Blazar Documentation

Release 15.1.0.dev1

OpenStack Foundation

Mar 14, 2025

CONTENTS

Blazar is the *Resource Reservation Service* for OpenStack.

If you are new to Blazar, please start with the *Introduction*.

CHAPTER

ONE

INSTALLATION GUIDE

1.1 Blazar Installation Guide

1.1.1 Installation using DevStack

This section includes instructions for Blazar installation using DevStack. DevStack configures both the host reservation and the instance reservation.

1. Download DevStack:

```
git clone https://opendev.org/openstack/devstack.git
```

2. Create a local.conf file in the devstack directory. You can use the following sample local.conf:

```
[[local|localrc]]
ADMIN_PASSWORD=password
DATABASE_PASSWORD=$ADMIN_PASSWORD
RABBIT_PASSWORD=$ADMIN_PASSWORD
SERVICE_PASSWORD=$ADMIN_PASSWORD
DEST=/opt/stack/
LOGFILE=$DEST/logs/stack.sh.log
HOST_IP=127.0.0.1
GIT_BASE=https://opendev.org/
RECLONE=yes
enable_plugin blazar https://opendev.org/openstack/blazar
```

3. Run DevStack as the stack user:

```
./stack.sh
```

4. Source the admin credentials:

```
. openrc admin admin
```

5. Now you can add hosts to Blazar:

```
blazar host-create hostname
```

1.1.2 Installation without DevStack

This section includes instructions for Blazar installation. You can use the host reservation and the instance reservation once you finish the install guide.

Download all Blazar related repos:

```
git clone https://opendev.org/openstack/blazar
git clone https://opendev.org/openstack/blazar-nova
git clone https://opendev.org/openstack/python-blazarclient
```

Install all these projects to your working environment via:

```
python setup.py install
```

or

```
python setup.py develop
```

Next you need to configure Blazar and Nova. First, generate a blazar.conf sample:

```
cd /path/to/blazar
tox -e genconfig
mv etc/blazar/blazar.conf.sample /etc/blazar/blazar.conf
```

Then edit */etc/blazar/blazar.conf* using the following example:

```
[DEFAULT]
host=<blazar_host>
port=<blazar_port>
os_auth_host=<auth_host>
os_auth_port=<auth_port>
os_auth_protocol=<http, for example>
os_auth_version=v3
os_admin_username=<username>
os_admin_password=<password>
os_admin_project_name=<project_name>
identity_service=<identity_service_name>
os_region_name=<region_name>

[manager]
plugins=physical.host.plugin,virtual.instance.plugin

[keystone_authtoken]
auth_type=<password, for example>
project_domain_name=<project_domain_name>
project_name=<project_name>
user_domain_name=<user_domain_name>
username=<username>
password=<password>
auth_url=<identity_service_url>
```

*os_admin_** flags refer to the Blazar service user. If you do not have this user, create it:

```
openstack user create --password <password> --project <project_name> --email
↪<email-address> <username>
openstack role add --project <project_name> --user <username> <admin_role>
```

Next you need to configure Nova. Please add the following lines to nova.conf file:

```
[filter_scheduler]
available_filters = nova.scheduler.filters.all_filters
available_filters = blazarnova.scheduler.filters.blazar_filter.BlazarFilter
enabled_filters = AvailabilityZoneFilter,ComputeFilter,
↪ComputeCapabilitiesFilter,ImagePropertiesFilter,
↪ServerGroupAntiAffinityFilter,ServerGroupAffinityFilter,SameHostFilter,
↪DifferentHostFilter,BlazarFilter
```

Restart nova-scheduler to use the new configuration file.

Next you need to create a Nova aggregate to use as a free pool for host reservation:

```
openstack aggregate create freepool
```

And we need to create the reservation service in Keystone with its endpoints:

```
openstack service create --name blazar --description "OpenStack Reservation␣
↪Service" reservation
openstack endpoint create --region <region> blazar admin "<auth_protocol>://
↪<blazar_host>:<blazar_port>/v1"
openstack endpoint create --region <region> blazar internal "<auth_protocol>:/
↪/<blazar_host>:<blazar_port>/v1"
openstack endpoint create --region <region> blazar public "<auth_protocol>://
↪<blazar_host>:<blazar_port>/v1"
```

And, finally, we need to create a database for Blazar:

```
mysql -u<user> -p<password> -h<host> -e "DROP DATABASE IF EXISTS blazar;"
mysql -u<user> -p<password> -h<host> -e "CREATE DATABASE blazar CHARACTER SET␣
↪utf8;"
```

Then edit the database section of */etc/blazar/blazar.conf*:

```
[database]
connection=mysql+pymysql://<user>:<password>@<host>/blazar?charset=utf8
```

To start Blazar services use:

```
blazar-api --config-file /etc/blazar/blazar.conf
blazar-manager --config-file /etc/blazar/blazar.conf
```

Now you can use python-blazarclient to communicate with Blazar.

1.1.3 Install Blazar Dashboard

Please see Blazar Dashboard installation guide (external link).

CHAPTER

TWO

CONFIGURATION REFERENCE

2.1 Blazar Configuration Reference

2.1.1 Configuration

Reference

blazar.conf

DEFAULT

executor_thread_pool_size

Type
integer

Default
64

Size of executor thread pool when executor is threading or eventlet.

Table 1: 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

`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.

Warning

This option is deprecated for removal. Its value may be silently ignored in the future.

Reason
The backdoor_port option is deprecated and will be removed in a future release.

`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.

Warning

This option is deprecated for removal. Its value may be silently ignored in the future.

Reason
The backdoor_socket option is deprecated and will be removed in a future release.

`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.

`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 2: 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 3: 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 4: 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.

Warning

This option is deprecated for removal. Its value may be silently ignored in the future.

Reason
This function is known to have bene broken for long time, and depends on the unmaintained library

`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.

`log_color`

Type
boolean

Default
`False`

(Optional) Set the color key according to log levels. This option takes effect only when logging to stderr or stdout is used. This option is ignored if log_config_append is set.

