 4476
`ironic.objects.portgroup`, 4495
`ironic.objects.trait`, 4510
`ironic.objects.volume_connector`, 4519
`ironic.objects.volume_target`, 4534
`ironic.version`, 4552
`mount()` (*in module ironic.common.utils*), 2683

N

`name` (*ironic.api.functions.FunctionArgument attribute*), 2384
`name` (*ironic.api.functions.FunctionDefinition attribute*), 2386
`name` (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057
`name` (*ironic.db.sqlalchemy.models.BIOSSetting attribute*), 3058
`name` (*ironic.db.sqlalchemy.models.DeployTemplate attribute*), 3061
`name` (*ironic.db.sqlalchemy.models.Node attribute*), 3064
`name` (*ironic.db.sqlalchemy.models.Port attribute*), 3067
`name` (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3069
`name()` (*in module ironic.common.args*), 2415
`name()` (*ironic.objects.allocation.Allocation property*), 4349
`name()` (*ironic.objects.allocation.AllocationCRUDPayload property*), 4353
`name()` (*ironic.objects.bios.BIOSSetting property*), 4369
`name()` (*ironic.objects.deploy_template.DeployTemplate property*), 4404
`name()` (*ironic.objects.deploy_template.DeployTemplateCRUDPa property*), 4408
`name()` (*ironic.objects.node.Node property*), 4445

name() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
name() (*ironic.objects.node.NodeCRUDPayload property*), 4453
name() (*ironic.objects.node.NodePayload property*), 4462
name() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
name() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
name() (*ironic.objects.port.Port property*), 4490
name() (*ironic.objects.port.PortCRUDPayload property*), 4494
name() (*ironic.objects.portgroup.Portgroup property*), 4506
name() (*ironic.objects.portgroup.PortgroupCRUDPayload property*), 4510
names() (*ironic.common.driver_factory.BaseDriverFactory property*), 2453
need_power_on() (*ironic.drivers.base.NetworkInterface method*), 4251
need_power_on() (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3607
need_prepare_ramdisk() (*in module ironic.drivers.utils*), 4336
network (*ironic.drivers.base.BareDriver attribute*), 4173
network_data (*ironic.db.sqlalchemy.models.Node attribute*), 3064
network_data() (*ironic.objects.node.Node property*), 4445
network_data_schema() (*in module ironic.api.controllers.v1.node*), 2232
network_interface (*ironic.db.sqlalchemy.models.Node attribute*), 3064
network_interface() (*ironic.objects.node.Node property*), 4445
network_interface() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
network_interface() (*ironic.objects.node.NodeCRUDPayload property*), 4453
network_interface() (*ironic.objects.node.NodePayload property*), 4462
network_interface() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
network_interface() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
NetworkError, 2482
NetworkInterface (*class in ironic.drivers.base*), 4243
NetworkInterfaceFactory (*class in ironic.common.driver_factory*), 2453
NeutronDHCPApi (*class in ironic.dhcp.neutron*), 3201
NeutronNetwork (*class in ironic.drivers.modules.network.neutron*), 3603
NeutronNetworkInterfaceMixin (*class in ironic.common.neutron*), 2573
NeutronVIFPortIDMixin (*class in ironic.drivers.modules.network.common*), 3578
new_region() (*ironic.common.image_format_inspector.FileInsp method*), 2505
NIC (*in module ironic.common.components*), 2446
NoBIOS (*class in ironic.drivers.modules.noop*), 4085
NoConsole (*class in ironic.drivers.modules.noop*), 4091
NoConsolePid, 2482
Node (*class in ironic.db.sqlalchemy.models*), 3062
Node (*class in ironic.objects.node*), 4434
node (*ironic.db.sqlalchemy.models.NodeTag attribute*), 3066
node (*ironic.db.sqlalchemy.models.NodeTrait attribute*), 3066
node() (*ironic.conductor.task_manager.TaskManager property*), 2870
node_cache_bios_settings() (*in module ironic.conductor.utils*), 2893
node_cache_vendor() (*in module ironic.conductor.utils*), 2893
NODE_CLASS_OID (*in module ironic.drivers.modules.irmc.inspect*), 3732
node_convert_with_links() (*in module ironic.api.controllers.v1.node*), 2232
node_get_boot_mode() (*in module ironic.conductor.utils*), 2894
node_id (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057
node_id (*ironic.db.sqlalchemy.models.BIOSSetting attribute*), 3058

node_id (*ironic.db.sqlalchemy.models.NodeTag attribute*), 3066

node_id (*ironic.db.sqlalchemy.models.NodeTrait attribute*), 3066

node_id (*ironic.db.sqlalchemy.models.Port attribute*), 3067

node_id (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3069

node_id (*ironic.db.sqlalchemy.models.VolumeConnector attribute*), 3070

node_id (*ironic.db.sqlalchemy.models.VolumeTarget attribute*), 3070

node_id() (*ironic.objects.allocation.Allocation property*), 4349

node_id() (*ironic.objects.bios.BIOSSetting property*), 4369

node_id() (*ironic.objects.port.Port property*), 4490

node_id() (*ironic.objects.portgroup.Portgroup property*), 4506

node_id() (*ironic.objects.trait.Trait property*), 4514

node_id() (*ironic.objects.volume_connector.VolumeConnector property*), 4530

node_id() (*ironic.objects.volume_target.VolumeTarget property*), 4547

node_list_convert_with_links() (*in module ironic.api.controllers.v1.node*), 2232

node_mapping (*ironic.objects.deployment.Deployment attribute*), 4414

node_patch_schema() (*in module ironic.api.controllers.v1.node*), 2232

node_patch_validator() (*in module ironic.api.controllers.v1.node*), 2232

node_power_action() (*in module ironic.conductor.utils*), 2895

node_sanitize() (*in module ironic.api.controllers.v1.node*), 2232

node_schema() (*in module ironic.api.controllers.v1.node*), 2235

node_set_boot_device() (*in module ironic.conductor.utils*), 2897

node_set_boot_mode() (*in module ironic.conductor.utils*), 2899

node_states_convert() (*in module ironic.api.controllers.v1.node*), 2235

node_tag_exists() (*ironic.db.api.Connection method*), 3159

node_tag_exists() (*ironic.db.sqlalchemy.api.Connection method*), 3015

node_trait_exists() (*ironic.db.api.Connection method*), 3160

node_trait_exists() (*ironic.db.sqlalchemy.api.Connection method*), 3016

node_uuid() (*ironic.objects.allocation.AllocationCRUDPayloa property*), 4353

node_uuid() (*ironic.objects.deployment.Deployment property*), 4414

node_uuid() (*ironic.objects.port.PortCRUDPayload property*), 4494

node_uuid() (*ironic.objects.portgroup.PortgroupCRUDPayloa property*), 4510

node_uuid() (*ironic.objects.volume_connector.VolumeConnecto property*), 4534

node_uuid() (*ironic.objects.volume_target.VolumeTargetCRUD property*), 4552

node_validator() (*in module ironic.api.controllers.v1.node*), 2235

node_wait_for_power_state() (*in module ironic.conductor.utils*), 2901

NodeAllocationController (*class in ironic.api.controllers.v1.allocation*), 2154

NodeAlreadyExists, 2483

NodeAssociated, 2484

NodeBiosController (*class in ironic.api.controllers.v1.bios*), 2155

NodeCleaningFailure, 2484

NodeConsoleController (*class in ironic.api.controllers.v1.node*), 2193

NodeConsoleNotEnabled, 2484

NodeConsoleNotification (*class in ironic.objects.node*), 4454

NodeCorrectedPowerStateNotification (*class in ironic.objects.node*), 4455

NodeCorrectedPowerStatePayload (*class in ironic.objects.node*), 4456

NodeCRUDNotification (*class in ironic.objects.node*), 4450

NodeCRUDPayload (*class in ironic.objects.node*), 4451

NodeInMaintenance, 2484

NodeIsRetired, 2484

NodeLocked, 2484

NodeMaintenanceController (*class in ironic.api.controllers.v1.node*), 2195

NodeMaintenanceFailure, 2484

NodeMaintenanceNotification (*class in ironic.objects.node*), 4459

NodeManagementController (*class in ironic.api.controllers.v1.node*), 2196

NodeNotFound, 2484

NodeNotLocked, 2485

NodePayload (*class in ironic.objects.node*), 4460

NodeProtected, 2485

nodes (*ironic.api.controllers.v1.chassis.ChassisController attribute*), 2160

NodesController (*class in ironic.api.controllers.v1.node*), 2214

NodeSetPowerStateNotification (*class in ironic.objects.node*), 4463

NodeSetPowerStatePayload (*class in ironic.objects.node*), 4464

NodeSetProvisionStateNotification (*class in ironic.objects.node*), 4466

NodeSetProvisionStatePayload (*class in ironic.objects.node*), 4467

NodeStatesController (*class in ironic.api.controllers.v1.node*), 2197

NodeTag (*class in ironic.db.sqlalchemy.models*), 3065

NodeTagNotFound, 2485

NodeTrait (*class in ironic.db.sqlalchemy.models*), 3066

NodeTraitNotFound, 2485

NodeTraitsController (*class in ironic.api.controllers.v1.node*), 2210

NodeVendorPassthruController (*class in ironic.api.controllers.v1.node*), 2213

NodeVIFController (*class in ironic.api.controllers.v1.node*), 2212

NoDriversLoaded, 2482

NoExceptionTracebackHook (*class in ironic.api.hooks*), 2392

NoFreeConductorWorker, 2483

NoFreeIPMITerminalPorts, 2483

NoFreePhysicalPorts, 2483

NoInspect (*class in ironic.drivers.modules.noop*), 4094

non_vendor_interfaces() (*ironic.drivers.base.BareDriver property*), 4173

NoneDHCPApi (*class in ironic.dhcp.none*), 3207

NoopManagement (*class in ironic.drivers.modules.noop_mgmt*), 4105

NoopNetwork (*class in ironic.drivers.modules.network.noop*), 3613

NoopStorage (*class in ironic.drivers.modules.storage.noop*), 3731

NoRAID (*class in ironic.drivers.modules.noop*), 4095

NoRescue (*class in ironic.drivers.modules.noop*), 4100

normalize_mac() (*in module ironic.drivers.utils*), 4337

NOSTATE (*in module ironic.common.states*), 2656

NotAcceptable, 2485

NotAuthorized, 2485

NotFound, 2486

NotificationBase (*class in ironic.objects.notification*), 4472

NotificationLevel (*class in ironic.objects.fields*), 4420

NotificationLevelField (*class in ironic.objects.fields*), 4420

NotificationPayloadBase (*class in ironic.objects.notification*), 4473

NotificationPayloadError, 2486

NotificationPublisher (*class in ironic.objects.notification*), 4475

NotificationSchemaKeyError, 2486

NotificationSchemaObjectError, 2486

NotificationStatus (*class in ironic.objects.fields*), 4420

NotificationStatusField (*class in ironic.objects.fields*), 4421

notify_conductor_resume_clean() (*in module ironic.conductor.utils*), 2902

notify_conductor_resume_deploy() (*in module ironic.conductor.utils*), 2902

notify_conductor_resume_operation() (*in module ironic.conductor.utils*), 2902

NoValidDefaultForInterface, 2483

NoValidHost, 2483

NoVendor (*class in ironic.drivers.modules.noop*), 4103

O

obj (*ironic.api.controllers.v1.utils.PassthruResponse attribute*), 2264

OBJ_BASE_CLASS (*ironic.objects.base.IronicObjectSerializer attribute*), 4362

OBJ_PROJECT_NAMESPACE (*ironic.objects.base.IronicObject at-*

tribute), 4354
obj_refresh() (*ironic.objects.base.IronicObject method*), 4360
OBJ_SERIAL_NAMESPACE (*ironic.objects.base.IronicObject attribute*), 4354
object() (*ironic.objects.notification.EventType property*), 4471
object_action() (*ironic.conductor.manager.ConductorManager method*), 2724
object_action() (*ironic.conductor.rpcapi.ConductorAPI method*), 2817
object_action() (*ironic.objects.indirection.IronicObjectIndirectionAPI method*), 4424
object_backport_versions() (*ironic.conductor.manager.ConductorManager method*), 2726
object_backport_versions() (*ironic.conductor.rpcapi.ConductorAPI method*), 2819
object_backport_versions() (*ironic.objects.indirection.IronicObjectIndirectionAPI method*), 4426
object_class_action() (*ironic.objects.indirection.IronicObjectIndirectionAPI method*), 4428
object_class_action_versions() (*ironic.conductor.manager.ConductorManager method*), 2728
object_class_action_versions() (*ironic.conductor.rpcapi.ConductorAPI method*), 2822
object_class_action_versions() (*ironic.objects.indirection.IronicObjectIndirectionAPI method*), 4430
object_to_dict() (*in module ironic.api.controllers.v1.utils*), 2332
ObjectField (*class in ironic.objects.fields*), 4421
objects() (*ironic.objects.bios.BIOSSettingList property*), 4375
objects() (*ironic.objects.trait.TraitList property*), 4518
OFF (*in module ironic.common.indicator_states*), 2557
oid_device (*ironic.drivers.modules.snmp.SNMPDriverAPCMasterSwitch attribute*), 4144
oid_device (*ironic.drivers.modules.snmp.SNMPDriverAPCMas attribute*), 4145
oid_device (*ironic.drivers.modules.snmp.SNMPDriverAPCRac attribute*), 4146
oid_device (*ironic.drivers.modules.snmp.SNMPDriverAten attribute*), 4147
oid_device (*ironic.drivers.modules.snmp.SNMPDriverBaytech attribute*), 4151
oid_device (*ironic.drivers.modules.snmp.SNMPDriverCyberPo attribute*), 4152
oid_device (*ironic.drivers.modules.snmp.SNMPDriverEatonPo attribute*), 4153
oid_device (*ironic.drivers.modules.snmp.SNMPDriverTeltronix attribute*), 4156
oid_device() (*ironic.drivers.modules.snmp.SNMPDriverSimpl property*), 4154
oid_enterprise (*ironic.drivers.modules.snmp.SNMPDriverBase attribute*), 4148
oid_poweroff (*ironic.drivers.modules.snmp.SNMPDriverEaton attribute*), 4153
oid_poweron (*ironic.drivers.modules.snmp.SNMPDriverEaton attribute*), 4153
oid_status (*ironic.drivers.modules.snmp.SNMPDriverEatonPo attribute*), 4153
ON (*in module ironic.common.indicator_states*), 2557
on_enter() (*in module ironic.common.states*), 2661
on_exit() (*in module ironic.common.states*), 2662
one_button_secure_erase() (*ironic.drivers.modules.ilo.management.Ilo5Management method*), 3453
online (*ironic.db.sqlalchemy.models.Conductor attribute*), 3059
online_data_migrations ironic-dbsync command line option, 1919
online_data_migrations command line option
    --help, 1920
    --max-count <NUMBER>, 1920
    --option <MIGRATION.KEY=VALUE>, 1920
    -h, 1920
online_data_migrations() (*ironic.cmd.dbsync.DBCommand method*), 2399
OperationNotPermitted, 2486
optional_interfaces()

(*ironic.drivers.base.BareDriver property*), 4174

`or_valid()` (*in module ironic.common.args*), 2416

`override_api_url()` (*in module ironic.drivers.modules.image_utils*), 3999

`owner` (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057

`owner` (*ironic.db.sqlalchemy.models.Node attribute*), 3064

`owner()` (*ironic.objects.allocation.Allocation property*), 4349

`owner()` (*ironic.objects.allocation.AllocationCRUDPayload property*), 4353

`owner()` (*ironic.objects.node.Node property*), 4445

`owner()` (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458

`owner()` (*ironic.objects.node.NodeCRUDPayload property*), 4453

`owner()` (*ironic.objects.node.NodePayload property*), 4462

`owner()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465

`owner()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469

P

`ParsableErrorMiddleware` (*class in ironic.api.middleware*), 2382

`ParsableErrorMiddleware` (*class in ironic.api.middleware.parsable_error*), 2380

