OpenStack-Ansible Documentation: os_swift role

Release 18.1.0.dev233

OpenStack-Ansible Contributors

Jun 07, 2022

CONTENTS

Home OpenStack-Ansible Swift

CHAPTER

ONE

CONFIGURING SWIFT

Home OpenStack-Ansible Swift

1.1 Storage devices

This section offers a set of prerequisite instructions for setting up Object Storage (swift) storage devices. The storage devices must be set up before installing swift.

Procedureă5.1.ăConfiguring and mounting storage devices

Object Storage recommends a minimum of three swift hosts with five storage disks. The example commands in this procedure use the storage devices `sdc` through to `sdg`.

1. Determine the storage devices on the node to be used for swift.
2. Format each device on the node used for storage with XFS. While formatting the devices, add a unique label for each device.

   Without labels, a failed drive causes mount points to shift and data to become inaccessible.

   For example, create the file systems on the devices using the `mkfs` command:

```
# apt-get install xfsprogs
# mkfs.xfs -f -i size=1024 -L sdc /dev/sdc
# mkfs.xfs -f -i size=1024 -L sdd /dev/sdd
# mkfs.xfs -f -i size=1024 -L sde /dev/sde
# mkfs.xfs -f -i size=1024 -L sdf /dev/sdf
# mkfs.xfs -f -i size=1024 -L sdg /dev/sdg
```

3. Add the mount locations to the `fstab` file so that the storage devices are remounted on boot. The following example mount options are recommended when using XFS:

```
LABEL=sdc /srv/node/sdc xfs noatime,nodiratime,nobarrier,logbufs=8,
↪auto 0 0
LABEL=sdd /srv/node/sdd xfs noatime,nodiratime,nobarrier,logbufs=8,
↪auto 0 0
LABEL=sde /srv/node/sde xfs noatime,nodiratime,nobarrier,logbufs=8,
↪auto 0 0
LABEL=sdf /srv/node/sdf xfs noatime,nodiratime,nobarrier,logbufs=8,
↪auto 0 0
LABEL=sdg /srv/node/sdg xfs noatime,nodiratime,nobarrier,logbufs=8,
↪auto 0 0
```

4. Create the mount points for the devices using the `mkdir` command:

```
# mkdir -p /srv/node/sdc
# mkdir -p /srv/node/sdd
# mkdir -p /srv/node/sde
# mkdir -p /srv/node/sdf
# mkdir -p /srv/node/sdg
```

The mount point is referenced as the `mount_point` parameter in the `swift.yml` file (`/etc/rpc_deploy/conf.d/swift.yml`):

```
# mount /srv/node/sdc
# mount /srv/node/sdd
# mount /srv/node/sde
# mount /srv/node/sdf
# mount /srv/node/sdg
```

To view an annotated example of the `swift.yml` file, see this link.

For the following mounted devices:

Device	Mount location
/dev/sdc	/srv/node/sdc
/dev/sdd	/srv/node/sdd
/dev/sde	/srv/node/sde
/dev/sdf	/srv/node/sdf
/dev/sdg	/srv/node/sdg

Table: Tableă5.1.ăMounted devices

The entry in the `swift.yml`:

```
#    drives:
#        - name: sdc
#        - name: sdd
#        - name: sde
#        - name: sdf
#        - name: sdg
#    mount_point: /srv/node
```

Home OpenStack-Ansible Swift

1.2 Configuring the service

Procedureă5.2.ăUpdating the Object Storage configuration "swift.yml" file

1. Copy the `/etc/openstack_deploy/conf.d/swift.yml.example` file to `/etc/openstack_deploy/conf.d/swift.yml`:

```
# cp /etc/openstack_deploy/conf.d/swift.yml.example \
   /etc/openstack_deploy/conf.d/swift.yml
```

2. Update the global override values:

```
# global_overrides:
#   swift:
#     part_power: 8
#     weight: 100
#     min_part_hours: 1
#     repl_number: 3
#     storage_network: 'br-storage'
#     replication_network: 'br-repl'
#     drives:
#       - name: sdc
#       - name: sdd
#       - name: sde
#       - name: sdf
#     mount_point: /srv/node
#     account:
#     container:
#     storage_policies:
#       - policy:
#           name: gold
#           index: 0
#           default: True
#       - policy:
#           name: silver
#           index: 1
#           repl_number: 3
#           deprecated: True
#     statsd_host: statsd.example.com
#     statsd_port: 8125
#     statsd_metric_prefix:
#     statsd_default_sample_rate: 1.0
#     statsd_sample_rate_factor: 1.0
```

part_power Set the partition power value based on the total amount of storage the entire ring uses.

Multiply the maximum number of drives ever used with the swift installation by 100 and round that value up to the closest power of two value. For example, a maximum of six drives, times 100, equals 600. The nearest power of two above 600 is two to the power of nine, so the partition power is nine. The partition power cannot be changed after the swift rings are built.

weight The default weight is 100. If the drives are different sizes, set the weight value to avoid uneven distribution of data. For example, a 1 TB disk would have a weight of 100, while a 2 TB drive would have a weight of 200.

min_part_hours The default value is 1. Set the minimum partition hours to the amount of time to lock a partitions replicas after moving a partition. Moving multiple replicas at the same time makes data inaccessible. This value can be set separately in the swift, container, account, and policy sections with the value in lower sections superseding the value in the swift section.

repl_number The default value is 3. Set the replication number to the number of replicas of each object. This value can be set separately in the swift, container, account, and policy sections with the value in the more granular sections superseding the value in the swift section.

storage_network By default, the swift services listen on the default management IP. Option-

ally, specify the interface of the storage network.

