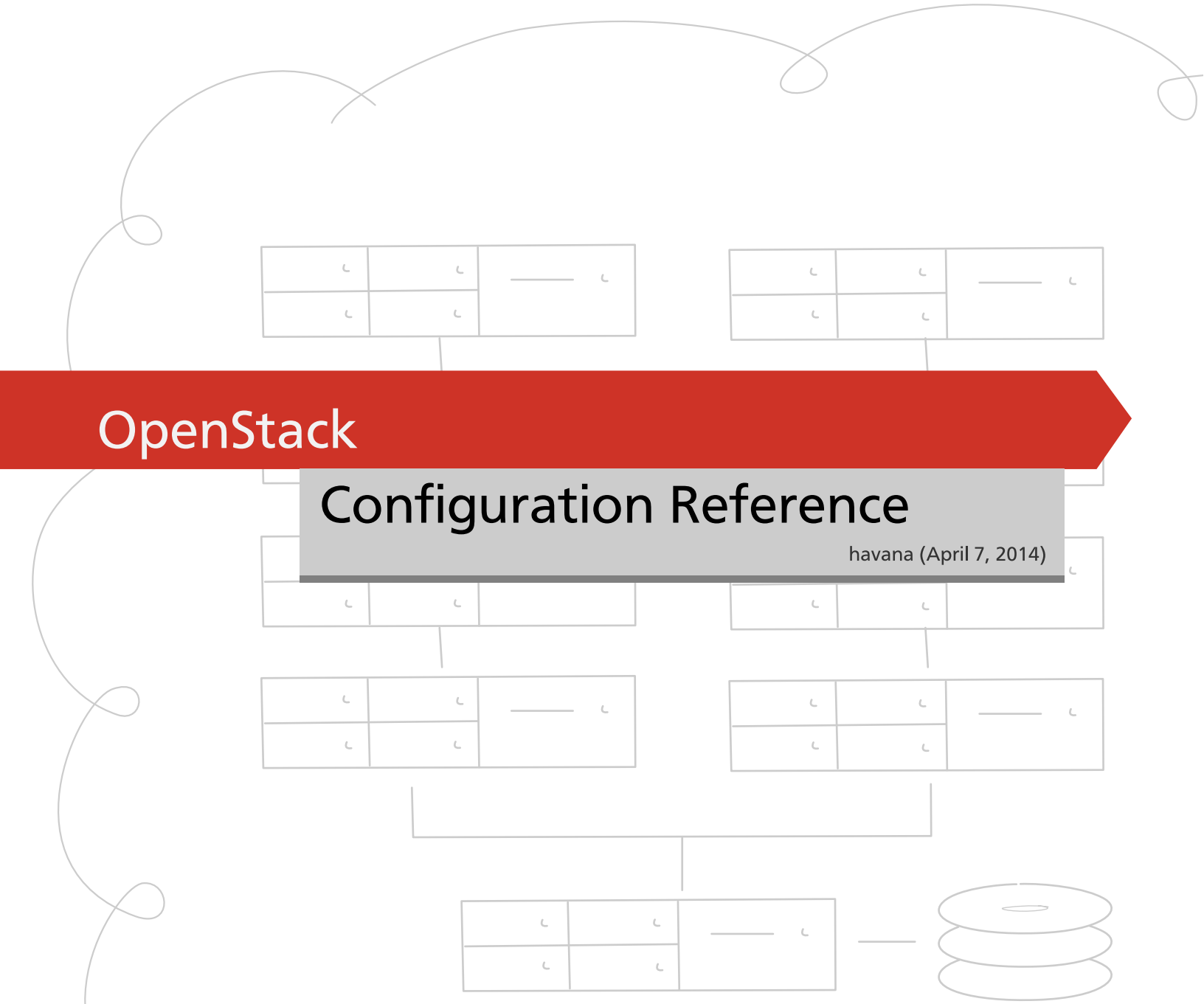


OpenStack

Configuration Reference

havana (April 7, 2014)

docs.openstack.org

OpenStack Configuration Reference

havana (2014-04-07)

This document is for system administrators who want to look up configuration options. It contains lists of configuration options available with OpenStack and uses auto-generation to generate options and the descriptions from the code for each project. It includes sample configuration files.

Table of Contents

List of Figures

List of Tables

OpenStack configuration overview

OpenStack is a collection of open source project components that enable setting up cloud services. Each component uses similar configuration techniques and a common framework for INI file options.

This guide pulls together multiple references and configuration options for the following OpenStack components:

- OpenStack Identity
- OpenStack Compute
- OpenStack Image Service
- OpenStack Networking
- OpenStack Dashboard
- OpenStack Object Storage
- OpenStack Block Storage

Document change history

This version of the guide replaces and obsoletes all previous versions. The following table describes the most recent changes:

Revision Date	Summary of Changes
October 17, 2013	• Havana release.
August 16, 2013	• Moves Block Storage driver configuration information from the *Block Storage Administration Guide* to this reference.
June 10, 2013	• Initial creation of Configuration Reference.

1. Identity Service

Table of Contents

The OpenStack Identity Service has several configuration options.

Identity Service configuration files

keystone.conf

The Identity Service `/etc/keystone/keystone.conf` configuration file is an INI-format file with sections.

The `[DEFAULT]` section configures general configuration values.

Specific sections, such as the `[sql]` and `[ec2]` sections, configure individual services.

Table 1.1. keystone.conf file sections

Section	Description
`[DEFAULT]`	General configuration.
`[sql]`	Optional storage back-end configuration.
`[ec2]`	Amazon EC2 authentication driver configuration.
`[s3]`	Amazon S3 authentication driver configuration.
`[identity]`	Identity Service system driver configuration.
`[catalog]`	Service catalog driver configuration.
`[token]`	Token driver configuration.
`[policy]`	Policy system driver configuration for RBAC.
`[signing]`	Cryptographic signatures for PKI based tokens.
`[ssl]`	SSL configuration.

When you start the Identity Service, you can use the `--config-file` parameter to specify a configuration file.

If you do not specify a configuration file, the Identity Service looks for the `keystone.conf` configuration file in these directories in this order:

1. `~/.keystone`

2. `~/`

3. `/etc/keystone`

4. `/etc`

keystone-paste.ini	The `/etc/keystone/keystone-paste.ini` file configures the Identity Service WSGI middleware pipeline.

Certificates for PKI

PKI stands for Public Key Infrastructure. Tokens are documents, cryptographically signed using the X509 standard. In order to work correctly token generation requires a public/private key pair. The public key must be signed in an X509 certificate, and the certificate used to sign it must be available as Certificate Authority (CA) certificate. These files can be generated either using the **keystone-manage** utility, or externally generated. The files need to be in the locations specified by the top level Identity Service configuration file `keystone.conf` as specified in the above section. Additionally, the private key should only be readable by the system user that will run the Identity Service.

Warning

The certificates can be world readable, but the private key cannot be. The private key should only be readable by the account that is going to sign tokens. When generating files with the **keystone-mange pki_setup** command, your best option is to run as the pki user. If you run nova-manage as root, you can append –keystone-user and –keystone-group parameters to set the username and group keystone is going to run under.

The values that specify where to read the certificates are under the `[signing]` section of the configuration file. The configuration values are:

- `token_format` - Determines the algorithm used to generate tokens. Can be either `UUID` or `PKI`. Defaults to `PKI`.
- `certfile` - Location of certificate used to verify tokens. Default is `/etc/keystone/ssl/certs/signing_cert.pem`.
- `keyfile` - Location of private key used to sign tokens. Default is `/etc/keystone/ssl/private/signing_key.pem`.
- `ca_certs` - Location of certificate for the authority that issued the above certificate. Default is `/etc/keystone/ssl/certs/ca.pem`.
- `key_size` - Default is `1024`.
- `valid_days` - Default is `3650`.
- `ca_password` - Password required to read the `ca_file`. Default is `None`.

If `token_format=UUID`, a typical token looks like `53f7f6ef0cc344b5be706bcc8b1479e1`. If `token_format=PKI`, a typical token is a much longer string, such as:

```
MIIKtgYJKoZIhvcNAQcCoIIKpzCCCqMCAQExCTAHBgUrDgMCGjCCCY8GCSqGSIb3DQEHAaCCCYAEggl8eyJhY2Nlc3Mi
MFQxNTo1MjowNi43MzMxOTgiLCAiZXhwaXJlcyI6ICIyMDEzLTA1LTMxVDE1OjUyOjA2WiIsICJpZCI6ICJwbGFjZWhv
bCwgImVuYWJsZWQiOiB0cnVlLCAiaWQiOiAiYzJjNTliNGQzZDI4NGQ4ZmEwOWYxNjljYjE4MDBlMDYiLCAibmFtZSI6
b2ludHMiOiBbeyJhZG1pblVSTCI6ICJodHRwOi8vMTkyLjE2OC4yNy4xMDA6ODc3NC92Mi9jMmM1OWI0ZDNkMjg0ZDhm
T25lIiwgImludGVybmFsVVJMIjogImh0dHA6Ly8xOTIuMTY4LjI3LjEwMDo4Nzc0L3YyL2MyYzU5YjRkM2QyODRkOGZh
ODRhNGNhZTk3MmViNzcwOTgzZTJlIiwgInB1YmxpY1VSTCI6ICJodHRwOi8vMTkyLjE2OC4yNy4xMDA6ODc3NC92Mi9j
ImVuZHBvaW50c19saW5rcyI6IFtdLCAidHlwZSI6ICJjb21wdXRlIiwgIm5hbWUiOiAibm92YSJ9LCB7ImVuZHBvaW50
LjEwMDozMzMzIiwgInJlZ2lvbiI6ICJSZWdpb25PbmUiLCAiaW50ZXJuYWxVUkwiOiAiaHR0cDovLzE5Mi4xNjguMjcu
MGU2YWNlNDU4NjZmMzAiLCAicHVibGljVVJMIjogImh0dHA6Ly8xOTIuMTY4LjI3LjEwMDozMzMzIn1dLCAiZW5kcG9p
OiAiczMifSwgeyJlbmRwb2ludHMiOiBbeyJhZG1pblVSTCI6ICJodHRwOi8vMTkyLjE2OC4yNy4xMDA6OTI5MiIsICJy
Imh0dHA6Ly8xOTIuMTY4LjI3LjEwMDo5MjkyIiwgImlkIjogIjczODQzNTJhNTQ0MjQ1NzVhM2NkOTVkN2E0YzNjZGY1
MDA6OTI5MiJ9XSwgImVuZHBvaW50c19saW5rcyI6IFtdLCAidHlwZSI6ICJpbWFnZSIsICJuYW1lIjogImdsYW5jZSJ9
Ly8xOTIuMTY4LjI3LjEwMDo4Nzc2L3YxL2MyYzU5YjRkM2QyODRkOGZhMDlmMTY5Y2IxODAwZTA2IiwgInJlZ2lvbiI6
LzE5Mi4xNjguMjcuMTAwOjg3NzYvdjEvYzJjNTliNGQzZDI4NGQ4ZmEwOWYxNjljYjE4MDBlMDYiLCAiaWQiOiAiMzQ3
bGljVVJMIjogImh0dHA6Ly8xOTIuMTY4LjI3LjEwMDo4Nzc2L3YxL2MyYzU5YjRkM2QyODRkOGZhMDlmMTY5Y2IxODAw
IjogInZvbHVtZSIsICJuYW1lIjogImNpbmRlciJ9LCB7ImVuZHBvaW50cyI6IFt7ImFkbWluVVJMIjogImh0dHA6Ly8x
InJlZ2lvbiI6ICJSZWdpb25PbmUiLCAiaW50ZXJuYWxVUkwiOiAiaHR0cDovLzE5Mi4xNjguMjcuMTAwOjg3NzMvc2Vy
YWE1NDAzMDMzNzI5YzY3MjIiLCAicHVibGljVVJMIjogImh0dHA6Ly8xOTIuMTY4LjI3LjEwMDo4NzczL3NlcnZpY2Vz
eXBlIjogImVjMiIsICJuYW1lIjogImVjMiJ9LCB7ImVuZHBvaW50cyI6IFt7ImFkbWluVVJMIjogImh0dHA6Ly8xOTIu
ZWdpb25PbmUiLCAiaW50ZXJuYWxVUkwiOiAiaHR0cDovLzE5Mi4xNjguMjcuMTAwOjUwMDAvdjIuMCIsICJpZCI6ICJi
dWJsaWNVUkwiOiAiaHR0cDovLzE5Mi4xNjguMjcuMTAwOjUwMDAvdjIuMCJ9XSwgImVuZHBvaW50c19saW5rcyI6IFtd
b25lIn1dLCAidXNlciI6IHsidXNlcm5hbWUiOiAiZGVtbyIsICJyb2xlc19saW5rcyI6IFtdLCAiaWQiOiAiZTVhMTM3
OiBbeyJuYW1lIjogImFub3RoZXJyb2xlIn0sIHsibmFtZSI6ICJNZW1iZXIifV0sICJuYW1lIjogImRlbW8ifSwgIm1l
YWRiODM3NDVkYzQzNGJhMzk5ODllNjBjOTIzYWZhMjgiLCAiMzM2ZTFiNjE1N2Y3NGFmZGJhNWUwYTYwMWUwNjM5MmYi
zCB-AIBATBcMFcxCzAJBgNVBAYTAlVTMQ4wDAYD
VQQIEwVVbnNldDEOMAwGA1UEBxMFVW5zZXQxDjAMBgNVBAoTBVVuc2V0MRgwFgYDVQQDEw93d3cuZXhhbXBsZS5jb20C
nouriuiCgFayIqCssK3SVdhOMINiuJtqv0sE-wBDFiEj-Prcudqlz-n+6q7VgV4mwMPszz39-rwp
+P5l4AjrJasUm7FrO-4l02tPLaaZXU1gBQ1jUG5e5aL5jPDP08HbCWuX6wr-QQQB
SrWY8lF3HrTcJT23sZIleg==
```

Sign certificate issued by external CA

You can use a signing certificate issued by an external CA instead of generated by **keystone-manage**. However, certificate issued by external CA must satisfy the following conditions:

- all certificate and key files must be in Privacy Enhanced Mail (PEM) format
- private key files must not be protected by a password

When using signing certificate issued by an external CA, you do not need to specify `key_size`, `valid_days`, and `ca_password` as they will be ignored.

The basic workflow for using a signing certificate issued by an external CA involves:

1. Request Signing Certificate from External CA
2. Convert certificate and private key to PEM if needed
3. Install External Signing Certificate

Request a signing certificate from an external CA

One way to request a signing certificate from an external CA is to first generate a PKCS #10 Certificate Request Syntax (CRS) using OpenSSL CLI.

First create a certificate request configuration file (e.g. `cert_req.conf`):

```
[ req ]
default_bits            = 1024
default_keyfile         = keystonekey.pem
default_md              = sha1

prompt                  = no
distinguished_name      = distinguished_name

[ distinguished_name ]
countryName             = US
stateOrProvinceName     = CA
localityName            = Sunnyvale
organizationName        = OpenStack
organizationalUnitName  = Keystone
commonName              = Keystone Signing
emailAddress            = keystone@openstack.org
```

Then generate a CRS with OpenSSL CLI. **Do not encrypt the generated private key. Must use the -nodes option.**

For example:

```
$ openssl req -newkey rsa:1024 -keyout signing_key.pem -keyform PEM \
  -out signing_cert_req.pem -outform PEM -config cert_req.conf -nodes
```

If everything is successfully, you should end up with `signing_cert_req.pem` and `signing_key.pem`. Send `signing_cert_req.pem` to your CA to request a token signing certificate and make sure to ask the certificate to be in PEM format. Also, make sure your trusted CA certificate chain is also in PEM format.

Install an external signing certificate

Assuming you have the following already:

- `signing_cert.pem` - (Keystone token) signing certificate in PEM format
- `signing_key.pem` - corresponding (non-encrypted) private key in PEM format
- `cacert.pem` - trust CA certificate chain in PEM format

Copy the above to your certificate directory. For example:

```
# mkdir -p /etc/keystone/ssl/certs
# cp signing_cert.pem /etc/keystone/ssl/certs/
# cp signing_key.pem /etc/keystone/ssl/certs/
# cp cacert.pem /etc/keystone/ssl/certs/
# chmod -R 700 /etc/keystone/ssl/certs
```

Note

Make sure the certificate directory is only accessible by root.

If your certificate directory path is different from the default `/etc/keystone/ssl/certs`, make sure it is reflected in the `[signing]` section of the configuration file.

Configure the Identity Service with SSL

You can configure the Identity Service to support two-way SSL.

You must obtain the x509 certificates externally and configure them.

The Identity Service provides a set of sample certificates in the `examples/pki/certs` and `examples/pki/private` directories:

Certificate types

cacert.pem	Certificate Authority chain to validate against.
ssl_cert.pem	Public certificate for Identity Service server.
middleware.pem	Public and private certificate for Identity Service middleware/client.
cakey.pem	Private key for the CA.
ssl_key.pem	Private key for the Identity Service server.

Note

You can choose names for these certificates. You can also combine the public/private keys in the same file, if you wish. These certificates are provided as an example.

SSL configuration

To enable SSL with client authentication, modify the `[ssl]` section in the `etc/keystone.conf` file. The following SSL configuration example uses the included sample certificates:

```
[ssl]
enable = True
certfile = <path to keystone.pem>
keyfile = <path to keystonekey.pem>
ca_certs = <path to ca.pem>
cert_required = True
```

Options

- `enable`. True enables SSL. Default is False.
- `certfile`. Path to the Identity Service public certificate file.
- `keyfile`. Path to the Identity Service private certificate file. If you include the private key in the certfile, you can omit the keyfile.
- `ca_certs`. Path to the CA trust chain.
- `cert_required`. Requires client certificate. Default is False.

External authentication with the Identity Service

When the Identity Service runs in `apache-httpd`, you can use external authentication methods that differ from the authentication provided by the identity store back-end. For example, you can use an SQL identity back-end together with X.509 authentication, Kerberos, and so on instead of using the user name and password combination.

Use HTTPD authentication

Web servers, like Apache HTTP, support many methods of authentication. The Identity Service can allow the web server to perform the authentication. The web server then passes the authenticated user to the Identity Service by using the REMOTE_USER environment variable. This user must already exist in the Identity Service back-end so as to get a token from the controller. To use this method, the Identity Service should run on `apache-httpd`.

Use X.509

The following Apache configuration snippet authenticates the user based on a valid X.509 certificate from a known CA:

```
    <VirtualHost _default_:5000>
        SSLEngine on
        SSLCertificateFile    /etc/ssl/certs/ssl.cert
        SSLCertificateKeyFile /etc/ssl/private/ssl.key

        SSLCACertificatePath /etc/ssl/allowed_cas
        SSLCARevocationPath  /etc/ssl/allowed_cas
        SSLUserName          SSL_CLIENT_S_DN_CN
        SSLVerifyClient      require
        SSLVerifyDepth       10

        (...)
    </VirtualHost>
```

Configure the Identity Service with an LDAP back-end

As an alternative to the SQL database backing store, the Identity Service can use a directory server to provide the Identity Service, for example:

```
dn: dc=AcmeExample,dc=org
dc: AcmeExample
objectClass: dcObject
objectClass: organizationalUnit
ou: AcmeExample

dn: ou=Groups,dc=AcmeExample,dc=org
objectClass: top
objectClass: organizationalUnit
ou: groups

dn: ou=Users,dc=AcmeExample,dc=org
objectClass: top
objectClass: organizationalUnit
ou: users

dn: ou=Roles,dc=AcmeExample,dc=org
objectClass: top
objectClass: organizationalUnit
ou: roles
```

The corresponding entries in the `keystone.conf` configuration file are:

```
[ldap]
url = ldap://localhost
user = dc=Manager,dc=AcmeExample,dc=org
password = badpassword
suffix = dc=AcmeExample,dc=org
use_dumb_member = False
allow_subtree_delete = False

user_tree_dn = ou=Users,dc=AcmeExample,dc=com
user_objectclass = inetOrgPerson

tenant_tree_dn = ou=Groups,dc=AcmeExample,dc=com
tenant_objectclass = groupOfNames

role_tree_dn = ou=Roles,dc=AcmeExample,dc=com
role_objectclass = organizationalRole
```

The default object classes and attributes are intentionally simple. They reflect the common standard objects according to the LDAP RFCs. However, in a live deployment, you can override the correct attributes to support a preexisting, complex schema. For example, in the user object, the objectClass posixAccount from RFC2307 is very common. If this is the underlying objectclass, then the *uid* field should probably be *uidNumber* and *username* field either *uid* or *cn*. To change these two fields, the corresponding entries in the Keystone configuration file are:

```
[ldap]
user_id_attribute = uidNumber
user_name_attribute = cn
```

Depending on your deployment, you can modify a set of allowed actions for each object type. For example, you might set the following options:

```
[ldap]
user_allow_create = False
user_allow_update = False
user_allow_delete = False

tenant_allow_create = True
tenant_allow_update = True
tenant_allow_delete = True

role_allow_create = True
role_allow_update = True
role_allow_delete = True
```

If the back-end provides too much output, you can filter users, tenants, and roles. For example:

```
[ldap]
user_filter = (memberof=CN=acme-users,OU=workgroups,DC=AcmeExample,DC=com)
tenant_filter =
role_filter =
```

If the directory server has not enabled the `boolean` type for the user, you can use configuration options to extract the value from an integer attribute. For example, in an Active Directory, as follows:

```
[ldap]
user_enabled_attribute = userAccountControl
user_enabled_mask      = 2
user_enabled_default   = 512
```

The attribute is an integer. Bit 1 contains the enabled attribute. If the *user_enabled_mask* mask is not 0, it gets its value from the `user_enabled_attribute` field and it performs an ADD operation by using the *user_enabled_mask* value. If the value matches the mask, the account is disabled.

It also saves the value without mask to the `identity` user in the `enabled_nomask` attribute. In case you must change it to enable or disable a user, you can use this value because it contains more information than the status such as, password expiration. The *user_enabled_mask* value is required to create a default value on the integer attribute (512 = NORMAL ACCOUNT on AD).

If Active Directory classes and attributes do not match the specified classes in the LDAP module, so you can modify them, as follows:

```
[ldap]
user_objectclass         = person
user_id_attribute        = cn
user_name_attribute      = cn
user_mail_attribute      = mail
user_enabled_attribute   = userAccountControl
user_enabled_mask        = 2
user_enabled_default     = 512
user_attribute_ignore    = tenant_id,tenants
tenant_objectclass       = groupOfNames
tenant_id_attribute      = cn
tenant_member_attribute  = member
tenant_name_attribute    = ou
tenant_desc_attribute    = description
tenant_enabled_attribute = extensionName
tenant_attribute_ignore  =
role_objectclass         = organizationalRole
role_id_attribute        = cn
role_name_attribute      = ou
role_member_attribute    = roleOccupant
role_attribute_ignore    =
```

Configure the Identity Service for token binding

Token binding refers to the practice of embedding information from external authentication providers (like a company's Kerberos server) inside the token such that a client may enforce that the token only be used in conjunction with that specified authentication. This is an additional security mechanism as it means that if a token is stolen it will not be usable without also providing the external authentication.

To activate token binding you must specify the types of authentication that token binding should be used for in `keystone.conf`:

```
[token]
    bind = kerberos
```

Currently only `kerberos` is supported.

To enforce checking of token binding the `enforce_token_bind` parameter should be set to one of the following modes:

- `disabled` disable token bind checking
- `permissive` enable bind checking, if a token is bound to a mechanism that is unknown to the server then ignore it. This is the default.
- `strict` enable bind checking, if a token is bound to a mechanism that is unknown to the server then this token should be rejected.
- `required` enable bind checking and require that at least 1 bind mechanism is used for tokens.
- `named` enable bind checking and require that the specified authentication mechanism is used:

```
[token]
    enforce_token_bind = kerberos
```

Note

Do not set `enforce_token_bind = named` as there is not an authentication mechanism called `named`.

Identity Service sample configuration files

- `etc/keystone.conf.sample`

```
[DEFAULT]

#
# Options defined in keystone
#

# A "shared secret" that can be used to bootstrap Keystone.
# This "token" does not represent a user, and carries no
# explicit authorization. To disable in production (highly
# recommended), remove AdminTokenAuthMiddleware from your
# paste application pipelines (for example, in keystone-
# paste.ini). (string value)
#admin_token=ADMIN

# The IP address of the network interface for the public
# service to listen on. (string value)
# Deprecated group/name - [DEFAULT]/bind_host
#public_bind_host=0.0.0.0

# The IP address of the network interface for the admin
# service to listen on. (string value)
# Deprecated group/name - [DEFAULT]/bind_host
#admin_bind_host=0.0.0.0

# The port which the OpenStack Compute service listens on.
# (integer value)
#compute_port=8774

# The port number which the admin service listens on. (integer
```

```
# value)
#admin_port=35357

# The port number which the public service listens on.
# (integer value)
#public_port=5000

# The base public endpoint URL for Keystone that is advertised
# to clients (NOTE: this does NOT affect how Keystone listens
# for connections). Defaults to the base host URL of the
# request. E.g. a request to http://server:5000/v2.0/users
# will default to http://server:5000. You should only need to
# set this value if the base URL contains a path (e.g.
# /prefix/v2.0) or the endpoint should be found on a different
# server. (string value)
#public_endpoint=<None>

# The base admin endpoint URL for Keystone that is advertised
# to clients (NOTE: this does NOT affect how Keystone listens
# for connections). Defaults to the base host URL of the
# request. E.g. a request to http://server:35357/v2.0/users
# will default to http://server:35357. You should only need to
# set this value if the base URL contains a path (e.g.
# /prefix/v2.0) or the endpoint should be found on a different
# server. (string value)
#admin_endpoint=<None>

# onready allows you to send a notification when the process
# is ready to serve. For example, to have it notify using
# systemd, one could set shell command: "onready = systemd-
# notify --ready" or a module with notify() method: "onready =
# keystone.common.systemd". (string value)
#onready=<None>

# Enforced by optional sizelimit middleware
# (keystone.middleware:RequestBodySizeLimiter). (integer
# value)
#max_request_body_size=114688

# Limit the sizes of user & project ID/names. (integer value)
#max_param_size=64

# Similar to max_param_size, but provides an exception for
# token values. (integer value)
#max_token_size=8192

# During a SQL upgrade member_role_id will be used to create a
# new role that will replace records in the
# user_tenant_membership table with explicit role grants.
# After migration, the member_role_id will be used in the API
# add_user_to_project. (string value)
#member_role_id=9fe2ff9ee4384b1894a90878d3e92bab

# During a SQL upgrade member_role_name will be used to create
# a new role that will replace records in the
# user_tenant_membership table with explicit role grants.
# After migration, member_role_name will be ignored. (string
# value)
#member_role_name=_member_
```

```
# The value passed as the keyword "rounds" to passlib's
# encrypt method. (integer value)
#crypt_strength=40000

# Set this to true if you want to enable TCP_KEEPALIVE on
# server sockets, i.e. sockets used by the Keystone wsgi
# server for client connections. (boolean value)
#tcp_keepalive=false

# Sets the value of TCP_KEEPIDLE in seconds for each server
# socket. Only applies if tcp_keepalive is true. Not supported
# on OS X. (integer value)
#tcp_keepidle=600

# The maximum number of entities that will be returned in a
# collection, with no limit set by default. This global limit
# may be then overridden for a specific driver, by specifying
# a list_limit in the appropriate section (e.g. [assignment]).
# (integer value)
#list_limit=<None>

# Set this to false if you want to enable the ability for
# user, group and project entities to be moved between domains
# by updating their domain_id. Allowing such movement is not
# recommended if the scope of a domain admin is being
# restricted by use of an appropriate policy file (see
# policy.v3cloudsample as an example). (boolean value)
#domain_id_immutable=true


#
# Options defined in oslo.messaging
#

# Use durable queues in amqp. (boolean value)
# Deprecated group/name - [DEFAULT]/rabbit_durable_queues
#amqp_durable_queues=false

# Auto-delete queues in amqp. (boolean value)
#amqp_auto_delete=false

# Size of RPC connection pool. (integer value)
#rpc_conn_pool_size=30

# Modules of exceptions that are permitted to be recreated
# upon receiving exception data from an rpc call. (list value)
#allowed_rpc_exception_modules=oslo.messaging.exceptions,nova.exception,
cinder.exception,exceptions

# Qpid broker hostname. (string value)
#qpid_hostname=localhost

# Qpid broker port. (integer value)
#qpid_port=5672

# Qpid HA cluster host:port pairs. (list value)
#qpid_hosts=$qpid_hostname:$qpid_port

# Username for Qpid connection. (string value)
#qpid_username=
```

```
# Password for Qpid connection. (string value)
#qpid_password=

# Space separated list of SASL mechanisms to use for auth.
# (string value)
#qpid_sasl_mechanisms=

# Seconds between connection keepalive heartbeats. (integer
# value)
#qpid_heartbeat=60

# Transport to use, either 'tcp' or 'ssl'. (string value)
#qpid_protocol=tcp

# Whether to disable the Nagle algorithm. (boolean value)
#qpid_tcp_nodelay=true

# The qpid topology version to use.  Version 1 is what was
# originally used by impl_qpid.  Version 2 includes some
# backwards-incompatible changes that allow broker federation
# to work.  Users should update to version 2 when they are
# able to take everything down, as it requires a clean break.
# (integer value)
#qpid_topology_version=1

# SSL version to use (valid only if SSL enabled). valid values
# are TLSv1, SSLv23 and SSLv3. SSLv2 may be available on some
# distributions. (string value)
#kombu_ssl_version=

# SSL key file (valid only if SSL enabled). (string value)
#kombu_ssl_keyfile=

# SSL cert file (valid only if SSL enabled). (string value)
#kombu_ssl_certfile=

# SSL certification authority file (valid only if SSL
# enabled). (string value)
#kombu_ssl_ca_certs=

# How long to wait before reconnecting in response to an AMQP
# consumer cancel notification. (floating point value)
#kombu_reconnect_delay=1.0

# The RabbitMQ broker address where a single node is used.
# (string value)
#rabbit_host=localhost

# The RabbitMQ broker port where a single node is used.
# (integer value)
#rabbit_port=5672

# RabbitMQ HA cluster host:port pairs. (list value)
#rabbit_hosts=$rabbit_host:$rabbit_port

# Connect over SSL for RabbitMQ. (boolean value)
#rabbit_use_ssl=false

# The RabbitMQ userid. (string value)
```

```
#rabbit_userid=guest

# The RabbitMQ password. (string value)
#rabbit_password=guest

# the RabbitMQ login method (string value)
#rabbit_login_method=AMQPLAIN

# The RabbitMQ virtual host. (string value)
#rabbit_virtual_host=/

# How frequently to retry connecting with RabbitMQ. (integer
# value)
#rabbit_retry_interval=1

# How long to backoff for between retries when connecting to
# RabbitMQ. (integer value)
#rabbit_retry_backoff=2

# Maximum number of RabbitMQ connection retries. Default is 0
# (infinite retry count). (integer value)
#rabbit_max_retries=0

# Use HA queues in RabbitMQ (x-ha-policy: all). If you change
# this option, you must wipe the RabbitMQ database. (boolean
# value)
#rabbit_ha_queues=false

# If passed, use a fake RabbitMQ provider. (boolean value)
#fake_rabbit=false

# ZeroMQ bind address. Should be a wildcard (*), an ethernet
# interface, or IP. The "host" option should point or resolve
# to this address. (string value)
#rpc_zmq_bind_address=*

# MatchMaker driver. (string value)
#rpc_zmq_matchmaker=oslo.messaging._drivers.matchmaker.MatchMakerLocalhost

# ZeroMQ receiver listening port. (integer value)
#rpc_zmq_port=9501

# Number of ZeroMQ contexts, defaults to 1. (integer value)
#rpc_zmq_contexts=1

# Maximum number of ingress messages to locally buffer per
# topic. Default is unlimited. (integer value)
#rpc_zmq_topic_backlog=<None>

# Directory for holding IPC sockets. (string value)
#rpc_zmq_ipc_dir=/var/run/openstack

# Name of this node. Must be a valid hostname, FQDN, or IP
# address. Must match "host" option, if running Nova. (string
# value)
#rpc_zmq_host=keystone

# Seconds to wait before a cast expires (TTL). Only supported
# by impl_zmq. (integer value)
#rpc_cast_timeout=30
```

```
# Heartbeat frequency. (integer value)
#matchmaker_heartbeat_freq=300

# Heartbeat time-to-live. (integer value)
#matchmaker_heartbeat_ttl=600

# Host to locate redis. (string value)
#host=127.0.0.1

# Use this port to connect to redis host. (integer value)
#port=6379

# Password for Redis server (optional). (string value)
#password=<None>

# Size of RPC greenthread pool. (integer value)
#rpc_thread_pool_size=64

# Driver or drivers to handle sending notifications. (multi
# valued)
#notification_driver=

# AMQP topic used for OpenStack notifications. (list value)
# Deprecated group/name - [rpc_notifier2]/topics
#notification_topics=notifications

# Seconds to wait for a response from a call. (integer value)
#rpc_response_timeout=60

# A URL representing the messaging driver to use and its full
# configuration. If not set, we fall back to the rpc_backend
# option and driver specific configuration. (string value)
#transport_url=<None>

# The messaging driver to use, defaults to rabbit. Other
# drivers include qpid and zmq. (string value)
#rpc_backend=rabbit

# The default exchange under which topics are scoped. May be
# overridden by an exchange name specified in the
# transport_url option. (string value)
#control_exchange=openstack


#
# Options defined in keystone.notifications
#

# Default publisher_id for outgoing notifications (string
# value)
#default_publisher_id=<None>


#
# Options defined in keystone.middleware.ec2_token
#

# URL to get token from ec2 request. (string value)
#keystone_ec2_url=http://localhost:5000/v2.0/ec2tokens
```

```
# Required if EC2 server requires client certificate. (string
# value)
#keystone_ec2_keyfile=<None>

# Client certificate key filename. Required if EC2 server
# requires client certificate. (string value)
#keystone_ec2_certfile=<None>

# A PEM encoded certificate authority to use when verifying
# HTTPS connections. Defaults to the system CAs. (string
# value)
#keystone_ec2_cafile=<None>

# Disable SSL certificate verification. (boolean value)
#keystone_ec2_insecure=false


#
# Options defined in keystone.openstack.common.eventlet_backdoor
#

# 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
# service's log file. (string value)
#backdoor_port=<None>


#
# Options defined in keystone.openstack.common.lockutils
#

# Whether to disable inter-process locks (boolean value)
#disable_process_locking=false

# Directory to use for lock files. (string value)
#lock_path=<None>


#
# Options defined in keystone.openstack.common.log
#

# Print debugging output (set logging level to DEBUG instead
# of default WARNING level). (boolean value)
#debug=false

# Print more verbose output (set logging level to INFO instead
# of default WARNING level). (boolean value)
#verbose=false

# Log output to standard error (boolean value)
#use_stderr=true

# Format string to use for log messages with context (string
```

```
# value)
#logging_context_format_string=%(asctime)s.%(msecs)03d %(process)d
 %(levelname)s %(name)s [%(request_id)s %(user_identity)s] %(instance)s
%(message)s

# Format string to use for log messages without context
# (string value)
#logging_default_format_string=%(asctime)s.%(msecs)03d %(process)d
 %(levelname)s %(name)s [-] %(instance)s%(message)s

# Data to append to log format when level is DEBUG (string
# value)
#logging_debug_format_suffix=%(funcName)s %(pathname)s:%(lineno)d

# Prefix each line of exception output with this format
# (string value)
#logging_exception_prefix=%(asctime)s.%(msecs)03d %(process)d TRACE %(name)s
 %(instance)s

# List of logger=LEVEL pairs (list value)
#default_log_levels=amqp=WARN,amqplib=WARN,boto=WARN,qpid=WARN,sqlalchemy=
WARN,suds=INFO,iso8601=WARN,requests.packages.urllib3.connectionpool=WARN

# Publish error events (boolean value)
#publish_errors=false

# Make deprecations fatal (boolean value)
#fatal_deprecations=false

# If an instance is passed with the log message, format it
# like this (string value)
#instance_format="[instance: %(uuid)s] "

# If an instance UUID is passed with the log message, format
# it like this (string value)
#instance_uuid_format="[instance: %(uuid)s] "

# The name of logging configuration file. It does not disable
# existing loggers, but just appends specified logging
# configuration to any other existing logging options. Please
# see the Python logging module documentation for details on
# logging configuration files. (string value)
# Deprecated group/name - [DEFAULT]/log_config
#log_config_append=<None>

# DEPRECATED. A logging.Formatter log message format string
# which may use any of the available logging.LogRecord
# attributes. This option is deprecated.  Please use
# logging_context_format_string and
# logging_default_format_string instead. (string value)
#log_format=<None>

# Format string for %%(asctime)s in log records. Default:
# %(default)s (string value)
#log_date_format=%Y-%m-%d %H:%M:%S

# (Optional) Name of log file to output to. If no default is
# set, logging will go to stdout. (string value)
# Deprecated group/name - [DEFAULT]/logfile
#log_file=<None>
```

```
# (Optional) The base directory used for relative --log-file
# paths (string value)
# Deprecated group/name - [DEFAULT]/logdir
#log_dir=<None>

# Use syslog for logging. Existing syslog format is DEPRECATED
# during I, and then will be changed in J to honor RFC5424
# (boolean value)
#use_syslog=false

# (Optional) Use syslog rfc5424 format for logging. If
# enabled, will add APP-NAME (RFC5424) before the MSG part of
# the syslog message.  The old format without APP-NAME is
# deprecated in I, and will be removed in J. (boolean value)
#use_syslog_rfc_format=false

# Syslog facility to receive log lines (string value)
#syslog_log_facility=LOG_USER


#
# Options defined in keystone.openstack.common.policy
#

# JSON file containing policy (string value)
#policy_file=policy.json

# Rule enforced when requested rule is not found (string
# value)
#policy_default_rule=default


[assignment]

#
# Options defined in keystone
#

# Assignment backend driver. (string value)
#driver=<None>

# Toggle for assignment caching. This has no effect unless
# global caching is enabled. (boolean value)
#caching=true

# TTL (in seconds) to cache assignment data. This has no
# effect unless global caching is enabled. (integer value)
#cache_time=<None>

# Maximum number of entities that will be returned in an
# assignment collection. (integer value)
#list_limit=<None>


[auth]

#
# Options defined in keystone
#
```

```
# Default auth methods. (list value)
#methods=external,password,token

# The password auth plugin module. (string value)
#password=keystone.auth.plugins.password.Password

# The token auth plugin module. (string value)
#token=keystone.auth.plugins.token.Token

# The external (REMOTE_USER) auth plugin module. (string
# value)
#external=keystone.auth.plugins.external.DefaultDomain


[cache]

#
# Options defined in keystone
#

# Prefix for building the configuration dictionary for the
# cache region. This should not need to be changed unless
# there is another dogpile.cache region with the same
# configuration name. (string value)
#config_prefix=cache.keystone

# Default TTL, in seconds, for any cached item in the
# dogpile.cache region. This applies to any cached method that
# doesn't have an explicit cache expiration time defined for
# it. (integer value)
#expiration_time=600

# Dogpile.cache backend module. It is recommended that
# Memcache (dogpile.cache.memcache) or Redis
# (dogpile.cache.redis) be used in production deployments.
# Small workloads (single process) like devstack can use the
# dogpile.cache.memory backend. (string value)
#backend=keystone.common.cache.noop

# Use a key-mangling function (sha1) to ensure fixed length
# cache-keys. This is toggle-able for debugging purposes, it
# is highly recommended to always leave this set to true.
# (boolean value)
#use_key_mangler=true

# Arguments supplied to the backend module. Specify this
# option once per argument to be passed to the dogpile.cache
# backend. Example format: "<argname>:<value>". (multi valued)
#backend_argument=

# Proxy classes to import that will affect the way the
# dogpile.cache backend functions. See the dogpile.cache
# documentation on changing-backend-behavior. (list value)
#proxies=

# Global toggle for all caching using the should_cache_fn
# mechanism. (boolean value)
#enabled=false
```

```
# Extra debugging from the cache backend (cache keys,
# get/set/delete/etc calls). This is only really useful if you
# need to see the specific cache-backend get/set/delete calls
# with the keys/values.  Typically this should be left set to
# false. (boolean value)
#debug_cache_backend=false


[catalog]

#
# Options defined in keystone
#

# Catalog template file name for use with the template catalog
# backend. (string value)
#template_file=default_catalog.templates

# Catalog backend driver. (string value)
#driver=keystone.catalog.backends.sql.Catalog

# Maximum number of entities that will be returned in a
# catalog collection. (integer value)
#list_limit=<None>


[credential]

#
# Options defined in keystone
#

# Credential backend driver. (string value)
#driver=keystone.credential.backends.sql.Credential


[database]

#
# Options defined in keystone.openstack.common.db.options
#

# The file name to use with SQLite (string value)
#sqlite_db=keystone.sqlite

# If True, SQLite uses synchronous mode (boolean value)
#sqlite_synchronous=true

# The backend to use for db (string value)
# Deprecated group/name - [DEFAULT]/db_backend
#backend=sqlalchemy

# The SQLAlchemy connection string used to connect to the
# database (string value)
# Deprecated group/name - [DEFAULT]/sql_connection
# Deprecated group/name - [DATABASE]/sql_connection
# Deprecated group/name - [sql]/connection
#connection=<None>

# 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= (string
# value)
#mysql_sql_mode=TRADITIONAL

# Timeout before idle sql connections are reaped (integer
# value)
# Deprecated group/name - [DEFAULT]/sql_idle_timeout
# Deprecated group/name - [DATABASE]/sql_idle_timeout
# Deprecated group/name - [sql]/idle_timeout
#idle_timeout=3600

# Minimum number of SQL connections to keep open in a pool
# (integer value)
# Deprecated group/name - [DEFAULT]/sql_min_pool_size
# Deprecated group/name - [DATABASE]/sql_min_pool_size
#min_pool_size=1

# Maximum number of SQL connections to keep open in a pool
# (integer value)
# Deprecated group/name - [DEFAULT]/sql_max_pool_size
# Deprecated group/name - [DATABASE]/sql_max_pool_size
#max_pool_size=<None>

# Maximum db connection retries during startup. (setting -1
# implies an infinite retry count) (integer value)
# Deprecated group/name - [DEFAULT]/sql_max_retries
# Deprecated group/name - [DATABASE]/sql_max_retries
#max_retries=10

# Interval between retries of opening a sql connection
# (integer value)
# Deprecated group/name - [DEFAULT]/sql_retry_interval
# Deprecated group/name - [DATABASE]/reconnect_interval
#retry_interval=10

# If set, use this value for max_overflow with sqlalchemy
# (integer value)
# Deprecated group/name - [DEFAULT]/sql_max_overflow
# Deprecated group/name - [DATABASE]/sqlalchemy_max_overflow
#max_overflow=<None>

# Verbosity of SQL debugging information. 0=None,
# 100=Everything (integer value)
# Deprecated group/name - [DEFAULT]/sql_connection_debug
#connection_debug=0

# Add python stack traces to SQL as comment strings (boolean
# value)
# Deprecated group/name - [DEFAULT]/sql_connection_trace
#connection_trace=false

# If set, use this value for pool_timeout with sqlalchemy
# (integer value)
# Deprecated group/name - [DATABASE]/sqlalchemy_pool_timeout
#pool_timeout=<None>

# Enable the experimental use of database reconnect on
# connection lost (boolean value)
```

```
#use_db_reconnect=false

# seconds between db connection retries (integer value)
#db_retry_interval=1

# Whether to increase interval between db connection retries,
# up to db_max_retry_interval (boolean value)
#db_inc_retry_interval=true

# max seconds between db connection retries, if
# db_inc_retry_interval is enabled (integer value)
#db_max_retry_interval=10

# maximum db connection retries before error is raised.
# (setting -1 implies an infinite retry count) (integer value)
#db_max_retries=20


[ec2]

#
# Options defined in keystone
#

# EC2Credential backend driver. (string value)
#driver=keystone.contrib.ec2.backends.kvs.Ec2


[endpoint_filter]

#
# Options defined in keystone
#

# Endpoint Filter backend driver (string value)
#driver=keystone.contrib.endpoint_filter.backends.sql.EndpointFilter

# Toggle to return all active endpoints if no filter exists.
# (boolean value)
#return_all_endpoints_if_no_filter=true


[federation]

#
# Options defined in keystone
#

# Federation backend driver. (string value)
#driver=keystone.contrib.federation.backends.sql.Federation

# Value to be used when filtering assertion parameters from
# the environment. (string value)
#assertion_prefix=


[identity]

#
# Options defined in keystone
```

```
#

# This references the domain to use for all Identity API v2
# requests (which are not aware of domains). A domain with
# this ID will be created for you by keystone-manage db_sync
# in migration 008. The domain referenced by this ID cannot be
# deleted on the v3 API, to prevent accidentally breaking the
# v2 API. There is nothing special about this domain, other
# than the fact that it must exist to order to maintain
# support for your v2 clients. (string value)
#default_domain_id=default

# A subset (or all) of domains can have their own identity
# driver, each with their own partial configuration file in a
# domain configuration directory. Only values specific to the
# domain need to be placed in the domain specific
# configuration file. This feature is disabled by default; set
# to true to enable. (boolean value)
#domain_specific_drivers_enabled=false

# Path for Keystone to locate the domain specific identity
# configuration files if domain_specific_drivers_enabled is
# set to true. (string value)
#domain_config_dir=/etc/keystone/domains

# Identity backend driver. (string value)
#driver=keystone.identity.backends.sql.Identity

# Maximum supported length for user passwords; decrease to
# improve performance. (integer value)
#max_password_length=4096

# Maximum number of entities that will be returned in an
# identity collection. (integer value)
#list_limit=<None>


[kvs]

#
# Options defined in keystone
#

# Extra dogpile.cache backend modules to register with the
# dogpile.cache library. (list value)
#backends=

# Prefix for building the configuration dictionary for the KVS
# region. This should not need to be changed unless there is
# another dogpile.cache region with the same configuration
# name. (string value)
#config_prefix=keystone.kvs

# Toggle to disable using a key-mangling function to ensure
# fixed length keys. This is toggle-able for debugging
# purposes, it is highly recommended to always leave this set
# to true. (boolean value)
#enable_key_mangler=true

# Default lock timeout for distributed locking. (integer
```

```
# value)
#default_lock_timeout=5


[ldap]

#
# Options defined in keystone
#

# URL for connecting to the LDAP server. (string value)
#url=ldap://localhost

# User BindDN to query the LDAP server. (string value)
#user=<None>

# Password for the BindDN to query the LDAP server. (string
# value)
#password=<None>

# LDAP server suffix (string value)
#suffix=cn=example,cn=com

# If true, will add a dummy member to groups. This is required
# if the objectclass for groups requires the "member"
# attribute. (boolean value)
#use_dumb_member=false

# DN of the "dummy member" to use when "use_dumb_member" is
# enabled. (string value)
#dumb_member=cn=dumb,dc=nonexistent

# Delete subtrees using the subtree delete control. Only
# enable this option if your LDAP server supports subtree
# deletion. (boolean value)
#allow_subtree_delete=false

# The LDAP scope for queries, this can be either "one"
# (onelevel/singleLevel) or "sub" (subtree/wholeSubtree).
# (string value)
#query_scope=one

# Maximum results per page; a value of zero ("0") disables
# paging. (integer value)
#page_size=0

# The LDAP dereferencing option for queries. This can be
# either "never", "searching", "always", "finding" or
# "default". The "default" option falls back to using default
# dereferencing configured by your ldap.conf. (string value)
#alias_dereferencing=default

# Override the system's default referral chasing behavior for
# queries. (boolean value)
#chase_referrals=<None>

# Search base for users. (string value)
#user_tree_dn=<None>

# LDAP search filter for users. (string value)
```

```
#user_filter=<None>

# LDAP objectclass for users. (string value)
#user_objectclass=inetOrgPerson

# LDAP attribute mapped to user id. (string value)
#user_id_attribute=cn

# LDAP attribute mapped to user name. (string value)
#user_name_attribute=sn

# LDAP attribute mapped to user email. (string value)
#user_mail_attribute=email

# LDAP attribute mapped to password. (string value)
#user_pass_attribute=userPassword

# LDAP attribute mapped to user enabled flag. (string value)
#user_enabled_attribute=enabled

# Bitmask integer to indicate the bit that the enabled value
# is stored in if the LDAP server represents "enabled" as a
# bit on an integer rather than a boolean. A value of "0"
# indicates the mask is not used. If this is not set to "0"
# the typical value is "2". This is typically used when
# "user_enabled_attribute = userAccountControl". (integer
# value)
#user_enabled_mask=0

# Default value to enable users. This should match an
# appropriate int value if the LDAP server uses non-boolean
# (bitmask) values to indicate if a user is enabled or
# disabled. If this is not set to "True" the typical value is
# "512". This is typically used when "user_enabled_attribute =
# userAccountControl". (string value)
#user_enabled_default=True

# List of attributes stripped off the user on update. (list
# value)
#user_attribute_ignore=default_project_id,tenants

# LDAP attribute mapped to default_project_id for users.
# (string value)
#user_default_project_id_attribute=<None>

# Allow user creation in LDAP backend. (boolean value)
#user_allow_create=true

# Allow user updates in LDAP backend. (boolean value)
#user_allow_update=true

# Allow user deletion in LDAP backend. (boolean value)
#user_allow_delete=true

# If true, Keystone uses an alternative method to determine if
# a user is enabled or not by checking if they are a member of
# the "user_enabled_emulation_dn" group. (boolean value)
#user_enabled_emulation=false

# DN of the group entry to hold enabled users when using
```

```
# enabled emulation. (string value)
#user_enabled_emulation_dn=<None>

# List of additional LDAP attributes used for mapping
# additional attribute mappings for users. Attribute mapping
# format is <ldap_attr>:<user_attr>, where ldap_attr is the
# attribute in the LDAP entry and user_attr is the Identity
# API attribute. (list value)
#user_additional_attribute_mapping=

# Search base for projects (string value)
#tenant_tree_dn=<None>

# LDAP search filter for projects. (string value)
#tenant_filter=<None>

# LDAP objectclass for projects. (string value)
#tenant_objectclass=groupOfNames

# LDAP attribute mapped to project id. (string value)
#tenant_id_attribute=cn

# LDAP attribute mapped to project membership for user.
# (string value)
#tenant_member_attribute=member

# LDAP attribute mapped to project name. (string value)
#tenant_name_attribute=ou

# LDAP attribute mapped to project description. (string value)
#tenant_desc_attribute=description

# LDAP attribute mapped to project enabled. (string value)
#tenant_enabled_attribute=enabled

# LDAP attribute mapped to project domain_id. (string value)
#tenant_domain_id_attribute=businessCategory

# List of attributes stripped off the project on update. (list
# value)
#tenant_attribute_ignore=

# Allow project creation in LDAP backend. (boolean value)
#tenant_allow_create=true

# Allow project update in LDAP backend. (boolean value)
#tenant_allow_update=true

# Allow project deletion in LDAP backend. (boolean value)
#tenant_allow_delete=true

# If true, Keystone uses an alternative method to determine if
# a project is enabled or not by checking if they are a member
# of the "tenant_enabled_emulation_dn" group. (boolean value)
#tenant_enabled_emulation=false

# DN of the group entry to hold enabled projects when using
# enabled emulation. (string value)
#tenant_enabled_emulation_dn=<None>
```

```
# Additional attribute mappings for projects. Attribute
# mapping format is <ldap_attr>:<user_attr>, where ldap_attr
# is the attribute in the LDAP entry and user_attr is the
# Identity API attribute. (list value)
#tenant_additional_attribute_mapping=

# Search base for roles. (string value)
#role_tree_dn=<None>

# LDAP search filter for roles. (string value)
#role_filter=<None>

# LDAP objectclass for roles. (string value)
#role_objectclass=organizationalRole

# LDAP attribute mapped to role id. (string value)
#role_id_attribute=cn

# LDAP attribute mapped to role name. (string value)
#role_name_attribute=ou

# LDAP attribute mapped to role membership. (string value)
#role_member_attribute=roleOccupant

# List of attributes stripped off the role on update. (list
# value)
#role_attribute_ignore=

# Allow role creation in LDAP backend. (boolean value)
#role_allow_create=true

# Allow role update in LDAP backend. (boolean value)
#role_allow_update=true

# Allow role deletion in LDAP backend. (boolean value)
#role_allow_delete=true

# Additional attribute mappings for roles. Attribute mapping
# format is <ldap_attr>:<user_attr>, where ldap_attr is the
# attribute in the LDAP entry and user_attr is the Identity
# API attribute. (list value)
#role_additional_attribute_mapping=

# Search base for groups. (string value)
#group_tree_dn=<None>

# LDAP search filter for groups. (string value)
#group_filter=<None>

# LDAP objectclass for groups. (string value)
#group_objectclass=groupOfNames

# LDAP attribute mapped to group id. (string value)
#group_id_attribute=cn

# LDAP attribute mapped to group name. (string value)
#group_name_attribute=ou

# LDAP attribute mapped to show group membership. (string
# value)
```

```
#group_member_attribute=member

# LDAP attribute mapped to group description. (string value)
#group_desc_attribute=description

# List of attributes stripped off the group on update. (list
# value)
#group_attribute_ignore=

# Allow group creation in LDAP backend. (boolean value)
#group_allow_create=true

# Allow group update in LDAP backend. (boolean value)
#group_allow_update=true

# Allow group deletion in LDAP backend. (boolean value)
#group_allow_delete=true

# Additional attribute mappings for groups. Attribute mapping
# format is <ldap_attr>:<user_attr>, where ldap_attr is the
# attribute in the LDAP entry and user_attr is the Identity
# API attribute. (list value)
#group_additional_attribute_mapping=

# CA certificate file path for communicating with LDAP
# servers. (string value)
#tls_cacertfile=<None>

# CA certificate directory path for communicating with LDAP
# servers. (string value)
#tls_cacertdir=<None>

# Enable TLS for communicating with LDAP servers. (boolean
# value)
#use_tls=false

# Valid options for tls_req_cert are demand, never, and allow.
# (string value)
#tls_req_cert=demand


[matchmaker_ring]

#
# Options defined in oslo.messaging
#

# Matchmaker ring file (JSON). (string value)
# Deprecated group/name - [DEFAULT]/matchmaker_ringfile
#ringfile=/etc/oslo/matchmaker_ring.json


[memcache]

#
# Options defined in keystone
#

# Memcache servers in the format of "host:port". (list value)
#servers=localhost:11211
```

```
# Number of compare-and-set attempts to make when using
# compare-and-set in the token memcache back end. (integer
# value)
#max_compare_and_set_retry=16


[oauth1]

#
# Options defined in keystone
#

# Credential backend driver. (string value)
#driver=keystone.contrib.oauth1.backends.sql.OAuth1

# Duration (in seconds) for the OAuth Request Token. (integer
# value)
#request_token_duration=28800

# Duration (in seconds) for the OAuth Access Token. (integer
# value)
#access_token_duration=86400


[os_inherit]

#
# Options defined in keystone
#

# role-assignment inheritance to projects from owning domain
# can be optionally enabled. (boolean value)
#enabled=false


[paste_deploy]

#
# Options defined in keystone
#

# Name of the paste configuration file that defines the
# available pipelines. (string value)
#config_file=keystone-paste.ini


[policy]

#
# Options defined in keystone
#

# Policy backend driver. (string value)
#driver=keystone.policy.backends.sql.Policy

# Maximum number of entities that will be returned in a policy
# collection. (integer value)
#list_limit=<None>
```

```
[revoke]

#
# Options defined in keystone
#

# An implementation of the backend for persisting revocation
# events. (string value)
#driver=keystone.contrib.revoke.backends.kvs.Revoke

# This value (calculated in seconds) is added to token
# expiration before a revocation event may be removed from the
# backend. (integer value)
#expiration_buffer=1800

# Toggle for revocation event cacheing. This has no effect
# unless global caching is enabled. (boolean value)
#caching=true


[signing]

#
# Options defined in keystone
#

# Deprecated in favor of provider in the [token] section.
# (string value)
#token_format=<None>

# Path of the certfile for token signing. (string value)
#certfile=/etc/keystone/ssl/certs/signing_cert.pem

# Path of the keyfile for token signing. (string value)
#keyfile=/etc/keystone/ssl/private/signing_key.pem

# Path of the CA for token signing. (string value)
#ca_certs=/etc/keystone/ssl/certs/ca.pem

# Path of the CA key for token signing. (string value)
#ca_key=/etc/keystone/ssl/private/cakey.pem

# Key size (in bits) for token signing cert (auto generated
# certificate). (integer value)
#key_size=2048

# Days the token signing cert is valid for (auto generated
# certificate). (integer value)
#valid_days=3650

# Certificate subject (auto generated certificate) for token
# signing. (string value)
#cert_subject=/C=US/ST=Unset/L=Unset/O=Unset/CN=www.example.com


[ssl]

#
# Options defined in keystone
```

```
#

# Toggle for SSL support on the Keystone eventlet servers.
# (boolean value)
#enable=false

# Path of the certfile for SSL. (string value)
#certfile=/etc/keystone/ssl/certs/keystone.pem

# Path of the keyfile for SSL. (string value)
#keyfile=/etc/keystone/ssl/private/keystonekey.pem

# Path of the ca cert file for SSL. (string value)
#ca_certs=/etc/keystone/ssl/certs/ca.pem

# Path of the CA key file for SSL. (string value)
#ca_key=/etc/keystone/ssl/private/cakey.pem

# Require client certificate. (boolean value)
#cert_required=false

# SSL key length (in bits) (auto generated certificate).
# (integer value)
#key_size=1024

# Days the certificate is valid for once signed (auto
# generated certificate). (integer value)
#valid_days=3650

# SSL certificate subject (auto generated certificate).
# (string value)
#cert_subject=/C=US/ST=Unset/L=Unset/O=Unset/CN=localhost


[stats]

#
# Options defined in keystone
#

# Stats backend driver. (string value)
#driver=keystone.contrib.stats.backends.kvs.Stats


[token]

#
# Options defined in keystone
#

# External auth mechanisms that should add bind information to
# token, e.g., kerberos,x509. (list value)
#bind=

# Enforcement policy on tokens presented to Keystone with bind
# information. One of disabled, permissive, strict, required
# or a specifically required bind mode, e.g., kerberos or x509
# to require binding to that authentication. (string value)
#enforce_token_bind=permissive
```

```
# Amount of time a token should remain valid (in seconds).
# (integer value)
#expiration=3600

# Controls the token construction, validation, and revocation
# operations. Core providers are
# "keystone.token.providers.[pki|uuid].Provider". (string
# value)
#provider=<None>

# Token persistence backend driver. (string value)
#driver=keystone.token.backends.sql.Token

# Toggle for token system cacheing. This has no effect unless
# global caching is enabled. (boolean value)
#caching=true

# Time to cache the revocation list and the revocation events
# if revoke extension is enabled (in seconds). This has no
# effect unless global and token caching are enabled. (integer
# value)
#revocation_cache_time=3600

# Time to cache tokens (in seconds). This has no effect unless
# global and token caching are enabled. (integer value)
#cache_time=<None>

# Revoke token by token identifier. Setting revoke_by_id to
# true enables various forms of enumerating tokens, e.g. `list
# tokens for user`. These enumerations are processed to
# determine the list of tokens to revoke. Only disable if you
# are switching to using the Revoke extension with a backend
# other than KVS, which stores events in memory. (boolean
# value)
#revoke_by_id=true


[trust]

#
# Options defined in keystone
#

# Delegation and impersonation features can be optionally
# disabled. (boolean value)
#enabled=true

# Trust backend driver. (string value)
#driver=keystone.trust.backends.sql.Trust
```