`parse()` (*in module ironic.drivers.modules.irmc.packaging_version*), 3565

`parse_args()` (*in module ironic.common.config*), 2447

`parse_driver_info()` (*in module ironic.common.pxe_utils*), 2626

`parse_driver_info()` (*in module ironic.drivers.modules.drac.common*), 3248

`parse_driver_info()` (*in module ironic.drivers.modules.ibmc.utils*), 3359

`parse_driver_info()` (*in module ironic.drivers.modules.ilo.boot*), 3401

`parse_driver_info()` (*in module ironic.drivers.modules.ilo.common*), 3423

`parse_driver_info()` (*in module ironic.drivers.modules.irmc.common*), 3525

`parse_driver_info()` (*in module ironic.drivers.modules.redfish.utils*), 3716

`parse_driver_info()` (*in module ironic.drivers.modules.xclarity.common*), 3738

`parse_headers()` (*ironic.api.controllers.base.Version static method*), 2374

`parse_image_id()` (*in module ironic.common.glance_service.service_utils*), 2407

`parse_instance_info()` (*in module ironic.drivers.modules.deploy_utils*), 3890

`parse_instance_info_capabilities()` (*in module ironic.common.utils*), 2684

`passthru()` (*in module ironic.drivers.base*), 4301

`PassthruResponse` (*class in ironic.api.controllers.v1.utils*), 2264

`PasswordFileFailedToCreate`, 2486

`patch()` (*in module ironic.common.args*), 2417

`patch()` (*ironic.api.controllers.v1.allocation.AllocationsControll method*), 2152

`patch()` (*ironic.api.controllers.v1.chassis.ChassisController method*), 2160

`patch()` (*ironic.api.controllers.v1.deploy_template.DeployTempl method*), 2172

`patch()` (*ironic.api.controllers.v1.node.NodesController method*), 2228

`patch()` (*ironic.api.controllers.v1.port.PortsController method*), 2248

`patch()` (*ironic.api.controllers.v1.portgroup.PortgroupsControlle method*), 2256

`patch()` (*ironic.api.controllers.v1.volume_connector.VolumeCon method*), 2355

`patch()` (*ironic.api.controllers.v1.volume_target.VolumeTargetsC method*), 2368

`patch_update_changed_fields()` (*in module ironic.api.controllers.v1.utils*), 2334

`patch_validate_allowed_fields()` (*in module ironic.api.controllers.v1.utils*), 2336

`patched_validate_with_schema()` (*in module ironic.api.controllers.v1.utils*),

2337
PatchError, 2486
PathNotFound, 2486
payload() (*ironic.objects.allocation.AllocationCRUDNotification property*), 4352
payload() (*ironic.objects.chassis.ChassisCRUDNotification property*), 4386
payload() (*ironic.objects.deploy_template.DeployTemplateCRUDNotification property*), 4407
payload() (*ironic.objects.node.NodeConsoleNotification property*), 4455
payload() (*ironic.objects.node.NodeCorrectedPowerStateNotification property*), 4456
payload() (*ironic.objects.node.NodeCRUDNotification property*), 4451
payload() (*ironic.objects.node.NodeMaintenanceNotification property*), 4460
payload() (*ironic.objects.node.NodeSetPowerStateNotification property*), 4463
payload() (*ironic.objects.node.NodeSetProvisionStateNotification property*), 4467
payload() (*ironic.objects.port.PortCRUDNotification property*), 4494
payload() (*ironic.objects.portgroup.PortgroupCRUDNotification property*), 4509
payload() (*ironic.objects.volume_connector.VolumeConnectorCRUDNotification property*), 4533
payload() (*ironic.objects.volume_target.VolumeTargetCRUDNotification property*), 4551
physical_network (*ironic.db.sqlalchemy.models.Port attribute*), 3068
physical_network() (*ironic.objects.port.Port property*), 4490
physical_network() (*ironic.objects.port.PortCRUDPayload property*), 4494
PHYSNET_PARAM_NAME (*in module ironic.common.neutron*), 2574
PlaybookNotFound, 3222
plug_port_to_tenant_network() (*in module ironic.drivers.modules.network.common*), 3593
policy_deprecation_check() (*in module ironic.api.hooks*), 2395
pop_node_nested_field() (*in module ironic.common.utils*), 2686
populate_node_uuid() (*in module ironic.api.controllers.v1.utils*), 2339
populate_schema() (*ironic.objects.notification.NotificationPayloadBase method*), 4474
populate_storage_driver_internal_info() (*in module ironic.drivers.modules.deploy_utils*), 3891
Port (*class in ironic.db.sqlalchemy.models*), 3067
Port (*class in ironic.objects.port*), 4476
port_changed() (*ironic.drivers.base.NetworkInterface method*), 4252
port_changed() (*ironic.drivers.modules.network.common.NeutronVIFPort method*), 3581
port_changed() (*ironic.drivers.modules.network.noop.NoopNetwork method*), 3616
port_sanitize() (*in module ironic.api.controllers.v1.port*), 2250
PortAlreadyExists, 2487
PortCRUDNotification (*class in ironic.objects.port*), 4493
PortCRUDPayload (*class in ironic.objects.port*), 4494
PortgroupAddressConflict, 2487
Portgroup (*class in ironic.db.sqlalchemy.models*), 3068
Portgroup (*class in ironic.objects.portgroup*), 4495
portgroup_changed() (*ironic.drivers.base.NetworkInterface method*), 4252
portgroup_changed() (*ironic.drivers.modules.network.common.NeutronVIFPort method*), 3581
portgroup_changed() (*ironic.drivers.modules.network.noop.NoopNetwork method*), 3617
portgroup_id (*ironic.db.sqlalchemy.models.Port attribute*), 3068
portgroup_id() (*ironic.objects.port.Port property*), 4490
portgroup_uuid() (*ironic.objects.port.PortCRUDPayload property*), 4494
PortgroupAlreadyExists, 2487
PortgroupCRUDNotification (*class in ironic.objects.portgroup*), 4508
PortgroupCRUDPayload (*class in ironic.objects.portgroup*), 4509

PortgroupDuplicateName, 2487
PortgroupMACAlreadyExists, 2487
PortgroupNotEmpty, 2487
PortgroupNotFound, 2488
PortgroupPhysnetInconsistent, 2488
portgroups() (*ironic.conductor.task_manager.TaskManager property*), 2870
PortgroupsController (*class in ironic.api.controllers.v1.portgroup*), 2251
PortNotFound, 2487
ports() (*ironic.conductor.task_manager.TaskManager property*), 2870
PortsController (*class in ironic.api.controllers.v1.port*), 2240
post() (*ironic.api.controllers.v1.allocation.AllocationsController method*), 2153
post() (*ironic.api.controllers.v1.chassis.ChassisController method*), 2161
post() (*ironic.api.controllers.v1.deploy_template.DeployTemplatesController method*), 2173
post() (*ironic.api.controllers.v1.event.EventsController method*), 2184
post() (*ironic.api.controllers.v1.node.NodesController method*), 2229
post() (*ironic.api.controllers.v1.node.NodeVIFController method*), 2212
post() (*ironic.api.controllers.v1.port.PortsController method*), 2249
post() (*ironic.api.controllers.v1.portgroup.PortgroupsController method*), 2257
post() (*ironic.api.controllers.v1.ramdisk.HeartbeatController method*), 2258
post() (*ironic.api.controllers.v1.volume_connector.VolumeConnectorsController method*), 2358
post() (*ironic.api.controllers.v1.volume_target.VolumeTargetsController method*), 2371
post() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3560
post_clean_step_hook() (*in module ironic.drivers.modules.agent_base*), 3806
post_configuration() (*ironic.drivers.modules.redfish.bios.RedfishBIOS method*), 3627
post_create_configuration() (*ironic.drivers.modules.redfish.raid.RedfishRAID method*), 3697
post_delete_configuration() (*ironic.drivers.modules.redfish.raid.RedfishRAID method*), 3699
post_deploy_step_hook() (*in module ironic.drivers.modules.agent_base*), 3808
POST_FINISHEDPOST_STATE (*in module ironic.drivers.modules.ilo.common*), 3404
POST_INPOST_STATE (*in module ironic.drivers.modules.ilo.common*), 3405
POST_INPOSTDISCOVERY_STATE (*in module ironic.drivers.modules.ilo.common*), 3405
POST_NULL_STATE (*in module ironic.drivers.modules.ilo.common*), 3405
POST_POWEROFF_STATE (*in module ironic.drivers.modules.ilo.common*), 3405
post_process() (*ironic.common.image_format_inspector.FileInspector method*), 2505
post_process() (*ironic.common.image_format_inspector.VHDXInspector method*), 2518
post_process() (*ironic.common.image_format_inspector.VMDKInspector method*), 2520
post_reset() (*ironic.drivers.modules.redfish.bios.RedfishBIOS method*), 3628
POST_RESET_STATE (*in module ironic.drivers.modules.ilo.common*), 3405
POST_UNKNOWN_STATE (*in module ironic.drivers.modules.ilo.common*), 3405
POWER (*in module ironic.common.components*), 2447
power (*ironic.drivers.base.BareDriver attribute*), 4174
power() (*ironic.api.controllers.v1.node.NodeStatesController method*), 2198
POWER_FAILURE (*in module ironic.common.faults*), 2492
power_interface (*ironic.db.sqlalchemy.models.Node attribute*), 3064
power_interface() (*ironic.objects.node.Node property*), 4445
power_interface() (*ironic.objects.node.NodeCorrectedPowerStatePayload*

property), 4458
power_interface() (*ironic.objects.node.NodeCRUDPayload property*), 4453
power_interface() (*ironic.objects.node.NodePayload property*), 4462
power_interface() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
power_interface() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
POWER_OFF (*in module ironic.common.states*), 2656
power_off() (*ironic.drivers.modules.agent_client.AgentClient method*), 3834
power_off() (*ironic.drivers.modules.snmp.SNMPDriverBase method*), 4148
POWER_ON (*in module ironic.common.states*), 2656
power_on() (*ironic.drivers.modules.snmp.SNMPDriverBase method*), 4148
power_on_node_if_needed() (*in module ironic.conductor.utils*), 2903
power_reset() (*ironic.drivers.modules.snmp.SNMPDriverBase method*), 4149
power_state (*ironic.db.sqlalchemy.models.Node attribute*), 3064
power_state() (*ironic.drivers.modules.snmp.SNMPDriverBase method*), 4149
power_state() (*ironic.objects.node.Node property*), 4445
power_state() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
power_state() (*ironic.objects.node.NodeCRUDPayload property*), 4453
power_state() (*ironic.objects.node.NodePayload property*), 4462
power_state() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
power_state() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
power_state_error_handler() (*in module ironic.conductor.utils*), 2904
power_state_for_network_configuration() (*in module ironic.conductor.utils*), 2905
power_update() (*in module ironic.common.nova*), 2598
PowerInterface (*class in ironic.drivers.base*), 4262
PowerStateFailure, 2488
pre() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3561
pre_create_configuration() (*ironic.drivers.modules.redfish.raid.RedfishRAID method*), 3700
pre_delete_configuration() (*ironic.drivers.modules.redfish.raid.RedfishRAID method*), 3701
prepare() (*ironic.drivers.base.DeployInterface method*), 4200
prepare() (*ironic.drivers.modules.agent.AgentDeploy method*), 3754
prepare() (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3217
prepare() (*ironic.drivers.modules.fake.FakeDeploy method*), 3931
prepare() (*ironic.drivers.modules.iscsi_deploy.ISCSIDeploy method*), 4059
prepare() (*ironic.drivers.modules.pxe.PXEAnacondaDeploy method*), 4122
prepare() (*ironic.drivers.modules.pxe.PXERamdiskDeploy method*), 4128
prepare_agent_boot() (*in module ironic.drivers.modules.deploy_utils*), 3892
prepare_boot_iso() (*in module ironic.drivers.modules.image_utils*), 3999
prepare_cleaning() (*ironic.drivers.base.DeployInterface method*), 4201
prepare_cleaning() (*ironic.drivers.modules.agent_base.AgentBaseMixin method*), 3775
prepare_cleaning() (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3218
prepare_configdrive_image() (*in module ironic.drivers.modules.image_utils*), 4001
prepare_deploy_iso() (*in module ironic.drivers.modules.image_utils*), 4003

prepare_disk_image() (*in module ironic.drivers.modules.image_utils*), 4006

prepare_floppy_image() (*in module ironic.drivers.modules.image_utils*), 4008

prepare_host() (*ironic.conductor.base_manager.BaseConductorManager method*), 2706

prepare_image() (*ironic.drivers.modules.agent_client.AgentClient method*), 3835

prepare_inband_cleaning() (*in module ironic.drivers.modules.deploy_utils*), 3893

prepare_instance() (*ironic.drivers.base.BootInterface method*), 4188

prepare_instance() (*ironic.drivers.modules.fake.FakeBoot method*), 3919

prepare_instance() (*ironic.drivers.modules.ilo.boot.IloiPXEBoot method*), 3398

prepare_instance() (*ironic.drivers.modules.ilo.boot.IloPXEBoot method*), 3374

prepare_instance() (*ironic.drivers.modules.ilo.boot.IloUefiHttpsBoot method*), 3379

prepare_instance() (*ironic.drivers.modules.ilo.boot.IloVirtualMediaBoot method*), 3389

prepare_instance() (*ironic.drivers.modules.irmc.boot.IRMCVirtualMediaBoot method*), 3511

prepare_instance() (*ironic.drivers.modules.pxe_base.PXEBaseMixin method*), 4134

prepare_instance() (*ironic.drivers.modules.redfish.boot.RedfishVirtualMediaBoot method*), 3635

prepare_instance_boot() (*ironic.drivers.modules.agent.AgentDeployMixin method*), 3758

prepare_instance_boot() (*ironic.drivers.modules.iscsi_deploy.ISCSIDeploy method*), 4062

prepare_instance_kickstart_config() (*in module ironic.common.pxe_utils*), 2627

prepare_instance_pxe_config() (*in module ironic.common.pxe_utils*), 2629

prepare_instance_to_boot() (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3788

prepare_node_for_deploy() (*in module ironic.drivers.modules.ilo.boot*), 3403

prepare_ramdisk() (*ironic.drivers.base.BootInterface method*), 4188

prepare_ramdisk() (*ironic.drivers.modules.fake.FakeBoot method*), 3919

prepare_ramdisk() (*ironic.drivers.modules.ilo.boot.IloiPXEBoot method*), 3399

prepare_ramdisk() (*ironic.drivers.modules.ilo.boot.IloPXEBoot method*), 3375

prepare_ramdisk() (*ironic.drivers.modules.ilo.boot.IloUefiHttpsBoot method*), 3381

prepare_ramdisk() (*ironic.drivers.modules.ilo.boot.IloVirtualMediaBoot method*), 3392

prepare_ramdisk() (*ironic.drivers.modules.irmc.boot.IRMCPXEBoot method*), 3506

prepare_ramdisk() (*ironic.drivers.modules.irmc.boot.IRMCVirtualMediaBoo method*), 3512

prepare_ramdisk() (*ironic.drivers.modules.pxe_base.PXEBaseMixin method*), 4134

prepare_ramdisk() (*ironic.drivers.modules.redfish.boot.RedfishVirtualMediaE method*), 3637

prepare_service() (*in module ironic.common.service*), 2646

previous_provision_state() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469

previous_target_provision_state() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469