If the `storage_network` is not set, but the `storage_ips` per host are set (or the `storage_ip` is not on the `storage_network` interface) the proxy server is unable to connect to the storage services.

`replication_network` Optionally, specify a dedicated replication network interface, so dedicated replication can be setup. If this value is not specified, no dedicated `replication_network` is set.

Replication does not work properly if the `repl_ip` is not able to connect to other hosts `repl_ip`.

`drives` Set the default drives per host. This is useful when all hosts have the same drives. These can be overridden on a per host basis.

`mount_point` Set the `mount_point` value to the location where the swift drives are mounted. For example, with a mount point of `/srv/node` and a drive of `sdc`, a drive is mounted at `/srv/node/sdc` on the `swift_host`. This can be overridden on a per-host basis.

`storage_policies` Storage policies determine on which hardware data is stored, how the data is stored across that hardware, and in which region the data resides. Each storage policy must have a unique `name` and a unique `index`. There must be a storage policy with an index of 0 in the `swift.yml` file to use any legacy containers created before storage policies were instituted.

`default` Set the default value to `yes` for at least one policy. This is the default storage policy for any non-legacy containers that are created.

`deprecated` Set the deprecated value to `yes` to turn off storage policies.

For account and container rings, `min_part_hours` and `repl_number` are the only values that can be set. Setting them in this section overrides the defaults for the specific ring.

`statsd_host` Swift supports sending extra metrics to a `statsd` host. This option sets the `statsd` host to receive `statsd` metrics. Specifying this here applies to all hosts in the cluster.

If `statsd_host` is left blank or omitted, then `statsd` are disabled.

All `statsd` settings can be overridden or you can specify deeper in the structure if you want to only catch `statsdv` metrics on certain hosts.

`statsd_port` Optionally, use this to specify the `statsd` servers port you are sending metrics to. Defaults to 8125 of omitted.

`statsd_default_sample_rate` and `statsd_sample_rate_factor` These `statsd` related options are more complex and are used to tune how many samples are sent to `statsd`. Omit them unless you need to tweak these settings, if so first read: https://docs.openstack.org/swift/latest/admin_guide.html

3. Update the swift proxy hosts values:

```
# swift-proxy_hosts:
#   infra-node1:
#     ip: 192.0.2.1
#     statsd_metric_prefix: proxy01
#   infra-node2:
```

(continues on next page)

(continued from previous page)

```
#     ip: 192.0.2.2
#     statsd_metric_prefix: proxy02
#   infra-node3:
#     ip: 192.0.2.3
#     statsd_metric_prefix: proxy03
```

swift-proxy_hosts Set the `IP` address of the hosts so Ansible connects to to deploy the `swift-proxy` containers. The `swift-proxy_hosts` value matches the infra nodes.

statsd_metric_prefix This metric is optional, and also only evaluated it you have defined `statsd_host` somewhere. It allows you define a prefix to add to all `statsd` metrics sent from this hose. If omitted, use the node name.

1. Update the swift hosts values:

```
# swift_hosts:
#   swift-node1:
#     ip: 192.0.2.4
#     container_vars:
#       swift_vars:
#         zone: 0
#         statsd_metric_prefix: node1
#   swift-node2:
#     ip: 192.0.2.5
#     container_vars:
#       swift_vars:
#         zone: 1
#         statsd_metric_prefix: node2
#   swift-node3:
#     ip: 192.0.2.6
#     container_vars:
#       swift_vars:
#         zone: 2
#         statsd_metric_prefix: node3
#   swift-node4:
#     ip: 192.0.2.7
#     container_vars:
#       swift_vars:
#         zone: 3
#   swift-node5:
#     ip: 192.0.2.8
#     container_vars:
#       swift_vars:
#         storage_ip: 198.51.100.8
#         repl_ip: 203.0.113.8
#         zone: 4
#         region: 3
#         weight: 200
#         groups:
#           - account
#           - container
#           - silver
#         drives:
#           - name: sdb
#             weight: 75
#             groups:
```

(continues on next page)

(continued from previous page)

```
#                 - gold
#             - name: sdc
#             - name: sdd
#             - name: sde
#             - name: sdf
```

swift_hosts Specify the hosts to be used as the storage nodes. The `ip` is the address of the host to which Ansible connects. Set the name and IP address of each swift host. The `swift_hosts` section is not required.

swift_vars Contains the swift host specific values.

storage_ip and repl_ip Base these values on the IP addresses of the hosts `storage_network` or `replication_network`. For example, if the `storage_network` is `br-storage` and host1 has an IP address of 1.1.1.1 on `br-storage`, then this is the IP address in use for `storage_ip`. If only the `storage_ip` is specified, then the `repl_ip` defaults to the `storage_ip`. If neither are specified, both default to the host IP address.

zone The default is 0. Optionally, set the swift zone for the ring.

region Optionally, set the swift region for the ring.

weight The default weight is 100. If the drives are different sizes, set the weight value to avoid uneven distribution of data. This value can be specified on a host or drive basis (if specified at both, the drive setting takes precedence).

groups Set the groups to list the rings to which a hosts drive belongs. This can be set on a per drive basis which overrides the host setting.

drives Set the names of the drives on the swift host. Specify at least one name.

statsd_metric_prefix

This metric is optional, and only evaluates if `statsd_host` is defined somewhere. This allows you to define a prefix to add to all `statsd` metrics sent from the hose. If omitted, use the node name.