`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 set to 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
[%(global_request_id)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 %(project)s %(domain)s %(system_scope)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=WARN', 'amqplib=WARN', 'boto=WARN', 'qpid=WARN',
'sqlalchemy=WARN', 'suds=INFO', 'oslo.messaging=INFO',
'oslo_messaging=INFO', 'iso8601=WARN', 'requests.packages.
urllib3.connectionpool=WARN', 'urllib3.connectionpool=WARN',
'websocket=WARN', 'requests.packages.urllib3.util.retry=WARN',
'urllib3.util.retry=WARN', 'keystonemiddleware=WARN',
'routes.middleware=WARN', 'stevedore=WARN', 'taskflow=WARN',
'keystoneauth=WARN', 'oslo.cache=INFO', 'oslo_policy=INFO',
'dogpile.core.dogpile=INFO']
```

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`

Valid Values
CRITICAL, ERROR, INFO, WARNING, DEBUG,

Log level name used by rate limiting. 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`

The strategy to use for auth: noauth or keystone.

`port`

Type
integer

Default
`1234`

Minimum Value
0

Maximum Value
65535

Port that will be used to listen on

`enable_v1_api`

Type
boolean

Default
`True`

Deploy the v1 API.

`host`

Type
host address

Default
`0.0.0.0`

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

`log_exchange`

Type
boolean

Default
`False`

Log request/response exchange details: environ, headers and bodies

`cleaning_time`

Type
integer

Default
`0`

Minimum Value
0

The minimum interval [minutes] between the end of a lease and the start of the next lease for the same resource. This interval is used for cleanup.

`os_auth_protocol`

Type
string

Default
`http`

Protocol used to access OpenStack Identity service

`os_auth_host`

Type
host address

Default
`127.0.0.1`

IP or hostname of machine on which OpenStack Identity service is located

`os_auth_port`

Type
string

Default
`5000`

Port of OpenStack Identity service.

`os_auth_prefix`

Type
string

Default
`''`

Prefix of URL to access OpenStack Identity service.

os_admin_username

Type
string

Default
admin

This OpenStack user is used to treat trusts. The user must have admin role in <os_admin_project_name> project.

os_admin_password

Type
string

Default
blazar

Password of the admin user to treat trusts.

os_admin_project_name

Type
string

Default
admin

Name of project where the user is admin.

os_auth_version

Type
string

Default
v3

Blazar uses API v3 to allow trusts using.

os_admin_user_domain_name

Type
string

Default
Default

A domain name the os_admin_username belongs to.

os_admin_project_domain_name

Type
string

Default
Default

A domain name the os_admin_project_name belongs to

`cafile`

Type
string

Default
<None>

Path of the custom CA certificates bundle.

`db_driver`

Type
string

Default
`blazar.db`

Driver to use for database access

`command`

Type
unknown type

Default
<None>

Available commands

`identity_service`

Type
string

Default
`identity`

Identity service to use.

`os_region_name`

Type
string

Default
<None>

Region name of this node. This is used when picking the URL in the service catalog.

Possible values:

- Any string representing region name

`endpoint_type`

Type
string

Default
`internal`

Valid Values
public, admin, internal

Type of the keystone endpoint to use. This endpoint will be looked up in the keystone catalog and should be one of public, internal or admin.

`keystone_client_version`

Type
string

Default
3

Keystoneclient version

api

`api_v2_controllers`

Type
list

Default
`['oshosts', 'leases']`

API extensions to use

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.

`backend`

Type
string

Default
`sqlalchemy`

The back end to use for the database.

`connection`

Type
string

Default
<None>

The SQLAlchemy connection string to use to connect to the database.

`slave_connection`

Type
string

Default
<None>

The SQLAlchemy connection string to use to connect to the slave database.

`asyncio_connection`

Type
string

Default
<None>

The SQLAlchemy asyncio connection string to use to connect to the database.

`asyncio_slave_connection`

Type
string

Default
<None>

The SQLAlchemy asyncio 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_wsrep_sync_wait`

Type
integer

Default
<None>

For Galera only, configure wsrep_sync_wait causality checks on new connections. Default is None, meaning dont configure any setting.

`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.

`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.

`max_retries`

Type
integer

Default
10

Maximum number of database connection retries during startup. Set to -1 to specify an infinite retry count.

`retry_interval`

Type
integer

Default
10

Interval between retries of opening a SQL connection.

`max_overflow`

Type
integer

Default
50

If set, use this value for max_overflow with SQLAlchemy.

`connection_debug`

Type
integer

Default
0

Minimum Value
0

Maximum Value
100

Verbosity of SQL debugging information: 0=None, 100=Everything.

`connection_trace`

Type
boolean

Default
`False`

Add Python stack traces to SQL as comment strings.

`pool_timeout`

Type
integer

Default
<None>

If set, use this value for pool_timeout with SQLAlchemy.

`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&

enforcement

`external_service_base_endpoint`

Type
string

Default
<None>

The URL of the external service API.

`external_service_check_create_endpoint`

Type
string

Default
<None>

Overrides check-create endpoint with another URL.

`external_service_check_update_endpoint`

Type
string

Default
<None>

Overrides check-update endpoint with another URL.

`external_service_on_end_endpoint`

Type
string

Default
<None>

Overrides on-end endpoint with another URL.

`external_service_token`

Type
string

Default
''

Token used for authentication with the external service.

`max_lease_duration`

Type
integer

Default
-1

Maximum lease duration in seconds. If this is set to -1, there is not limit.

`max_lease_duration_exempt_project_ids`

Type
list

Default
[]

Allow list of project ids exempt from filter constraints.

`enabled_filters`

Type
list

Default
[]

List of enabled usage enforcement filters.

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.

`allowed_source_ranges`

Type
list

Default
`[]`

A list of network addresses to limit source ip allowed to access healthcheck information. Any request from ip outside of these network addresses are ignored.

`ignore_proxied_requests`

Type
boolean

Default
`False`

Ignore requests with proxy headers.

`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.

`enable_by_file_paths`

Type
list

Default
[]

Check the presence of files. Used by EnableByFilesHealthcheck plugin.

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 5: 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 6: 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
`True`

(Optional) Use the advanced (eventlet safe) memcached client pool.

`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.

`memcache_sasl_enabled`

Type
boolean

Default
`False`

Enable the SASL(Simple Authentication and Security Layer) if the SASL_enable is true, else disable.

`memcache_username`

Type
string

Default
`''`

the user name for the SASL

`memcache_password`

Type
string

Default
`''`

the username password for SASL

`auth_type`

Type
unknown type

Default
<None>

Authentication type to load

Table 7: Deprecated Variations

Group	Name
keystone_authtoken	auth_plugin

`auth_section`

Type
unknown type

Default
<None>

Config Section from which to load plugin specific options

manager

rpc_topic

Type
string

Default
`blazar.manager`

The topic Blazar uses for blazar-manager messages.

plugins

Type
list

Default
`['dummy.vm.plugin']`

All plugins to use (one for every resource type to support.)

minutes_before_end_lease

Type
integer

Default
`60`

Minimum Value
0

Minutes prior to the end of a lease in which actions like notification and snapshot are taken. If this is set to 0, then these actions are not taken.

event_max_retries

Type
integer

Default
1

Minimum Value
0

Maximum Value
50

Number of times to retry an event action.

notifications

publisher_id

Type
string

Default
`blazar.lease`

Publisher ID for notifications

nova

endpoint_type

Type
string

Default
`internal`

Valid Values
public, admin, internal

Type of the nova endpoint to use. This endpoint will be looked up in the keystone catalog and should be one of public, internal or admin.

nova_client_version

Type
string

Default
`2`

Novaclient version

Table 8: Deprecated Variations

Group	Name
DEFAULT	nova_client_version

compute_service

Type
string

Default
`compute`

Nova name in keystone

Table 9: Deprecated Variations

Group	Name
DEFAULT	compute_service

image_prefix

Type
string

Default
`reserved_`

Prefix for VM images if you want to create snapshots

Table 10: Deprecated Variations

Group	Name
DEFAULT	image_prefix

aggregate_freepool_name

Type
string

Default
`freepool`

Name of the special aggregate where all hosts are candidate for physical host reservation

Table 11: Deprecated Variations

Group	Name
physical:host	aggregate_freepool_name

project_id_key

Type
string

Default
`blazar:project`

Aggregate metadata value for key matching project_id

Table 12: Deprecated Variations

Group	Name
physical:host	project_id_key

blazar_owner

Type
string

Default
`blazar:owner`

Aggregate metadata key for knowing owner project_id

Table 13: Deprecated Variations

Group	Name
physical:host	blazar_owner

`az_aware`

Type
boolean

Default
`True`

A flag to store original availability zone

oslo_concurrency

`disable_process_locking`

Type
boolean

Default
`False`

Enables or disables inter-process locks.

`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.

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

`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

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.

topics

Type
list

Default
['notifications']

AMQP topic used for OpenStack notifications.

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. If rabbit_quorum_queue is enabled, queues will be durable and this value will be ignored.

`amqp_auto_delete`

Type
boolean

Default
`False`

Auto-delete queues in AMQP.

`rpc_conn_pool_size`

Type
integer

Default
`30`

Minimum Value
1

Size of RPC connection pool.

`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

`ssl`

Type
boolean

Default
`False`

Connect over 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.

`ssl_key_file`

Type
string

Default
''

SSL key file (valid only if SSL enabled).

`ssl_cert_file`

Type
string

Default
''

SSL cert file (valid only if SSL enabled).

`ssl_ca_file`

Type
string

Default
''

SSL certification authority file (valid only if SSL enabled).

`ssl_enforce_fips_mode`

Type
boolean

Default
`False`

Global toggle for enforcing the OpenSSL FIPS mode. This feature requires Python support. This is available in Python 3.9 in all environments and may have been backported to older Python versions on select environments. If the Python executable used does not support OpenSSL FIPS mode, an exception will be raised.

`heartbeat_in_pthread`

Type
boolean

Default
`False`

(DEPRECATED) It is recommend not to use this option anymore. 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. This option should be set to True only for the wsgi services.

Warning

This option is deprecated for removal. Its value may be silently ignored in the future.

Reason
The option is related to Eventlet which will be removed. In addition this has never worked as expected with services using eventlet for core service framework.

`kombu_reconnect_delay`

Type
floating point

Default
`1.0`

Minimum Value
0.0

Maximum Value
4.5

How long to wait (in seconds) before reconnecting in response to an AMQP consumer cancel notification.

`kombu_reconnect_splay`

Type
floating point

Default
`0.0`

Minimum Value
0.0

Random time to wait for when reconnecting in response to an AMQP consumer cancel notification.

`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 14: 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, EXTERNAL, RABBIT-CR-DEMO

The RabbitMQ login method.

rabbit_retry_interval

Type
integer

Default
1

Minimum Value
1

How frequently to retry connecting with RabbitMQ.

rabbit_retry_backoff

Type
integer

Default
2

Minimum Value
0

How long to backoff for between retries when connecting to RabbitMQ.

`rabbit_interval_max`

Type
integer

Default
`30`

Minimum Value
1

Maximum interval of RabbitMQ connection retries.

`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}

`rabbit_quorum_queue`

Type
boolean

Default
`False`

Use quorum queues in RabbitMQ (x-queue-type: quorum). The quorum queue is a modern queue type for RabbitMQ implementing a durable, replicated FIFO queue based on the Raft consensus algorithm. It is available as of RabbitMQ 3.8.0. If set this option will conflict with the HA queues (`rabbit_ha_queues`) aka mirrored queues, in other words the HA queues should be disabled. Quorum queues are also durable by default so the amqp_durable_queues option is ignored when this option is enabled.

`rabbit_transient_quorum_queue`

Type
boolean

Default
`False`

Use quorum queues for transients queues in RabbitMQ. Enabling this option will then make sure those queues are also using quorum kind of rabbit queues, which are HA by default.

`rabbit_quorum_delivery_limit`

Type
integer

Default
0

Each time a message is redelivered to a consumer, a counter is incremented. Once the redelivery count exceeds the delivery limit the message gets dropped or dead-lettered (if a DLX exchange has been configured) Used only when rabbit_quorum_queue is enabled, Default 0 which means dont set a limit.

`rabbit_quorum_max_memory_length`

Type
integer

Default
0

By default all messages are maintained in memory if a quorum queue grows in length it can put memory pressure on a cluster. This option can limit the number of messages in the quorum queue. Used only when rabbit_quorum_queue is enabled, Default 0 which means dont set a limit.

Table 15: Deprecated Variations

Group	Name
oslo_messaging_rabbit	rabbit_quroum_max_memory_length

`rabbit_quorum_max_memory_bytes`

Type
integer

Default
0

By default all messages are maintained in memory if a quorum queue grows in length it can put memory pressure on a cluster. This option can limit the number of memory bytes used by the quorum queue. Used only when rabbit_quorum_queue is enabled, Default 0 which means dont set a limit.

Table 16: Deprecated Variations

Group	Name
oslo_messaging_rabbit	rabbit_quroum_max_memory_bytes

`rabbit_transient_queues_ttl`

Type
integer

Default
1800

Minimum Value
0

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. Setting 0 as value will disable the x-expires. If doing so, make sure you have a rabbitmq policy to delete the queues or you deployment will create an infinite number of queue over time.In case rabbit_stream_fanout is set to True, this option will control data retention policy (x-max-age) for messages in the fanout queue rather then the queue duration itself. So the oldest data in the stream queue will be discarded from it once reaching TTL Setting to 0 will disable x-max-age for stream which make stream grow indefinitely filling up the diskspace

`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
`3`

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

`use_queue_manager`

Type
boolean

Default
`False`

Should we use consistant queue names or random ones

`hostname`

Type
string

Default
`node1.example.com`

This option has a sample default set, which means that its actual default value may vary from the one documented above.

Hostname used by queue manager. Defaults to the value returned by socket.gethostname().

`processname`

Type
string

Default
`nova-api`

This option has a sample default set, which means that its actual default value may vary from the one documented above.

Process name used by queue manager

`rabbit_stream_fanout`

Type
boolean

Default
`False`

Use stream queues in RabbitMQ (x-queue-type: stream). Streams are a new persistent and replicated data structure (queue type) in RabbitMQ which models an append-only log with non-destructive consumer semantics. It is available as of RabbitMQ 3.9.0. If set this option will replace all fanout queues with only one stream queue.

oslo_middleware

max_request_body_size

Type
integer

Default
`114688`

The maximum body size for each request, in bytes.

Table 17: Deprecated Variations

Group	Name
DEFAULT	osapi_max_request_body_size
DEFAULT	max_request_body_size

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.

http_basic_auth_user_file

Type
string

Default
`/etc/htpasswd`

HTTP basic auth password file.

oslo_policy

enforce_scope

Type
boolean

Default
`True`

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.

Warning

This option is deprecated for removal. Its value may be silently ignored in the future.

Reason
This configuration was added temporarily to facilitate a smooth transition to the new RBAC. OpenStack will always enforce scope checks. This configuration option is deprecated and will be removed in the 2025.2 cycle.

`enforce_new_defaults`

Type
boolean

Default
`True`

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. If `False`, the deprecated policy check string is logically ORd with the new policy check string, allowing for a graceful upgrade experience between releases with new policies, which is the default behavior.

`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.

`policy_default_rule`

Type
string

Default
`default`

Default rule. Enforced when a requested rule is not found.

`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.

`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

`remote_timeout`

Type
floating point

Default
`60`

Minimum Value
0

Timeout in seconds for REST based policy check

physical:host

`blazar_az_prefix`

Type
string

Default
`blazar_`

Prefix for Availability Zones created by Blazar

`before_end`

Type
string

Default
`''`

Actions which we will be taken before the end of the lease

`enable_notification_monitor`

Type
boolean

Default
`False`

Enable notification-based resource monitoring. If it is enabled, the blazar-manager monitors states of compute hosts by subscribing to notifications of Nova.

`notification_topics`

Type
list

Default
`['notifications', 'versioned_notifications']`

Notification topics to subscribe to.

`enable_polling_monitor`

Type
boolean

Default
`False`

Enable polling-based resource monitoring. If it is enabled, the blazar-manager monitors states of compute hosts by polling the Nova API.

`polling_interval`

Type
integer

Default
`60`

Minimum Value
1

Interval (seconds) of polling for health checking.

`healing_interval`

Type
integer

Default
`60`

Minimum Value
0

Interval (minutes) of reservation healing. If 0 is specified, the interval is infinite and all the reservations in the future is healed at one time.

`randomize_host_selection`

Type
boolean

Default
`False`

Allocate hosts for reservations randomly.

nova.conf

Please add the following lines to the nova.conf configuration file:

```
[filter_scheduler]
available_filters = nova.scheduler.filters.all_filters
available_filters = blazarnova.scheduler.filters.blazar_filter.BlazarFilter
enabled_filters = AvailabilityZoneFilter,ComputeFilter,
↪ComputeCapabilitiesFilter,ImagePropertiesFilter,
↪ServerGroupAntiAffinityFilter,ServerGroupAffinityFilter,SameHostFilter,
↪DifferentHostFilter,BlazarFilter
```

2.1.2 Policy

Reference

Policies

Warning

Using a JSON-formatted policy file is deprecated since Blazar 7.0.0 (Wallaby). This oslopolicy-convert-json-to-yaml tool will migrate your existing JSON-formatted policy file to YAML in a backward-compatible way.

The following is an overview of all available policies in Blazar. For a sample configuration file, refer to *Sample Policy File*.

To change policies, please create a policy file in */etc/blazar/* and specify the policy file name at the *oslo_policy/policy_file* option in *blazar.conf*.

blazar

admin

Default
`is_admin:True or role:admin`

Default rule for most Admin APIs.

admin_or_owner

Default
`rule:admin or project_id:%(project_id)s`

Default rule for most non-Admin APIs.

project_member_api

Default
`role:member and project_id:%(project_id)s`

Default rule for Project Member (non-Admin) APIs.

project_reader_api

Default
`role:reader and project_id:%(project_id)s`

Default rule for Project Reader (read-only) APIs.

project_member_or_admin

Default
`rule:project_member_api or rule:admin`

Default rule for Project Member or Admin APIs.

project_reader_or_admin

Default
`rule:project_reader_api or rule:admin`

Default rule for Project Reader or Admin APIs.

blazar:leases:get

Default
`rule:project_reader_or_admin`

Operations

- **GET** `/{api_version}/leases`
- **GET** `/{api_version}/leases/{lease_id}`

Scope Types

- **project**

Policy rule for List/Show Lease(s) API.

`blazar:leases:post`

Default
`rule:project_member_or_admin`

Operations

- **POST** `/{api_version}/leases`

Scope Types

- **project**

Policy rule for Create Lease API.

`blazar:leases:put`

Default
`rule:project_member_or_admin`

Operations

- **PUT** `/{api_version}/leases/{lease_id}`

Scope Types

- **project**

Policy rule for Update Lease API.

`blazar:leases:delete`

Default
`rule:project_member_or_admin`

Operations

- **DELETE** `/{api_version}/leases/{lease_id}`

Scope Types

- **project**

Policy rule for Delete Lease API.

`blazar:oshosts:get`

Default
`rule:admin`

Operations

- **GET** /{api_version}/os-hosts
- **GET** /{api_version}/os-hosts/{host_id}

Scope Types

- **project**

Policy rule for List/Show Host(s) API.

`blazar:oshosts:post`

Default
rule:admin

Operations

- **POST** /{api_version}/os-hosts

Scope Types

- **project**

Policy rule for Create Host API.

`blazar:oshosts:put`

Default
rule:admin

Operations

- **PUT** /{api_version}/os-hosts/{host_id}

Scope Types

- **project**

Policy rule for Update Host API.

`blazar:oshosts:delete`

Default
rule:admin

Operations

- **DELETE** /{api_version}/os-hosts/{host_id}

Scope Types

- **project**

Policy rule for Delete Host API.

`blazar:oshosts:get_allocations`

Default
rule:admin

Operations

- **GET** /{api_version}/os-hosts/allocations
- **GET** /{api_version}/os-hosts/{host_id}/allocation

Scope Types

- **project**

Policy rule for List/Get Host(s) Allocations API.

`blazar:oshosts:get_resource_properties`

Default
`rule:admin`

Operations

- **GET** `/{api_version}/os-hosts/resource_properties`

Scope Types

- **project**

Policy rule for Resource Properties API.

`blazar:oshosts:update_resource_properties`

Default
`rule:admin`

Operations

- **PATCH** `/{api_version}/os-hosts/resource_properties/{property_name}`

Scope Types

- **project**

Policy rule for Resource Properties API.

`blazar:floatingips:get`

Default
`rule:project_reader_or_admin`

Operations

- **GET** `/{api_version}/floatingips`
- **GET** `/{api_version}/floatingips/{floatingip_id}`

Scope Types

- **project**

Policy rule for List/Show FloatingIP(s) API.

`blazar:floatingips:post`

Default
`rule:admin`

Operations

- **POST** `/{api_version}/floatingips`

Scope Types

- **project**

Policy rule for Create Floating IP API.

`blazar:floatingips:delete`

Default
`rule:admin`

Operations

- **DELETE** `/{api_version}/floatingips/{floatingip_id}`

Scope Types

- **project**

Policy rule for Delete Floating IP API.

CHAPTER

THREE

CLI REFERENCE

3.1 Command-Line Interface Reference

3.1.1 Host Reservation

Prerequisites

The following packages should be installed:

- blazar
- blazar-nova
- python-blazarclient

1. Add hosts into the freepool

1. Add hosts into the Blazar freepool using the host-create command:

```
# Using the blazar CLI
blazar host-create compute-1

# Using the openstack CLI
openstack reservation host create compute-1
```

2. Check hosts in the freepool:

```
# Using the blazar CLI
blazar host-list

# Using the openstack CLI
openstack reservation host list
```

Result:

```
+----+---------------------+-------+----------+----------+
| id | hypervisor_hostname | vcpus | memory_mb | local_gb |
+----+---------------------+-------+----------+----------+
| 1  | compute-1           |     2 |      3951 |       38 |
+----+---------------------+-------+----------+----------+
```

3. (Optional) Add extra capabilities to host to add other properties. These can be used to filter hosts when creating a reservation.

```
# Using the blazar CLI
blazar host-update --extra gpu=True compute-1

# Using the openstack CLI
openstack reservation host set --extra gpu=True compute-1
```

Result:

```
Updated host: compute-1
```

Multiple `--extra` parameters can be included. They can also be specified in `host-create`. Properties can be made private or public. By default, they are public.

```
# Using the blazar CLI
blazar host-capability-update gpu --private

# Using the openstack CLI
openstack reservation host capability update gpu --private
```

Result:

```
Updated host extra capability: gpu
```

2. Create a lease

1. Create a lease (compute host reservation) using lease-create command:

```
# Using the blazar CLI
blazar lease-create --physical-reservation min=1,max=1,hypervisor_properties=
↪'[">=", "$vcpus", "2"]' --start-date "2020-06-08 12:00" --end-date "2020-06-
↪09 12:00" lease-1

# Using the openstack CLI
openstack reservation lease create --reservation resource_type=physical:host,
↪min=1,max=1,hypervisor_properties='[">=", "$vcpus", "2"]' --start-date
↪"2020-06-08 12:00" --end-date "2020-06-09 12:00" lease-1
```

Note

The `--physical-reservation` flag is not available in the openstack client, instead use `--reservation resource_type=physical:host` as shown above.

Result:

```
+---------------+-------------------------------------------------------------
↪-------------------------------------------------------------------------
↪----+
| Field         | Value                                                       
↪                                                                           
↪    |
```

(continues on next page)

(continued from previous page)

```
+--------------+-------------------------------------------------------------------
↪-------------------------------------------------------------------------------
↪----+
| action       |                                                                  ␣
↪                                                                               ␣
↪    |
| created_at   | 2020-06-08 02:43:40                                              ␣
↪                                                                               ␣
↪    |
| end_date     | 2020-06-09T12:00:00.000000                                       ␣
↪                                                                               ␣
↪    |
| events       | {"status": "UNDONE", "lease_id": "6638c31e-f6c8-4982-9b98-
↪d2ca0a8cb646", "event_type": "before_end_lease", "created_at": "2020-06-08 ␣
↪      |
|              | 02:43:40", "updated_at": null, "time": "2020-06-08T12:00:00.
↪000000", "id": "420caf25-dba5-4ac3-b377-50503ea5c886"}                          ␣
↪    |
|              | {"status": "UNDONE", "lease_id": "6638c31e-f6c8-4982-9b98-
↪d2ca0a8cb646", "event_type": "start_lease", "created_at": "2020-06-08␣
↪02:43:40",  |
|              | "updated_at": null, "time": "2020-06-08T12:00:00.000000",
↪"id": "b9696139-55a1-472d-baff-5fade2c15243"}                                   ␣
↪      |
|              | {"status": "UNDONE", "lease_id": "6638c31e-f6c8-4982-9b98-
↪d2ca0a8cb646", "event_type": "end_lease", "created_at": "2020-06-08 02:43:40
↪",    |
|              | "updated_at": null, "time": "2020-06-09T12:00:00.000000",
↪"id": "ff9e6f52-db50-475a-81f1-e6897fdc769d"}                                   ␣
↪      |
| id           | 6638c31e-f6c8-4982-9b98-d2ca0a8cb646                             ␣
↪                                                                               ␣
↪    |
| name         | lease-1                                                          ␣
↪                                                                               ␣
↪    |
| project_id   | 4527fa2138564bd4933887526d01bc95                                 ␣
↪                                                                               ␣
↪    |
| reservations | {"status": "pending", "lease_id": "6638c31e-f6c8-4982-9b98-
↪d2ca0a8cb646", "resource_id": "8", "max": 1, "created_at": "2020-06-08     ␣
↪     |
|              | 02:43:40", "min": 1, "updated_at": null, "hypervisor_
↪properties": "[\">=\", \"$vcpus\", \"2\"]", "resource_properties": "", "id
↪":          |
|              | "4d3dd68f-0e3f-4f6b-bef7-617525c74ccb", "resource_type":
↪"physical:host"}                                                                ␣
↪      |
| start_date   | 2020-06-08T12:00:00.000000                                       ␣
```

(continues on next page)

(continued from previous page)

```
↪                                                                                ␣
↪    |
| status         |                                                               ␣
↪                                                                                ␣
↪    |
| status_reason |                                                                ␣
↪                                                                                ␣
↪    |
| trust_id       | ba4c321878d84d839488216de0a9e945                              ␣
↪                                                                                ␣
↪    |
| updated_at     |                                                               ␣
↪                                                                                ␣
↪    |
| user_id        |                                                               ␣
↪                                                                                ␣
↪    |
+---------------+-------------------------------------------------------------
↪-------------------------------------------------------------------------------
↪----+
```

2. Check leases:

```
# Using the blazar CLI
blazar lease-list

# Using the openstack CLI
openstack reservation lease list
```

Result:

```
+--------------------------------------+---------+---------------------------
↪+---------------------------+
| id                                   | name    | start_date                ␣
↪| end_date                  |
+--------------------------------------+---------+---------------------------
↪+---------------------------+
| 6638c31e-f6c8-4982-9b98-d2ca0a8cb646 | lease-1 | 2020-06-08T12:00:00.000000␣
↪| 2020-06-09T12:00:00.000000 |
+--------------------------------------+---------+---------------------------
↪+---------------------------+
```

3. Alternatively, create leases with resource properties. First list properties.

```
# Using the blazar CLI
blazar host-capability-list

# Using the openstack CLI
openstack reservation host capability list
```

Result:

```
+----------+
| property |
+----------+
| gpu      |
+----------+
```

List possible values for a property

```
# Using the blazar CLI
blazar host-capability-show gpu

# Using the openstack CLI
openstack reservation host capability show gpu
```

Result:

```
+-------------------+-------+
| Field             | Value |
+-------------------+-------+
| capability_values | True  |
|                   | False |
| private           | False |
| property          | gpu   |
+-------------------+-------+
```

Create a lease.

```
# Using the blazar CLI
blazar lease-create --physical-reservation min=1,max=1,resource_properties='[
↪"=", "$gpu", "True"]' --start-date "2020-06-08 12:00" --end-date "2020-06-
↪09 12:00" lease-1

# Using the openstack CLI
openstack reservation lease create --reservation resource_type=physical:host,
↪min=1,max=1,resource_properties='[">=", "$vcpus", "2"]' --start-date "2020-
↪06-08 12:00" --end-date "2020-06-09 12:00" lease-1
```

Result:

```
+--------------+-------------------------------------------------------------
↪-------------------------------------------------------------------------
↪----+
| Field        | Value                                                       
↪                                                                          
↪    |
+--------------+-------------------------------------------------------------
↪-------------------------------------------------------------------------
↪----+
| action       |                                                             
↪                                                                          
↪    |
```

(continues on next page)

(continued from previous page)

```
| created_at    | 2020-06-08 02:43:40                                              ␣
↪                                                                                ␣
↪    |
| end_date      | 2020-06-09T12:00:00.000000                                       ␣
↪                                                                                ␣
↪    |
| events        | {"status": "UNDONE", "lease_id": "6638c31e-f6c8-4982-9b98-
↪d2ca0a8cb646", "event_type": "before_end_lease", "created_at": "2020-06-08 ␣
↪      |
|               | 02:43:40", "updated_at": null, "time": "2020-06-08T12:00:00.
↪000000", "id": "420caf25-dba5-4ac3-b377-50503ea5c886"}                         ␣
↪    |
|               | {"status": "UNDONE", "lease_id": "6638c31e-f6c8-4982-9b98-
↪d2ca0a8cb646", "event_type": "start_lease", "created_at": "2020-06-08␣
↪02:43:40",  |
|               | "updated_at": null, "time": "2020-06-08T12:00:00.000000",
↪"id": "b9696139-55a1-472d-baff-5fade2c15243"}                                  ␣
↪      |
|               | {"status": "UNDONE", "lease_id": "6638c31e-f6c8-4982-9b98-
↪d2ca0a8cb646", "event_type": "end_lease", "created_at": "2020-06-08 02:43:40
↪",     |
|               | "updated_at": null, "time": "2020-06-09T12:00:00.000000",
↪"id": "ff9e6f52-db50-475a-81f1-e6897fdc769d"}                                  ␣
↪      |
| id            | 6638c31e-f6c8-4982-9b98-d2ca0a8cb646                             ␣
↪                                                                                ␣
↪    |
| name          | lease-1                                                          ␣
↪                                                                                ␣
↪    |
| project_id    | 4527fa2138564bd4933887526d01bc95                                 ␣
↪                                                                                ␣
↪    |
| reservations  | {"status": "pending", "lease_id": "6638c31e-f6c8-4982-9b98-
↪d2ca0a8cb646", "resource_id": "8", "max": 1, "created_at": "2020-06-08     ␣
↪     |
|               | 02:43:40", "min": 1, "updated_at": null, "hypervisor_
↪properties": "", "resource_properties": "[\"=\", \"$gpu\", \"True\"]", "id
↪":           |
|               | "4d3dd68f-0e3f-4f6b-bef7-617525c74ccb", "resource_type":
↪"physical:host"}                                                               ␣
↪       |
| start_date    | 2020-06-08T12:00:00.000000                                       ␣
↪                                                                                ␣
↪    |
| status        |                                                                  ␣
↪                                                                                ␣
↪    |
| status_reason |                                                                  ␣
```

(continues on next page)

(continued from previous page)

```
↪                                                                              ␣
↪    |
| trust_id      | ba4c321878d84d839488216de0a9e945                              ␣
↪                                                                              ␣
↪    |
| updated_at    |                                                              ␣
↪                                                                              ␣
↪    |
| user_id       |                                                              ␣
↪                                                                              ␣
↪    |
+---------------+------------------------------------------------------------------
↪----------------------------------------------------------------------------
↪----+
```

3. Use the leased resources

1. Create a server: Please specify the reservation id as a scheduler hint.