priority (*ironic.db.sqlalchemy.models.DeployTemplateStep attribute*), 3062

process_event() (*ironic.common.fsm.FSM method*), 2497

process_event() (*ironic.conductor.task_manager.TaskManager*

method), 2870
process_fw_on() (*ironic.drivers.modules.ilo.firmware_processor.FirmwareProcessor method*), 3444
process_launcher() (*in module ironic.common.service*), 2646
process_next_step() (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3789
process_next_step() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3799
process_next_step() (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3218
properties (*ironic.db.sqlalchemy.models.Node attribute*), 3064
properties (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3069
properties (*ironic.db.sqlalchemy.models.VolumeTarget attribute*), 3070
properties() (*ironic.api.controllers.v1.driver.DriversController method*), 2179
properties() (*ironic.objects.node.Node property*), 4445
properties() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
properties() (*ironic.objects.node.NodeCRUDPayload property*), 4453
properties() (*ironic.objects.node.NodePayload property*), 4462
properties() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
properties() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
properties() (*ironic.objects.portgroup.Portgroup property*), 4506
properties() (*ironic.objects.portgroup.PortgroupCRUDPayload property*), 4510
properties() (*ironic.objects.volume_target.VolumeTarget property*), 4547
properties() (*ironic.objects.volume_target.VolumeTargetCRUDPayload property*), 4552
protected (*ironic.db.sqlalchemy.models.Node attribute*), 3064
protected() (*ironic.objects.node.Node property*), 4445
protected() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
protected() (*ironic.objects.node.NodeCRUDPayload property*), 4453
protected() (*ironic.objects.node.NodePayload property*), 4462
protected() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
protected() (*ironic.objects.node.NodeSetProvisionStatePayloa property*), 4469
protected_reason (*ironic.db.sqlalchemy.models.Node attribute*), 3064
protected_reason() (*ironic.objects.node.Node property*), 4445
protected_reason() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
protected_reason() (*ironic.objects.node.NodeCRUDPayload property*), 4453
protected_reason() (*ironic.objects.node.NodePayload property*), 4462
protected_reason() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
protected_reason() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
provider() (*ironic.common.dhcp_factory.DHCPFactory property*), 2450
provision() (*ironic.api.controllers.v1.node.NodeStatesControl method*), 2200
provision_state (*ironic.db.sqlalchemy.models.Node attribute*), 3064
provision_state() (*ironic.objects.node.Node property*), 4445
provision_state() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
provision_state() (*ironic.objects.node.NodeCRUDPayload property*), 4453
provision_state() (*ironic.objects.node.NodePayload property*), 4462
provision_state() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4465
provision_state() (*ironic.objects.node.NodeSetProvisionStatePayload*

property), 4469
provision_updated_at (*ironic.db.sqlalchemy.models.Node attribute*), 3065
provision_updated_at() (*ironic.objects.node.Node property*), 4445
provision_updated_at() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
provision_updated_at() (*ironic.objects.node.NodeCRUDPayload property*), 4453
provision_updated_at() (*ironic.objects.node.NodePayload property*), 4462
provision_updated_at() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466
provision_updated_at() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4469
provisioning_error_handler() (*in module ironic.conductor.utils*), 2906
public() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3562
PublicUrlHook (*class in ironic.api.hooks*), 2393
publish_image() (*ironic.drivers.modules.image_utils.ImageHandler method*), 3994
publisher() (*ironic.objects.allocation.AllocationCRUDNotification property*), 4352
publisher() (*ironic.objects.chassis.ChassisCRUDNotification property*), 4386
publisher() (*ironic.objects.deploy_template.DeployTemplateCRUDNotification property*), 4407
publisher() (*ironic.objects.node.NodeConsoleNotification property*), 4455
publisher() (*ironic.objects.node.NodeCorrectedPowerStateNotification property*), 4456
publisher() (*ironic.objects.node.NodeCRUDNotification property*), 4451
publisher() (*ironic.objects.node.NodeMaintenanceNotification property*), 4460
publisher() (*ironic.objects.node.NodeSetPowerStateNotification property*), 4463
publisher() (*ironic.objects.node.NodeSetProvisionStateNotification property*), 4467
publisher() (*ironic.objects.notification.NotificationBase property*), 4473
publisher() (*ironic.objects.port.PortCRUDNotification property*), 4494
publisher() (*ironic.objects.portgroup.PortgroupCRUDNotifica property*), 4509
publisher() (*ironic.objects.volume_connector.VolumeConnecto property*), 4533
publisher() (*ironic.objects.volume_target.VolumeTargetCRUD property*), 4551
put() (*ironic.api.controllers.v1.node.BootDeviceController method*), 2185
put() (*ironic.api.controllers.v1.node.IndicatorController method*), 2189
put() (*ironic.api.controllers.v1.node.InjectNmiController method*), 2191
put() (*ironic.api.controllers.v1.node.NodeConsoleController method*), 2194
put() (*ironic.api.controllers.v1.node.NodeMaintenanceControlle method*), 2195
put() (*ironic.api.controllers.v1.node.NodeTraitsController method*), 2211
PXE (*in module ironic.common.boot_devices*), 2426
pxe_enabled (*ironic.db.sqlalchemy.models.Port attribute*), 3068
pxe_enabled() (*ironic.objects.port.Port property*), 4490
pxe_enabled() (*ironic.objects.port.PortCRUDPayload property*), 4495
PXEAnacondaDeploy (*class in ironic.drivers.modules.pxe*), 4118
PXEBaseMixin (*class in ironic.drivers.modules.pxe_base*), 4131
PXEBoot (*class in ironic.drivers.modules.pxe*), 4126
PXERamdiskDeploy (*class in ironic.drivers.modules.pxe*), 4126

Q

QcowInspector (*class in ironic.common.image_format_inspector*), 2512
QEDInspector (*class in ironic.common.image_format_inspector*), 2511

R

raid (*ironic.api.controllers.v1.driver.DriversController attribute*), 2180
raid (*ironic.drivers.base.BareDriver attribute*), 4174

raid() (*ironic.api.controllers.v1.node.NodeStatesController method*), 2208
RAID_APPLY_CONFIGURATION_ARGSINFO (*in module ironic.drivers.base*), 4278
RAID_APPLY_CONFIGURATION_ARGSINFO (*ironic.drivers.modules.ibmc.raid.IbmcRAID attribute*), 3352
raid_config (*ironic.db.sqlalchemy.models.Node attribute*), 3065
raid_config() (*ironic.objects.node.Node property*), 4445
raid_interface (*ironic.db.sqlalchemy.models.Node attribute*), 3065
raid_interface() (*ironic.objects.node.Node property*), 4445
raid_interface() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4458
raid_interface() (*ironic.objects.node.NodeCRUDPayload property*), 4453
raid_interface() (*ironic.objects.node.NodePayload property*), 4462
raid_interface() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466
raid_interface() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470
RAIDInterface (*class in ironic.drivers.base*), 4268
ramdisk_ref() (*ironic.objects.deployment.Deployment property*), 4414
read() (*ironic.common.image_format_inspector.InfoWrapper method*), 2511
REBOOT (*in module ironic.common.states*), 2656
reboot() (*ironic.drivers.base.PowerInterface method*), 4265
reboot() (*ironic.drivers.modules.agent_client.AgentClient method*), 3837
reboot() (*ironic.drivers.modules.agent_power.AgentPower method*), 3847
reboot() (*ironic.drivers.modules.drac.power.DracWSManPower method*), 3281
reboot() (*ironic.drivers.modules.fake.FakePower method*), 3965
reboot() (*ironic.drivers.modules.ibmc.power.IBMCPower method*), 3346
reboot() (*ironic.drivers.modules.ilo.power.IloPower method*), 3484
reboot() (*ironic.drivers.modules.ipmitool.IPMIPower method*), 4032
reboot() (*ironic.drivers.modules.irmc.power.IRMCPower method*), 3569
reboot() (*ironic.drivers.modules.redfish.power.RedfishPower method*), 3686
reboot() (*ironic.drivers.modules.snmp.SNMPPower method*), 4158
reboot() (*ironic.drivers.modules.xclarity.power.XClarityPower method*), 3747
reboot_and_finish_deploy() (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3790
reboot_to_finish_step() (*in module ironic.drivers.modules.deploy_utils*), 3895
reboot_to_instance() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3800
reboot_to_instance() (*ironic.drivers.modules.pxe.PXEAnacondaDeploy method*), 4123
REBUILD (*in module ironic.common.states*), 2656
RedfishBIOS (*class in ironic.drivers.modules.redfish.bios*), 3622
RedfishConnectionError, 2488
RedfishError, 2488
RedfishHardware (*class in ironic.drivers.redfish*), 4325
RedfishInspect (*class in ironic.drivers.modules.redfish.inspect*), 3643
RedfishManagement (*class in ironic.drivers.modules.redfish.management*), 3647
RedfishPower (*class in ironic.drivers.modules.redfish.power*), 3683
RedfishRAID (*class in ironic.drivers.modules.redfish.raid*), 3691
RedfishVendorPassthru (*class in ironic.drivers.modules.redfish.vendor*), 3718
RedfishVirtualMediaBoot (*class in ironic.drivers.modules.redfish.boot*), 3630
refresh() (*ironic.objects.allocation.Allocation

method), 4349

refresh() (*ironic.objects.chassis.Chassis method*), 4384

refresh() (*ironic.objects.conductor.Conductor method*), 4390

refresh() (*ironic.objects.deploy_template.DeployTemplate method*), 4404

refresh() (*ironic.objects.deployment.Deployment method*), 4415

refresh() (*ironic.objects.node.Node method*), 4445

refresh() (*ironic.objects.port.Port method*), 4490

refresh() (*ironic.objects.portgroup.Portgroup method*), 4506

refresh() (*ironic.objects.volume_connector.VolumeConnector method*), 4530

refresh() (*ironic.objects.volume_target.VolumeTarget method*), 4547

refresh_clean_steps() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3800

refresh_steps() (*ironic.drivers.modules.agent_base.AgentBaseMixin method*), 3776

refresh_steps() (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3791

refresh_steps() (*ironic.drivers.modules.agent_base.HeartbeatMixin method*), 3800

region() (*ironic.common.image_format_inspector.FileInspector method*), 2506

register() (*ironic.objects.conductor.Conductor class method*), 4391

register_all() (*in module ironic.objects*), 4552

register_auth_opts() (*in module ironic.conf.auth*), 2919

register_conductor() (*ironic.db.api.Connection method*), 3161

register_conductor() (*ironic.db.sqlalchemy.api.Connection method*), 3017

register_conductor_hardware_interfaces() (*ironic.db.api.Connection method*), 3163

register_conductor_hardware_interfaces() (*ironic.db.sqlalchemy.api.Connection method*), 3019

register_hardware_interfaces() (*ironic.objects.conductor.Conductor method*), 4393

register_opts() (*in module ironic.conf.agent*), 2918

register_opts() (*in module ironic.conf.anaconda*), 2918

register_opts() (*in module ironic.conf.ansible*), 2919

register_opts() (*in module ironic.conf.api*), 2919

register_opts() (*in module ironic.conf.audit*), 2919

register_opts() (*in module ironic.conf.cinder*), 2920

register_opts() (*in module ironic.conf.conductor*), 2920

register_opts() (*in module ironic.conf.console*), 2920

register_opts() (*in module ironic.conf.database*), 2921

register_opts() (*in module ironic.conf.default*), 2921

register_opts() (*in module ironic.conf.deploy*), 2921

register_opts() (*in module ironic.conf.dhcp*), 2921

register_opts() (*in module ironic.conf.disk_utils*), 2921

register_opts() (*in module ironic.conf.drac*), 2921

register_opts() (*in module ironic.conf.glance*), 2921

register_opts() (*in module ironic.conf.healthcheck*), 2922

register_opts() (*in module ironic.conf.ibmc*), 2922

register_opts() (*in module ironic.conf.ilo*), 2922

register_opts() (*in module ironic.conf.inspector*), 2922

register_opts() (*in module ironic.conf.ipmi*), 2922

register_opts() (*in module ironic.conf.irmc*), 2922

register_opts() (*in module ironic.conf.iscsi*), 2922

register_opts() (*in module ironic.conf.metrics*), 2923

register_opts() (*in module ironic.conf.metrics_statsd*), 2923

register_opts() (*in module*

ironic.conf.molds), 2923
register_opts() (*in module ironic.conf.neutron*), 2923
register_opts() (*in module ironic.conf.nova*), 2923
register_opts() (*in module ironic.conf.pxe*), 2925
register_opts() (*in module ironic.conf.redfish*), 2925
register_opts() (*in module ironic.conf.service_catalog*), 2925
register_opts() (*in module ironic.conf.snmp*), 2925
register_opts() (*in module ironic.conf.swift*), 2925
register_opts() (*in module ironic.conf.xclarity*), 2926
registration_hook() (*ironic.objects.base.IronicObjectRegistry method*), 4362
reject_fields_in_newer_versions() (*in module ironic.api.controllers.v1.node*), 2235
reject_patch_in_newer_versions() (*in module ironic.api.controllers.v1.node*), 2235
release() (*ironic.drivers.modules.irmc.packaging_version.Version property*), 3563
release() (*ironic.objects.node.Node class method*), 4446
release_node() (*ironic.db.api.Connection method*), 3165
release_node() (*ironic.db.sqlalchemy.api.Connection method*), 3021
release_port() (*in module ironic.drivers.modules.console_utils*), 3860
release_resources() (*ironic.conductor.task_manager.TaskManager method*), 2872
remove() (*ironic.drivers.modules.ilo.firmware_processor.FirmwareImageLocation method*), 3443
remove_agent_url() (*in module ironic.conductor.utils*), 2908
remove_cleaning_network() (*ironic.drivers.base.NetworkInterface method*), 4253
remove_cleaning_network() (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3599
remove_cleaning_network() (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3608
remove_cleaning_network() (*ironic.drivers.modules.network.noop.NoopNetwork method*), 3617
remove_http_instance_symlink() (*in module ironic.drivers.modules.deploy_utils*), 3895
remove_image_from_swift() (*in module ironic.drivers.modules.ilo.common*), 3424
remove_image_from_web_server() (*in module ironic.drivers.modules.ilo.common*), 3425
remove_inspection_network() (*ironic.drivers.base.NetworkInterface method*), 4254
remove_inspection_network() (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3599
remove_inspection_network() (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3608
remove_large_keys() (*in module ironic.common.utils*), 2687
remove_neutron_ports() (*in module ironic.common.neutron*), 2586
remove_node_rescue_password() (*in module ironic.conductor.utils*), 2908
remove_node_traits() (*ironic.conductor.manager.ConductorManager method*), 2730
remove_node_traits() (*ironic.conductor.rpcapi.ConductorAPI method*), 2824
remove_ports_from_network() (*in module ironic.common.neutron*), 2586
remove_provisioning_network() (*ironic.drivers.base.NetworkInterface method*), 4255
remove_provisioning_network() (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3600
remove_provisioning_network() (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3609
remove_provisioning_network() (*ironic.drivers.modules.network.noop.NoopNetwork*