In the following example, `swift-node5` shows values in the `swift_hosts` section that override the global values. Groups are set, which overrides the global settings for drive `sdb`. The weight is overridden for the host and specifically adjusted for drive `sdb`.

```
#  swift-node5:
#     ip: 192.0.2.8
#     container_vars:
#       swift_vars:
#         storage_ip: 198.51.100.8
#         repl_ip: 203.0.113.8
#         zone: 4
#         region: 3
#         weight: 200
#         groups:
#           - account
#           - container
#           - silver
#         drives:
```

(continues on next page)

(continued from previous page)

```
#            - name: sdb
#              weight: 75
#              groups:
#                - gold
#            - name: sdc
#            - name: sdd
#            - name: sde
#            - name: sdf
```

1. Ensure the `swift.yml` is in the `/etc/openstack_deploy/conf.d/` folder.

Home OpenStack-Ansible Swift

1.3 Integrate with the Image Service (glance)

As an option, you can create images in Image Service (glance) and store them using Object Storage (swift).

If there is an existing glance backend (for example, cloud files) but you want to add swift to use as the glance backend, you can re-add any images from glance after moving to swift. Images are no longer available if there is a change in the glance variables when you begin using swift.

Procedureă5.3.ăIntegrating Object Storage with Image Service

This procedure requires the following:

- Object Storage v2.2.0

1. Update the glance options in the `/etc/openstack_deploy/user_variables.yml` file:

```
# Glance Options
glance_default_store: swift
glance_swift_store_auth_address: '{{ keystone_service_internalurl }}'
glance_swift_store_container: glance_images
glance_swift_store_endpoint_type: internalURL
glance_swift_store_key: '{{ glance_service_password }}'
glance_swift_store_region: RegionOne
glance_swift_store_user: 'service:glance'
```

   - `glance_default_store`: Set the default store to `swift`.
   - `glance_swift_store_auth_address`: Set to the local authentication address using the `'{{ keystone_service_internalurl }}'` variable.
   - `glance_swift_store_container`: Set the container name.
   - `glance_swift_store_endpoint_type`: Set the endpoint type to `internalURL`.
   - `glance_swift_store_key`: Set the glance password using the `{{ glance_service_password }}` variable.
   - `glance_swift_store_region`: Set the region. The default value is `RegionOne`.
   - `glance_swift_store_user`: Set the tenant and user name to `'service:glance'`.

2. Rerun the glance configuration plays.
3. Run the glance playbook:

```
# cd /opt/openstack-ansible/playbooks
# openstack-ansible os-glance-install.yml --tags "glance-config"
```

Home OpenStack-Ansible Swift

1.4 Add to existing deployment

Complete the following procedure to deploy swift on an existing deployment.

1. The section called Configure and mount storage devices
2. The section called Configure an Object Storage deployment
3. Optionally, allow all keystone users to use swift by setting `swift_allow_all_users` in the `user_variables.yml` file to `True`. Any users with the `_member_` role (all authorized keystone users) can create containers and upload objects to swift.

   If this value is `False`, by default only users with the `admin` role or role set in `swift_operator_role` can create containers or manage tenants.

   When the backend type for the glance is set to `swift`, glance can access the swift cluster regardless of whether this value is `True` or `False`.
4. Run the swift play:

```
# cd /opt/openstack-ansible/playbooks
# openstack-ansible os-swift-install.yml
```

Home OpenStack-Ansible Swift

1.5 Storage policies

Storage policies allow segmenting the cluster for various purposes through the creation of multiple object rings. Using policies, different devices can belong to different rings with varying levels of replication. By supporting multiple object rings, swift can segregate the objects within a single cluster.

Use storage policies for the following situations:

- Differing levels of replication: A provider may want to offer 2x replication and 3x replication, but does not want to maintain two separate clusters. They can set up a 2x policy and a 3x policy and assign the nodes to their respective rings.
- Improving performance: Just as solid state drives (SSD) can be used as the exclusive members of an account or database ring, an SSD-only object ring can be created to implement a low-latency or high performance policy.
- Collecting nodes into groups: Different object rings can have different physical servers so that objects in specific storage policies are always placed in a specific data center or geography.
- Differing storage implementations: A policy can be used to direct traffic to collected nodes that use a different disk file (for example: Kinetic, GlusterFS).

Most storage clusters do not require more than one storage policy. The following problems can occur if using multiple storage policies per cluster:

- Creating a second storage policy without any specified drives (all drives are part of only the account, container, and default storage policy groups) creates an empty ring for that storage policy.
- Only use a non-default storage policy if specified when creating a container, using the `X-Storage-Policy: <policy-name>` header. After creating the container, it uses the storage policy. Other containers continue using the default or another specified storage policy.