- `etc/keystone-paste.ini`

```
# Keystone PasteDeploy configuration file.

[filter:debug]
paste.filter_factory = keystone.common.wsgi:Debug.factory

[filter:build_auth_context]
```

```
paste.filter_factory = keystone.middleware:AuthContextMiddleware.factory

[filter:token_auth]
paste.filter_factory = keystone.middleware:TokenAuthMiddleware.factory

[filter:admin_token_auth]
paste.filter_factory = keystone.middleware:AdminTokenAuthMiddleware.factory

[filter:xml_body]
paste.filter_factory = keystone.middleware:XmlBodyMiddleware.factory

[filter:xml_body_v2]
paste.filter_factory = keystone.middleware:XmlBodyMiddlewareV2.factory

[filter:xml_body_v3]
paste.filter_factory = keystone.middleware:XmlBodyMiddlewareV3.factory

[filter:json_body]
paste.filter_factory = keystone.middleware:JsonBodyMiddleware.factory

[filter:user_crud_extension]
paste.filter_factory = keystone.contrib.user_crud:CrudExtension.factory

[filter:crud_extension]
paste.filter_factory = keystone.contrib.admin_crud:CrudExtension.factory

[filter:ec2_extension]
paste.filter_factory = keystone.contrib.ec2:Ec2Extension.factory

[filter:ec2_extension_v3]
paste.filter_factory = keystone.contrib.ec2:Ec2ExtensionV3.factory

[filter:federation_extension]
paste.filter_factory = keystone.contrib.federation.
routers:FederationExtension.factory

[filter:oauth1_extension]
paste.filter_factory = keystone.contrib.oauth1.routers:OAuth1Extension.
factory

[filter:s3_extension]
paste.filter_factory = keystone.contrib.s3:S3Extension.factory

[filter:endpoint_filter_extension]
paste.filter_factory = keystone.contrib.endpoint_filter.
routers:EndpointFilterExtension.factory

[filter:simple_cert_extension]
paste.filter_factory = keystone.contrib.simple_cert:SimpleCertExtension.
factory

[filter:revoke_extension]
paste.filter_factory = keystone.contrib.revoke.routers:RevokeExtension.
factory

[filter:url_normalize]
paste.filter_factory = keystone.middleware:NormalizingFilter.factory

[filter:sizelimit]
paste.filter_factory = keystone.middleware:RequestBodySizeLimiter.factory
```

```
[filter:stats_monitoring]
paste.filter_factory = keystone.contrib.stats:StatsMiddleware.factory

[filter:stats_reporting]
paste.filter_factory = keystone.contrib.stats:StatsExtension.factory

[filter:access_log]
paste.filter_factory = keystone.contrib.access:AccessLogMiddleware.factory

[app:public_service]
paste.app_factory = keystone.service:public_app_factory

[app:service_v3]
paste.app_factory = keystone.service:v3_app_factory

[app:admin_service]
paste.app_factory = keystone.service:admin_app_factory

[pipeline:public_api]
pipeline = sizelimit url_normalize build_auth_context token_auth
 admin_token_auth xml_body_v2 json_body ec2_extension user_crud_extension
 public_service

[pipeline:admin_api]
pipeline = sizelimit url_normalize build_auth_context token_auth
 admin_token_auth xml_body_v2 json_body ec2_extension s3_extension
 crud_extension admin_service

[pipeline:api_v3]
pipeline = sizelimit url_normalize build_auth_context token_auth
 admin_token_auth xml_body_v3 json_body ec2_extension_v3 s3_extension
 simple_cert_extension service_v3

[app:public_version_service]
paste.app_factory = keystone.service:public_version_app_factory

[app:admin_version_service]
paste.app_factory = keystone.service:admin_version_app_factory

[pipeline:public_version_api]
pipeline = sizelimit url_normalize xml_body public_version_service

[pipeline:admin_version_api]
pipeline = sizelimit url_normalize xml_body admin_version_service

[composite:main]
use = egg:Paste#urlmap
/v2.0 = public_api
/v3 = api_v3
/ = public_version_api

[composite:admin]
use = egg:Paste#urlmap
/v2.0 = admin_api
/v3 = api_v3
/ = admin_version_api
```

- `etc/logging.conf.sample`

```
[loggers]
keys=root,access

[handlers]
keys=production,file,access_file,devel

[formatters]
keys=minimal,normal,debug


###########
# Loggers #
###########

[logger_root]
level=WARNING
handlers=file

[logger_access]
level=INFO
qualname=access
handlers=access_file


################
# Log Handlers #
################

[handler_production]
class=handlers.SysLogHandler
level=ERROR
formatter=normal
args=(('localhost', handlers.SYSLOG_UDP_PORT), handlers.SysLogHandler.
LOG_USER)

[handler_file]
class=handlers.WatchedFileHandler
level=WARNING
formatter=normal
args=('error.log',)

[handler_access_file]
class=handlers.WatchedFileHandler
level=INFO
formatter=minimal
args=('access.log',)

[handler_devel]
class=StreamHandler
level=NOTSET
formatter=debug
args=(sys.stdout,)


##################
# Log Formatters #
##################

[formatter_minimal]
```

```
format=%(message)s

[formatter_normal]
format=(%(name)s): %(asctime)s %(levelname)s %(message)s

[formatter_debug]
format=(%(name)s): %(asctime)s %(levelname)s %(module)s %(funcName)s
 %(message)s
```

2. Compute

Table of Contents

The OpenStack Compute service is a cloud computing fabric controller, the main part of an IaaS system. It can be used for hosting and manging cloud computing systems. This section describes the OpenStack Compute configuration options.

Post-Installation Configuration

Configuring your Compute installation involves many configuration files: the `nova.conf` file, the `api-paste.ini` file, and related Image and Identity management configuration files. This section contains the basics for a simple multi-node installation, but Compute can be configured many ways. You can find networking options and hypervisor options described in separate chapters.

Setting Configuration Options in the `nova.conf` File

The configuration file `nova.conf` is installed in `/etc/nova` by default. A default set of options are already configured in `nova.conf` when you install manually.

Create a `nova` group, so you can set permissions on the configuration file:

```
$ sudo addgroup nova
```

The `nova.conf` file should have its owner set to `root:nova`, and mode set to `0640`, since the file could contain your MySQL server's username and password. You also want to ensure that the `nova` user belongs to the `nova` group.

```
$ sudo usermod -g nova nova
$ chown -R username:nova /etc/nova
$ chmod 640 /etc/nova/nova.conf
```

General Compute configuration overview

Most configuration information is available in the `nova.conf` configuration option file, which is in the `/etc/nova` directory.

You can use a particular configuration option file by using the `option` (`nova.conf`) parameter when running one of the `nova-*` services. This inserts configuration option definitions from the given configuration file name, which may be useful for debugging or performance tuning.

If you want to maintain the state of all the services, you can use the `state_path` configuration option to indicate a top-level directory for storing data related to the state of Compute including images if you are using the Compute object store.

You can place comments in the `nova.conf` file by entering a new line with a # sign at the beginning of the line. To see a listing of all possible configuration options, refer to the tables in this guide. Here are some general purpose configuration options that you can use to learn more about the configuration option file and the node.

Table 2.1. Description of configuration options for common

Configuration option=Default value	Description
bindir=/usr/local/bin	(StrOpt) Directory where nova binaries are installed
compute_topic=compute	(StrOpt) the topic compute nodes listen on
console_topic=console	(StrOpt) the topic console proxy nodes listen on
consoleauth_topic=consoleauth	(StrOpt) the topic console auth proxy nodes listen on
disable_process_locking=False	(BoolOpt) Whether to disable inter-process locks
host=docwork	(StrOpt) 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, a valid hostname, FQDN, or IP address
host=127.0.0.1	(StrOpt) Host to locate redis
lock_path=None	(StrOpt) Directory to use for lock files.
memcached_servers=None	(ListOpt) Memcached servers or None for in process cache.
my_ip=192.168.122.99	(StrOpt) ip address of this host
notification_driver=[]	(MultiStrOpt) Driver or drivers to handle sending notifications
notification_topics=notifications	(ListOpt) AMQP topic used for OpenStack notifications
notify_api_faults=False	(BoolOpt) If set, send api.fault notifications on caught exceptions in the API service.
notify_on_state_change=None	(StrOpt) If set, send compute.instance.update notifications on instance state changes. Valid values are None for no notifications, "vm_state" for notifications on VM state changes, or "vm_and_task_state" for notifications on VM and task state changes.
pybasedir=/home/docwork/openstack-manuals-new/tools/autogenerate-config-docs/nova	(StrOpt) Directory where the nova python module is installed
report_interval=10	(IntOpt) seconds between nodes reporting state to datastore
rootwrap_config=/etc/nova/rootwrap.conf	(StrOpt) Path to the rootwrap configuration file to use for running commands as root
service_down_time=60	(IntOpt) maximum time since last check-in for up service
state_path=$pybasedir	(StrOpt) Top-level directory for maintaining nova's state
tempdir=None	(StrOpt) Explicitly specify the temporary working directory

Example `nova.conf` Configuration Files

The following sections describe many of the configuration option settings that can go into the `nova.conf` files. Copies of each `nova.conf` file need to be copied to each compute node. Here are some sample `nova.conf` files that offer examples of specific configurations.

Small, private cloud

Here is a simple example `nova.conf` file for a small private cloud, with all the cloud controller services, database server, and messaging server on the same server. In this case, CONTROLLER_IP represents the IP address of a central server, BRIDGE_INTERFACE represents the bridge such as br100, the NETWORK_INTERFACE represents an interface to your VLAN setup, and passwords are represented as DB_PASSWORD_COMPUTE for your Compute (nova) database password, and RABBIT PASSWORD represents the password to your message queue installation.

```
[DEFAULT]

# LOGS/STATE
verbose=True
logdir=/var/log/nova
state_path=/var/lib/nova
lock_path=/var/lock/nova
rootwrap_config=/etc/nova/rootwrap.conf

# SCHEDULER
compute_scheduler_driver=nova.scheduler.filter_scheduler.FilterScheduler

# VOLUMES
# configured in cinder.conf

# COMPUTE
libvirt_type=qemu
compute_driver=libvirt.LibvirtDriver
instance_name_template=instance-%08x
api_paste_config=/etc/nova/api-paste.ini

# COMPUTE/APIS: if you have separate configs for separate services
# this flag is required for both nova-api and nova-compute
allow_resize_to_same_host=True

# APIS
osapi_compute_extension=nova.api.openstack.compute.contrib.standard_extensions
ec2_dmz_host=192.168.206.130
s3_host=192.168.206.130

# RABBITMQ
rabbit_host=192.168.206.130

# GLANCE
image_service=nova.image.glance.GlanceImageService
glance_api_servers=192.168.206.130:9292

# NETWORK
network_manager=nova.network.manager.FlatDHCPManager
force_dhcp_release=True
dhcpbridge_flagfile=/etc/nova/nova.conf
firewall_driver=nova.virt.libvirt.firewall.IptablesFirewallDriver
# Change my_ip to match each host
my_ip=192.168.206.130
public_interface=eth0
vlan_interface=eth0
flat_network_bridge=br100
flat_interface=eth0
```

```
# NOVNC CONSOLE
novncproxy_base_url=http://192.168.206.130:6080/vnc_auto.html
# Change vncserver_proxyclient_address and vncserver_listen to match each
 compute host
vncserver_proxyclient_address=192.168.206.130
vncserver_listen=192.168.206.130

# AUTHENTICATION
auth_strategy=keystone
[keystone_authtoken]
auth_host = 127.0.0.1
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = nova
signing_dirname = /tmp/keystone-signing-nova

# DATABASE
[database]
connection=mysql://nova:yourpassword@192.168.206.130/nova
```

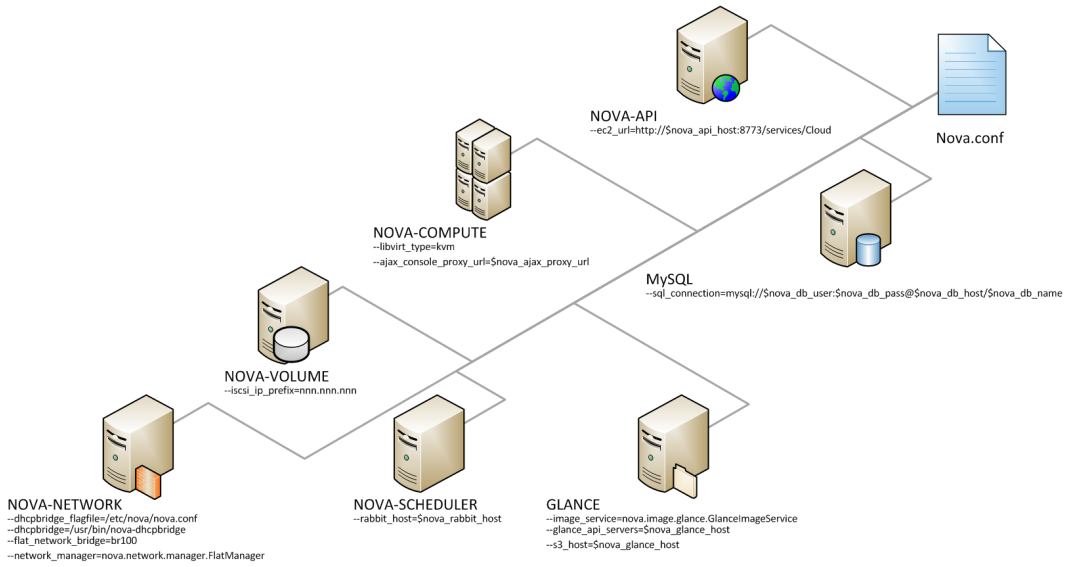
KVM, Flat, MySQL, and Glance, OpenStack or EC2 API

This example `nova.conf` file is from an internal Rackspace test system used for demonstrations.

```
[DEFAULT]

# LOGS/STATE
verbose=True
logdir=/var/log/nova
state_path=/var/lib/nova
lock_path=/var/lock/nova
rootwrap_config=/etc/nova/rootwrap.conf

# SCHEDULER
compute_scheduler_driver=nova.scheduler.filter_scheduler.FilterScheduler

# VOLUMES
# configured in cinder.conf

# COMPUTE
libvirt_type=qemu
compute_driver=libvirt.LibvirtDriver
instance_name_template=instance-%08x
api_paste_config=/etc/nova/api-paste.ini

# COMPUTE/APIS: if you have separate configs for separate services
# this flag is required for both nova-api and nova-compute
allow_resize_to_same_host=True

# APIS
osapi_compute_extension=nova.api.openstack.compute.contrib.standard_extensions
ec2_dmz_host=192.168.206.130
s3_host=192.168.206.130

# RABBITMQ
rabbit_host=192.168.206.130
```

```
# GLANCE
image_service=nova.image.glance.GlanceImageService
glance_api_servers=192.168.206.130:9292

# NETWORK
network_manager=nova.network.manager.FlatDHCPManager
force_dhcp_release=True
dhcpbridge_flagfile=/etc/nova/nova.conf
firewall_driver=nova.virt.libvirt.firewall.IptablesFirewallDriver
# Change my_ip to match each host
my_ip=192.168.206.130
public_interface=eth0
vlan_interface=eth0
flat_network_bridge=br100
flat_interface=eth0

# NOVNC CONSOLE
novncproxy_base_url=http://192.168.206.130:6080/vnc_auto.html
# Change vncserver_proxyclient_address and vncserver_listen to match each
 compute host
vncserver_proxyclient_address=192.168.206.130
vncserver_listen=192.168.206.130

# AUTHENTICATION
auth_strategy=keystone
[keystone_authtoken]
auth_host = 127.0.0.1
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = nova
signing_dirname = /tmp/keystone-signing-nova

# DATABASE
[database]
connection=mysql://nova:yourpassword@192.168.206.130/nova
```

Figure 2.1. KVM, Flat, MySQL, and Glance, OpenStack or EC2 API

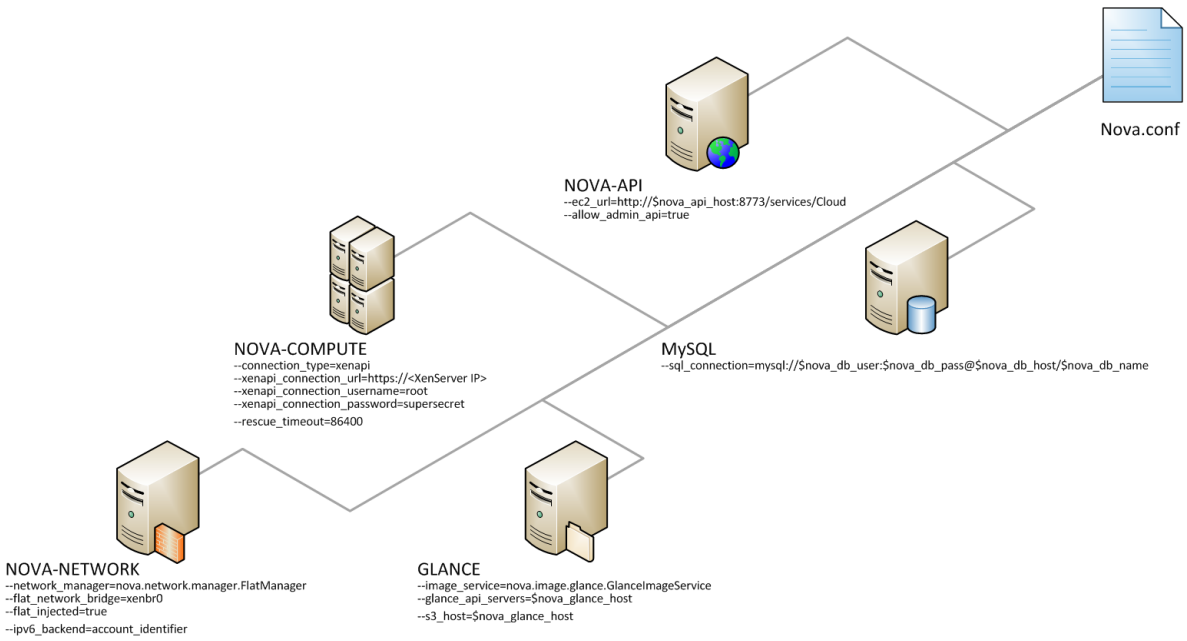


XenServer, Flat networking, MySQL, and Glance, OpenStack API

This example `nova.conf` file is from an internal Rackspace test system.

```
verbose
nodaemon
network_manager=nova.network.manager.FlatManager
image_service=nova.image.glance.GlanceImageService
flat_network_bridge=xenbr0
compute_driver=xenapi.XenAPIDriver
xenapi_connection_url=https://<XenServer IP>
xenapi_connection_username=root
xenapi_connection_password=supersecret
xenapi_image_upload_handler=nova.virt.xenapi.image.glance.GlanceStore
rescue_timeout=86400
use_ipv6=true

# To enable flat_injected, currently only works on Debian-based systems
flat_injected=true
ipv6_backend=account_identifier
ca_path=./nova/CA

# Add the following to your conf file if you're running on Ubuntu Maverick
xenapi_remap_vbd_dev=true
[database]
connection=mysql://root:<password>@127.0.0.1/nova
```

Figure 2.2. KVM, Flat, MySQL, and Glance, OpenStack or EC2 API



Configuring Logging

You can use `nova.conf` file to configure where Compute logs events, the level of logging, and log formats.

To customize log formats for OpenStack Compute, use these configuration option settings.

Table 2.2. Description of configuration options for logging

Configuration option=Default value	Description
debug=False	(BoolOpt) Print debugging output (set logging level to DEBUG instead of default WARNING level).
default_log_levels=amqplib=WARN,sqlalchemy=WARN,boto=WARN,suds=INFO,keystone=INFO,eventlet.wsgi.server=WARN	(ListOpt) list of logger=LEVEL pairs
fatal_deprecations=False	(BoolOpt) make deprecations fatal
fatal_exception_format_errors=False	(BoolOpt) make exception message format errors fatal
instance_format=[instance: %(uuid)s]	(StrOpt) If an instance is passed with the log message, format it like this
instance_uuid_format=[instance: %(uuid)s]	(StrOpt) If an instance UUID is passed with the log message, format it like this
log_config=None	(StrOpt) If this option is specified, the logging configuration file specified is used and overrides any

Configuration option=Default value	Description
	other logging options specified. Please see the Python logging module documentation for details on logging configuration files.
log_date_format=%Y-%m-%d %H:%M:%S	(StrOpt) Format string for %%(asctime)s in log records. Default: %(default)s
log_dir=None	(StrOpt) (Optional) The base directory used for relative –log-file paths
log_file=None	(StrOpt) (Optional) Name of log file to output to. If no default is set, logging will go to stdout.
log_format=None	(StrOpt) DEPRECATED. A logging.Formatter log message format string which may use any of the available logging.LogRecord attributes. This option is deprecated. Please use logging_context_format_string and logging_default_format_string instead.
logging_context_format_string=%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user)s %(tenant)s] %(instance)s%(message)s	(StrOpt) format string to use for log messages with context
logging_debug_format_suffix=%(funcName)s %(pathname)s:%(lineno)d	(StrOpt) data to append to log format when level is DEBUG
logging_default_format_string=%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s %(message)s	(StrOpt) format string to use for log messages without context
logging_exception_prefix=%(asctime)s.%(msecs)03d %(process)d TRACE %(name)s %(instance)s	(StrOpt) prefix each line of exception output with this format
publish_errors=False	(BoolOpt) publish error events
syslog_log_facility=LOG_USER	(StrOpt) syslog facility to receive log lines
use_stderr=True	(BoolOpt) Log output to standard error
use_syslog=False	(BoolOpt) Use syslog for logging.
verbose=False	(BoolOpt) Print more verbose output (set logging level to INFO instead of default WARNING level).

Configuring Hypervisors

See the section called "Hypervisors" [71] for details.

Configuring Authentication and Authorization

There are different methods of authentication for the OpenStack Compute project, including no authentication. The preferred system is the OpenStack Identity Service, code-named Keystone.

To customize authorization settings for Compute, see these configuration settings in `nova.conf`.

Table 2.3. Description of configuration options for authentication

Configuration option=Default value	Description
api_rate_limit=False	(BoolOpt) whether to use per-user rate limiting for the api.
auth_strategy=noauth	(StrOpt) The strategy to use for auth: noauth or keystone.

To customize certificate authority settings for Compute, see these configuration settings in `nova.conf`.

Table 2.4. Description of configuration options for ca

Configuration option=Default value	Description
ca_file=cacert.pem	(StrOpt) Filename of root CA
ca_file=None	(StrOpt) CA certificate file to use to verify connecting clients
ca_path=$state_path/CA	(StrOpt) Where we keep our root CA
cert_file=None	(StrOpt) Certificate file to use when starting the server securely
cert_manager=nova.cert.manager.CertManager	(StrOpt) full class name for the Manager for cert
cert_topic=cert	(StrOpt) the topic cert nodes listen on
crl_file=crl.pem	(StrOpt) Filename of root Certificate Revocation List
key_file=private/cakey.pem	(StrOpt) Filename of private key
key_file=None	(StrOpt) Private key file to use when starting the server securely
keys_path=$state_path/keys	(StrOpt) Where we keep our keys
project_cert_subject=/C=US/ST=California/O=OpenStack/OU=NovaDev/CN=project-ca-%.16s-%s	(StrOpt) Subject for certificate for projects, %s for project, timestamp
use_project_ca=False	(BoolOpt) Should we use a CA for each project?
user_cert_subject=/C=US/ST=California/O=OpenStack/OU=NovaDev/CN=%.16s-%.16s-%s	(StrOpt) Subject for certificate for users, %s for project, user, timestamp

To customize Compute and the Identity service to use LDAP as a backend, refer to these configuration settings in `nova.conf`.

Table 2.5. Description of configuration options for ldap

Configuration option=Default value	Description
ldap_dns_base_dn=ou=hosts,dc=example,dc=org	(StrOpt) Base DN for DNS entries in ldap
ldap_dns_password=password	(StrOpt) password for ldap DNS
ldap_dns_servers=['dns.example.org']	(MultiStrOpt) DNS Servers for ldap dns driver
ldap_dns_soa_expiry=86400	(StrOpt) Expiry interval (in seconds) for ldap dns driver Statement of Authority
ldap_dns_soa_hostmaster=hostmaster@example.org	(StrOpt) Hostmaster for ldap dns driver Statement of Authority
ldap_dns_soa_minimum=7200	(StrOpt) Minimum interval (in seconds) for ldap dns driver Statement of Authority
ldap_dns_soa_refresh=1800	(StrOpt) Refresh interval (in seconds) for ldap dns driver Statement of Authority
ldap_dns_soa_retry=3600	(StrOpt) Retry interval (in seconds) for ldap dns driver Statement of Authority
ldap_dns_url=ldap://ldap.example.com:389	(StrOpt) URL for ldap server which will store dns entries
ldap_dns_user=uid=admin,ou=people,dc=example,dc=org	(StrOpt) user for ldap DNS

Configure Compute to use IPv6 addresses

You can configure Compute to use both IPv4 and IPv6 addresses for communication by putting it into a IPv4/IPv6 dual stack mode. In IPv4/IPv6 dual stack mode, instances can acquire their IPv6 global unicast address by stateless address autoconfiguration mechanism [RFC 4862/2462]. IPv4/IPv6 dual stack mode works with `VlanManager` and `FlatDHCPManager` networking modes. In `VlanManager`, different 64bit global routing

prefix is used for each project. In `FlatDHCPManager`, one 64bit global routing prefix is used for all instances.

This configuration has been tested with VM images that have IPv6 stateless address autoconfiguration capability (must use EUI-64 address for stateless address autoconfiguration), a requirement for any VM you want to run with an IPv6 address. Each node that executes a `nova-*` service must have `python-netaddr` and `radvd` installed.

On all nova-nodes, install python-netaddr:

```
$ sudo apt-get install python-netaddr
```

On all `nova-network` nodes install `radvd` and configure IPv6 networking:

```
$ sudo apt-get install radvd
$ sudo bash -c "echo 1 > /proc/sys/net/ipv6/conf/all/forwarding"
$ sudo bash -c "echo 0 > /proc/sys/net/ipv6/conf/all/accept_ra"
```

Edit the `nova.conf` file on all nodes to set the use_ipv6 configuration option to True. Restart all nova- services.

When using the command **nova network-create** you can add a fixed range for IPv6 addresses. You must specify public or private after the create parameter.

```
$ nova network-create public --fixed-range-v4 fixed_range_v4 --vlan vlan_id --
vpn vpn_start --fixed-range-v6 fixed_range_v6
```

You can set IPv6 global routing prefix by using the `--fixed_range_v6` parameter. The default is: `fd00::/48`. When you use `FlatDHCPManager`, the command uses the original value of `--fixed_range_v6`. When you use `VlanManager`, the command creates prefixes of subnet by incrementing subnet id. Guest VMs uses this prefix for generating their IPv6 global unicast address.

Here is a usage example for `VlanManager`:

```
$ nova network-create public --fixed-range-v4 10.0.1.0/24 --vlan 100 --vpn
 1000 --fixed-range-v6 fd00:1::/48
```

Here is a usage example for `FlatDHCPManager`:

```
$ nova network-create public  --fixed-range-v4 10.0.2.0/24 --fixed-range-v6
 fd00:1::/48
```

Table 2.6. Description of configuration options for ipv6

Configuration option=Default value	Description
fixed_range_v6=fd00::/48	(StrOpt) Fixed IPv6 address block
gateway_v6=None	(StrOpt) Default IPv6 gateway
ipv6_backend=rfc2462	(StrOpt) Backend to use for IPv6 generation
use_ipv6=False	(BoolOpt) use ipv6

Configure migrations

Note

Only cloud administrators can perform live migrations. If your cloud is configured to use cells, you can perform live migration within but not between cells.

Migration enables an administrator to move a virtual machine instance from one compute host to another. This feature is useful when a compute host requires maintenance. Migration can also be useful to redistribute the load when many VM instances are running on a specific physical machine.

The migration types are:

- **Migration** (or non-live migration). The instance is shut down (and the instance knows that it was rebooted) for a period of time to be moved to another hypervisor.
- **Live migration** (or true live migration). Almost no instance downtime. Useful when the instances must be kept running during the migration.

The types of *live migration* are:

- **Shared storage-based live migration**. Both hypervisors have access to shared storage.
- **Block live migration**. No shared storage is required.
- **Volume-backed live migration**. When instances are backed by volumes rather than ephemeral disk, no shared storage is required, and migration is supported (currently only in libvirt-based hypervisors).

The following sections describe how to configure your hosts and compute nodes for migrations by using the KVM and XenServer hypervisors.

KVM-Libvirt

Prerequisites

- **Hypervisor:** KVM with libvirt
- **Shared storage:** `NOVA-INST-DIR/instances/` (for example, `/var/lib/nova/instances`) has to be mounted by shared storage. This guide uses NFS but other options, including the OpenStack Gluster Connector are available.
- **Instances:** Instance can be migrated with iSCSI based volumes

Notes

- Because the Compute service does not use the libvirt live migration functionality by default, guests are suspended before migration and might experience several minutes of downtime. For details, see the section called "Enable true live migration" [49].

- This guide assumes the default value for `instances_path` in your `nova.conf` file (*`NOVA-INST-DIR`*`/instances`). If you have changed the `state_path` or `instances_path` variables, modify accordingly.
- You must specify `vncserver_listen=0.0.0.0` or live migration does not work correctly.

Example Compute installation environment

- Prepare at least three servers; for example, `HostA`, `HostB`, and `HostC`.
- `HostA` is the *Cloud Controller*, and should run these services: `nova-api`, `nova-scheduler`, `nova-network`, `cinder-volume`, and `nova-objectstore`.
- `HostB` and `HostC` are the *compute nodes* that run `nova-compute`.
- Ensure that *`NOVA-INST-DIR`* (set with `state_path` in the `nova.conf` file) is the same on all hosts.
- In this example, `HostA` is the NFSv4 server that exports *`NOVA-INST-DIR`*`/instances`, and `HostB` and `HostC` mount it.

Procedure 2.1. To configure your system

1. Configure your DNS or `/etc/hosts` and ensure it is consistent across all hosts. Make sure that the three hosts can perform name resolution with each other. As a test, use the **ping** command to ping each host from one another.

   ```
   $ ping HostA
   $ ping HostB
   $ ping HostC
   ```

2. Ensure that the UID and GID of your nova and libvirt users are identical between each of your servers. This ensures that the permissions on the NFS mount works correctly.

3. Follow the instructions at the Ubuntu NFS HowTo to setup an NFS server on `HostA`, and NFS Clients on `HostB` and `HostC`.

   The aim is to export *`NOVA-INST-DIR`*`/instances` from `HostA`, and have it readable and writable by the nova user on `HostB` and `HostC`.

4. Using your knowledge from the Ubuntu documentation, configure the NFS server at `HostA` by adding this line to the `/etc/exports` file:

   ```
   NOVA-INST-DIR/instances HostA/255.255.0.0(rw,sync,fsid=0,no_root_squash)
   ```

   Change the subnet mask (`255.255.0.0`) to the appropriate value to include the IP addresses of `HostB` and `HostC`. Then restart the NFS server:

   ```
   $ /etc/init.d/nfs-kernel-server restart
   $ /etc/init.d/idmapd restart
   ```

5. Set the 'execute/search' bit on your shared directory.

On both compute nodes, make sure to enable the 'execute/search' bit to allow qemu to be able to use the images within the directories. On all hosts, run the following command:

```
$ chmod o+x NOVA-INST-DIR/instances
```

6. Configure NFS at HostB and HostC by adding this line to the `/etc/fstab` file:

```
HostA:/ /NOVA-INST-DIR/instances nfs4 defaults 0 0
```

Make sure that you can mount the exported directory can be mounted:

```
$ mount -a -v
```

Check that HostA can see the "*NOVA-INST-DIR*`/instances/`" directory:

```
$ ls -ld NOVA-INST-DIR/instances/
```

```
drwxr-xr-x 2 nova nova 4096 2012-05-19 14:34 nova-install-dir/instances/
```

Perform the same check at HostB and HostC, paying special attention to the permissions (nova should be able to write):

```
$ ls -ld NOVA-INST-DIR/instances/
```

```
drwxr-xr-x 2 nova nova 4096 2012-05-07 14:34 nova-install-dir/instances/
```

```
$ df -k
```

```
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/sda1            921514972   4180880 870523828   1% /
none                  16498340      1228  16497112   1% /dev
none                  16502856         0  16502856   0% /dev/shm
none                  16502856       368  16502488   1% /var/run
none                  16502856         0  16502856   0% /var/lock
none                  16502856         0  16502856   0% /lib/init/rw
HostA:        921515008 101921792 772783104  12% /var/lib/nova/instances
 ( <--- this line is important.)
```

7. Update the libvirt configurations so that the calls can be made securely. These methods enable remote access over TCP and are not documented here, please consult your network administrator for assistance in deciding how to configure access.

   - SSH tunnel to libvirtd's UNIX socket
   - libvirtd TCP socket, with GSSAPI/Kerberos for auth+data encryption
   - libvirtd TCP socket, with TLS for encryption and x509 client certs for authentication
   - libvirtd TCP socket, with TLS for encryption and Kerberos for authentication

   Restart libvirt. After you run the command, ensure that libvirt is successfully restarted:

```
$ stop libvirt-bin && start libvirt-bin
$ ps -ef | grep libvirt
```

```
root 1145 1 0 Nov27 ? 00:00:03 /usr/sbin/libvirtd -d -l
```

8. Configure your firewall to allow libvirt to communicate between nodes.

For information about ports that are used with libvirt, see the libvirt documentation By default, libvirt listens on TCP port 16509 and an ephemeral TCP range from 49152 to 49261 is used for the KVM communications. As this guide has disabled libvirt auth, you should take good care that these ports are only open to hosts within your installation.

9. You can now configure options for live migration. In most cases, you do not need to configure any options. The following chart is for advanced usage only.

Table 2.7. Description of configuration options for livemigration

Configuration option=Default value	Description
live_migration_bandwidth=0	(IntOpt) Maximum bandwidth to be used during migration, in Mbps
live_migration_flag=VIR_MIGRATE_UNDEFINE_SOURCE, VIR_MIGRATE_PEER2PEER	(StrOpt) Migration flags to be set for live migration
live_migration_retry_count=30	(IntOpt) Number of 1 second retries needed in live_migration
live_migration_uri=qemu+tcp://%s/system	(StrOpt) Migration target URI (any included "%s" is replaced with the migration target hostname)

Enable true live migration

By default, the Compute service does not use the libvirt live migration functionality. To enable this functionality, add the following line to the `nova.conf` file:

```
live_migration_flag=VIR_MIGRATE_UNDEFINE_SOURCE,VIR_MIGRATE_PEER2PEER,
VIR_MIGRATE_LIVE
```

The Compute service does not use libvirt's live migration by default because there is a risk that the migration process never ends. This can happen if the guest operating system dirties blocks on the disk faster than they can migrated.

XenServer

Shared storage

Prerequisites

- **Compatible XenServer hypervisors**. For more information, see the Requirements for Creating Resource Pools section of the *XenServer Administrator's Guide*.
- **Shared storage**. An NFS export, visible to all XenServer hosts.

Note

For the supported NFS versions, see the NFS VHD section of the *XenServer Administrator's Guide*.

To use shared storage live migration with XenServer hypervisors, the hosts must be joined to a XenServer pool. To create that pool, a host aggregate must be created with special metadata. This metadata is used by the XAPI plug-ins to establish the pool.

Procedure 2.2. To use shared storage live migration with XenServer hypervisors

1. Add an NFS VHD storage to your master XenServer, and set it as default SR. For more information, please refer to the NFS VHD section in the *XenServer Administrator's Guide*.

2. Configure all the compute nodes to use the default sr for pool operations. Add this line to your `nova.conf` configuration files across your compute nodes:

```
sr_matching_filter=default-sr:true
```

3. Create a host aggregate:

```
$ nova aggregate-create <name-for-pool> <availability-zone>
```

The command displays a table that contains the ID of the newly created aggregate.

Now add special metadata to the aggregate, to mark it as a hypervisor pool:

```
$ nova aggregate-set-metadata <aggregate-id> hypervisor_pool=true
```

```
$ nova aggregate-set-metadata <aggregate-id> operational_state=created
```

Make the first compute node part of that aggregate:

```
$ nova aggregate-add-host <aggregate-id> <name-of-master-compute>
```

At this point, the host is part of a XenServer pool.

4. Add additional hosts to the pool:

```
$ nova aggregate-add-host <aggregate-id> <compute-host-name>
```

Note

At this point, the added compute node and the host are shut down, to join the host to the XenServer pool. The operation fails, if any server other than the compute node is running/suspended on your host.

Block migration

Prerequisites

- **Compatible XenServer hypervisors**. The hypervisors must support the Storage XenMotion feature. See your XenServer manual to make sure your edition has this feature.

Notes

- To use block migration, you must use the `--block-migrate` parameter with the live migration command.

- Block migration works only with EXT local storage SRs, and the server must not have any volumes attached.

Configuring Resize

Resize (or Server resize) is the ability to change the flavor of a server, thus allowing it to upscale or downscale according to user needs. For this feature to work properly, some underlying virt layers may need further configuration; this section describes the required configuration steps for each hypervisor layer provided by OpenStack.

XenServer

To get resize to work with XenServer (and XCP), please refer to the Dom0 Modifications for Resize/Migration Support section in the OpenStack Compute Administration Guide.

Database configuration

You can configure OpenStack Compute to use any SQLAlchemy-compatible database. The database name is `nova`. The `nova-conductor` service is the only service that writes to the database. The other Compute services access the database through the `nova-conductor` service.

To ensure that the database schema is current, run the following command:

```
$ nova-manage db sync
```

If `nova-conductor` is not used, entries to the database are mostly written by the `nova-scheduler` service, although all the services need to be able to update entries in the database.

In either case, use these settings to configure the connection string for the nova database.

Table 2.8. Description of configuration options for db

Configuration option=Default value	Description
backend=sqlalchemy	(StrOpt) The backend to use for db
connection_trace=False	(BoolOpt) Add python stack traces to SQL as comment strings
connection=sqlite:////home/docwork/openstack-manuals-new/tools/autogenerate-config-docs/nova/nova/openstack/common/db/$sqlite_db	(StrOpt) The SQLAlchemy connection string used to connect to the database
connection_debug=0	(IntOpt) Verbosity of SQL debugging information. 0=None, 100=Everything
db_backend=sqlalchemy	(StrOpt) The backend to use for bare-metal database
db_check_interval=60	(IntOpt) Seconds between getting fresh cell info from db.
db_driver=nova.db	(StrOpt) driver to use for database access
idle_timeout=3600	(IntOpt) timeout before idle sql connections are reaped
max_pool_size=None	(IntOpt) Maximum number of SQL connections to keep open in a pool
max_overflow=None	(IntOpt) If set, use this value for max_overflow with sqlalchemy
max_retries=10	(IntOpt) maximum db connection retries during startup. (setting -1 implies an infinite retry count)

Configuration option=Default value	Description
min_pool_size=1	(IntOpt) Minimum number of SQL connections to keep open in a pool
pool_timeout=None	(IntOpt) If set, use this value for pool_timeout with sqlalchemy
retry_interval=10	(IntOpt) interval between retries of opening a sql connection
slave_connection=	(StrOpt) The SQLAlchemy connection string used to connect to the slave database
sql_connection=sqlite:///$state_path/baremetal_$sqlite_db	(StrOpt) The SQLAlchemy connection string used to connect to the bare-metal database
sqlite_db=nova.sqlite	(StrOpt) the filename to use with sqlite
sqlite_synchronous=True	(BoolOpt) If true, use synchronous mode for sqlite

Components Configuration

Configure the Oslo RPC messaging system

OpenStack projects use AMQP, an open standard for messaging middleware. OpenStack services that run on multiple servers to talk to each other. OpenStack Oslo RPC supports three implementations of AMQP: RabbitMQ, Qpid, and ZeroMQ.

Configure RabbitMQ

OpenStack Oslo RPC uses RabbitMQ by default. Use these options to configure the RabbitMQ message system. The `rpc_backend` option is not required as long as RabbitMQ is the default messaging system. However, if it is included the configuration, you must set it to `nova.openstack.common.rpc.impl_kombu`.

```
rpc_backend=nova.openstack.common.rpc.impl_kombu
```

You can use these additional options to configure the RabbitMQ messaging system. You can configure messaging communication for different installation scenarios, tune retries for RabbitMQ, and define the size of the RPC thread pool. To monitor notifications through RabbitMQ, you must set the `notification_driver` option to `nova.notifier.rabbit_notifier` in the `nova.conf` file. The default for sending usage data is sixty seconds plus a random number of seconds from zero to sixty.

Table 2.9. Description of configuration options for rabbitmq

Configuration option=Default value	Description
rabbit_ha_queues=False	(BoolOpt) use H/A queues in RabbitMQ (x-ha-policy: all).You need to wipe RabbitMQ database when changing this option.
rabbit_host=localhost	(StrOpt) The RabbitMQ broker address where a single node is used
rabbit_hosts=$rabbit_host:$rabbit_port	(ListOpt) RabbitMQ HA cluster host:port pairs
rabbit_max_retries=0	(IntOpt) maximum retries with trying to connect to RabbitMQ (the default of 0 implies an infinite retry count)
rabbit_password=guest	(StrOpt) the RabbitMQ password
rabbit_port=5672	(IntOpt) The RabbitMQ broker port where a single node is used

Configuration option=Default value	Description
rabbit_retry_backoff=2	(IntOpt) how long to backoff for between retries when connecting to RabbitMQ
rabbit_retry_interval=1	(IntOpt) how frequently to retry connecting with RabbitMQ
rabbit_use_ssl=False	(BoolOpt) connect over SSL for RabbitMQ
rabbit_userid=guest	(StrOpt) the RabbitMQ userid
rabbit_virtual_host=/	(StrOpt) the RabbitMQ virtual host

Table 2.10. Description of configuration options for kombu

Configuration option=Default value	Description
kombu_ssl_ca_certs=	(StrOpt) SSL certification authority file (valid only if SSL enabled)
kombu_ssl_certfile=	(StrOpt) SSL cert file (valid only if SSL enabled)
kombu_ssl_keyfile=	(StrOpt) SSL key file (valid only if SSL enabled)
kombu_ssl_version=	(StrOpt) SSL version to use (valid only if SSL enabled). valid values are TLSv1, SSLv23 and SSLv3. SSLv2 may be available on some distributions

Configure Qpid

Use these options to configure the Qpid messaging system for OpenStack Oslo RPC. Qpid is not the default messaging system, so you must enable it by setting the `rpc_backend` option in the `nova.conf` file.

```
rpc_backend=nova.openstack.common.rpc.impl_qpid
```

This critical option points the compute nodes to the Qpid broker (server). Set `qpid_hostname` to the host name where the broker runs in the `nova.conf` file.

Note

The `--qpid_hostname` option accepts a host name or IP address value.

```
qpid_hostname=hostname.example.com
```

If the Qpid broker listens on a port other than the AMQP default of 5672, you must set the `qpid_port` option to that value:

```
qpid_port=12345
```

If you configure the Qpid broker to require authentication, you must add a user name and password to the configuration:

```
qpid_username=username
qpid_password=password
```

By default, TCP is used as the transport. To enable SSL, set the `qpid_protocol` option:

```
qpid_protocol=ssl
```

This table lists additional options that you use to configure the Qpid messaging driver for OpenStack Oslo RPC. These options are used infrequently.

Table 2.11. Description of configuration options for qpid

Configuration option=Default value	Description
qpid_heartbeat=60	(IntOpt) Seconds between connection keepalive heartbeats
qpid_hostname=localhost	(StrOpt) Qpid broker hostname
qpid_hosts=$qpid_hostname:$qpid_port	(ListOpt) Qpid HA cluster host:port pairs
qpid_password=	(StrOpt) Password for qpid connection
qpid_port=5672	(IntOpt) Qpid broker port
qpid_protocol=tcp	(StrOpt) Transport to use, either 'tcp' or 'ssl'
qpid_sasl_mechanisms=	(StrOpt) Space separated list of SASL mechanisms to use for auth
qpid_tcp_nodelay=True	(BoolOpt) Disable Nagle algorithm
qpid_topology_version=1	(IntOpt) The qpid topology version to use. Version 1 is what was originally used by impl_qpid. Version 2 includes some backwards-incompatible changes that allow broker federation to work. Users should update to version 2 when they are able to take everything down, as it requires a clean break.
qpid_username=	(StrOpt) Username for qpid connection

Configure ZeroMQ

Use these options to configure the ZeroMQ messaging system for OpenStack Oslo RPC. ZeroMQ is not the default messaging system, so you must enable it by setting the `rpc_backend` option in the `nova.conf` file.