method), 3618
`remove_rescuing_network()` (*ironic.drivers.base.NetworkInterface method*), 4255
`remove_rescuing_network()` (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3601
`remove_rescuing_network()` (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3610
`remove_single_or_list_of_files()` (*in module ironic.drivers.modules.ilo.common*), 3426
`remove_vifs_from_node()` (*in module ironic.common.network*), 2572
`render_template()` (*in module ironic.common.utils*), 2687
`replace_node_id_with_uuid()` (*in module ironic.api.controllers.v1.utils*), 2341
`replace_node_uuid_with_id()` (*in module ironic.api.controllers.v1.utils*), 2342
`RequestContext` (*class in ironic.common.context*), 2447
`RequestContextSerializer` (*class in ironic.common.rpc*), 2641
`require_exclusive_lock()` (*in module ironic.conductor.task_manager*), 2877
`RESCUE` (*in module ironic.common.states*), 2657
`rescue` (*ironic.drivers.base.BareDriver attribute*), 4175
`rescue()` (*ironic.drivers.base.RescueInterface method*), 4280
`rescue()` (*ironic.drivers.modules.agent.AgentRescue method*), 3767
`rescue()` (*ironic.drivers.modules.fake.FakeRescue method*), 3974
`rescue()` (*ironic.drivers.modules.noop.NoRescue method*), 4101
`RESCUE_ABORT_FAILURE` (*in module ironic.common.faults*), 2493
`rescue_interface` (*ironic.db.sqlalchemy.models.Node attribute*), 3065
`rescue_interface()` (*ironic.objects.node.Node property*), 4447
`rescue_interface()` (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459
`rescue_interface()` (*ironic.objects.node.NodeCRUDPayload property*), 4453
`rescue_interface()` (*ironic.objects.node.NodePayload property*), 4462
`rescue_interface()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466
`rescue_interface()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470
`rescue_or_deploy_mode()` (*in module ironic.drivers.modules.deploy_utils*), 3895
`RESCUEFAIL` (*in module ironic.common.states*), 2657
`RescueInterface` (*class in ironic.drivers.base*), 4278
`RESCUEWAIT` (*in module ironic.common.states*), 2657
`RESCUING` (*in module ironic.common.states*), 2658
`rescuing_error_handler()` (*in module ironic.conductor.utils*), 2909
`reservation` (*ironic.db.sqlalchemy.models.Node attribute*), 3065
`reservation()` (*ironic.objects.node.Node property*), 4447
`reserve()` (*ironic.objects.node.Node class method*), 4447
`reserve_node()` (*ironic.db.api.Connection method*), 3167
`reserve_node()` (*ironic.db.sqlalchemy.api.Connection method*), 3023
`reset()` (*ironic.common.hash_ring.HashRingManager class method*), 2498
`reset()` (*ironic.common.wsgi_service.WSGIService method*), 2696
`reset_bios_to_default()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3468
`reset_idrac()` (*ironic.drivers.modules.drac.management.DracRedfishMa method*), 3266
`reset_idrac()` (*ironic.drivers.modules.drac.management.DracWSManM method*), 3273
`reset_ilo()` (*ironic.drivers.modules.ilo.management.IloManag method*), 3469
`reset_ilo_credential()`

(*ironic.drivers.modules.ilo.management.IloManagement method*), 3470

`reset_secure_boot_keys_to_default()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3471

`reset_secure_boot_keys_to_default()` (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3670

`resolve_type()` (*ironic.api.functions.FunctionArgument method*), 2384

`resolve_types()` (*ironic.api.functions.FunctionDefinition method*), 2386

`resource_class` (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057

`resource_class` (*ironic.db.sqlalchemy.models.Node attribute*), 3065

`resource_class()` (*ironic.objects.allocation.Allocation property*), 4350

`resource_class()` (*ironic.objects.allocation.AllocationCRUDPayload property*), 4353

`resource_class()` (*ironic.objects.node.Node property*), 4448

`resource_class()` (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459

`resource_class()` (*ironic.objects.node.NodeCRUDPayload property*), 4454

`resource_class()` (*ironic.objects.node.NodePayload property*), 4462

`resource_class()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466

`resource_class()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470

`restore_boot_device()` (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3671

`restore_irmc_bios_config()` (*ironic.drivers.modules.irmc.management.IRMCManagement method*), 3543

`restore_power_state_if_needed()` (*in module ironic.conductor.utils*), 2910

`retired` (*ironic.db.sqlalchemy.models.Node attribute*), 3065

`retired()` (*ironic.objects.node.Node property*), 4448

`retired()` (*ironic.objects.node.NodeCorrectedPowerStatePaylo property*), 4459

`retired()` (*ironic.objects.node.NodeCRUDPayload property*), 4454

`retired()` (*ironic.objects.node.NodePayload property*), 4462

`retired()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466

`retired()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470

`retired_reason` (*ironic.db.sqlalchemy.models.Node attribute*), 3065

`retired_reason()` (*ironic.objects.node.Node property*), 4448

`retired_reason()` (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459

`retired_reason()` (*ironic.objects.node.NodeCRUDPayload property*), 4454

`retired_reason()` (*ironic.objects.node.NodePayload property*), 4462

`retired_reason()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466

`retired_reason()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470

`retry_interval` (*ironic.drivers.modules.snmp.SNMPDriverBase attribute*), 4150

`retry_on_outdated_cache()` (*in module ironic.drivers.modules.snmp*), 4163

`return_type` (*ironic.api.functions.FunctionDefinition attribute*), 2386

`revert_dictionary()` (*in module ironic.drivers.modules.ibmc.utils*), 3360

revision
- ironic-dbsync command line option, 1919

revision ironic-dbsync command line option
- --autogenerate, 1921
- --help, 1921
- --message <MESSAGE>, 1921

-h, 1921
-m <MESSAGE>, 1921
revision() (*in module ironic.db.migration*), 3193
revision() (*in module ironic.db.sqlalchemy.migration*), 3054
revision() (*ironic.cmd.dbsync.DBCommand method*), 2399
ring() (*ironic.common.hash_ring.HashRingManager property*), 2498
rmtree_without_raise() (*in module ironic.common.utils*), 2689
rollback_ports() (*in module ironic.common.neutron*), 2587
root() (*in module ironic.api.controllers.root*), 2376
root_device() (*ironic.objects.deployment.Deployment property*), 4415
root_gib() (*ironic.objects.deployment.Deployment property*), 4415
RootController (*class in ironic.api.controllers.root*), 2376
RPC_API_VERSION (*ironic.conductor.manager.ConductorManager attribute*), 2718
RPC_API_VERSION (*ironic.conductor.rpcapi.ConductorAPI attribute*), 2751
RPCHook (*class in ironic.api.hooks*), 2394
RPCService (*class in ironic.common.rpc_service*), 2645

S

SAFE (*in module ironic.common.boot_devices*), 2426
safe_rstrip() (*in module ironic.common.utils*), 2689
safety_check() (*ironic.common.image_format_inspector.FileInspector method*), 2506
safety_check() (*ironic.common.image_format_inspector.QcowInspector method*), 2513
safety_check() (*ironic.common.image_format_inspector.QEDInspector method*), 2511
safety_check() (*ironic.common.image_format_inspector.VMDKInspector method*), 2521
safety_check_image() (*in module ironic.common.images*), 2554
sanitize_dict() (*in module ironic.api.controllers.v1.utils*), 2343
SAS (*in module ironic.common.raid*), 2632
SATA (*in module ironic.common.raid*), 2632
save() (*ironic.objects.allocation.Allocation method*), 4350
save() (*ironic.objects.bios.BIOSSetting method*), 4369
save() (*ironic.objects.bios.BIOSSettingList class method*), 4375
save() (*ironic.objects.chassis.Chassis method*), 4384
save() (*ironic.objects.conductor.Conductor method*), 4394
save() (*ironic.objects.deploy_template.DeployTemplate method*), 4405
save() (*ironic.objects.node.Node method*), 4449
save() (*ironic.objects.port.Port method*), 4491
save() (*ironic.objects.portgroup.Portgroup method*), 4507
save() (*ironic.objects.volume_connector.VolumeConnector method*), 4531
save() (*ironic.objects.volume_target.VolumeTarget method*), 4549
save_configuration() (*in module ironic.common.molds*), 2565
SCHEMA (*ironic.objects.allocation.AllocationCRUDPayload attribute*), 4352
SCHEMA (*ironic.objects.chassis.ChassisCRUDPayload attribute*), 4386
SCHEMA (*ironic.objects.deploy_template.DeployTemplateCRUDPa attribute*), 4407
SCHEMA (*ironic.objects.node.NodeCRUDPayload attribute*), 4452
SCHEMA (*ironic.objects.node.NodePayload attribute*), 4460
SCHEMA (*ironic.objects.node.NodeSetProvisionStatePayload attribute*), 4468
SCHEMA (*ironic.objects.notification.NotificationPayloadBase attribute*), 4474
SCHEMA (*ironic.objects.port.PortCRUDPayload attribute*), 4494
SCHEMA (*ironic.objects.portgroup.PortgroupCRUDPayload attribute*), 4509
SCHEMA (*ironic.objects.volume_connector.VolumeConnectorCRUI attribute*), 4534
SCHEMA (*ironic.objects.volume_target.VolumeTargetCRUDPayloa attribute*), 4551
schema() (*in module ironic.common.args*), 2417
SCSI (*in module ironic.common.raid*), 2633

second_method() (*ironic.drivers.modules.fake.FakeVendorB method*), 3987
send_raw() (*in module ironic.drivers.modules.ipmitool*), 4052
send_raw() (*ironic.drivers.modules.ipmitool.VendorPassthru method*), 4045
serialize_context() (*ironic.common.rpc.RequestContextSerializer method*), 2642
serialize_entity() (*ironic.common.rpc.RequestContextSerializer method*), 2643
serialize_entity() (*ironic.objects.base.IronicObjectSerializer method*), 4362
service() (*ironic.objects.notification.NotificationPublisher property*), 4475
ServiceUnavailable, 2488
SessionCache (*class in ironic.drivers.modules.redfish.utils*), 3711
set() (*ironic.drivers.modules.snmp.SNMPClient method*), 4142
set_arg_types() (*ironic.api.functions.FunctionDefinition method*), 2386
set_async_step_flags() (*in module ironic.drivers.modules.deploy_utils*), 3895
set_bios_config() (*ironic.drivers.modules.drac.vendor_passthru.DracWSManVendorPassthru method*), 3324
set_boot_device() (*in module ironic.drivers.modules.drac.management*), 3276
set_boot_device() (*ironic.conductor.manager.ConductorManager method*), 2730
set_boot_device() (*ironic.conductor.rpcapi.ConductorAPI method*), 2826
set_boot_device() (*ironic.drivers.base.ManagementInterface method*), 4235
set_boot_device() (*ironic.drivers.modules.drac.management.DracWSManManagement method*), 3273
set_boot_device() (*ironic.drivers.modules.fake.FakeManagement method*), 3958
set_boot_device() (*ironic.drivers.modules.ibmc.management.IBMCManage method*), 3337
set_boot_device() (*ironic.drivers.modules.ilo.management.IloManagement method*), 3472
set_boot_device() (*ironic.drivers.modules.ipmitool.IPMIManagement method*), 4025
set_boot_device() (*ironic.drivers.modules.irmc.management.IRMCManage method*), 3543
set_boot_device() (*ironic.drivers.modules.noop_mgmt.NoopManagement method*), 4115
set_boot_device() (*ironic.drivers.modules.redfish.management.RedfishMana method*), 3672
set_boot_device() (*ironic.drivers.modules.xclarity.management.XClarityMa method*), 3744
set_boot_mode() (*in module ironic.drivers.modules.ilo.common*), 3427
set_boot_mode() (*ironic.drivers.base.ManagementInterface method*), 4237
set_boot_mode() (*ironic.drivers.modules.ibmc.management.IBMCManage method*), 3339
set_boot_mode() (*ironic.drivers.modules.ilo.management.IloManagement method*), 3474
set_boot_mode() (*ironic.drivers.modules.redfish.management.RedfishMana method*), 3675
set_config() (*in module ironic.drivers.modules.drac.bios*), 3241
set_console_mode() (*ironic.conductor.manager.ConductorManager method*), 2730
set_console_mode() (*ironic.conductor.rpcapi.ConductorAPI method*), 2829
set_defaults() (*in module ironic.common.rpc*), 2644
set_failed_state() (*in module ironic.drivers.modules.deploy_utils*), 3897
set_indicator_state() (*ironic.conductor.manager.ConductorManager*

method), 2730
`set_indicator_state()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2831
`set_indicator_state()` (*ironic.drivers.base.ManagementInterface method*), 4240
`set_indicator_state()` (*ironic.drivers.modules.redfish.management.RedfishManagement method*), 3677
`set_iscsi_boot_target()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3476
`set_node_cleaning_steps()` (*in module ironic.conductor.steps*), 2857
`set_node_deployment_steps()` (*in module ironic.conductor.steps*), 2859
`set_node_nested_field()` (*in module ironic.common.utils*), 2689
`set_node_tags()` (*ironic.db.api.Connection method*), 3168
`set_node_tags()` (*ironic.db.sqlalchemy.api.Connection method*), 3024
`set_node_traits()` (*ironic.db.api.Connection method*), 3169
`set_node_traits()` (*ironic.db.sqlalchemy.api.Connection method*), 3026
`set_options()` (*ironic.api.functions.FunctionDefinition method*), 2386
`set_power_state()` (*ironic.drivers.base.PowerInterface method*), 4266
`set_power_state()` (*ironic.drivers.modules.agent_power.AgentPower method*), 3848
`set_power_state()` (*ironic.drivers.modules.drac.power.DracWSManPower method*), 3283
`set_power_state()` (*ironic.drivers.modules.fake.FakePower method*), 3966
`set_power_state()` (*ironic.drivers.modules.ibmc.power.IBMCPower method*), 3348
`set_power_state()` (*ironic.drivers.modules.ilo.power.IloPower method*), 3486
`set_power_state()` (*ironic.drivers.modules.ipmitool.IPMIPower method*), 4033
`set_power_state()` (*ironic.drivers.modules.irmc.power.IRMCPower method*), 3571
`set_power_state()` (*ironic.drivers.modules.redfish.power.RedfishPower method*), 3688
`set_power_state()` (*ironic.drivers.modules.snmp.SNMPPower method*), 4160
`set_power_state()` (*ironic.drivers.modules.xclarity.power.XClarityPower method*), 3748
`set_raid_settings()` (*in module ironic.drivers.modules.drac.raid*), 3312
`set_secure_boot_mode()` (*in module ironic.drivers.modules.ilo.common*), 3428
`set_secure_boot_mode()` (*in module ironic.drivers.modules.irmc.common*), 3527
`set_secure_boot_state()` (*ironic.drivers.base.ManagementInterface method*), 4242
`set_secure_boot_state()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3476
`set_secure_boot_state()` (*ironic.drivers.modules.irmc.management.IRMCManagen method*), 3545
`set_secure_boot_state()` (*ironic.drivers.modules.redfish.management.RedfishMana method*), 3679
`set_spawn_error_hook()` (*ironic.conductor.task_manager.TaskManager method*), 2873
`set_target_raid_config()` (*ironic.conductor.manager.ConductorManager method*), 2730
`set_target_raid_config()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2833
`setup()` (*in module ironic.common.profiler*), 2603
`setup_app()` (*in module ironic.api.app*), 2383
`setup_uefi_https()` (*in module ironic.drivers.modules.ilo.common*), 3429
`setup_vmedia()` (*in module*