For more information about storage policies, see: Storage Policies

Object Storage (swift) is a multi-tenant Object Storage system. It is highly scalable, can manage large amounts of unstructured data, and provides a RESTful HTTP API.

The following procedure describes how to set up storage devices and modify the Object Storage configuration files to enable swift usage.

1. The section called Configure and mount storage devices
2. The section called Configure an Object Storage deployment
3. Optionally, allow all Identity (keystone) users to use swift by setting `swift_allow_all_users` in the `user_variables.yml` file to `True`. Any users with the `_member_` role (all authorized keystone users) can create containers and upload objects to Object Storage.

   If this value is `False`, then by default, only users with the admin role or role set in `swift_operator_role` are allowed to create containers or manage tenants.

   When the backend type for the Image Service (glance) is set to `swift`, glance can access the swift cluster regardless of whether this value is `True` or `False`.

1.6 Overview

Object Storage (swift) is configured using the `/etc/openstack_deploy/conf.d/swift.yml` file and the `/etc/openstack_deploy/user_variables.yml` file.

When installing swift, use the group variables in the `/etc/openstack_deploy/conf.d/swift.yml` file for the Ansible playbooks. Some variables cannot be changed after they are set, while some changes require re-running the playbooks. The values in the `swift_hosts` section supersede values in the `swift` section.

To view the configuration files, including information about which variables are required and which are optional, see AppendixăA, *OSA Example test environment configuration*.

CHAPTER

TWO

DEFAULT VARIABLES

```
# Enable/Disable Telemetry projects
swift_ceilometer_enabled: False
swift_gnocchi_enabled: False

## Verbosity Options
debug: False

# Set the host which will execute the shade modules
# for the service setup. The host must already have
# clouds.yaml properly configured.
swift_service_setup_host: "{{ openstack_service_setup_host | default(
↪'localhost') }}"
swift_service_setup_host_python_interpreter: "{{ openstack_service_setup_
↪host_python_interpreter | default((swift_service_setup_host == 'localhost
↪') | ternary(ansible_playbook_python, ansible_python['executable'])) }}"

# Set the package install state for distribution and pip packages
# Options are 'present' and 'latest'
swift_package_state: "latest"
swift_pip_package_state: "latest"

# Set installation method.
swift_install_method: "source"
swift_venv_python_executable: "{{ openstack_venv_python_executable |␣
↪default('python2') }}"

# Git repo details for swift
swift_git_repo: https://opendev.org/openstack/swift
swift_git_install_branch: master

swift_upper_constraints_url: "{{ requirements_git_url | default('https://
↪releases.openstack.org/constraints/upper/' ~ requirements_git_install_
↪branch | default('master')) }}"
swift_git_constraints:
  - "git+{{ swift_git_repo }}@{{ swift_git_install_branch }}#egg=swift"
  - "--constraint {{ swift_upper_constraints_url }}"

swift_pip_install_args: "{{ pip_install_options | default('') }}"

# Name of the virtual env to deploy into
swift_venv_tag: "{{ venv_tag | default('untagged') }}"
swift_bin: "{{ _swift_bin }}"
```

(continues on next page)

(continued from previous page)

```
# Set the full path to the swift recon cron
recon_cron_path: "{{ swift_bin }}/swift-recon-cron"


## Swift User / Group
swift_system_user_name: swift
swift_system_group_name: swift
swift_system_shell: /bin/bash
swift_system_comment: swift system user
swift_system_home_folder: "/var/lib/{{ swift_system_user_name }}"

## Auth token
swift_delay_auth_decision: true

## Swift middleware
# NB: The order is important!
swift_middleware_list:
  - catch_errors
  - gatekeeper
  - healthcheck
  - proxy-logging
  - "{% if swift_ceilometer_enabled | bool %}ceilometer{% endif %}"
  - cache
  - container_sync
  - bulk
  - tempurl
  - ratelimit
  - authtoken
  - keystoneauth
  - staticweb
  - copy
  - container-quotas
  - account-quotas
  - slo
  - dlo
  - versioned_writes
  - proxy-logging
  - proxy-server

# Setup tempauth users list (user_<account>_<username> = <password> <roles>
↪)
swift_tempauth_users:
  - "user_admin_admin = admin .admin .reseller_admin"

## Swift default ports
swift_proxy_bind_address: "{{ openstack_service_bind_address | default('0.
↪0.0.0') }}"
swift_proxy_port: "8080"
# You can change the object, container, account ports.
# This will update the ring, on the next playbook run,
# without requiring a rebalance.
# NB: There is service downtime, during the run, between
# the service restart and the ring updating.
swift_object_port: "6000"
swift_container_port: "6001"
swift_account_port: "6002"
```

(continues on next page)