Table 2.12. Description of configuration options for zeromq

Configuration option=Default value	Description
rpc_zmq_bind_address=*	(StrOpt) ZeroMQ bind address. Should be a wildcard (*), an ethernet interface, or IP. The "host" option should point or resolve to this address.
rpc_zmq_contexts=1	(IntOpt) Number of ZeroMQ contexts, defaults to 1
rpc_zmq_host=docwork	(StrOpt) Name of this node. Must be a valid hostname, FQDN, or IP address. Must match "host" option, if running Nova.
rpc_zmq_ipc_dir=/var/run/openstack	(StrOpt) Directory for holding IPC sockets
rpc_zmq_matchmaker=nova.openstack.common.rpc.matchmaker.MatchMakerLocalhost	(StrOpt) MatchMaker driver
rpc_zmq_port=9501	(IntOpt) ZeroMQ receiver listening port
rpc_zmq_topic_backlog=None	(IntOpt) Maximum number of ingress messages to locally buffer per topic. Default is unlimited.

Configure messaging

Use these options to configure the RabbitMQ and Qpid messaging drivers.

Table 2.13. Description of configuration options for rpc

Configuration option=Default value	Description
amqp_durable_queues=False	(BoolOpt) Use durable queues in amqp.
amqp_auto_delete=False	(BoolOpt) Auto-delete queues in amqp.

Configuration option=Default value	Description
baseapi=None	(StrOpt) Set a version cap for messages sent to the base api in any service
control_exchange=openstack	(StrOpt) AMQP exchange to connect to if using RabbitMQ or Qpid
matchmaker_heartbeat_freq=300	(IntOpt) Heartbeat frequency
matchmaker_heartbeat_ttl=600	(IntOpt) Heartbeat time-to-live.
ringfile=/etc/oslo/matchmaker_ring.json	(StrOpt) Matchmaker ring file (JSON)
rpc_backend=nova.openstack.common.rpc.impl_kombu	(StrOpt) The messaging module to use, defaults to kombu.
rpc_cast_timeout=30	(IntOpt) Seconds to wait before a cast expires (TTL). Only supported by impl_zmq.
rpc_conn_pool_size=30	(IntOpt) Size of RPC connection pool
rpc_driver_queue_base=cells.intercell	(StrOpt) Base queue name to use when communicating between cells. Various topics by message type will be appended to this.
rpc_response_timeout=60	(IntOpt) Seconds to wait for a response from call or multicall
rpc_thread_pool_size=64	(IntOpt) Size of RPC thread pool
topics=notifications	(ListOpt) AMQP topic(s) used for OpenStack notifications

Configure the Compute API

The Compute API, run by the `nova-api` daemon, is the component of OpenStack Compute that receives and responds to user requests, whether they be direct API calls, or via the CLI tools or dashboard.

Configure Compute API password handling

The OpenStack Compute API enables users to specify an administrative password when they create or rebuild a server instance. If the user does not specify a password, a random password is generated and returned in the API response.

In practice, how the admin password is handled depends on the hypervisor in use and might require additional configuration of the instance. For example, you might have to install an agent to handle the password setting. If the hypervisor and instance configuration do not support setting a password at server create time, the password that is returned by the create API call is misleading because it was ignored.

To prevent this confusion, use the `enable_instance_password` configuration option to disable the return of the admin password for installations that do not support setting instance passwords.

Configure Compute API rate limiting

OpenStack Compute supports API rate limiting for the OpenStack API. The rate limiting allows an administrator to configure limits on the type and number of API calls that can be made in a specific time interval.

When API rate limits are exceeded, HTTP requests return an error with a status code of 413 Request entity too large, and includes an HTTP `Retry-After` header. The response body includes the error details and the delay before you should retry the request.

Rate limiting is not available for the EC2 API.

Define limits

To define limits, set these values:

- The **HTTP method** used in the API call, typically one of GET, PUT, POST, or DELETE.
- A **human readable URI** that is used as a friendly description of where the limit is applied.
- A **regular expression**. The limit is applied to all URIs that match the regular expression and HTTP method.
- A **limit value** that specifies the maximum count of units before the limit takes effect.
- An **interval** that specifies time frame to which the limit is applied. The interval can be SECOND, MINUTE, HOUR, or DAY.

Rate limits are applied in relative order to the HTTP method, going from least to most specific. For example, although the default threshold for POST to */servers is 50 each day, you cannot POST to */servers more than 10 times in a single minute because the rate limits for any POST is 10 each minute.

Default limits

Normally, you install OpenStack Compute with the following limits enabled:

Table 2.14. Default API rate limits

HTTP method	API URI	API regular expression	Limit
POST	any URI (*)	.*	10 per minute
POST	/servers	^/servers	50 per day
PUT	any URI (*)	.*	10 per minute
GET	*changes-since*	.*changes-since.*	3 per minute
DELETE	any URI (*)	.*	100 per minute

Configure and change limits

As part of the WSGI pipeline, the `etc/nova/api-paste.ini` file defines the actual limits.

To enable limits, include the `ratelimit`' filter in the API pipeline specification. If the `ratelimit` filter is removed from the pipeline, limiting is disabled. You must also define the rate limit filter. The lines appear as follows:

```
[pipeline:openstack_compute_api_v2]
pipeline = faultwrap authtoken keystonecontext ratelimit osapi_compute_app_v2

[pipeline:openstack_volume_api_v1]
pipeline = faultwrap authtoken keystonecontext ratelimit osapi_volume_app_v1

[filter:ratelimit]
paste.filter_factory = nova.api.openstack.compute.
limits:RateLimitingMiddleware.factory
```

To modify the limits, add a `limits` specification to the `[filter:ratelimit]` section of the file. Specify the limits in this order:

1. HTTP method
2. friendly URI
3. regex
4. limit
5. interval

The following example shows the default rate-limiting values:

```
[filter:ratelimit]
paste.filter_factory = nova.api.openstack.compute.
limits:RateLimitingMiddleware.factory
limits =(POST, "*", .*, 10, MINUTE);(POST, "*/servers", ^/servers, 50, DAY);
(PUT, "*", .*, 10, MINUTE);(GET, "*changes-since*", .*changes-since.*, 3,
 MINUTE);(DELETE, "*", .*, 100, MINUTE)
```

Configuration reference

The following table lists the Compute API configuration options:

Table 2.15. Description of configuration options for api

Configuration option=Default value	Description
enable_new_services=True	(BoolOpt) Services to be added to the available pool on create
enabled_apis=ec2,osapi_compute,metadata	(ListOpt) a list of APIs to enable by default
enabled_ssl_apis=	(ListOpt) a list of APIs with enabled SSL
instance_name_template=instance-%08x	(StrOpt) Template string to be used to generate instance names
multi_instance_display_name_template=%(name)s-%(uuid)s	(StrOpt) When creating multiple instances with a single request using the os-multiple-create API extension, this template will be used to build the display name for each instance. The benefit is that the instances end up with different hostnames. To restore legacy behavior of every instance having the same name, set this option to "%(name)s". Valid keys for the template are: name, uuid, count.
non_inheritable_image_properties=cache_in_nova,bittorrent	(ListOpt) These are image properties which a snapshot should not inherit from an instance
null_kernel=nokernel	(StrOpt) kernel image that indicates not to use a kernel, but to use a raw disk image instead
osapi_compute_ext_list=	(ListOpt) Specify list of extensions to load when using osapi_compute_extension option with nova.api.openstack.compute.contrib.select_extensions
osapi_compute_extension=['nova.api.openstack.compute.contrib.standard_extensions']	(MultiStrOpt) osapi compute extension to load
osapi_compute_link_prefix=None	(StrOpt) Base URL that will be presented to users in links to the OpenStack Compute API
osapi_compute_listen=0.0.0.0	(StrOpt) IP address for OpenStack API to listen
osapi_compute_listen_port=8774	(IntOpt) list port for osapi compute

Configuration option=Default value	Description
osapi_compute_workers=None	(IntOpt) Number of workers for OpenStack API service
osapi_hide_server_address_states=building	(ListOpt) List of instance states that should hide network info
servicegroup_driver=db	(StrOpt) The driver for servicegroup service (valid options are: db, zk, mc)
snapshot_name_template=snapshot-%s	(StrOpt) Template string to be used to generate snapshot names
use_forwarded_for=False	(BoolOpt) Treat X-Forwarded-For as the canonical remote address. Only enable this if you have a sanitizing proxy.
use_tpool=False	(BoolOpt) Enable the experimental use of thread pooling for all DB API calls

Configure the EC2 API

You can set options in the `nova.conf` configuration file to control which network address and port the EC2 API listens on, the formatting of some API responses, and authentication related options.

To customize these options for OpenStack EC2 API, use these configuration option settings:

Table 2.16. Description of configuration options for ec2

Configuration option=Default value	Description
ec2_dmz_host=$my_ip	(StrOpt) the internal ip of the ec2 api server
ec2_host=$my_ip	(StrOpt) the ip of the ec2 api server
ec2_listen=0.0.0.0	(StrOpt) IP address for EC2 API to listen
ec2_listen_port=8773	(IntOpt) port for ec2 api to listen
ec2_path=/services/Cloud	(StrOpt) the path prefix used to call the ec2 api server
ec2_port=8773	(IntOpt) the port of the ec2 api server
ec2_private_dns_show_ip=False	(BoolOpt) Return the IP address as private dns hostname in describe instances
ec2_scheme=http	(StrOpt) the protocol to use when connecting to the ec2 api server (http, https)
ec2_strict_validation=True	(BoolOpt) Validate security group names according to EC2 specification
ec2_timestamp_expiry=300	(IntOpt) Time in seconds before ec2 timestamp expires
ec2_workers=None	(IntOpt) Number of workers for EC2 API service
keystone_ec2_url=http://localhost:5000/v2.0/ec2tokens	(StrOpt) URL to get token from ec2 request.
lockout_attempts=5	(IntOpt) Number of failed auths before lockout.
lockout_minutes=15	(IntOpt) Number of minutes to lockout if triggered.
lockout_window=15	(IntOpt) Number of minutes for lockout window.
region_list=	(ListOpt) list of region=fqdn pairs separated by commas

Configure quotas

To prevent system capacities from being exhausted without notification, you can set up quotas. Quotas are operational limits. For example, the number of gigabytes allowed per tenant can be controlled so that cloud resources are optimized. Quotas are currently enforced at the tenant (or project) level, rather than by user.

Manage Compute service quotas

As an administrative user, you can use the **nova quota-*** commands, which are provided by the `python-novaclient` package, to update the Compute Service quotas for a specific tenant or tenant user, as well as update the quota defaults for a new tenant.

Table 2.17. Compute Quota Descriptions

Quota Name	Description
`cores`	Number of instance cores (VCPUs) allowed per tenant.
`fixed-ips`	Number of fixed IP addresses allowed per tenant. This number must be equal to or greater than the number of allowed instances.
`floating-ips`	Number of floating IP addresses allowed per tenant.
`injected-file-content-bytes`	Number of content bytes allowed per injected file.
`injected-file-path-bytes`	Number of bytes allowed per injected file path.
`injected-files`	Number of injected files allowed per tenant.
`instances`	Number of instances allowed per tenant.
`key-pairs`	Number of key pairs allowed per user.
`metadata-items`	Number of metadata items allowed per instance.
`ram`	Megabytes of instance ram allowed per tenant.
`security-groups`	Number of security groups per tenant.
`security-group-rules`	Number of rules per security group.

View and update Compute quotas for a tenant (project)

Procedure 2.3. To view and update default quota values

1. List all default quotas for all tenants, as follows:

```
$ nova quota-defaults
```

For example:

```
$ nova quota-defaults
+-----------------------------+-------+
| Quota                       | Limit |
+-----------------------------+-------+
| instances                   | 10    |
| cores                       | 20    |
| ram                         | 51200 |
| floating_ips                | 10    |
| fixed_ips                   | -1    |
| metadata_items              | 128   |
| injected_files              | 5     |
| injected_file_content_bytes | 10240 |
| injected_file_path_bytes    | 255   |
| key_pairs                   | 100   |
| security_groups             | 10    |
| security_group_rules        | 20    |
+-----------------------------+-------+
```

2. Update a default value for a new tenant, as follows:

```
$ nova quota-class-update --key value default
```

For example:

```
$ nova quota-class-update --instances 15 default
```

Procedure 2.4. To view quota values for an existing tenant (project)

1. Place the tenant ID in a useable variable, as follows:

```
$ tenant=$(keystone tenant-list | awk '/tenantName/ {print $2}')
```

2. List the currently set quota values for a tenant, as follows:

```
$ nova quota-show --tenant $tenant
```

For example:

```
$ nova quota-show --tenant $tenant
+-----------------------------+-------+
| Quota                       | Limit |
+-----------------------------+-------+
| instances                   | 10    |
| cores                       | 20    |
| ram                         | 51200 |
| floating_ips                | 10    |
| fixed_ips                   | -1    |
| metadata_items              | 128   |
| injected_files              | 5     |
| injected_file_content_bytes | 10240 |
| injected_file_path_bytes    | 255   |
| key_pairs                   | 100   |
| security_groups             | 10    |
| security_group_rules        | 20    |
+-----------------------------+-------+
```

Procedure 2.5. To update quota values for an existing tenant (project)

1. Obtain the tenant ID, as follows:

```
$ tenant=$(keystone tenant-list | awk '/tenantName/ {print $2}')
```

2. Update a particular quota value, as follows:

```
# nova quota-update --quotaName quotaValue tenantID
```

For example:

```
# nova quota-update --floating-ips 20 $tenant
# nova quota-show --tenant $tenant
+-----------------------------+-------+
| Quota                       | Limit |
+-----------------------------+-------+
| instances                   | 10    |
| cores                       | 20    |
| ram                         | 51200 |
| floating_ips                | 20    |
| fixed_ips                   | -1    |
| metadata_items              | 128   |
| injected_files              | 5     |
| injected_file_content_bytes | 10240 |
```

```
| injected_file_path_bytes    | 255   |
| key_pairs                   | 100   |
| security_groups             | 10    |
| security_group_rules        | 20    |
+-----------------------------+-------+
```

Note

To view a list of options for the **quota-update** command, run:

```
$ nova help quota-update
```

View and update Compute quotas for a tenant user

Procedure 2.6. To view quota values for a tenant user

1. Place the user ID in a useable variable, as follows:

```
$ tenantUser=$(keystone user-list | awk '/userName/ {print $2}')
```

2. Place the user's tenant ID in a useable variable, as follows:

```
$ tenant=$(keystone tenant-list | awk '/tenantName/ {print $2}')
```

3. List the currently set quota values for a tenant user, as follows:

```
$ nova quota-show --user $tenantUser --tenant $tenant
```

For example:

```
$ nova quota-show --user $tenantUser --tenant $tenant
+-----------------------------+-------+
| Quota                       | Limit |
+-----------------------------+-------+
| instances                   | 10    |
| cores                       | 20    |
| ram                         | 51200 |
| floating_ips                | 20    |
| fixed_ips                   | -1    |
| metadata_items              | 128   |
| injected_files              | 5     |
| injected_file_content_bytes | 10240 |
| injected_file_path_bytes    | 255   |
| key_pairs                   | 100   |
| security_groups             | 10    |
| security_group_rules        | 20    |
+-----------------------------+-------+
```

Procedure 2.7. To update quota values for a tenant user

1. Place the user ID in a useable variable, as follows:

```
$ tenantUser=$(keystone user-list | awk '/userName/ {print $2}')
```

2. Place the user's tenant ID in a useable variable, as follows:

```
$ tenant=$(keystone tenant-list | awk '/userName/ {print $2}')
```

3. Update a particular quota value, as follows:

```
# nova quota-update  --user $tenantUser --quotaName quotaValue $tenant
```

For example:

```
# nova quota-update --user $tenantUser --floating-ips 12 $tenant
# nova quota-show --user $tenantUser --tenant $tenant
+-----------------------------+-------+
| Quota                       | Limit |
+-----------------------------+-------+
| instances                   | 10    |
| cores                       | 20    |
| ram                         | 51200 |
| floating_ips                | 12    |
| fixed_ips                   | -1    |
| metadata_items              | 128   |
| injected_files              | 5     |
| injected_file_content_bytes | 10240 |
| injected_file_path_bytes    | 255   |
| key_pairs                   | 100   |
| security_groups             | 10    |
| security_group_rules        | 20    |
+-----------------------------+-------+
```

Note

To view a list of options for the **quota-update** command, run:

```
$ nova help quota-update
```

Configure remote console access

To provide a remote console or remote desktop access to guest virtual machines, use VNC or SPICE HTML5 through either the OpenStack dashboard or the command line. Best practice is to select one or the other to run.

VNC console proxy

The VNC proxy is an OpenStack component that enables compute service users to access their instances through VNC clients.

The VNC console connection works as follows:

1. A user connects to the API and gets an `access_url` such as, `http://ip:port/?token=xyz`.
2. The user pastes the URL in a browser or uses it as a client parameter.
3. The browser or client connects to the proxy.
4. The proxy talks to `nova-consoleauth` to authorize the token for the user, and maps the token to the *private* host and port of the VNC server for an instance.

   The compute host specifies the address that the proxy should use to connect through the `nova.conf` file option, `vncserver_proxyclient_address`. In this way, the VNC proxy works as a bridge between the public network and private host network.

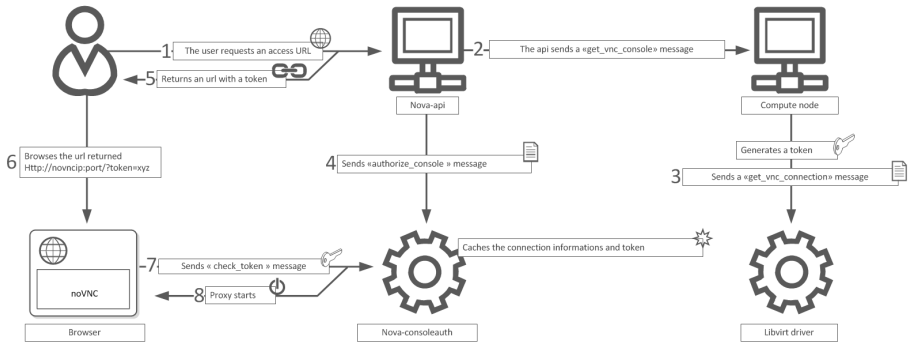
5. The proxy initiates the connection to VNC server and continues to proxy until the session ends.

The proxy also tunnels the VNC protocol over WebSockets so that the noVNC client can talk VNC.

In general, the VNC proxy:

- Bridges between the public network where the clients live and the private network where vncservers live.
- Mediates token authentication.
- Transparently deals with hypervisor-specific connection details to provide a uniform client experience.

Figure 2.3. noVNC process



About nova-consoleauth

Both client proxies leverage a shared service to manage token authentication called `nova-consoleauth`. This service must be running for either proxy to work. Many proxies of either type can be run against a single `nova-consoleauth` service in a cluster configuration.

Do not confuse the `nova-consoleauth` shared service with `nova-console`, which is a XenAPI-specific service that most recent VNC proxy architectures do not use.

Typical deployment

A typical deployment has the following components:

- A `nova-consoleauth` process. Typically runs on the controller host.
- One or more `nova-novncproxy` services. Supports browser-based noVNC clients. For simple deployments, this service typically runs on the same machine as `nova-api` because it operates as a proxy between the public network and the private compute host network.
- One or more `nova-xvpvncproxy` services. Supports the special Java client discussed here. For simple deployments, this service typically runs on the same machine as `nova-`

`api` because it acts as a proxy between the public network and the private compute host network.

- One or more compute hosts. These compute hosts must have correctly configured options, as follows.

VNC configuration options

Table 2.18. Description of configuration options for vnc

Configuration option=Default value	Description
novncproxy_base_url=http://127.0.0.1:6080/vnc_auto.html	(StrOpt) location of vnc console proxy, in the form "http://127.0.0.1:6080/vnc_auto.html"
vnc_enabled=True	(BoolOpt) enable vnc related features
vnc_keymap=en-us	(StrOpt) keymap for vnc
vnc_password=None	(StrOpt) VNC password
vnc_port=5900	(IntOpt) VNC starting port
vnc_port_total=10000	(IntOpt) Total number of VNC ports
vncserver_listen=127.0.0.1	(StrOpt) IP address on which instance vncservers should listen
vncserver_proxyclient_address=127.0.0.1	(StrOpt) the address to which proxy clients (like nova-xvpvncproxy) should connect

Note

To support live migration, you cannot specify a specific IP address for `vncserver_listen`, because that IP address does not exist on the destination host.

Note

The `vncserver_proxyclient_address` defaults to `127.0.0.1`, which is the address of the compute host that nova instructs proxies to use when connecting to instance servers.

For all-in-one XenServer domU deployments, set this to 169.254.0.1.

For multi-host XenServer domU deployments, set to a dom0 management IP on the same network as the proxies.

For multi-host libvirt deployments, set to a host management IP on the same network as the proxies.

nova-novncproxy (noVNC)

You must install the noVNC package, which contains the `nova-novncproxy` service.

As root, run the following command:

```
# apt-get install novnc
```

The service starts automatically on installation.

To restart it, run the following command:

```
# service novnc restart
```

The configuration option parameter should point to your `nova.conf` file, which includes the message queue server address and credentials.

By default, `nova-novncproxy` binds on `0.0.0.0:6080`.

To connect the service to your nova deployment, add the following configuration options to your `nova.conf` file:

- `vncserver_listen=0.0.0.0`

  Specifies the address on which the VNC service should bind. Make sure it is assigned one of the compute node interfaces. This address is the one used by your domain file.

  ```
  <graphics type="vnc" autoport="yes" keymap="en-us" listen="0.0.0.0"/>
  ```

  Note

  To use live migration, use the `0.0.0.0` address.

- `vncserver_ proxyclient_ address =127.0.0.1`

  The address of the compute host that nova instructs proxies to use when connecting to instance `vncservers`.

Frequently asked questions about VNC access to virtual machines

- **Q: What is the difference between `nova-xvpvncproxy` and `nova-novncproxy`?**

  A: `nova-xvpvncproxy`, which ships with nova, is a proxy that supports a simple Java client. `nova-novncproxy` uses noVNC to provide VNC support through a web browser.

- **Q: I want VNC support in the Dashboard. What services do I need?**

  A: You need `nova-novncproxy`, `nova-consoleauth`, and correctly configured compute hosts.

- **Q: When I use nova get-vnc-console or click on the VNC tab of the Dashboard, it hangs. Why?**

  A: Make sure you are running `nova-consoleauth` (in addition to `nova-novncproxy`). The proxies rely on `nova-consoleauth` to validate tokens, and waits for a reply from them until a timeout is reached.

- **Q: My VNC proxy worked fine during my all-in-one test, but now it doesn't work on multi host. Why?**

  A: The default options work for an all-in-one install, but changes must be made on your compute hosts once you start to build a cluster. As an example, suppose you have two servers:

```
PROXYSERVER (public_ip=172.24.1.1, management_ip=192.168.1.1)
COMPUTESERVER (management_ip=192.168.1.2)
```

Your `nova-compute` configuration file must set the following values:

```
# These flags help construct a connection data structure
vncserver_proxyclient_address=192.168.1.2
novncproxy_base_url=http://172.24.1.1:6080/vnc_auto.html
xvpvncproxy_base_url=http://172.24.1.1:6081/console

# This is the address where the underlying vncserver (not the proxy)
# will listen for connections.
vncserver_listen=192.168.1.2
```

Note

`novncproxy_base_url` and `xvpvncproxy_base_url` use a public IP; this is the URL that is ultimately returned to clients, which generally do not have access to your private network. Your PROXYSERVER must be able to reach `vncserver_proxyclient_address`, because that is the address over which the VNC connection is proxied.

- **Q: My noVNC does not work with recent versions of web browsers. Why?**

  A: Make sure you have installed `python-numpy`, which is required to support a newer version of the WebSocket protocol (HyBi-07+).

- **Q: How do I adjust the dimensions of the VNC window image in the OpenStack dashboard?**

  A: These values are hard-coded in a Django HTML template. To alter them, edit the `_detail_vnc.html` template file. The location of this file varies based on Linux distribution. On Ubuntu 12.04, the file is at `/usr/share/pyshared/horizon/dashboards/nova/instances/templates/instances/_detail_vnc.html`.

  Modify the `width` and `height` options, as follows:

```
<iframe src="{{ vnc_url }}" width="720" height="430"></iframe>
```

SPICE console

OpenStack Compute supports VNC consoles to guests. The VNC protocol is fairly limited, lacking support for multiple monitors, bi-directional audio, reliable cut-and-paste, video streaming and more. SPICE is a new protocol that aims to address the limitations in VNC and provide good remote desktop support.

SPICE support in OpenStack Compute shares a similar architecture to the VNC implementation. The OpenStack dashboard uses a SPICE-HTML5 widget in its console tab that communicates to the `nova-spicehtml5proxy` service by using SPICE-over-websockets. The `nova-spicehtml5proxy` service communicates directly with the hypervisor process by using SPICE.

Note

If you do not configure SPICE correctly, Compute falls back on VNC.

The following table shows the options to configure SPICE as the console for OpenStack Compute:

Table 2.19. Description of configuration options for spice

Configuration option=Default value	Description
agent_enabled=True	(BoolOpt) enable spice guest agent support
enabled=False	(BoolOpt) enable spice related features
enabled=False	(BoolOpt) Whether the V3 API is enabled or not
html5proxy_base_url=http://127.0.0.1:6082/spice_auto.html	(StrOpt) location of spice html5 console proxy, in the form "http://127.0.0.1:6082/spice_auto.html"
keymap=en-us	(StrOpt) keymap for spice
server_listen=127.0.0.1	(StrOpt) IP address on which instance spice server should listen
server_proxyclient_address=127.0.0.1	(StrOpt) the address to which proxy clients (like nova-spicehtml5proxy) should connect

Configuring Compute Service Groups

To effectively manage and utilize compute nodes, the Compute service must know their statuses. For example, when a user launches a new VM, the Compute scheduler should send the request to a live node (with enough capacity too, of course). From the Grizzly release and later, the Compute service queries the ServiceGroup API to get the node liveness information.

When a compute worker (running the `nova-compute` daemon) starts, it calls the join API to join the compute group, so that every service that is interested in the information (for example, the scheduler) can query the group membership or the status of a particular node. Internally, the ServiceGroup client driver automatically updates the compute worker status.

The following drivers are implemented: database and ZooKeeper. Further drivers are in review or development, such as memcache.

Database ServiceGroup driver

Compute uses the database driver, which is the default driver, to track node liveness. In a compute worker, this driver periodically sends a **db update** command to the database, saying “I'm OK” with a timestamp. A pre-defined timeout (`service_down_time`) determines if a node is dead.

The driver has limitations, which may or may not be an issue for you, depending on your setup. The more compute worker nodes that you have, the more pressure you put on the database. By default, the timeout is 60 seconds so it might take some time to detect node failures. You could reduce the timeout value, but you must also make the DB update more frequently, which again increases the DB workload.

Fundamentally, the data that describes whether the node is alive is "transient" — After a few seconds, this data is obsolete. Other data in the database is persistent, such as the entries that describe who owns which VMs. However, because this data is stored in the same database, is treated the same way. The ServiceGroup abstraction aims to treat them separately.

ZooKeeper ServiceGroup driver

The ZooKeeper ServiceGroup driver works by using ZooKeeper ephemeral nodes. ZooKeeper, in contrast to databases, is a distributed system. Its load is divided among several servers. At a compute worker node, after establishing a ZooKeeper session, it creates an ephemeral znode in the group directory. Ephemeral znodes have the same lifespan as the session. If the worker node or the `nova-compute` daemon crashes, or a network partition is in place between the worker and the ZooKeeper server quorums, the ephemeral znodes are removed automatically. The driver gets the group membership by running the **ls** command in the group directory.

To use the ZooKeeper driver, you must install ZooKeeper servers and client libraries. Setting up ZooKeeper servers is outside the scope of this article. For the rest of the article, assume these servers are installed, and their addresses and ports are `192.168.2.1:2181`, `192.168.2.2:2181`, `192.168.2.3:2181`.

To use ZooKeeper, you must install client-side Python libraries on every nova node: `python-zookeeper` – the official Zookeeper Python binding and `evzookeeper` – the library to make the binding work with the eventlet threading model.

The relevant configuration snippet in the `/etc/nova/nova.conf` file on every node is:

```
servicegroup_driver="zk"

[zookeeper]
address="192.168.2.1:2181,192.168.2.2:2181,192.168.2.3:2181"
```

Table 2.20. Description of configuration options for zookeeper

Configuration option=Default value	Description
address=None	(StrOpt) The ZooKeeper addresses for servicegroup service in the format of host1:port,host2:port,host3:port
recv_timeout=4000	(IntOpt) recv_timeout parameter for the zk session
sg_prefix=/servicegroups	(StrOpt) The prefix used in ZooKeeper to store ephemeral nodes
sg_retry_interval=5	(IntOpt) Number of seconds to wait until retrying to join the session

Fibre Channel support in Compute

Fibre Channel support in OpenStack Compute is remote block storage attached to Compute nodes for VMs.

In the Grizzly release, Fibre Channel supported only the KVM hypervisor.

Compute and Block Storage for Fibre Channel do not support automatic zoning. Fibre Channel arrays must be pre-zoned or directly attached to the KVM hosts.

KVM host requirements

You must install these packages on the KVM host:

- sysfstools - Nova uses the systool application in this package.
- sg3-utils - Nova uses the sg_scan and sginfo applications.

Installing the multipath-tools package is optional.

Install required packages

Use these commands to install the system packages:

- For systems running Ubuntu:

```
$ sudo apt-get install sysfstools sg3-utils multipath-tools
```

- For systems running Red Hat:

```
$ sudo yum install sysfstools sg3_utils multipath-tools
```

Configure multiple Compute nodes

To distribute your VM load across more than one server, you can connect an additional `nova-compute` node to a cloud controller node. You can reproduce this configuration on multiple compute servers to build a true multi-node OpenStack Compute cluster.

To build and scale the Compute platform, you distribute services across many servers. While you can accomplish this in other ways, this section describes how to add compute nodes and scale out the `nova-compute` service.

For a multi-node installation, you make changes to only the `nova.conf` file and copy it to additional compute nodes. Ensure that each `nova.conf` file points to the correct IP addresses for the respective services.

1. By default, `nova-network` sets the bridge device based on the setting in `flat_network_bridge`. Update your IP information in the `/etc/network/interfaces` file by using this template:

```
# The loopback network interface
auto lo
    iface lo inet loopback

# The primary network interface
auto br100
iface br100 inet static
    bridge_ports    eth0
    bridge_stp      off
    bridge_maxwait  0
    bridge_fd       0
    address xxx.xxx.xxx.xxx
    netmask xxx.xxx.xxx.xxx
    network xxx.xxx.xxx.xxx
    broadcast xxx.xxx.xxx.xxx
    gateway xxx.xxx.xxx.xxx
    # dns-* options are implemented by the resolvconf package, if
 installed
    dns-nameservers xxx.xxx.xxx.xxx
```

2. Restart networking:

```
$ sudo service networking restart
```

3. Bounce the relevant services to take the latest updates:

```
$ sudo service libvirtd restart
$ sudo service nova-compute restart
```

4. To avoid issues with KVM and permissions with the Compute Service, run these commands to ensure that your VMs run optimally:

```
# chgrp kvm /dev/kvm
# chmod g+rwx /dev/kvm
```

5. Any server that does not have **nova-api** running on it requires an iptables entry so that images can get metadata information.

   On compute nodes, configure iptables with this command:

```
# iptables -t nat -A PREROUTING -d 169.254.169.254/32 -p tcp -m tcp --
dport 80 -j DNAT --to-destination $NOVA_API_IP:8773
```

6. Confirm that your compute node can talk to your cloud controller.

   From the cloud controller, run this database query:

```
$ mysql -u$MYSQL_USER -p$MYSQL_PASS nova -e 'select * from services;'
```

```
+---------------------+---------------------+------------+---------
+----+----------+---------------+-----------+--------------+----------
+-------------------+
| created_at          | updated_at          | deleted_at | deleted |
 id | host     | binary         | topic     | report_count | disabled |
 availability_zone |
+---------------------+---------------------+------------+---------
+----+----------+---------------+-----------+--------------+----------
+-------------------+
| 2011-01-28 22:52:46 | 2011-02-03 06:55:48 | NULL       |       0 |  1 |
 osdemo02 | nova-network   | network   |        46064 |        0 | nova
           |
| 2011-01-28 22:52:48 | 2011-02-03 06:55:57 | NULL       |       0 |  2 |
 osdemo02 | nova-compute   | compute   |        46056 |        0 | nova
           |
| 2011-01-28 22:52:52 | 2011-02-03 06:55:50 | NULL       |       0 |  3 |
 osdemo02 | nova-scheduler | scheduler |        46065 |        0 | nova
           |
| 2011-01-29 23:49:29 | 2011-02-03 06:54:26 | NULL       |       0 |  4 |
 osdemo01 | nova-compute   | compute   |        37050 |        0 | nova
           |
| 2011-01-30 23:42:24 | 2011-02-03 06:55:44 | NULL       |       0 |  9 |
 osdemo04 | nova-compute   | compute   |        28484 |        0 | nova
           |
| 2011-01-30 21:27:28 | 2011-02-03 06:54:23 | NULL       |       0 |  8 |
 osdemo05 | nova-compute   | compute   |        29284 |        0 | nova
           |
+---------------------+---------------------+------------+---------
+----+----------+---------------+-----------+--------------+----------
+-------------------+
```

In this example, the `osdemo` hosts all run the `nova-compute` service. When you launch instances, they allocate on any node that runs `nova-compute` from this list.

Hypervisors

OpenStack Compute supports many hypervisors, which might make it difficult for you to choose one. Most installations use only one hypervisor. However you can use the section called "ComputeFilter" [112] and the section called "ImagePropertiesFilter" [113] to schedule to different hypervisors within the same installation. The following links help you choose a hypervisor. See http://wiki.openstack.org/HypervisorSupportMatrix for a detailed list of features and support across the hypervisors.

The following hypervisors are supported:

- KVM - Kernel-based Virtual Machine. The virtual disk formats that it supports it inherits from QEMU since it uses a modified QEMU program to launch the virtual machine. The supported formats include raw images, the qcow2, and VMware formats.
- LXC - Linux Containers (through libvirt), use to run Linux-based virtual machines.
- QEMU - Quick EMUlator, generally only used for development purposes.
- UML - User Mode Linux, generally only used for development purposes.
- VMWare vSphere 4.1 update 1 and newer, runs VMWare-based Linux and Windows images through a connection with a vCenter server or directly with an ESXi host.
- Xen - XenServer, Xen Cloud Platform (XCP), use to run Linux or Windows virtual machines. You must install the `nova-compute` service in a para-virtualized VM.
- PowerVM - Server virtualization with IBM PowerVM, use to run AIX, IBM i and Linux environments on IBM POWER technology.
- Hyper-V - Server virtualization with Microsoft's Hyper-V, use to run Windows, Linux, and FreeBSD virtual machines. Runs `nova-compute` natively on the Windows virtualization platform.
- Bare Metal - Not a hypervisor in the traditional sense, this driver provisions physical hardware through pluggable sub-drivers (for example, PXE for image deployment, and IPMI for power management).
- Dockeris an open-source engine which automates the deployment of >applications as highly portable, self-sufficient containers which are >independent of hardware, language, framework, packaging system and hosting >provider.

Hypervisor configuration basics

The node where the `nova-compute` service is installed and running is the machine that runs all the virtual machines, referred to as the compute node in this guide.

By default, the selected hypervisor is KVM. To change to another hypervisor, change the `libvirt_type` option in `nova.conf` and restart the `nova-compute` service.

Here are the general `nova.conf` options that are used to configure the compute node's hypervisor. Specific options for particular hypervisors can be found in following sections.

Table 2.21. Description of configuration options for hypervisor

Configuration option=Default value	Description
block_migration_flag=VIR_MIGRATE_UNDEFINE_SOURCE, VIR_MIGRATE_PEER2PEER, VIR_MIGRATE_NON_SHARED_INC	(StrOpt) Migration flags to be set for block migration
checksum_base_images=False	(BoolOpt) Write a checksum for files in _base to disk
default_ephemeral_format=None	(StrOpt) The default format an ephemeral_volume will be formatted with on creation.
disk_cachemodes=	(ListOpt) Specific cachemodes to use for different disk types e.g: ["file=directsync","block=none"]
force_raw_images=True	(BoolOpt) Force backing images to raw format
inject_password=True	(BoolOpt) Whether baremetal compute injects password or not
libvirt_cpu_mode=None	(StrOpt) Set to "host-model" to clone the host CPU feature flags; to "host-passthrough" to use the host CPU model exactly; to "custom" to use a named CPU model; to "none" to not set any CPU model. If libvirt_type="kvm|qemu", it will default to "host-model", otherwise it will default to "none"
libvirt_cpu_model=None	(StrOpt) Set to a named libvirt CPU model (see names listed in /usr/share/libvirt/cpu_map.xml). Only has effect if libvirt_cpu_mode="custom" and libvirt_type="kvm|qemu"
libvirt_disk_prefix=None	(StrOpt) Override the default disk prefix for the devices attached to a server, which is dependent on libvirt_type. (valid options are: sd, xvd, uvd, vd)
libvirt_images_rbd_ceph_conf=	(StrOpt) path to the ceph configuration file to use
libvirt_images_type=default	(StrOpt) VM Images format. Acceptable values are: raw, qcow2, lvm,rbd, default. If default is specified, then use_cow_images flag is used instead of this one.
libvirt_images_rbd_pool=rbd	(StrOpt) the RADOS pool in which rbd volumes are stored
libvirt_images_volume_group=None	(StrOpt) LVM Volume Group that is used for VM images, when you specify libvirt_images_type=lvm.
libvirt_inject_key=True	(BoolOpt) Inject the ssh public key at boot time
libvirt_inject_partition=1	(IntOpt) The partition to inject to : -2 => disable, -1 => inspect (libguestfs only), 0 => not partitioned, >0 => partition number
libvirt_inject_password=False	(BoolOpt) Inject the admin password at boot time, without an agent.
libvirt_iscsi_use_multipath=False	(BoolOpt) use multipath connection of the iSCSI volume
libvirt_iser_use_multipath=False	(BoolOpt) use multipath connection of the iSER volume
libvirt_lvm_snapshot_size=1000	(IntOpt) The amount of storage (in megabytes) to allocate for LVM snapshot copy-on-write blocks.
libvirt_nonblocking=True	(BoolOpt) Use a separated OS thread pool to realize non-blocking libvirt calls
libvirt_ovs_bridge=br-int	(StrOpt) Name of Integration Bridge used by Open vSwitch
libvirt_snapshot_compression=False	(BoolOpt) Compress snapshot images when possible. This currently applies exclusively to qcow2 images
libvirt_snapshots_directory=$instances_path/snapshots	(StrOpt) Location where libvirt driver will store snapshots before uploading them to image service
libvirt_sparse_logical_volumes=False	(BoolOpt) Create sparse logical volumes (with virtualsize) if this flag is set to True.
libvirt_type=kvm	(StrOpt) Libvirt domain type (valid options are: kvm, lxc, qemu, uml, xen)

Configuration option=Default value	Description
libvirt_uri=	(StrOpt) Override the default libvirt URI (which is dependent on libvirt_type)
libvirt_use_virtio_for_bridges=True	(BoolOpt) Use virtio for bridge interfaces with KVM/ QEMU
libvirt_vif_driver=nova.virt.libvirt.vif.LibvirtGenericVIFDriver	(StrOpt) The libvirt VIF driver to configure the VIFs.
libvirt_volume_drivers=iscsi=nova.virt.libvirt.volume.LibvirtISCSIVolumeDriver, iser=nova.virt.libvirt.volume.LibvirtISERVolumeDriver, local=nova.virt.libvirt.volume.LibvirtVolumeDriver, fake=nova.virt.libvirt.volume.LibvirtFakeVolumeDriver, rbd=nova.virt.libvirt.volume.LibvirtNetVolumeDriver, sheepdog=nova.virt.libvirt.volume.LibvirtNetVolumeDriver, nfs=nova.virt.libvirt.volume.LibvirtNFSVolumeDriver, aoe=nova.virt.libvirt.volume.LibvirtAOEVolumeDriver, glusterfs=nova.virt.libvirt.volume.LibvirtGlusterfsVolumeDriver, fibre_channel=nova.virt.libvirt.volume.LibvirtFibreChannelVolumeDriver, scality=nova.virt.libvirt.volume.LibvirtScalityVolumeDriver	(StrOpt) Libvirt handlers for remote volumes.
libvirt_wait_soft_reboot_seconds=120	(IntOpt) Number of seconds to wait for instance to shut down after soft reboot request is made. We fall back to hard reboot if instance does not shutdown within this window.
preallocate_images=none	(StrOpt) VM image preallocation mode: "none" => no storage provisioning is done up front, "space" => storage is fully allocated at instance start
remove_unused_base_images=True	(BoolOpt) Should unused base images be removed?
remove_unused_kernels=False	(BoolOpt) Should unused kernel images be removed? This is only safe to enable if all compute nodes have been updated to support this option. This will enabled by default in future.
remove_unused_original_minimum_age_seconds=86400	(IntOpt) Unused unresized base images younger than this will not be removed
remove_unused_resized_minimum_age_seconds=3600	(IntOpt) Unused resized base images younger than this will not be removed
rescue_image_id=None	(StrOpt) Rescue ami image
rescue_kernel_id=None	(StrOpt) Rescue aki image
rescue_ramdisk_id=None	(StrOpt) Rescue ari image
rescue_timeout=0	(IntOpt) Automatically unrescue an instance after N seconds. Set to 0 to disable.
snapshot_image_format=None	(StrOpt) Snapshot image format (valid options are : raw, qcow2, vmdk, vdi). Defaults to same as source image
timeout_nbd=10	(IntOpt) time to wait for a NBD device coming up
use_cow_images=True	(BoolOpt) Whether to use cow images
use_usb_tablet=True	(BoolOpt) Sync virtual and real mouse cursors in Windows VMs
vcpu_pin_set=None	(StrOpt) Which pcpus can be used by vcpus of instance e.g: "4-12,^8,15"
virt_mkfs=['default=mkfs.ext3 -L %(fs_label)s -F %(target)s', 'linux=mkfs.ext3 -L %(fs_label)s -F %(target)s', 'windows=mkfs.ntfs –force –fast –label %(fs_label)s %(target)s']	(MultiStrOpt) mkfs commands for ephemeral device. The format is <os_type>=<mkfs command>

KVM

KVM is configured as the default hypervisor for Compute.

Note

This document contains several sections about hypervisor selection. If you are reading this document linearly, you do not want to load the KVM module before you install `nova-compute`. The `nova-compute` service depends on qemu-kvm, which installs `/lib/udev/rules.d/45-qemu-kvm.rules`, which sets the correct permissions on the /dev/kvm device node.

To enable KVM explicitly, add the following configuration options to the `/etc/nova/nova.conf` file:

```
compute_driver=libvirt.LibvirtDriver
libvirt_type=kvm
```

The KVM hypervisor supports the following virtual machine image formats:

- Raw
- QEMU Copy-on-write (qcow2)
- QED Qemu Enhanced Disk
- VMWare virtual machine disk format (vmdk)

This section describes how to enable KVM on your system. For more information, see the following distribution-specific documentation:

- Fedora: Getting started with virtualization from the Fedora project wiki.
- Ubuntu: KVM/Installation from the Community Ubuntu documentation.
- Debian: Virtualization with KVM from the Debian handbook.
- RHEL: Installing virtualization packages on an existing Red Hat Enterprise Linux system from the *Red Hat Enterprise Linux Virtualization Host Configuration and Guest Installation Guide*.
- openSUSE: Installing KVM from the openSUSE Virtualization with KVM manual.
- SLES: Installing KVM from the SUSE Linux Enterprise Server Virtualization with KVM manual.

Enable KVM

To perform these steps, you must be logged in as the `root` user.

1. To determine whether the `svm` or `vmx` CPU extensions are present, run this command:

   ```
   # grep -E 'svm|vmx' /proc/cpuinfo
   ```

   This command generates output if the CPU is hardware-virtualization capable. Even if output is shown, you might still need to enable virtualization in the system BIOS for full support.

   If no output appears, consult your system documentation to ensure that your CPU and motherboard support hardware virtualization. Verify that any relevant hardware virtualization options are enabled in the system BIOS.

The BIOS for each manufacturer is different. If you must enable virtualization in the BIOS, look for an option containing the words `virtualization`, VT, `VMX`, or `SVM`.

2. To list the loaded kernel modules and verify that the `kvm` modules are loaded, run this command:

```
# lsmod | grep kvm
```

If the output includes `kvm_intel` or `kvm_amd`, the `kvm` hardware virtualization modules are loaded and your kernel meets the module requirements for OpenStack Compute.

If the output does not show that the `kvm` module is loaded, run this command to load it:

```
# modprobe -a kvm
```

Run the command for your CPU. For Intel, run this command:

```
# modprobe -a kvm-intel
```

For AMD, run this command:

```
# modprobe -a kvm-amd
```

Because a KVM installation can change user group membership, you might need to log in again for changes to take effect.

If the kernel modules do not load automatically, use the procedures listed in these subsections.

If the checks indicate that required hardware virtualization support or kernel modules are disabled or unavailable, you must either enable this support on the system or find a system with this support.

Note

Some systems require that you enable VT support in the system BIOS. If you believe your processor supports hardware acceleration but the previous command did not produce output, reboot your machine, enter the system BIOS, and enable the VT option.

If KVM acceleration is not supported, configure Compute to use a different hypervisor, such as QEMU or Xen.

These procedures help you load the kernel modules for Intel-based and AMD-based processors if they do not load automatically during KVM installation.

Intel-based processors

If your compute host is Intel-based, run these commands as root to load the kernel modules:

```
# modprobe kvm
# modprobe kvm-intel
```

Add these lines to the `/etc/modules` file so that these modules load on reboot:

```
kvm
kvm-intel
```

AMD-based processors

If your compute host is AMD-based, run these commands as root to load the kernel modules:

```
# modprobe kvm
# modprobe kvm-amd
```

Add these lines to `/etc/modules` file so that these modules load on reboot:

```
kvm
kvm-amd
```

Specify the CPU model of KVM guests

The Compute service enables you to control the guest CPU model that is exposed to KVM virtual machines. Use cases include:

- To maximize performance of virtual machines by exposing new host CPU features to the guest
- To ensure a consistent default CPU across all machines, removing reliance of variable QEMU defaults

In libvirt, the CPU is specified by providing a base CPU model name (which is a shorthand for a set of feature flags), a set of additional feature flags, and the topology (sockets/cores/threads). The libvirt KVM driver provides a number of standard CPU model names. These models are defined in the `/usr/share/libvirt/cpu_map.xml` file. Check this file to determine which models are supported by your local installation.

Two Compute configuration options define which type of CPU model is exposed to the hypervisor when using KVM: `libvirt_cpu_mode` and `libvirt_cpu_model`.

The `libvirt_cpu_mode` option can take one of the following values: `none`, `host-passthrough`, `host-model`, and `custom`.

Host model (default for KVM & QEMU)

If your `nova.conf` file contains `libvirt_cpu_mode=host-model`, libvirt identifies the CPU model in `/usr/share/libvirt/cpu_map.xml` file that most closely matches the host, and requests additional CPU flags to complete the match. This configuration provides the maximum functionality and performance and maintains good reliability and compatibility if the guest is migrated to another host with slightly different host CPUs.

Host pass through

If your `nova.conf` file contains `libvirt_cpu_mode=host-passthrough`, libvirt tells KVM to pass through the host CPU with no modifications. The difference to host-model, instead of just matching feature flags, every last detail of the host CPU is matched. This gives absolutely best performance, and can be important to some apps which check low level CPU details, but it comes at a cost with respect to migration: the guest can only be migrated to an exactly matching host CPU.

Custom

If your `nova.conf` file contains `libvirt_cpu_mode=custom`, you can explicitly specify one of the supported named model using the libvirt_cpu_model configuration option. For example, to configure the KVM guests to expose Nehalem CPUs, your `nova.conf` file should contain:

```
libvirt_cpu_mode=custom
libvirt_cpu_model=Nehalem
```

None (default for all libvirt-driven hypervisors other than KVM & QEMU)

If your `nova.conf` file contains `libvirt_cpu_mode=none`, libvirt does not specify a CPU model. Instead, the hypervisor chooses the default model. This setting is equivalent to the Compute service behavior prior to the Folsom release.

Guest agent support

With the Havana release, support for guest agents was added, allowing optional access between compute nods and guests through a socket, using the qmp protocol.

To enable this feature, you must set `hw_qemu_guest_agent=yes` as a metadata parameter on the image you wish to use to create guest-agent-capable instances from. You can explicitly disable the feature by setting `hw_qemu_guest_agent=no` in the image metadata.

KVM performance tweaks

The VHostNet kernel module improves network performance. To load the kernel module, run the following command as root:

```
# modprobe vhost_net
```

Troubleshoot KVM

Trying to launch a new virtual machine instance fails with the ERR0Rstate, and the following error appears in the `/var/log/nova/nova-compute.log` file:

```
libvirtError: internal error no supported architecture for os type 'hvm'
```

This message indicates that the KVM kernel modules were not loaded.

If you cannot start VMs after installation without rebooting, the permissions might not be correct. This can happen if you load the KVM module before you install `nova-compute`. To check whether the group is set to kvm, run:

```
# ls -l /dev/kvm
```

If it is not set to kvm, run:

```
# sudo udevadm trigger
```

QEMU

From the perspective of the Compute service, the QEMU hypervisor is very similar to the KVM hypervisor. Both are controlled through libvirt, both support the same feature

set, and all virtual machine images that are compatible with KVM are also compatible with QEMU. The main difference is that QEMU does not support native virtualization. Consequently, QEMU has worse performance than KVM and is a poor choice for a production deployment.

The typical uses cases for QEMU are

- Running on older hardware that lacks virtualization support.
- Running the Compute service inside of a virtual machine for development or testing purposes, where the hypervisor does not support native virtualization for guests.

To enable QEMU, add these settings to `nova.conf`:

```
compute_driver=libvirt.LibvirtDriver
libvirt_type=qemu
```

For some operations you may also have to install the **guestmount** utility:

On Ubuntu:

```
$> sudo apt-get install guestmount
```

On RHEL, Fedora or CentOS:

```
$> sudo yum install libguestfs-tools
```

On openSUSE:

```
$> sudo zypper install guestfs-tools
```

The QEMU hypervisor supports the following virtual machine image formats:

- Raw
- QEMU Copy-on-write (qcow2)
- VMWare virtual machine disk format (vmdk)

Tips and fixes for QEMU on RHEL

If you are testing OpenStack in a virtual machine, you need to configure nova to use qemu without KVM and hardware virtualization. The second command relaxes SELinux rules to allow this mode of operation (https://bugzilla.redhat.com/show_bug.cgi?id=753589). The last two commands here work around a libvirt issue fixed in RHEL 6.4. Note nested virtualization will be the much slower TCG variety, and you should provide lots of memory to the top level guest, as the OpenStack-created guests default to 2GM RAM with no overcommit.

Note

The second command, **setsebool**, may take a while.

```
$> sudo openstack-config --set /etc/nova/nova.conf DEFAULT libvirt_type qemu
       $> sudo setsebool -P virt_use_execmem on
       $> sudo ln -s /usr/libexec/qemu-kvm /usr/bin/qemu-system-x86_64
```

```
$> sudo service libvirtd restart
```

Xen, XenAPI, XenServer, and XCP

This section describes Xen, XenAPI, XenServer, and XCP, their differences, and how to use them with OpenStack. After you understand how the Xen and KVM architectures differ, you can determine when to use each architecture in your OpenStack cloud.