ironic.drivers.modules.ilo.common), 3432
setup_vmedia_for_boot() (*in module ironic.drivers.modules.ilo.common*), 3434
should_manage_boot() (*ironic.drivers.modules.agent.AgentDeploy method*), 3756
should_manage_boot() (*ironic.drivers.modules.agent_base.AgentBaseMixin method*), 3777
should_manage_boot() (*ironic.drivers.modules.pxe.PXEAnacondaDeploy method*), 4124
should_write_image() (*ironic.drivers.base.StorageInterface method*), 4284
should_write_image() (*ironic.drivers.modules.fake.FakeStorage method*), 3980
should_write_image() (*ironic.drivers.modules.storage.cinder.CinderStorage method*), 3724
should_write_image() (*ironic.drivers.modules.storage.external.ExternalStorage method*), 3728
should_write_image() (*ironic.drivers.modules.storage.noop.NoopStorage method*), 3733
show() (*ironic.common.glance_service.image_service.GlanceImageService method*), 2402
show() (*ironic.common.image_service.BaseImageService method*), 2525
show() (*ironic.common.image_service.FileImageService method*), 2528
show() (*ironic.common.image_service.HttpImageService method*), 2532
sig (*in module ironic.api.functions*), 2386
signature (*class in ironic.api.functions*), 2386
simple_headers (*ironic.api.app.IronicCORS attribute*), 2382
skip_automated_cleaning() (*in module ironic.conductor.utils*), 2910
SNMPClient (*class in ironic.drivers.modules.snmp*), 4140
SNMPDriverAPCMasterSwitch (*class in ironic.drivers.modules.snmp*), 4143
SNMPDriverAPCMasterSwitchPlus (*class in ironic.drivers.modules.snmp*), 4144
SNMPDriverAPCRackPDU (*class in ironic.drivers.modules.snmp*), 4145
SNMPDriverAten (*class in ironic.drivers.modules.snmp*), 4146
SNMPDriverAuto (*class in ironic.drivers.modules.snmp*), 4147
SNMPDriverBase (*class in ironic.drivers.modules.snmp*), 4147
SNMPDriverBaytechMRP27 (*class in ironic.drivers.modules.snmp*), 4150
SNMPDriverCyberPower (*class in ironic.drivers.modules.snmp*), 4151
SNMPDriverEatonPower (*class in ironic.drivers.modules.snmp*), 4152
SNMPDriverSimple (*class in ironic.drivers.modules.snmp*), 4153
SNMPDriverTeltronix (*class in ironic.drivers.modules.snmp*), 4155
SNMPFailure, 2488
SNMPHardware (*class in ironic.drivers.snmp*), 4326
SNMPPower (*class in ironic.drivers.modules.snmp*), 4156
SOFT_POWER_OFF (*in module ironic.common.states*), 2658
SOFT_REBOOT (*in module ironic.common.states*), 2658
spawn_after() (*ironic.conductor.task_manager.TaskManager method*), 2874
spawn_cleaning_error_handler() (*in module ironic.conductor.utils*), 2911
spawn_deploying_error_handler() (*in module ironic.conductor.utils*), 2911
spawn_rescue_error_handler() (*in module ironic.conductor.utils*), 2911
STABLE_STATES (*in module ironic.common.states*), 2658
stamp
  ironic-dbsync command line option, 1919
stamp command line option
  --help, 1921
  --revision <REVISION>, 1921
  -h, 1921
stamp() (*in module ironic.db.migration*), 3193
stamp() (*in module ironic.db.sqlalchemy.migration*), 3055
stamp() (*ironic.cmd.dbsync.DBCommand method*), 2399
standalone_ports_supported (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3069

standalone_ports_supported() (*ironic.objects.portgroup.Portgroup property*), 4508
standalone_ports_supported() (*ironic.objects.portgroup.PortgroupCRUDPayload property*), 4510
START (*ironic.objects.fields.NotificationStatus attribute*), 4421
start() (*ironic.common.rpc_service.RPCService method*), 2645
start() (*ironic.common.wsgi_service.WSGIService method*), 2696
start_console() (*ironic.drivers.base.ConsoleInterface method*), 4194
start_console() (*ironic.drivers.modules.fake.FakeConsole method*), 3925
start_console() (*ironic.drivers.modules.ipmitool.IPMIShellinaboxConsole method*), 4038
start_console() (*ironic.drivers.modules.ipmitool.IPMISocatConsole method*), 4041
start_console() (*ironic.drivers.modules.noop.NoConsole method*), 4092
start_deploy() (*in module ironic.conductor.deployments*), 2714
start_iscsi_target() (*ironic.drivers.modules.agent_client.AgentClient method*), 3839
start_shellinabox_console() (*in module ironic.drivers.modules.console_utils*), 3861
start_socat_console() (*in module ironic.drivers.modules.console_utils*), 3862
state (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057
state() (*ironic.objects.allocation.Allocation property*), 4351
state() (*ironic.objects.allocation.AllocationCRUDPayload property*), 4353
state() (*ironic.objects.deployment.Deployment property*), 4415
states (*ironic.api.controllers.v1.node.NodesController attribute*), 2230
status() (*ironic.objects.notification.EventType property*), 4471
status_code (*ironic.api.controllers.v1.utils.PassthruResponse attribute*), 2264
status_code (*ironic.api.functions.FunctionDefinition attribute*), 2386
status_off (*ironic.drivers.modules.snmp.SNMPDriverEatonPo attribute*), 4153
status_on (*ironic.drivers.modules.snmp.SNMPDriverEatonPow attribute*), 4153
status_pending_off (*ironic.drivers.modules.snmp.SNMPDriverEatonPower attribute*), 4153
status_pending_on (*ironic.drivers.modules.snmp.SNMPDriverEatonPower attribute*), 4153
step (*ironic.db.sqlalchemy.models.DeployTemplateStep attribute*), 3062
step_sanitize() (*in module ironic.api.controllers.v1.deploy_template*), 2174
steps() (*ironic.objects.deploy_template.DeployTemplate property*), 4406
steps() (*ironic.objects.deploy_template.DeployTemplateCRUDP property*), 4408
stop() (*ironic.common.rpc_service.RPCService method*), 2645
stop() (*ironic.common.wsgi_service.WSGIService method*), 2696
stop_console() (*ironic.drivers.base.ConsoleInterface method*), 4194
stop_console() (*ironic.drivers.modules.fake.FakeConsole method*), 3926
stop_console() (*ironic.drivers.modules.ipmitool.IPMIShellinaboxConsole method*), 4039
stop_console() (*ironic.drivers.modules.ipmitool.IPMISocatConsole method*), 4042
stop_console() (*ironic.drivers.modules.noop.NoConsole method*), 4093
stop_shellinabox_console() (*in module ironic.drivers.modules.console_utils*), 3864
stop_socat_console() (*in module ironic.drivers.modules.console_utils*), 3864
storage (*ironic.drivers.base.BareDriver attribute*), 4175
storage_interface (*ironic.db.sqlalchemy.models.Node*

attribute), 3065
storage_interface() (*ironic.objects.node.Node property*), 4450
storage_interface() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459
storage_interface() (*ironic.objects.node.NodeCRUDPayload property*), 4454
storage_interface() (*ironic.objects.node.NodePayload property*), 4462
storage_interface() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466
storage_interface() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470
StorageError, 2489
StorageInterface (*class in ironic.drivers.base*), 4282
StorageInterfaceFactory (*class in ironic.common.driver_factory*), 2453
store_agent_certificate() (*in module ironic.conductor.utils*), 2911
store_ramdisk_logs() (*in module ironic.drivers.utils*), 4338
string (*ironic.api.controllers.base.Version attribute*), 2375
string() (*in module ironic.common.args*), 2418
string_list() (*in module ironic.common.args*), 2419
StringAcceptsCallable (*class in ironic.objects.fields*), 4421
StringField (*class in ironic.objects.fields*), 4422
StringFieldThatAcceptsCallable (*class in ironic.objects.fields*), 4423
STUCK_STATES_TREATED_AS_FAIL (*in module ironic.common.states*), 2659
SUCCESS (*ironic.objects.fields.NotificationStatus attribute*), 4421
supported (*ironic.drivers.base.BaseInterface attribute*), 4183
supported (*ironic.drivers.hardware_type.AbstractHardwareType attribute*), 4312
supported (*ironic.drivers.modules.iscsi_deploy.ISCSIDeploy attribute*), 4062
supported() (*ironic.api.controllers.v1.node.BootDeviceController method*), 2186
supported_bios_interfaces() (*ironic.drivers.drac.IDRACHardware property*), 4301
supported_bios_interfaces() (*ironic.drivers.fake_hardware.FakeHardware property*), 4304
supported_bios_interfaces() (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4312
supported_bios_interfaces() (*ironic.drivers.ilo.IloHardware property*), 4318
supported_bios_interfaces() (*ironic.drivers.irmc.IRMCHardware property*), 4323
supported_bios_interfaces() (*ironic.drivers.redfish.RedfishHardware property*), 4325
supported_boot_interfaces() (*ironic.drivers.drac.IDRACHardware property*), 4302
supported_boot_interfaces() (*ironic.drivers.fake_hardware.FakeHardware property*), 4304
supported_boot_interfaces() (*ironic.drivers.generic.GenericHardware property*), 4308
supported_boot_interfaces() (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4313
supported_boot_interfaces() (*ironic.drivers.ilo.Ilo5Hardware property*), 4317
supported_boot_interfaces() (*ironic.drivers.ilo.IloHardware property*), 4318
supported_boot_interfaces() (*ironic.drivers.irmc.IRMCHardware property*), 4323
supported_boot_interfaces() (*ironic.drivers.redfish.RedfishHardware property*), 4325
SUPPORTED_BOOT_MODE_LEGACY_BIOS_AND_UEFI (*in module ironic.drivers.modules.ilo.common*), 3496
SUPPORTED_BOOT_MODE_LEGACY_BIOS_ONLY (*in module ironic.drivers.modules.ilo.common*), 3496
SUPPORTED_BOOT_MODE_UEFI_ONLY

(*in module ironic.drivers.modules.ilo.common*), 3406

`supported_console_interfaces()` (*ironic.drivers.fake_hardware.FakeHardware property*), 4304

`supported_console_interfaces()` (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4313

`supported_console_interfaces()` (*ironic.drivers.ilo.IloHardware property*), 4319

`supported_console_interfaces()` (*ironic.drivers.ipmi.IPMIHardware property*), 4321

`supported_console_interfaces()` (*ironic.drivers.irmc.IRMCHardware property*), 4323

`supported_deploy_interfaces()` (*ironic.drivers.fake_hardware.FakeHardware property*), 4305

`supported_deploy_interfaces()` (*ironic.drivers.generic.GenericHardware property*), 4308

`supported_deploy_interfaces()` (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4313

`supported_inspect_interfaces()` (*ironic.drivers.drac.IDRACHardware property*), 4302

`supported_inspect_interfaces()` (*ironic.drivers.fake_hardware.FakeHardware property*), 4305

`supported_inspect_interfaces()` (*ironic.drivers.generic.GenericHardware property*), 4308

`supported_inspect_interfaces()` (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4313

`supported_inspect_interfaces()` (*ironic.drivers.ilo.IloHardware property*), 4319

`supported_inspect_interfaces()` (*ironic.drivers.irmc.IRMCHardware property*), 4324

`supported_inspect_interfaces()` (*ironic.drivers.redfish.RedfishHardware property*), 4325

`supported_management_interfaces()` (*ironic.drivers.drac.IDRACHardware property*), 4302

`supported_management_interfaces()` (*ironic.drivers.fake_hardware.FakeHardware property*), 4305

`supported_management_interfaces()` (*ironic.drivers.generic.ManualManagementHardware property*), 4310

`supported_management_interfaces()` (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4313

`supported_management_interfaces()` (*ironic.drivers.ibmc.IBMCHardware property*), 4315

`supported_management_interfaces()` (*ironic.drivers.ilo.Ilo5Hardware property*), 4317

`supported_management_interfaces()` (*ironic.drivers.ilo.IloHardware property*), 4319

`supported_management_interfaces()` (*ironic.drivers.intel_ipmi.IntelIPMIHardware property*), 4320

`supported_management_interfaces()` (*ironic.drivers.ipmi.IPMIHardware property*), 4321

`supported_management_interfaces()` (*ironic.drivers.irmc.IRMCHardware property*), 4324

`supported_management_interfaces()` (*ironic.drivers.redfish.RedfishHardware property*), 4325

`supported_management_interfaces()` (*ironic.drivers.snmp.SNMPHardware property*), 4327

`supported_management_interfaces()` (*ironic.drivers.xclarity.XClarityHardware property*), 4340

`supported_network_interfaces()` (*ironic.drivers.fake_hardware.FakeHardware property*), 4305

`supported_network_interfaces()` (*ironic.drivers.generic.GenericHardware property*), 4308

`supported_network_interfaces()` (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4314

`supported_power_interfaces()` (*ironic.drivers.drac.IDRACHardware property*), 4302

`supported_power_interfaces()` (*ironic.drivers.fake_hardware.FakeHardware property*), 4306 supported_power_interfaces() (*ironic.drivers.generic.ManualManagementHardware property*), 4310

supported_power_interfaces() (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4314

supported_power_interfaces() (*ironic.drivers.ibmc.IBMCHardware property*), 4316

supported_power_interfaces() (*ironic.drivers.ilo.IloHardware property*), 4319

supported_power_interfaces() (*ironic.drivers.ipmi.IPMIHardware property*), 4322

supported_power_interfaces() (*ironic.drivers.irmc.IRMCHardware property*), 4324

supported_power_interfaces() (*ironic.drivers.redfish.RedfishHardware property*), 4326

supported_power_interfaces() (*ironic.drivers.snmp.SNMPHardware property*), 4327

supported_power_interfaces() (*ironic.drivers.xclarity.XClarityHardware property*), 4340

supported_raid_interfaces() (*ironic.drivers.drac.IDRACHardware property*), 4302

supported_raid_interfaces() (*ironic.drivers.fake_hardware.FakeHardware property*), 4306

supported_raid_interfaces() (*ironic.drivers.generic.GenericHardware property*), 4308

supported_raid_interfaces() (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4314

supported_raid_interfaces() (*ironic.drivers.ibmc.IBMCHardware property*), 4316

supported_raid_interfaces() (*ironic.drivers.ilo.Ilo5Hardware property*), 4318

supported_raid_interfaces() (*ironic.drivers.irmc.IRMCHardware property*), 4324

supported_raid_interfaces() (*ironic.drivers.redfish.RedfishHardware property*), 4326

supported_rescue_interfaces() (*ironic.drivers.fake_hardware.FakeHardware property*), 4306

supported_rescue_interfaces() (*ironic.drivers.generic.GenericHardware property*), 4309

supported_rescue_interfaces() (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4314

supported_storage_interfaces() (*ironic.drivers.fake_hardware.FakeHardware property*), 4306

supported_storage_interfaces() (*ironic.drivers.generic.GenericHardware property*), 4309

supported_storage_interfaces() (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4314

supported_vendor_interfaces() (*ironic.drivers.drac.IDRACHardware property*), 4303

supported_vendor_interfaces() (*ironic.drivers.fake_hardware.FakeHardware property*), 4306

supported_vendor_interfaces() (*ironic.drivers.generic.ManualManagementHardware property*), 4310

supported_vendor_interfaces() (*ironic.drivers.hardware_type.AbstractHardwareType property*), 4315

supported_vendor_interfaces() (*ironic.drivers.ibmc.IBMCHardware property*), 4316

supported_vendor_interfaces() (*ironic.drivers.ilo.IloHardware property*), 4319

supported_vendor_interfaces() (*ironic.drivers.ipmi.IPMIHardware property*), 4322

supported_vendor_interfaces() (*ironic.drivers.redfish.RedfishHardware property*), 4326

supports_is_smartnic() (*ironic.objects.port.Port class method*), 4492

supports_physical_network() (*ironic.objects.port.Port class method*), 4492

supports_power_sync() (*ironic.drivers.base.PowerInterface method*), 4267

supports_power_sync() (*ironic.drivers.modules.agent_power.AgentPower method*), 3849
supports_version() (*ironic.objects.base.IronicObject class method*), 4360
swap_mib() (*ironic.objects.deployment.Deployment property*), 4415
swift_temp_url() (*ironic.common.glance_service.image_service.GlanceImageService method*), 2403
SwiftAPI (*class in ironic.common.swift*), 2662
SwiftObjectNotFoundError, 2489
SwiftOperationError, 2489
switch_pxe_config() (*in module ironic.drivers.modules.deploy_utils*), 3899
switch_to_tenant_network() (*ironic.drivers.modules.agent_base.AgentOobStepsMixin method*), 3793
sync() (*ironic.drivers.modules.agent_client.AgentClient method*), 3841
sync_boot_mode() (*in module ironic.drivers.modules.boot_mode_utils*), 3856
sync_node_setting() (*ironic.objects.bios.BIOSSettingList class method*), 4376
SYS_OBJ_OID (*ironic.drivers.modules.snmp.SNMPDriverAuto attribute*), 4147
SYSTEM (*in module ironic.common.components*), 2447
system_id (*ironic.drivers.modules.snmp.SNMPDriverAPCMasterSwitch attribute*), 4144
system_id (*ironic.drivers.modules.snmp.SNMPDriverAPCMasterSwitchPlus attribute*), 4145
system_id (*ironic.drivers.modules.snmp.SNMPDriverAPCRackPDU attribute*), 4146
system_id (*ironic.drivers.modules.snmp.SNMPDriverAten attribute*), 4147
system_id (*ironic.drivers.modules.snmp.SNMPDriverCyberPower attribute*), 4152
system_id (*ironic.drivers.modules.snmp.SNMPDriverEatonPower attribute*), 4153
system_id (*ironic.drivers.modules.snmp.SNMPDriverTeltronix attribute*), 4156