(continued from previous page)

```
# Default swift ring settings:
swift_default_replication_number: 3
swift_default_min_part_hours: 1
swift_default_host_zone: 0
swift_default_host_region: 1
swift_default_drive_weight: 100

## Swift service defaults
swift_service_name: swift
swift_service_user_name: swift
swift_service_project_name: service
swift_service_project_domain_id: "default"
swift_service_project_domain_name: "Default"
swift_service_user_domain_id: "default"
swift_service_role_name: "admin"
swift_service_type: object-store
swift_service_proto: http
swift_service_publicuri_proto: "{{ openstack_service_publicuri_proto |␣
↪default(swift_service_proto) }}"
swift_service_adminuri_proto: "{{ openstack_service_adminuri_proto |␣
↪default(swift_service_proto) }}"
swift_service_internaluri_proto: "{{ openstack_service_internaluri_proto |␣
↪default(swift_service_proto) }}"
swift_service_description: "Object Storage Service"
swift_service_publicuri: "{{ swift_service_publicuri_proto }}://{{␣
↪external_lb_vip_address }}:{{ swift_proxy_port }}"
swift_service_publicurl: "{{ swift_service_publicuri }}/v1/AUTH_%(tenant_
↪id)s"
swift_service_adminuri: "{{ swift_service_adminuri_proto }}://{{ internal_
↪lb_vip_address }}:{{ swift_proxy_port }}"
swift_service_adminurl: "{{ swift_service_adminuri }}/v1/AUTH_%(tenant_id)s
↪"
swift_service_internaluri: "{{ swift_service_internaluri_proto }}://{{␣
↪internal_lb_vip_address }}:{{ swift_proxy_port }}"
swift_service_internalurl: "{{ swift_service_internaluri }}/v1/AUTH_
↪%(tenant_id)s"
swift_service_region: RegionOne
statsd_host:
statsd_port: 8125
statsd_default_sample_rate: 1.0
statsd_sample_rate_factor: 1.0
statsd_metric_prefix:

# Set the file limits
swift_hard_open_file_limits: 10240
swift_soft_open_file_limits: 4096
swift_max_file_limits: "{{ swift_hard_open_file_limits * 24 }}"

## Keystone authentication middleware
swift_keystone_auth_plugin: "{{ swift_keystone_auth_type }}"
swift_keystone_auth_type: "password"

swift_dispersion_user: dispersion
swift_dispersion_user_domain_name: "Default"
```

(continues on next page)

(continued from previous page)

```
swift_operator_role: swiftoperator
swift_allow_versions: True
# This will allow all users to create containers and upload to swift if␣
↪set to True
swift_allow_all_users: False
# If you want to regenerate the swift keys, on a run, for rsync purposes␣
↪set this var to True otherwise keys will be generated on the first run␣
↪and not regenerated each run.
swift_recreate_keys: False
swift_sorting_method: shuffle
# Set the fallocate_reserve value which will reserve space and fail on␣
↪PUTs above this value in bytes (Default 10GB)
swift_fallocate_reserve: "1%"
swift_account_fallocate_reserve: "{{ swift_fallocate_reserve }}"
swift_container_fallocate_reserve: "{{ swift_fallocate_reserve }}"
swift_object_fallocate_reserve: "{{ swift_fallocate_reserve }}"
# Set this to true to disable fallocate
swift_disable_fallocate: false
swift_account_disable_fallocate: "{{ swift_disable_fallocate }}"
swift_container_disable_fallocate: "{{ swift_disable_fallocate }}"
swift_object_disable_fallocate: "{{ swift_disable_fallocate }}"

# This variable will protect against changing swift_hash_path_* variables␣
↪unintentionally.
# If you wish to change them intentionally set the swift_force_change_
↪hashes variable to True.
swift_force_change_hashes: False

## Swift ceilometer variables
swift_reselleradmin_role: ResellerAdmin

## Oslo Messaging

# Notify
swift_oslomsg_notify_host_group: "{{ oslomsg_notify_host_group | default(
↪'rabbitmq_all') }}"
swift_oslomsg_notify_setup_host: "{{ (swift_oslomsg_notify_host_group in␣
↪groups) | ternary(groups[swift_oslomsg_notify_host_group][0], 'localhost
↪') }}"
swift_oslomsg_notify_transport: "{{ oslomsg_notify_transport | default(
↪'rabbit') }}"
swift_oslomsg_notify_servers: "{{ oslomsg_notify_servers | default('127.0.
↪0.1') }}"
swift_oslomsg_notify_port: "{{ oslomsg_notify_port | default('5672') }}"
swift_oslomsg_notify_use_ssl: "{{ oslomsg_notify_use_ssl | default(False) }
↪}"
swift_oslomsg_notify_userid: swift
swift_oslomsg_notify_vhost: /swift

## General Swift configuration
# We are not capping the default value for these swift variables which␣
↪define
# the number of worker threads for each of the swift services (except the␣
↪swift
# proxy workers when proxy is in a container) because of the performace␣
↪impact
```

(continues on next page)