Xen terminology

Xen. A hypervisor that provides the fundamental isolation between virtual machines. Xen is open source (GPLv2) and is managed by Xen.org, an cross-industry organization.

Xen is a component of many different products and projects. The hypervisor itself is very similar across all these projects, but the way that it is managed can be different, which can cause confusion if you're not clear which tool stack you are using. Make sure you know what tool stack you want before you get started.

Xen Cloud Platform (XCP). An open source (GPLv2) tool stack for Xen. It is designed specifically as a platform for enterprise and cloud computing, and is well integrated with OpenStack. XCP is available both as a binary distribution, installed from an iso, and from Linux distributions, such as xcp-xapi in Ubuntu. The current versions of XCP available in Linux distributions do not yet include all the features available in the binary distribution of XCP.

Citrix XenServer. A commercial product. It is based on XCP, and exposes the same tool stack and management API. As an analogy, think of XenServer being based on XCP in the way that Red Hat Enterprise Linux is based on Fedora. XenServer has a free version (which is very similar to XCP) and paid-for versions with additional features enabled. Citrix provides support for XenServer, but as of July 2012, they do not provide any support for XCP. For a comparison between these products see the XCP Feature Matrix.

Both XenServer and XCP include Xen, Linux, and the primary control daemon known as **xapi**.

The API shared between XCP and XenServer is called **XenAPI**. OpenStack usually refers to XenAPI, to indicate that the integration works equally well on XCP and XenServer. Sometimes, a careless person will refer to XenServer specifically, but you can be reasonably confident that anything that works on XenServer will also work on the latest version of XCP. Read the XenAPI Object Model Overview for definitions of XenAPI specific terms such as SR, VDI, VIF and PIF.

Privileged and unprivileged domains

A Xen host runs a number of virtual machines, VMs, or domains (the terms are synonymous on Xen). One of these is in charge of running the rest of the system, and is known as "domain 0," or "dom0." It is the first domain to boot after Xen, and owns the storage and networking hardware, the device drivers, and the primary control software. Any other VM is unprivileged, and are known as a "domU" or "guest". All customer VMs are unprivileged of course, but you should note that on Xen the OpenStack control software (`nova-compute`) also runs in a domU. This gives a level of security isolation between the privileged system software and the OpenStack software (much of which is customer-facing). This architecture is described in more detail later.

There is an ongoing project to split domain 0 into multiple privileged domains known as **driver domains** and **stub domains**. This would give even better separation between critical components. This technology is what powers Citrix XenClient RT, and is likely to be added into XCP in the next few years. However, the current architecture just has three levels of separation: dom0, the OpenStack domU, and the completely unprivileged customer VMs.

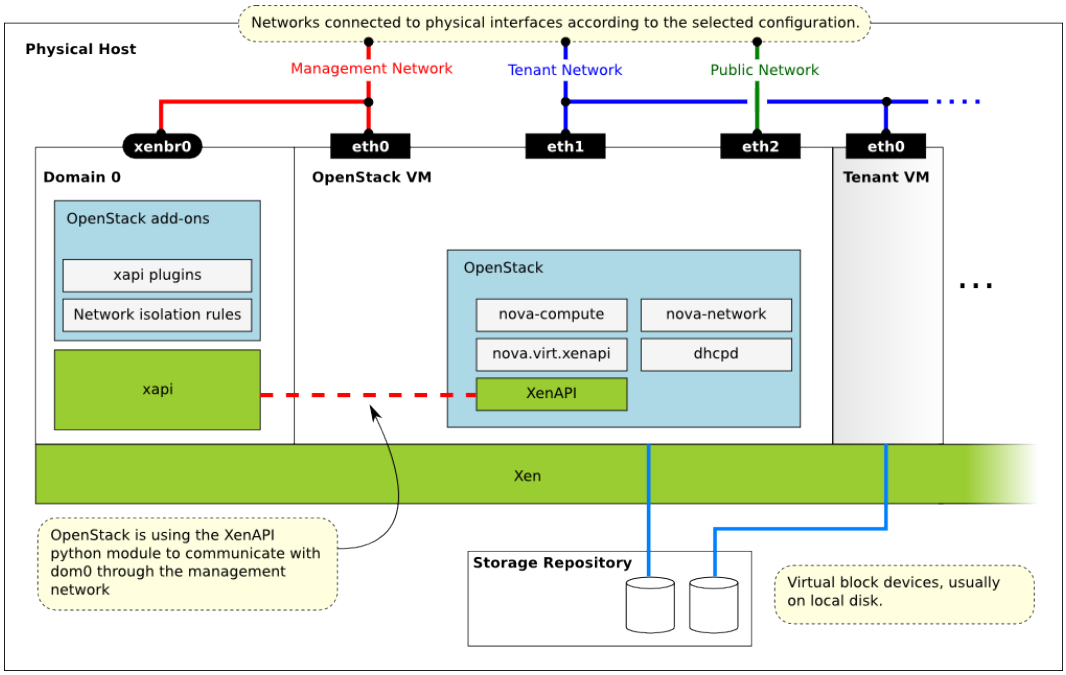
Paravirtualized versus hardware virtualized domains

A Xen virtual machine can be **paravirtualized (PV)** or **hardware virtualized (HVM)**. This refers to the interaction between Xen, domain 0, and the guest VM's kernel. PV guests are aware of the fact that they are virtualized and will co-operate with Xen and domain 0; this gives them better performance characteristics. HVM guests are not aware of their environment, and the hardware has to pretend that they are running on an unvirtualized machine. HVM guests have the advantage that there is no need to modify the guest operating system, which is essential when running Windows.

In OpenStack, customer VMs may run in either PV or HVM mode. However, the OpenStack domU (that's the one running `nova-compute`) **must** be running in PV mode.

XenAPI Deployment Architecture

When you deploy OpenStack on XCP or XenServer, you get something similar to this:



Key things to note:

- The hypervisor: Xen
- Domain 0: runs xapi and some small pieces from OpenStack (some xapi plug-ins and network isolation rules). The majority of this is provided by XenServer or XCP (or yourself using Kronos).
- OpenStack VM: The `nova-compute` code runs in a paravirtualized virtual machine, running on the host under management. Each host runs a local instance of `nova-`

`compute`. It will often also be running nova-network (depending on your network mode). In this case, nova-network is managing the addresses given to the tenant VMs through DHCP.

- Nova uses the XenAPI Python library to talk to xapi, and it uses the Management Network to reach from the domU to dom0 without leaving the host.

Some notes on the networking:

- The above diagram assumes FlatDHCP networking (the DevStack default).
- There are three main OpenStack Networks:
  - Management network - RabbitMQ, MySQL, etc. Please note that the VM images are downloaded by the XenAPI plug-ins, so make sure that the images can be downloaded through the management network. It usually means binding those services to the management interface.
  - Tenant network - controlled by nova-network. The parameters of this network depend on the networking model selected (Flat, Flat DHCP, VLAN).
  - Public network - floating IPs, public API endpoints.
- The networks shown here need to be connected to the corresponding physical networks within the data center. In the simplest case, three individual physical network cards could be used. It is also possible to use VLANs to separate these networks. Please note, that the selected configuration must be in line with the networking model selected for the cloud. (In case of VLAN networking, the physical channels have to be able to forward the tagged traffic.)

XenAPI pools

The host-aggregates feature enables you to create pools of XenServer hosts to enable live migration when using shared storage. However, you cannot configure shared storage.

Further reading

Here are some of the resources available to learn more about Xen:

- Citrix XenServer official documentation: http://docs.vmd.citrix.com/XenServer.
- What is Xen? by Xen.org: http://xen.org/files/Marketing/WhatisXen.pdf.
- Xen Hypervisor project: http://xen.org/products/xenhyp.html.
- XCP project: http://xen.org/products/cloudxen.html.
- Further XenServer and OpenStack information: http://wiki.openstack.org/XenServer.

Install XenServer and XCP

Before you can run OpenStack with XCP or XenServer, you must install the software on an appropriate server.

Note

Xen is a type 1 hypervisor: When your server starts, Xen is the first software that runs. Consequently, you must install XenServer or XCP before you install the operating system where you want to run OpenStack code. The OpenStack services then run in a virtual machine that you install on top of XenServer.

Before you can install your system, decide whether to install a free or paid edition of Citrix XenServer or Xen Cloud Platform from Xen.org. Download the software from these locations:

- http://www.citrix.com/XenServer/download
- http://www.xen.org/download/xcp/index.html

When you install many servers, you might find it easier to perform PXE boot installations of XenServer or XCP. You can also package any post-installation changes that you want to make to your XenServer by creating your own XenServer supplemental pack.

You can also install the xcp-xenapi package on Debian-based distributions to get XCP. However, this is not as mature or feature complete as above distributions. This modifies your boot loader to first boot Xen and boot your existing OS on top of Xen as Dom0. The xapi daemon runs in Dom0. Find more details at http://wiki.xen.org/wiki/Project_Kronos.

Important

Make sure you use the EXT type of storage repository (SR). Features that require access to VHD files (such as copy on write, snapshot and migration) do not work when you use the LVM SR. Storage repository (SR) is a XenAPI-specific term relating to the physical storage where virtual disks are stored.

On the XenServer/XCP installation screen, choose the **XenDesktop Optimized** option. If you use an answer file, make sure you use `srtype="ext"` in the `installation` tag of the answer file.

Post-installation steps

Complete these steps to install OpenStack in your XenServer system:

1. For resize and migrate functionality, complete the changes described in the *Configure resize* section in the *OpenStack Configuration Reference*.
2. Install the VIF isolation rules to help prevent mac and IP address spoofing.
3. Install the XenAPI plug-ins. See the following section.
4. To support AMI type images, you must set up `/boot/guest` symlink/directory in Dom0. For detailed instructions, see next section.
5. To support resize/migration, set up an ssh trust relation between your XenServer hosts, and ensure `/images` is properly set up. See next section for more details.
6. Create a Paravirtualized virtual machine that can run the OpenStack compute code.
7. Install and configure the `nova-compute` in the above virtual machine.

For more information, see how DevStack performs the last three steps for developer deployments. For more information about DevStack, see *Getting Started With XenServer and Devstack* (https://github.com/openstack-dev/devstack/blob/master/tools/xen/README.md). Find more information about the first step, see *Multi Tenancy Networking Protections in XenServer* (https://github.com/openstack/nova/blob/master/plugins/xenserver/doc/networking.rst). For information about how to install the XenAPI plug-ins, see *XenAPI README* (https://github.com/openstack/nova/blob/master/plugins/xenserver/xenapi/README).

Install the XenAPI plug-ins

When you use Xen as the hypervisor for OpenStack Compute, you can install a Python script (or any executable) on the host side, and call that through the XenAPI. These scripts are called plug-ins. The XenAPI plug-ins live in the nova code repository. These plug-ins have to be copied to the Dom0 for the hypervisor, to the appropriate directory, where xapi can find them. There are several options for the installation. The important thing is to ensure that the version of the plug-ins are in line with the nova installation by only installing plug-ins from a matching nova repository.

Manually install the plug-in

1. Create temporary files/directories:

```
$ NOVA_ZIPBALL=$(mktemp)
$ NOVA_SOURCES=$(mktemp -d)
```

2. Get the source from github. The example assumes the master branch is used. Amend the URL to match the version being used:

```
$ wget -qO "$NOVA_ZIPBALL" https://github.com/openstack/nova/archive/
master.zip
$ unzip "$NOVA_ZIPBALL" -d "$NOVA_SOURCES"
```

   (Alternatively) To use the official Ubuntu packages, use the following commands to get the nova code base:

```
$ ( cd $NOVA_SOURCES && apt-get source python-nova --download-only )
$ ( cd $NOVA_SOURCES && for ARCHIVE in *.tar.gz; do tar -xzf $ARCHIVE;
 done )
```

3. Copy the plug-ins to the hypervisor:

```
$ PLUGINPATH=$(find $NOVA_SOURCES -path '*/xapi.d/plugins' -type d -print)
$ tar -czf - -C "$PLUGINPATH" ./ | ssh root@xenserver tar -xozf - -C /etc/
xapi.d/plugins/
```

4. Remove the temporary files/directories:

```
$ rm "$NOVA_ZIPBALL"
$ rm -rf "$NOVA_SOURCES"
```

Package a XenServer supplemental pack

Follow these steps to produce a supplemental pack from the nova sources, and package it as a XenServer supplemental pack.

1. Create RPM packages. Given you have the nova sources. Use one of the methods in the section called “Manually install the plug-in” [83]:

```
$ cd nova/plugins/xenserver/xenapi/contrib
$ ./build-rpm.sh
```

These commands leave an `.rpm` file in the `rpmbuild/RPMS/noarch/` directory.

2. Pack the RPM packages to a Supplemental Pack, using the XenServer DDK (the following command should be issued on the XenServer DDK virtual appliance, after the produced rpm file has been copied over):

```
$ /usr/bin/build-supplemental-pack.sh \
> --output=output_directory \
> --vendor-code=novaplugin \
> --vendor-name=openstack \
> --label=novaplugins \
> --text="nova plugins" \
> --version=0 \
> full_path_to_rpmfile
```

This command produces an `.iso` file in the output directory specified. Copy that file to the hypervisor.

3. Install the Supplemental Pack. Log in to the hypervisor, and issue:

```
# xe-install-supplemental-pack path_to_isofile
```

Prepare for AMI type images

To support AMI type images in your OpenStack installation, you must create a `/boot/guest` directory inside Dom0. The OpenStack VM extracts the kernel and ramdisk from the AKI and ARI images puts them in this location.

OpenStack maintains the contents of this directory and its size should not increase during normal operation. However, in case of power failures or accidental shutdowns, some files might be left over. To prevent these files from filling the Dom0 disk, set up this directory as a symlink that points to a subdirectory of the local SR.

Run these commands in Dom0 to achieve this setup:

```
# LOCAL_SR=$(xe sr-list name-label="Local storage" --minimal)
# LOCALPATH="/var/run/sr-mount/$LOCAL_SR/os-guest-kernels"
# mkdir -p "$LOCALPATH"
# ln -s "$LOCALPATH" /boot/guest
```

Modify Dom0 for resize/migration support

To resize servers with XenServer and XCP, you must:

- Establish a root trust between all hypervisor nodes of your deployment:

  To do so, generate an ssh key-pair with the **ssh-keygen** command. Ensure that each of your dom0's `authorized_keys` file (located in `/root/.ssh/authorized_keys`) contains the public key fingerprint (located in `/root/.ssh/id_rsa.pub`).

- Provide an `/images` mount point to the dom0 for your hypervisor:

  Dom0 space is at a premium so creating a directory in dom0 is potentially dangerous and likely to fail especially when you resize large servers. The least you can do is to symlink `/`

`images` to your local storage SR. The following instructions work for an English-based installation of XenServer (and XCP) and in the case of ext3-based SR (with which the resize functionality is known to work correctly).

```
# LOCAL_SR=$(xe sr-list name-label="Local storage" --minimal)
# IMG_DIR="/var/run/sr-mount/$LOCAL_SR/images"
# mkdir -p "$IMG_DIR"
# ln -s "$IMG_DIR" /images
```

Xen boot from ISO

XenServer, through the XenAPI integration with OpenStack, provides a feature to boot instances from an ISO file. To activate the Boot From ISO feature, you must configure the SR elements on XenServer host, as follows:

1. Create an ISO-typed SR, such as an NFS ISO library, for instance. For this, using XenCenter is a simple method. You must export an NFS volume from a remote NFS server. Make sure it is exported in read-write mode.

2. On the compute host, find and record the uuid of this ISO SR:

   ```
   # xe host-list
   ```

3. Locate the uuid of the NFS ISO library:

   ```
   # xe sr-list content-type=iso
   ```

4. Set the uuid and configuration. Even if an NFS mount point is not local, you must specify `local-storage-iso`.

   ```
   # xe sr-param-set uuid=[iso sr uuid] other-config:i18n-key=local-storage-iso
   ```

5. Make sure the host-uuid from `xe pbd-list` equals the uuid of the host you found previously:

   ```
   # xe sr-uuid=[iso sr uuid]
   ```

6. You can now add images through the OpenStack Image Service with `disk-format=iso`, and boot them in OpenStack Compute:

   ```
   # glance image-create --name=fedora_iso --disk-format=iso --container-format=bare < Fedora-16-x86_64-netinst.iso
   ```

Xen configuration reference

The following section discusses some commonly changed options in XenServer. The table below provides a complete reference of all configuration options available for configuring Xen with OpenStack.

The recommended way to use Xen with OpenStack is through the XenAPI driver. To enable the XenAPI driver, add the following configuration options `/etc/nova/nova.conf` and restart the `nova-compute` service:

```
compute_driver=xenapi.XenAPIDriver
xenapi_connection_url=http://your_xenapi_management_ip_address
xenapi_connection_username=root
xenapi_connection_password=your_password
```

These connection details are used by the OpenStack Compute service to contact your hypervisor and are the same details you use to connect XenCenter, the XenServer management console, to your XenServer or XCP box.

Note

The `xenapi_connection_url` is generally the management network IP address of the XenServer. Though it is possible to use the internal network IP Address (169.250.0.1) to contact XenAPI, this does not allow live migration between hosts, and other functionalities like host aggregates do not work.

It is possible to manage Xen using libvirt, though this is not well-tested or supported. To experiment using Xen through libvirt add the following configuration options `/etc/nova/nova.conf`:

```
compute_driver=libvirt.LibvirtDriver
libvirt_type=xen
```

Agent

If you don't have the guest agent on your VMs, it takes a long time for nova to decide the VM has successfully started. Generally a large timeout is required for Windows instances, bug you may want to tweak `agent_version_timeout`

Firewall

If using nova-network, IPTables is supported:

```
firewall_driver=nova.virt.firewall.IptablesFirewallDriver
```

Alternately, doing the isolation in Dom0:

```
firewall_driver=nova.virt.xenapi.firewall.Dom0IptablesFirewallDriver
```

VNC proxy address

Assuming you are talking to XenAPI through the host local management network, and XenServer is on the address: 169.254.0.1, you can use the following: `vncserver_proxyclient_address=169.254.0.1`

Storage

You can specify which Storage Repository to use with nova by looking at the following flag. The default is to use the local-storage setup by the default installer:

```
sr_matching_filter="other-config:i18n-key=local-storage"
```

Another good alternative is to use the "default" storage (for example if you have attached NFS or any other shared storage):

```
sr_matching_filter="default-sr:true"
```

Note

To use a XenServer pool, you must create the pool by using the Host Aggregates feature.

Xen configuration reference

Table 2.22. Description of configuration options for xen

Configuration option=Default value	Description
agent_resetnetwork_timeout=60	(IntOpt) number of seconds to wait for agent reply to resetnetwork request
agent_timeout=30	(IntOpt) number of seconds to wait for agent reply
agent_version_timeout=300	(IntOpt) number of seconds to wait for agent to be fully operational
cache_images=all	(StrOpt) Cache glance images locally. `all` will cache all images, `some` will only cache images that have the image_property `cache_in_nova=True`, and `none` turns off caching entirely
console_driver=nova.console.xvp.XVPConsoleProxy	(StrOpt) Driver to use for the console proxy
console_vmrc_error_retries=10	(IntOpt) number of retries for retrieving VMRC information
console_vmrc_port=443	(IntOpt) port for VMware VMRC connections
console_xvp_conf=/etc/xvp.conf	(StrOpt) generated XVP conf file
console_xvp_conf_template=$pybasedir/nova/console/xvp.conf.template	(StrOpt) XVP conf template
console_xvp_log=/var/log/xvp.log	(StrOpt) XVP log file
console_xvp_multiplex_port=5900	(IntOpt) port for XVP to multiplex VNC connections on
console_xvp_pid=/var/run/xvp.pid	(StrOpt) XVP master process pid file
default_os_type=linux	(StrOpt) Default OS type
iqn_prefix=iqn.2010-10.org.openstack	(StrOpt) IQN Prefix
max_kernel_ramdisk_size=16777216	(IntOpt) Maximum size in bytes of kernel or ramdisk images
sr_matching_filter=default-sr:true	(StrOpt) Filter for finding the SR to be used to install guest instances on. To use the Local Storage in default XenServer/XCP installations set this flag to other-config:i18n-key=local-storage. To select an SR with a different matching criteria, you could set it to other-config:my_favorite_sr=true. On the other hand, to fall back on the Default SR, as displayed by XenCenter, set this flag to: default-sr:true
stub_compute=False	(BoolOpt) Stub calls to compute worker for tests
target_host=None	(StrOpt) iSCSI Target Host
target_port=3260	(StrOpt) iSCSI Target Port, 3260 Default
use_join_force=True	(BoolOpt) To use for hosts with different CPUs
xen_hvmloader_path=/usr/lib/xen/boot/hvmloader	(StrOpt) Location where the Xen hvmloader is kept
xenapi_agent_path=usr/sbin/xe-update-networking	(StrOpt) Specifies the path in which the xenapi guest agent should be located. If the agent is present, network configuration is not injected into the image. Used if compute_driver=xenapi.XenAPIDriver and flat_injected=True
xenapi_check_host=True	(BoolOpt) Ensure compute service is running on host XenAPI connects to.
xenapi_connection_concurrent=5	(IntOpt) Maximum number of concurrent XenAPI connections. Used only if compute_driver=xenapi.XenAPIDriver
xenapi_connection_password=None	(StrOpt) Password for connection to XenServer/Xen Cloud Platform. Used only if compute_driver=xenapi.XenAPIDriver

Configuration option=Default value	Description
xenapi_connection_url=None	(StrOpt) URL for connection to XenServer/Xen Cloud Platform. A special value of unix://local can be used to connect to the local unix socket. Required if compute_driver=xenapi.XenAPIDriver
xenapi_connection_username=root	(StrOpt) Username for connection to XenServer/Xen Cloud Platform. Used only if compute_driver=xenapi.XenAPIDriver
xenapi_disable_agent=False	(BoolOpt) Disables the use of the XenAPI agent in any image regardless of what image properties are present.
xenapi_image_compression_level=None	(IntOpt) Compression level for images, e.g., 9 for gzip -9. Range is 1-9, 9 being most compressed but most CPU intensive on dom0.
xenapi_image_upload_handler=nova.virt.xenapi.image.glance.GlanceStore	(StrOpt) Dom0 plugin driver used to handle image uploads.
xenapi_ipxe_boot_menu_url=None	(StrOpt) URL to the iPXE boot menu
xenapi_login_timeout=10	(IntOpt) Timeout in seconds for XenAPI login.
xenapi_ipxe_mkisofs_cmd=mkisofs	(StrOpt) Name and optionally path of the tool used for ISO image creation
xenapi_num_vbd_unplug_retries=10	(IntOpt) Maximum number of retries to unplug VBD
xenapi_ipxe_network_name=None	(StrOpt) Name of network to use for booting iPXE ISOs
xenapi_ovs_integration_bridge=xapi1	(StrOpt) Name of Integration Bridge used by Open vSwitch
xenapi_remap_vbd_dev=False	(BoolOpt) Used to enable the remapping of VBD dev (Works around an issue in Ubuntu Maverick)
xenapi_remap_vbd_dev_prefix=sd	(StrOpt) Specify prefix to remap VBD dev to (ex. /dev/xvdb -> /dev/sdb)
xenapi_running_timeout=60	(IntOpt) number of seconds to wait for instance to go to running state
xenapi_sparse_copy=True	(BoolOpt) Whether to use sparse_copy for copying data on a resize down (False will use standard dd). This speeds up resizes down considerably since large runs of zeros won't have to be rsynced
xenapi_sr_base_path=/var/run/sr-mount	(StrOpt) Base path to the storage repository
xenapi_torrent_base_url=None	(StrOpt) Base URL for torrent files.
xenapi_torrent_download_stall_cutoff=600	(IntOpt) Number of seconds a download can remain at the same progress percentage w/o being considered a stall
xenapi_torrent_images=none	(StrOpt) Whether or not to download images via Bit Torrent (all|some|none).
xenapi_torrent_listen_port_end=6891	(IntOpt) End of port range to listen on
xenapi_torrent_listen_port_start=6881	(IntOpt) Beginning of port range to listen on
xenapi_torrent_max_last_accessed=86400	(IntOpt) Cached torrent files not accessed within this number of seconds can be reaped
xenapi_torrent_max_seeder_processes_per_host=1	(IntOpt) Maximum number of seeder processes to run concurrently within a given dom0. (-1 = no limit)
xenapi_torrent_seed_chance=1.0	(FloatOpt) Probability that peer will become a seeder. (1.0 = 100%)
xenapi_torrent_seed_duration=3600	(IntOpt) Number of seconds after downloading an image via BitTorrent that it should be seeded for other peers.
xenapi_use_agent_default=False	(BoolOpt) Determines if the xenapi agent should be used when the image used does not contain a hint to declare if the agent is present or not. The hint is a glance property "xenapi_use_agent" that has the value "true" or "false".

Configuration option=Default value	Description
	Note that waiting for the agent when it is not present will significantly increase server boot times.
xenapi_vhd_coalesce_max_attempts=5	(IntOpt) Max number of times to poll for VHD to coalesce. Used only if compute_driver=xenapi.XenAPIDriver
xenapi_vhd_coalesce_poll_interval=5.0	(FloatOpt) The interval used for polling of coalescing vhds. Used only if compute_driver=xenapi.XenAPIDriver
xenapi_vif_driver=nova.virt.xenapi.vif.XenAPIBridgeDriver	(StrOpt) The XenAPI VIF driver using XenServer Network APIs.

LXC (Linux containers)

LXC (also known as Linux containers) is a virtualization technology that works at the operating system level. This is different from hardware virtualization, the approach used by other hypervisors such as KVM, Xen, and VMWare. LXC (as currently implemented using libvirt in the nova project) is not a secure virtualization technology for multi-tenant environments (specifically, containers may affect resource quotas for other containers hosted on the same machine). Additional containment technologies, such as AppArmor, may be used to provide better isolation between containers, although this is not the case by default. For all these reasons, the choice of this virtualization technology is not recommended in production.

If your compute hosts do not have hardware support for virtualization, LXC will likely provide better performance than QEMU. In addition, if your guests need to access to specialized hardware (e.g., GPUs), this may be easier to achieve with LXC than other hypervisors.

Note

Some OpenStack Compute features may be missing when running with LXC as the hypervisor. See the hypervisor support matrix for details.

To enable LXC, ensure the following options are set in `/etc/nova/nova.conf` on all hosts running the `nova-compute` service.

```
compute_driver=libvirt.LibvirtDriver
libvirt_type=lxc
```

On Ubuntu 12.04, enable LXC support in OpenStack by installing the `nova-compute-lxc` package.

VMware vSphere

Introduction

OpenStack Compute supports the VMware vSphere product family and enables access to advanced features such as vMotion, High Availability, and Dynamic Resource Scheduling (DRS). This section describes how to configure VMware-based virtual machine images for launch. vSphere versions 4.1 and newer are supported.

The VMware vCenter driver enables the `nova-compute` service to communicate with a VMware vCenter server that manages one or more ESX host clusters. The driver aggregates the ESX hosts in each cluster to present one large hypervisor entity for each cluster to

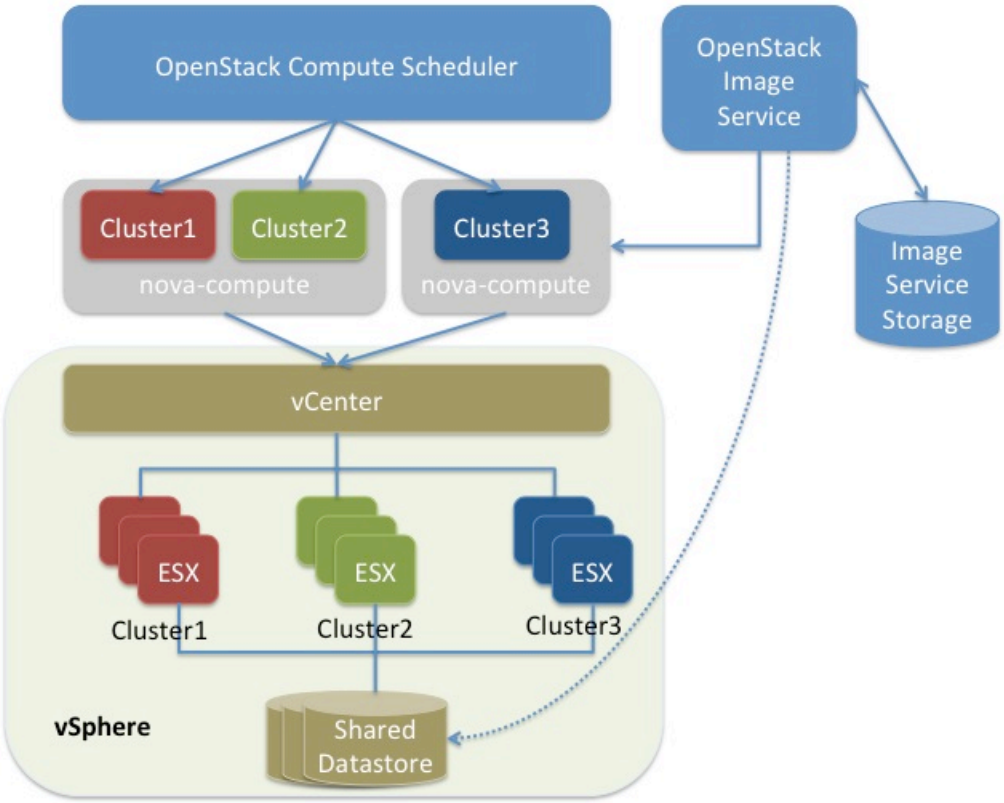
the Compute scheduler. Because individual ESX hosts are not exposed to the scheduler, Compute schedules to the granularity of clusters and vCenter uses DRS to select the actual ESX host within the cluster. When a virtual machine makes its way into a vCenter cluster, it can use all vSphere features.

The following sections describe how to configure the VMware vCenter driver.

High-level architecture

The following diagram shows a high-level view of the VMware driver architecture:

Figure 2.4. VMware driver architecture



As the figure shows, the OpenStack Compute Scheduler sees three hypervisors that each correspond to a cluster in vCenter. `Nova-compute` contains the VMware driver. You can run with multiple `nova-compute` services. While Compute schedules at the granularity of a cluster, the VMware driver inside `nova-compute` interacts with the vCenter APIs to select an appropriate ESX host within the cluster. Internally, vCenter uses DRS for placement.

The VMware vCenter driver also interacts with the OpenStack Image Service to copy VMDK images from the Image Service back end store. The dotted line in the figure represents VMDK images being copied from the OpenStack Image Service to the vSphere data store. VMDK images are cached in the data store so the copy operation is only required the first time that the VMDK image is used.

After OpenStack boots a VM into a vSphere cluster, the VM becomes visible in vCenter and can access vSphere advanced features. At the same time, the VM is visible in the OpenStack dashboard and you can manage it as you would any other OpenStack VM. You can perform advanced vSphere operations in vCenter while you configure OpenStack resources such as VMs through the OpenStack dashboard.

The figure does not show how networking fits into the architecture. Both `nova-network` and the OpenStack Networking Service are supported. For details, see the section called "Networking with VMware vSphere" [96].

Configuration overview

To get started with the VMware vCenter driver, complete the following high-level steps:

1. Configure vCenter correctly. See the section called "Prerequisites and limitations" [91].
2. Configure `nova.conf` for the VMware vCenter driver. See the section called "VMware vCenter driver" [92].
3. Load desired VMDK images into the OpenStack Image Service. See the section called "Images with VMware vSphere" [93].
4. Configure networking with either `nova-network` or the OpenStack Networking Service. See the section called "Networking with VMware vSphere" [96].

Prerequisites and limitations

Use the following list to prepare a vSphere environment that runs with the VMware vCenter driver:

1. **vCenter inventory**. Make sure that any vCenter used by OpenStack contains a single data center. This restriction applies to all releases prior to Havana stable update 2013.2.2, released February 13th 2014.
2. **DRS**. For any cluster that contains multiple ESX hosts, enable DRS and enable *fully automated* placement.
3. **Shared storage**. Only shared storage is supported and data stores must be shared among all hosts in a cluster. It is recommended to remove data stores not intended for OpenStack from clusters being configured for OpenStack.
4. **Clusters and data stores**. Do not use OpenStack clusters and data stores for other purposes. If you do, OpenStack displays incorrect usage information.
5. **Networking**. The networking configuration depends on the desired networking model. See the section called "Networking with VMware vSphere" [96].
6. **Security groups**. If you use the VMware driver with OpenStack Networking and the NSX plug-in, security groups are supported. If you use `nova-network`, security groups are not supported.

Note

The NSX plug-in is the only plug-in that is validated for vSphere.

7. **VNC**. The port range 5900 - 6105 (inclusive) is automatically enabled for VNC connections on every ESX host in all clusters under OpenStack control. For more information about using a VNC client to connect to virtual machine, see http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1246.

   Note

   In addition to the default VNC port numbers (5900 to 6000) specified in the above document, the following ports are also used: 6101, 6102, and 6105.

   You must modify the ESXi firewall configuration to allow the VNC ports. Additionally, for the firewall modifications to persist after a reboot, you must create a custom vSphere Installation Bundle (VIB) which is then installed onto the running ESXi host or added to a custom image profile used to install ESXi hosts. For details about how to create a VIB for persisting the firewall configuration modifications, see http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=2007381.

8. **Ephemeral Disks**. Ephemeral disks are not supported. A future stable release will address this temporary limitation.

VMware vCenter driver

Use the VMware vCenter driver (VMwareVCDriver) to connect OpenStack Compute with vCenter. This recommended configuration enables access through vCenter to advanced vSphere features like vMotion, High Availability, and Dynamic Resource Scheduling (DRS).

VMwareVCDriver configuration options

When you use the VMwareVCDriver (vCenter versions 5.1 and later) with OpenStack Compute, add the following VMware-specific configuration options to the `nova.conf` file:

```
[DEFAULT]
compute_driver=vmwareapi.VMwareVCDriver

[vmware]
host_ip=<vCenter host IP>
host_username=<vCenter username>
host_password=<vCenter password>
cluster_name=<vCenter cluster name>
datastore_regex=<optional datastore regex>
```

Note

- vSphere vCenter versions 5.0 and earlier: You must specify the location of the WSDL files by adding the `wsdl_location=http://127.0.0.1:8080/vmware/SDK/wsdl/vim25/vimService.wsdl` setting to the above configuration. For more information, see vSphere 5.0 and earlier additional set up.

  Clusters: The vCenter driver can support multiple clusters. To use more than one cluster, simply add multiple `cluster_name` lines in `nova.conf` with the appropriate cluster name. Clusters and data stores used by the vCenter driver should not contain any VMs other than those created by the driver.

- Data stores: The `datastore_regex` field specifies the data stores to use with Compute. For example, `datastore_regex="nas.*"` selects all the data stores that have a name starting with "nas". If this line is omitted, Compute uses the first data store returned by the vSphere API. It is recommended not to use this field and instead remove data stores that are not intended for OpenStack.

A `nova-compute` service can control one or more clusters containing multiple ESX hosts, making `nova-compute` a critical service from a high availability perspective. Because the host that runs `nova-compute` can fail while the vCenter and ESX still run, you must protect the `nova-compute` service against host failures.

Note

Many `nova.conf` options are relevant to libvirt but do not apply to this driver.

You must complete additional configuration for environments that use vSphere 5.0 and earlier. See the section called "vSphere 5.0 and earlier additional set up" [97].

Images with VMware vSphere

The vCenter driver supports images in the VMDK format. Disks in this format can be obtained from VMware Fusion or from an ESX environment. It is also possible to convert other formats, such as qcow2, to the VMDK format using the `qemu-img` utility. After a VMDK disk is available, load it into the OpenStack Image Service. Then, you can use it with the VMware vCenter driver. The following sections provide additional details on the supported disks and the commands used for conversion and upload.

Supported image types

Upload images to the OpenStack Image Service in VMDK format. The following VMDK disk types are supported:

- *VMFS Flat Disks* (includes thin, thick, zeroedthick, and eagerzeroedthick). Note that once a VMFS thin disk is exported from VMFS to a non-VMFS location, like the OpenStack Image Service, it becomes a preallocated flat disk. This impacts the transfer time from the OpenStack Image Service to the data store when the full preallocated flat disk, rather than the thin disk, must be transferred.

- *Monolithic Sparse disks*. Sparse disks get imported from the OpenStack Image Service into ESX as thin provisioned disks. Monolithic Sparse disks can be obtained from VMware Fusion or can be created by converting from other virtual disk formats using the `qemu-img` utility.

The following table shows the `vmware_disktype` property that applies to each of the supported VMDK disk types:

Table 2.23. OpenStack Image Service disk type settings

vmware_disktype property	VMDK disk type
sparse	Monolithic Sparse
thin	VMFS flat, thin provisioned
preallocated (default)	VMFS flat, thick/zeroedthick/eagerzeroedthick

The vmware_disktype property is set when an image is loaded into the OpenStack Image Service. For example, the following command creates a Monolithic Sparse image by setting vmware_disktype to sparse:

```
$ glance image-create name="ubuntu-sparse" disk_format=vmdk \
container_format=bare is_public=true \
--property vmware_disktype="sparse" \
--property vmware_ostype="ubuntu64Guest" < ubuntuLTS-sparse.vmdk
```

Note that specifying thin does not provide any advantage over preallocated with the current version of the driver. Future versions might restore the thin properties of the disk after it is downloaded to a vSphere data store.

Convert and load images

Using the qemu-img utility, disk images in several formats (such as, qcow2) can be converted to the VMDK format.

For example, the following command can be used to convert a qcow2 Ubuntu Precise cloud image:

```
$ qemu-img convert -f raw ~/Downloads/precise-server-cloudimg-amd64-disk1.img
 \
-O vmdk precise-server-cloudimg-amd64-disk1.vmdk
```

VMDK disks converted through qemu-img are *always* monolithic sparse VMDK disks with an IDE adapter type. Using the previous example of the Precise Ubuntu image after the qemu-img conversion, the command to upload the VMDK disk should be something like:

```
$ glance image-create --name precise-cloud --is-public=True \
--container-format=bare --disk-format=vmdk \
--property vmware_disktype="sparse" \
--property vmware_adaptertype="ide" < \
precise-server-cloudimg-amd64-disk1.vmdk
```

Note that the vmware_disktype is set to *sparse* and the vmware_adaptertype is set to *ide* in the previous command.

If the image did not come from the qemu-img utility, the vmware_disktype and vmware_adaptertype might be different. To determine the image adapter type from an image file, use the following command and look for the ddb.adapterType= line:

```
$ head -20 <vmdk file name>
```

Assuming a preallocated disk type and an iSCSI lsiLogic adapter type, the following command uploads the VMDK disk:

```
$ glance image-create name="ubuntu-thick-scsi" disk_format=vmdk \
container_format=bare is_public=true \
--property vmware_adaptertype="lsiLogic" \
--property vmware_disktype="preallocated" \
--property vmware_ostype="ubuntu64Guest" < ubuntuLTS-flat.vmdk
```

Currently, OS boot VMDK disks with an IDE adapter type cannot be attached to a virtual SCSI controller and likewise disks with one of the SCSI adapter types (such as, busLogic, lsiLogic) cannot be attached to the IDE controller. Therefore, as the previous examples

show, it is important to set the `vmware_adaptertype` property correctly. The default adapter type is lsiLogic, which is SCSI, so you can omit the *`vmware_adaptertype`* property if you are certain that the image adapter type is lsiLogic.

Tag VMware images

In a mixed hypervisor environment, OpenStack Compute uses the `hypervisor_type` tag to match images to the correct hypervisor type. For VMware images, set the hypervisor type to `vmware`. Other valid hypervisor types include: xen, qemu, kvm, lxc, uml, hyperv, and powervm.

```
$ glance image-create name="ubuntu-thick-scsi" disk_format=vmdk \
container_format=bare is_public=true \
--property vmware_adaptertype="lsiLogic" \
--property vmware_disktype="preallocated" \
--property hypervisor_type="vmware" \
--property vmware_ostype="ubuntu64Guest" < ubuntuLTS-flat.vmdk
```

Optimize images

Monolithic Sparse disks are considerably faster to download but have the overhead of an additional conversion step. When imported into ESX, sparse disks get converted to VMFS flat thin provisioned disks. The download and conversion steps only affect the first launched instance that uses the sparse disk image. The converted disk image is cached, so subsequent instances that use this disk image can simply use the cached version.

To avoid the conversion step (at the cost of longer download times) consider converting sparse disks to thin provisioned or preallocated disks before loading them into the OpenStack Image Service. Below are some tools that can be used to pre-convert sparse disks.

1. **Using vSphere CLI (or sometimes called the remote CLI or rCLI) tools**

   Assuming that the sparse disk is made available on a data store accessible by an ESX host, the following command converts it to preallocated format:

   ```
   vmkfstools --server=ip_of_some_ESX_host -i /vmfs/volumes/datastore1/sparse.
   vmdk /vmfs/volumes/datastore1/converted.vmdk
   ```

   (Note that the vifs tool from the same CLI package can be used to upload the disk to be converted. The vifs tool can also be used to download the converted disk if necessary.)

2. **Using vmkfstools directly on the ESX host**

   If the SSH service is enabled on an ESX host, the sparse disk can be uploaded to the ESX data store via scp and the vmkfstools local to the ESX host can use used to perform the conversion: (After logging in to the host via ssh)

   ```
   vmkfstools -i /vmfs/volumes/datastore1/sparse.vmdk /vmfs/volumes/datastore1/
   converted.vmdk
   ```

3. **vmware-vdiskmanager**

   `vmware-vdiskmanager` is a utility that comes bundled with VMware Fusion and VMware Workstation. Below is an example of converting a sparse disk to preallocated format:

```
'/Applications/VMware Fusion.app/Contents/Library/vmware-vdiskmanager' -r
 sparse.vmdk -t 4 converted.vmdk
```

In all of the above cases, the converted vmdk is actually a pair of files: the descriptor file *converted.vmdk* and the actual virtual disk data file *converted-flat.vmdk*. The file to be uploaded to the OpenStack Image Service is *converted-flat.vmdk*.

Image handling

The ESX hypervisor requires a copy of the VMDK file in order to boot up a virtual machine. As a result, the vCenter OpenStack Compute driver must download the VMDK via HTTP from the OpenStack Image Service to a data store that is visible to the hypervisor. To optimize this process, the first time a VMDK file is used, it gets cached in the data store. Subsequent virtual machines that need the VMDK use the cached version and don't have to copy the file again from the OpenStack Image Service.

Even with a cached VMDK, there is still a copy operation from the cache location to the hypervisor file directory in the shared data store. To avoid this copy, boot the image in linked_clone mode. To learn how to enable this mode, see the section called "Configuration reference" [99]. Note also that it is possible to override the linked_clone mode on a per-image basis by using the `vmware_linked_clone` property in the OpenStack Image Service.

Networking with VMware vSphere

The VMware driver supports networking with the `nova-network` service or the OpenStack Networking Service. Depending on your installation, complete these configuration steps before you provision VMs:

- **The `nova-network` service with the FlatManager or FlatDHCPManager**. Create a port group with the same name as the `flat_network_bridge` value in the `nova.conf` file. The default value is `br100`.

  All VM NICs are attached to this port group.

  Ensure that the flat interface of the node that runs the `nova-network` service has a path to this network.

- **The `nova-network` service with the VlanManager**. Set the `vlan_interface` configuration option to match the ESX host interface that handles VLAN-tagged VM traffic.

  OpenStack Compute automatically creates the corresponding port groups.

- **The OpenStack Networking Service**. If you use OVS as the l2 agent, create a port group with the same name as the `DEFAULT.neutron_ovs_bridge` value in the `nova.conf` file. Otherwise, create a port group with the same name as the `vmware.integration_bridge` value in the `nova.conf` file. In both cases, the default value is `br-int`.

  All VM NICs are attached to this port group for management by the OpenStack Networking plug-in.

Volumes with VMware vSphere

The VMware driver supports attaching volumes from the OpenStack Block Storage service. The VMware VMDK driver for OpenStack Block Storage is recommended and should be used for managing volumes based on vSphere data stores. More information about the VMware VMDK driver can be found at: VMware VMDK Driver. Also an iscsi volume driver provides limited support and can be used only for attachments.

vSphere 5.0 and earlier additional set up

Users of vSphere 5.0 or earlier must host their WSDL files locally. These steps are applicable for vCenter 5.0 or ESXi 5.0 and you can either mirror the WSDL from the vCenter or ESXi server that you intend to use or you can download the SDK directly from VMware. These workaround steps fix a known issue with the WSDL that was resolved in later versions.

When setting the VMwareVCDriver configuration options, you must include the `wsdl_location` option. For more information, see VMwareVCDriver configuration options above.

Procedure 2.8. Mirror WSDL from vCenter (or ESXi)

1. Set the `VMWAREAPI_IP` shell variable to the IP address for your vCenter or ESXi host from where you plan to mirror files. For example:

```
$ export VMWAREAPI_IP=<your_vsphere_host_ip>
```

2. Create a local file system directory to hold the WSDL files:

```
$ mkdir -p /opt/stack/vmware/wsdl/5.0
```

3. Change into the new directory.

```
$ cd /opt/stack/vmware/wsdl/5.0
```

4. Use your OS-specific tools to install a command-line tool that can download files like **wget**.

5. Download the files to the local file cache:

```
wget  --no-check-certificate https://$VMWAREAPI_IP/sdk/vimService.wsdl
wget  --no-check-certificate https://$VMWAREAPI_IP/sdk/vim.wsdl
wget  --no-check-certificate https://$VMWAREAPI_IP/sdk/core-types.xsd
wget  --no-check-certificate https://$VMWAREAPI_IP/sdk/query-messagetypes.
xsd
wget  --no-check-certificate https://$VMWAREAPI_IP/sdk/query-types.xsd
wget  --no-check-certificate https://$VMWAREAPI_IP/sdk/vim-messagetypes.
xsd
wget  --no-check-certificate https://$VMWAREAPI_IP/sdk/vim-types.xsd
wget  --no-check-certificate https://$VMWAREAPI_IP/sdk/reflect-
messagetypes.xsd
wget  --no-check-certificate https://$VMWAREAPI_IP/sdk/reflect-types.xsd
```

Because the `reflect-types.xsd` and `reflect-messagetypes.xsd` files do not fetch properly, you must stub out these files. Use the following XML listing to replace the missing file content. The XML parser underneath Python can be very particular and if you put a space in the wrong place, it can break the parser. Copy the following contents and formatting carefully.

```
<?xml version="1.0" encoding="UTF-8"?>
  <schema
     targetNamespace="urn:reflect"
     xmlns="http://www.w3.org/2001/XMLSchema"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema"
     elementFormDefault="qualified">
  </schema>
```

6. Now that the files are locally present, tell the driver to look for the SOAP service WSDLs in the local file system and not on the remote vSphere server. Add the following setting to the `nova.conf` file for your `nova-compute` node:

```
[vmware]
wsdl_location=file:///opt/stack/vmware/wsdl/5.0/vimService.wsdl
```

Alternatively, download the version appropriate SDK from http://www.vmware.com/support/developer/vc-sdk/ and copy it to the `/opt/stack/vmware` file. Make sure that the WSDL is available, in for example `/opt/stack/vmware/SDK/wsdl/vim25/vimService.wsdl`. You must point `nova.conf` to fetch this WSDL file from the local file system by using a URL.

When using the VMwareVCDriver (vCenter) with OpenStack Compute with vSphere version 5.0 or earlier, `nova.conf` must include the following extra config option:

```
[vmware]
wsdl_location=file:///opt/stack/vmware/SDK/wsdl/vim25/vimService.wsdl
```

VMware ESX driver

This section covers details of using the VMwareESXDriver. The ESX Driver has not been extensively tested and is not recommended. To configure the VMware vCenter driver instead, see the section called “VMware vCenter driver” [92].

VMwareESXDriver configuration options

When you use the VMwareESXDriver (no vCenter) with OpenStack Compute, add the following VMware-specific configuration options to the `nova.conf` file:

```
[DEFAULT]
compute_driver=vmwareapi.VMwareESXDriver

[vmware]
host_ip=<ESXi host IP>
host_username=<ESXi host username>
host_password=<ESXi host password>
wsdl_location=http://127.0.0.1:8080/vmware/SDK/wsdl/vim25/vimService.wsdl
```

Remember that you will have one `nova-compute` service for each ESXi host. It is recommended that this host run as a VM on the same ESXi host that it manages.

Note

Many `nova.conf` options are relevant to libvirt but do not apply to this driver.

Requirements and limitations

The ESXDriver cannot use many of the vSphere platform advanced capabilities, namely vMotion, high availability, and DRS.

Configuration reference

Table 2.24. Description of configuration options for vmware

Configuration option=Default value	Description
api_retry_count=10	(IntOpt) The number of times we retry on failures, e.g., socket error, etc. Used only if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.
cluster_name=None	(MultiStrOpt) Name of a VMware Cluster ComputeResource. Used only if compute_driver is vmwareapi.VMwareVCDriver.
datastore_regex=None	(StrOpt) Regex to match the name of a datastore. Used only if compute_driver is vmwareapi.VMwareVCDriver.
host_ip=None	(StrOpt) URL for connection to VMware ESX/VC host. Required if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.
host_username=None	(StrOpt) Username for connection to VMware ESX/VC host. Used only if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.
host_password=None	(StrOpt) Password for connection to VMware ESX/VC host. Used only if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.
integration_bridge=br-int	(StrOpt) Name of Integration Bridge
maximum_objects=100	(IntOpt) The maximum number of ObjectContent data objects that should be returned in a single result. A positive value will cause the operation to suspend the retrieval when the count of objects reaches the specified maximum. The server may still limit the count to something less than the configured value. Any remaining objects may be retrieved with additional requests.
task_poll_interval=5.0	(FloatOpt) The interval used for polling of remote tasks. Used only if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.
use_linked_clone=True	(BoolOpt) Whether to use linked clone
wsdl_location=None	(StrOpt) Optional VIM Service WSDL Location e.g http://<server>/vimService.wsdl. Optional over-ride to default location for bug work-arounds

PowerVM

Introduction

PowerVM compute driver connects to an Integrated Virtualization Manager (IVM) to perform PowerVM Logical Partition (LPAR) deployment and management. The driver supports file-based deployment using images from the OpenStack Image Service.

Note

Hardware Management Console (HMC) is not yet supported.

For more detailed information about PowerVM Virtualization system, refer to the IBM Redbook publication: IBM PowerVM Virtualization Introduction and Configuration.

Configuration

To enable the PowerVM compute driver, add the following configuration options `/etc/nova/nova.conf`:

```
compute_driver=nova.virt.powervm.PowerVMDriver
powervm_mgr_type=ivm
powervm_mgr=powervm_hostname_or_ip_address
powervm_mgr_user=padmin
powervm_mgr_passwd=padmin_user_password
powervm_img_remote_path=/path/to/remote/image/directory
powervm_img_local_path=/path/to/local/image/directory/on/compute/host
```

Table 2.25. Description of configuration options for powervm

Configuration option=Default value	Description
powervm_img_local_path=/tmp	(StrOpt) Local directory to download glance images to. Make sure this path can fit your biggest image in glance
powervm_img_remote_path=/home/padmin	(StrOpt) PowerVM image remote path where images will be moved. Make sure this path can fit your biggest image in glance
powervm_mgr=None	(StrOpt) PowerVM manager host or ip
powervm_mgr_passwd=None	(StrOpt) PowerVM manager user password
powervm_mgr_type=ivm	(StrOpt) PowerVM manager type (ivm, hmc)
powervm_mgr_user=None	(StrOpt) PowerVM manager user name

Limitations

PowerVM LPARs names have a limit of 31 characters. Since OpenStack Compute instance names are mapped to LPAR names in Power Systems, make sure `instance_name_template` config option in `nova.conf` yields names that have 31 or fewer characters.