T

table_args() (*in module ironic.db.sqlalchemy.models*), 3072
tag (*ironic.db.sqlalchemy.models.NodeTag attribute*), 3066
take_over() (*ironic.drivers.base.DeployInterface method*), 4203
take_over() (*ironic.drivers.modules.agent_base.AgentBaseMix method*), 3777
take_over() (*ironic.drivers.modules.ansible.deploy.AnsibleDep method*), 3219
take_over() (*ironic.drivers.modules.fake.FakeDeploy method*), 3933
take_over_allocation() (*ironic.db.api.Connection method*), 3171
take_over_allocation() (*ironic.db.sqlalchemy.api.Connection method*), 3027
target (*ironic.conductor.manager.ConductorManager attribute*), 2730
target_power_state (*ironic.db.sqlalchemy.models.Node attribute*), 3065
target_power_state() (*ironic.objects.node.Node property*), 4450
target_power_state() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459
target_power_state() (*ironic.objects.node.NodeCRUDPayload property*), 4454
target_power_state() (*ironic.objects.node.NodePayload property*), 4462
target_power_state() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466
target_power_state() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470
target_provision_state (*ironic.db.sqlalchemy.models.Node attribute*), 3065
target_provision_state() (*ironic.objects.node.Node property*), 4450
target_provision_state() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459
target_provision_state() (*ironic.objects.node.NodeCRUDPayload property*), 4454
target_provision_state()

(*ironic.objects.node.NodePayload property*), 4462
`target_provision_state()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466
`target_provision_state()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470
`target_raid_config` (*ironic.db.sqlalchemy.models.Node attribute*), 3065
`target_raid_config()` (*ironic.objects.node.Node property*), 4450
`target_state()` (*ironic.common.fsm.FSM property*), 2498
`TaskManager` (*class in ironic.conductor.task_manager*), 2869
`tear_down()` (*ironic.drivers.base.DeployInterface method*), 4204
`tear_down()` (*ironic.drivers.modules.agent_base.AgentBaseMixin method*), 3777
`tear_down()` (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3220
`tear_down()` (*ironic.drivers.modules.fake.FakeDeploy method*), 3934
`tear_down_agent()` (*ironic.drivers.modules.agent_base.AgentDeployMixin method*), 3792
`tear_down_agent()` (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3221
`tear_down_cleaning()` (*ironic.drivers.base.DeployInterface method*), 4205
`tear_down_cleaning()` (*ironic.drivers.modules.agent_base.AgentBaseMixin method*), 3779
`tear_down_cleaning()` (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3221
`tear_down_inband_cleaning()` (*in module ironic.drivers.modules.deploy_utils*), 3902
`tear_down_storage_configuration()` (*in module ironic.drivers.modules.deploy_utils*), 3903
`tempdir()` (*in module ironic.common.utils*), 2690
`template_sanitize()` (*in module ironic.api.controllers.v1.deploy_template*), 2174
`TemporaryFailure`, 2489
`TempUrlCacheElement` (*class in ironic.common.glance_service.image_service*), 2406
`TFTPImageCache` (*class in ironic.common.pxe_utils*), 2605
`third_method_sync()` (*ironic.drivers.modules.fake.FakeVendorB method*), 3987
`to_event_type_field()` (*ironic.objects.notification.EventType method*), 4471
`to_policy_values()` (*ironic.common.context.RequestContext method*), 2448
`to_power()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466
`touch()` (*ironic.objects.conductor.Conductor method*), 4394
`touch_conductor()` (*ironic.db.api.Connection method*), 3173
`touch_conductor()` (*ironic.db.sqlalchemy.api.Connection method*), 3029
`touch_node_provisioning()` (*ironic.db.api.Connection method*), 3173
`touch_node_provisioning()` (*ironic.db.sqlalchemy.api.Connection method*), 3030
`touch_provisioning()` (*ironic.objects.node.Node method*), 4450
`trace_cls()` (*in module ironic.common.profiler*), 2604
`TraceDisabled` (*class in ironic.common.image_format_inspector*), 2514
`Trait` (*class in ironic.objects.trait*), 4510
`trait` (*ironic.db.sqlalchemy.models.NodeTrait attribute*), 3067
`trait()` (*ironic.objects.trait.Trait property*), 4514
`TraitList` (*class in ironic.objects.trait*), 4514
`traits` (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057
`traits()` (*ironic.objects.allocation.Allocation property*), 4351

traits() (*ironic.objects.allocation.AllocationCRUDPayload property*), 4353

traits() (*ironic.objects.node.Node property*), 4450

traits() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459

traits() (*ironic.objects.node.NodeCRUDPayload property*), 4454

traits() (*ironic.objects.node.NodePayload property*), 4462

traits() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466

traits() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470

translate_from_glance() (*in module ironic.common.glance_service.service_utils*), 2408

translate_xclarity_power_action() (*in module ironic.drivers.modules.xclarity.common*), 3739

translate_xclarity_power_state() (*in module ironic.drivers.modules.xclarity.common*), 3740

try_set_boot_device() (*in module ironic.drivers.modules.deploy_utils*), 3904

type (*ironic.db.sqlalchemy.models.VolumeConnector attribute*), 3070

type() (*ironic.objects.volume_connector.VolumeConnector property*), 4532

type() (*ironic.objects.volume_connector.VolumeConnectorCRUDPayload property*), 4534

types() (*in module ironic.common.args*), 2421

U

UEFI (*in module ironic.common.boot_modes*), 2427

umount() (*in module ironic.common.utils*), 2690

Unauthorized, 2489

unbind_neutron_port() (*in module ironic.common.neutron*), 2588

unconfigure_tenant_networks() (*ironic.drivers.base.NetworkInterface method*), 4256

unconfigure_tenant_networks() (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3601

unconfigure_tenant_networks() (*ironic.drivers.modules.network.neutron.NeutronNetwork method*), 3610

unconfigure_tenant_networks() (*ironic.drivers.modules.network.noop.NoopNetwork method*), 3618

[illegible] (*ironic.drivers.modules.snmp.SNMPDriverBaytechMRP attribute*), 4151

unix_file_modification_datetime() (*in module ironic.common.utils*), 2691

UNKNOWN (*in module ironic.common.indicator_states*), 2557

UnknownArgument, 2489

UnknownAttribute, 2489

unpublish_image() (*ironic.drivers.modules.image_utils.ImageHandler method*), 3995

unpublish_image_for_node() (*ironic.drivers.modules.image_utils.ImageHandler class method*), 3996

unregister() (*ironic.objects.conductor.Conductor method*), 4394

unregister_all_hardware_interfaces() (*ironic.objects.conductor.Conductor method*), 4394

unregister_conductor() (*ironic.db.api.Connection method*), 3174

unregister_conductor() (*ironic.db.sqlalchemy.api.Connection method*), 3030

unregister_conductor_hardware_interfaces() (*ironic.db.api.Connection method*), 3175

unregister_conductor_hardware_interfaces() (*ironic.db.sqlalchemy.api.Connection method*), 3031

unrescue() (*ironic.drivers.base.RescueInterface method*), 4281

unrescue() (*ironic.drivers.modules.agent.AgentRescue method*), 3768

unrescue() (*ironic.drivers.modules.fake.FakeRescue method*), 3975

unrescue() (*ironic.drivers.modules.noop.NoRescue method*), 4102

UNRESCUEFAIL (*in module ironic.common.states*), 2659

UNRESCUING (*in module ironic.common.states*), 2659

unset_node_tags() (*ironic.db.api.Connection method*), 3175

unset_node_tags() (*ironic.db.sqlalchemy.api.Connection*

method), 3032
`unset_node_traits()` (*ironic.db.api.Connection method*), 3176
`unset_node_traits()` (*ironic.db.sqlalchemy.api.Connection method*), 3032
`UNSTABLE_STATES` (*in module ironic.common.states*), 2660
`UnsupportedDriverExtension`, 2490
`update_allocation()` (*ironic.db.api.Connection method*), 3176
`update_allocation()` (*ironic.db.sqlalchemy.api.Connection method*), 3033
`UPDATE_ALLOWED_STATES` (*in module ironic.common.states*), 2660
`update_bios_setting_list()` (*ironic.db.api.Connection method*), 3178
`update_bios_setting_list()` (*ironic.db.sqlalchemy.api.Connection method*), 3034
`update_boot_mode()` (*in module ironic.drivers.modules.ilo.common*), 3438
`update_chassis()` (*ironic.db.api.Connection method*), 3181
`update_chassis()` (*ironic.db.sqlalchemy.api.Connection method*), 3037
`update_cipher_suite_cmd()` (*in module ironic.drivers.modules.ipmitool*), 4054
`update_deploy_template()` (*ironic.db.api.Connection method*), 3181
`update_deploy_template()` (*ironic.db.sqlalchemy.api.Connection method*), 3037
`update_dhcp()` (*ironic.common.dhcp_factory.DHCPFactory method*), 2450
`update_dhcp_opts()` (*ironic.dhcp.base.BaseDHCP method*), 3195
`update_dhcp_opts()` (*ironic.dhcp.neutron.NeutronDHCPApi method*), 3202
`update_dhcp_opts()` (*ironic.dhcp.none.NoneDHCPApi method*), 3208
`update_driver_config()` (*ironic.drivers.modules.image_utils.ImageHandler method*), 3997
`update_firmware()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3478
`update_firmware()` (*ironic.drivers.modules.redfish.management.RedfishMana method*), 3680
`update_firmware_sum()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3479
`update_ipmi_properties()` (*in module ironic.drivers.modules.ilo.common*), 3439
`update_ipmi_properties()` (*in module ironic.drivers.modules.irmc.common*), 3528
`update_neutron_port()` (*in module ironic.common.neutron*), 2589
`update_next_step_index()` (*in module ironic.conductor.utils*), 2912
`update_node()` (*ironic.conductor.manager.ConductorManager method*), 2730
`update_node()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2836
`update_node()` (*ironic.db.api.Connection method*), 3183
`update_node()` (*ironic.db.sqlalchemy.api.Connection method*), 3039
`update_object_meta()` (*ironic.common.swift.SwiftAPI method*), 2668
`update_opt_defaults()` (*in module ironic.conf.opts*), 2925
`update_port()` (*ironic.conductor.manager.ConductorManager method*), 2730
`update_port()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2838
`update_port()` (*ironic.db.api.Connection method*), 3185
`update_port()` (*ironic.db.sqlalchemy.api.Connection method*), 3041
`update_port_address()` (*in module*

ironic.common.neutron), 2591
update_port_dhcp_opts() (*ironic.dhcp.base.BaseDHCP method*), 3198
update_port_dhcp_opts() (*ironic.dhcp.neutron.NeutronDHCPApi method*), 3204
update_port_dhcp_opts() (*ironic.dhcp.none.NoneDHCPApi method*), 3211
update_portgroup() (*ironic.conductor.manager.ConductorManager method*), 2730
update_portgroup() (*ironic.conductor.rpcapi.ConductorAPI method*), 2839
update_portgroup() (*ironic.db.api.Connection method*), 3186
update_portgroup() (*ironic.db.sqlalchemy.api.Connection method*), 3042
update_raid_config() (*in module ironic.drivers.modules.redfish.raid*), 3710
update_raid_info() (*in module ironic.common.raid*), 2636
update_state_in_older_versions() (*in module ironic.api.controllers.v1.node*), 2235
update_to_latest_versions() (*ironic.db.api.Connection method*), 3187
update_to_latest_versions() (*ironic.db.sqlalchemy.api.Connection method*), 3043
update_volume_connector() (*ironic.conductor.manager.ConductorManager method*), 2730
update_volume_connector() (*ironic.conductor.rpcapi.ConductorAPI method*), 2841
update_volume_connector() (*ironic.db.api.Connection method*), 3188
update_volume_connector() (*ironic.db.sqlalchemy.api.Connection method*), 3045
update_volume_target() (*ironic.conductor.manager.ConductorManager method*), 2730
update_volume_target() (*ironic.conductor.rpcapi.ConductorAPI method*), 2844
update_volume_target() (*ironic.db.api.Connection method*), 3190
update_volume_target() (*ironic.db.sqlalchemy.api.Connection method*), 3047
updated_at (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057
updated_at (*ironic.db.sqlalchemy.models.BIOSSetting attribute*), 3058
updated_at (*ironic.db.sqlalchemy.models.Chassis attribute*), 3058
updated_at (*ironic.db.sqlalchemy.models.Conductor attribute*), 3059
updated_at (*ironic.db.sqlalchemy.models.ConductorHardwareI attribute*), 3060
updated_at (*ironic.db.sqlalchemy.models.DeployTemplate attribute*), 3061
updated_at (*ironic.db.sqlalchemy.models.DeployTemplateStep attribute*), 3062
updated_at (*ironic.db.sqlalchemy.models.Node attribute*), 3065
updated_at (*ironic.db.sqlalchemy.models.NodeTag attribute*), 3066
updated_at (*ironic.db.sqlalchemy.models.NodeTrait attribute*), 3067
updated_at (*ironic.db.sqlalchemy.models.Port attribute*), 3068
updated_at (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3069
updated_at (*ironic.db.sqlalchemy.models.VolumeConnector attribute*), 3070
updated_at (*ironic.db.sqlalchemy.models.VolumeTarget attribute*), 3071
updated_at() (*ironic.objects.allocation.Allocation property*), 4351
updated_at() (*ironic.objects.allocation.AllocationCRUDNotifi property*), 4352
updated_at() (*ironic.objects.allocation.AllocationCRUDPaylo property*), 4353
updated_at() (*ironic.objects.bios.BIOSSetting property*), 4370
updated_at() (*ironic.objects.bios.BIOSSettingList property*), 4377
updated_at() (*ironic.objects.chassis.Chassis property*), 4385
updated_at() (*ironic.objects.chassis.ChassisCRUDNotification property*), 4386