```
openstack server create --flavor <flavor> --image <image> --network <network>␣
↪--hint reservation=4d3dd68f-0e3f-4f6b-bef7-617525c74ccb <server-name>
```

3.1.2 Instance Reservation

Prerequisites

The following packages should be installed:

- blazar
- blazar-nova
- python-blazarclient

1. Add hosts into the freepool

1. Add hosts into the Blazar freepool using the host-create command:

```
# Using the blazar CLI
blazar host-create compute-1

# Using the openstack CLI
openstack reservation host create compute-1
```

2. Check hosts in the freepool:

```
# Using the blazar CLI
blazar host-list

# Using the openstack CLI
openstack reservation host list
```

Result:

```
+----+---------------------+-------+----------+----------+
| id | hypervisor_hostname | vcpus | memory_mb | local_gb |
+----+---------------------+-------+----------+----------+
| 1  | compute-1           |     2 |      3951 |       38 |
+----+---------------------+-------+----------+----------+
```

2. Create a lease

1. Create a lease (instance reservation) using lease-create command:

```
# Using the blazar CLI
blazar lease-create --reservation resource_type=virtual:instance,vcpus=1,
↪memory_mb=1024,disk_gb=20,amount=1 --start-date "2020-07-24 20:00" --end-
↪date "2020-08-09 21:00" lease-1

# Using the openstack CLI
openstack reservation lease create --reservation resource_
↪type=virtual:instance,vcpus=1,memory_mb=1024,disk_gb=20,amount=1 --start-
↪date "2020-07-24 20:00" --end-date "2020-08-09 21:00" lease-1
```

Result:

```
+--------------+------------------------------------------------------------
↪-----------------------------------------------------------+
| Field        | Value                                                      ␣
↪                                                          |
+--------------+------------------------------------------------------------
↪-----------------------------------------------------------+
| action       |                                                            ␣
↪                                                          |
| created_at   | 2017-07-31 07:55:59                                        ␣
↪                                                          |
| end_date     | 2020-08-09T21:00:00.000000                                 ␣
↪                                                          |
| events       | {"status": "UNDONE", "lease_id": "becf2f3b-0177-4c0f-a7e7-
↪0123370849a3", "event_type": "end_lease", "created_at":        |
|              | "2017-07-31 07:55:59", "updated_at": null, "time": "2020-08-
↪09T21:00:00.000000", "id": "0f269526-c32d-4e53-bc6b-          |
|              | 09fb7adf4354"}                                             ␣
↪                                                          |
|              | {"status": "UNDONE", "lease_id": "becf2f3b-0177-4c0f-a7e7-
↪0123370849a3", "event_type": "start_lease", "created_at":      |
|              | "2017-07-31 07:55:59", "updated_at": null, "time": "2020-07-
↪24T20:00:00.000000", "id": "7dbf3904-7d23-4db3-bfbd-          |
|              | 5cc8cb9d4d92"}                                             ␣
↪                                                          |
|              | {"status": "UNDONE", "lease_id": "becf2f3b-0177-4c0f-a7e7-
↪0123370849a3", "event_type": "before_end_lease", "created_at": |
|              | "2017-07-31 07:55:59", "updated_at": null, "time": "2020-08-
↪07T21:00:00.000000", "id": "f16151d4-04b4-403c-               |
```

(continues on next page)

(continued from previous page)

```
|               | b0d7-f60d3810e37e"}                                                  ␣
↪                                                                                |
| id            | becf2f3b-0177-4c0f-a7e7-0123370849a3                                 ␣
↪                                                                                |
| name          | lease-1                                                              ␣
↪                                                                                |
| project_id    | 6f6f9b596d47441294eb40f565063833                                     ␣
↪                                                                                |
| reservations  | {"status": "pending", "memory_mb": 1024, "lease_id":
↪"becf2f3b-0177-4c0f-a7e7-0123370849a3", "disk_gb": 20,                 |
|               | "resource_id": "061198b0-53e4-4545-9d85-405ca93a7bdf",
↪"created_at": "2017-07-31 07:55:59", "updated_at": "2017-07-31    |
|               | 07:55:59", "aggregate_id": 3, "server_group_id": "ba03ebb4-
↪e55c-4da4-9d39-87e13354f3b7", "amount": 1, "affinity": null,  |
|               | "flavor_id": "db83d6fd-c69c-4259-92cf-012db2e55a58", "id":
↪"db83d6fd-c69c-4259-92cf-012db2e55a58", "vcpus": 1,            |
|               | "resource_type": "virtual:instance"}                                 ␣
↪                                                                                |
| start_date    | 2020-07-24T20:00:00.000000                                           ␣
↪                                                                                |
| status        |                                                                      ␣
↪                                                                                |
| status_reason |                                                                      ␣
↪                                                                                |
| trust_id      | 65da707498914c7992ee7170647a3472                                     ␣
↪                                                                                |
| updated_at    |                                                                      ␣
↪                                                                                |
| user_id       |                                                                      ␣
↪                                                                                |
+---------------+------------------------------------------------------------------
↪-----------------------------------------------------------+
```

2. Check leases:

```
# Using the blazar CLI
blazar lease-list

# Using the openstack CLI
openstack reservation lease list
```

Result:

```
+--------------------------------------+---------+-----------------------------
↪+----------------------------+
| id                                   | name    | start_date                 ␣
↪| end_date                   |
+--------------------------------------+---------+-----------------------------
↪+----------------------------+
| becf2f3b-0177-4c0f-a7e7-0123370849a3 | lease-1 | 2020-07-24T20:00:00.000000␣
```

(continues on next page)

(continued from previous page)

```
↪| 2020-08-09T21:00:00.000000 |
+--------------------------------------+---------+----------------------------
↪+----------------------------+
```

3. Use the leased resources

While the reservation you created is active you can see and use the flavor of your reservation.

```
openstack flavor list
```

Result:

```
+--------------------------------------+-----------------------------------------
↪-----------+----------+------+-----------+------+-------+-------------+---
↪--------+
| ID                                   | Name                                    ␣
↪            | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor |␣
↪Is_Public |
+--------------------------------------+-----------------------------------------
↪-----------+----------+------+-----------+------+-------+-------------+---
↪--------+
| 1                                    | m1.tiny                                 ␣
↪            | 512       | 1    | 0         |      | 1     | 1.0         |␣
↪True      |
| 2                                    | m1.small                                ␣
↪            | 2048      | 20   | 0         |      | 1     | 1.0         |␣
↪True      |
| 3                                    | m1.medium                               ␣
↪            | 4096      | 40   | 0         |      | 2     | 1.0         |␣
↪True      |
| 4                                    | m1.large                                ␣
↪            | 8192      | 80   | 0         |      | 4     | 1.0         |␣
↪True      |
| 5                                    | m1.xlarge                               ␣
↪            | 16384     | 160  | 0         |      | 8     | 1.0         |␣
↪True      |
| c1                                   | cirros256                               ␣
↪            | 256       | 0    | 0         |      | 1     | 1.0         |␣
↪True      |
| d1                                   | ds512M                                  ␣
↪            | 512       | 5    | 0         |      | 1     | 1.0         |␣
↪True      |
| d2                                   | ds1G                                    ␣
↪            | 1024      | 10   | 0         |      | 1     | 1.0         |␣
↪True      |
| d3                                   | ds2G                                    ␣
↪            | 2048      | 10   | 0         |      | 2     | 1.0         |␣
↪True      |
| d4                                   | ds4G                                    ␣
↪            | 4096      | 20   | 0         |      | 4     | 1.0         |␣
```

(continues on next page)

(continued from previous page)

```
↪True      |
| db83d6fd-c69c-4259-92cf-012db2e55a58 | reservation:db83d6fd-c69c-4259-92cf-
↪012db2e55a58 | 1024      | 20    | 0          |       | 1     | 1.0         |␣
↪False     |
+--------------------------------------+--------------------------------------
↪-----------+----------+------+----------+------+-------+-------------+---
↪--------+
```

1. Create a server: Please specify the flavor of the reservation.

```
openstack server create --flavor db83d6fd-c69c-4259-92cf-012db2e55a58 --image
↪<image> --network <network> <server-name>
```

Affinity

A lease can be created with the optional `--affinity` parameter. This provides the following behavior:

- `affinity=True`: Instances will be deployed on the same host for this reservation, by adding them to a server group with an affinity policy.
- `affinity=False`: Instances will be deployed on different hosts for this reservation, by adding them to a server group with an anti-affinity policy.
- `affinity=None` (default): Instances can be deployed on any host, regardless of other instances in this reservation. No server group is created.

3.1.3 Flavor-based Instance Reservation

Prerequisites

The following packages should be installed:

- blazar
- blazar-nova
- python-blazarclient

1. Add hosts into the freepool

1. Add hosts into the Blazar freepool using the host-create command:

```
# Using the blazar CLI
blazar host-create compute-1

# Using the openstack CLI
openstack reservation host create compute-1
```

2. Check hosts in the freepool:

```
# Using the blazar CLI
blazar host-list
```

(continues on next page)

(continued from previous page)

```
# Using the openstack CLI
openstack reservation host list
```

Result:

```
+----+---------------------+-------+----------+----------+
| id | hypervisor_hostname | vcpus | memory_mb | local_gb |
+----+---------------------+-------+----------+----------+
| 1  | compute-1           |     2 |      3951 |       38 |
+----+---------------------+-------+----------+----------+
```

2. Create a lease

1. Create a lease (instance reservation) using lease-create command:

```
# Using the blazar CLI
blazar lease-create --reservation resource_type=flavor:instance,flavor_id=3,
↪amount=1 --start-date "2024-08-23 12:00" --end-date "2024-09-08 13:00"␣
↪lease-1

# Using the openstack CLI
openstack reservation lease create --reservation resource_
↪type=flavor:instance,flavor_id=3,amount=1 --start-date "2024-08-23 14:00" --
↪end-date "2024-09-08 15:00" lease-1
```

Result:

```
+--------------+--------------------------------------------------------------------
↪-----------------------------------------------------------------------+
| Field        | Value                                                                 ␣
↪                                                                      |
+--------------+--------------------------------------------------------------------
↪-----------------------------------------------------------------------+
| created_at   | 2024-08-23 12:40:53                                                   ␣
↪                                                                      |
| degraded     | False                                                                 ␣
↪                                                                      |
| end_date     | 2024-09-08T15:00:00.000000                                            ␣
↪                                                                      |
| events       | {                                                                     ␣
↪                                                                      |
|              |     "id": "29e939d9-a158-4376-86d7-5093c3a379cb",                     ␣
↪                                                                      |
|              |     "lease_id": "d7779e56-2b78-4465-8f19-9cc916d97b11",               ␣
↪                                                                      |
|              |     "event_type": "end_lease",                                        ␣
↪                                                                      |
|              |     "time": "2024-09-08T15:00:00.000000",                             ␣
↪                                                                      |
|              |     "status": "UNDONE",                                               ␣
```

(continues on next page)

(continued from previous page)