(continued from previous page)

```
# that may be seen due to capping worker threads for swift services.
# We would like to calculate the default value using vCPUs for good␣
↪performance
# of swift services.

# If ``swift_account_server_replicator_workers`` is unset the system will␣
↪use half the number
# of available VCPUS to compute the number of api workers to use.
# swift_account_server_replicator_workers: 16

# If ``swift_server_replicator_workers`` is unset the system will use half␣
↪the number
# of available VCPUS to compute the number of api workers to use.
# swift_server_replicator_workers: 16

# If ``swift_object_replicator_workers`` is unset the system will use half␣
↪the number
# of available VCPUS to compute the number of api workers to use.
# swift_object_replicator_workers: 16

# If ``swift_account_server_workers`` is unset the system will use half the␣
↪number
# of available VCPUS to compute the number of api workers to use.
# swift_account_server_workers: 16

# If ``swift_container_server_workers`` is unset the system will use half␣
↪the number
# of available VCPUS to compute the number of api workers to use.
# swift_container_server_workers: 16

# If ``swift_object_server_workers`` is unset the system will use half the␣
↪number
# of available VCPUS to compute the number of api workers to use.
# swift_object_server_workers: 16

# If ``swift_proxy_server_workers`` is unset the system will use half the␣
↪number
# of available VCPUS to compute the number of api workers to use. Capping␣
↪this
# value at 16 if the swift proxy is in a container and user did not define
# this variable.
swift_proxy_server_workers_max: 16
swift_proxy_server_workers_not_capped: "{{ [(ansible_processor_vcpus//
↪ansible_processor_threads_per_core)|default(1), 1] | max * 2 }}"
swift_proxy_server_workers_capped: "{{ [swift_proxy_server_workers_max,␣
↪swift_proxy_server_workers_not_capped|int] | min }}"
swift_proxy_server_workers: "{{ (inventory_hostname == physical_host) |␣
↪ternary(swift_proxy_server_workers_not_capped, swift_proxy_server_
↪workers_capped) }}"

# These are the storage addresses used to define the networks for swift␣
↪storage and replication
# These are calculated by the tasks based on the "storage_network" and
↪"replication_network" values
# set in the swift variables, if you set these per host the value won't be␣
↪calculated.
```

(continues on next page)

(continued from previous page)

```
# Setting swift_vars.storage_ip or swift_vars.repl_ip will take precedence.
# If none are set it will default to the "ansible_host" value.
# swift_storage_address: 127.0.0.1
# swift_replication_address: 127.0.0.1

# This var is calculated by the play itself, and should not need to be set
# It is defaulted for the benefit of the swift_proxy host which needs it
# for the swift-init-systemd.j2 template file.
swift_dedicated_replication: False

swift_service_in_ldap: false

# Basic swift configuration for the cluster
swift: {}
swift_vars: {}
swift_proxy_vars: {}

# Example basic swift configuration for the cluster
# swift:
#   part_power: 8
#   storage_network: 'br-storage'
#   replication_network: 'br-storage'
#   drives:
#     - name: swift1.img
#     - name: swift2.img
#     - name: swift3.img
#   mount_point: /srv
#   storage_policies:
#     - policy:
#         name: default
#         index: 0
#         default: True

# Set rsync max_connections vars
swift_max_rsync_connections: 4
swift_account_max_rsync_connections: "{{ swift_max_rsync_connections }}"
swift_container_max_rsync_connections: "{{ swift_max_rsync_connections }}"
swift_object_max_rsync_connections: "{{ swift_max_rsync_connections }}"

# Set Swift to use rsync module per object server drive
swift_rsync_module_per_drive: False

# Set Swift to use reverse lookup - requires name resolution or hosts␣
↪entries
swift_rsync_reverse_lookup: False

# Set the managed regions as a list of swift regions to manage
# Use for global clusters, default when not set is all regions.
# swift_managed_regions:
#   - 1
#   - 2

# swift_do_setup and swift_do_sync control which parts of the swift
# role get run. You should never need to adjust these, they are set
# within the swift-setup and swift-sync roles to ensure only the
# appropriate tasks within the os-swift role are run.
```

(continues on next page)

(continued from previous page)

```
swift_do_setup: True
swift_do_sync: True

# Example swift_container_sync_realms to specify container_sync realms
# This can exist for multiple realms (in a list)
# swift_container_sync_realms:
#   - name: realm1
#     # You may want to put swift_realm_keyx in user_secrets.yml or␣
↪ansible-vault
#     # Otherwise specify it manually below.
#     key1: {{ swift_realm_key1 }}
#     # key2 is optional and used for rotating/deprecated keys
#     key2: {{ swift_realm_key2 }}
#     clustername1: https://<cluster1-ip>/v1
#     clustername2: https://<cluster2-ip>/v1

swift_pip_packages:
  - ceilometermiddleware
  - cryptography
  - dnspython
  - ecdsa
  - keystonemiddleware
  - osprofiler
  - pyeclib
  - python-keystoneclient
  - pymemcache
  - python-memcached
  - python-swiftclient
  - swift
  - systemd-python

# Memcached override
swift_memcached_servers: "{{ memcached_servers }}"

swift_account_replicator_init_overrides: {}
swift_account_replicator_server_init_overrides: {}
swift_account_server_init_overrides: {}
swift_account_auditor_init_overrides: {}
swift_account_reaper_init_overrides: {}
swift_container_replicator_init_overrides: {}
swift_container_replicator_server_init_overrides: {}
swift_container_server_init_overrides: {}
swift_container_auditor_init_overrides: {}
swift_container_sync_init_overrides: {}
swift_container_updater_init_overrides: {}
swift_container_reconciler_init_overrides: {}
swift_object_replicator_init_overrides: {}
swift_object_replicator_server_init_overrides: {}
swift_object_server_init_overrides: {}
swift_object_auditor_init_overrides: {}
swift_object_updater_init_overrides: {}
swift_object_expirer_init_overrides: {}
swift_object_reconstructor_init_overrides: {}
swift_proxy_server_init_overrides: {}

# Default options applied to all swift service units
```

