OpenStack-Ansible Documentation: galera_server role

Release 18.1.0.dev383

OpenStack-Ansible Contributors

Feb 25, 2026

CONTENTS

Ansible role to install and configure a Galera cluster powered by MariaDB

To clone or view the source code for this repository, visit the role repository for galera_server.

CONTENTS

DEFAULT VARIABLES

```
# Set the package install state for distribution packages
# Options are 'present' and 'latest'
galera_package_state: "latest"

galera_cluster_members: "{{ groups['galera_all'] }}"
galera_server_bootstrap_node: "{{ galera_cluster_members[0] }}"
galera_ignore_cluster_state: false
galera_upgrade: false
galera_force_bootstrap: false

galera_wsrep_node_name: "{{ inventory_hostname }}"
galera_cluster_name: openstack_galera_cluster
galera_server_bind_address: "{{ openstack_service_bind_address | default('0.0.
↪0.0') }}"
galera_server_proxy_protocol_networks: ""

# The galera server-id should be set on all cluster nodes to ensure
#  that replication is handled correctly and the error
#  "Warning: You should set server-id to a non-0 value if master_host is
#   set; we will force server id to 2, but this MySQL server will not act
#   as a slave." is no longer present.
# galera_server_id: 0

# These are here to stub out the internal ROLE API.
#  if these are used they should be set within the
#  distro specific variable files found in vars/
galera_debconf_items: []
galera_mariadb_service_name: mariadb
galera_mariadb_client_binary: mariadb
galera_mariadb_server_package: "{{ (galera_install_method == 'external_repo')␣
↪| ternary(_galera_mariadb_external_repo_package, 'mariadb-server') }}"

# The major version used to select the repo URL path
galera_major_version: 11.8
galera_minor_version: 3

# Set the URL for the MariaDB repository
galera_repo_host: "archive.mariadb.org"
```

```yaml
galera_repo_url: "{{ _galera_repo_url }}"

# Set the repo information for the MariaDB repository
galera_repo: "{{ _galera_repo }}"

# Mappings from Ansible reported architecture to distro release architecture
galera_architecture_mapping: "{{ _galera_architecture_mapping }}"

# Set the gpg keys needed to be imported
# This should be a list of dicts, with each dict
# giving a set of arguments to the applicable
# package module. The following is an example for
# systems using the apt package manager.
# galera_gpg_keys:
#   - id: '0xF1656F24C74CD1D8'
#     keyserver: 'hkp://keyserver.ubuntu.com:80'
#     validate_certs: no
galera_gpg_keys: "{{ _galera_gpg_keys | default([]) }}"

galera_monitoring_user: monitoring
galera_monitoring_user_password: ""
galera_monitoring_port: 3307
galera_monitoring_max_connections: 10

# WARNING: Set this to open IP rules for galera monitoring.
# This is REQUIRED to run a working openstack-ansible deployment.
# If it's undefined the galera cluster state can't be reported,
# and haproxy would fail to do proper load balancing on the cluster.
# Because this opens connections to the cluster status, this
# should be restricted, which we do in the integrated build.
# Please override accordingly to your use case.
# This can be replaced with other hostnames, cidr, ips, and ips + wildcards.
# See https://www.freedesktop.org/software/systemd/man/systemd.resource-
↪control.html
#
# galera_monitoring_allowed_source: "0.0.0.0/0"

# Additional users to add or remove
galera_additional_users: []
#   - name: "my_username"
#     host: '%'
#     password: "my_password"
#     priv: "*.*:USAGE"
#     state: present

# Choose 'distro' or 'external_repo'
galera_install_method: external_repo

# Enable or disable the installation of galera development packages
```

```yaml