```
↪                                                                                |
|               |     "created_at": "2024-08-23 12:40:53",                        ␣
↪                                                                                |
|               |     "updated_at": null                                          ␣
↪                                                                                |
|               | }                                                               ␣
↪                                                                                |
|               | {                                                               ␣
↪                                                                                |
|               |     "id": "6c985a6e-416a-496a-9714-ae7b2dea0753",                ␣
↪                                                                                |
|               |     "lease_id": "d7779e56-2b78-4465-8f19-9cc916d97b11",          ␣
↪                                                                                |
|               |     "event_type": "before_end_lease",                           ␣
↪                                                                                |
|               |     "time": "2024-09-08T14:00:00.000000",                       ␣
↪                                                                                |
|               |     "status": "UNDONE",                                         ␣
↪                                                                                |
|               |     "created_at": "2024-08-23 12:40:53",                        ␣
↪                                                                                |
|               |     "updated_at": null                                          ␣
↪                                                                                |
|               | }                                                               ␣
↪                                                                                |
|               | {                                                               ␣
↪                                                                                |
|               |     "id": "e3308415-78b1-441d-8881-74e9569e0635",                ␣
↪                                                                                |
|               |     "lease_id": "d7779e56-2b78-4465-8f19-9cc916d97b11",          ␣
↪                                                                                |
|               |     "event_type": "start_lease",                                ␣
↪                                                                                |
|               |     "time": "2024-08-23T14:00:00.000000",                       ␣
↪                                                                                |
|               |     "status": "UNDONE",                                         ␣
↪                                                                                |
|               |     "created_at": "2024-08-23 12:40:53",                        ␣
↪                                                                                |
|               |     "updated_at": null                                          ␣
↪                                                                                |
|               | }                                                               ␣
↪                                                                                |
| id            | d7779e56-2b78-4465-8f19-9cc916d97b11                            ␣
↪                                                                                |
| name          | lease-1                                                         ␣
↪                                                                                |
| project_id    | 09dd299a860b4933a2ebc271377e77a6                                ␣
↪                                                                                |
```

(continues on next page)

(continued from previous page)

```
| reservations | {                                                              ␣
↪                                                                             |
|              |     "id": "c1fb5090-e046-42a5-8287-2aa380a8d31a",              ␣
↪                                                                             |
|              |     "lease_id": "d7779e56-2b78-4465-8f19-9cc916d97b11",        ␣
↪                                                                             |
|              |     "resource_id": "41448036-4bd0-4b03-bd72-829ad636024b",     ␣
↪                                                                             |
|              |     "resource_type": "flavor:instance",                        ␣
↪                                                                             |
|              |     "status": "pending",                                       ␣
↪                                                                             |
|              |     "missing_resources": false,                                ␣
↪                                                                             |
|              |     "resources_changed": false,                                ␣
↪                                                                             |
|              |     "created_at": "2024-08-23 12:40:53",                       ␣
↪                                                                             |
|              |     "updated_at": "2024-08-23 12:40:53",                       ␣
↪                                                                             |
|              |     "vcpus": 2,                                                ␣
↪                                                                             |
|              |     "memory_mb": 4096,                                         ␣
↪                                                                             |
|              |     "disk_gb": 40,                                             ␣
↪                                                                             |
|              |     "amount": 1,                                               ␣
↪                                                                             |
|              |     "affinity": null,                                          ␣
↪                                                                             |
|              |     "resource_properties": "{\"id\": \"3\", \"name\": \"m1.
↪medium\", \"ram\": 4096, \"disk\": 40, \"swap\": \"\", \"OS-FLV-EXT-     |
|              | DATA:ephemeral\": 0, \"OS-FLV-DISABLED:disabled\": false, \
↪"vcpus\": 2, \"os-flavor-access:is_public\": true, \"rxtx_factor\": 1.0, |
|              | \"extra_specs\": {\"hw_rng:allowed\": \"True\"}}",             ␣
↪                                                                             |
|              |     "flavor_id": "c1fb5090-e046-42a5-8287-2aa380a8d31a",       ␣
↪                                                                             |
|              |     "aggregate_id": 6,                                         ␣
↪                                                                             |
|              |     "server_group_id": null                                    ␣
↪                                                                             |
|              | }                                                              ␣
↪                                                                             |
| start_date   | 2024-08-23T14:00:00.000000                                     ␣
↪                                                                             |
| status       | PENDING                                                        ␣
↪                                                                             |
| trust_id     | 8023d766983a493c898991430493e81a                               ␣
```

(continues on next page)

(continued from previous page)

```
↪                                                                              |
| updated_at    | 2024-08-23 12:40:53                                             ␣
↪                                                                              |
| user_id       | 3ecacbbc4acc467faa664ee26eea115d                                ␣
↪                                                                              |
+---------------+--------------------------------------------------------------------
↪-------------------------------------------------------------------+
```

2. Check leases:

```
# Using the blazar CLI
blazar lease-list

# Using the openstack CLI
openstack reservation lease list
```

Result:

```
+--------------------------------------+---------+----------------------------
↪+----------------------------+
| id                                   | name    | start_date                 ␣
↪| end_date                   |
+--------------------------------------+---------+----------------------------
↪+----------------------------+
| d7779e56-2b78-4465-8f19-9cc916d97b11 | lease-1 | 2024-08-23T14:00:00.000000␣
↪| 2024-09-08T15:00:00.000000 |
+--------------------------------------+---------+----------------------------
↪+----------------------------+
```

3. Use the leased resources

While the reservation you created is active you can see and use the flavor of your reservation.

```
openstack flavor list
```

Result:

```
+--------------------------------------+-----------------------------------------
↪-----------+-------+------+-----------+-------+-----------+
| ID                                   | Name                                    ␣
↪           |   RAM | Disk | Ephemeral | VCPUs | Is Public |
+--------------------------------------+-----------------------------------------
↪-----------+-------+------+-----------+-------+-----------+
| 1                                    | m1.tiny                                 ␣
↪           |   512 |    1 |         0 |     1 | True      |
| 2                                    | m1.small                                ␣
↪           |  2048 |   20 |         0 |     1 | True      |
| 3                                    | m1.medium                               ␣
↪           |  4096 |   40 |         0 |     2 | True      |
| 4                                    | m1.large                                ␣
```

(continues on next page)

(continued from previous page)

```
↪              |  8192 |   80 |          0 |     4 | True       |
| 42                                   | m1.nano                              ␣
↪              |   192 |    1 |          0 |     1 | True       |
| 5                                    | m1.xlarge                            ␣
↪              | 16384 |  160 |          0 |     8 | True       |
| 84                                   | m1.micro                             ␣
↪              |   256 |    1 |          0 |     1 | True       |
| c1                                   | cirros256                            ␣
↪              |   256 |    1 |          0 |     1 | True       |
| c1fb5090-e046-42a5-8287-2aa380a8d31a | reservation:c1fb5090-e046-42a5-8287-
↪2aa380a8d31a |  4096 |   40 |          0 |     2 | False      |
+--------------------------------------+--------------------------------------
↪------------+-------+------+----------+-------+-----------+
```

1. Create a server: Please specify the flavor of the reservation.

```
openstack server create --flavor c1fb5090-e046-42a5-8287-2aa380a8d31a  --
↪image <image> --network <network> <server-name>
```

3.1.4 Floating IP Reservation

Prerequisites

The following packages should be installed:

- blazar
- neutron
- python-blazarclient

The floating IP plugin should be enabled in `blazar.conf`:

```
[manager]
plugins = virtual.floatingip.plugin
```

1. Create reservable floating IPs

1. The operator should create floating IPs as reservable resources using the floatingip-create command. They must select floating IPs that are not part of an allocation pool in Neutron. For example, to create a reservable floating IP with address `172.24.4.2` from the Neutron network with ID `81fabec7-00ae-497a-b485-72f4bf187d3e`, run:

```
# Using the blazar CLI
blazar floatingip-create 81fabec7-00ae-497a-b485-72f4bf187d3e 172.24.4.2

# Using the openstack CLI
openstack reservation floatingip create 81fabec7-00ae-497a-b485-72f4bf187d3e␣
↪172.24.4.2
```

2. Check reservable floating IPs:

```
# Using the blazar CLI
blazar floatingip-list

# Using the openstack CLI
openstack reservation floatingip list
```

Result:

```
+--------------------------------------+--------------------+----------------
↪----------------------+
| id                                   | floating_ip_address | floating_
↪network_id                   |
+--------------------------------------+--------------------+----------------
↪----------------------+
| 67720c36-4d53-41e6-acec-7d3fb9436fd5 | 172.24.4.2          | 81fabec7-00ae-
↪497a-b485-72f4bf187d3e |
+--------------------------------------+--------------------+----------------
↪----------------------+
```

2. Create a lease

1. Create a lease (floating IP reservation) using the lease-create command. Note that `python-blazarclient` version 2.2.1 or greater is required to use this feature. When you use `resource_type=virtual:floatingip`, the following parameters are supported:
   - `network_id`: UUID of the external network to reserve from (required)
   - `amount`: number of floating IPs to reserve (optional, defaults to 1)
   - `required_floatingips`: list of specific floating IPs to allocate (optional, must be formatted as a JSON array)

```
# Using the blazar CLI
blazar lease-create --reservation 'resource_type=virtual:floatingip,network_
↪id=81fabec7-00ae-497a-b485-72f4bf187d3e,amount=2,required_floatingips=["172.
↪24.4.2","172.24.4.3"]' fip-lease

# Using the openstack CLI
openstack reservation lease create --reservation 'resource_
↪type=virtual:floatingip,network_id=81fabec7-00ae-497a-b485-72f4bf187d3e,
↪amount=2,required_floatingips=["172.24.4.2","172.24.4.3"]' fip-lease
```

Result:

```
Created a new lease:
+-------------+--------------------------------------------------------------+
| Field       | Value                                                        |
+-------------+--------------------------------------------------------------+
| created_at  | 2019-09-23 08:33:22                                          |
| degraded    | False                                                        |
| end_date    | 2019-09-24T08:33:00.000000                                   |
| events      | {                                                            |
```

(continues on next page)

(continued from previous page)

```
|              |     "status": "UNDONE",                                      |
|              |     "lease_id": "d67f3bcf-cb82-4c7d-aa4d-49cc48586d89",      |
|              |     "event_type": "before_end_lease",                        |
|              |     "created_at": "2019-09-23 08:33:22",                     |
|              |     "updated_at": null,                                      |
|              |     "time": "2019-09-24T07:33:00.000000",                    |
|              |     "id": "628e6eec-d157-4e6a-9238-47c008f357be"             |
|              | }                                                            |
|              | {                                                            |
|              |     "status": "UNDONE",                                      |
|              |     "lease_id": "d67f3bcf-cb82-4c7d-aa4d-49cc48586d89",      |
|              |     "event_type": "end_lease",                               |
|              |     "created_at": "2019-09-23 08:33:22",                     |
|              |     "updated_at": null,                                      |
|              |     "time": "2019-09-24T08:33:00.000000",                    |
|              |     "id": "d8a56235-3171-4097-8dd6-425788f4dd73"             |
|              | }                                                            |
|              | {                                                            |
|              |     "status": "UNDONE",                                      |
|              |     "lease_id": "d67f3bcf-cb82-4c7d-aa4d-49cc48586d89",      |
|              |     "event_type": "start_lease",                             |
|              |     "created_at": "2019-09-23 08:33:22",                     |
|              |     "updated_at": null,                                      |
|              |     "time": "2019-09-23T08:33:00.000000",                    |
|              |     "id": "f7322caf-9470-4281-b980-dcd76b3e476c"             |
|              | }                                                            |
| id           | d67f3bcf-cb82-4c7d-aa4d-49cc48586d89                         |
| name         | fip-lease                                                    |
| project_id   | 10b4b88b67e141aeb093fec48c93232c                             |
| reservations | {                                                            |
|              |     "status": "pending",                                     |
|              |     "lease_id": "d67f3bcf-cb82-4c7d-aa4d-49cc48586d89",      |
|              |     "resource_id": "ae205735-970e-4f91-a2fc-c99fc7cc45fc",   |
|              |     "network_id": "81fabec7-00ae-497a-b485-72f4bf187d3e",    |
|              |     "created_at": "2019-09-23 08:33:22",                     |
|              |     "updated_at": "2019-09-23 08:33:22",                     |
|              |     "required_floatingips": [                                |
|              |         "172.24.4.2",                                        |
|              |         "172.24.4.3"                                         |
|              |     ],                                                       |
|              |     "missing_resources": false,                              |
|              |     "amount": 2,                                             |
|              |     "id": "30f72423-db81-4f13-bc78-b931c4a96b48",            |
|              |     "resource_type": "virtual:floatingip",                   |
|              |     "resources_changed": false                               |
|              | }                                                            |
| start_date   | 2019-09-23T08:33:00.000000                                   |
| status       | PENDING                                                      |
| trust_id     | 0617c18ba83d4ec29832b0ec19c5ae5e                             |
```

(continues on next page)

(continued from previous page)