Hyper-V virtualization platform

It is possible to use Hyper-V as a compute node within an OpenStack Deployment. The `nova-compute` service runs as "openstack-compute," a 32-bit service directly upon the Windows platform with the Hyper-V role enabled. The necessary Python components as well as the `nova-compute` service are installed directly onto the Windows platform. Windows Clustering Services are not needed for functionality within the OpenStack infrastructure. The use of the Windows Server 2012 platform is recommend for the best experience and is the platform for active development. The following Windows platforms have been tested as compute nodes:

- **Windows Server 2008r2**

  Both Server and Server Core with the Hyper-V role enabled (Shared Nothing Live migration is not supported using 2008r2)

- **Windows Server 2012**

  Server and Core (with the Hyper-V role enabled), and Hyper-V Server

Hyper-V configuration

The following sections discuss how to prepare the Windows Hyper-V node for operation as an OpenStack Compute node. Unless stated otherwise, any configuration information should work for both the Windows 2008r2 and 2012 platforms.

Local Storage Considerations

The Hyper-V compute node needs to have ample storage for storing the virtual machine images running on the compute nodes. You may use a single volume for all, or partition it into an OS volume and VM volume. It is up to the individual deploying to decide.

Configure NTP

Network time services must be configured to ensure proper operation of the Hyper-V compute node. To set network time on your Hyper-V host you will need to run the following commands

```
C:\net stop w32time
```

```
C:\w32tm /config /manualpeerlist:pool.ntp.org,0x8 /syncfromflags:MANUAL
```

```
C:\net start w32time
```

Configure Hyper-V virtual switching

Information regarding the Hyper-V virtual Switch can be located here: http://technet.microsoft.com/en-us/library/hh831823.aspx

To quickly enable an interface to be used as a Virtual Interface the following PowerShell may be used:

```
PS C:\$if = Get-NetIPAddress –IPAddress 192* | Get-NetIPInterface
```

```
PS C:\New-VMSwitch -NetAdapterName $if.ifAlias -Name yourbridgename –
AllowManagementOS $false
```

Enable iSCSI initiator service

To prepare the Hyper-V node to be able to attach to volumes provided by cinder you must first make sure the Windows iSCSI initiator service is running and started automatically.

```
C:\sc start MSiSCSI
```

```
C:\sc config MSiSCSI start="auto"
```

Configure shared nothing live migration

Detailed information on the configuration of live migration can be found here: http://technet.microsoft.com/en-us/library/jj134199.aspx

The following outlines the steps of shared nothing live migration.

1. The target hosts ensures that live migration is enabled and properly configured in Hyper-V.
2. The target hosts checks if the image to be migrated requires a base VHD and pulls it from Glance if not already available on the target host.
3. The source hosts ensures that live migration is enabled and properly configured in Hyper-V.
4. The source hosts initiates a Hyper-V live migration.
5. The source hosts communicates to the manager the outcome of the operation.

The following two configuration options/flags are needed in order to support Hyper-V live migration and must be added to your `nova.conf` on the Hyper-V compute node:

- `instances_shared_storage=False`

  This needed to support "shared nothing" Hyper-V live migrations. It is used in nova/compute/manager.py

- `limit_cpu_features=True`

  This flag is needed to support live migration to hosts with different CPU features. This flag is checked during instance creation in order to limit the CPU features used by the VM.

- `instances_path=DRIVELETTER:\PATH\TO\YOUR\INSTANCES`

Additional Requirements:

- Hyper-V 2012 RC or Windows Server 2012 RC with Hyper-V role enabled
- A Windows domain controller with the Hyper-V compute nodes as domain members
- The instances_path command line option/flag needs to be the same on all hosts
- The openstack-compute service deployed with the setup must run with domain credentials. You can set the service credentials with:

```
C:\sc config openstack-compute obj="DOMAIN\username" password="password"
```

How to setup live migration on Hyper-V

To enable shared nothing live migration run the 3 PowerShell instructions below on each Hyper-V host:

```
PS C:\Enable-VMMigration
```

```
PS C:\Set-VMMigrationNetwork IP_ADDRESS
```

```
PS C:\Set-VMHost –VirtualMachineMigrationAuthenticationTypeKerberos
```

Note

Please replace the IP_ADDRESS with the address of the interface which will provide the virtual switching for nova-network.

Additional Reading

Here's an article that clarifies the various live migration options in Hyper-V:

http://ariessysadmin.blogspot.ro/2012/04/hyper-v-live-migration-of-windows.html

Python Requirements

Python

Python 2.7.3 must be installed prior to installing the OpenStack Compute Driver on the Hyper-V server. Download and then install the MSI for windows here:

- http://www.python.org/ftp/python/2.7.3/python-2.7.3.msi
- Install the MSI accepting the default options.
- The installation will put python in C:/python27.

Setuptools

You will require pip to install the necessary python module dependencies. The installer will install under the C:\python27 directory structure. Setuptools for Python 2.7 for Windows can be download from here:

http://pypi.python.org/packages/2.7/s/setuptools/setuptools-0.6c11.win32-py2.7.exe

Python Dependencies

The following packages need to be downloaded and manually installed onto the Compute Node

- **MySQL-python**

http://codegood.com/download/10/

- **pywin32**

  Download and run the installer from the following location

  http://sourceforge.net/projects/pywin32/files/pywin32/Build%20217/pywin32-217.win32-py2.7.exe

- **greenlet**

  Select the link below:

  http://www.lfd.uci.edu/~gohlke/pythonlibs/

  You will need to scroll down to the greenlet section for the following file: greenlet-0.4.0.win32-py2.7.#exe

  Click on the file, to initiate the download. Once the download is complete, run the installer.

The following python packages need to be installed via easy_install or pip. Run the following replacing PACKAGENAME with the packages below:

```
C:\c:\Python27\Scripts\pip.exe install PACKAGE_NAME
```

- amqplib
- anyjson
- distribute
- eventlet
- httplib2
- iso8601
- jsonschema
- kombu
- netaddr
- paste
- paste-deploy
- prettytable
- python-cinderclient

- python-glanceclient
- python-keystoneclient
- repoze.lru
- routes
- sqlalchemy
- simplejson
- warlock
- webob
- wmi

Install Nova-compute

Using git on Windows to retrieve source

Git be used to download the necessary source code. The installer to run Git on Windows can be downloaded here:

http://code.google.com/p/msysgit/downloads/list?q=full+installer+official+git

Download the latest installer. Once the download is complete double click the installer and follow the prompts in the installation wizard. The default should be acceptable for the needs of the document.

Once installed you may run the following to clone the Nova code.

```
C:\git.exe clone https://github.com/openstack/nova.git
```

Configure Nova.conf

The `nova.conf` file must be placed in `C:\etc\nova` for running OpenStack on Hyper-V. Below is a sample `nova.conf` for Windows:

```
[DEFAULT]
verbose=true
force_raw_images=false
auth_strategy=keystone
fake_network=true
vswitch_name=openstack-br
logdir=c:\openstack\
state_path=c:\openstack\
lock_path=c:\openstack\
instances_path=e:\Hyper-V\instances
policy_file=C:\Program Files (x86)\OpenStack\nova\etc\nova\policy.json
api_paste_config=c:\openstack\nova\etc\nova\api-paste.ini
rabbit_host=IP_ADDRESS
glance_api_servers=IP_ADDRESS:9292
```

```
image_service=nova.image.glance.GlanceImageService
instances_shared_storage=false
limit_cpu_features=true
compute_driver=nova.virt.hyperv.driver.HyperVDriver
volume_api_class=nova.volume.cinder.API
[database]
connection=mysql://nova:passwd@IP_ADDRESS/nova
```

The following table contains a reference of all options for hyper-v

Table 2.26. Description of configuration options for hyperv

Configuration option=Default value	Description
dynamic_memory_ratio=1.0	(FloatOpt) Enables dynamic memory allocation (ballooning) when set to a value greater than 1. The value expresses the ratio between the total RAM assigned to an instance and its startup RAM amount. For example a ratio of 2.0 for an instance with 1024MB of RAM implies 512MB of RAM allocated at startup
enable_instance_metrics_collection=False	(BoolOpt) Enables metrics collections for an instance by using Hyper-V's metric APIs. Collected data can by retrieved by other apps and services, e.g.: Ceilometer. Requires Hyper-V / Windows Server 2012 and above
force_hyperv_utils_v1=False	(BoolOpt) Force V1 WMI utility classes
instances_path_share=	(StrOpt) The name of a Windows share name mapped to the "instances_path" dir and used by the resize feature to copy files to the target host. If left blank, an administrative share will be used, looking for the same "instances_path" used locally
limit_cpu_features=False	(BoolOpt) Required for live migration among hosts with different CPU features
qemu_img_cmd=qemu-img.exe	(StrOpt) qemu-img is used to convert between different image types
vswitch_name=None	(StrOpt) External virtual switch Name, if not provided, the first external virtual switch is used

Prepare images for use with Hyper-V

Hyper-V currently supports only the VHD file format for virtual machine instances. Detailed instructions for installing virtual machines on Hyper-V can be found here:

http://technet.microsoft.com/en-us/library/cc772480.aspx

Once you have successfully created a virtual machine, you can then upload the image to glance using the native glance-client:

```
C:\glance image-create --name="VM_IMAGE_NAME" --is-public=true --container-
format=bare --disk-format=vhd
```

Run Compute with Hyper-V

To start the `nova-compute` service, run this command from a console in the Windows server:

```
C:\C:\python27\python.exe c:\openstack\nova\bin\nova-compute.py
```

Troubleshoot Hyper-V configuration

- I ran the `nova-manage service list` command from my controller; however, I'm not seeing smiley faces for Hyper-V compute nodes, what do I do?

  Verify that you are synchronized with a network time source. Instructions for configuring NTP on your Hyper-V compute node are located here

Bare metal driver

The baremetal driver is a hypervisor driver for OpenStack Nova Compute. Within the OpenStack framework, it has the same role as the drivers for other hypervisors (libvirt, xen, etc), and yet it is presently unique in that the hardware is not virtualized - there is no hypervisor between the tenants and the physical hardware. It exposes hardware through the OpenStack APIs, using pluggable sub-drivers to deliver machine imaging (PXE) and power control (IPMI). With this, provisioning and management of physical hardware is accomplished by using common cloud APIs and tools, such as the Orchestration module (heat) or salt-cloud. However, due to this unique situation, using the baremetal driver requires some additional preparation of its environment, the details of which are beyond the scope of this guide.

Note

Some OpenStack Compute features are not implemented by the baremetal hypervisor driver. See the hypervisor support matrix for details.

For the Baremetal driver to be loaded and function properly, ensure that the following options are set in `/etc/nova/nova.conf` on your `nova-compute` hosts.

```
[default]
compute_driver=nova.virt.baremetal.driver.BareMetalDriver
firewall_driver = nova.virt.firewall.NoopFirewallDriver
scheduler_host_manager=nova.scheduler.baremetal_host_manager.
BaremetalHostManager
ram_allocation_ratio=1.0
reserved_host_memory_mb=0
```

Many configuration options are specific to the Baremetal driver. Also, some additional steps are required, such as building the baremetal deploy ramdisk. See the main wiki page for details and implementation suggestions.

Table 2.27. Description of configuration options for baremetal

Configuration option=Default value	Description
db_backend=sqlalchemy	(StrOpt) The backend to use for bare-metal database
deploy_kernel=None	(StrOpt) Default kernel image ID used in deployment phase
deploy_ramdisk=None	(StrOpt) Default ramdisk image ID used in deployment phase
driver=nova.virt.baremetal.pxe.PXE	(StrOpt) Baremetal driver back-end (pxe or tilera)

Configuration option=Default value	Description
driver=nova.cells.rpc_driver.CellsRPCDriver	(StrOpt) Cells communication driver to use
instance_type_extra_specs=	(ListOpt) a list of additional capabilities corresponding to instance_type_extra_specs for this compute host to advertise. Valid entries are name=value, pairs For example, "key1:val1, key2:val2"
ipmi_power_retry=5	(IntOpt) maximal number of retries for IPMI operations
net_config_template=$pybasedir/nova/virt/baremetal/net-dhcp.ubuntu.template	(StrOpt) Template file for injected network config
power_manager=nova.virt.baremetal.ipmi.IPMI	(StrOpt) Baremetal power management method
pxe_append_params=None	(StrOpt) additional append parameters for baremetal PXE boot
pxe_bootfile_name=pxelinux.0	(StrOpt) This gets passed to Neutron as the bootfile dhcp parameter when the dhcp_options_enabled is set.
pxe_config_template=$pybasedir/nova/virt/baremetal/pxe_config.template	(StrOpt) Template file for PXE configuration
pxe_deploy_timeout=0	(IntOpt) Timeout for PXE deployments. Default: 0 (unlimited)
pxe_network_config=False	(BoolOpt) If set, pass the network configuration details to the initramfs via cmdline.
sql_connection=sqlite:///$state_path/baremetal_$sqlite_db	(StrOpt) The SQLAlchemy connection string used to connect to the bare-metal database
terminal=shellinaboxd	(StrOpt) path to baremetal terminal program
terminal_cert_dir=None	(StrOpt) path to baremetal terminal SSL cert(PEM)
terminal_pid_dir=$state_path/baremetal/console	(StrOpt) path to directory stores pidfiles of baremetal_terminal
tftp_root=/tftpboot	(StrOpt) Baremetal compute node's tftp root path
use_unsafe_iscsi=False	(BoolOpt) Do not set this out of dev/test environments. If a node does not have a fixed PXE IP address, volumes are exported with globally opened ACL
vif_driver=nova.virt.baremetal.vif_driver.BareMetalVIFDriver	(StrOpt) Baremetal VIF driver.
virtual_power_host_key=None	(StrOpt) ssh key for virtual power host_user
virtual_power_host_pass=	(StrOpt) password for virtual power host_user
virtual_power_host_user=	(StrOpt) user to execute virtual power commands as
virtual_power_ssh_host=	(StrOpt) ip or name to virtual power host
virtual_power_ssh_port=22	(IntOpt) Port to use for ssh to virtual power host
virtual_power_type=virsh	(StrOpt) base command to use for virtual power(vbox,virsh)

Docker driver

The Docker driver is a hypervisor driver for OpenStack Compute, introduced with the Havana release. Docker is an open-source engine which automates the deployment of applications as highly portable, self-sufficient containers which are independent of hardware, language, framework, packaging system and hosting provider. Docker extends LXC with a high level API providing a lightweight virtualization solution that runs processes in isolation. It provides a way to automate software deployment in a secure and repeatable environment. A standard container in Docker contains a software component along with all of its dependencies - binaries, libraries, configuration files, scripts, virtualenvs, jars, gems and tarballs. Docker can be run on any x86_64 Linux kernel that supports cgroups and aufs. Docker is a way of managing LXC containers on a single machine. However used behind OpenStack Compute makes Docker much more powerful since it is then possible to

manage several hosts which will then manage hundreds of containers. The current Docker project aims for full OpenStack compatibility. Containers don't aim to be a replacement for VMs, they are just complementary in the sense that they are better for specific use cases. Compute's support for VMs is currently advanced thanks to the variety of hypervisors running VMs. However it's not the case for containers even though libvirt/LXC is a good starting point. Docker aims to go the second level of integration.

Note

Some OpenStack Compute features are not implemented by the docker driver. See the hypervisor support matrix for details.

To enable Docker, ensure the following options are set in `/etc/nova/nova-compute.conf` on all hosts running the `nova-compute` service.

```
compute_driver=docker.DockerDriver
```

Glance also needs to be configured to support the Docker container format, in `/etc/glance-api.conf`:

```
container_formats = ami,ari,aki,bare,ovf,docker
```

Table 2.28. Description of configuration options for docker

Configuration option=Default value	Description
docker_registry_default_port=5042	(IntOpt) Default TCP port to find the docker-registry container

Scheduling

Compute uses the `nova-scheduler` service to determine how to dispatch compute and volume requests. For example, the `nova-scheduler` service determines which host a VM should launch on. The term *host* in the context of filters means a physical node that has a `nova-compute` service running on it. You can configure the scheduler through a variety of options.

Compute is configured with the following default scheduler options:

```
scheduler_driver=nova.scheduler.multi.MultiScheduler
compute_scheduler_driver=nova.scheduler.filter_scheduler.FilterScheduler
scheduler_available_filters=nova.scheduler.filters.all_filters
scheduler_default_filters=AvailabilityZoneFilter,RamFilter,ComputeFilter
least_cost_functions=nova.scheduler.least_cost.compute_fill_first_cost_fn
compute_fill_first_cost_fn_weight=-1.0
```

By default, the compute scheduler is configured as a filter scheduler, as described in the next section. In the default configuration, this scheduler considers hosts that meet all the following criteria:

- Are in the requested availability zone (`AvailabilityZoneFilter`).
- Have sufficient RAM available (`RamFilter`).
- Are capable of servicing the request (`ComputeFilter`).

For information on the volume scheduler, refer the Block Storage section of *OpenStack Cloud Administrator Guide* for information.

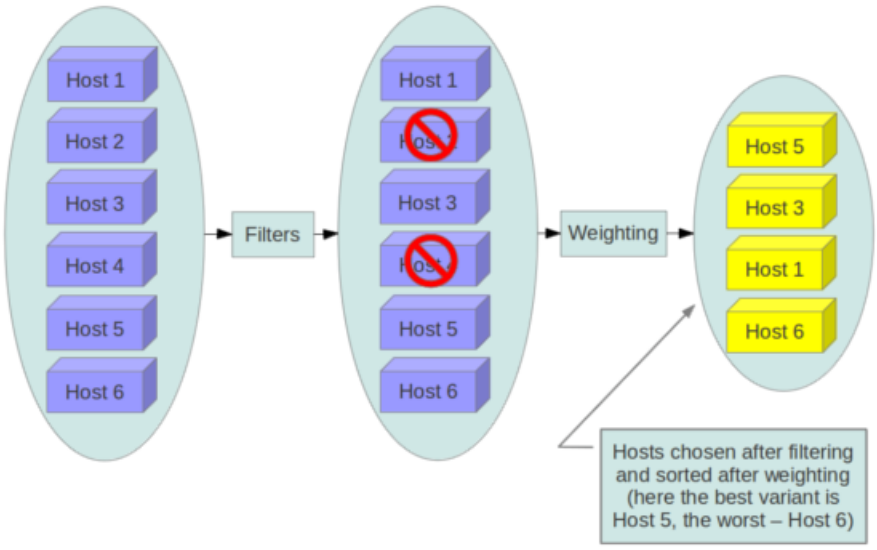
Filter scheduler

The Filter Scheduler (`nova.scheduler.filter_scheduler.FilterScheduler`) is the default scheduler for scheduling virtual machine instances. It supports filtering and weighting to make informed decisions on where a new instance should be created. You can use this scheduler to schedule compute requests but not volume requests. For example, you can use it with only the `compute_scheduler_driver` configuration option.

Filters

When the Filter Scheduler receives a request for a resource, it first applies filters to determine which hosts are eligible for consideration when dispatching a resource. Filters are binary: either a host is accepted by the filter, or it is rejected. Hosts that are accepted by the filter are then processed by a different algorithm to decide which hosts to use for that request, described in the Weights section.

Figure 2.5. Filtering



The `scheduler_available_filters` configuration option in `nova.conf` provides the Compute service with the list of the filters that are used by the scheduler. The default setting specifies all of the filter that are included with the Compute service:

```
scheduler_available_filters=nova.scheduler.filters.all_filters
```

This configuration option can be specified multiple times. For example, if you implemented your own custom filter in Python called `myfilter.MyFilter` and you wanted to use both the built-in filters and your custom filter, your `nova.conf` file would contain:

```
scheduler_available_filters=nova.scheduler.filters.all_filters
scheduler_available_filters=myfilter.MyFilter
```

The `scheduler_default_filters` configuration option in `nova.conf` defines the list of filters that are applied by the `nova-scheduler` service. As mentioned, the default filters are:

```
scheduler_default_filters=AvailabilityZoneFilter,RamFilter,ComputeFilter
```

The following sections describe the available filters.

AggregateCoreFilter

Implements blueprint per-aggregate-resource-ratio. AggregateCoreFilter supports per-aggregate `cpu_allocation_ratio`. If the per-aggregate value is not found, the value falls back to the global setting.

AggregateInstanceExtraSpecsFilter

Matches properties defined in an instance type's extra specs against admin-defined properties on a host aggregate. Works with specifications that are unscoped, or are scoped with `aggregate_instance_extra_specs`. See the host aggregates section for documentation on how to use this filter.

AggregateMultiTenancyIsolation

Isolates tenants to specific host aggregates. If a host is in an aggregate that has the metadata key `filter_tenant_id` it only creates instances from that tenant (or list of tenants). A host can be in different aggregates. If a host does not belong to an aggregate with the metadata key, it can create instances from all tenants.

AggregateRamFilter

Implements blueprint `per-aggregate-resource-ratio`. Supports per-aggregate `ram_allocation_ratio`. If per-aggregate value is not found, it falls back to the default setting.

AllHostsFilter

This is a no-op filter, it does not eliminate any of the available hosts.

AvailabilityZoneFilter

Filters hosts by availability zone. This filter must be enabled for the scheduler to respect availability zones in requests.

ComputeCapabilitiesFilter

Matches properties defined in an instance type's extra specs against compute capabilities.

If an extra specs key contains a colon ":", anything before the colon is treated as a namespace, and anything after the colon is treated as the key to be matched. If a namespace is present and is not 'capabilities', it is ignored by this filter.

Note

Disable the ComputeCapabilitiesFilter when using a Bare Metal configuration, due to bug 1129485

ComputeFilter

Passes all hosts that are operational and enabled.

In general, this filter should always be enabled.

CoreFilter

Only schedule instances on hosts if there are sufficient CPU cores available. If this filter is not set, the scheduler may over provision a host based on cores (for example, the virtual cores running on an instance may exceed the physical cores).

This filter can be configured to allow a fixed amount of vCPU overcommitment by using the `cpu_allocation_ratio` Configuration option in `nova.conf`. The default setting is:

```
cpu_allocation_ratio=16.0
```

With this setting, if 8 vCPUs are on a node, the scheduler allows instances up to 128 vCPU to be run on that node.

To disallow vCPU overcommitment set:

```
cpu_allocation_ratio=1.0
```

DifferentHostFilter

Schedule the instance on a different host from a set of instances. To take advantage of this filter, the requester must pass a scheduler hint, using `different_host` as the key and a list of instance uuids as the value. This filter is the opposite of the `SameHostFilter`. Using the **nova** command-line tool, use the `--hint` flag. For example:

```
$ nova boot --image cedef40a-ed67-4d10-800e-17455edce175 --flavor 1 \
  --hint different_host=a0cf03a5-d921-4877-bb5c-86d26cf818e1 \
  --hint different_host=8c19174f-4220-44f0-824a-cd1eeef10287 server-1
```

With the API, use the `os:scheduler_hints` key. For example:

```
{
'server': {
    'name': 'server-1',
    'imageRef': 'cedef40a-ed67-4d10-800e-17455edce175',
    'flavorRef': '1'
},
'os:scheduler_hints': {
    'different_host': ['a0cf03a5-d921-4877-bb5c-86d26cf818e1',
                       '8c19174f-4220-44f0-824a-cd1eeef10287'],
}
```

DiskFilter

Only schedule instances on hosts if there is sufficient disk space available for root and ephemeral storage.

This filter can be configured to allow a fixed amount of disk overcommitment by using the `disk_allocation_ratio` Configuration option in `nova.conf`. The default setting is:

```
disk_allocation_ratio=1.0
```

Adjusting this value to greater than 1.0 enables scheduling instances while over committing disk resources on the node. This might be desirable if you use an image format that is sparse or copy on write such that each virtual instance does not require a 1:1 allocation of virtual disk to physical storage.

GroupAffinityFilter

The GroupAffinityFilter ensures that an instance is scheduled on to a host from a set of group hosts. To take advantage of this filter, the requester must pass a scheduler hint, using `group` as the key and an arbitrary name as the value. Using the **nova** command-line tool, use the `--hint` flag. For example:

```
$ nova boot --image cedef40a-ed67-4d10-800e-17455edce175 --flavor 1 \
  --hint group=foo server-1
```

GroupAntiAffinityFilter

The GroupAntiAffinityFilter ensures that each instance in a group is on a different host. To take advantage of this filter, the requester must pass a scheduler hint, using `group` as the key and an arbitrary name as the value. Using the **nova** command-line tool, use the `--hint` flag. For example:

```
$ nova boot --image cedef40a-ed67-4d10-800e-17455edce175 --flavor 1 \
  --hint group=foo server-1
```

ImagePropertiesFilter

Filters hosts based on properties defined on the instance's image. It passes hosts that can support the specified image properties contained in the instance. Properties include the architecture, hypervisor type, and virtual machine mode. for example, an instance might require a host that runs an ARM-based processor and QEMU as the hypervisor. An image can be decorated with these properties by using:

```
$ glance image-update img-uuid --property architecture=arm --property
 hypervisor_type=qemu
```

The image properties that the filter checks for are:

- `architecture`: Architecture describes the machine architecture required by the image. Examples are i686, x86_64, arm, and ppc64.
- `hypervisor_type`: Hypervisor type describes the hypervisor required by the image. Examples are xen, kvm, qemu, xenapi, and powervm.
- `vm_mode`: Virtual machine mode describes the hypervisor application binary interface (ABI) required by the image. Examples are 'xen' for Xen 3.0 paravirtual ABI, 'hvm' for native ABI, 'uml' for User Mode Linux paravirtual ABI, exe for container virt executable ABI.

IsolatedHostsFilter

Allows the admin to define a special (isolated) set of images and a special (isolated) set of hosts, such that the isolated images can only run on the isolated hosts, and the isolated hosts can only run isolated images.

The admin must specify the isolated set of images and hosts in the `nova.conf` file using the `isolated_hosts` and `isolated_images` configuration options. For example:

```
isolated_hosts=server1,server2
isolated_images=342b492c-128f-4a42-8d3a-c5088cf27d13,ebd267a6-ca86-4d6c-9a0e-bd132d6b7d09
```

JsonFilter

The JsonFilter allows a user to construct a custom filter by passing a scheduler hint in JSON format. The following operators are supported:

- =
- <
- >
- in
- <=
- >=
- not
- or
- and

The filter supports the following variables:

- $free_ram_mb
- $free_disk_mb
- $total_usable_ram_mb
- $vcpus_total
- $vcpus_used

Using the **nova** command-line tool, use the `--hint` flag:

```
$ nova boot --image 827d564a-e636-4fc4-a376-d36f7ebe1747 \
  --flavor 1 --hint query='[">=","$free_ram_mb",1024]' server1
```

With the API, use the `os:scheduler_hints` key:

```
    {
    'server': {
        'name': 'server-1',
        'imageRef': 'cedef40a-ed67-4d10-800e-17455edce175',
        'flavorRef': '1'
    },
    'os:scheduler_hints': {
        'query': '[">=","$free_ram_mb",1024]',
    }
}
```

RamFilter

Only schedule instances on hosts that have sufficient RAM available. If this filter is not set, the scheduler may over provision a host based on RAM (for example, the RAM allocated by virtual machine instances may exceed the physical RAM).

This filter can be configured to allow a fixed amount of RAM overcommitment by using the `ram_allocation_ratio` configuration option in `nova.conf`. The default setting is:

```
ram_allocation_ratio=1.5
```

With this setting, if there is 1GB of free RAM, the scheduler allows instances up to size 1.5GB to be run on that instance.

RetryFilter

Filter out hosts that have already been attempted for scheduling purposes. If the scheduler selects a host to respond to a service request, and the host fails to respond to the request, this filter prevents the scheduler from retrying that host for the service request.

This filter is only useful if the `scheduler_max_attempts` configuration option is set to a value greater than zero.

SameHostFilter

Schedule the instance on the same host as another instance in a set of instances. To take advantage of this filter, the requester must pass a scheduler hint, using `same_host` as the key and a list of instance uuids as the value. This filter is the opposite of the `DifferentHostFilter`. Using the **nova** command-line tool, use the `--hint` flag:

```
$ nova boot --image cedef40a-ed67-4d10-800e-17455edce175 --flavor 1 \
  --hint same_host=a0cf03a5-d921-4877-bb5c-86d26cf818e1 \
  --hint same_host=8c19174f-4220-44f0-824a-cd1eeef10287 server-1
```

With the API, use the `os:scheduler_hints` key:

```
    {
    'server': {
        'name': 'server-1',
        'imageRef': 'cedef40a-ed67-4d10-800e-17455edce175',
        'flavorRef': '1'
    },
    'os:scheduler_hints': {
        'same_host': ['a0cf03a5-d921-4877-bb5c-86d26cf818e1',
                      '8c19174f-4220-44f0-824a-cd1eeef10287'],
    }
}
```

SimpleCIDRAffinityFilter

Schedule the instance based on host IP subnet range. To take advantage of this filter, the requester must specify a range of valid IP address in CIDR format, by passing two scheduler hints:

`build_near_host_ip`	The first IP address in the subnet (for example, `192.168.1.1`)
`cidr`	The CIDR that corresponds to the subnet (for example, `/24`)

Using the **nova** command-line tool, use the `--hint` flag. For example, to specify the IP subnet `192.168.1.1/24`

```
$ nova boot --image cedef40a-ed67-4d10-800e-17455edce175 --flavor 1 \
  --hint build_near_host_ip=192.168.1.1 --hint cidr=/24 server-1
```

With the API, use the `os:scheduler_hints` key:

```
{
    {
    'server': {
        'name': 'server-1',
        'imageRef': 'cedef40a-ed67-4d10-800e-17455edce175',
        'flavorRef': '1'
    },
    'os:scheduler_hints': {
        'build_near_host_ip': '192.168.1.1',
        'cidr': '24'
    }
}
```

Weights

The Filter Scheduler weighs hosts based on the config option `scheduler_weight_classes`, this defaults to `nova.scheduler.weights.all_weighers`, which selects the only weigher available – the RamWeigher. Hosts are then weighed and sorted with the largest weight winning.

```
scheduler_weight_classes=nova.scheduler.weights.all_weighers
ram_weight_multiplier=1.0
```

The default is to spread instances across all hosts evenly. Set the `ram_weight_multiplier` option to a negative number if you prefer stacking instead of spreading.

Chance Scheduler

As an administrator, you work with the Filter Scheduler. However, the Compute service also uses the Chance Scheduler, `nova.scheduler.chance.ChanceScheduler`, which randomly selects from lists of filtered hosts. It is the default volume scheduler.

Host aggregates

Host aggregates are a mechanism to further partition an availability zone; while availability zones are visible to users, host aggregates are only visible to administrators. Host Aggregates provide a mechanism to allow administrators to assign key-value pairs to groups of machines. Each node can have multiple aggregates, each aggregate can have multiple key-value pairs, and the same key-value pair can be assigned to multiple aggregates. This information can be used in the scheduler to enable advanced scheduling, to set up hypervisor resource pools or to define logical groups for migration.

Command-line interface

The **nova** command-line tool supports the following aggregate-related commands.

nova aggregate-list Print a list of all aggregates.

nova aggregate-create <*name*> <*availability-zone*>	Create a new aggregate named <*name*> in availability zone <*availability-zone*>. Returns the ID of the newly created aggregate. Hosts can be made available to multiple availability zones, but administrators should be careful when adding the host to a different host aggregate within the same availability zone and pay attention when using the **aggregate-set-metadata** and **aggregate-update** commands to avoid user confusion when they boot instances in different availability zones. An error occurs if you cannot add a particular host to an aggregate zone for which it is not intended.
nova aggregate-delete <*id*>	Delete an aggregate with id <*id*>.
nova aggregate-details <*id*>	Show details of the aggregate with id <*id*>.
nova aggregate-add-host <*id*> <*host*>	Add host with name <*host*> to aggregate with id <*id*>.
nova aggregate-remove-host <*id*> <*host*>	Remove the host with name <*host*> from the aggregate with id <*id*>.
nova aggregate-set-metadata <*id*> <*key=value*> [<*key=value*> ...]	Add or update metadata (key-value pairs) associated with the aggregate with id <*id*>.
nova aggregate-update <*id*> <*name*> [<*availability_zone*>]	Update the name and availability zone (optional) for the aggregate.
nova host-list	List all hosts by service.
nova host-update – maintenance [enable | disable]	Put/resume host into/from maintenance.

Note

Only administrators can access these commands. If you try to use these commands and the user name and tenant that you use to access the Compute service do not have the `admin` role or the appropriate privileges, these errors occur:

```
ERROR: Policy doesn't allow compute_extension:aggregates to be
 performed. (HTTP 403) (Request-ID: req-299fbff6-6729-4cef-93b2-
e7e1f96b4864)
```

```
ERROR: Policy doesn't allow compute_extension:hosts to be performed.
 (HTTP 403) (Request-ID: req-ef2400f6-6776-4ea3-b6f1-7704085c27d1)
```

Configure scheduler to support host aggregates

One common use case for host aggregates is when you want to support scheduling instances to a subset of compute hosts because they have a specific capability. For example, you may want to allow users to request compute hosts that have SSD drives if they need

access to faster disk I/O, or access to compute hosts that have GPU cards to take advantage of GPU-accelerated code.

To configure the scheduler to support host aggregates, the `scheduler_default_filters` configuration option must contain the `AggregateInstanceExtraSpecsFilter` in addition to the other filters used by the scheduler. Add the following line to `/etc/nova/nova.conf` on the host that runs the `nova-scheduler` service to enable host aggregates filtering, as well as the other filters that are typically enabled:

```
scheduler_default_filters=AggregateInstanceExtraSpecsFilter,
AvailabilityZoneFilter,RamFilter,ComputeFilter
```

Example: Specify compute hosts with SSDs

This example configures the Compute service to enable users to request nodes that have solid-state drives (SSDs). You create a `fast-io` host aggregate in the `nova` availability zone and you add the `ssd=true` key-value pair to the aggregate. Then, you add the `node1`, and `node2` compute nodes to it.

```
$ nova aggregate-create fast-io nova
+----+---------+-------------------+-------+----------+
| Id | Name    | Availability Zone | Hosts | Metadata |
+----+---------+-------------------+-------+----------+
| 1  | fast-io | nova              |       |          |
+----+---------+-------------------+-------+----------+

$ nova aggregate-set-metadata 1 ssd=true
+----+---------+-------------------+-------+-------------------+
| Id | Name    | Availability Zone | Hosts | Metadata          |
+----+---------+-------------------+-------+-------------------+
| 1  | fast-io | nova              | []    | {u'ssd': u'true'} |
+----+---------+-------------------+-------+-------------------+

$ nova aggregate-add-host 1 node1
+----+---------+-------------------+------------+-------------------+
| Id | Name    | Availability Zone | Hosts      | Metadata          |
+----+---------+-------------------+------------+-------------------+
| 1  | fast-io | nova              | [u'node1'] | {u'ssd': u'true'} |
+----+---------+-------------------+------------+-------------------+

$ nova aggregate-add-host 1 node2
+----+---------+-------------------+---------------------+-------------------+
| Id | Name    | Availability Zone | Hosts               | Metadata
 |
+----+---------+-------------------+---------------------+-------------------
+
| 1  | fast-io | nova              | [u'node1', u'node2'] | {u'ssd': u'true'}
 |
+----+---------+-------------------+---------------------+-------------------
+
```

Use the **nova flavor-create** command to create the `ssd.large` flavor called with an ID of 6, 8GB of RAM, 80GB root disk, and 4 vCPUs.

```
$ nova flavor-create ssd.large 6 8192 80 4
+----+-----------+-----------+------+-----------+------+-------+-------------
+-----------+-------------+
```

```
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor |
 Is_Public | extra_specs |
+----+-----------+-----------+------+-----------+------+-------+-------------
+-----------+-------------+
| 6  | ssd.large | 8192      | 80   | 0         |      | 4     | 1           |
 True      | {}          |
+----+-----------+-----------+------+-----------+------+-------+-------------
+-----------+-------------+
```

Once the flavor is created, specify one or more key-value pairs that match the key-value pairs on the host aggregates. In this case, that is the `ssd=true` key-value pair. Setting a key-value pair on a flavor is done using the **nova flavor-key set_key** command.

```
# nova flavor-key set_key --name=ssd.large  --key=ssd --value=true
```

Once it is set, you should see the `extra_specs` property of the `ssd.large` flavor populated with a key of `ssd` and a corresponding value of `true`.

```
$ nova flavor-show ssd.large
+----------------------------+-------------------+
| Property                   | Value             |
+----------------------------+-------------------+
| OS-FLV-DISABLED:disabled   | False             |
| OS-FLV-EXT-DATA:ephemeral  | 0                 |
| disk                       | 80                |
| extra_specs                | {u'ssd': u'true'} |
| id                         | 6                 |
| name                       | ssd.large         |
| os-flavor-access:is_public | True              |
| ram                        | 8192              |
| rxtx_factor                | 1.0               |
| swap                       |                   |
| vcpus                      | 4                 |
+----------------------------+-------------------+
```

Now, when a user requests an instance with the `ssd.large` flavor, the scheduler only considers hosts with the `ssd=true` key-value pair. In this example, these are `node1` and `node2`.

XenServer hypervisor pools to support live migration

When using the XenAPI-based hypervisor, the Compute service uses host aggregates to manage XenServer Resource pools, which are used in supporting live migration.

Configuration reference

Table 2.29. Description of configuration options for scheduling

Configuration option=Default value	Description
cpu_allocation_ratio=16.0	(FloatOpt) Virtual CPU to physical CPU allocation ratio which affects all CPU filters. This configuration specifies a global ratio for CoreFilter. For AggregateCoreFilter, it will fall back to this configuration value if no per-aggregate setting found.
disk_allocation_ratio=1.0	(FloatOpt) virtual disk to physical disk allocation ratio
isolated_hosts=	(ListOpt) Host reserved for specific images
isolated_images=	(ListOpt) Images to run on isolated host

Configuration option=Default value	Description
max_instances_per_host=50	(IntOpt) Ignore hosts that have too many instances
max_io_ops_per_host=8	(IntOpt) Ignore hosts that have too many builds/resizes/ snaps/migrations
ram_allocation_ratio=1.5	(FloatOpt) Virtual ram to physical ram allocation ratio which affects all ram filters. This configuration specifies a global ratio for RamFilter. For AggregateRamFilter, it will fall back to this configuration value if no per-aggregate setting found.
ram_weight_multiplier=10.0	(FloatOpt) Multiplier used for weighing ram. Negative numbers mean to stack vs spread.
ram_weight_multiplier=1.0	(FloatOpt) Multiplier used for weighing ram. Negative numbers mean to stack vs spread.
reserved_host_disk_mb=0	(IntOpt) Amount of disk in MB to reserve for the host
reserved_host_memory_mb=512	(IntOpt) Amount of memory in MB to reserve for the host
restrict_isolated_hosts_to_isolated_images=True	(BoolOpt) Whether to force isolated hosts to run only isolated images
scheduler_available_filters=['nova.scheduler.filters.all_filters']	(MultiStrOpt) Filter classes available to the scheduler which may be specified more than once. An entry of "nova.scheduler.filters.standard_filters" maps to all filters included with nova.
scheduler_default_filters=RetryFilter,AvailabilityZoneFilter,RamFilter,ComputeFilter,ComputeCapabilitiesFilter,ImagePropertiesFilter	(ListOpt) Which filter class names to use for filtering hosts when not specified in the request.
scheduler_driver=nova.scheduler.filter_scheduler.FilterScheduler	(StrOpt) Default driver to use for the scheduler
scheduler_filter_classes=nova.cells.filters.all_filters	(ListOpt) Filter classes the cells scheduler should use. An entry of "nova.cells.filters.all_filters"maps to all cells filters included with nova.
scheduler_host_manager=nova.scheduler.host_manager.HostManager	(StrOpt) The scheduler host manager class to use
scheduler_host_subset_size=1	(IntOpt) New instances will be scheduled on a host chosen randomly from a subset of the N best hosts. This property defines the subset size that a host is chosen from. A value of 1 chooses the first host returned by the weighing functions. This value must be at least 1. Any value less than 1 will be ignored, and 1 will be used instead
scheduler_json_config_location=	(StrOpt) Absolute path to scheduler configuration JSON file.
scheduler_manager=nova.scheduler.manager.SchedulerManager	(StrOpt) full class name for the Manager for scheduler
scheduler_max_attempts=3	(IntOpt) Maximum number of attempts to schedule an instance
scheduler_retries=10	(IntOpt) How many retries when no cells are available.
scheduler_retry_delay=2	(IntOpt) How often to retry in seconds when no cells are available.
scheduler_topic=scheduler	(StrOpt) the topic scheduler nodes listen on
scheduler_weight_classes=nova.cells.weights.all_weighers	(ListOpt) Weigher classes the cells scheduler should use. An entry of "nova.cells.weights.all_weighers"maps to all cell weighers included with nova.
scheduler_weight_classes=nova.scheduler.weights.all_weighers	(ListOpt) Which weight class names to use for weighing hosts

Cells

Cells functionality allows you to scale an OpenStack Compute cloud in a more distributed fashion without having to use complicated technologies like database and message queue clustering. It is intended to support very large deployments.

When this functionality is enabled, the hosts in an OpenStack Compute cloud are partitioned into groups called cells. Cells are configured as a tree. The top-level cell should have a host that runs a `nova-api` service, but no `nova-compute` services. Each child cell should run all of the typical `nova-*` services in a regular Compute cloud except for `nova-api`. You can think of cells as a normal Compute deployment in that each cell has its own database server and message queue broker.

The `nova-cells` service handles communication between cells and selects cells for new instances. This service is required for every cell. Communication between cells is pluggable, and currently the only option is communication through RPC.

Cells scheduling is separate from host scheduling. `nova-cells` first picks a cell (now randomly, but future releases plan to add filtering/weighing functionality, and decisions will be based on broadcasts of capacity/capabilities). Once a cell is selected and the new build request reaches its `nova-cells` service, it is sent over to the host scheduler in that cell and the build proceeds as it would have without cells.

Warning

Cell functionality is currently considered experimental.

Cell configuration options

Cells are disabled by default. All cell-related configuration options go under a `[cells]` section in `nova.conf`. The following cell-related options are currently supported:

`enable`	Set this is `True` to turn on cell functionality, which is off by default.
`name`	Name of the current cell. This must be unique for each cell.
`capabilities`	List of arbitrary *key=value* pairs defining capabilities of the current cell. Values include `hypervisor=xenserver;kvm,os=linux;windows`.
`call_timeout`	How long in seconds to wait for replies from calls between cells.
scheduler_filter_classes	Filter classes that the cells scheduler should use. By default, uses "`nova.cells.filters.all_filters`" to map to all cells filters included with Compute.
scheduler_weight_classes	Weight classes the cells scheduler should use. By default, uses "`nova.cells.weights.all_weighers`" to map to all cells weight algorithms (weighers) included with Compute.
ram_weight_multiplier	Multiplier used for weighing ram. Negative numbers mean you want Compute to stack VMs on one host instead of spreading out new VMs to more hosts in the cell. Default value is 10.0.

Configure the API (top-level) cell

The compute API class must be changed in the API cell so that requests can be proxied through nova-cells down to the correct cell properly. Add the following to `nova.conf` in the API cell:

```
[DEFAULT]
compute_api_class=nova.compute.cells_api.ComputeCellsAPI
...

[cells]
enable=True
name=api
```

Configure the child cells

Add the following to `nova.conf` in the child cells, replacing *`cell1`* with the name of each cell:

```
[DEFAULT]
# Disable quota checking in child cells.  Let API cell do it exclusively.
quota_driver=nova.quota.NoopQuotaDriver

[cells]
enable=True
name=cell1
```

Configure the database in each cell

Before bringing the services online, the database in each cell needs to be configured with information about related cells. In particular, the API cell needs to know about its immediate children, and the child cells need to know about their immediate agents. The information needed is the RabbitMQ server credentials for the particular cell.

Use the **nova-manage cell create** command to add this information to the database in each cell:

```
$ nova-manage cell create -h
Options:
  -h, --help            show this help message and exit
  --name=<name>         Name for the new cell
  --cell_type=<parent|child>
                        Whether the cell is a parent or child
  --username=<username>
                        Username for the message broker in this cell
  --password=<password>
                        Password for the message broker in this cell
  --hostname=<hostname>
                        Address of the message broker in this cell
  --port=<number>       Port number of the message broker in this cell
  --virtual_host=<virtual_host>
                        The virtual host of the message broker in this cell
  --woffset=<float>
                        (weight offset) It might be used by some cell
 scheduling code in the future
  --wscale=<float>
```

```
                        (weight scale) It might be used by some cell
 scheduling code in the future
```

As an example, assume we have an API cell named `api` and a child cell named `cell1`. Within the api cell, we have the following RabbitMQ server info:

```
rabbit_host=10.0.0.10
rabbit_port=5672
rabbit_username=api_user
rabbit_password=api_passwd
rabbit_virtual_host=api_vhost
```

And in the child cell named `cell1` we have the following RabbitMQ server info:

```
rabbit_host=10.0.1.10
rabbit_port=5673
rabbit_username=cell1_user
rabbit_password=cell1_passwd
rabbit_virtual_host=cell1_vhost
```

We would run this in the API cell, as root.

```
# nova-manage cell create --name=cell1 --cell_type=child --username=cell1_user
 --password=cell1_passwd --hostname=10.0.1.10 --port=5673 --virtual_host=
cell1_vhost --woffset=1.0 --wscale=1.0
```

Repeat the above for all child cells.

In the child cell, we would run the following, as root:

```
# nova-manage cell create --name=api --cell_type=parent --username=api1_user
 --password=api1_passwd --hostname=10.0.0.10 --port=5672 --virtual_host=
api_vhost --woffset=1.0 --wscale=1.0
```

Table 2.30. Description of configuration options for cells

Configuration option=Default value	Description
call_timeout=60	(IntOpt) Seconds to wait for response from a call to a cell.
capabilities=hypervisor=xenserver;kvm,os=linux;windows	(ListOpt) Key/Multi-value list with the capabilities of the cell
cell_type=None	(StrOpt) Type of cell: api or compute
cells_config=None	(StrOpt) Configuration file from which to read cells configuration. If given, overrides reading cells from the database.
driver=nova.virt.baremetal.pxe.PXE	(StrOpt) Baremetal driver back-end (pxe or tilera)
driver=nova.cells.rpc_driver.CellsRPCDriver	(StrOpt) Cells communication driver to use
enable=False	(BoolOpt) Enable cell functionality
instance_update_num_instances=1	(IntOpt) Number of instances to update per periodic task run
instance_updated_at_threshold=3600	(IntOpt) Number of seconds after an instance was updated or deleted to continue to update cells
manager=nova.cells.manager.CellsManager	(StrOpt) Manager for cells
manager=nova.conductor.manager.ConductorManager	(StrOpt) full class name for the Manager for conductor
max_hop_count=10	(IntOpt) Maximum number of hops for cells routing.
mute_child_interval=300	(IntOpt) Number of seconds after which a lack of capability and capacity updates signals the child cell is to be treated as a mute.

Configuration option=Default value	Description
mute_weight_multiplier=-10.0	(FloatOpt) Multiplier used to weigh mute children. (The value should be negative.)
mute_weight_value=1000.0	(FloatOpt) Weight value assigned to mute children. (The value should be positive.)
name=nova	(StrOpt) name of this cell
reserve_percent=10.0	(FloatOpt) Percentage of cell capacity to hold in reserve. Affects both memory and disk utilization
topic=cells	(StrOpt) the topic cells nodes listen on
topic=conductor	(StrOpt) the topic conductor nodes listen on

Cell scheduling configuration

To determine the best cell for launching a new instance, Compute uses a set of filters and weights configured in `/etc/nova/nova.conf`. The following options are available to prioritize cells for scheduling:

- `scheduler_filter_classes` - Specifies the list of filter classes. By default `nova.cells.weights.all_filters` is specified, which maps to all cells filters included with Compute (see the section called "Filters" [110].

- `scheduler_weight_classes` - Specifies the list of weight classes. By default `nova.cells.weights.all_weighers` is specified, which maps to all cell weight algorithms (weighers) included with Compute. The following modules are available:

  - `mute_child`: Downgrades the likelihood of child cells being chosen for scheduling requests, which haven't sent capacity or capability updates in a while. Options include `mute_weight_multiplier` (multiplier for mute children; value should be negative) and `mute_weight_value` (assigned to mute children; should be a positive value).

  - `ram_by_instance_type`: Select cells with the most RAM capacity for the instance type being requested. Because higher weights win, Compute returns the number of available units for the instance type requested. The `ram_weight_multiplier` option defaults to 10.0 that adds to the weight by a factor of 10. Use a negative number to stack VMs on one host instead of spreading out new VMs to more hosts in the cell.

  - `weight_offset`: Allows modifying the database to weight a particular cell. You can use this when you want to disable a cell (for example, '0'), or to set a default cell by making its weight_offset very high (for example, '999999999999999'). The highest weight will be the first cell to be scheduled for launching an instance.

Additionally, the following options are available for the cell scheduler:

- `scheduler_retries` - Specifies how many times the scheduler tries to launch a new instance when no cells are available (default=10).

- `scheduler_retry_delay` - Specifies the delay (in seconds) between retries (default=2).

As an admin user, you can also add a filter that directs builds to a particular cell. The `policy.json` file must have a line with

`"cells_scheduler_filter:TargetCellFilter" : "is_admin:True"` to let an admin user specify a scheduler hint to direct a build to a particular cell.

Optional cell configuration

Cells currently keeps all inter-cell communication data, including user names and passwords, in the database. This is undesirable and unnecessary since cells data isn't updated very frequently. Instead, create a JSON file to input cells data specified via a `[cells]cells_config` option. When specified, the database is no longer consulted when reloading cells data. The file will need the columns present in the Cell model (excluding common database fields and the `id` column). The queue connection information must be specified through a `transport_url` field, instead of `username`, `password`, and so on. The `transport_url` has the following form:

```
rabbit://<username>:<password>@<hostname>:<port>/<virtual_host>
```

The scheme can be either `qpid` or `rabbit`, as shown previously. The following sample shows this optional configuration:

```
{
    "parent": {
        "name": "parent",
        "api_url": "http://api.example.com:8774",
        "transport_url": "rabbit://rabbit.example.com",
        "weight_offset": 0.0,
        "weight_scale": 1.0,
        "is_parent": true,
    },
    "cell1": {
        "name": "cell1",
        "api_url": "http://api.example.com:8774",
        "transport_url": "rabbit://rabbit1.example.com",
        "weight_offset": 0.0,
        "weight_scale": 1.0,
        "is_parent": false,
    },
    "cell2": {
        "name": "cell2",
        "api_url": "http://api.example.com:8774",
        "transport_url": "rabbit://rabbit2.example.com",
        "weight_offset": 0.0,
        "weight_scale": 1.0,
        "is_parent": false,
    }
}
```

Conductor

The `nova-conductor` service enables OpenStack to function without compute nodes accessing the database. Conceptually, it implements a new layer on top of `nova-compute`. It should not be deployed on compute nodes, or else the security benefits of removing database access from `nova-compute` are negated. Just like other nova services such as `nova-api` or nova-scheduler, it can be scaled horizontally. You can run multiple instances of `nova-conductor` on different machines as needed for scaling purposes.

In the Grizzly release, the methods exposed by `nova-conductor` are relatively simple methods used by `nova-compute` to offload its database operations. Places where `nova-compute` previously performed database access are now talking to `nova-conductor`. However, we have plans in the medium to long term to move more and more of what is currently in `nova-compute` up to the `nova-conductor` layer. The compute service will start to look like a less intelligent slave service to `nova-conductor`. The conductor service will implement long running complex operations, ensuring forward progress and graceful error handling. This will be especially beneficial for operations that cross multiple compute nodes, such as migrations or resizes.