galera_install_devel: false

# Enable or disable the installation of galera server
galera_install_server: false

# Enable or disable the galera monitoring check capability
galera_monitoring_check_enabled: true

# Set the monitoring port used with the galera monitoring check.
galera_monitoring_check_port: 9200

galera_root_user: admin

# WARNING: This option is deprecated and will be removed in v12.0
galera_gcache_size: 1024M

galera_data_dir: /var/lib/mysql
galera_max_heap_table_size: 32M
galera_tmp_table_size: 32M
galera_tmp_dir: /var/lib/mysql/#tmp
galera_ignore_db_dirs:
  - "'#tmp'"
  - "lost+found"

galera_file_limits: 164679
galera_wait_timeout: "{{ openstack_db_connection_recycle_time | default('600
→') }}"
# Increase this value if large SST transfers cause mysql startup to fail due
# to timeout
galera_startup_timeout: 1800

## charset options
galera_default_charset: utf8mb3
galera_default_collation: general_ci
# NOTE: Default collation set will be updated with defined above defaults
galera_default_collation_set:
  utf8mb3: utf8mb3_uca1400_ai_ci
  utf8mb4: utf8mb4_uca1400_ai_ci
  ucs2: ucs2_uca1400_ai_ci
  utf16: utf16_uca1400_ai_ci
  utf32: utf32_uca1400_ai_ci

## innodb options
galera_innodb_buffer_pool_size: 4096M
galera_innodb_log_file_size: 1024M
galera_innodb_log_buffer_size: 128M

## wsrep configuration
galera_wsrep_address: "{{ management_address | default(ansible_host) }}"
```

```yaml
galera_wsrep_address_port: "{{ galera_wsrep_address }}:3306"
galera_wsrep_cluster_port: 4567
galera_wsrep_cluster_address: >-
  {% set _var = [] -%}
  {% for cluster_host in galera_cluster_members -%}
  {% set _addr = hostvars[cluster_host]['galera_wsrep_address']
                 | default(hostvars[cluster_host]['ansible_host']) -%}
  {% if _var.append(_addr) %}{% endif -%}
  {% endfor -%}
  {# If only 1 cluster member is present output an empty string so the
     single-node member will re-bootstrap correctly upon restart #}
  {{ _var | join(',') if galera_cluster_members | length > 1 else '' }}

galera_wsrep_node_incoming_address: "{{ galera_wsrep_address }}"
## Cap the maximum number of threads / workers when a user value is
→unspecified.
galera_wsrep_slave_threads_max: 16
galera_wsrep_slave_threads: "{{ [[ansible_facts['processor_vcpus'] |
→default(2), 2] | max, galera_wsrep_slave_threads_max] | min }}"
galera_wsrep_retry_autocommit: 3
galera_wsrep_debug: NONE
galera_wsrep_sst_method: mariabackup
galera_wsrep_provider_options:
  - { option: "gcache.size", value: "{{ galera_gcache_size }}" }
  - { option: "gmcast.listen_addr", value: "tcp://{{ galera_wsrep_node_
→incoming_address }}:{{ galera_wsrep_cluster_port }}" }
galera_wsrep_sst_auth_user: "{{ galera_root_user }}"
galera_wsrep_sst_auth_password: "{{ galera_root_password }}"

# mariabackup parallel/sync threads
galera_mariabackup_threads: 4

# Galera slow/unindexed query logging
galera_slow_query_logging: 0
galera_slow_query_log_file: "/var/log/mysql/mariadb-slow.log"
galera_unindexed_query_logging: 0

## Tunable overrides
galera_my_cnf_overrides: {}
galera_cluster_cnf_overrides: {}
galera_debian_cnf_overrides: {}
galera_encryption_overrides: {}
galera_init_overrides: {}

# Set the max connections value for galera. Set this value to override the
# computed value which is (100 x vCPUs) with a cap of 1600. If computed, the
# lowest value throughout the cluster will be used which is something to note
# if deploying galera on different hardware.
# galera_max_connections: 500
```

```yaml
# This is only applied if the ansible_facts['pkg_mgr'] is 'apt'
galera_distro_package_pins:
  - package: "*"
    release: MariaDB