```
| updated_at    | 2019-09-23 08:33:23                                                |
| user_id       | 9e43ffa598d14bac91fc889c2e15cd13                                   |
+---------------+--------------------------------------------------------------------+
```

2. Check leases:

```
# Using the blazar CLI
blazar lease-list

# Using the openstack CLI
openstack reservation lease list
```

Result:

```
+--------------------------------------+-----------+--------------------------
↪--+----------------------------+
| id                                   | name      | start_date               ␣
↪  | end_date                   |
+--------------------------------------+-----------+--------------------------
↪--+----------------------------+
| d67f3bcf-cb82-4c7d-aa4d-49cc48586d89 | fip-lease | 2019-09-23T08:33:00.
↪000000 | 2019-09-24T08:33:00.000000 |
+--------------------------------------+-----------+--------------------------
↪--+----------------------------+
```

3. Update a lease

1. Update a lease (floating IP reservation) using the lease-update command. Note that `python-blazarclient` version 2.2.1 or greater is required to use this feature. After passing the existing reservation ID to the `--reservation` option, you can modify start or end dates as well as some reservation parameters:

   - `amount`: you can modify the number of floating IPs to reserve. Reducing `amount` is supported only for pending reservations.
   - `required_floatingips`: you can only reset the list of specific floating IPs to allocate to an empty list

```
# Using the blazar CLI
blazar lease-update --reservation 'id=e80033e6-5279-461d-9573-dec137233434,
↪amount=3,required_floatingips=[]' fip-lease

# Using the openstack CLI
openstack reservation lease update --reservation 'id=e80033e6-5279-461d-9573-
↪dec137233434,amount=3,required_floatingips=[]' fip-lease
```

Result:

```
Updated lease: fip-lease
```

2. Check updated lease:

```
# Using the openstack CLI
blazar lease-show fip-lease

# Using the openstack CLI
openstack reservation lease show fip-lease
```

Result:

```
+--------------+------------------------------------------------------------------+
| Field        | Value                                                            |
+--------------+------------------------------------------------------------------+
| created_at   | 2019-09-23 08:09:51                                              |
| degraded     | False                                                            |
| end_date     | 2019-09-24T08:09:00.000000                                       |
| events       | {                                                                |
|              |     "status": "UNDONE",                                          |
|              |     "lease_id": "5d528d8d-c023-4792-ae77-cb6d4dc2c162",          |
|              |     "event_type": "before_end_lease",                            |
|              |     "created_at": "2019-09-23 08:09:51",                         |
|              |     "updated_at": null,                                          |
|              |     "time": "2019-09-24T07:09:00.000000",                        |
|              |     "id": "352521cc-bfe9-4881-9a3e-2ac770671144"                 |
|              | }                                                                |
|              | {                                                                |
|              |     "status": "DONE",                                            |
|              |     "lease_id": "5d528d8d-c023-4792-ae77-cb6d4dc2c162",          |
|              |     "event_type": "start_lease",                                 |
|              |     "created_at": "2019-09-23 08:09:51",                         |
|              |     "updated_at": "2019-09-23 08:10:10",                         |
|              |     "time": "2019-09-23T08:09:00.000000",                        |
|              |     "id": "59e1e170-660e-4a2d-a9e7-167fd5741ff5"                 |
|              | }                                                                |
|              | {                                                                |
|              |     "status": "UNDONE",                                          |
|              |     "lease_id": "5d528d8d-c023-4792-ae77-cb6d4dc2c162",          |
|              |     "event_type": "end_lease",                                   |
|              |     "created_at": "2019-09-23 08:09:51",                         |
|              |     "updated_at": null,                                          |
|              |     "time": "2019-09-24T08:09:00.000000",                        |
|              |     "id": "fda0d28d-afe5-4ebb-bea0-50ab1f8d7182"                 |
|              | }                                                                |
| id           | 5d528d8d-c023-4792-ae77-cb6d4dc2c162                             |
| name         | fip-lease                                                        |
| project_id   | 10b4b88b67e141aeb093fec48c93232c                                 |
| reservations | {                                                                |
|              |     "status": "active",                                          |
|              |     "lease_id": "5d528d8d-c023-4792-ae77-cb6d4dc2c162",          |
|              |     "resource_id": "543a350b-c703-48c9-a97e-2e787c26e385",       |
|              |     "network_id": "81fabec7-00ae-497a-b485-72f4bf187d3e",        |
|              |     "created_at": "2019-09-23 08:09:51",                         |
```

(continues on next page)

(continued from previous page)

```
|              |     "updated_at": "2019-09-23 08:10:10",                    |
|              |     "required_floatingips": [],                             |
|              |     "missing_resources": false,                             |
|              |     "amount": 3,                                            |
|              |     "id": "e80033e6-5279-461d-9573-dec137233434",            |
|              |     "resource_type": "virtual:floatingip",                  |
|              |     "resources_changed": false                              |
|              | }                                                           |
| start_date   | 2019-09-23T08:09:00.000000                                  |
| status       | ACTIVE                                                      |
| trust_id     | 707391571cd14bd9bfc8eaf986163b37                            |
| updated_at   | 2019-09-23 08:15:51                                         |
| user_id      | 9e43ffa598d14bac91fc889c2e15cd13                            |
+--------------+-------------------------------------------------------------+
```

4. Use the leased resources

1. Once the lease becomes active, the allocated floating IPs are tagged with the reservation ID, in this case `e80033e6-5279-461d-9573-dec137233434`, and can be displayed with the following command:

```
openstack floating ip list --tags reservation:e80033e6-5279-461d-9573-
↪dec137233434
```

Result:

```
+--------------------------------------+---------------------+----------------
↪--+------+--------------------------------------+--------------------------
↪-------+
| ID                                   | Floating IP Address | Fixed IP␣
↪Address | Port | Floating Network                     | Project                  ␣
↪             |
+--------------------------------------+---------------------+----------------
↪--+------+--------------------------------------+--------------------------
↪-------+
| 3954b799-4957-4e9f-96b7-46f72604c973 | 172.24.4.4          | None           ␣
↪  | None | 81fabec7-00ae-497a-b485-72f4bf187d3e |␣
↪10b4b88b67e141aeb093fec48c93232c |
| ae26069c-f7e9-4b8d-8ca0-6770c025dfae | 172.24.4.3          | None           ␣
↪  | None | 81fabec7-00ae-497a-b485-72f4bf187d3e |␣
↪10b4b88b67e141aeb093fec48c93232c |
| b427c171-30fe-45c4-a00b-3d5ca9b00306 | 172.24.4.2          | None           ␣
↪  | None | 81fabec7-00ae-497a-b485-72f4bf187d3e |␣
↪10b4b88b67e141aeb093fec48c93232c |
+--------------------------------------+---------------------+----------------
↪--+------+--------------------------------------+--------------------------
↪-------+
```

2. Use the reserved floating IP like a regular one, for example by attaching it to an instance with `openstack server add floating ip`.

Two command-line interfaces exist: one as a standalone `blazar` client and another integrated with the `openstack` client. Examples are given for both where applicable, as shown below:

```
# Using the blazar CLI
blazar lease-list

# Using the openstack CLI
openstack reservation lease list
```

CHAPTER

FOUR

API REFERENCE

4.1 Blazar REST API docs

CHAPTER

FIVE

USER GUIDE

5.1 User Guide

5.1.1 Introduction

Blazar is the *Resource Reservation Service* for OpenStack.

The idea of creating Blazar originated with two different use cases:

- Compute host reservation (when user with admin privileges can reserve hardware resources that are dedicated to the sole use of a project)
- Virtual machine (instance) reservation (when user may ask Blazar to provide them working VM not necessarily now, but also in the future)

These ideas have been transformed to a more general view: with Blazar, user can request the resources of cloud environment to be provided (leased) to their project for specific amount of time, immediately or in the future.

Currently, Blazar supports reservations of:

- Nova resources:
  - Compute hosts / hypervisors
  - Instances / servers
- Neutron resources:
  - Floating IPs

In terms of benefits added, Blazar:

- improves visibility of cloud resources consumption (current and planned for future);
- enables cloud resource planning based on current and future demand from end users;
- automates the processes of resource allocation and reclaiming.

Glossary of terms

Reservation is an allocation of cloud resources to a particular project. Main properties of a reservation are its status, resource type, identifier and the lease it belongs to.

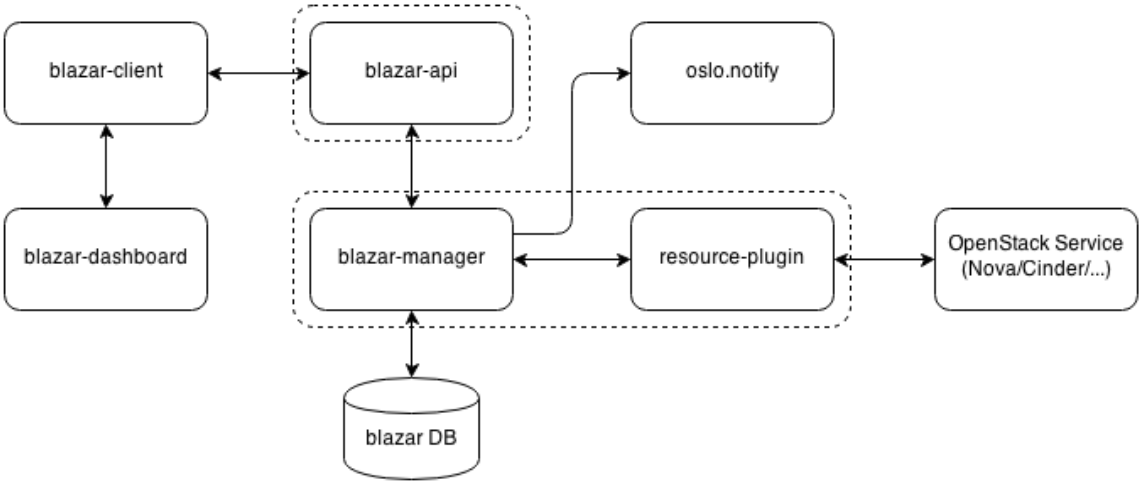
Lease is a negotiation agreement between the provider (Blazar, using OpenStack resources) and the consumer (user) where the former agrees to make some kind of cloud resources available to the latter, based on a set of lease terms presented by the consumer. Technically speaking, lease is a group of

reservations granted to a particular project upon request. Main properties of a lease are its start time, end time, set of reservations and set of events.

Event is something that may happen to a lease. In the simplest case, an event might describe lease start and lease end. It might also be a notification to user (e.g. about an upcoming lease expiration).

5.1.2 Blazar architecture

Blazars architecture is described by the following diagram:



blazar-client - provides a Python implementation of communication with Blazar API (`blazar-api` service). It works as a library, a standalone `blazar` CLI client, as well as a plugin of the `openstack` client.

blazar-api - waits for REST API calls from the outside world to redirect them to the manager. `blazar-api` communicates with `blazar-manager` via RPC.

blazar-manager - implements all logic and operations with leases, reservations and events. Communicates with Blazar DB and stores there data on leases, reservations and events. `blazar-manager` service is responsible for handling events created for leases and running all relevant actions. `blazar-manager` uses resource-plugins to work with other services resources.

resource-plugin - responsible for exact actions to apply to other services resources. All resource plugins reside in the same process as `blazar-manager`.

Nova resource reservation

Blazars integration with Nova is based on the following components working in tandem:

- Nova
  - Nova host aggregates are used to control which hosts have their access managed by Blazar. The canonical name for the Blazar-owned host aggregate is `freepool`.
  - Nova flavors are used for instance reservations.
- Blazar-Nova
  - The `blazar-nova` package offers the `BlazarFilter` filter for nova-scheduler. `BlazarFilter` ensures that Nova does not schedule regular (non-reservation-belonging) instances on Blazar-owned hosts (unless preemption is enabled) and that host lease boundaries are respected.

- Placement
  - Nested resource providers are used to control which hosts are managed by Blazar.
  - Reservation classes are created and nested resource providers inventories are updated for instance reservations.

It is worth noting that Blazar integrations are one-way, i.e. the other services never call Blazar. Even the `BlazarFilter` operates in such a way that all required data is present in Nova and Placement.

Note

Only the compute host reservation is compatible with preemptible instances at the moment.

Compute host reservation

Compute host reservation is analogous to a dedicated virtualisation host offering. The user asks for a certain number of hosts with specified characteristics, such as:

- the region,
- the availability zone,
- the host capabilities extra specs,
- the number of CPU cores,
- the amount of available RAM,
- the amount of available disk space.

Matching hosts are reserved for the sole use in the users project. Other projects will not share the same hosts during the lease period.

Virtual instance reservation

Virtual instance reservation offers a more granular compute resource reservation that does not book entire hypervisors but instead allows using Blazar-owned hosts via special time-limited flavors set up by Blazar. The user asks for a certain number of instances with specific flavor characteristics, such as:

- the number of provided CPU cores,
- the amount of provided RAM,
- the amount of provided disk space,
- affinity rule.

When the lease is active, a dedicated flavor is presented to the leasee. No other project may use this flavor. Blazar ensures, albeit in a best-effort manner, that supporting compute resources are reserved, e.g., it will not allow for oversubscribing with multiple reservation types (hosts supporting the virtual instance reservation cannot also be targeted for compute host reservation at the same time).

Flavor-based instance reservation

Flavor-based instance reservation is similar in design to virtual instance reservation. However, instead of accepting specific amounts of resources as arguments, the reservation is based on an existing Nova flavor which must be accessible to the user.

Note

The implementation of flavor-based instance reservation is still incomplete. Only a limited number of flavor extra specs is supported. Virtual instance reservations and flavor-based instance reservations cannot yet be created on the same resources. Affinity rules and resource traits are not yet supported.

Neutron resource reservation

Apart from compute resources of Nova, Blazar allows to reserve certain Neutron resources. At the moment, these are only floating IPs.

The Neutron integration thus far does not require changes to the Neutron environment. The Blazar interaction looks to Neutron like any other service client interaction.

Floating IPs reservation

Blazar admin may register floating IPs with Blazar which can then later be leased to end users. End users request floating IPs from a chosen network and they will be created in users project once the lease starts. The allowed floating IPs *must not* exist in subnets allocation pools. Blazar will validate this *only once* when admin registers the floating IP with Blazar. The integration *will break* if admin then adds the same floating IP to the allocation pool as Blazar will try to own it and fail.

5.1.3 State machines

Blazar objects (leases, reservations, and events) each have a status. This document describes statuses in detail.

Lease status

Lease statuses are categorized into two types: stable or transitional. In the state machine shown below, stable statuses are drawn as black nodes while transitional statuses are drawn as gray nodes. Status transitions are triggered by an API call or an event in a lease.

A lease has the following four stable statuses:

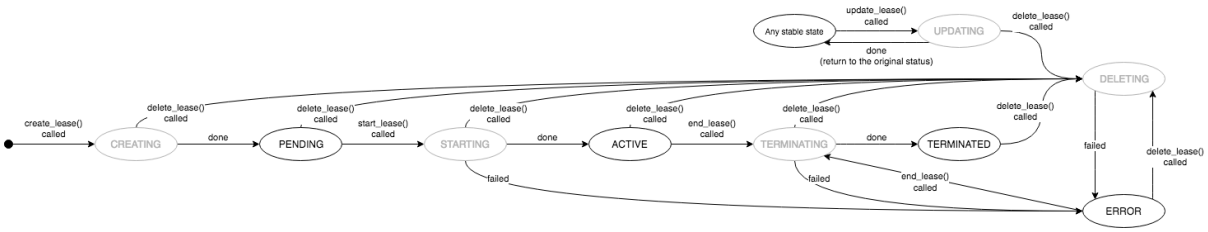
- **PENDING**: A lease has been successfully created and is ready to start. The lease stays in this status until it starts.
- **ACTIVE**: A lease has been started and is active.
- **TERMINATED**: A lease has been successfully terminated.
- **ERROR**: Unrecoverable failures happened to the lease.

Transitional statuses are as follows:

- **CREATING**: A lease is being created.
- **STARTING**: A lease is being started.
- **UPDATING**: A lease is being updated.

- **TERMINATING**: A lease is being terminated.
- **DELETING**: A lease is being deleted. Any status can change to this status because delete is the highest prioritized operation. e.g. when a lease hangs up in the STARTING status, delete should be allowed.

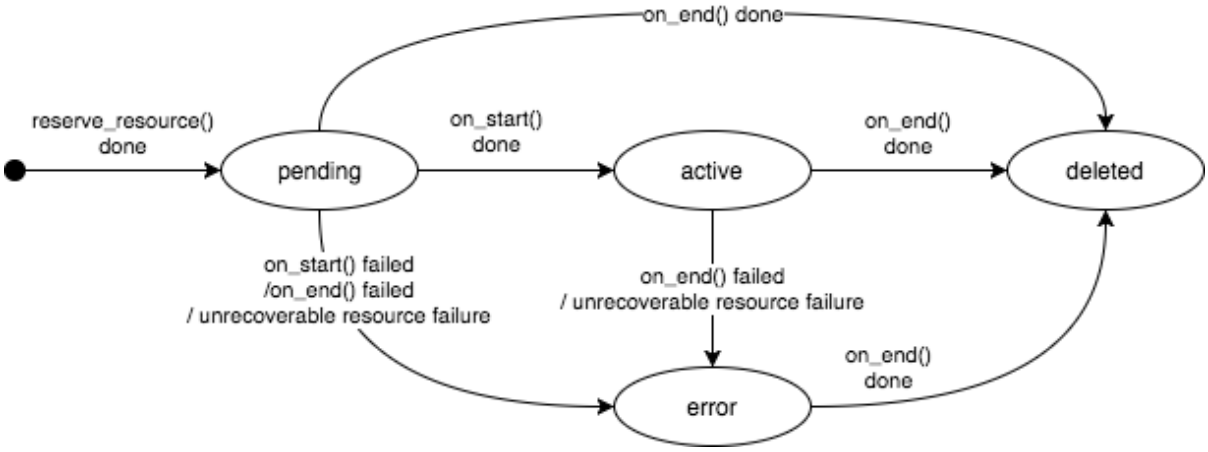
If some action can cause an invalid status transition, the action is denied. E.g. If a user sends an Update Lease request while it is starting, the Update Lease request is denied because the transition from **STARTING** to **UPDATING** is invalid.



Reservation status

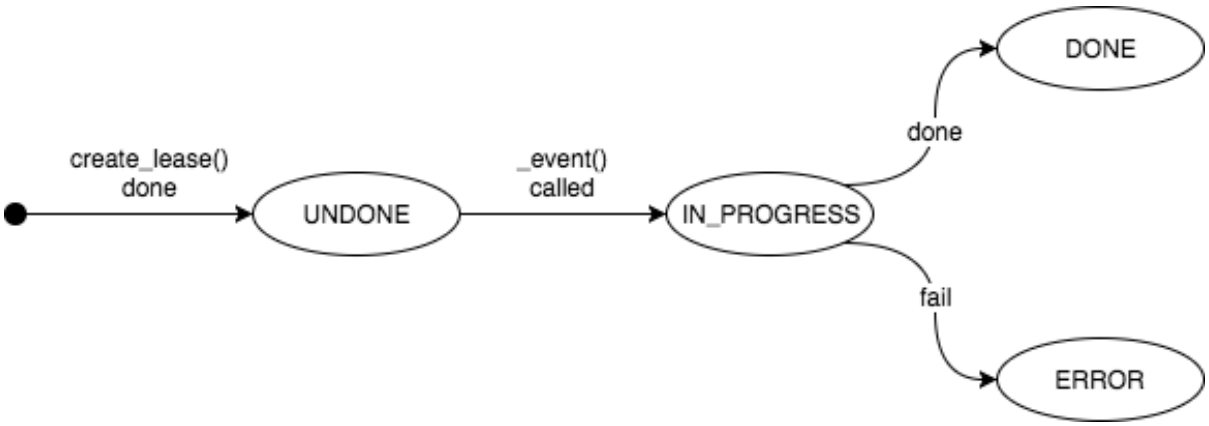
A reservation has the following four statuses. Lowercase letters are used for backward compatibility:

- **pending**: A reservation has been successfully created and is ready to start. The reservation stays in this status until it starts.
- **active**: A reservation has been started and is active.
- **deleted**: Reserved resources have been successfully released.
- **error**: Unrecoverable failures happened to resources.



Event status

Event statuses are as follows.



Relationships between statuses

The following table shows conditions of statuses of reservations and events that have to be satisfied for each lease status.

Lease	Reservations	Events
CREATING	pending	start_lease: UNDONE , end_lease: UNDONE
PENDING	pending	start_lease: UNDONE , end_lease: UNDONE
STARTING	pending or active or error	start_lease: IN_PROGRESS , end_lease: UNDONE
ACTIVE	active	start_lease: DONE , end_lease: UNDONE
TERMINATING	active or deleted or error	start_lease: DONE , end_lease: IN_PROGRESS
TERMINATED	deleted	start_lease: DONE , end_lease: DONE
DELETING	Any status	Any status
UPDATING	Any status	Any status other than IN_PROGRESS

blazar/status module

The *blazar/status* module defines and manages these statuses.

5.1.4 Resource Monitoring

Blazar monitors states of resources and heals reservations which are expected to suffer from resource failure. Resource specific functionality, e.g., calling Nova APIs, is provided as a monitoring plugin. The following sections describe the resource monitoring feature in detail.

Monitoring Type

Blazar supports 2 types of monitoring - push-based and polling-based.

1. Push-based monitoring

   The monitor listens to notification messages sent by other components, e.g., sent by Nova for the host monitoring plugin. And it picks up messages which refer to the resources managed by Blazar. Event types, topics to subscribe and notification callbacks are provided by monitoring plugins.

2. Polling-based monitoring

The blazar-manager periodically calls a states check method of monitoring plugins. Then, the monitoring plugins check states of resources, e.g., *List Hypervisors* of the Compute API is used for the host monitoring plugin.

Admins can enable/disable these monitoring by setting configuration options.

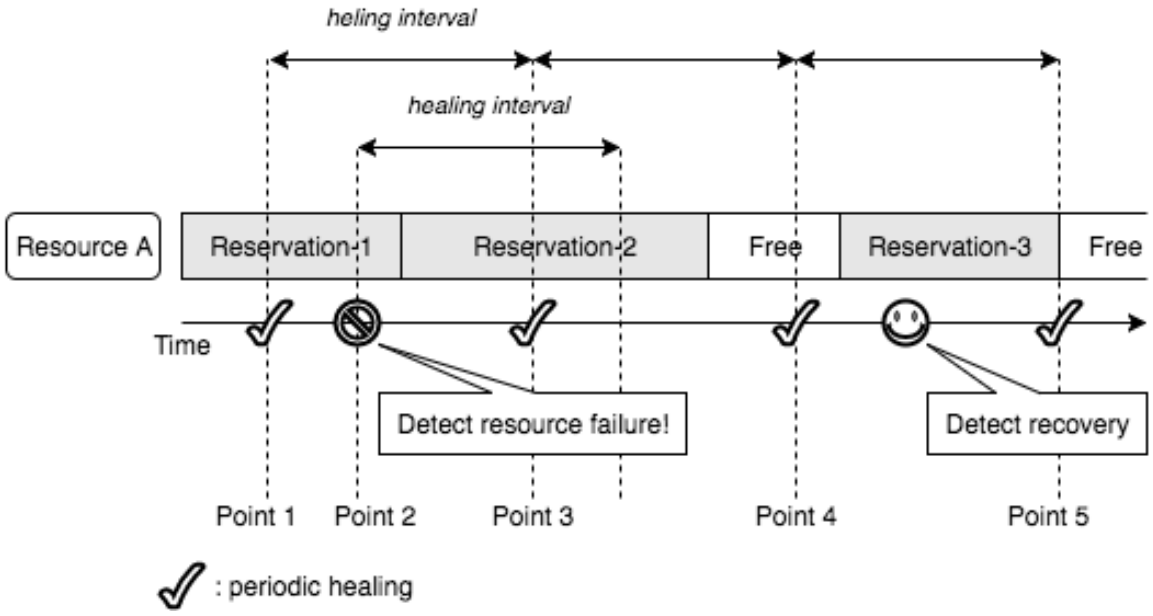
Healing

When the monitor detects a resource failure, it heals reservations which are expected to suffer from the failure. Note that it does not immediately heal all of reservations for the failed resource because the resource is expected to recover sometime in the future, i.e., the monitor heals only reservations which are active or will start soon.

In addition, the monitor periodically checks validity of reservations and heals invalid reservations. Therefore, even though the failed resource did not recover in the last interval, the periodic task heals invalid reservations which will start in the next interval.

The healing flow is as follows:

1. Resource A is reserved for the *Reservation-1*, *Reservation-2* and *Reservation-3* as shown in the following diagram.
2. At the point 1, the periodic task in the manager checks if there is any reservation to heal and it detects there is not.
3. At the point 2, the manager detects a failure of the resource A. Then, it heals active reservations and reservations which will start in the *healing interval*. In this case, *Reservation-1* and *Reservation-2* are healed immediately.
4. At the point 3, the periodic task checks if there is any reservation to heal. In this case, the task finds out there is no reservation to heal because the resource has not yet recovered but no reservation will start in next interval. *Reservation-2* has been already healed in step 3.
5. At the point 4, the periodic task checks if there is any reservation to heal again. In this case, the task finds out *Reservation-3* needs to be healed because it will start in the next interval and the resource has not yet recovered.
6. Before the point 5, the manager detects a recovery of the resource.
7. At the point 5, the periodic task finds out there is no failed resource and nothing to do.



Flags

Leases and reservations have flags that indicate states of reserved resources. Reservations have the following two flags:

- **missing_resources**: If any resource allocated to the reservation fails and no alternative resource found, this flag is set *True*.
- **resources_changed**: If any resource allocated to the *active* reservation and alternative resource is reallocated, this flag is set *True*.

Leases have the following flag:

- **degraded**: If the **missing_resources** or the **resources_changed** flags of any reservation included in the lease is *True*, then it is *True*.

Lease owners can see health of the lease and reservations included in the lease by checking these flags.

Monitoring Resources

Resource specific functionality is provided as a monitoring plugin. The following resource is currently supported.

Compute Host Monitor

Compute host monitor detects failure and recovery of compute hosts. If it detects failures, it triggers healing of host reservations and instance reservations. This document describes the compute host monitor plugin in detail.

Monitoring Type

Both of the push-based and the polling-based monitoring types are supported for the compute host monitor. These monitors can be enabled/disabled by the following configuration options:

- **enable_notification_monitor**: Set *True* to enable it.
- **enable_polling_monitor**: Set *True* to enable it.

Failure Detection

Compute host monitor detects failure and recovery hosts by subscribing Nova notifications or polling the *List Hypervisors* of Nova API. If any failure is detected, Blazar sets the *reservable* field of the failed host *False* and heals suffering reservations as follows.

Reservation Healing

If a host failure is detected, Blazar tries to heal host/instance reservations which use the failed host by reserving alternative host. The length of the *healing interval* can be configured by the *healing_interval* option.

Configurations

To enable the compute host monitor, enable *enable_notification_monitor* or *enable_polling_monitor* option, and set *healing_interval* as appropriate for your cloud. See also the *blazar.conf* in detail.

CHAPTER

SIX

ADMIN GUIDE

6.1 Administrator Guide

6.1.1 blazar-status

Synopsis