(continues on next page)

(continued from previous page)

```
swift_service_defaults:
  Service:
    LimitNOFILE: "{{ swift_soft_open_file_limits }}:{{ swift_hard_open_
↪file_limits }}"
    Environment:
      ? "PYPY_GC_MIN={{ swift_pypy_gc_min }}"
      ? "PYPY_GC_MAX={{ swift_pypy_gc_max }}"

swift_services:
  swift-proxy-server:
    group: swift_proxy
    service_name: "swift-proxy-server"
    execstarts: "{{ swift_bin }}/swift-proxy-server /etc/swift/proxy-
↪server/proxy-server.conf"
    init_config_overrides: "{{ swift_proxy_server_init_overrides }}"
    start_order: 1
  swift-account-server:
    group: swift_acc
    service_name: "swift-account-server"
    execstarts: "{{ swift_bin }}/swift-account-server /etc/swift/account-
↪server/account-server.conf"
    init_config_overrides: "{{ swift_account_server_init_overrides }}"
    start_order: 2
  swift-account-replicator-server:
    group: swift_acc
    service_name: "swift-account-replicator-server"
    execstarts: "{{ swift_bin }}/swift-account-server /etc/swift/account-
↪server/account-server-replicator.conf"
    service_en: "{{ swift_dedicated_replication | bool }}"
    init_config_overrides: "{{ swift_account_replicator_server_init_
↪overrides }}"
    start_order: 3
  swift-container-server:
    group: swift_cont
    service_name: swift-container-server
    execstarts: "{{ swift_bin }}/swift-container-server /etc/swift/
↪container-server/container-server.conf"
    init_config_overrides: "{{ swift_container_server_init_overrides }}"
    start_order: 4
  swift-container-replicator-server:
    group: swift_cont
    service_name: "swift-container-replicator-server"
    execstarts: "{{ swift_bin }}/swift-container-server /etc/swift/
↪container-server/container-server-replicator.conf"
    service_en: "{{ swift_dedicated_replication | bool }}"
    init_config_overrides: "{{ swift_container_replicator_server_init_
↪overrides }}"
    start_order: 5
  swift-object-server:
    group: swift_obj
    service_name: swift-object-server
    execstarts: "{{ swift_bin }}/swift-object-server /etc/swift/object-
↪server/object-server.conf"
    init_config_overrides: "{{ swift_object_server_init_overrides }}"
    start_order: 6
  swift-object-replicator-server:
```

(continues on next page)

(continued from previous page)

```
    group: swift_obj
    service_name: "swift-object-replicator-server"
    execstarts: "{{ swift_bin }}/swift-object-server /etc/swift/object-
↪server/object-server-replicator.conf"
    service_en: "{{ swift_dedicated_replication | bool }}"
    init_config_overrides: "{{ swift_object_replicator_server_init_
↪overrides }}"
    start_order: 7

  swift-account-auditor:
    group: swift_acc
    service_name: swift-account-auditor
    execstarts: "{{ swift_bin }}/swift-account-auditor {{ swift_dedicated_
↪replication | ternary('/etc/swift/account-server/account-server-
↪replicator.conf', '/etc/swift/account-server/account-server.conf') }}"
    init_config_overrides: "{{ swift_account_auditor_init_overrides }}"
    start_order: 8
  swift-account-reaper:
    group: swift_acc
    service_name: swift-account-reaper
    execstarts: "{{ swift_bin }}/swift-account-reaper /etc/swift/account-
↪server/account-server.conf"
    init_config_overrides: "{{ swift_account_reaper_init_overrides }}"
    start_order: 9
  swift-account-replicator:
    group: swift_acc
    service_name: swift-account-replicator
    execstarts: "{{ swift_bin }}/swift-account-replicator {{ swift_
↪dedicated_replication | ternary('/etc/swift/account-server/account-
↪server-replicator.conf', '/etc/swift/account-server/account-server.conf
↪') }}"
    init_config_overrides: "{{ swift_account_replicator_init_overrides }}"
    start_order: 10

  swift-container-auditor:
    group: swift_cont
    service_name: "swift-container-auditor"
    execstarts: "{{ swift_bin }}/swift-container-auditor {{ swift_
↪dedicated_replication | ternary('/etc/swift/container-server/container-
↪server-replicator.conf', '/etc/swift/container-server/container-server.
↪conf') }}"
    init_config_overrides: "{{ swift_container_auditor_init_overrides }}"
    start_order: 11
  swift-container-reconciler:
    group: swift_cont
    service_name: "swift-container-reconciler"
    execstarts: "{{ swift_bin }}/swift-container-reconciler /etc/swift/
↪container-server/container-reconciler.conf"
    init_config_overrides: "{{ swift_container_reconciler_init_overrides }}
↪"
    start_order: 12
  swift-container-replicator:
    group: swift_cont
    service_name: "swift-container-replicator"
    execstarts: "{{ swift_bin }}/swift-container-replicator {{ swift_
↪dedicated_replication | ternary('/etc/swift/container-server/container-
↪server-replicator.conf', '/etc/swift/container-server/container-server.
↪conf') }}"
```

(continues on next page)