Table 2.31. Description of configuration options for conductor

Configuration option=Default value	Description
manager=nova.cells.manager.CellsManager	(StrOpt) Manager for cells
manager=nova.conductor.manager.ConductorManager	(StrOpt) full class name for the Manager for conductor
migrate_max_retries=-1	(IntOpt) Number of times to retry live-migration before failing. If == -1, try until out of hosts. If == 0, only try once, no retries.
topic=cells	(StrOpt) the topic cells nodes listen on
topic=conductor	(StrOpt) the topic conductor nodes listen on
use_local=False	(BoolOpt) Perform nova-conductor operations locally
workers=None	(IntOpt) Number of workers for OpenStack Conductor service

Security hardening

OpenStack Compute can be integrated with various third-party technologies to increase security. For more information, see the *OpenStack Security Guide*.

Trusted compute pools

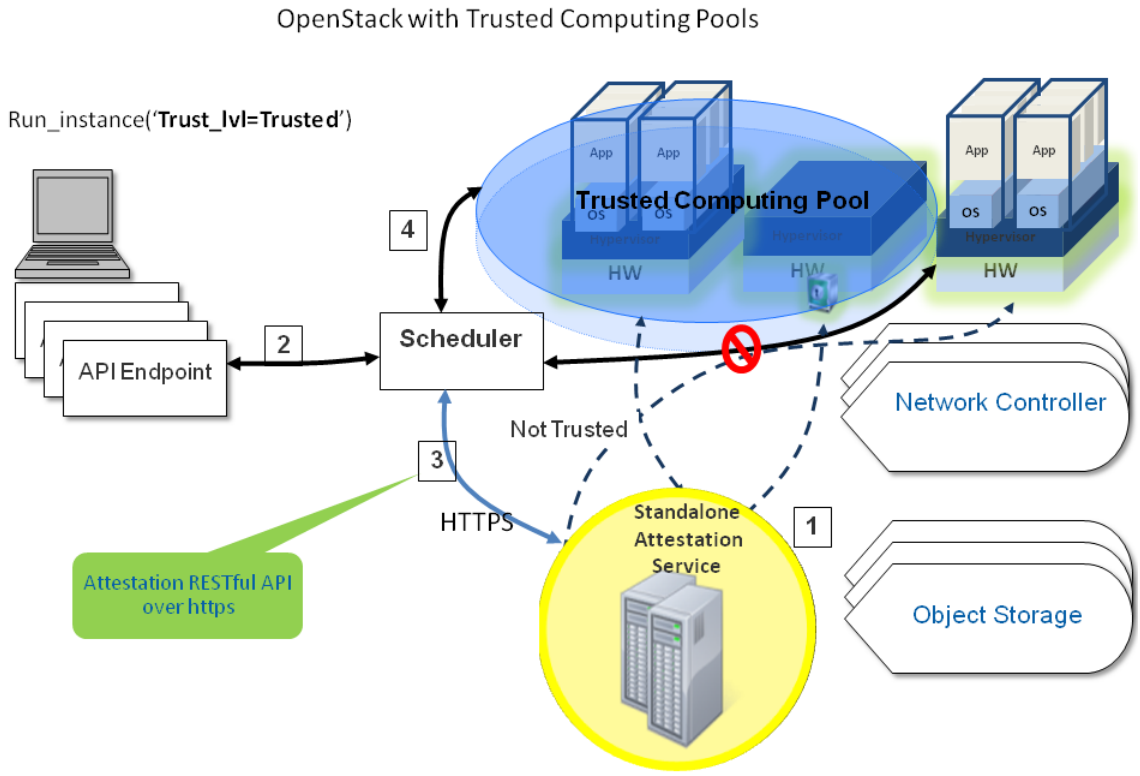
Trusted compute pools enable administrators to designate a group of compute hosts as *trusted*. These hosts use hardware-based security features, such as the Intel Trusted Execution Technology (TXT), to provide an additional level of security. Combined with an external stand-alone web-based remote attestation server, cloud providers can ensure that the compute node runs only software with verified measurements and can ensure a secure cloud stack.

Through the trusted compute pools, cloud subscribers can request services to run on verified compute nodes.

The remote attestation server performs node verification as follows:

1. Compute nodes boot with Intel TXT technology enabled.
2. The compute node BIOS, hypervisor, and OS are measured.
3. Measured data is sent to the attestation server when challenged by attestation server.
4. The attestation server verifies those measurements against a good and known database to determine nodes' trustworthiness.

A description of how to set up an attestation service is beyond the scope of this document. For an open source project that you can use to implement an attestation service, see the Open Attestation project.



Configure Compute to use trusted compute pools

1. Configure the Compute service with the connection information for the attestation service.

   Specify these connection options in the `trusted_computing` section in the `nova.conf` configuration file:

server	Host name or IP address of the host that runs the attestation service
port	HTTPS port for the attestation service
server_ca_file	Certificate file used to verify the attestation server's identity.
api_url	The attestation service URL path.
auth_blob	An authentication blob, which is required by the attestation service.

2. To enable scheduling support for trusted compute pools, add the following lines to the DEFAULT and `trusted_computing` sections in the `/etc/nova/nova.conf`

file. Edit the details in the `trusted_computing` section based on the details of your attestation service:

```
[DEFAULT]
compute_scheduler_driver=nova.scheduler.filter_scheduler.FilterScheduler
scheduler_available_filters=nova.scheduler.filters.all_filters
scheduler_default_filters=AvailabilityZoneFilter,RamFilter,ComputeFilter,
TrustedFilter

[trusted_computing]
server=10.1.71.206
port=8443
server_ca_file=/etc/nova/ssl.10.1.71.206.crt
# If using OAT v1.5, use this api_url:
api_url=/AttestationService/resources
# If using OAT pre-v1.5, use this api_url:
#api_url=/OpenAttestationWebServices/V1.0
auth_blob=i-am-openstack
```

3. Restart the `nova-compute` and `nova-scheduler` services.

Configuration reference

Table 2.32. Description of configuration options for trustedcomputing

Configuration option=Default value	Description
attestation_api_url=/OpenAttestationWebServices/V1.0	(StrOpt) attestation web API URL
attestation_auth_blob=None	(StrOpt) attestation authorization blob - must change
attestation_auth_timeout=60	(IntOpt) Attestation status cache valid period length
attestation_port=8443	(StrOpt) attestation server port
attestation_server=None	(StrOpt) attestation server http
attestation_server_ca_file=None	(StrOpt) attestation server Cert file for Identity verification

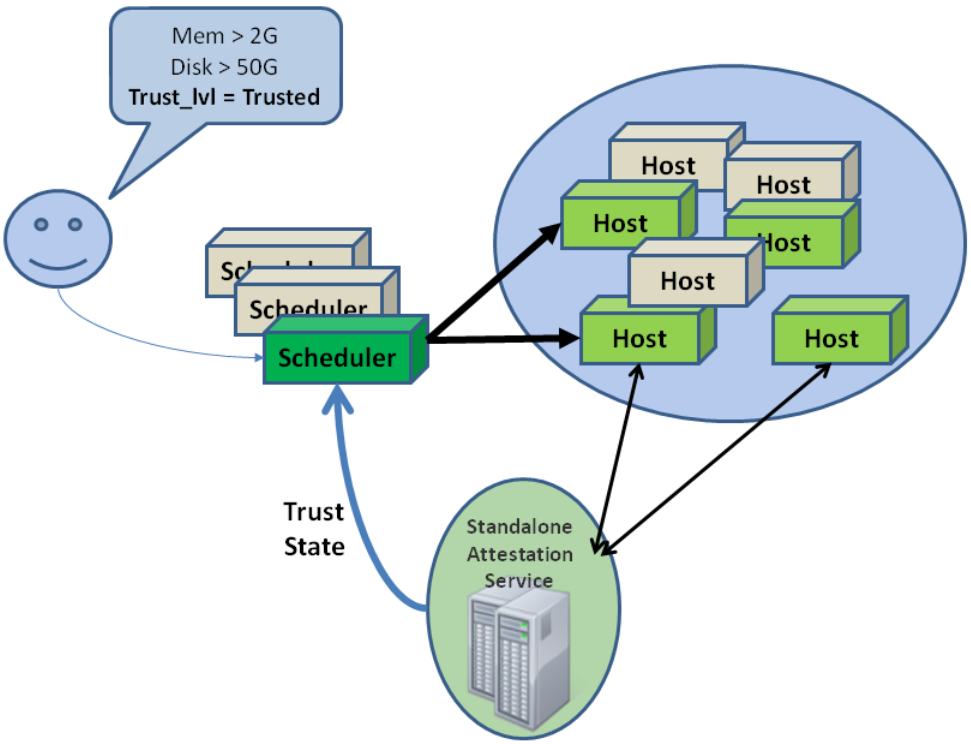
Specify trusted flavors

You must configure one or more flavors as *trusted*. Users can request trusted nodes by specifying a trusted flavor when they boot an instance.

Use the **nova flavor-key set** command to set a flavor as trusted. For example, to set the `m1.tiny` flavor as trusted:

```
# nova flavor-key m1.tiny set trust:trusted_host trusted
```

To request that their instances run on a trusted host, users can specify a trusted flavor on the **nova boot** command:



Compute configuration files: nova.conf

File format for nova.conf

Overview

The Compute service supports a large number of configuration options. These options are specified in the `/etc/nova/nova.conf` configuration file.

The configuration file is in INI file format, with options specified as `key=value` pairs, grouped into sections. Almost all configuration options are in the `DEFAULT` section. For example:

```
[DEFAULT]
debug=true
verbose=true

[trusted_computing]
server=10.3.4.2
```

Types of configuration options

Each configuration option has an associated type that indicates which values can be set. The supported option types are:

BoolOpt — Boolean option. Value must be either `true` or `false` . Example:

```
debug=false
```

StrOpt — String option. Value is an arbitrary string. Example:

```
my_ip=10.0.0.1
```

IntOption — Integer option. Value must be an integer. Example:

```
glance_port=9292
```

MultiStrOpt — String option. Same as StrOpt, except that it can be declared multiple times to indicate multiple values. Example:

```
ldap_dns_servers=dns1.example.org
ldap_dns_servers=dns2.example.org
```

ListOpt — List option. Value is a list of arbitrary strings separated by commas. Example:

```
enabled_apis=ec2,osapi_compute,metadata
```

FloatOpt — Floating-point option. Value must be a floating-point number. Example:

```
ram_allocation_ratio=1.5
```

Important

Do not specify quotes around Nova options.

Sections

Configuration options are grouped by section. The Compute configuration file supports the following sections.

`[DEFAULT]` — Contains most configuration options. If the documentation for a configuration option does not specify its section, assume that it appears in this section.

`[cells]` — Use options in this section to configure cells functionality. For details, see the Cells section (../config-reference/content/section_compute-cells.html) in the *OpenStack Configuration Reference*.

`[baremetal]` — Use options in this section to configure the baremetal hypervisor driver.

`[conductor]` — Use options in this section to configure the `nova-conductor` service.

`[trusted_computing]` — Use options in this section to configure the trusted computing pools functionality and how to connect to a remote attestation service.

Variable substitution

The configuration file supports variable substitution. After you set a configuration option, it can be referenced in later configuration values when you precede it with `$`. This example defines `my_ip` and then uses `$my_ip` as a variable:

```
my_ip=10.2.3.4
glance_host=$my_ip
metadata_host=$my_ip
```

If you need a value to contain the $ symbol, escape it with $$. For example, if your LDAP DNS password was `$xkj432`, specify it, as follows:

```
ldap_dns_password=$$xkj432
```

The Compute code uses the Python `string.Template.safe_substitute()` method to implement variable substitution. For more details on how variable substitution is resolved, see http://docs.python.org/2/library/string.html#template-strings and http://www.python.org/dev/peps/pep-0292/.

Whitespace

To include whitespace in a configuration value, use a quoted string. For example:

```
ldap_dns_passsword='a password with spaces'
```

Define an alternate location for nova.conf

All `nova-*` services and the **nova-manage** command-line client load the configuration file. To define an alternate location for the configuration file, pass the *`--config-file /path/to/nova.conf`* parameter when you start a `nova-*` service or call a **nova-manage** command.

Configuration options

For a complete list of all available configuration options for each OpenStack Compute service, run bin/nova-<servicename> –help.

Table 2.33. Description of configuration options for api

Configuration option=Default value	Description
enable_new_services=True	(BoolOpt) Services to be added to the available pool on create
enabled_apis=ec2,osapi_compute,metadata	(ListOpt) a list of APIs to enable by default
enabled_ssl_apis=	(ListOpt) a list of APIs with enabled SSL
instance_name_template=instance-%08x	(StrOpt) Template string to be used to generate instance names
multi_instance_display_name_template=%(name)s-%(uuid)s	(StrOpt) When creating multiple instances with a single request using the os-multiple-create API extension, this template will be used to build the display name for each instance. The benefit is that the instances end up with different hostnames. To restore legacy behavior of every instance having the same name, set this option to

Configuration option=Default value	Description
	"%(name)s". Valid keys for the template are: name, uuid, count.
non_inheritable_image_properties=cache_in_nova,bittorrent	(ListOpt) These are image properties which a snapshot should not inherit from an instance
null_kernel=nokernel	(StrOpt) kernel image that indicates not to use a kernel, but to use a raw disk image instead
osapi_compute_ext_list=	(ListOpt) Specify list of extensions to load when using osapi_compute_extension option with nova.api.openstack.compute.contrib.select_extensions
osapi_compute_extension=['nova.api.openstack.compute.contrib.standard_extensions']	(MultiStrOpt) osapi compute extension to load
osapi_compute_link_prefix=None	(StrOpt) Base URL that will be presented to users in links to the OpenStack Compute API
osapi_compute_listen=0.0.0.0	(StrOpt) IP address for OpenStack API to listen
osapi_compute_listen_port=8774	(IntOpt) list port for osapi compute
osapi_compute_workers=None	(IntOpt) Number of workers for OpenStack API service
osapi_hide_server_address_states=building	(ListOpt) List of instance states that should hide network info
servicegroup_driver=db	(StrOpt) The driver for servicegroup service (valid options are: db, zk, mc)
snapshot_name_template=snapshot-%s	(StrOpt) Template string to be used to generate snapshot names
use_forwarded_for=False	(BoolOpt) Treat X-Forwarded-For as the canonical remote address. Only enable this if you have a sanitizing proxy.
use_tpool=False	(BoolOpt) Enable the experimental use of thread pooling for all DB API calls

Table 2.34. Description of configuration options for authentication

Configuration option=Default value	Description
api_rate_limit=False	(BoolOpt) whether to use per-user rate limiting for the api.
auth_strategy=noauth	(StrOpt) The strategy to use for auth: noauth or keystone.

Table 2.35. Description of configuration options for availabilityzones

Configuration option=Default value	Description
default_availability_zone=nova	(StrOpt) default compute node availability_zone
default_schedule_zone=None	(StrOpt) availability zone to use when user doesn't specify one
internal_service_availability_zone=internal	(StrOpt) availability_zone to show internal services under

Table 2.36. Description of configuration options for baremetal

Configuration option=Default value	Description
db_backend=sqlalchemy	(StrOpt) The backend to use for bare-metal database
deploy_kernel=None	(StrOpt) Default kernel image ID used in deployment phase
deploy_ramdisk=None	(StrOpt) Default ramdisk image ID used in deployment phase
driver=nova.virt.baremetal.pxe.PXE	(StrOpt) Baremetal driver back-end (pxe or tilera)
driver=nova.cells.rpc_driver.CellsRPCDriver	(StrOpt) Cells communication driver to use
instance_type_extra_specs=	(ListOpt) a list of additional capabilities corresponding to instance_type_extra_specs for this compute host to

Configuration option=Default value	Description
	advertise. Valid entries are name=value, pairs For example, "key1:val1, key2:val2"
ipmi_power_retry=5	(IntOpt) maximal number of retries for IPMI operations
net_config_template=$pybasedir/nova/virt/baremetal/net-dhcp.ubuntu.template	(StrOpt) Template file for injected network config
power_manager=nova.virt.baremetal.ipmi.IPMI	(StrOpt) Baremetal power management method
pxe_append_params=None	(StrOpt) additional append parameters for baremetal PXE boot
pxe_bootfile_name=pxelinux.0	(StrOpt) This gets passed to Neutron as the bootfile dhcp parameter when the dhcp_options_enabled is set.
pxe_config_template=$pybasedir/nova/virt/baremetal/pxe_config.template	(StrOpt) Template file for PXE configuration
pxe_deploy_timeout=0	(IntOpt) Timeout for PXE deployments. Default: 0 (unlimited)
pxe_network_config=False	(BoolOpt) If set, pass the network configuration details to the initramfs via cmdline.
sql_connection=sqlite:///$state_path/baremetal_$sqlite_db	(StrOpt) The SQLAlchemy connection string used to connect to the bare-metal database
terminal=shellinaboxd	(StrOpt) path to baremetal terminal program
terminal_cert_dir=None	(StrOpt) path to baremetal terminal SSL cert(PEM)
terminal_pid_dir=$state_path/baremetal/console	(StrOpt) path to directory stores pidfiles of baremetal_terminal
tftp_root=/tftpboot	(StrOpt) Baremetal compute node's tftp root path
use_unsafe_iscsi=False	(BoolOpt) Do not set this out of dev/test environments. If a node does not have a fixed PXE IP address, volumes are exported with globally opened ACL
vif_driver=nova.virt.baremetal.vif_driver.BareMetalVIFDriver	(StrOpt) Baremetal VIF driver.
virtual_power_host_key=None	(StrOpt) ssh key for virtual power host_user
virtual_power_host_pass=	(StrOpt) password for virtual power host_user
virtual_power_host_user=	(StrOpt) user to execute virtual power commands as
virtual_power_ssh_host=	(StrOpt) ip or name to virtual power host
virtual_power_ssh_port=22	(IntOpt) Port to use for ssh to virtual power host
virtual_power_type=virsh	(StrOpt) base command to use for virtual power(vbox,virsh)

Table 2.37. Description of configuration options for ca

Configuration option=Default value	Description
ca_file=cacert.pem	(StrOpt) Filename of root CA
ca_file=None	(StrOpt) CA certificate file to use to verify connecting clients
ca_path=$state_path/CA	(StrOpt) Where we keep our root CA
cert_file=None	(StrOpt) Certificate file to use when starting the server securely
cert_manager=nova.cert.manager.CertManager	(StrOpt) full class name for the Manager for cert
cert_topic=cert	(StrOpt) the topic cert nodes listen on
crl_file=crl.pem	(StrOpt) Filename of root Certificate Revocation List
key_file=private/cakey.pem	(StrOpt) Filename of private key
key_file=None	(StrOpt) Private key file to use when starting the server securely

Configuration option=Default value	Description
keys_path=$state_path/keys	(StrOpt) Where we keep our keys
project_cert_subject=/C=US/ST=California/O=OpenStack/OU=NovaDev/CN=project-ca-%.16s-%s	(StrOpt) Subject for certificate for projects, %s for project, timestamp
use_project_ca=False	(BoolOpt) Should we use a CA for each project?
user_cert_subject=/C=US/ST=California/O=OpenStack/OU=NovaDev/CN=%.16s-%.16s-%s	(StrOpt) Subject for certificate for users, %s for project, user, timestamp

Table 2.38. Description of configuration options for cells

Configuration option=Default value	Description
call_timeout=60	(IntOpt) Seconds to wait for response from a call to a cell.
capabilities=hypervisor=xenserver;kvm,os=linux;windows	(ListOpt) Key/Multi-value list with the capabilities of the cell
cell_type=None	(StrOpt) Type of cell: api or compute
cells_config=None	(StrOpt) Configuration file from which to read cells configuration. If given, overrides reading cells from the database.
driver=nova.virt.baremetal.pxe.PXE	(StrOpt) Baremetal driver back-end (pxe or tilera)
driver=nova.cells.rpc_driver.CellsRPCDriver	(StrOpt) Cells communication driver to use
enable=False	(BoolOpt) Enable cell functionality
instance_update_num_instances=1	(IntOpt) Number of instances to update per periodic task run
instance_updated_at_threshold=3600	(IntOpt) Number of seconds after an instance was updated or deleted to continue to update cells
manager=nova.cells.manager.CellsManager	(StrOpt) Manager for cells
manager=nova.conductor.manager.ConductorManager	(StrOpt) full class name for the Manager for conductor
max_hop_count=10	(IntOpt) Maximum number of hops for cells routing.
mute_child_interval=300	(IntOpt) Number of seconds after which a lack of capability and capacity updates signals the child cell is to be treated as a mute.
mute_weight_multiplier=-10.0	(FloatOpt) Multiplier used to weigh mute children. (The value should be negative.)
mute_weight_value=1000.0	(FloatOpt) Weight value assigned to mute children. (The value should be positive.)
name=nova	(StrOpt) name of this cell
reserve_percent=10.0	(FloatOpt) Percentage of cell capacity to hold in reserve. Affects both memory and disk utilization
topic=cells	(StrOpt) the topic cells nodes listen on
topic=conductor	(StrOpt) the topic conductor nodes listen on

Table 2.39. Description of configuration options for common

Configuration option=Default value	Description
bindir=/usr/local/bin	(StrOpt) Directory where nova binaries are installed
compute_topic=compute	(StrOpt) the topic compute nodes listen on
console_topic=console	(StrOpt) the topic console proxy nodes listen on
consoleauth_topic=consoleauth	(StrOpt) the topic console auth proxy nodes listen on
disable_process_locking=False	(BoolOpt) Whether to disable inter-process locks
host=docwork	(StrOpt) Name of this node. This can be an opaque identifier. It is not necessarily a hostname, FQDN, or IP

Configuration option=Default value	Description
	address. However, the node name must be valid within an AMQP key, and if using ZeroMQ, a valid hostname, FQDN, or IP address
host=127.0.0.1	(StrOpt) Host to locate redis
lock_path=None	(StrOpt) Directory to use for lock files.
memcached_servers=None	(ListOpt) Memcached servers or None for in process cache.
my_ip=192.168.122.99	(StrOpt) ip address of this host
notification_driver=[]	(MultiStrOpt) Driver or drivers to handle sending notifications
notification_topics=notifications	(ListOpt) AMQP topic used for OpenStack notifications
notify_api_faults=False	(BoolOpt) If set, send api.fault notifications on caught exceptions in the API service.
notify_on_state_change=None	(StrOpt) If set, send compute.instance.update notifications on instance state changes. Valid values are None for no notifications, "vm_state" for notifications on VM state changes, or "vm_and_task_state" for notifications on VM and task state changes.
pybasedir=/home/docwork/openstack-manuals-new/tools/autogenerate-config-docs/nova	(StrOpt) Directory where the nova python module is installed
report_interval=10	(IntOpt) seconds between nodes reporting state to datastore
rootwrap_config=/etc/nova/rootwrap.conf	(StrOpt) Path to the rootwrap configuration file to use for running commands as root
service_down_time=60	(IntOpt) maximum time since last check-in for up service
state_path=$pybasedir	(StrOpt) Top-level directory for maintaining nova's state
tempdir=None	(StrOpt) Explicitly specify the temporary working directory

Table 2.40. Description of configuration options for compute

Configuration option=Default value	Description
base_dir_name=_base	(StrOpt) Where cached images are stored under $instances_path.This is NOT the full path - just a folder name.For per-compute-host cached images, set to _base_$my_ip
checksum_interval_seconds=3600	(IntOpt) How frequently to checksum base images
compute_api_class=nova.compute.api.API	(StrOpt) The full class name of the compute API class to use (deprecated)
compute_driver=None	(StrOpt) Driver to use for controlling virtualization. Options include: libvirt.LibvirtDriver, xenapi.XenAPIDriver, fake.FakeDriver, baremetal.BareMetalDriver, vmwareapi.VMwareESXDriver, vmwareapi.VMwareVCDriver
compute_manager=nova.compute.manager.ComputeManager	(StrOpt) full class name for the Manager for compute
compute_stats_class=nova.compute.stats.Stats	(StrOpt) Class that will manage stats for the local compute host
console_host=docwork	(StrOpt) Console proxy host to use to connect to instances on this host.
console_manager=nova.console.manager.ConsoleProxyManager	(StrOpt) full class name for the Manager for console proxy
default_flavor=m1.small	(StrOpt) default flavor to use for the EC2 API only. The Nova API does not support a default flavor.
default_notification_level=INFO	(StrOpt) Default notification level for outgoing notifications
default_publisher_id=None	(StrOpt) Default publisher_id for outgoing notifications

Configuration option=Default value	Description
enable_instance_password=True	(BoolOpt) Allows use of instance password during server creation
heal_instance_info_cache_interval=60	(IntOpt) Number of seconds between instance info_cache self healing updates
host_state_interval=120	(IntOpt) Interval in seconds for querying the host status
image_cache_manager_interval=2400	(IntOpt) Number of seconds to wait between runs of the image cache manager
image_info_filename_pattern=$instances_path/$base_dir_name/%(image)s.info	(StrOpt) Allows image information files to be stored in non-standard locations
instance_build_timeout=0	(IntOpt) Amount of time in seconds an instance can be in BUILD before going into ERROR status.Set to 0 to disable.
instance_delete_interval=300	(IntOpt) Interval in seconds for retrying failed instance file deletes
instance_usage_audit=False	(BoolOpt) Generate periodic compute.instance.exists notifications
instance_usage_audit_period=month	(StrOpt) time period to generate instance usages for. Time period must be hour, day, month or year
instances_path=$state_path/instances	(StrOpt) where instances are stored on disk
maximum_instance_delete_attempts=5	(IntOpt) The number of times to attempt to reap an instance's files.
reboot_timeout=0	(IntOpt) Automatically hard reboot an instance if it has been stuck in a rebooting state longer than N seconds. Set to 0 to disable.
reclaim_instance_interval=0	(IntOpt) Interval in seconds for reclaiming deleted instances
resize_confirm_window=0	(IntOpt) Automatically confirm resizes after N seconds. Set to 0 to disable.
resume_guests_state_on_host_boot=False	(BoolOpt) Whether to start guests that were running before the host rebooted
running_deleted_instance_action=log	(StrOpt) Action to take if a running deleted instance is detected.Valid options are 'noop', 'log' and 'reap'. Set to 'noop' to disable.
running_deleted_instance_poll_interval=1800	(IntOpt) Number of seconds to wait between runs of the cleanup task.
running_deleted_instance_timeout=0	(IntOpt) Number of seconds after being deleted when a running instance should be considered eligible for cleanup.
shelved_offload_time=0	(IntOpt) Time in seconds before a shelved instance is eligible for removing from a host. -1 never offload, 0 offload when shelved
shelved_poll_interval=3600	(IntOpt) Interval in seconds for polling shelved instances to offload
sync_power_state_interval=600	(IntOpt) interval to sync power states between the database and the hypervisor

Table 2.41. Description of configuration options for conductor

Configuration option=Default value	Description
manager=nova.cells.manager.CellsManager	(StrOpt) Manager for cells
manager=nova.conductor.manager.ConductorManager	(StrOpt) full class name for the Manager for conductor
migrate_max_retries=-1	(IntOpt) Number of times to retry live-migration before failing. If == -1, try until out of hosts. If == 0, only try once, no retries.
topic=cells	(StrOpt) the topic cells nodes listen on

Configuration option=Default value	Description
topic=conductor	(StrOpt) the topic conductor nodes listen on
use_local=False	(BoolOpt) Perform nova-conductor operations locally
workers=None	(IntOpt) Number of workers for OpenStack Conductor service

Table 2.42. Description of configuration options for configdrive

Configuration option=Default value	Description
config_drive_cdrom=False	(BoolOpt) Attaches the Config Drive image as a cdrom drive instead of a disk drive
config_drive_format=iso9660	(StrOpt) Config drive format. One of iso9660 (default) or vfat
config_drive_inject_password=False	(BoolOpt) Sets the admin password in the config drive image
config_drive_skip_versions=1.0 2007-01-19 2007-03-01 2007-08-29 2007-10-10 2007-12-15 2008-02-01 2008-09-01	(StrOpt) List of metadata versions to skip placing into the config drive
config_drive_tempdir=None	(StrOpt) Where to put temporary files associated with config drive creation
force_config_drive=None	(StrOpt) Set to force injection to take place on a config drive (if set, valid options are: always)
mkisofs_cmd=genisoimage	(StrOpt) Name and optionally path of the tool used for ISO image creation

Table 2.43. Description of configuration options for console

Configuration option=Default value	Description
console_public_hostname=docwork	(StrOpt) Publicly visible name for this console host
console_token_ttl=600	(IntOpt) How many seconds before deleting tokens
consoleauth_manager=nova.consoleauth.manager.ConsoleAuthManager	(StrOpt) Manager for console auth

Table 2.44. Description of configuration options for db

Configuration option=Default value	Description
backend=sqlalchemy	(StrOpt) The backend to use for db
connection_trace=False	(BoolOpt) Add python stack traces to SQL as comment strings
connection=sqlite:////home/docwork/openstack-manuals-new/tools/autogenerate-config-docs/nova/nova/openstack/common/db/$sqlite_db	(StrOpt) The SQLAlchemy connection string used to connect to the database
connection_debug=0	(IntOpt) Verbosity of SQL debugging information. 0=None, 100=Everything
db_backend=sqlalchemy	(StrOpt) The backend to use for bare-metal database
db_check_interval=60	(IntOpt) Seconds between getting fresh cell info from db.
db_driver=nova.db	(StrOpt) driver to use for database access
idle_timeout=3600	(IntOpt) timeout before idle sql connections are reaped
max_pool_size=None	(IntOpt) Maximum number of SQL connections to keep open in a pool
max_overflow=None	(IntOpt) If set, use this value for max_overflow with sqlalchemy
max_retries=10	(IntOpt) maximum db connection retries during startup. (setting -1 implies an infinite retry count)

Configuration option=Default value	Description
min_pool_size=1	(IntOpt) Minimum number of SQL connections to keep open in a pool
pool_timeout=None	(IntOpt) If set, use this value for pool_timeout with sqlalchemy
retry_interval=10	(IntOpt) interval between retries of opening a sql connection
slave_connection=	(StrOpt) The SQLAlchemy connection string used to connect to the slave database
sql_connection=sqlite:///$state_path/baremetal_$sqlite_db	(StrOpt) The SQLAlchemy connection string used to connect to the bare-metal database
sqlite_db=nova.sqlite	(StrOpt) the filename to use with sqlite
sqlite_synchronous=True	(BoolOpt) If true, use synchronous mode for sqlite

Table 2.45. Description of configuration options for ec2

Configuration option=Default value	Description
ec2_dmz_host=$my_ip	(StrOpt) the internal ip of the ec2 api server
ec2_host=$my_ip	(StrOpt) the ip of the ec2 api server
ec2_listen=0.0.0.0	(StrOpt) IP address for EC2 API to listen
ec2_listen_port=8773	(IntOpt) port for ec2 api to listen
ec2_path=/services/Cloud	(StrOpt) the path prefix used to call the ec2 api server
ec2_port=8773	(IntOpt) the port of the ec2 api server
ec2_private_dns_show_ip=False	(BoolOpt) Return the IP address as private dns hostname in describe instances
ec2_scheme=http	(StrOpt) the protocol to use when connecting to the ec2 api server (http, https)
ec2_strict_validation=True	(BoolOpt) Validate security group names according to EC2 specification
ec2_timestamp_expiry=300	(IntOpt) Time in seconds before ec2 timestamp expires
ec2_workers=None	(IntOpt) Number of workers for EC2 API service
keystone_ec2_url=http://localhost:5000/v2.0/ec2tokens	(StrOpt) URL to get token from ec2 request.
lockout_attempts=5	(IntOpt) Number of failed auths before lockout.
lockout_minutes=15	(IntOpt) Number of minutes to lockout if triggered.
lockout_window=15	(IntOpt) Number of minutes for lockout window.
region_list=	(ListOpt) list of region=fqdn pairs separated by commas

Table 2.46. Description of configuration options for fping

Configuration option=Default value	Description
fping_path=/usr/sbin/fping	(StrOpt) Full path to fping.

Table 2.47. Description of configuration options for glance

Configuration option=Default value	Description
allowed_direct_url_schemes=	(ListOpt) A list of url scheme that can be downloaded directly via the direct_url. Currently supported schemes: [file].
filesystems=	(ListOpt) A list of filesystems that will be configured in this file under the sections image_file_url:<list entry name>
glance_api_insecure=False	(BoolOpt) Allow to perform insecure SSL (https) requests to glance

Configuration option=Default value	Description
glance_api_servers=$glance_host:$glance_port	(ListOpt) A list of the glance api servers available to nova. Prefix with https:// for ssl-based glance api servers. ([hostname|ip]:port)
glance_host=$my_ip	(StrOpt) default glance hostname or ip
glance_num_retries=0	(IntOpt) Number retries when downloading an image from glance
glance_port=9292	(IntOpt) default glance port
glance_protocol=http	(StrOpt) Default protocol to use when connecting to glance. Set to https for SSL.
osapi_glance_link_prefix=None	(StrOpt) Base URL that will be presented to users in links to glance resources

Table 2.48. Description of configuration options for hyperv

Configuration option=Default value	Description
dynamic_memory_ratio=1.0	(FloatOpt) Enables dynamic memory allocation (ballooning) when set to a value greater than 1. The value expresses the ratio between the total RAM assigned to an instance and its startup RAM amount. For example a ratio of 2.0 for an instance with 1024MB of RAM implies 512MB of RAM allocated at startup
enable_instance_metrics_collection=False	(BoolOpt) Enables metrics collections for an instance by using Hyper-V's metric APIs. Collected data can by retrieved by other apps and services, e.g.: Ceilometer. Requires Hyper-V / Windows Server 2012 and above
force_hyperv_utils_v1=False	(BoolOpt) Force V1 WMI utility classes
instances_path_share=	(StrOpt) The name of a Windows share name mapped to the "instances_path" dir and used by the resize feature to copy files to the target host. If left blank, an administrative share will be used, looking for the same "instances_path" used locally
limit_cpu_features=False	(BoolOpt) Required for live migration among hosts with different CPU features
qemu_img_cmd=qemu-img.exe	(StrOpt) qemu-img is used to convert between different image types
vswitch_name=None	(StrOpt) External virtual switch Name, if not provided, the first external virtual switch is used

Table 2.49. Description of configuration options for hypervisor

Configuration option=Default value	Description
block_migration_flag=VIR_MIGRATE_UNDEFINE_SOURCE, VIR_MIGRATE_PEER2PEER, VIR_MIGRATE_NON_SHARED_INC	(StrOpt) Migration flags to be set for block migration
checksum_base_images=False	(BoolOpt) Write a checksum for files in _base to disk
default_ephemeral_format=None	(StrOpt) The default format an ephemeral_volume will be formatted with on creation.
disk_cachemodes=	(ListOpt) Specific cachemodes to use for different disk types e.g: ["file=directsync","block=none"]
force_raw_images=True	(BoolOpt) Force backing images to raw format
inject_password=True	(BoolOpt) Whether baremetal compute injects password or not
libvirt_cpu_mode=None	(StrOpt) Set to "host-model" to clone the host CPU feature flags; to "host-passthrough" to use the host CPU model exactly; to "custom" to use a named CPU model; to "none"

Configuration option=Default value	Description
	to not set any CPU model. If libvirt_type="kvm|qemu", it will default to "host-model", otherwise it will default to "none"
libvirt_cpu_model=None	(StrOpt) Set to a named libvirt CPU model (see names listed in /usr/share/libvirt/cpu_map.xml). Only has effect if libvirt_cpu_mode="custom" and libvirt_type="kvm|qemu"
libvirt_disk_prefix=None	(StrOpt) Override the default disk prefix for the devices attached to a server, which is dependent on libvirt_type. (valid options are: sd, xvd, uvd, vd)
libvirt_images_rbd_ceph_conf=	(StrOpt) path to the ceph configuration file to use
libvirt_images_type=default	(StrOpt) VM Images format. Acceptable values are: raw, qcow2, lvm,rbd, default. If default is specified, then use_cow_images flag is used instead of this one.
libvirt_images_rbd_pool=rbd	(StrOpt) the RADOS pool in which rbd volumes are stored
libvirt_images_volume_group=None	(StrOpt) LVM Volume Group that is used for VM images, when you specify libvirt_images_type=lvm.
libvirt_inject_key=True	(BoolOpt) Inject the ssh public key at boot time
libvirt_inject_partition=1	(IntOpt) The partition to inject to : -2 => disable, -1 => inspect (libguestfs only), 0 => not partitioned, >0 => partition number
libvirt_inject_password=False	(BoolOpt) Inject the admin password at boot time, without an agent.
libvirt_iscsi_use_multipath=False	(BoolOpt) use multipath connection of the iSCSI volume
libvirt_iser_use_multipath=False	(BoolOpt) use multipath connection of the iSER volume
libvirt_lvm_snapshot_size=1000	(IntOpt) The amount of storage (in megabytes) to allocate for LVM snapshot copy-on-write blocks.
libvirt_nonblocking=True	(BoolOpt) Use a separated OS thread pool to realize non-blocking libvirt calls
libvirt_ovs_bridge=br-int	(StrOpt) Name of Integration Bridge used by Open vSwitch
libvirt_snapshot_compression=False	(BoolOpt) Compress snapshot images when possible. This currently applies exclusively to qcow2 images
libvirt_snapshots_directory=$instances_path/snapshots	(StrOpt) Location where libvirt driver will store snapshots before uploading them to image service
libvirt_sparse_logical_volumes=False	(BoolOpt) Create sparse logical volumes (with virtualsize) if this flag is set to True.
libvirt_type=kvm	(StrOpt) Libvirt domain type (valid options are: kvm, lxc, qemu, uml, xen)
libvirt_uri=	(StrOpt) Override the default libvirt URI (which is dependent on libvirt_type)
libvirt_use_virtio_for_bridges=True	(BoolOpt) Use virtio for bridge interfaces with KVM/ QEMU
libvirt_vif_driver=nova.virt.libvirt.vif.LibvirtGenericVIFDriver	(StrOpt) The libvirt VIF driver to configure the VIFs.
libvirt_volume_drivers=iscsi=nova.virt.libvirt.volume.LibvirtISCSIVolumeDriver, iser=nova.virt.libvirt.volume.LibvirtISERVolumeDriver, local=nova.virt.libvirt.volume.LibvirtVolumeDriver, fake=nova.virt.libvirt.volume.LibvirtFakeVolumeDriver, rbd=nova.virt.libvirt.volume.LibvirtNetVolumeDriver, sheepdog=nova.virt.libvirt.volume.LibvirtNetVolumeDriver, nfs=nova.virt.libvirt.volume.LibvirtNFSVolumeDriver, aoe=nova.virt.libvirt.volume.LibvirtAOEVolumeDriver, glusterfs=nova.virt.libvirt.volume.LibvirtGlusterfsVolumeDriver, fibre_channel=nova.virt.libvirt.volume.LibvirtFibreChannelVolumeDriver, scality=nova.virt.libvirt.volume.LibvirtScalityVolumeDriver	(StrOpt) Libvirt handlers for remote volumes.

Configuration option=Default value	Description
libvirt_wait_soft_reboot_seconds=120	(IntOpt) Number of seconds to wait for instance to shut down after soft reboot request is made. We fall back to hard reboot if instance does not shutdown within this window.
preallocate_images=none	(StrOpt) VM image preallocation mode: "none" => no storage provisioning is done up front, "space" => storage is fully allocated at instance start
remove_unused_base_images=True	(BoolOpt) Should unused base images be removed?
remove_unused_kernels=False	(BoolOpt) Should unused kernel images be removed? This is only safe to enable if all compute nodes have been updated to support this option. This will enabled by default in future.
remove_unused_original_minimum_age_seconds=86400	(IntOpt) Unused unresized base images younger than this will not be removed
remove_unused_resized_minimum_age_seconds=3600	(IntOpt) Unused resized base images younger than this will not be removed
rescue_image_id=None	(StrOpt) Rescue ami image
rescue_kernel_id=None	(StrOpt) Rescue aki image
rescue_ramdisk_id=None	(StrOpt) Rescue ari image
rescue_timeout=0	(IntOpt) Automatically unrescue an instance after N seconds. Set to 0 to disable.
snapshot_image_format=None	(StrOpt) Snapshot image format (valid options are : raw, qcow2, vmdk, vdi). Defaults to same as source image
timeout_nbd=10	(IntOpt) time to wait for a NBD device coming up
use_cow_images=True	(BoolOpt) Whether to use cow images
use_usb_tablet=True	(BoolOpt) Sync virtual and real mouse cursors in Windows VMs
vcpu_pin_set=None	(StrOpt) Which pcpus can be used by vcpus of instance e.g: "4-12,^8,15"
virt_mkfs=['default=mkfs.ext3 -L %(fs_label)s -F %(target)s', 'linux=mkfs.ext3 -L %(fs_label)s -F %(target)s', 'windows=mkfs.ntfs –force –fast –label %(fs_label)s %(target)s']	(MultiStrOpt) mkfs commands for ephemeral device. The format is <os_type>=<mkfs command>

Table 2.50. Description of configuration options for ipv6

Configuration option=Default value	Description
fixed_range_v6=fd00::/48	(StrOpt) Fixed IPv6 address block
gateway_v6=None	(StrOpt) Default IPv6 gateway
ipv6_backend=rfc2462	(StrOpt) Backend to use for IPv6 generation
use_ipv6=False	(BoolOpt) use ipv6

Table 2.51. Description of configuration options for kombu

Configuration option=Default value	Description
kombu_ssl_ca_certs=	(StrOpt) SSL certification authority file (valid only if SSL enabled)
kombu_ssl_certfile=	(StrOpt) SSL cert file (valid only if SSL enabled)
kombu_ssl_keyfile=	(StrOpt) SSL key file (valid only if SSL enabled)
kombu_ssl_version=	(StrOpt) SSL version to use (valid only if SSL enabled). valid values are TLSv1, SSLv23 and SSLv3. SSLv2 may be available on some distributions

Table 2.52. Description of configuration options for ldap

Configuration option=Default value	Description
ldap_dns_base_dn=ou=hosts,dc=example,dc=org	(StrOpt) Base DN for DNS entries in ldap
ldap_dns_password=password	(StrOpt) password for ldap DNS
ldap_dns_servers=['dns.example.org']	(MultiStrOpt) DNS Servers for ldap dns driver
ldap_dns_soa_expiry=86400	(StrOpt) Expiry interval (in seconds) for ldap dns driver Statement of Authority
ldap_dns_soa_hostmaster=hostmaster@example.org	(StrOpt) Hostmaster for ldap dns driver Statement of Authority
ldap_dns_soa_minimum=7200	(StrOpt) Minimum interval (in seconds) for ldap dns driver Statement of Authority
ldap_dns_soa_refresh=1800	(StrOpt) Refresh interval (in seconds) for ldap dns driver Statement of Authority
ldap_dns_soa_retry=3600	(StrOpt) Retry interval (in seconds) for ldap dns driver Statement of Authority
ldap_dns_url=ldap://ldap.example.com:389	(StrOpt) URL for ldap server which will store dns entries
ldap_dns_user=uid=admin,ou=people,dc=example,dc=org	(StrOpt) user for ldap DNS

Table 2.53. Description of configuration options for livemigration

Configuration option=Default value	Description
live_migration_bandwidth=0	(IntOpt) Maximum bandwidth to be used during migration, in Mbps
live_migration_flag=VIR_MIGRATE_UNDEFINE_SOURCE, VIR_MIGRATE_PEER2PEER	(StrOpt) Migration flags to be set for live migration
live_migration_retry_count=30	(IntOpt) Number of 1 second retries needed in live_migration
live_migration_uri=qemu+tcp://%s/system	(StrOpt) Migration target URI (any included "%s" is replaced with the migration target hostname)

Table 2.54. Description of configuration options for logging

Configuration option=Default value	Description
debug=False	(BoolOpt) Print debugging output (set logging level to DEBUG instead of default WARNING level).
default_log_levels=amqplib=WARN,sqlalchemy=WARN,boto=WARN,suds=INFO,keystone=INFO,eventlet.wsgi.server=WARN	(ListOpt) list of logger=LEVEL pairs
fatal_deprecations=False	(BoolOpt) make deprecations fatal
fatal_exception_format_errors=False	(BoolOpt) make exception message format errors fatal
instance_format=[instance: %(uuid)s]	(StrOpt) If an instance is passed with the log message, format it like this
instance_uuid_format=[instance: %(uuid)s]	(StrOpt) If an instance UUID is passed with the log message, format it like this
log_config=None	(StrOpt) If this option is specified, the logging configuration file specified is used and overrides any other logging options specified. Please see the Python logging module documentation for details on logging configuration files.
log_date_format=%Y-%m-%d %H:%M:%S	(StrOpt) Format string for %%(asctime)s in log records. Default: %(default)s
log_dir=None	(StrOpt) (Optional) The base directory used for relative –log-file paths
log_file=None	(StrOpt) (Optional) Name of log file to output to. If no default is set, logging will go to stdout.

Configuration option=Default value	Description
log_format=None	(StrOpt) DEPRECATED. A logging.Formatter log message format string which may use any of the available logging.LogRecord attributes. This option is deprecated. Please use logging_context_format_string and logging_default_format_string instead.
logging_context_format_string=%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user)s %(tenant)s] %(instance)s%(message)s	(StrOpt) format string to use for log messages with context
logging_debug_format_suffix=%(funcName)s %(pathname)s:%(lineno)d	(StrOpt) data to append to log format when level is DEBUG
logging_default_format_string=%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s %(message)s	(StrOpt) format string to use for log messages without context
logging_exception_prefix=%(asctime)s.%(msecs)03d %(process)d TRACE %(name)s %(instance)s	(StrOpt) prefix each line of exception output with this format
publish_errors=False	(BoolOpt) publish error events
syslog_log_facility=LOG_USER	(StrOpt) syslog facility to receive log lines
use_stderr=True	(BoolOpt) Log output to standard error
use_syslog=False	(BoolOpt) Use syslog for logging.
verbose=False	(BoolOpt) Print more verbose output (set logging level to INFO instead of default WARNING level).

Table 2.55. Description of configuration options for metadata

Configuration option=Default value	Description
metadata_host=$my_ip	(StrOpt) the ip for the metadata api server
metadata_listen=0.0.0.0	(StrOpt) IP address for metadata api to listen
metadata_listen_port=8775	(IntOpt) port for metadata api to listen
metadata_manager=nova.api.manager.MetadataManager	(StrOpt) OpenStack metadata service manager
metadata_port=8775	(IntOpt) the port for the metadata api port
metadata_workers=None	(IntOpt) Number of workers for metadata service
vendordata_driver=nova.api.metadata.vendordata_json.JsonFileVendorData	(StrOpt) Driver to use for vendor data
vendordata_jsonfile_path=None	(StrOpt) File to load json formated vendor data from

Table 2.56. Description of configuration options for network

Configuration option=Default value	Description
allow_same_net_traffic=True	(BoolOpt) Whether to allow network traffic from same network
auto_assign_floating_ip=False	(BoolOpt) Autoassigning floating ip to VM
cnt_vpn_clients=0	(IntOpt) Number of addresses reserved for vpn clients
create_unique_mac_address_attempts=5	(IntOpt) Number of attempts to create unique mac address
default_access_ip_network_name=None	(StrOpt) Name of network to use to set access ips for instances
default_floating_pool=nova	(StrOpt) Default pool for floating ips
defer_iptables_apply=False	(BoolOpt) Whether to batch up the application of IPTables rules during a host restart and apply all at the end of the init phase
dhcp_domain=novalocal	(StrOpt) domain to use for building the hostnames
dhcp_lease_time=120	(IntOpt) Lifetime of a DHCP lease in seconds

Configuration option=Default value	Description
dhcpbridge=$bindir/nova-dhcpbridge	(StrOpt) location of nova-dhcpbridge
dhcpbridge_flagfile=['/etc/nova/nova-dhcpbridge.conf']	(MultiStrOpt) location of flagfiles for dhcpbridge
dns_server=[]	(MultiStrOpt) if set, uses specific dns server for dnsmasq. Canbe specified multiple times.
dns_update_periodic_interval=-1	(IntOpt) Number of seconds to wait between runs of updates to DNS entries.
dnsmasq_config_file=	(StrOpt) Override the default dnsmasq settings with this file
firewall_driver=None	(StrOpt) Firewall driver (defaults to hypervisor specific iptables driver)
fixed_ip_disassociate_timeout=600	(IntOpt) Seconds after which a deallocated ip is disassociated
flat_injected=False	(BoolOpt) Whether to attempt to inject network setup into guest
flat_interface=None	(StrOpt) FlatDhcp will bridge into this interface if set
flat_network_bridge=None	(StrOpt) Bridge for simple network instances
flat_network_dns=8.8.4.4	(StrOpt) Dns for simple network
floating_ip_dns_manager=nova.network.noop_dns_driver.NoopDNSDriver	(StrOpt) full class name for the DNS Manager for floating IPs
force_dhcp_release=True	(BoolOpt) If True, send a dhcp release on instance termination
force_snat_range=[]	(MultiStrOpt) Traffic to this range will always be snatted to the fallback ip, even if it would normally be bridged out of the node. Can be specified multiple times.
forward_bridge_interface=['all']	(MultiStrOpt) An interface that bridges can forward to. If this is set to all then all traffic will be forwarded. Can be specified multiple times.
gateway=None	(StrOpt) Default IPv4 gateway
injected_network_template=$pybasedir/nova/virt/interfaces.template	(StrOpt) Template file for injected network
injected_network_template=$pybasedir/nova/virt/baremetal/interfaces.template	(StrOpt) Template file for injected network
injected_network_template=$pybasedir/nova/virt/interfaces.template	(StrOpt) Template file for injected network
injected_network_template=$pybasedir/nova/virt/baremetal/interfaces.template	(StrOpt) Template file for injected network
instance_dns_domain=	(StrOpt) full class name for the DNS Zone for instance IPs
instance_dns_manager=nova.network.noop_dns_driver.NoopDNSDriver	(StrOpt) full class name for the DNS Manager for instance IPs
iptables_bottom_regex=	(StrOpt) Regular expression to match iptables rule that shouldalways be on the bottom.
iptables_drop_action=DROP	(StrOpt) The table that iptables to jump to when a packet is to be dropped.
iptables_top_regex=	(StrOpt) Regular expression to match iptables rule that shouldalways be on the top.
l3_lib=nova.network.l3.LinuxNetL3	(StrOpt) Indicates underlying L3 management library
linuxnet_interface_driver=nova.network.linux_net.LinuxBridgeInterfaceDriver	(StrOpt) Driver used to create ethernet devices.
linuxnet_ovs_integration_bridge=br-int	(StrOpt) Name of Open vSwitch bridge used with linuxnet
multi_host=False	(BoolOpt) Default value for multi_host in networks. Also, if set, some rpc network calls will be sent directly to host.

Configuration option=Default value	Description
network_allocate_retries=0	(IntOpt) Number of times to retry network allocation on failures
network_api_class=nova.network.api.API	(StrOpt) The full class name of the network API class to use
network_device_mtu=None	(StrOpt) MTU setting for vlan
network_driver=nova.network.linux_net	(StrOpt) Driver to use for network creation
network_manager=nova.network.manager.VlanManager	(StrOpt) full class name for the Manager for network
network_size=256	(IntOpt) Number of addresses in each private subnet
network_topic=network	(StrOpt) the topic network nodes listen on
networks_path=$state_path/networks	(StrOpt) Location to keep network config files
num_networks=1	(IntOpt) Number of networks to support
public_interface=eth0	(StrOpt) Interface for public IP addresses
routing_source_ip=$my_ip	(StrOpt) Public IP of network host
security_group_api=nova	(StrOpt) The full class name of the security API class
send_arp_for_ha=False	(BoolOpt) send gratuitous ARPs for HA setup
send_arp_for_ha_count=3	(IntOpt) send this many gratuitous ARPs for HA setup
share_dhcp_address=False	(BoolOpt) If True in multi_host mode, all compute hosts share the same dhcp address. The same IP address used for DHCP will be added on each nova-network node which is only visible to the vms on the same host.
teardown_unused_network_gateway=False	(BoolOpt) If True, unused gateway devices (VLAN and bridge) are deleted in VLAN network mode with multi hosted networks
update_dns_entries=False	(BoolOpt) If True, when a DNS entry must be updated, it sends a fanout cast to all network hosts to update their DNS entries in multi host mode
use_network_dns_servers=False	(BoolOpt) if set, uses the dns1 and dns2 from the network ref.as dns servers.
use_neutron_default_nets=False	(StrOpt) Control for checking for default networks
use_single_default_gateway=False	(BoolOpt) Use single default gateway. Only first nic of vm will get default gateway from dhcp server
vlan_interface=None	(StrOpt) vlans will bridge into this interface if set
vlan_interface=vmnic0	(StrOpt) Physical ethernet adapter name for vlan networking
vlan_start=100	(IntOpt) First VLAN for private networks

Table 2.57. Description of configuration options for periodic

Configuration option=Default value	Description
periodic_enable=True	(BoolOpt) enable periodic tasks
periodic_fuzzy_delay=60	(IntOpt) range of seconds to randomly delay when starting the periodic task scheduler to reduce stampeding. (Disable by setting to 0)
run_external_periodic_tasks=True	(BoolOpt) Some periodic tasks can be run in a separate process. Should we run them here?