```
blazar-status <category> <command> [<args>]
```

Description

blazar-status is a tool that provides routines for checking the status of a Blazar deployment.

Options

The standard pattern for executing a **blazar-status** command is:

```
blazar-status <category> <command> [<args>]
```

Run without arguments to see a list of available command categories:

```
blazar-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:

```
blazar-status upgrade
```

These sections describe the available categories and arguments for **blazar-status**.

Upgrade

blazar-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

3.0.0 (Stein)

- Placeholder to be filled in with checks as they are added in Stein.

6.1.2 Usage Enforcement

Synopsis

Usage enforcement and lease constraints can be implemented by operators via custom usage enforcement filters or an external service.

Description

Usage enforcement filters are called on `lease_create`, `lease_update` and `on_end` operations. The filters check whether or not lease values or allocation criteria pass admin defined thresholds. There are currently two filters provided out-of-the-box. `MaxLeaseDurationFilter` restricts the duration of leases. `ExternalServiceFilter` calls a third-party service for implementing policies using a URL configured in `blazar.conf`.

Options

All filters are a subclass of the BaseFilter class located in `blazar/enforcement/filter/base_filter.py`. Custom filters must implement methods for `check_create`, `check_update`, and `on_end`. The `MaxLeaseDurationFilter` is a good example to follow. Filters are enabled in `blazar.conf` under the `[enforcement]` group. For example, enabling the `MaxLeaseDurationFilter` to limit lease durations to only one day would work as follows:

```
[enforcement]
enabled_filters = MaxLeaseDurationFilter
max_lease_duration = 86400
```

Do note that filter config options follow filter names - the prefix is always the snake case of the filter name (`MaxLeaseDurationFilter` becomes `max_lease_duration`; in this case it is special that there is nothing beyond the prefix but there is also `max_lease_duration_exempt_project_ids`).

MaxLeaseDurationFilter

This filter simply examines the lease `start_date` and `end_date` attributes and rejects the lease if its duration exceeds a threshold. It supports two configuration options:

- `max_lease_duration`
- `max_lease_duration_exempt_project_ids`

See the *blazar.conf* page for a description of these options.

ExternalServiceFilter

This filter delegates the decision for each API to an external HTTP service. The service must use token-based authentication, accepting (or ignoring) the static token sent by Blazar in the `X-Auth-Token` header. The following endpoints should be implemented:

- `POST /check-create`
- `POST /check-update`
- `POST /on-end`

The exact URLs can be overridden and not all have to be used (although we imagine a proper implementation requires at least both checks unless lease updates are disabled in the first place).

The external service should return `204 No Content` if the parameters meet defined criteria and `403 Forbidden` if not. The service may send a JSON body response with the `403 Forbidden` reply, including the rejection reasoning in the field named `message` as in:

```
{
  "message": "You shall not pass!"
}
```

An example of data the external service will receive in a request body (do note all dates and times are encoded as strings following the ISO8601 standard that is expected in JSON to represent dates and times):

- Request example:

```
{
  "context": {
    "user_id": "c631173e-dec0-4bb7-a0c3-f7711153c06c",
    "project_id": "a0b86a98-b0d3-43cb-948e-00689182efd4",
    "auth_url": "https://api.example.com:5000/v3",
    "region_name": "RegionOne"
  },
  "current_lease": {
    "start_date": "2020-05-13T00:00:00.012345+02:00",
    "end_time": "2020-05-14T23:59:00.012345+02:00",
    "reservations": [
      {
        "resource_type": "physical:host",
        "min": 1,
        "max": 2,
        "hypervisor_properties": "[]",
        "resource_properties": "[\"==\", \"$availability_zone\", \"az1\"]",
        "allocations": [
          {
            "id": "1",
            "hypervisor_hostname": "32af5a7a-e7a3-4883-a643-828e3f63bf54",
            "extra": {
              "availability_zone": "az1"
            }
```

(continues on next page)

(continued from previous page)


```
          }
        ]
      }
    ]
  },
  "lease": {
    "start_date": "2020-05-13T00:00:00.012345+02:00",
    "end_time": "2020-05-14T23:59:00.012345+02:00",
    "reservations": [
      {
        "resource_type": "physical:host",
        "min": 2,
        "max": 3,
        "hypervisor_properties": "[]",
        "resource_properties": "[\"==\", \"$availability_zone\", \"az1\"]",
        "allocations": [
          {
            "id": "1",
            "hypervisor_hostname": "32af5a7a-e7a3-4883-a643-828e3f63bf54",
            "extra": {
              "availability_zone": "az1"
            }
          },
          {
            "id": "2",
            "hypervisor_hostname": "af69aabd-8386-4053-a6dd-1a983787bd7f",
            "extra": {
              "availability_zone": "az1"
            }
          }
        ]
      }
    ]
  }
}
```


The `current_lease` field is present only in `check-update` requests and describes the existing lease. In both checks the `lease` field describes the new lease. In `on-end`, the `lease` field describes the lease that has just ended.

There is no guarantee on the delivery of the `on-end` event and it should be considered an optimisation rather than a reliable mechanism.

CHAPTER

SEVEN

FOR CONTRIBUTORS

7.1 Contributor Guide

7.1.1 How to contribute

Getting started

- Read the OpenStack Developers Guide
- Login to OpenStack Gerrit using your Launchpad ID
  - Sign the OpenStack Individual Contributor License Agreement
  - Check that your email is listed in Gerrit identities
- Subscribe to Blazar-related projects on OpenStack Gerrit. Go to your settings and in the watched projects add *openstack/blazar*, *openstack/blazar-nova*, *openstack/python-blazarclient* and *openstack/blazar-dashboard*.

As all bugs/blueprints are listed in Blazar Launchpad, you may keep track on them and choose some to work on.

How to keep in touch with community

- If youre not yet subscribed to the OpenStack general mailing list or to the OpenStack development mailing list, please do. Blazar-related emails must be send with **[blazar]** in the subject.
- All questions may be asked on our IRC channel #openstack-blazar on OFTC.
- We also have bi-weekly meetings on #openstack-blazar. Please check meeting details.

Your first commit to Blazar

- Read the OpenStack development workflow documentation
- Clone the corresponding Blazar repository: blazar, blazar-nova, client, blazar-dashboard
- Apply and commit your changes
- Make sure all code checks and tests have passed
- Send your patch for review
- Monitor the status of your change on https://review.openstack.org/

7.1.2 Development guidelines

Coding Guidelines

PEP8 checks should pass for all Blazar code. You may check it using the following command:

```
tox -e pep8
```

Also you should keep your code clear using more code style checks via pylint:

```
tox -e pylint
```

If you see any pep8/pylint errors in your code, it is mandatory to fix them before sending your change for review.

Testing Guidelines

Blazar repositories have unit tests that are run on all submitted code, and it is recommended for developers to execute them themselves to catch regressions early. Developers are also expected to keep the test suite up-to-date with any submitted code changes.

Unit tests might be run in TOX environments via the commands:

```
tox -e py36
tox -e py37
```

for Python 3.6 and Python 3.7 accordingly.

Note that the Blazar code base is not yet compatible with Python 3, so tests will be failing.

Note that some tests might use databases, the script `tools/test-setup.sh` sets up databases for the unit tests.

Documentation Guidelines

Currently Blazar docs are partially written on OpenStack wiki pages, and partially using Sphinx / RST located in the main repo in *doc* directory.

To build Sphinx / RST docs locally run the following command:

```
tox -e docs
```

Then you can access generated docs in the *doc/build/* directory, for example, the main page would be *doc/build/html/index.html*.

7.2 References

The blazar project has lots of complicated parts in it where it helps to have an overview to understand how the internals of a particular part work.

7.2.1 Internals

The following is a dive into some of the internals in blazar.

- *REST API Version History*: How blazar uses API microversion.

REST API Version History

This documents the changes made to the REST API with every microversion change. The description for each version should be a verbose one which has enough information to be suitable for use in user documentation.

1.0

This is the initial version of the v1.0 API which supports microversions. The v1.0 API is from the REST API userss point of view exactly the same as v1 except with strong input validation.

A user can specify a header in the API request:

```
OpenStack-API-Version: <version>
```

where <version> is any valid api version for this API.

If no version is specified then the API will behave as if a version request of v1.0 was requested.

CHAPTER

EIGHT

SPECS

CHAPTER

NINE

INDICES AND TABLES

- genindex
- search