(continued from previous page)

```
    init_config_overrides: "{{ swift_container_replicator_init_overrides }}
↪"
    start_order: 13
  swift-container-sync:
    group: swift_cont
    service_name: "swift-container-sync"
    execstarts: "{{ swift_bin }}/swift-container-sync /etc/swift/container-
↪server/container-server.conf"
    init_config_overrides: "{{ swift_container_sync_init_overrides }}"
    start_order: 14
  swift-container-updater:
    group: swift_cont
    service_name: "swift-container-updater"
    execstarts: "{{ swift_bin }}/swift-container-updater /etc/swift/
↪container-server/container-server.conf"
    init_config_overrides: "{{ swift_container_updater_init_overrides }}"
    start_order: 15

  swift-object-auditor:
    group: swift_obj
    service_name: "swift-object-auditor"
    execstarts: "{{ swift_bin }}/swift-object-auditor {{ swift_dedicated_
↪replication | ternary('/etc/swift/object-server/object-server-replicator.
↪conf', '/etc/swift/object-server/object-server.conf') }}"
    init_config_overrides: "{{ swift_object_auditor_init_overrides }}"
    start_order: 16
  swift-object-expirer:
    group: swift_obj
    service_name: "swift-object-expirer"
    execstarts: "{{ swift_bin }}/swift-object-expirer /etc/swift/object-
↪server/object-expirer.conf"
    init_config_overrides: "{{ swift_object_expirer_init_overrides }}"
    start_order: 17
  swift-object-reconstructor:
    group: swift_obj
    service_name: "swift-object-reconstructor"
    execstarts: "{{ swift_bin }}/swift-object-reconstructor {{ swift_
↪dedicated_replication | ternary('/etc/swift/object-server/object-server-
↪replicator.conf', '/etc/swift/object-server/object-server.conf') }}"
    init_config_overrides: "{{ swift_object_reconstructor_init_overrides }}
↪"
    start_order: 18
  swift-object-replicator:
    group: swift_obj
    service_name: "swift-object-replicator"
    execstarts: "{{ swift_bin }}/swift-object-replicator {{ swift_
↪dedicated_replication | ternary('/etc/swift/object-server/object-server-
↪replicator.conf', '/etc/swift/object-server/object-server.conf') }}"
    init_config_overrides: "{{ swift_object_replicator_init_overrides }}"
    start_order: 19
  swift-object-updater:
    group: swift_obj
    service_name: "swift-object-updater"
    execstarts: "{{ swift_bin }}/swift-object-updater /etc/swift/object-
↪server/object-server.conf"
    init_config_overrides: "{{ swift_object_updater_init_overrides }}"
```

(continues on next page)

(continued from previous page)

```
    start_order: 20

# Set to True to reset the clock on the last time a rebalance happened,
# circumventing the min_part_hours check.
# USE WITH EXTREME CAUTION
# If you run the swift playbook with this option enabled, before a swift
# replication pass completes, you may introduce unavailability in your
# cluster. This has an end-user impact.
swift_pretend_min_part_hours_passed: False

# Set this option to enable or disable the pypy interpreter for swift
swift_pypy_enabled: false
swift_pypy_archive:
  url: "https://bitbucket.org/pypy/pypy/downloads/pypy2-v5.9.0-linux64.tar.
↪bz2"
  sha256: "790febd4f09e22d6e2f81154efc7dc4b2feec72712aaf4f82aa91b550abb4b48
↪"
swift_pypy_version: "{{ swift_pypy_archive['url'] | basename | replace('.
↪tar.bz2', '') }}"
swift_pypy_env: "/opt/pypy-runtime/{{ swift_pypy_version }}/bin/pypy"
# Set the Garbage Collection (GC) options for pypy if you would like to␣
↪tune these
# More info on pypy garbage collection can be found here:
# http://doc.pypy.org/en/latest/gc_info.html
swift_pypy_gc_min: "15M"
swift_pypy_gc_max: "1GB"

## Tunable overrides
swift_swift_conf_overrides: {}
swift_swift_dispersion_conf_overrides: {}
swift_proxy_server_conf_overrides: {}
swift_account_server_conf_overrides: {}
swift_account_server_replicator_conf_overrides: {}
swift_container_server_conf_overrides: {}
swift_container_reconciler_conf_overrides: {}
swift_container_server_replicator_conf_overrides: {}
swift_container_sync_realms_conf_overrides: {}
swift_drive_audit_conf_overrides: {}
swift_internal_client_conf_overrides: {}
swift_object_server_conf_overrides: {}
swift_object_expirer_conf_overrides: {}
swift_object_server_replicator_conf_overrides: {}
swift_memcache_conf_overrides: {}
```

CHAPTER

THREE

EXAMPLE PLAYBOOK

```
---
- name: Install swift server
  hosts: swift_all
  user: root
  roles:
    - { role: "os_swift", tags: [ "os-swift" ] }
  vars:
    external_lb_vip_address: 172.16.24.1
    internal_lb_vip_address: 192.168.0.1
```

CHAPTER

FOUR

DEPENDENCIES

This role needs pip >= 7.1 installed on the target host.

CHAPTER

FIVE

TAGS

This role supports two tags: `swift-install` and `swift-config`.

The `swift-install` tag can be used to install the software.

The `swift-config` tag can be used to maintain configuration of the service, and do runtime operations.