updated_at() (*ironic.objects.chassis.ChassisCRUDPayload property*), 4387
updated_at() (*ironic.objects.conductor.Conductor property*), 4395
updated_at() (*ironic.objects.deploy_template.DeployTemplate property*), 4406
updated_at() (*ironic.objects.deploy_template.DeployTemplateCRUDNotification property*), 4407
updated_at() (*ironic.objects.deploy_template.DeployTemplateCRUDPayload property*), 4408
updated_at() (*ironic.objects.deployment.Deployment property*), 4415
updated_at() (*ironic.objects.node.Node property*), 4450
updated_at() (*ironic.objects.node.NodeConsoleNotification property*), 4455
updated_at() (*ironic.objects.node.NodeCorrectedPowerStateNotification property*), 4456
updated_at() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459
updated_at() (*ironic.objects.node.NodeCRUDNotification property*), 4451
updated_at() (*ironic.objects.node.NodeCRUDPayload property*), 4454
updated_at() (*ironic.objects.node.NodeMaintenanceNotification property*), 4460
updated_at() (*ironic.objects.node.NodePayload property*), 4463
updated_at() (*ironic.objects.node.NodeSetPowerStateNotification property*), 4463
updated_at() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466
updated_at() (*ironic.objects.node.NodeSetProvisionStateNotification property*), 4467
updated_at() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470
updated_at() (*ironic.objects.notification.EventType property*), 4472
updated_at() (*ironic.objects.notification.NotificationBase property*), 4473
updated_at() (*ironic.objects.notification.NotificationPayloadBase property*), 4475
updated_at() (*ironic.objects.notification.NotificationPublisher property*), 4475
updated_at() (*ironic.objects.port.Port property*), 4493
updated_at() (*ironic.objects.port.PortCRUDNotification property*), 4494
updated_at() (*ironic.objects.port.PortCRUDPayload property*), 4495
updated_at() (*ironic.objects.portgroup.Portgroup property*), 4508
updated_at() (*ironic.objects.portgroup.PortgroupCRUDNotific property*), 4509
updated_at() (*ironic.objects.portgroup.PortgroupCRUDPaylo property*), 4510
updated_at() (*ironic.objects.trait.Trait property*), 4514
updated_at() (*ironic.objects.trait.TraitList property*), 4518
updated_at() (*ironic.objects.volume_connector.VolumeConnec property*), 4532
updated_at() (*ironic.objects.volume_connector.VolumeConnec property*), 4533
updated_at() (*ironic.objects.volume_connector.VolumeConnec property*), 4534
updated_at() (*ironic.objects.volume_target.VolumeTarget property*), 4550
updated_at() (*ironic.objects.volume_target.VolumeTargetCRU property*), 4551
updated_at() (*ironic.objects.volume_target.VolumeTargetCRU property*), 4552
upgrade
  ironic-dbsync command line option, 1919
upgrade command line option
  --help, 1922
  --revision <REVISION>, 1922
  -h, 1922
upgrade() (*in module ironic.db.migration*), 3193
upgrade() (*in module ironic.db.sqlalchemy.migration*), 3055
upgrade() (*ironic.cmd.dbsync.DBCommand method*), 2399
upgrade_lock() (*ironic.conductor.task_manager.TaskManager method*), 2875
url (*ironic.common.glance_service.image_service.TempUrlCache attribute*), 2406
url_expires_at (*ironic.common.glance_service.image_service.TempUrlCa attribute*), 2406
uuid (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057
uuid (*ironic.db.sqlalchemy.models.Chassis attribute*), 3058
uuid (*ironic.db.sqlalchemy.models.DeployTemplate attribute*), 3061
uuid (*ironic.db.sqlalchemy.models.Node attribute*), 3065
uuid (*ironic.db.sqlalchemy.models.Port at-*

tribute), 3068
`uuid` (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3069
`uuid` (*ironic.db.sqlalchemy.models.VolumeConnector attribute*), 3070
`uuid` (*ironic.db.sqlalchemy.models.VolumeTarget attribute*), 3071
`uuid()` (*in module ironic.common.args*), 2421
`uuid()` (*ironic.objects.allocation.Allocation property*), 4351
`uuid()` (*ironic.objects.allocation.AllocationCRUDPayload property*), 4353
`uuid()` (*ironic.objects.chassis.Chassis property*), 4385
`uuid()` (*ironic.objects.chassis.ChassisCRUDPayload property*), 4387
`uuid()` (*ironic.objects.deploy_template.DeployTemplate property*), 4406
`uuid()` (*ironic.objects.deploy_template.DeployTemplateCRUDPayload property*), 4408
`uuid()` (*ironic.objects.deployment.Deployment property*), 4415
`uuid()` (*ironic.objects.node.Node property*), 4450
`uuid()` (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459
`uuid()` (*ironic.objects.node.NodeCRUDPayload property*), 4454
`uuid()` (*ironic.objects.node.NodePayload property*), 4463
`uuid()` (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466
`uuid()` (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470
`uuid()` (*ironic.objects.port.Port property*), 4493
`uuid()` (*ironic.objects.port.PortCRUDPayload property*), 4495
`uuid()` (*ironic.objects.portgroup.Portgroup property*), 4508
`uuid()` (*ironic.objects.portgroup.PortgroupCRUDPayload property*), 4510
`uuid()` (*ironic.objects.volume_connector.VolumeConnector property*), 4532
`uuid()` (*ironic.objects.volume_connector.VolumeConnectorCRUDPayload property*), 4534
`uuid()` (*ironic.objects.volume_target.VolumeTarget property*), 4550
`uuid()` (*ironic.objects.volume_target.VolumeTargetCRUDPayload property*), 4552
`uuid_or_name()` (*in module ironic.common.args*), 2422
`UUIDField` (*class in ironic.objects.fields*), 4423

V

`validate()` (*in module ironic.common.args*), 2424
`validate()` (*in module ironic.drivers.modules.iscsi_deploy*), 4083
`validate()` (*ironic.api.controllers.v1.node.NodesController method*), 2230
`validate()` (*ironic.drivers.base.BaseInterface method*), 4184
`validate()` (*ironic.drivers.base.NetworkInterface method*), 4257
`validate()` (*ironic.drivers.base.RAIDInterface method*), 4276
`validate()` (*ironic.drivers.base.VendorInterface method*), 4288
`validate()` (*ironic.drivers.modules.agent.AgentDeploy method*), 3756
`validate()` (*ironic.drivers.modules.agent.AgentRescue method*), 3770
`validate()` (*ironic.drivers.modules.agent_power.AgentPower method*), 3850
`validate()` (*ironic.drivers.modules.ansible.deploy.AnsibleDepl method*), 3222
`validate()` (*ironic.drivers.modules.drac.bios.DracWSManBIOS method*), 3229
`validate()` (*ironic.drivers.modules.drac.inspect.DracWSManIn method*), 3253
`validate()` (*ironic.drivers.modules.drac.management.DracWSM method*), 3275
`validate()` (*ironic.drivers.modules.drac.power.DracWSManPo method*), 3284
`validate()` (*ironic.drivers.modules.drac.vendor_passthru.Drac method*), 3326
`validate()` (*ironic.drivers.modules.fake.FakeBIOS method*), 3915
`validate()` (*ironic.drivers.modules.fake.FakeBoot method*), 3921
`validate()` (*ironic.drivers.modules.fake.FakeConsole method*), 3926
`validate()` (*ironic.drivers.modules.fake.FakeDeploy method*), 3935
`validate()` (*ironic.drivers.modules.fake.FakeInspect method*), 3939
`validate()` (*ironic.drivers.modules.fake.FakeManagement method*), 3960
`validate()` (*ironic.drivers.modules.fake.FakePower method*), 3967
`validate()` (*ironic.drivers.modules.fake.FakeRescue*

method), 3976
`validate()` (*ironic.drivers.modules.fake.FakeStorage method*), 3981
`validate()` (*ironic.drivers.modules.fake.FakeVendorA method*), 3984
`validate()` (*ironic.drivers.modules.fake.FakeVendorB method*), 3987
`validate()` (*ironic.drivers.modules.ibmc.management.IBMCManagement method*), 3341
`validate()` (*ironic.drivers.modules.ibmc.power.IBMCPower method*), 3351
`validate()` (*ironic.drivers.modules.ibmc.vendor.IBMCVendor method*), 3365
`validate()` (*ironic.drivers.modules.ilo.bios.IloBIOS method*), 3371
`validate()` (*ironic.drivers.modules.ilo.boot.IloUefiHttpsBoot method*), 3383
`validate()` (*ironic.drivers.modules.ilo.boot.IloVirtualMediaBoot method*), 3394
`validate()` (*ironic.drivers.modules.ilo.console.IloConsoleInterface method*), 3442
`validate()` (*ironic.drivers.modules.ilo.inspect.IloInspect method*), 3450
`validate()` (*ironic.drivers.modules.ilo.management.IloManagement method*), 3480
`validate()` (*ironic.drivers.modules.ilo.power.IloPower method*), 3487
`validate()` (*ironic.drivers.modules.ilo.vendor.VendorPassthru method*), 3498
`validate()` (*ironic.drivers.modules.inspector.Inspector method*), 4014
`validate()` (*ironic.drivers.modules.ipmitool.IPMIConsole method*), 4017
`validate()` (*ironic.drivers.modules.ipmitool.IPMIManagement method*), 4027
`validate()` (*ironic.drivers.modules.ipmitool.IPMIPower method*), 4035
`validate()` (*ironic.drivers.modules.ipmitool.VendorPassthru method*), 4047
`validate()` (*ironic.drivers.modules.irmc.bios.IRMCBIOS method*), 3505
`validate()` (*ironic.drivers.modules.irmc.boot.IRMCVirtualMediaBoot method*), 3514
`validate()` (*ironic.drivers.modules.irmc.inspect.IRMCInspect method*), 3530
`validate()` (*ironic.drivers.modules.irmc.management.IRMCManagement method*), 3547
`validate()` (*ironic.drivers.modules.irmc.power.IRMCPower method*), 3573
`validate()` (*ironic.drivers.modules.iscsi_deploy.ISCSIDeploy method*), 4062
`validate()` (*ironic.drivers.modules.network.flat.FlatNetwork method*), 3602
`validate()` (*ironic.drivers.modules.network.neutron.NeutronNe method*), 3611
`validate()` (*ironic.drivers.modules.noop.FailMixin method*), 4085
`validate()` (*ironic.drivers.modules.noop_mgmt.NoopManagem method*), 4116
`validate()` (*ironic.drivers.modules.pxe.PXEAnacondaDeploy method*), 4124
`validate()` (*ironic.drivers.modules.pxe.PXERamdiskDeploy method*), 4129
`validate()` (*ironic.drivers.modules.pxe_base.PXEBaseMixin method*), 4136
`validate()` (*ironic.drivers.modules.redfish.bios.RedfishBIOS method*), 3629
`validate()` (*ironic.drivers.modules.redfish.boot.RedfishVirtualM method*), 3639
`validate()` (*ironic.drivers.modules.redfish.inspect.RedfishInspe method*), 3645
`validate()` (*ironic.drivers.modules.redfish.management.Redfish method*), 3681
`validate()` (*ironic.drivers.modules.redfish.power.RedfishPower method*), 3690
`validate()` (*ironic.drivers.modules.redfish.vendor.RedfishVendo method*), 3719
`validate()` (*ironic.drivers.modules.snmp.SNMPPower method*), 4162
`validate()` (*ironic.drivers.modules.storage.cinder.CinderStorag method*), 3725
`validate()` (*ironic.drivers.modules.storage.external.ExternalSt method*), 3729
`validate()` (*ironic.drivers.modules.storage.noop.NoopStorage method*), 3734
`validate()` (*ironic.drivers.modules.xclarity.management.XClar method*), 3745
`validate()` (*ironic.drivers.modules.xclarity.power.XClarityPow method*), 3750
`validate()` (*ironic.drivers.utils.MixinVendorInterface method*), 4328
`validate_and_normalize_datapath_id()` (*in module ironic.common.utils*), 2691
`validate_and_normalize_mac()` (*in module ironic.common.utils*), 2693
`validate_boot_parameters_for_trusted_boot()` (*in module ironic.common.pxe_utils*), 2631
`validate_capabilities()` (*in module ironic.drivers.modules.deploy_utils*), 3905
`validate_checksum()` (*in module*

ironic.common.checksum_utils), 2432
validate_conductor_group() (*in module ironic.common.utils*), 2694
validate_configuration() (*in module ironic.common.raid*), 2637
validate_deploy_steps() (*in module ironic.conductor.deployments*), 2715
validate_driver_interfaces() (*ironic.conductor.manager.ConductorManager method*), 2730
validate_driver_interfaces() (*ironic.conductor.rpcapi.ConductorAPI method*), 2847
validate_href() (*ironic.common.image_service.BaseImageService method*), 2526
validate_href() (*ironic.common.image_service.FileImageService method*), 2529
validate_href() (*ironic.common.image_service.HttpImageService method*), 2533
validate_http_provisioning_configuration() (*in module ironic.drivers.modules.agent*), 3772
validate_image_properties() (*in module ironic.drivers.modules.deploy_utils*), 3906
validate_image_proxies() (*in module ironic.drivers.modules.agent*), 3773
validate_inspection() (*ironic.common.neutron.NeutronNetworkInterfaceMixin method*), 2573
validate_inspection() (*ironic.drivers.base.BootInterface method*), 4190
validate_inspection() (*ironic.drivers.base.NetworkInterface method*), 4258
validate_inspection() (*ironic.drivers.modules.ilo.boot.IloUefiHttpsBoot method*), 3385
validate_inspection() (*ironic.drivers.modules.ilo.boot.IloVirtualMediaBoot method*), 3395
validate_inspection() (*ironic.drivers.modules.network.noop.NoopNetwork method*), 3619
validate_inspection() (*ironic.drivers.modules.pxe_base.PXEBaseMixin method*), 4138
validate_inspection() (*ironic.drivers.modules.redfish.boot.RedfishVirtualMediaB method*), 3641
validate_instance_info_traits() (*in module ironic.conductor.utils*), 2913
validate_job_queue() (*in module ironic.drivers.modules.drac.job*), 3257
validate_kickstart_file() (*in module ironic.common.pxe_utils*), 2631
validate_kickstart_template() (*in module ironic.common.pxe_utils*), 2631
validate_limit() (*in module ironic.api.controllers.v1.utils*), 2344
validate_network() (*in module ironic.common.neutron*), 2591
validate_network_data() (*in module ironic.api.controllers.v1.node*), 2236
validate_network_port() (*in module ironic.common.utils*), 2694
validate_node() (*in module ironic.conductor.deployments*), 2715
validate_port_info() (*in module ironic.common.neutron*), 2593
validate_port_physnet() (*in module ironic.conductor.utils*), 2913
validate_raid_config() (*ironic.drivers.base.RAIDInterface method*), 4277
validate_raid_config() (*ironic.drivers.modules.noop.NoRAID method*), 4099
validate_raid_config() (*ironic.drivers.modules.redfish.raid.RedfishRAID method*), 3702
validate_rescue() (*ironic.drivers.base.BootInterface method*), 4191
validate_rescue() (*ironic.drivers.base.NetworkInterface method*), 4259
validate_rescue() (*ironic.drivers.modules.ilo.boot.IloUefiHttpsBoot method*), 3386
validate_rescue() (*ironic.drivers.modules.ilo.boot.IloVirtualMediaBoot method*), 3396
validate_rescue() (*ironic.drivers.modules.irmc.boot.IRMCVirtualMediaBoo method*), 3516
validate_rescue() (*ironic.drivers.modules.network.neutron.NeutronNetwork*

method), 3612

validate_rescue() (*ironic.drivers.modules.pxe_base.PXEBaseMixin method*), 4139

validate_sort_dir() (*in module ironic.api.controllers.v1.utils*), 2344

validate_user_deploy_steps_and_templates() (*in module ironic.conductor.steps*), 2860

value (*ironic.db.sqlalchemy.models.BIOSSetting attribute*), 3058

value() (*ironic.objects.bios.BIOSSetting property*), 4370

value_power_off (*ironic.drivers.modules.snmp.SNMPDriverAPCMasterSwitch attribute*), 4144

value_power_off (*ironic.drivers.modules.snmp.SNMPDriverAPCMasterSwitchPlus attribute*), 4145

value_power_off (*ironic.drivers.modules.snmp.SNMPDriverAPCRackPDU attribute*), 4146