    priority: 999
  - package: "mariadb-*"
    version: "1:{{ galera_major_version }}.{{ galera_minor_version }}*"
    priority: 1001

# Galera Server SSL functionality.

# Storage location for SSL certificate authority
galera_pki_dir: "{{ openstack_pki_dir | default('/etc/pki/galera-ca') }}"

# Create a certificate authority if one does not already exist
galera_pki_create_ca: "{{ openstack_pki_authorities is not defined | bool }}"
galera_pki_regen_ca: ""

galera_pki_authorities:
  - name: "MariaDBRoot"
    country: "GB"
    state_or_province_name: "England"
    organization_name: "Example Corporation"
    organizational_unit_name: "IT Security"
    cn: "MariaDB Root CA"
    provider: selfsigned
    basic_constraints: "CA:TRUE"
    key_usage:
      - digitalSignature
      - cRLSign
      - keyCertSign
    not_after: "+3650d"
  - name: "MariaDBIntermediate"
    country: "GB"
    state_or_province_name: "England"
    organization_name: "Example Corporation"
    organizational_unit_name: "IT Security"
    cn: "MariaDB Intermediate CA"
    provider: ownca
    basic_constraints: "CA:TRUE,pathlen:0"
    key_usage:
      - digitalSignature
      - cRLSign
      - keyCertSign
    not_after: "+3650d"
    signed_by: "MariaDBRoot"

# Installation details for certificate authorities
```

```yaml
galera_pki_install_ca:
  - name: "MariaDBRoot"
    condition: "{{ galera_pki_create_ca }}"

# Galera server certificate
galera_pki_keys_path: "{{ galera_pki_dir ~ '/certs/private/' }}"
galera_pki_certs_path: "{{ galera_pki_dir ~ '/certs/certs/' }}"
galera_pki_intermediate_cert_name: "{{ openstack_pki_service_intermediate_
↪cert_name | default('MariaDBIntermediate') }}"
galera_pki_intermediate_cert_path: >-
  {{
    galera_pki_dir ~ '/roots/' ~ galera_pki_intermediate_cert_name ~ '/certs/
↪' ~ galera_pki_intermediate_cert_name ~ '.crt'
  }}
galera_pki_regen_cert: ""
galera_pki_certificates:
  - name: "galera_{{ ansible_facts['hostname'] }}"
    provider: ownca
    cn: "{{ ansible_facts['hostname'] }}"
    san: >-
      {{
        'DNS:' ~ ansible_facts['hostname'] ~ ',' ~ (
          (galera_address | ansible.utils.ipaddr) is string) | ternary('IP',
↪'DNS') ~ ':' ~ galera_address ~
          ',IP:' ~ management_address
      }}
    signed_by: "{{ galera_pki_intermediate_cert_name }}"

galera_use_ssl: false
galera_ssl_verify: true
galera_ssl_cert: /etc/ssl/certs/galera.pem
galera_ssl_key: /etc/mysql/ssl/galera.key
galera_ssl_ca_cert: /etc/ssl/certs/galera-ca.pem
galera_require_secure_transport: false
galera_tls_version: "TLSv1.2,TLSv1.3"

## These options should be specified in user_variables if necessary,
↪otherwise self-signed certs are used.
# galera_user_ssl_cert: /etc/openstack_deploy/self_signed_certs/galera.pem
# galera_user_ssl_key: /etc/openstack_deploy/self_signed_certs/galera.key
# galera_user_ssl_ca_cert: /etc/openstack_deploy/self_signed_certs/galera-ca.
↪pem

# This option is used for creating the CA and overriding the Galera address
↪on the clients side.
# Should be set to either internal VIP or VIP FQDN, depending on what is
↪currently used in the env.
galera_address: "{{ ansible_host }}"
```

```yaml
# Installation details for SSL certificates
galera_pki_install_certificates:
  - src: "{{ galera_user_ssl_cert | default(galera_pki_certs_path ~ 'galera_'␣
→~ ansible_facts['hostname'] ~ '-chain.crt') }}"
    dest: "{{ galera_ssl_cert }}"
    owner: "root"
    group: "root"
    mode: "0644"
  - src: "{{ galera_user_ssl_key | default(galera_pki_keys_path ~ 'galera_' ~␣
→ansible_facts['hostname'] ~ '.key.pem') }}"
    dest: "{{ galera_ssl_key }}"
    owner: "mysql"
    group: "root"