Table 2.58. Description of configuration options for policy

Configuration option=Default value	Description
allow_instance_snapshots=True	(BoolOpt) Permit instance snapshot operations.

Configuration option=Default value	Description
allow_migrate_to_same_host=False	(BoolOpt) Allow migrate machine to the same host. Useful when testing in single-host environments.
allow_resize_to_same_host=False	(BoolOpt) Allow destination machine to match source for resize. Useful when testing in single-host environments.
max_age=0	(IntOpt) number of seconds between subsequent usage refreshes
max_local_block_devices=3	(IntOpt) Maximum number of devices that will result in a local image being created on the hypervisor node. Setting this to 0 means nova will allow only boot from volume. A negative number means unlimited.
osapi_compute_unique_server_name_scope=	(StrOpt) When set, compute API will consider duplicate hostnames invalid within the specified scope, regardless of case. Should be empty, "project" or "global".
osapi_max_limit=1000	(IntOpt) the maximum number of items returned in a single response from a collection resource
osapi_max_request_body_size=114688	(IntOpt) the maximum body size per each osapi request(bytes)
password_length=12	(IntOpt) Length of generated instance admin passwords
policy_default_rule=default	(StrOpt) Rule checked when requested rule is not found
policy_file=policy.json	(StrOpt) JSON file representing policy
reservation_expire=86400	(IntOpt) number of seconds until a reservation expires
resize_fs_using_block_device=True	(BoolOpt) Attempt to resize the filesystem by accessing the image over a block device. This is done by the host and may not be necessary if the image contains a recent version of cloud-init. Possible mechanisms require the nbd driver (for qcow and raw), or loop (for raw).
until_refresh=0	(IntOpt) count of reservations until usage is refreshed

Table 2.59. Description of configuration options for powervm

Configuration option=Default value	Description
powervm_img_local_path=/tmp	(StrOpt) Local directory to download glance images to. Make sure this path can fit your biggest image in glance
powervm_img_remote_path=/home/padmin	(StrOpt) PowerVM image remote path where images will be moved. Make sure this path can fit your biggest image in glance
powervm_mgr=None	(StrOpt) PowerVM manager host or ip
powervm_mgr_passwd=None	(StrOpt) PowerVM manager user password
powervm_mgr_type=ivm	(StrOpt) PowerVM manager type (ivm, hmc)
powervm_mgr_user=None	(StrOpt) PowerVM manager user name

Table 2.60. Description of configuration options for qpid

Configuration option=Default value	Description
qpid_heartbeat=60	(IntOpt) Seconds between connection keepalive heartbeats
qpid_hostname=localhost	(StrOpt) Qpid broker hostname
qpid_hosts=$qpid_hostname:$qpid_port	(ListOpt) Qpid HA cluster host:port pairs
qpid_password=	(StrOpt) Password for qpid connection
qpid_port=5672	(IntOpt) Qpid broker port
qpid_protocol=tcp	(StrOpt) Transport to use, either 'tcp' or 'ssl'

Configuration option=Default value	Description
qpid_sasl_mechanisms=	(StrOpt) Space separated list of SASL mechanisms to use for auth
qpid_tcp_nodelay=True	(BoolOpt) Disable Nagle algorithm
qpid_topology_version=1	(IntOpt) The qpid topology version to use. Version 1 is what was originally used by impl_qpid. Version 2 includes some backwards-incompatible changes that allow broker federation to work. Users should update to version 2 when they are able to take everything down, as it requires a clean break.
qpid_username=	(StrOpt) Username for qpid connection

Table 2.61. Description of configuration options for neutron

Configuration option=Default value	Description
dhcp_options_enabled=False	(BoolOpt) Use per-port DHCP options with Neutron
neutron_admin_auth_url=http://localhost:5000/v2.0	(StrOpt) auth url for connecting to neutron in admin context
neutron_admin_password=None	(StrOpt) password for connecting to neutron in admin context
neutron_admin_tenant_name=None	(StrOpt) tenant name for connecting to neutron in admin context
neutron_admin_username=None	(StrOpt) username for connecting to neutron in admin context
neutron_api_insecure=False	(BoolOpt) if set, ignore any SSL validation issues
neutron_auth_strategy=keystone	(StrOpt) auth strategy for connecting to neutron in admin context
neutron_ca_certificates_file=None	(StrOpt) Location of ca certicates file to use for neutronclient requests.
neutron_default_tenant_id=default	(StrOpt) Default tenant id when creating neutron networks
neutron_extension_sync_interval=600	(IntOpt) Number of seconds before querying neutron for extensions
neutron_metadata_proxy_shared_secret=	(StrOpt) Shared secret to validate proxies Neutron metadata requests
neutron_ovs_bridge=br-int	(StrOpt) Name of Integration Bridge used by Open vSwitch
neutron_region_name=None	(StrOpt) region name for connecting to neutron in admin context
neutron_url=http://127.0.0.1:9696	(StrOpt) URL for connecting to neutron
neutron_url_timeout=30	(IntOpt) timeout value for connecting to neutron in seconds
service_neutron_metadata_proxy=False	(BoolOpt) Set flag to indicate Neutron will proxy metadata requests and resolve instance ids.

Table 2.62. Description of configuration options for quota

Configuration option=Default value	Description
bandwidth_poll_interval=600	(IntOpt) interval to pull bandwidth usage info
bandwidth_update_interval=600	(IntOpt) Seconds between bandwidth updates for cells.
enable_network_quota=False	(BoolOpt) Enables or disables quotaing of tenant networks
quota_cores=20	(IntOpt) number of instance cores allowed per project
quota_driver=nova.quota.DbQuotaDriver	(StrOpt) default driver to use for quota checks

Configuration option=Default value	Description
quota_fixed_ips=-1	(IntOpt) number of fixed ips allowed per project (this should be at least the number of instances allowed)
quota_floating_ips=10	(IntOpt) number of floating ips allowed per project
quota_injected_file_content_bytes=10240	(IntOpt) number of bytes allowed per injected file
quota_injected_file_path_bytes=255	(IntOpt) number of bytes allowed per injected file path
quota_injected_files=5	(IntOpt) number of injected files allowed
quota_instances=10	(IntOpt) number of instances allowed per project
quota_key_pairs=100	(IntOpt) number of key pairs per user
quota_metadata_items=128	(IntOpt) number of metadata items allowed per instance
quota_ram=51200	(IntOpt) megabytes of instance ram allowed per project
quota_security_group_rules=20	(IntOpt) number of security rules per security group
quota_security_groups=10	(IntOpt) number of security groups per project

Table 2.63. Description of configuration options for rabbitmq

Configuration option=Default value	Description
rabbit_ha_queues=False	(BoolOpt) use H/A queues in RabbitMQ (x-ha-policy: all).You need to wipe RabbitMQ database when changing this option.
rabbit_host=localhost	(StrOpt) The RabbitMQ broker address where a single node is used
rabbit_hosts=$rabbit_host:$rabbit_port	(ListOpt) RabbitMQ HA cluster host:port pairs
rabbit_max_retries=0	(IntOpt) maximum retries with trying to connect to RabbitMQ (the default of 0 implies an infinite retry count)
rabbit_password=guest	(StrOpt) the RabbitMQ password
rabbit_port=5672	(IntOpt) The RabbitMQ broker port where a single node is used
rabbit_retry_backoff=2	(IntOpt) how long to backoff for between retries when connecting to RabbitMQ
rabbit_retry_interval=1	(IntOpt) how frequently to retry connecting with RabbitMQ
rabbit_use_ssl=False	(BoolOpt) connect over SSL for RabbitMQ
rabbit_userid=guest	(StrOpt) the RabbitMQ userid
rabbit_virtual_host=/	(StrOpt) the RabbitMQ virtual host

Table 2.64. Description of configuration options for rpc

Configuration option=Default value	Description
amqp_durable_queues=False	(BoolOpt) Use durable queues in amqp.
amqp_auto_delete=False	(BoolOpt) Auto-delete queues in amqp.
baseapi=None	(StrOpt) Set a version cap for messages sent to the base api in any service
control_exchange=openstack	(StrOpt) AMQP exchange to connect to if using RabbitMQ or Qpid
matchmaker_heartbeat_freq=300	(IntOpt) Heartbeat frequency
matchmaker_heartbeat_ttl=600	(IntOpt) Heartbeat time-to-live.
ringfile=/etc/oslo/matchmaker_ring.json	(StrOpt) Matchmaker ring file (JSON)
rpc_backend=nova.openstack.common.rpc.impl_kombu	(StrOpt) The messaging module to use, defaults to kombu.
rpc_cast_timeout=30	(IntOpt) Seconds to wait before a cast expires (TTL). Only supported by impl_zmq.

Configuration option=Default value	Description
rpc_conn_pool_size=30	(IntOpt) Size of RPC connection pool
rpc_driver_queue_base=cells.intercell	(StrOpt) Base queue name to use when communicating between cells. Various topics by message type will be appended to this.
rpc_response_timeout=60	(IntOpt) Seconds to wait for a response from call or multicall
rpc_thread_pool_size=64	(IntOpt) Size of RPC thread pool
topics=notifications	(ListOpt) AMQP topic(s) used for OpenStack notifications

Table 2.65. Description of configuration options for s3

Configuration option=Default value	Description
buckets_path=$state_path/buckets	(StrOpt) path to s3 buckets
image_decryption_dir=/tmp	(StrOpt) parent dir for tempdir used for image decryption
s3_access_key=notchecked	(StrOpt) access key to use for s3 server for images
s3_affix_tenant=False	(BoolOpt) whether to affix the tenant id to the access key when downloading from s3
s3_host=$my_ip	(StrOpt) hostname or ip for OpenStack to use when accessing the s3 api
s3_listen=0.0.0.0	(StrOpt) IP address for S3 API to listen
s3_listen_port=3333	(IntOpt) port for s3 api to listen
s3_port=3333	(IntOpt) port used when accessing the s3 api
s3_secret_key=notchecked	(StrOpt) secret key to use for s3 server for images
s3_use_ssl=False	(BoolOpt) whether to use ssl when talking to s3

Table 2.66. Description of configuration options for scheduling

Configuration option=Default value	Description
cpu_allocation_ratio=16.0	(FloatOpt) Virtual CPU to physical CPU allocation ratio which affects all CPU filters. This configuration specifies a global ratio for CoreFilter. For AggregateCoreFilter, it will fall back to this configuration value if no per-aggregate setting found.
disk_allocation_ratio=1.0	(FloatOpt) virtual disk to physical disk allocation ratio
isolated_hosts=	(ListOpt) Host reserved for specific images
isolated_images=	(ListOpt) Images to run on isolated host
max_instances_per_host=50	(IntOpt) Ignore hosts that have too many instances
max_io_ops_per_host=8	(IntOpt) Ignore hosts that have too many builds/resizes/snaps/migrations
ram_allocation_ratio=1.5	(FloatOpt) Virtual ram to physical ram allocation ratio which affects all ram filters. This configuration specifies a global ratio for RamFilter. For AggregateRamFilter, it will fall back to this configuration value if no per-aggregate setting found.
ram_weight_multiplier=10.0	(FloatOpt) Multiplier used for weighing ram. Negative numbers mean to stack vs spread.
ram_weight_multiplier=1.0	(FloatOpt) Multiplier used for weighing ram. Negative numbers mean to stack vs spread.
reserved_host_disk_mb=0	(IntOpt) Amount of disk in MB to reserve for the host
reserved_host_memory_mb=512	(IntOpt) Amount of memory in MB to reserve for the host
restrict_isolated_hosts_to_isolated_images=True	(BoolOpt) Whether to force isolated hosts to run only isolated images

Configuration option=Default value	Description
scheduler_available_filters=['nova.scheduler.filters.all_filters']	(MultiStrOpt) Filter classes available to the scheduler which may be specified more than once. An entry of "nova.scheduler.filters.standard_filters" maps to all filters included with nova.
scheduler_default_filters=RetryFilter,AvailabilityZoneFilter,RamFilter,ComputeFilter,ComputeCapabilitiesFilter,ImagePropertiesFilter	(ListOpt) Which filter class names to use for filtering hosts when not specified in the request.
scheduler_driver=nova.scheduler.filter_scheduler.FilterScheduler	(StrOpt) Default driver to use for the scheduler
scheduler_filter_classes=nova.cells.filters.all_filters	(ListOpt) Filter classes the cells scheduler should use. An entry of "nova.cells.filters.all_filters"maps to all cells filters included with nova.
scheduler_host_manager=nova.scheduler.host_manager.HostManager	(StrOpt) The scheduler host manager class to use
scheduler_host_subset_size=1	(IntOpt) New instances will be scheduled on a host chosen randomly from a subset of the N best hosts. This property defines the subset size that a host is chosen from. A value of 1 chooses the first host returned by the weighing functions. This value must be at least 1. Any value less than 1 will be ignored, and 1 will be used instead
scheduler_json_config_location=	(StrOpt) Absolute path to scheduler configuration JSON file.
scheduler_manager=nova.scheduler.manager.SchedulerManager	(StrOpt) full class name for the Manager for scheduler
scheduler_max_attempts=3	(IntOpt) Maximum number of attempts to schedule an instance
scheduler_retries=10	(IntOpt) How many retries when no cells are available.
scheduler_retry_delay=2	(IntOpt) How often to retry in seconds when no cells are available.
scheduler_topic=scheduler	(StrOpt) the topic scheduler nodes listen on
scheduler_weight_classes=nova.cells.weights.all_weighers	(ListOpt) Weigher classes the cells scheduler should use. An entry of "nova.cells.weights.all_weighers"maps to all cell weighers included with nova.
scheduler_weight_classes=nova.scheduler.weights.all_weighers	(ListOpt) Which weight class names to use for weighing hosts

Table 2.67. Description of configuration options for spice

Configuration option=Default value	Description
agent_enabled=True	(BoolOpt) enable spice guest agent support
enabled=False	(BoolOpt) enable spice related features
enabled=False	(BoolOpt) Whether the V3 API is enabled or not
html5proxy_base_url=http://127.0.0.1:6082/spice_auto.html	(StrOpt) location of spice html5 console proxy, in the form "http://127.0.0.1:6082/spice_auto.html"
keymap=en-us	(StrOpt) keymap for spice
server_listen=127.0.0.1	(StrOpt) IP address on which instance spice server should listen
server_proxyclient_address=127.0.0.1	(StrOpt) the address to which proxy clients (like nova-spicehtml5proxy) should connect

Table 2.68. Description of configuration options for testing

Configuration option=Default value	Description
allowed_rpc_exception_modules=nova.exception,cinder.exception,exceptions	(ListOpt) Modules of exceptions that are permitted to be recreatedupon receiving exception data from an rpc call.
backdoor_port=None	(StrOpt) 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

Configuration option=Default value	Description
	on the specified port number and not enabling backdoorif it 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 service's log file.
fake_call=False	(BoolOpt) If True, skip using the queue and make local calls
fake_network=False	(BoolOpt) If passed, use fake network devices and addresses
fake_rabbit=False	(BoolOpt) If passed, use a fake RabbitMQ provider
monkey_patch=False	(BoolOpt) Whether to log monkey patching
monkey_patch_modules=nova.api.ec2.cloud:nova.notifications.notify_decorator,nova.compute.api:nova.notifications.notify_decorator	(ListOpt) List of modules/decorators to monkey patch

Table 2.69. Description of configuration options for tilera

Configuration option=Default value	Description
tile_pdu_ip=10.0.100.1	(StrOpt) ip address of tilera pdu
tile_pdu_mgr=/tftpboot/pdu_mgr	(StrOpt) management script for tilera pdu
tile_pdu_off=2	(IntOpt) power status of tilera PDU is OFF
tile_pdu_on=1	(IntOpt) power status of tilera PDU is ON
tile_pdu_status=9	(IntOpt) power status of tilera PDU
tile_power_wait=9	(IntOpt) wait time in seconds until check the result after tilera power operations

Table 2.70. Description of configuration options for trustedcomputing

Configuration option=Default value	Description
attestation_api_url=/OpenAttestationWebServices/V1.0	(StrOpt) attestation web API URL
attestation_auth_blob=None	(StrOpt) attestation authorization blob - must change
attestation_auth_timeout=60	(IntOpt) Attestation status cache valid period length
attestation_port=8443	(StrOpt) attestation server port
attestation_server=None	(StrOpt) attestation server http
attestation_server_ca_file=None	(StrOpt) attestation server Cert file for Identity verification

Table 2.71. Description of configuration options for vmware

Configuration option=Default value	Description
api_retry_count=10	(IntOpt) The number of times we retry on failures, e.g., socket error, etc. Used only if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.
cluster_name=None	(MultiStrOpt) Name of a VMware Cluster ComputeResource. Used only if compute_driver is vmwareapi.VMwareVCDriver.
datastore_regex=None	(StrOpt) Regex to match the name of a datastore. Used only if compute_driver is vmwareapi.VMwareVCDriver.
host_ip=None	(StrOpt) URL for connection to VMware ESX/VC host. Required if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.
host_username=None	(StrOpt) Username for connection to VMware ESX/VC host. Used only if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.

Configuration option=Default value	Description
host_password=None	(StrOpt) Password for connection to VMware ESX/VC host. Used only if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.
integration_bridge=br-int	(StrOpt) Name of Integration Bridge
maximum_objects=100	(IntOpt) The maximum number of ObjectContent data objects that should be returned in a single result. A positive value will cause the operation to suspend the retrieval when the count of objects reaches the specified maximum. The server may still limit the count to something less than the configured value. Any remaining objects may be retrieved with additional requests.
task_poll_interval=5.0	(FloatOpt) The interval used for polling of remote tasks. Used only if compute_driver is vmwareapi.VMwareESXDriver or vmwareapi.VMwareVCDriver.
use_linked_clone=True	(BoolOpt) Whether to use linked clone
wsdl_location=None	(StrOpt) Optional VIM Service WSDL Location e.g http://<server>/vimService.wsdl. Optional over-ride to default location for bug work-arounds

Table 2.72. Description of configuration options for vnc

Configuration option=Default value	Description
novncproxy_base_url=http://127.0.0.1:6080/vnc_auto.html	(StrOpt) location of vnc console proxy, in the form "http://127.0.0.1:6080/vnc_auto.html"
vnc_enabled=True	(BoolOpt) enable vnc related features
vnc_keymap=en-us	(StrOpt) keymap for vnc
vnc_password=None	(StrOpt) VNC password
vnc_port=5900	(IntOpt) VNC starting port
vnc_port_total=10000	(IntOpt) Total number of VNC ports
vncserver_listen=127.0.0.1	(StrOpt) IP address on which instance vncservers should listen
vncserver_proxyclient_address=127.0.0.1	(StrOpt) the address to which proxy clients (like nova-xvpvncproxy) should connect

Table 2.73. Description of configuration options for volumes

Configuration option=Default value	Description
block_device_creation_timeout=10	(IntOpt) Time to wait for a block device to be created
cinder_api_insecure=False	(BoolOpt) Allow to perform insecure SSL requests to cinder
cinder_ca_certificates_file=None	(StrOpt) Location of ca certicates file to use for cinder client requests.
cinder_catalog_info=volume:cinder:publicURL	(StrOpt) Info to match when looking for cinder in the service catalog. Format is : separated values of the form: <service_type>:<service_name>:<endpoint_type>
cinder_cross_az_attach=True	(BoolOpt) Allow attach between instance and volume in different availability zones.
cinder_endpoint_template=None	(StrOpt) Override service catalog lookup with template for cinder endpoint e.g. http://localhost:8776/v1/%(project_id)s
cinder_http_retries=3	(IntOpt) Number of cinderclient retries on failed http calls
force_volumeutils_v1=False	(BoolOpt) Force V1 volume utility class

Configuration option=Default value	Description
glusterfs_mount_point_base=$state_path/mnt	(StrOpt) Dir where the glusterfs volume is mounted on the compute node
iscsi_iqn_prefix=iqn.2010-10.org.openstack.baremetal	(StrOpt) iSCSI IQN prefix used in baremetal volume connections.
nfs_mount_options=None	(StrOpt) Mount options passed to the nfs client. See section of the nfs man page for details
nfs_mount_point_base=$state_path/mnt	(StrOpt) Dir where the nfs volume is mounted on the compute node
num_aoe_discover_tries=3	(IntOpt) number of times to rediscover AoE target to find volume
num_iscsi_scan_tries=3	(IntOpt) number of times to rescan iSCSI target to find volume
num_iser_scan_tries=3	(IntOpt) number of times to rescan iSER target to find volume
os_region_name=None	(StrOpt) region name of this node
qemu_allowed_storage_drivers=	(ListOpt) Protocols listed here will be accessed directly from QEMU. Currently supported protocols: [gluster]
rbd_secret_uuid=None	(StrOpt) the libvirt uuid of the secret for the rbd_uservolumes
rbd_user=None	(StrOpt) the RADOS client name for accessing rbd volumes
scality_sofs_config=None	(StrOpt) Path or URL to Scality SOFS configuration file
scality_sofs_mount_point=$state_path/scality	(StrOpt) Base dir where Scality SOFS shall be mounted
volume_api_class=nova.volume.cinder.API	(StrOpt) The full class name of the volume API class to use
volume_attach_retry_count=10	(IntOpt) The number of times to retry to attach a volume
volume_attach_retry_interval=5	(IntOpt) Interval between volume attachment attempts, in seconds
volume_driver=nova.virt.baremetal.volume_driver.LibvirtVolumeDriver	(StrOpt) Baremetal volume driver.
volume_usage_poll_interval=0	(IntOpt) Interval in seconds for gathering volume usages

Table 2.74. Description of configuration options for vpn

Configuration option=Default value	Description
boot_script_template=$pybasedir/nova/cloudpipe/bootscript.template	(StrOpt) Template for cloudpipe instance boot script
dmz_cidr=	(ListOpt) A list of dmz range that should be accepted
dmz_mask=255.255.255.0	(StrOpt) Netmask to push into openvpn config
dmz_net=10.0.0.0	(StrOpt) Network to push into openvpn config
vpn_flavor=m1.tiny	(StrOpt) Flavor for vpn instances
vpn_image_id=0	(StrOpt) image id used when starting up a cloudpipe vpn server
vpn_ip=$my_ip	(StrOpt) Public IP for the cloudpipe VPN servers
vpn_key_suffix=-vpn	(StrOpt) Suffix to add to project name for vpn key and secgroups
vpn_start=1000	(IntOpt) First Vpn port for private networks

Table 2.75. Description of configuration options for wsgi

Configuration option=Default value	Description
api_paste_config=api-paste.ini	(StrOpt) File name for the paste.deploy config for nova-api
ssl_ca_file=None	(StrOpt) CA certificate file to use to verify connecting clients

Configuration option=Default value	Description
ssl_cert_file=None	(StrOpt) SSL certificate of API server
ssl_key_file=None	(StrOpt) SSL private key of API server
tcp_keepidle=600	(IntOpt) Sets the value of TCP_KEEPIDLE in seconds for each server socket. Not supported on OS X.
wsgi_log_format=%(client_ip)s "%(request_line)s" status: %(status_code)s len: %(body_length)s time: %(wall_seconds).7f	(StrOpt) A python format string that is used as the template to generate log lines. The following values can be formatted into it: client_ip, date_time, request_line, status_code, body_length, wall_seconds.

Table 2.76. Description of configuration options for xen

Configuration option=Default value	Description
agent_resetnetwork_timeout=60	(IntOpt) number of seconds to wait for agent reply to resetnetwork request
agent_timeout=30	(IntOpt) number of seconds to wait for agent reply
agent_version_timeout=300	(IntOpt) number of seconds to wait for agent to be fully operational
cache_images=all	(StrOpt) Cache glance images locally. `all` will cache all images, `some` will only cache images that have the image_property `cache_in_nova=True`, and `none` turns off caching entirely
console_driver=nova.console.xvp.XVPConsoleProxy	(StrOpt) Driver to use for the console proxy
console_vmrc_error_retries=10	(IntOpt) number of retries for retrieving VMRC information
console_vmrc_port=443	(IntOpt) port for VMware VMRC connections
console_xvp_conf=/etc/xvp.conf	(StrOpt) generated XVP conf file
console_xvp_conf_template=$pybasedir/nova/console/xvp.conf.template	(StrOpt) XVP conf template
console_xvp_log=/var/log/xvp.log	(StrOpt) XVP log file
console_xvp_multiplex_port=5900	(IntOpt) port for XVP to multiplex VNC connections on
console_xvp_pid=/var/run/xvp.pid	(StrOpt) XVP master process pid file
default_os_type=linux	(StrOpt) Default OS type
iqn_prefix=iqn.2010-10.org.openstack	(StrOpt) IQN Prefix
max_kernel_ramdisk_size=16777216	(IntOpt) Maximum size in bytes of kernel or ramdisk images
sr_matching_filter=default-sr:true	(StrOpt) Filter for finding the SR to be used to install guest instances on. To use the Local Storage in default XenServer/XCP installations set this flag to other-config:i18n-key=local-storage. To select an SR with a different matching criteria, you could set it to other-config:my_favorite_sr=true. On the other hand, to fall back on the Default SR, as displayed by XenCenter, set this flag to: default-sr:true
stub_compute=False	(BoolOpt) Stub calls to compute worker for tests
target_host=None	(StrOpt) iSCSI Target Host
target_port=3260	(StrOpt) iSCSI Target Port, 3260 Default
use_join_force=True	(BoolOpt) To use for hosts with different CPUs
xen_hvmloader_path=/usr/lib/xen/boot/hvmloader	(StrOpt) Location where the Xen hvmloader is kept
xenapi_agent_path=usr/sbin/xe-update-networking	(StrOpt) Specifies the path in which the xenapi guest agent should be located. If the agent is present, network configuration is not injected into the image. Used if compute_driver=xenapi.XenAPIDriver and flat_injected=True

Configuration option=Default value	Description
xenapi_check_host=True	(BoolOpt) Ensure compute service is running on host XenAPI connects to.
xenapi_connection_concurrent=5	(IntOpt) Maximum number of concurrent XenAPI connections. Used only if compute_driver=xenapi.XenAPIDriver
xenapi_connection_password=None	(StrOpt) Password for connection to XenServer/Xen Cloud Platform. Used only if compute_driver=xenapi.XenAPIDriver
xenapi_connection_url=None	(StrOpt) URL for connection to XenServer/Xen Cloud Platform. A special value of unix://local can be used to connect to the local unix socket. Required if compute_driver=xenapi.XenAPIDriver
xenapi_connection_username=root	(StrOpt) Username for connection to XenServer/Xen Cloud Platform. Used only if compute_driver=xenapi.XenAPIDriver
xenapi_disable_agent=False	(BoolOpt) Disables the use of the XenAPI agent in any image regardless of what image properties are present.
xenapi_image_compression_level=None	(IntOpt) Compression level for images, e.g., 9 for gzip -9. Range is 1-9, 9 being most compressed but most CPU intensive on dom0.
xenapi_image_upload_handler=nova.virt.xenapi.image.glance.GlanceStore	(StrOpt) Dom0 plugin driver used to handle image uploads.
xenapi_ipxe_boot_menu_url=None	(StrOpt) URL to the iPXE boot menu
xenapi_login_timeout=10	(IntOpt) Timeout in seconds for XenAPI login.
xenapi_ipxe_mkisofs_cmd=mkisofs	(StrOpt) Name and optionally path of the tool used for ISO image creation
xenapi_num_vbd_unplug_retries=10	(IntOpt) Maximum number of retries to unplug VBD
xenapi_ipxe_network_name=None	(StrOpt) Name of network to use for booting iPXE ISOs
xenapi_ovs_integration_bridge=xapi1	(StrOpt) Name of Integration Bridge used by Open vSwitch
xenapi_remap_vbd_dev=False	(BoolOpt) Used to enable the remapping of VBD dev (Works around an issue in Ubuntu Maverick)
xenapi_remap_vbd_dev_prefix=sd	(StrOpt) Specify prefix to remap VBD dev to (ex. /dev/xvdb -> /dev/sdb)
xenapi_running_timeout=60	(IntOpt) number of seconds to wait for instance to go to running state
xenapi_sparse_copy=True	(BoolOpt) Whether to use sparse_copy for copying data on a resize down (False will use standard dd). This speeds up resizes down considerably since large runs of zeros won't have to be rsynced
xenapi_sr_base_path=/var/run/sr-mount	(StrOpt) Base path to the storage repository
xenapi_torrent_base_url=None	(StrOpt) Base URL for torrent files.
xenapi_torrent_download_stall_cutoff=600	(IntOpt) Number of seconds a download can remain at the same progress percentage w/o being considered a stall
xenapi_torrent_images=none	(StrOpt) Whether or not to download images via Bit Torrent (all|some|none).
xenapi_torrent_listen_port_end=6891	(IntOpt) End of port range to listen on
xenapi_torrent_listen_port_start=6881	(IntOpt) Beginning of port range to listen on
xenapi_torrent_max_last_accessed=86400	(IntOpt) Cached torrent files not accessed within this number of seconds can be reaped
xenapi_torrent_max_seeder_processes_per_host=1	(IntOpt) Maximum number of seeder processes to run concurrently within a given dom0. (-1 = no limit)

Configuration option=Default value	Description
xenapi_torrent_seed_chance=1.0	(FloatOpt) Probability that peer will become a seeder. (1.0 = 100%)
xenapi_torrent_seed_duration=3600	(IntOpt) Number of seconds after downloading an image via BitTorrent that it should be seeded for other peers.
xenapi_use_agent_default=False	(BoolOpt) Determines if the xenapi agent should be used when the image used does not contain a hint to declare if the agent is present or not. The hint is a glance property "xenapi_use_agent" that has the value "true" or "false". Note that waiting for the agent when it is not present will significantly increase server boot times.
xenapi_vhd_coalesce_max_attempts=5	(IntOpt) Max number of times to poll for VHD to coalesce. Used only if compute_driver=xenapi.XenAPIDriver
xenapi_vhd_coalesce_poll_interval=5.0	(FloatOpt) The interval used for polling of coalescing vhds. Used only if compute_driver=xenapi.XenAPIDriver
xenapi_vif_driver=nova.virt.xenapi.vif.XenAPIBridgeDriver	(StrOpt) The XenAPI VIF driver using XenServer Network APIs.

Table 2.77. Description of configuration options for xvpnvncproxy

Configuration option=Default value	Description
xvpvncproxy_base_url=http://127.0.0.1:6081/console	(StrOpt) location of nova xvp vnc console proxy, in the form "http://127.0.0.1:6081/console"
xvpvncproxy_host=0.0.0.0	(StrOpt) Address that the XCP VNC proxy should bind to
xvpvncproxy_port=6081	(IntOpt) Port that the XCP VNC proxy should bind to

Table 2.78. Description of configuration options for zeromq

Configuration option=Default value	Description
rpc_zmq_bind_address=*	(StrOpt) ZeroMQ bind address. Should be a wildcard (*), an ethernet interface, or IP. The "host" option should point or resolve to this address.
rpc_zmq_contexts=1	(IntOpt) Number of ZeroMQ contexts, defaults to 1
rpc_zmq_host=docwork	(StrOpt) Name of this node. Must be a valid hostname, FQDN, or IP address. Must match "host" option, if running Nova.
rpc_zmq_ipc_dir=/var/run/openstack	(StrOpt) Directory for holding IPC sockets
rpc_zmq_matchmaker=nova.openstack.common.rpc.matchmaker.MatchMakerLocalhost	(StrOpt) MatchMaker driver
rpc_zmq_port=9501	(IntOpt) ZeroMQ receiver listening port
rpc_zmq_topic_backlog=None	(IntOpt) Maximum number of ingress messages to locally buffer per topic. Default is unlimited.

Table 2.79. Description of configuration options for zookeeper

Configuration option=Default value	Description
address=None	(StrOpt) The ZooKeeper addresses for servicegroup service in the format of host1:port,host2:port,host3:port
recv_timeout=4000	(IntOpt) recv_timeout parameter for the zk session
sg_prefix=/servicegroups	(StrOpt) The prefix used in ZooKeeper to store ephemeral nodes
sg_retry_interval=5	(IntOpt) Number of seconds to wait until retrying to join the session

3. Image Service

Table of Contents

Compute relies on an external image service to store virtual machine images and maintain a catalog of available images. By default, Compute is configured to use the OpenStack Image Service (Glance), which is currently the only supported image service.

Table 3.1. Description of configuration options for glance

Configuration option=Default value	Description
allowed_direct_url_schemes=	(ListOpt) A list of url scheme that can be downloaded directly via the direct_url. Currently supported schemes: [file].
filesystems=	(ListOpt) A list of filesystems that will be configured in this file under the sections image_file_url:<list entry name>
glance_api_insecure=False	(BoolOpt) Allow to perform insecure SSL (https) requests to glance
glance_api_servers=$glance_host:$glance_port	(ListOpt) A list of the glance api servers available to nova. Prefix with https:// for ssl-based glance api servers. ([hostname|ip]:port)
glance_host=$my_ip	(StrOpt) default glance hostname or ip
glance_num_retries=0	(IntOpt) Number retries when downloading an image from glance
glance_port=9292	(IntOpt) default glance port
glance_protocol=http	(StrOpt) Default protocol to use when connecting to glance. Set to https for SSL.
osapi_glance_link_prefix=None	(StrOpt) Base URL that will be presented to users in links to glance resources

If your installation requires euca2ools to register new images, you must run the `nova-objectstore` service. This service provides an Amazon S3 front-end for Glance, which is required by euca2ools.

Table 3.2. Description of configuration options for s3

Configuration option=Default value	Description
buckets_path=$state_path/buckets	(StrOpt) path to s3 buckets
image_decryption_dir=/tmp	(StrOpt) parent dir for tempdir used for image decryption
s3_access_key=notchecked	(StrOpt) access key to use for s3 server for images
s3_affix_tenant=False	(BoolOpt) whether to affix the tenant id to the access key when downloading from s3
s3_host=$my_ip	(StrOpt) hostname or ip for OpenStack to use when accessing the s3 api
s3_listen=0.0.0.0	(StrOpt) IP address for S3 API to listen
s3_listen_port=3333	(IntOpt) port for s3 api to listen
s3_port=3333	(IntOpt) port used when accessing the s3 api

Configuration option=Default value	Description
s3_secret_key=notchecked	(StrOpt) secret key to use for s3 server for images
s3_use_ssl=False	(BoolOpt) whether to use ssl when talking to s3

You can modify many of the OpenStack Image Catalogue and Delivery Service. The following tables provide a comprehensive list.

Table 3.3. Description of configuration options for common

Configuration option=Default value	Description
allow_additional_image_properties=True	(BoolOpt) Whether to allow users to specify image properties beyond what the image schema provides
api_limit_max=1000	(IntOpt) Maximum permissible number of items that could be returned by a request
backlog=4096	(IntOpt) The backlog value that will be used when creating the TCP listener socket.
bind_host=0.0.0.0	(StrOpt) Address to bind the server. Useful when selecting a particular network interface.
bind_port=None	(IntOpt) The port on which the server will listen.
data_api=glance.db.sqlalchemy.api	(StrOpt) Python module path of data access API
disable_process_locking=False	(BoolOpt) Whether to disable inter-process locks
limit_param_default=25	(IntOpt) Default value for the number of items returned by a request if not specified explicitly in the request
lock_path=None	(StrOpt) Directory to use for lock files.
metadata_encryption_key=None	(StrOpt) Key used for encrypting sensitive metadata while talking to the registry or database.
notifier_strategy=default	(StrOpt) Notifications can be sent when images are create, updated or deleted. There are three methods of sending notifications, logging (via the log_file directive), rabbit (via a rabbitmq queue), qpid (via a Qpid message queue), or noop (no notifications sent, the default).
os_region_name=None	(StrOpt) Region name of this node
property_protection_file=None	(StrOpt) The location of the property protection file.
show_image_direct_url=False	(BoolOpt) Whether to include the backend image storage location in image properties. Revealing storage location can be a security risk, so use this setting with caution!
use_tpool=False	(BoolOpt) Enable the use of thread pooling for all DB API calls
user_storage_quota=0	(IntOpt) Set a system wide quota for every user. This value is the total number of bytes that a user can use across all storage systems. A value of 0 means unlimited.
workers=1	(IntOpt) The number of child process workers that will be created to service API requests.

Table 3.4. Description of configuration options for api

Configuration option=Default value	Description
admin_role=admin	(StrOpt) Role used to identify an authenticated user as administrator.
allow_anonymous_access=False	(BoolOpt) Allow unauthenticated users to access the API with read-only privileges. This only applies when using ContextMiddleware.
db_auto_create=False	(BoolOpt) A boolean that determines if the database will be automatically created.

Configuration option=Default value	Description
default_store=file	(StrOpt) Default scheme to use to store image data. The scheme must be registered by one of the stores defined by the 'known_stores' config option.
default_publisher_id=$host	(StrOpt) Default publisher_id for outgoing notifications
enable_v1_api=True	(BoolOpt) Deploy the v1 OpenStack Images API.
enable_v2_api=True	(BoolOpt) Deploy the v2 OpenStack Images API.
image_size_cap=1099511627776	(IntOpt) Maximum size of image a user can upload in bytes. Defaults to 1099511627776 bytes (1 TB).
known_stores=glance.store.filesystem.Store,glance.store.http.Store,glance.store.rbd.Store,glance.store.s3.Store,glance.store.swift.Store,glance.store.sheepdog.Store,glance.store.cinder.Store	(ListOpt) List of which store classes and store class locations are currently known to glance at startup.
notification_driver=[]	(MultiStrOpt) Driver or drivers to handle sending notifications
owner_is_tenant=True	(BoolOpt) When true, this option sets the owner of an image to be the tenant. Otherwise, the owner of the image will be the authenticated user issuing the request.
send_identity_headers=False	(BoolOpt) Whether to pass through headers containing user and tenant information when making requests to the registry. This allows the registry to use the context middleware without the keystoneclients' auth_token middleware, removing calls to the keystone auth service. It is recommended that when using this option, secure communication between glance api and glance registry is ensured by means other than auth_token middleware.
show_multiple_locations=False	(BoolOpt) Whether to include the backend image locations in image properties. Revealing storage location can be a security risk, so use this setting with caution! The overrides show_image_direct_url.
use_user_token=True	(BoolOpt) Whether to pass through the user token when making requests to the registry.

Table 3.5. Description of configuration options for cinder

Configuration option=Default value	Description
cinder_catalog_info=volume:cinder:publicURL	(StrOpt) Info to match when looking for cinder in the service catalog. Format is : separated values of the form: <service_type>:<service_name>:<endpoint_type>
cinder_ca_certificates_file=None	(StrOpt) Location of ca certicates file to use for cinder client requests.
cinder_http_retries=3	(IntOpt) Number of cinderclient retries on failed http calls
cinder_endpoint_template=None	(StrOpt) Override service catalog lookup with template for cinder endpoint e.g. http://localhost:8776/v1/%(project_id)s
cinder_api_insecure=False	(BoolOpt) Allow to perform insecure SSL requests to cinder

Table 3.6. Description of configuration options for db

Configuration option=Default value	Description
sql_connection=sqlite:///glance.sqlite	(StrOpt) A valid SQLAlchemy connection string for the registry database. Default: %(default)s
sql_idle_timeout=3600	(IntOpt) Period in seconds after which SQLAlchemy should reestablish its connection to the database.
sql_max_retries=60	(IntOpt) The number of times to retry a connection to the SQLserver.

Configuration option=Default value	Description
sql_retry_interval=1	(IntOpt) The amount of time to wait (in seconds) before attempting to retry the SQL connection.
sqlalchemy_debug=False	(BoolOpt) Enable debug logging in sqlalchemy which prints every query and result

Table 3.7. Description of configuration options for filesystem

Configuration option=Default value	Description
filesystem_store_datadir=None	(StrOpt) Directory to which the Filesystem backend store writes images.
filesystem_store_metadata_file=None	(StrOpt) The path to a file which contains the metadata to be returned with any location associated with this store. The file must contain a valid JSON dict.

Table 3.8. Description of configuration options for gridfs

Configuration option=Default value	Description
mongodb_store_uri=None	(StrOpt) Hostname or IP address of the instance to connect to, or a mongodb URI, or a list of hostnames / mongodb URIs. If host is an IPv6 literal it must be enclosed in '[' and ']' characters following the RFC2732 URL syntax (e.g. '[::1]' for localhost)
mongodb_store_db=None	(StrOpt) Database to use

Table 3.9. Description of configuration options for imagecache

Configuration option=Default value	Description
cleanup_scrubber=False	(BoolOpt) A boolean that determines if the scrubber should clean up the files it uses for taking data. Only one server in your deployment should be designated the cleanup host.
cleanup_scrubber_time=86400	(IntOpt) Items must have a modified time that is older than this value in order to be candidates for cleanup.
delayed_delete=False	(BoolOpt) Turn on/off delayed delete.
image_cache_dir=None	(StrOpt) Base directory that the Image Cache uses.
image_cache_driver=sqlite	(StrOpt) The driver to use for image cache management.
image_cache_max_size=10737418240	(IntOpt) The maximum size in bytes that the cache can use.
image_cache_sqlite_db=cache.db	(StrOpt) The path to the sqlite file database that will be used for image cache management.
image_cache_stall_time=86400	(IntOpt) The amount of time to let an image remain in the cache without being accessed
scrub_time=0	(IntOpt) The amount of time in seconds to delay before performing a delete.
scrubber_datadir=/var/lib/glance/scrubber	(StrOpt) Directory that the scrubber will use to track information about what to delete. Make sure this is set in glance-api.conf and glance-scrubber.conf

Table 3.10. Description of configuration options for logging

Configuration option=Default value	Description
debug=False	(BoolOpt) Print debugging output (set logging level to DEBUG instead of default WARNING level).

Configuration option=Default value	Description
default_log_levels=amqplib=WARN,sqlalchemy=WARN,boto=WARN,suds=INFO,keystone=INFO,eventlet.wsgi.server=WARN	(ListOpt) list of logger=LEVEL pairs
default_notification_level=INFO	(StrOpt) Default notification level for outgoing notifications
fatal_deprecations=False	(BoolOpt) make deprecations fatal
instance_format=[instance: %(uuid)s]	(StrOpt) If an instance is passed with the log message, format it like this
instance_uuid_format=[instance: %(uuid)s]	(StrOpt) If an instance UUID is passed with the log message, format it like this
log_config=None	(StrOpt) If this option is specified, the logging configuration file specified is used and overrides any other logging options specified. Please see the Python logging module documentation for details on logging configuration files.
log_date_format=%Y-%m-%d %H:%M:%S	(StrOpt) Format string for %%(asctime)s in log records. Default: %(default)s
log_dir=None	(StrOpt) (Optional) The base directory used for relative –log-file paths
log_file=None	(StrOpt) (Optional) Name of log file to output to. If no default is set, logging will go to stdout.
log_format=None	(StrOpt) A logging.Formatter log message format string which may use any of the available logging.LogRecord attributes. This option is deprecated. Please use logging_context_format_string and logging_default_format_string instead.
logging_context_format_string=%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user)s %(tenant)s] %(instance)s%(message)s	(StrOpt) format string to use for log messages with context
logging_debug_format_suffix=%(funcName)s %(pathname)s:%(lineno)d	(StrOpt) data to append to log format when level is DEBUG
logging_default_format_string=%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s %(message)s	(StrOpt) format string to use for log messages without context
logging_exception_prefix=%(asctime)s.%(msecs)03d %(process)d TRACE %(name)s %(instance)s	(StrOpt) prefix each line of exception output with this format
publish_errors=False	(BoolOpt) publish error events
syslog_log_facility=LOG_USER	(StrOpt) syslog facility to receive log lines
use_stderr=True	(BoolOpt) Log output to standard error
use_syslog=False	(BoolOpt) Use syslog for logging.
verbose=False	(BoolOpt) Print more verbose output (set logging level to INFO instead of default WARNING level).

Table 3.11. Description of configuration options for paste

Configuration option=Default value	Description
config_file=None	(StrOpt) Name of the paste configuration file.
flavor=None	(StrOpt) Partial name of a pipeline in your paste configuration file with the service name removed. For example, if your paste section name is [pipeline:glance-api-keystone] use the value "keystone"

Table 3.12. Description of configuration options for policy

Configuration option=Default value	Description
policy_default_rule=default	(StrOpt) The default policy to use.

Configuration option=Default value	Description
policy_file=policy.json	(StrOpt) The location of the policy file.

Table 3.13. Description of configuration options for qpid

Configuration option=Default value	Description
qpid_heartbeat=60	(IntOpt) Seconds between connection keepalive heartbeats
qpid_hostname=localhost	(StrOpt) Qpid broker hostname
qpid_notification_exchange=glance	(StrOpt) Qpid exchange for notifications
qpid_notification_topic=notifications	(StrOpt) Qpid topic for notifications
qpid_password=	(StrOpt) Password for qpid connection
qpid_port=5672	(StrOpt) Qpid broker port
qpid_protocol=tcp	(StrOpt) Transport to use, either 'tcp' or 'ssl'
qpid_reconnect_interval=0	(IntOpt) Equivalent to setting max and min to the same value
qpid_reconnect_interval_max=0	(IntOpt) Maximum seconds between reconnection attempts
qpid_reconnect_interval_min=0	(IntOpt) Minimum seconds between reconnection attempts
qpid_reconnect_limit=0	(IntOpt) Max reconnections before giving up
qpid_reconnect_timeout=0	(IntOpt) Reconnection timeout in seconds
qpid_sasl_mechanisms=	(StrOpt) Space separated list of SASL mechanisms to use for auth
qpid_tcp_nodelay=True	(BoolOpt) Disable Nagle algorithm
qpid_username=	(StrOpt) Username for qpid connection

Table 3.14. Description of configuration options for rabbitmq

Configuration option=Default value	Description
rabbit_durable_queues=False	(BoolOpt) A boolean to determine if the queues used for messaging should be retained after a restart.
rabbit_host=localhost	(StrOpt) The host name of the rabbitmq server
rabbit_max_retries=0	(IntOpt) The maximum number of times to attempt to connect to the AMQP server.
rabbit_notification_exchange=glance	(StrOpt) Exchange name to use for connection when using rabbit strategy.
rabbit_notification_topic=notifications	(StrOpt) Topic to use for connection when using rabbit strategy.
rabbit_password=guest	(StrOpt) The password that will be used for authentication with the rabbitmq server.
rabbit_port=5672	(IntOpt) The port on which the rabbitmq server is listening
rabbit_retry_backoff=2	(IntOpt) This value multiplied by the number of connection attempts gives the amount of time in seconds to sleep between connection attempts to the AMQP server.
rabbit_retry_max_backoff=30	(IntOpt) The maximum amount of time to wait between connection attempts. The delay time will be the smaller of this value and the value of <rabbit_retry_backoff> * <the number of failed connection attempts so far>.
rabbit_use_ssl=False	(BoolOpt) A boolean value indicating if the selected rabbitmq server uses SSL.
rabbit_userid=guest	(StrOpt) The user ID for authentication with rabbitmq.

Configuration option=Default value	Description
rabbit_virtual_host=/	(StrOpt) The virtual host used in the rabbitmq connection.

Table 3.15. Description of configuration options for rbd

Configuration option=Default value	Description
rbd_store_ceph_conf=	(StrOpt) Ceph configuration file path.
rbd_store_chunk_size=4	(IntOpt) Images will be chunked into objects of this size (in megabytes). For best performance, this should be a power of two.
rbd_store_pool=rbd	(StrOpt) RADOS pool in which images are stored.
rbd_store_user=None	(StrOpt) RADOS user to authenticate as (only applicable if using cephx.)

Table 3.16. Description of configuration options for registry

Configuration option=Default value	Description
admin_password=None	(StrOpt) The administrators password.
admin_tenant_name=None	(StrOpt) The tenant name of the adminstrative user.
admin_user=None	(StrOpt) The administrators user name.
auth_region=None	(StrOpt) The region for the authentication service.
auth_strategy=noauth	(StrOpt) The strategy to use for authentication.
auth_url=None	(StrOpt) The URL to the keystone service.
registry_client_ca_file=None	(StrOpt) The path to the certifying authority cert file to use in SSL connections to the registry server.
registry_client_cert_file=None	(StrOpt) The path to the cert file to use in SSL connections to the registry server.
registry_client_insecure=False	(BoolOpt) When using SSL in connections to the registry server, do not require validation via a certifying authority.
registry_client_key_file=None	(StrOpt) The path to the key file to use in SSL connections to the registry server.
registry_client_protocol=http	(StrOpt) The protocol to use for communication with the registry server. Either http or https.
registry_client_timeout=600	(IntOpt) The period of time, in seconds, that the API server will wait for a registry request to complete. A value of 0 implies no timeout.
registry_host=0.0.0.0	(StrOpt) Address to find the registry server.
registry_port=9191	(IntOpt) Port the registry server is listening on.

Table 3.17. Description of configuration options for rpc

Configuration option=Default value	Description
allowed_rpc_exception_modules=openstack.common.exceptions,glance.common.exception	(ListOpt) Modules of exceptions that are permitted to be recreatedupon receiving exception data from an rpc call.

Table 3.18. Description of configuration options for s3

Configuration option=Default value	Description
s3_store_access_key=None	(StrOpt) The S3 query token access key.
s3_store_bucket=None	(StrOpt) The S3 bucket to be used to store the Glance data.
s3_store_bucket_url_format=subdomain	(StrOpt) The S3 calling format used to determine the bucket. Either subdomain or path can be used.

Configuration option=Default value	Description
s3_store_create_bucket_on_put=False	(BoolOpt) A boolean to determine if the S3 bucket should be created on upload if it does not exist or if an error should be returned to the user.
s3_store_host=None	(StrOpt) The host where the S3 server is listening.
s3_store_object_buffer_dir=None	(StrOpt) The local directory where uploads will be staged before they are transfered into S3.
s3_store_secret_key=None	(StrOpt) The S3 query token secret key.

Table 3.19. Description of configuration options for sheepdog

Configuration option=Default value	Description
sheepdog_store_chunk_size=64	(IntOpt) Images will be chunked into objects of this size (in megabytes). For best performance, this should be a power of two.
sheepdog_store_address=localhost	(StrOpt) IP address of sheep daemon.
sheepdog_store_port=7000	(StrOpt) Port of sheep daemon.

Table 3.20. Description of configuration options for ssl

Configuration option=Default value	Description
ca_file=None	(StrOpt) CA certificate file to use to verify connecting clients.
cert_file=None	(StrOpt) Certificate file to use when starting API server securely.
key_file=None	(StrOpt) Private key file to use when starting API server securely.