value_power_off (*ironic.drivers.modules.snmp.SNMPDriverAten attribute*), 4147

value_power_off (*ironic.drivers.modules.snmp.SNMPDriverBaytechMRP27 attribute*), 4151

value_power_off (*ironic.drivers.modules.snmp.SNMPDriverCyberPower attribute*), 4152

value_power_off (*ironic.drivers.modules.snmp.SNMPDriverEatonPower attribute*), 4153

value_power_off (*ironic.drivers.modules.snmp.SNMPDriverTeltronix attribute*), 4156

value_power_off() (*ironic.drivers.modules.snmp.SNMPDriverSimple property*), 4155

value_power_on (*ironic.drivers.modules.snmp.SNMPDriverAPCMasterSwitch attribute*), 4144

value_power_on (*ironic.drivers.modules.snmp.SNMPDriverAPCMasterSwitchPlus attribute*), 4145

value_power_on (*ironic.drivers.modules.snmp.SNMPDriverAPCRackPDU attribute*), 4146

value_power_on (*ironic.drivers.modules.snmp.SNMPDriverAten attribute*), 4147

value_power_on (*ironic.drivers.modules.snmp.SNMPDriverBaytechMRP27 attribute*), 4151

value_power_on (*ironic.drivers.modules.snmp.SNMPDriverCyberPower attribute*), 4152

value_power_on (*ironic.drivers.modules.snmp.SNMPDriverEatonPower attribute*), 4153

value_power_on (*ironic.drivers.modules.snmp.SNMPDriverTeltronix attribute*), 4156

value_power_on() (*ironic.drivers.modules.snmp.SNMPDriverSimple property*), 4155

value_within_timeout() (*in module ironic.conductor.utils*), 2916

VDIInspector (*class in ironic.common.image_format_inspector*), 2515

vendor (*ironic.drivers.base.BareDriver attribute*), 4175

vendor_interface (*ironic.db.sqlalchemy.models.Node attribute*), 3065

vendor_interface() (*ironic.objects.node.Node property*), 4450

vendor_interface() (*ironic.objects.node.NodeCorrectedPowerStatePayload property*), 4459

vendor_interface() (*ironic.objects.node.NodeCRUDPayload property*), 4454

vendor_interface() (*ironic.objects.node.NodePayload property*), 4463

vendor_interface() (*ironic.objects.node.NodeSetPowerStatePayload property*), 4466

vendor_interface() (*ironic.objects.node.NodeSetProvisionStatePayload property*), 4470

vendor_passthru (*ironic.api.controllers.v1.driver.DriversController attribute*), 2180

vendor_passthru (*ironic.api.controllers.v1.node.NodesController attribute*), 2231

vendor_passthru() (*in module ironic.api.controllers.v1.utils*), 2344

`vendor_passthru()` (*ironic.conductor.manager.ConductorManager method*), 2730
`vendor_passthru()` (*ironic.conductor.rpcapi.ConductorAPI method*), 2848
`VendorInterface` (*class in ironic.drivers.base*), 4285
`VendorMetadata` (*class in ironic.drivers.base*), 4289
`VendorPassthru` (*class in ironic.drivers.modules.ilo.vendor*), 3495
`VendorPassthru` (*class in ironic.drivers.modules.ipmitool*), 4043
`VendorPassthruException`, 2490
`VERBS` (*in module ironic.common.states*), 2660
`verify_firmware_update_args()` (*in module ironic.drivers.modules.ilo.firmware_processor*), 3447
`verify_image_checksum()` (*in module ironic.drivers.modules.ilo.common*), 3439
`verify_iscsi_connection()` (*in module ironic.drivers.modules.iscsi_deploy*), 4084
`verify_node_for_deallocation()` (*in module ironic.conductor.allocations*), 2701
`VERIFYING` (*in module ironic.common.states*), 2661
`version`
  `ironic-dbsync command line option`, 1919
`Version` (*class in ironic.api.controllers.base*), 2374
`Version` (*class in ironic.drivers.modules.irmc.packaging_version*), 3551
`version` (*ironic.db.sqlalchemy.models.Allocation attribute*), 3057
`version` (*ironic.db.sqlalchemy.models.BIOSSetting attribute*), 3058
`version` (*ironic.db.sqlalchemy.models.Chassis attribute*), 3058
`version` (*ironic.db.sqlalchemy.models.Conductor attribute*), 3059
`version` (*ironic.db.sqlalchemy.models.ConductorHardwareInterface attribute*), 3060
`version` (*ironic.db.sqlalchemy.models.DeployTemplate attribute*), 3061
`version` (*ironic.db.sqlalchemy.models.DeployTemplateStep attribute*), 3062
`version` (*ironic.db.sqlalchemy.models.IronicBase attribute*), 3062
`version` (*ironic.db.sqlalchemy.models.Node attribute*), 3065
`version` (*ironic.db.sqlalchemy.models.NodeTag attribute*), 3066
`version` (*ironic.db.sqlalchemy.models.NodeTrait attribute*), 3067
`version` (*ironic.db.sqlalchemy.models.Port attribute*), 3068
`version` (*ironic.db.sqlalchemy.models.Portgroup attribute*), 3069
`version` (*ironic.db.sqlalchemy.models.VolumeConnector attribute*), 3070
`version` (*ironic.db.sqlalchemy.models.VolumeTarget attribute*), 3071
`VERSION` (*ironic.objects.allocation.Allocation attribute*), 4341
`VERSION` (*ironic.objects.allocation.AllocationCRUDNotification attribute*), 4352
`VERSION` (*ironic.objects.allocation.AllocationCRUDPayload attribute*), 4352
`VERSION` (*ironic.objects.bios.BIOSSetting attribute*), 4365
`VERSION` (*ironic.objects.bios.BIOSSettingList attribute*), 4371
`VERSION` (*ironic.objects.chassis.Chassis attribute*), 4378
`VERSION` (*ironic.objects.chassis.ChassisCRUDNotification attribute*), 4386
`VERSION` (*ironic.objects.chassis.ChassisCRUDPayload attribute*), 4386
`VERSION` (*ironic.objects.conductor.Conductor attribute*), 4387
`VERSION` (*ironic.objects.deploy_template.DeployTemplate attribute*), 4395
`VERSION` (*ironic.objects.deploy_template.DeployTemplateCRUDN attribute*), 4407
`VERSION` (*ironic.objects.deploy_template.DeployTemplateCRUDP attribute*), 4407
`VERSION` (*ironic.objects.deployment.Deployment attribute*), 4408
`VERSION` (*ironic.objects.node.Node attribute*), 4434
`VERSION` (*ironic.objects.node.NodeConsoleNotification attribute*), 4454
`VERSION` (*ironic.objects.node.NodeCorrectedPowerStateNotificat attribute*), 4456

VERSION (*ironic.objects.node.NodeCorrectedPowerStatePayload attribute*), 4457

VERSION (*ironic.objects.node.NodeCRUDNotification attribute*), 4451

VERSION (*ironic.objects.node.NodeCRUDPayload attribute*), 4452

VERSION (*ironic.objects.node.NodeMaintenanceNotification attribute*), 4460

VERSION (*ironic.objects.node.NodePayload attribute*), 4460

VERSION (*ironic.objects.node.NodeSetPowerStateNotification attribute*), 4463

VERSION (*ironic.objects.node.NodeSetPowerStatePayload attribute*), 4464

VERSION (*ironic.objects.node.NodeSetProvisionStateNotification attribute*), 4467

VERSION (*ironic.objects.node.NodeSetProvisionStatePayload attribute*), 4468

VERSION (*ironic.objects.notification.EventType attribute*), 4471

VERSION (*ironic.objects.notification.NotificationBase attribute*), 4473

VERSION (*ironic.objects.notification.NotificationPayloadBase attribute*), 4474

VERSION (*ironic.objects.notification.NotificationPublisher attribute*), 4475

VERSION (*ironic.objects.port.Port attribute*), 4476

VERSION (*ironic.objects.port.PortCRUDNotification attribute*), 4493

VERSION (*ironic.objects.port.PortCRUDPayload attribute*), 4494

VERSION (*ironic.objects.portgroup.Portgroup attribute*), 4495

VERSION (*ironic.objects.portgroup.PortgroupCRUDNotification attribute*), 4509

VERSION (*ironic.objects.portgroup.PortgroupCRUDPayload attribute*), 4509

VERSION (*ironic.objects.trait.Trait attribute*), 4510

VERSION (*ironic.objects.trait.TraitList attribute*), 4514

VERSION (*ironic.objects.volume_connector.VolumeConnector attribute*), 4519

VERSION (*ironic.objects.volume_connector.VolumeConnectorCRUDNotification attribute*), 4533

VERSION (*ironic.objects.volume_connector.VolumeConnectorCRUDPayload attribute*), 4534

VERSION (*ironic.objects.volume_target.VolumeTarget attribute*), 4534

VERSION (*ironic.objects.volume_target.VolumeTargetCRUDNotification attribute*), 4551

VERSION (*ironic.objects.volume_target.VolumeTargetCRUDPaylo attribute*), 4551

version command line option
  - --help, 1922
  - -h, 1922

version() (*in module ironic.db.migration*), 3193

version() (*in module ironic.db.sqlalchemy.migration*), 3056

version() (*ironic.cmd.dbsync.DBCommand method*), 2399

VERSION_PATTERN (*in module ironic.drivers.modules.irmc.packaging_version*), 3550

VersionSelectorApplication (*class in ironic.api.app*), 2383

VHDInspector (*class in ironic.common.image_format_inspector*), 2516

VHDX_METADATA_TABLE_MAX_SIZE (*ironic.common.image_format_inspector.VHDXInspector attribute*), 2518

VHDXInspector (*class in ironic.common.image_format_inspector*), 2517

vif_attach() (*ironic.conductor.manager.ConductorManager method*), 2730

vif_attach() (*ironic.conductor.rpcapi.ConductorAPI method*), 2852

vif_attach() (*ironic.drivers.base.NetworkInterface method*), 4260

vif_attach() (*ironic.drivers.modules.network.common.Neutro method*), 3582

vif_attach() (*ironic.drivers.modules.network.noop.NoopNetw method*), 3619

vif_detach() (*ironic.conductor.manager.ConductorManager method*), 2730

vif_detach() (*ironic.conductor.rpcapi.ConductorAPI method*), 2854

vif_detach() (*ironic.drivers.base.NetworkInterface method*), 4261

vif_detach() (*ironic.drivers.modules.network.common.Neutro method*), 3585

vif_detach() (*ironic.drivers.modules.network.noop.NoopNetw method*), 3620

vif_list() (*ironic.conductor.manager.ConductorManager method*), 2731

vif_list() (*ironic.conductor.rpcapi.ConductorAPI method*), 2855

vif_list() (*ironic.drivers.base.NetworkInterface*

method), 4262
vif_list() (ironic.drivers.modules.network.common.VIFPortIDMixin method), 3588
vif_list() (ironic.drivers.modules.network.noop.NoopNetwork method), 3621
VifAlreadyAttached, 2490
VifInvalidForAttach, 2491
VifNotAttached, 2491
VIFPortIDMixin (class in ironic.drivers.modules.network.common), 3586
VIRTUAL_DISK_SIZE (ironic.common.image_format_inspector.VHDXInspector attribute), 2518
virtual_size() (ironic.common.image_format_inspector.FileInspector property), 2507
virtual_size() (ironic.common.image_format_inspector.ISOInspector property), 2510
virtual_size() (ironic.common.image_format_inspector.QcowInspector property), 2514
virtual_size() (ironic.common.image_format_inspector.VDIInspector property), 2516
virtual_size() (ironic.common.image_format_inspector.VHDInspector property), 2517
virtual_size() (ironic.common.image_format_inspector.VHDXInspector property), 2519
virtual_size() (ironic.common.image_format_inspector.VMDKInspector property), 2521
VMDKInspector (class in ironic.common.image_format_inspector), 2519
volume_connectors() (ironic.conductor.task_manager.TaskManager property), 2877
volume_create_error_handler() (ironic.drivers.modules.redfish.raid.RedfishRAID method), 3703
volume_id (ironic.db.sqlalchemy.models.VolumeTarget attribute), 3071
volume_id() (ironic.objects.volume_target.VolumeTarget property), 4550
volume_id() (ironic.objects.volume_target.VolumeTargetCRUDPayload property), 4552
volume_targets() (ironic.conductor.task_manager.TaskManager property), 2877
volume_type (ironic.db.sqlalchemy.models.VolumeTarget attribute), 3071
volume_type() (ironic.objects.volume_target.VolumeTarget property), 4550
volume_type() (ironic.objects.volume_target.VolumeTargetCRUDPayload property), 4552
VolumeConnector (class in ironic.db.sqlalchemy.models), 3069
VolumeConnector (class in ironic.objects.volume_connector), 4519
VolumeConnectorAlreadyExists, 2491
VolumeConnectorCRUDNotification (class in ironic.objects.volume_connector), 4532
VolumeConnectorCRUDPayload (class in ironic.objects.volume_connector), 4533
VolumeConnectorNotFound, 2491
VolumeConnectorsController (class in ironic.api.controllers.v1.volume_connector), 2347
VolumeConnectorTypeAndIdAlreadyExists, 2491
VolumeController (class in ironic.api.controllers.v1.volume), 2347
VolumeTarget (class in ironic.db.sqlalchemy.models), 3070
VolumeTarget (class in ironic.objects.volume_target), 4534
VolumeTargetAlreadyExists, 2491
VolumeTargetBootIndexAlreadyExists, 2491
VolumeTargetCRUDNotification (class in ironic.objects.volume_target), 4550
VolumeTargetCRUDPayload (class in ironic.objects.volume_target), 4551
VolumeTargetNotFound, 2492
VolumeTargetsController (class in ironic.api.controllers.v1.volume_target), 2360

W

wait() (ironic.common.wsgi_service.WSGIService method), 2697
wait_for_host_agent() (in module ironic.common.neutron), 2594
wait_for_job_completion() (in module ironic.drivers.modules.drac.job), 3257

`wait_for_port_status()` (*in module ironic.common.neutron*), 2596

`wait_until_get_system_ready()` (*in module ironic.drivers.modules.redfish.utils*), 3717

`WANBOOT` (*in module ironic.common.boot_devices*), 2426

`warn_about_unsafe_shred_parameters()` (*in module ironic.cmd.conductor*), 2398

`WARNING` (*ironic.objects.fields.NotificationLevel attribute*), 4420

`warning()` (*ironic.common.image_format_inspector.TraceDisabled method*), 2515

`wipe_cleaning_internal_info()` (*in module ironic.conductor.utils*), 2917

`wipe_deploy_internal_info()` (*in module ironic.conductor.utils*), 2917

`wipe_internal_info_on_power_off()` (*in module ironic.conductor.utils*), 2918

`wipe_token_and_url()` (*in module ironic.conductor.utils*), 2918

`wrap_ipv6()` (*in module ironic.common.utils*), 2695

`wrapfunc()` (*in module ironic.api.functions*), 2389

`write_image()` (*ironic.drivers.modules.agent.AgentDeployMixin method*), 3758

`write_image()` (*ironic.drivers.modules.ansible.deploy.AnsibleDeploy method*), 3222

`write_image()` (*ironic.drivers.modules.iscsi_deploy.ISCSIDeploy method*), 4063

`write_to_file()` (*in module ironic.common.utils*), 2695

`WSGIService` (*class in ironic.common.wsgi_service*), 2696

X

`XClarityError`, 2492

`XClarityHardware` (*class in ironic.drivers.xclarity*), 4340

`XClarityManagement` (*class in ironic.drivers.modules.xclarity.management*), 3741

`XClarityPower` (*class in ironic.drivers.modules.xclarity.power*), 3746