    mode: "0600"
  - src: "{{ galera_user_ssl_ca_cert | default(galera_pki_intermediate_cert_
→path) }}"
    dest: "{{ galera_ssl_ca_cert }}"
    owner: "root"
    group: "root"
    mode: "0644"

# MariaDB 10.1+ ships with 'PrivateDevices=True' in the systemd unit file. This
# provides some additional security, but it causes problems with systemd 219.
# While the security enhancements are helpful on bare metal hosts with␣
→multiple
# services running, they are not as helpful when MariaDB is running in a
# container with its own isolated namespaces.
#
# Related bugs:
#   https://bugs.launchpad.net/openstack-ansible/+bug/1697531
#   https://github.com/lxc/lxc/issues/1623
#   https://github.com/systemd/systemd/issues/6121
#
# Setting the following variable to 'yes' will disable the PrivateDevices
galera_disable_privatedevices: "{{ _galera_disable_privatedevices }}"

# install and configure the galera client as well as the server
galera_install_client: false
galera_client_package_install: "{{ galera_install_client }}"
galera_client_package_state: "latest"
galera_client_drop_config_file: "true"
galera_client_my_cnf_overrides: {}

# Delegated host for operating the certificate authority
galera_ssl_server: "{{ openstack_pki_setup_host | default('localhost') }}"

## Database info
galera_db_setup_host: "{{ openstack_db_setup_host | default(galera_cluster_
→members[0] | default('localhost')) }}"
```

```yaml
galera_db_setup_python_interpreter: >-
  {{
    openstack_db_setup_python_interpreter | default(
      (galera_db_setup_host == 'localhost') | ternary(ansible_playbook_python,
↪ ansible_facts['python']['executable'])
    )
  }}


# Configure backups of database
# copies is the number of full backups to be kept, the corresponding
# incremental backups will also be kept. Uses systemd timer instead of cron.
galera_mariadb_backups_enabled: false
# galera_mariadb_backups_group_gid: <specify a GID>
galera_mariadb_backups_group_name: backups
galera_mariadb_backups_path: "/var/backup/mariadb_backups"
galera_mariadb_backups_full_copies: 2
galera_mariadb_backups_full_on_calendar: "*-*-* 00:00:00"
galera_mariadb_backups_full_randomized_delay_sec: 0
galera_mariadb_backups_full_init_overrides: {}

galera_mariadb_backups_increment_on_calendar:
  - "*-*-* 06:00:00"
  - "*-*-* 12:00:00"
  - "*-*-* 18:00:00"
galera_mariadb_backups_increment_randomized_delay_sec: 0
galera_mariadb_backups_increment_init_overrides: {}
# galera_mariadb_backups_user is the name of the mariadb database user
galera_mariadb_backups_user: galera_mariadb_backup
galera_mariadb_backups_suffix: "{{ inventory_hostname }}"
galera_mariadb_backups_cnf_file: "/etc/mysql/mariabackup.cnf"
galera_mariadb_backups_nodes: ["{{ galera_cluster_members[0] }}"]
galera_mariadb_backups_compress: false
galera_mariadb_backups_compressor: gzip

galera_mariadb_encryption_enabled: false
galera_mariadb_encryption_plugin: "file_key_management"
galera_db_encryption_tmp_dir: ""
```

REQUIRED VARIABLES

To use this role, define the following variables:

```
galera_root_password: secrete
```

EXAMPLE PLAYBOOK

```yaml
---
- name: Install Galera server
  hosts: galera_all
  user: root
  serial:
    - 1
    - 100%
  roles:
    - galera_server
  vars:
    galera_install_server: true
    # NOTE: it is important to disable client installation on
    #       server, unless `galera_root_user: root` (default: admin)
    galera_install_client: false
    galera_root_password: secrete

- name: Install Galera client
  hosts: localhost
  roles:
    - galera_server
  vars:
    galera_install_server: false
    galera_install_client: true
    galera_root_password: secrete
```