Table 3.21. Description of configuration options for swift

Configuration option=Default value	Description
swift_enable_snet=False	(BoolOpt) Whether to use ServiceNET to communicate with the Swift storage servers.
swift_store_admin_tenants=	(ListOpt) A list of tenants that will be granted read/write access on all Swift containers created by Glance in multi-tenant mode.
swift_store_auth_address=None	(StrOpt) The address where the Swift authentication service is listening.
swift_store_auth_insecure=False	(BoolOpt) If True, swiftclient won't check for a valid SSL certificate when authenticating.
swift_store_auth_version=2	(StrOpt) Version of the authentication service to use. Valid versions are 2 for keystone and 1 for swauth and rackspace
swift_store_container=glance	(StrOpt) Container within the account that the account should use for storing images in Swift.
swift_store_create_container_on_put=False	(BoolOpt) A boolean value that determines if we create the container if it does not exist.
swift_store_endpoint_type=publicURL	(StrOpt) A string giving the endpoint type of the swift service to use (publicURL, adminURL or internalURL). This setting is only used if swift_store_auth_version is 2.
swift_store_key=None	(StrOpt) Auth key for the user authenticating against the Swift authentication service.
swift_store_large_object_chunk_size=200	(IntOpt) The amount of data written to a temporary disk buffer during the process of chunking the image file.

Configuration option=Default value	Description
swift_store_large_object_size=5120	(IntOpt) The size, in MB, that Glance will start chunking image files and do a large object manifest in Swift
swift_store_multi_tenant=False	(BoolOpt) If set to True, enables multi-tenant storage mode which causes Glance images to be stored in tenant specific Swift accounts.
swift_store_region=None	(StrOpt) The region of the swift endpoint to be used for single tenant. This setting is only necessary if the tenant has multiple swift endpoints.
swift_store_service_type=object-store	(StrOpt) A string giving the service type of the swift service to use. This setting is only used if swift_store_auth_version is 2.
swift_store_user=None	(StrOpt) The user to authenticate against the Swift authentication service

Table 3.22. Description of configuration options for testing

Configuration option=Default value	Description
pydev_worker_debug_host=None	(StrOpt) The hostname/IP of the pydev process listening for debug connections
pydev_worker_debug_port=5678	(IntOpt) The port on which a pydev process is listening for connections.

Table 3.23. Description of configuration options for wsgi

Configuration option=Default value	Description
backdoor_port=None	(IntOpt) port for eventlet backdoor to listen
eventlet_hub=poll	(StrOpt) Name of eventlet hub to use. Traditionally, we have only supported 'poll', however 'selects' may be appropriate for some platforms. See http://eventlet.net/doc/hubs.html for more details.
tcp_keepidle=600	(IntOpt) The value for the socket option TCP_KEEPIDLE. This is the time in seconds that the connection must be idle before TCP starts sending keepalive probes.

Image property protection

There are currently two types of properties in the Image Service: "core properties," which are defined by the system, and "additional properties," which are arbitrary key/value pairs that can be set on an image.

With the Havana release, any such property can be protected through configuration. When you put protections on a property, it limits the users who can perform CRUD operations on the property based on their user role. The use case is to enable the cloud provider to maintain extra properties on images so typically this would be an administrator who has access to protected properties, managed with `policy.json`. The extra property could be licensing information or billing information, for example.

Properties that don't have protections defined for them will act as they do now: the administrator can control core properties, with the image owner having control over additional properties.

Property protection can be set in `/etc/glance/property-protections.conf`, using roles found in `policy.json`.

4. Networking

Table of Contents

This chapter explains the configuration options and scenarios for OpenStack Networking. For installation prerequisites, steps, and use cases, refer to corresponding chapter in the *OpenStack Installation Guide*.

Networking configuration options

The options and descriptions listed in this introduction are autogenerated from the code in the Networking service project, which provides software-defined networking between VMs run in Compute. The list contains common options, while the subsections list the options for the various networking plug-ins.

Table 4.1. Description of configuration options for common

Configuration option=Default value	Description
admin_password=None	(StrOpt) Admin password
admin_tenant_name=None	(StrOpt) Admin tenant name
admin_user=None	(StrOpt) Admin user
allowed_rpc_exception_modules=neutron.openstack.common.exceptions,nova.exception,cinder.exception,exceptions	(ListOpt) Modules of exceptions that are permitted to be recreatedupon receiving exception data from an rpc call.
auth_region=None	(StrOpt) Authentication region
auth_strategy=keystone	(StrOpt) The type of authentication to use
auth_url=None	(StrOpt) Authentication URL
base_mac=fa:16:3e:00:00:00	(StrOpt) The base MAC address Neutron will use for VIFs
bind_host=0.0.0.0	(StrOpt) The host IP to bind to
bind_port=9696	(IntOpt) The port to bind to
core_plugin=None	(StrOpt) The core plugin Neutron will use
dhcp_agent_notification=True	(BoolOpt) Allow sending resource operation notification to DHCP agent
dhcp_lease_duration=86400	(IntOpt) DHCP lease duration
disable_process_locking=False	(BoolOpt) Whether to disable inter-process locks
force_gateway_on_subnet=False	(BoolOpt) Ensure that configured gateway is on subnet
host=docwork	(StrOpt) The hostname Neutron is running on
interface_driver=None	(StrOpt) The driver used to manage the virtual interface.
lock_path=None	(StrOpt) Directory to use for lock files. Default to a temp directory
mac_generation_retries=16	(IntOpt) How many times Neutron will retry MAC generation

Configuration option=Default value	Description
max_dns_nameservers=5	(IntOpt) Maximum number of DNS nameservers
max_fixed_ips_per_port=5	(IntOpt) Maximum number of fixed ips per port
max_subnet_host_routes=20	(IntOpt) Maximum number of host routes per subnet
meta_flavor_driver_mappings=None	(StrOpt) Mapping between flavor and LinuxInterfaceDriver
network_device_mtu=None	(IntOpt) MTU setting for device.
ovs_integration_bridge=br-int	(StrOpt) Name of Open vSwitch bridge to use
ovs_use_veth=False	(BoolOpt) Uses veth for an interface or not
periodic_fuzzy_delay=5	(IntOpt) Range of seconds to randomly delay when starting the periodic task scheduler to reduce stampeding. (Disable by setting to 0)
periodic_interval=40	(IntOpt) Seconds between running periodic tasks
root_helper=sudo	(StrOpt) Root helper application.
root_helper=sudo	(StrOpt) Root helper application.
state_path=/var/lib/neutron	(StrOpt) Where to store Neutron state files. This directory must be writable by the agent.

Networking plug-ins

OpenStack Networking introduces the concept of a plug-in, which is a back-end implementation of the OpenStack Networking API. A plug-in can use a variety of technologies to implement the logical API requests. Some OpenStack Networking plug-ins might use basic Linux VLANs and IP tables, while others might use more advanced technologies, such as L2-in-L3 tunneling or OpenFlow. These sections detail the configuration options for the various plug-ins.

BigSwitch configuration options

Table 4.2. Description of configuration options for bigswitch

Configuration option=Default value	Description
add_meta_server_route=True	(BoolOpt) Flag to decide if a route to the metadata server should be injected into the VM
max_router_rules=200	(IntOpt) Maximum number of router rules
neutron_id=neutron-[hostname]	(StrOpt) User defined identifier for this Neutron deployment
node_override_vif_802.1qbg=	(ListOpt) Nova compute nodes to manually set VIF type to 802.1qbg
node_override_vif_802.1qbh=	(ListOpt) Nova compute nodes to manually set VIF type to 802.1qbh
node_override_vif_binding_failed=	(ListOpt) Nova compute nodes to manually set VIF type to binding_failed
node_override_vif_bridge=	(ListOpt) Nova compute nodes to manually set VIF type to bridge
node_override_vif_hyperv=	(ListOpt) Nova compute nodes to manually set VIF type to hyperv
node_override_vif_ivs=	(ListOpt) Nova compute nodes to manually set VIF type to ivs
node_override_vif_other=	(ListOpt) Nova compute nodes to manually set VIF type to other

Configuration option=Default value	Description
node_override_vif_ovs=	(ListOpt) Nova compute nodes to manually set VIF type to ovs
node_override_vif_unbound=	(ListOpt) Nova compute nodes to manually set VIF type to unbound
server_auth=username:password	(StrOpt) The username and password for authenticating against the BigSwitch or Floodlight controller.
server_ssl=False	(BoolOpt) If True, Use SSL when connecting to the BigSwitch or Floodlight controller.
server_timeout=10	(IntOpt) Maximum number of seconds to wait for proxy request to connect and complete.
servers=localhost:8800	(StrOpt) A comma separated list of BigSwitch or Floodlight servers and port numbers. The plugin proxies the requests to the BigSwitch/Floodlight server, which performs the networking configuration. Note that only one server is needed per deployment, but you may wish to deploy multiple servers to support failover.
sync_data=False	(BoolOpt) Sync data on connect
tenant_default_router_rule=['*:any:any:permit']	(MultiStrOpt) The default router rules installed in new tenant routers. Repeat the config option for each rule. Format is <tenant>:<source>:<destination>:<action> Use an * to specify default for all tenants.
vif_type=ovs	(StrOpt) Virtual interface type to configure on Nova compute nodes
vif_types=unbound,binding_failed,ovs,ivs,bridge,802.1qbg,802.1qbh,hyperv,other	(ListOpt) List of allowed vif_type values.

Brocade configuration options

Table 4.3. Description of configuration options for brocade

Configuration option=Default value	Description
address=	(StrOpt) The address of the host to SSH to
ostype=NOS	(StrOpt) Currently unused
password=None	(StrOpt) HTTP password for authentication
physical_interface=eth0	(StrOpt) The network interface to use when creatinga port
username=None	(StrOpt) HTTP username for authentication

CISCO configuration options

Table 4.4. Description of configuration options for cisco

Configuration option=Default value	Description
default_network_profile=default_network_profile	(StrOpt) N1K default network profile
default_policy_profile=service_profile	(StrOpt) N1K default policy profile
host=[hostname]	(StrOpt) The hostname Neutron is running on
model_class=neutron.plugins.cisco.models.virt_phy_sw_v2.VirtualPhysicalSwitchModelV2	(StrOpt) Model Class
network_node_policy_profile=dhcp_pp	(StrOpt) N1K policy profile for network node
network_vlan_ranges=vlan:1:4095	(StrOpt) N1K Network VLAN Ranges
nexus_driver=neutron.plugins.cisco.test.nexus.fake_nexus_driver.CiscoNEXUSFakeDriver	(StrOpt) Cisco NEXUS Fake Driver
nexus_plugin=neutron.plugins.cisco.nexus.cisco_nexus_plugin_v2.NexusPlugin	(StrOpt) Nexus Switch to use
poll_duration=10	(StrOpt) N1K Policy profile polling duration in seconds

Configuration option=Default value	Description
provider_vlan_auto_create=True	(BoolOpt) Provider VLANs are automatically created as needed on the Nexus switch
provider_vlan_auto_trunk=True	(BoolOpt) Provider VLANs are automatically trunked as needed on the ports of the Nexus switch
provider_vlan_name_prefix=p-	(StrOpt) VLAN Name prefix for provider vlans
svi_round_robin=False	(BoolOpt) Distribute SVI interfaces over all switches
svi_round_robin=False	(BoolOpt) Distribute SVI interfaces over all switches
vlan_name_prefix=q-	(StrOpt) VLAN Name prefix
vlan_name_prefix=q-	(StrOpt) VLAN Name prefix
vswitch_plugin=neutron.plugins.openvswitch.ovs_neutron_plugin.OVSNeutronPluginV2	(StrOpt) Vswitch plugin to use
vxlan_id_ranges=5000:10000	(StrOpt) N1K VXLAN ID Ranges

CloudBase Hyper-V plug-in configuration options (deprecated)

Table 4.5. Description of configuration options for hyperv

Configuration option=Default value	Description
network_vlan_ranges=vlan:1:4095	(StrOpt) N1K Network VLAN Ranges
network_vlan_ranges=	(ListOpt) List of <physical_network>:<vlan_min>:<vlan_max> or <physical_network>

CloudBase Hyper-V Agent configuration options

Table 4.6. Description of configuration options for hyperv_agent

Configuration option=Default value	Description
enable_metrics_collection=False	(BoolOpt) Enables metrics collections for switch ports by using Hyper-V's metric APIs. Collected data can by retrieved by other apps and services, e.g.: Ceilometer. Requires Hyper-V / Windows Server 2012 and above
force_hyperv_utils_v1=False	(BoolOpt) Force V1 WMI utility classes
local_network_vswitch=private	(StrOpt) Private vswitch name used for local networks
physical_network_vswitch_mappings=	(ListOpt) List of <physical_network>:<vswitch> where the physical networks can be expressed with wildcards, e.g.: ."*":external"
polling_interval=2	(IntOpt) The number of seconds the agent will wait between polling for local device changes.

Linux bridge plug-in configuration options (deprecated)

Table 4.7. Description of configuration options for linuxbridge

Configuration option=Default value	Description
enable_vxlan=False	(BoolOpt) Enable VXLAN on the agent. Can be enabled when agent is managed by ml2 plugin using linuxbridge mechanism driver
l2_population=False	(BoolOpt) Use ml2 l2population mechanism driver to learn remote mac and IPs and improve tunnel scalability
network_vlan_ranges=	(ListOpt) List of <physical_network>:<vlan_min>:<vlan_max> or <physical_network>

Configuration option=Default value	Description
physical_interface_mappings=	(ListOpt) List of <physical_network>:<physical_interface>
physical_interface_mappings=	(ListOpt) List of <physical_network>:<physical_interface>
tenant_network_type=local	(StrOpt) Network type for tenant networks (local, flat, vlan or none)
tenant_network_type=local	(StrOpt) Network type for tenant networks (local, vlan, or none)
tenant_network_type=vlan	(StrOpt) Network type for tenant networks (local, ib, vlan, or none)
tenant_network_type=local	(StrOpt) N1K Tenant Network Type
tenant_network_type=local	(StrOpt) Network type for tenant networks (local, vlan, gre, vxlan, or none)
tos=None	(IntOpt) TOS for vxlan interface protocol packets.
ttl=None	(IntOpt) TTL for vxlan interface protocol packets.
vxlan_group=224.0.0.1	(StrOpt) Multicast group for vxlan interface.
vxlan_group=None	(StrOpt) Multicast group for VXLAN. If unset, disables VXLAN multicast mode.

Linux bridge Agent configuration options

Table 4.8. Description of configuration options for linuxbridge_agent

Configuration option=Default value	Description
enable_vxlan=False	(BoolOpt) Enable VXLAN on the agent. Can be enabled when agent is managed by ml2 plugin using linuxbridge mechanism driver
l2_population=False	(BoolOpt) Extension to use alongside ml2 plugin's l2population mechanism driver. It enables the plugin to populate VXLAN forwarding table.
l2_population=False	(BoolOpt) Use ml2 l2population mechanism driver to learn remote mac and IPs and improve tunnel scalability
physical_interface_mappings=	(ListOpt) List of <physical_network>:<physical_interface>
physical_interface_mappings=	(ListOpt) List of <physical_network>:<physical_interface>
tos=None	(IntOpt) TOS for vxlan interface protocol packets.
ttl=None	(IntOpt) TTL for vxlan interface protocol packets.
vxlan_group=224.0.0.1	(StrOpt) Multicast group for vxlan interface.
vxlan_group=None	(StrOpt) Multicast group for VXLAN. If unset, disables VXLAN multicast mode.

Mellanox configuration options

Table 4.9. Description of configuration options for mlnx

Configuration option=Default value	Description
daemon_endpoint=tcp://127.0.0.1:5001	(StrOpt) eswitch daemon end point
network_vlan_ranges=default:1000	(ListOpt) List of <physical_network>:<vlan_min>:<vlan_max> or <physical_network>
request_timeout=3000	(IntOpt) The number of milliseconds the agent will wait for response on request to daemon.
vnic_type=mlnx_direct	(StrOpt) Type of VM network interface: mlnx_direct or hostdev

Meta Plug-in configuration options

The Meta Plug-in allows you to use multiple plug-ins at the same time.

Table 4.10. Description of configuration options for meta

Configuration option=Default value	Description
default_flavor=	(StrOpt) Default flavor to use
default_l3_flavor=	(StrOpt) Default L3 flavor to use
extension_map=	(StrOpt) A list of extensions, per plugin, to load.
l3_plugin_list=	(StrOpt) List of L3 plugins to load
plugin_list=	(StrOpt) List of plugins to load
supported_extension_aliases=	(StrOpt) Supported extension aliases

Modular Layer 2 (ml2) configuration options

The Modular Layer 2 (ml2) plug-in has two components, network types and mechanisms, that can be configured separately. Such configuration options are described in the subsections.

Table 4.11. Description of configuration options for ml2

Configuration option=Default value	Description
mechanism_drivers=	(ListOpt) An ordered list of networking mechanism driver entrypoints to be loaded from the neutron.ml2.mechanism_drivers namespace.
tenant_network_types=local	(ListOpt) Ordered list of network_types to allocate as tenant networks.
type_drivers=local,flat,vlan,gre,vxlan	(ListOpt) List of network type driver entrypoints to be loaded from the neutron.ml2.type_drivers namespace.

Modular Layer 2 (ml2) Flat Type configuration options

Table 4.12. Description of configuration options for ml2_flat

Configuration option=Default value	Description
flat_networks=	(ListOpt) List of physical_network names with which flat networks can be created. Use * to allow flat networks with arbitrary physical_network names.
network_vlan_ranges=	(ListOpt) List of <physical_network>:<vlan_min>:<vlan_max> or <physical_network> specifying physical_network names usable for VLAN provider and tenant networks, as well as ranges of VLAN tags on each available for allocation to tenant networks.

Modular Layer 2 (ml2) VXLAN Type configuration options

Table 4.13. Description of configuration options for ml2_vxlan

Configuration option=Default value	Description
vni_ranges=	(ListOpt) Comma-separated list of <vni_min>:<vni_max> tuples enumerating ranges of VXLAN VNI IDs that are available for tenant network allocation

Configuration option=Default value	Description
vxlan_group=224.0.0.1	(StrOpt) Multicast group for vxlan interface.
vxlan_group=None	(StrOpt) Multicast group for VXLAN. If unset, disables VXLAN multicast mode.

Modular Layer 2 (ml2) Arista Mechanism configuration options

Table 4.14. Description of configuration options for ml2_arista

Configuration option=Default value	Description
eapi_host=	(StrOpt) Arista EOS IP address. This is required field.If not set, all communications to Arista EOSwill fail
eapi_password=	(StrOpt) Password for Arista EOS. This is required field.if not set, all communications to Arista EOSwill fail
eapi_username=	(StrOpt) Username for Arista EOS. This is required field.if not set, all communications to Arista EOSwill fail
region_name=RegionOne	(StrOpt) Defines Region Name that is assigned to this OpenStackController. This is useful when multipleOpenStack/Neutron controllers are managing the sameArista HW clusters. Note that this name must match withthe region name registered (or known) to keystoneservice. Authentication with Keysotne is performed byEOS. This is optional. If not set, a value of"RegionOne" is assumed
sync_interval=180	(IntOpt) Sync interval in seconds between Neutron plugin andEOS. This interval defines how often thesynchronization is performed. This is an optionalfield. If not set, a value of 180 seconds is assumed
use_fqdn=True	(BoolOpt) Defines if hostnames are sent to Arista EOS as FQDNs("node1.domain.com") or as short names ("node1").This is optional. If not set, a value of "True"is assumed.

Modular Layer 2 (ml2) Cisco Mechanism configuration options

Table 4.15. Description of configuration options for ml2_cisco

Configuration option=Default value	Description
managed_physical_network=None	(StrOpt) The physical network managed by the switches.

Modular Layer 2 (ml2) L2 Population Mechanism configuration options

Table 4.16. Description of configuration options for ml2_l2pop

Configuration option=Default value	Description
agent_boot_time=180	(IntOpt) Delay within which agent is expected to update existing ports whent it restarts

Modular Layer 2 (ml2) Tail-f NCS Mechanism configuration options

Table 4.17. Description of configuration options for ml2_ncs

Configuration option=Default value	Description
timeout=10	(IntOpt) HTTP timeout in seconds.
url=None	(StrOpt) HTTP URL of Tail-f NCS REST interface.

MidoNet configuration options

Table 4.18. Description of configuration options for midonet

Configuration option=Default value	Description
midonet_host_uuid_path=/etc/midolman/host_uuid.properties	(StrOpt) Path to midonet host uuid file
midonet_uri=http://localhost:8080/midonet-api	(StrOpt) MidoNet API server URI.
mode=dev	(StrOpt) Operational mode. Internal dev use only.
password=passw0rd	(StrOpt) MidoNet admin password.
project_id=77777777-7777-7777-7777-777777777777	(StrOpt) ID of the project that MidoNet admin userbelongs to.
provider_router_id=None	(StrOpt) Virtual provider router ID.
username=admin	(StrOpt) MidoNet admin username.

NEC configuration options

Table 4.19. Description of configuration options for nec

Configuration option=Default value	Description
cert_file=None	(StrOpt) Certificate file
default_router_provider=l3-agent	(StrOpt) Default router provider to use.
driver=trema	(StrOpt) Driver to use
enable_packet_filter=True	(BoolOpt) Enable packet filter
host=docwork	(StrOpt) The hostname Neutron is running on
port=6379	(IntOpt) Use this port to connect to redis host.
port=8888	(StrOpt) Port to connect to
router_providers=l3-agent,openflow	(ListOpt) List of enabled router providers.
use_ssl=False	(BoolOpt) Enable SSL on the API server

Nicira NVP configuration options

Table 4.20. Description of configuration options for nicira

Configuration option=Default value	Description
agent_mode=agent	(StrOpt) The mode used to implement DHCP/metadata services.
always_read_status=False	(BoolOpt) Always read operational status from backend on show operations. Enabling this option might slow down the system.
concurrent_connections=10	(IntOpt) Maximum concurrent connections to each NVP controller.
datacenter_moid=None	(StrOpt) Optional parameter identifying the ID of datacenter to deploy NSX Edges
datastore_id=None	(StrOpt) Optional parameter identifying the ID of datastore to deploy NSX Edges
default_interface_name=breth0	(StrOpt) Name of the interface on a L2 Gateway transport nodewhich should be used by default when setting up a network connection
default_l2_gw_service_uuid=None	(StrOpt) Unique identifier of the NVP L2 Gateway service which will be used by default for network gateways

Configuration option=Default value	Description
default_l3_gw_service_uuid=None	(StrOpt) Unique identifier of the NVP L3 Gateway service which will be used for implementing routers and floating IPs
default_transport_type=stt	(StrOpt) The default network tranport type to use (stt, gre, bridge, ipsec_gre, or ipsec_stt)
default_tz_uuid=None	(StrOpt) This is uuid of the default NVP Transport zone that will be used for creating tunneled isolated "Neutron" networks. It needs to be created in NVP before starting Neutron with the nvp plugin.
deployment_container_id=None	(StrOpt) Optional parameter identifying the ID of datastore to deploy NSX Edges
external_network=None	(StrOpt) Network ID for physical network connectivity
http_timeout=10	(IntOpt) Time before aborting a request
manager_uri=None	(StrOpt) uri for vsm
max_lp_per_bridged_ls=5000	(IntOpt) Maximum number of ports of a logical switch on a bridged transport zone (default 5000)
max_lp_per_overlay_ls=256	(IntOpt) Maximum number of ports of a logical switch on an overlay transport zone (default 256)
max_random_sync_delay=0	(IntOpt) Maximum value for the additional random delay in seconds between runs of the state synchronization task
metadata_mode=access_network	(StrOpt) If set to access_network this enables a dedicated connection to the metadata proxy for metadata server access via Neutron router. If set to dhcp_host_route this enables host route injection via the dhcp agent. This option is only useful if running on a host that does not support namespaces otherwise access_network should be used.
min_chunk_size=500	(IntOpt) Minimum number of resources to be retrieved from NVP during state synchronization
min_sync_req_delay=10	(IntOpt) Minimum delay, in seconds, between two state synchronization queries to NVP. It must not exceed state_sync_interval
nvp_cluster_uuid=None	(StrOpt) Optional paramter identifying the UUID of the cluster in NVP. This can be retrieved from NVP management console "admin" section.
nvp_controllers=None	(ListOpt) Lists the NVP controllers in this cluster
nvp_gen_timeout=-1	(IntOpt) Number of seconds a generation id should be valid for (default -1 meaning do not time out)
nvp_password=admin	(StrOpt) Password for NVP controllers in this cluster
nvp_user=admin	(StrOpt) User name for NVP controllers in this cluster
redirects=2	(IntOpt) Number of times a redirect should be followed
req_timeout=30	(IntOpt) Total time limit for a cluster request
resource_pool_id=default	(StrOpt) Shared resource pool id
retries=2	(IntOpt) Number of time a request should be retried
state_sync_interval=120	(IntOpt) Interval in seconds between runs of the state synchronization task. Set it to 0 to disable it
task_status_check_interval=2000	(IntOpt) Task status check interval
user=admin	(StrOpt) User name for vsm

Open vSwitch plug-in configuration options (deprecated)

Table 4.21. Description of configuration options for openvswitch

Configuration option=Default value	Description
network_vlan_ranges=	(ListOpt) List of <physical_network>:<vlan_min>:<vlan_max> or <physical_network>

Open vSwitch Agent configuration options

Table 4.22. Description of configuration options for openvswitch_agent

Configuration option=Default value	Description
bridge_mappings=	(StrOpt) N1K Bridge Mappings
bridge_mappings=	(ListOpt) List of <physical_network>:<bridge>
enable_tunneling=True	(BoolOpt) N1K Enable Tunneling
enable_tunneling=False	(BoolOpt) Enable tunneling support
int_peer_patch_port=patch-tun	(StrOpt) Peer patch port in integration bridge for tunnel bridge
integration_bridge=br-int	(StrOpt) N1K Integration Bridge
integration_bridge=br-int	(StrOpt) Integration bridge to use
l2_population=False	(BoolOpt) Extension to use alongside ml2 plugin's l2population mechanism driver. It enables the plugin to populate VXLAN forwarding table.
l2_population=False	(BoolOpt) Use ml2 l2population mechanism driver to learn remote mac and IPs and improve tunnel scalability
local_ip=10.0.0.3	(StrOpt) N1K Local IP
local_ip=	(StrOpt) Local IP address of the VXLAN endpoints.
local_ip=	(StrOpt) Local IP address of GRE tunnel endpoints.
tun_peer_patch_port=patch-int	(StrOpt) Peer patch port in tunnel bridge for integration bridge
tunnel_bridge=br-tun	(StrOpt) N1K Tunnel Bridge
tunnel_bridge=br-tun	(StrOpt) Tunnel bridge to use
tunnel_id_ranges=	(ListOpt) List of <tun_min>:<tun_max>
tunnel_id_ranges=	(ListOpt) Comma-separated list of <tun_min>:<tun_max> tuples enumerating ranges of GRE tunnel IDs that are available for tenant network allocation
tunnel_type=	(StrOpt) The type of tunnels to use when utilizing tunnels, either 'gre' or 'vxlan'
tunnel_types=	(ListOpt) Network types supported by the agent (gre and/or vxlan)
veth_mtu=None	(IntOpt) MTU size of veth interfaces
vxlan_udp_port=4789	(IntOpt) The UDP port to use for VXLAN tunnels.

PLUMgrid configuration options

Table 4.23. Description of configuration options for plumgrid

Configuration option=Default value	Description
director_server=localhost	(StrOpt) PLUMgrid Director server to connect to
director_server_port=8080	(StrOpt) PLUMgrid Director server port to connect to

Configuration option=Default value	Description
password=	(StrOpt) The SSH password to use
servertimeout=5	(IntOpt) PLUMgrid Director server timeout
username=username	(StrOpt) PLUMgrid Director admin username

Ryu configuration options

Table 4.24. Description of configuration options for ryu

Configuration option=Default value	Description
openflow_rest_api=127.0.0.1:8080	(StrOpt) OpenFlow REST API location
ovsdb_interface=None	(StrOpt) OVSDB interface to connect to
ovsdb_ip=None	(StrOpt) OVSDB IP to connect to
ovsdb_port=6634	(IntOpt) OVSDB port to connect to
tunnel_interface=None	(StrOpt) Tunnel interface to use
tunnel_ip=None	(StrOpt) Tunnel IP to use
tunnel_key_max=16777215	(IntOpt) Maximum tunnel ID to use
tunnel_key_min=1	(IntOpt) Minimum tunnel ID to use

Configure the Oslo RPC messaging system

OpenStack projects use an open standard for messaging middleware known as AMQP. This messaging middleware enables the OpenStack services that run on multiple servers to talk to each other. OpenStack Oslo RPC supports three implementations of AMQP: RabbitMQ, Qpid, and ZeroMQ.

Configure RabbitMQ

OpenStack Oslo RPC uses RabbitMQ by default. Use these options to configure the RabbitMQ message system. The `rpc_backend` option is optional as long as RabbitMQ is the default messaging system. However, if it is included the configuration, you must set it to `neutron.openstack.common.rpc.impl_kombu`.

```
rpc_backend=neutron.openstack.common.rpc.impl_kombu
```

Use these options to configure the RabbitMQ messaging system. You can configure messaging communication for different installation scenarios, tune retries for RabbitMQ, and define the size of the RPC thread pool. To monitor notifications through RabbitMQ, you must set the `notification_driver` option to `neutron.notifier.rabbit_notifier` in the `neutron.conf` file:

Table 4.25. Description of configuration options for rabbitmq

Configuration option=Default value	Description
rabbit_ha_queues=False	(BoolOpt) use H/A queues in RabbitMQ (x-ha-policy: all).You need to wipe RabbitMQ database when changing this option.
rabbit_host=localhost	(StrOpt) The RabbitMQ broker address where a single node is used
rabbit_hosts=$rabbit_host:$rabbit_port	(ListOpt) RabbitMQ HA cluster host:port pairs

Configuration option=Default value	Description
rabbit_max_retries=0	(IntOpt) maximum retries with trying to connect to RabbitMQ (the default of 0 implies an infinite retry count)
rabbit_password=guest	(StrOpt) the RabbitMQ password
rabbit_port=5672	(IntOpt) The RabbitMQ broker port where a single node is used
rabbit_retry_backoff=2	(IntOpt) how long to backoff for between retries when connecting to RabbitMQ
rabbit_retry_interval=1	(IntOpt) how frequently to retry connecting with RabbitMQ
rabbit_use_ssl=False	(BoolOpt) connect over SSL for RabbitMQ
rabbit_userid=guest	(StrOpt) the RabbitMQ userid
rabbit_virtual_host=/	(StrOpt) the RabbitMQ virtual host

Table 4.26. Description of configuration options for kombu

Configuration option=Default value	Description
kombu_ssl_ca_certs=	(StrOpt) SSL certification authority file (valid only if SSL enabled)
kombu_ssl_certfile=	(StrOpt) SSL cert file (valid only if SSL enabled)
kombu_ssl_keyfile=	(StrOpt) SSL key file (valid only if SSL enabled)
kombu_ssl_version=	(StrOpt) SSL version to use (valid only if SSL enabled)

Configure Qpid

Use these options to configure the Qpid messaging system for OpenStack Oslo RPC. Qpid is not the default messaging system, so you must enable it by setting the `rpc_backend` option in the `neutron.conf` file:

```
rpc_backend=neutron.openstack.common.rpc.impl_qpid
```

This critical option points the compute nodes to the Qpid broker (server). Set the `qpid_hostname` option to the host name where the broker runs in the `neutron.conf` file.

Note

The `--qpid_hostname` option accepts a host name or IP address value.

```
qpid_hostname=hostname.example.com
```

If the Qpid broker listens on a port other than the AMQP default of `5672`, you must set the `qpid_port` option to that value:

```
qpid_port=12345
```

If you configure the Qpid broker to require authentication, you must add a user name and password to the configuration:

```
qpid_username=username
qpid_password=password
```

By default, TCP is used as the transport. To enable SSL, set the `qpid_protocol` option:

```
qpid_protocol=ssl
```

Use these additional options to configure the Qpid messaging driver for OpenStack Oslo RPC. These options are used infrequently.

Table 4.27. Description of configuration options for qpid

Configuration option=Default value	Description
qpid_heartbeat=60	(IntOpt) Seconds between connection keepalive heartbeats
qpid_hostname=localhost	(StrOpt) Qpid broker hostname
qpid_hosts=$qpid_hostname:$qpid_port	(ListOpt) Qpid HA cluster host:port pairs
qpid_password=	(StrOpt) Password for qpid connection
qpid_port=5672	(IntOpt) Qpid broker port
qpid_protocol=tcp	(StrOpt) Transport to use, either 'tcp' or 'ssl'
qpid_sasl_mechanisms=	(StrOpt) Space separated list of SASL mechanisms to use for auth
qpid_tcp_nodelay=True	(BoolOpt) Disable Nagle algorithm
qpid_topology_version=1	(IntOpt) The qpid topology version to use. Version 1 is what was originally used by impl_qpid. Version 2 includes some backwards-incompatible changes that allow broker federation to work. Users should update to version 2 when they are able to take everything down, as it requires a clean break.
qpid_username=	(StrOpt) Username for qpid connection

Configure ZeroMQ

Use these options to configure the ZeroMQ messaging system for OpenStack Oslo RPC. ZeroMQ is not the default messaging system, so you must enable it by setting the `rpc_backend` option in the `neutron.conf` file:

Table 4.28. Description of configuration options for zeromq

Configuration option=Default value	Description
rpc_zmq_bind_address=*	(StrOpt) ZeroMQ bind address. Should be a wildcard (*), an ethernet interface, or IP. The "host" option should point or resolve to this address.
rpc_zmq_contexts=1	(IntOpt) Number of ZeroMQ contexts, defaults to 1
rpc_zmq_host=[hostname]	(StrOpt) Name of this node. Must be a valid hostname, FQDN, or IP address. Must match "host" option, if running Nova.
rpc_zmq_ipc_dir=/var/run/openstack	(StrOpt) Directory for holding IPC sockets
rpc_zmq_matchmaker=neutron.openstack.common.rpc.matchmaker.MatchMakerLocalhost	(StrOpt) MatchMaker driver
rpc_zmq_port=9501	(IntOpt) ZeroMQ receiver listening port

Configuration option=Default value	Description
rpc_zmq_topic_backlog=None	(IntOpt) Maximum number of ingress messages to locally buffer per topic. Default is unlimited.

Configure messaging

Use these common options to configure the RabbitMQ, Qpid, and ZeroMq messaging drivers:

Table 4.29. Description of configuration options for rpc

Configuration option=Default value	Description
amqp_auto_delete=False	(BoolOpt) Auto-delete queues in amqp.
amqp_durable_queues=False	(BoolOpt) Use durable queues in amqp.
control_exchange=neutron	(StrOpt) AMQP exchange to connect to if using RabbitMQ or Qpid
host=[hostname]	(StrOpt) The hostname Neutron is running on
matchmaker_heartbeat_freq=300	(IntOpt) Heartbeat frequency
matchmaker_heartbeat_ttl=600	(IntOpt) Heartbeat time-to-live.
password=	(StrOpt) The SSH password to use
port=8888	(StrOpt) Port to connect to
ringfile=/etc/oslo/matchmaker_ring.json	(StrOpt) Matchmaker ring file (JSON)
rpc_backend=neutron.openstack.common.rpc.impl_kombu	(StrOpt) The messaging module to use, defaults to kombu.
rpc_cast_timeout=30	(IntOpt) Seconds to wait before a cast expires (TTL). Only supported by impl_zmq.
rpc_conn_pool_size=30	(IntOpt) Size of RPC connection pool
rpc_response_timeout=60	(IntOpt) Seconds to wait for a response from call or multicall
rpc_support_old_agents=False	(BoolOpt) Enable server RPC compatibility with old agents
rpc_thread_pool_size=64	(IntOpt) Size of RPC thread pool
topics=notifications	(ListOpt) AMQP topic(s) used for openstack notifications

Table 4.30. Description of configuration options for notifier

Configuration option=Default value	Description
default_notification_level=INFO	(StrOpt) Default notification level for outgoing notifications
default_publisher_id=$host	(StrOpt) Default publisher_id for outgoing notifications
notification_driver=[]	(MultiStrOpt) Driver or drivers to handle sending notifications
notification_topics=notifications	(ListOpt) AMQP topic used for openstack notifications

Agent

Use the following options to alter agent-related settings.

Table 4.31. Description of configuration options for agent

Configuration option=Default value	Description
agent_down_time=5	(IntOpt) Seconds to regard the agent is down.

Configuration option=Default value	Description
external_pids=$state_path/external/pids	(StrOpt) Location to store child pid files
interface_driver=None	(StrOpt) The driver used to manage the virtual interface.
report_interval=4	(FloatOpt) Seconds between nodes reporting state to server
use_namespaces=True	(BoolOpt) Allow overlapping IP.

API

Use the following options to alter API-related settings.

Table 4.32. Description of configuration options for api

Configuration option=Default value	Description
allow_bulk=True	(BoolOpt) Allow the usage of the bulk API
allow_pagination=False	(BoolOpt) Allow the usage of the pagination
allow_sorting=False	(BoolOpt) Allow the usage of the sorting
api_extensions_path=	(StrOpt) The path for API extensions
api_paste_config=api-paste.ini	(StrOpt) The API paste config file to use
pagination_max_limit=-1	(StrOpt) The maximum number of items returned in a single response, value was 'infinite' or negative integer means no limit
run_external_periodic_tasks=True	(BoolOpt) Some periodic tasks can be run in a separate process. Should we run them here?
service_plugins=	(ListOpt) The service plugins Neutron will use
service_provider=[]	(MultiStrOpt) Defines providers for advanced services using the format: <service_type>:<name>:<driver>[:default]

Database

Use the following options to alter Database-related settings.

Table 4.33. Description of configuration options for db

Configuration option=Default value	Description
backend=sqlalchemy	(StrOpt) The backend to use for db
connection=sqlite://	(StrOpt) The SQLAlchemy connection string used to connect to the database
connection_debug=0	(IntOpt) Verbosity of SQL debugging information. 0=None, 100=Everything
connection_trace=False	(BoolOpt) Add python stack traces to SQL as comment strings
dhcp_agents_per_network=1	(IntOpt) Number of DHCP agents scheduled to host a network.
idle_timeout=3600	(IntOpt) timeout before idle sql connections are reaped
max_overflow=20	(IntOpt) If set, use this value for max_overflow with sqlalchemy
max_pool_size=10	(IntOpt) Maximum number of SQL connections to keep open in a pool
max_retries=10	(IntOpt) maximum db connection retries during startup. (setting -1 implies an infinite retry count)

Configuration option=Default value	Description
min_pool_size=1	(IntOpt) Minimum number of SQL connections to keep open in a pool
pool_timeout=10	(IntOpt) If set, use this value for pool_timeout with sqlalchemy
retry_interval=10	(IntOpt) interval between retries of opening a sql connection
slave_connection=	(StrOpt) The SQLAlchemy connection string used to connect to the slave database
sqlite_db=	(StrOpt) the filename to use with sqlite
sqlite_synchronous=True	(BoolOpt) If true, use synchronous mode for sqlite
use_tpool=False	(BoolOpt) Enable the experimental use of thread pooling for all DB API calls

Logging

Use the following options to alter logging settings.

Table 4.34. Description of configuration options for logging

Configuration option=Default value	Description
debug=False	(BoolOpt) Print debugging output (set logging level to DEBUG instead of default WARNING level).
default_log_levels=amqplib=WARN,sqlalchemy=WARN,boto=WARN,suds=INFO,keystone=INFO,eventlet.wsgi.server=WARN	(ListOpt) list of logger=LEVEL pairs
fatal_deprecations=False	(BoolOpt) make deprecations fatal
instance_format=[instance: %(uuid)s]	(StrOpt) If an instance is passed with the log message, format it like this
instance_uuid_format=[instance: %(uuid)s]	(StrOpt) If an instance UUID is passed with the log message, format it like this
log_config=None	(StrOpt) If this option is specified, the logging configuration file specified is used and overrides any other logging options specified. Please see the Python logging module documentation for details on logging configuration files.
log_date_format=%Y-%m-%d %H:%M:%S	(StrOpt) Format string for %%(asctime)s in log records. Default: %(default)s
log_dir=None	(StrOpt) (Optional) The base directory used for relative –log-file paths
log_file=None	(StrOpt) (Optional) Name of log file to output to. If no default is set, logging will go to stdout.
log_format=None	(StrOpt) A logging.Formatter log message format string which may use any of the available logging.LogRecord attributes. This option is deprecated. Please use logging_context_format_string and logging_default_format_string instead.
logging_context_format_string=%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [%(request_id)s %(user)s %(tenant)s] %(instance)s%(message)s	(StrOpt) format string to use for log messages with context
logging_debug_format_suffix=%(funcName)s %(pathname)s:%(lineno)d	(StrOpt) data to append to log format when level is DEBUG
logging_default_format_string=%(asctime)s.%(msecs)03d %(process)d %(levelname)s %(name)s [-] %(instance)s %(message)s	(StrOpt) format string to use for log messages without context
logging_exception_prefix=%(asctime)s.%(msecs)03d %(process)d TRACE %(name)s %(instance)s	(StrOpt) prefix each line of exception output with this format

Configuration option=Default value	Description
publish_errors=False	(BoolOpt) publish error events
syslog_log_facility=LOG_USER	(StrOpt) syslog facility to receive log lines
use_stderr=True	(BoolOpt) Log output to standard error
use_syslog=False	(BoolOpt) Use syslog for logging.
verbose=False	(BoolOpt) Print more verbose output (set logging level to INFO instead of default WARNING level).

Metadata Agent

Use the following options in the `metadata_agent.ini` file for the Metadata agent.

Table 4.35. Description of configuration options for metadata

Configuration option=Default value	Description
auth_strategy=keystone	(StrOpt) The type of authentication to use

Policy

Use the following options in the `neutron.conf` file to change policy settings.

Table 4.36. Description of configuration options for policy

Configuration option=Default value	Description
allow_overlapping_ips=False	(BoolOpt) Allow overlapping IP support in Neutron
policy_file=policy.json	(StrOpt) The policy file to use

Quotas

Use the following options in the `neutron.conf` file for the quota system.

Table 4.37. Description of configuration options for quotas

Configuration option=Default value	Description
default_quota=-1	(IntOpt) Default number of resource allowed per tenant, minus for unlimited
max_routes=30	(IntOpt) Maximum number of routes
quota_driver=neutron.db.quota_db.DbQuotaDriver	(StrOpt) Default driver to use for quota checks
quota_firewall=1	(IntOpt) Number of firewalls allowed per tenant, -1 for unlimited
quota_firewall_policy=1	(IntOpt) Number of firewall policies allowed per tenant, -1 for unlimited
quota_firewall_rule=-1	(IntOpt) Number of firewall rules allowed per tenant, -1 for unlimited
quota_floatingip=50	(IntOpt) Number of floating IPs allowed per tenant, -1 for unlimited
quota_items=network,subnet,port	(ListOpt) Resource name(s) that are supported in quota features
quota_network=10	(IntOpt) Number of networks allowed per tenant,minus for unlimited
quota_network_gateway=5	(IntOpt) Number of network gateways allowed per tenant, -1 for unlimited

Configuration option=Default value	Description
quota_packet_filter=100	(IntOpt) Number of packet_filters allowed per tenant, -1 for unlimited
quota_port=50	(IntOpt) Number of ports allowed per tenant, minus for unlimited
quota_router=10	(IntOpt) Number of routers allowed per tenant, -1 for unlimited
quota_security_group=10	(IntOpt) Number of security groups allowed per tenant,-1 for unlimited
quota_security_group_rule=100	(IntOpt) Number of security rules allowed per tenant, -1 for unlimited
quota_subnet=10	(IntOpt) Number of subnets allowed per tenant, minus for unlimited

Scheduler

Use the following options in the `neutron.conf` file to change scheduler settings.

Table 4.38. Description of configuration options for scheduler

Configuration option=Default value	Description
network_auto_schedule=True	(BoolOpt) Allow auto scheduling networks to DHCP agent.
network_scheduler_driver=neutron.scheduler.dhcp_agent_scheduler.ChanceScheduler	(StrOpt) Driver to use for scheduling network to DHCP agent
router_auto_schedule=True	(BoolOpt) Allow auto scheduling of routers to L3 agent.
router_scheduler_driver=neutron.scheduler.l3_agent_scheduler.ChanceScheduler	(StrOpt) Driver to use for scheduling router to a default L3 agent

Security Groups

Use the following options in the configuration file for your driver to change security group settings.

Table 4.39. Description of configuration options for securitygroups

Configuration option=Default value	Description
firewall_driver=neutron.agent.firewall.NoopFirewallDriver	(StrOpt) Driver for Security Groups Firewall

SSL

Use the following options in the `neutron.conf` file to enable SSL.

Table 4.40. Description of configuration options for ssl

Configuration option=Default value	Description
key_file=None	(StrOpt) Key file
ssl_ca_file=None	(StrOpt) CA certificate file to use to verify connecting clients
ssl_cert_file=None	(StrOpt) Certificate file to use when starting the server securely
ssl_key_file=None	(StrOpt) Private key file to use when starting the server securely

Configuration option=Default value	Description
use_ssl=False	(BoolOpt) Enable SSL on the API server
use_ssl=False	(BoolOpt) Use SSL to connect

Testing

Use the following options to alter testing-related features.

Table 4.41. Description of configuration options for testing

Configuration option=Default value	Description
backdoor_port=None	(IntOpt) port for eventlet backdoor to listen
fake_rabbit=False	(BoolOpt) If passed, use a fake RabbitMQ provider

WSGI

Use the following options in the `neutron.conf` file to configure the WSGI layer.

Table 4.42. Description of configuration options for wsgi

Configuration option=Default value	Description
backlog=4096	(IntOpt) Number of backlog requests to configure the socket with
retry_until_window=30	(IntOpt) Number of seconds to keep retrying to listen
tcp_keepidle=600	(IntOpt) Sets the value of TCP_KEEPIDLE in seconds for each server socket. Not supported on OS X.

Identity Service

Procedure 4.1. To configure the Identity Service for use with Networking

1. **Create the `get_id()` function**

   The `get_id()` function stores the ID of created objects, and removes error-prone copying and pasting of object IDs in later steps:

   a. Add the following function to your `.bashrc` file:

   ```
   $ function get_id () {
   echo `"$@" | awk '/ id / { print $4 }'`
   }
   ```

   b. Source the `.bashrc` file:

   ```
   $ source .bashrc
   ```

2. **Create the Networking service entry**

   OpenStack Networking must be available in the OpenStack Compute service catalog. Create the service:

   ```
   $ NEUTRON_SERVICE_ID=$(get_id keystone service-create --name neutron --
   type network --description 'OpenStack Networking Service')
   ```

3. **Create the Networking service endpoint entry**

   The way that you create an OpenStack Networking endpoint entry depends on whether you are using the SQL catalog driver or the template catalog driver:

   - If you use the *SQL driver*, run these command with these parameters: specified region ($REGION), IP address of the OpenStack Networking server ($IP), and service ID ($NEUTRON_SERVICE_ID, obtained in the previous step).

     ```
     $ keystone endpoint-create --region $REGION --service-id
      $NEUTRON_SERVICE_ID --publicurl 'http://$IP:9696/' --adminurl 'http://
     $IP:9696/' --internalurl 'http://$IP:9696/'
     ```

     For example:

     ```
     $ keystone endpoint-create --region myregion --service-id
      $NEUTRON_SERVICE_ID \
     --publicurl "http://10.211.55.17:9696/" --adminurl "http://10.211.55.
     17:9696/" --internalurl "http://10.211.55.17:9696/"
     ```

   - If you are using the *template driver*, add the following content to your OpenStack Compute catalog template file (default_catalog.templates), using these parameters: given region ($REGION) and IP address of the OpenStack Networking server ($IP).

     ```
     catalog.$REGION.network.publicURL = http://$IP:9696
     catalog.$REGION.network.adminURL = http://$IP:9696
     catalog.$REGION.network.internalURL = http://$IP:9696
     catalog.$REGION.network.name = Network Service
     ```

     For example:

     ```
     catalog.$Region.network.publicURL = http://10.211.55.17:9696
     catalog.$Region.network.adminURL = http://10.211.55.17:9696
     catalog.$Region.network.internalURL = http://10.211.55.17:9696
       catalog.$Region.network.name = Network Service
     ```

4. **Create the Networking service user**

   You must provide admin user credentials that OpenStack Compute and some internal components of OpenStack Networking can use to access the OpenStack Networking API. The suggested approach is to create a special `service` tenant, create a `neutron` user within this tenant, and to assign this user an `admin` role.

   a. Create the `admin` role:

      ```
      $ ADMIN_ROLE=$(get_id keystone role-create --name=admin)
      ```

   b. Create the `neutron` user:

      ```
      $ NEUTRON_USER=$(get_id keystone user-create --name=neutron --pass=
      "$NEUTRON_PASSWORD" --email=demo@example.com --tenant-id service)
      ```

   c. Create the `service` tenant:

      ```
      $ SERVICE_TENANT=$(get_id keystone tenant-create --name service --
      description "Services Tenant")
      ```

   d. Establish the relationship among the tenant, user, and role:

```
$ keystone user-role-add --user_id $NEUTRON_USER --role_id $ADMIN_ROLE
 --tenant_id $SERVICE_TENANT
```

For information about how to create service entries and users. see the *OpenStack Installation Guide* for your distribution (docs.openstack.org).

Compute

If you use OpenStack Networking, do not run the OpenStack Compute `nova-network` service (like you do in traditional OpenStack Compute deployments). Instead, OpenStack Compute delegates most network-related decisions to OpenStack Networking. OpenStack Compute proxies tenant-facing API calls to manage security groups and floating IPs to Networking APIs. However, operator-facing tools such as `nova-manage`, are not proxied and should not be used.

Warning

When you configure networking, you must use this guide. Do not rely on OpenStack Compute networking documentation or past experience with OpenStack Compute. If a **nova** command or configuration option related to networking is not mentioned in this guide, the command is probably not supported for use with OpenStack Networking. In particular, you cannot use CLI tools like **nova-manage** and **nova** to manage networks or IP addressing, including both fixed and floating IPs, with OpenStack Networking.

Note

It is strongly recommended that you uninstall `nova-network` and reboot any physical nodes that have been running `nova-network` before using them to run OpenStack Networking. Inadvertently running the `nova-network` process while using OpenStack Networking can cause problems, as can stale iptables rules pushed down by previously running `nova-network`.

To ensure that OpenStack Compute works properly with OpenStack Networking (rather than the legacy `nova-network` mechanism), you must adjust settings in the `nova.conf` configuration file.

Networking API and credential configuration

Each time a VM is provisioned or de-provisioned in OpenStack Compute, `nova-*` services communicate with OpenStack Networking using the standard API. For this to happen, you must configure the following items in the `nova.conf` file (used by each `nova-compute` and `nova-api` instance).

Table 4.43. nova.conf API and credential settings

Item	Configuration
`network_api_class`	Modify from the default to `nova.network.neutronv2.api.API`, to indicate that OpenStack Networking should be used rather than the traditional `nova-network` networking model.
`neutron_url`	Update to the hostname/IP and port of the `neutron-server` instance for this deployment.

Item	Configuration
`neutron_auth_strategy`	Keep the default `keystone` value for all production deployments.
`neutron_admin_tenant_name`	Update to the name of the service tenant created in the above section on OpenStack Identity configuration.
`neutron_admin_username`	Update to the name of the user created in the above section on OpenStack Identity configuration.
`neutron_admin_password`	Update to the password of the user created in the above section on OpenStack Identity configuration.
`neutron_admin_auth_url`	Update to the OpenStack Identity server IP and port. This is the Identity (keystone) admin API server IP and port value, and not the Identity service API IP and port.

Configure security groups

The OpenStack Networking Service provides security group functionality using a mechanism that is more flexible and powerful than the security group capabilities built into OpenStack Compute. Therefore, if you use OpenStack Networking, you should always disable built-in security groups and proxy all security group calls to the OpenStack Networking API . If you do not, security policies will conflict by being simultaneously applied by both services.

To proxy security groups to OpenStack Networking, use the following configuration values in `nova.conf`:

Table 4.44. nova.conf security group settings

Item	Configuration
`firewall_driver`	Update to `nova.virt.firewall.NoopFirewallDriver`, so that `nova-compute` does not perform iptables-based filtering itself.
`security_group_api`	Update to `neutron`, so that all security group requests are proxied to the OpenStack Network Service.

Configure metadata

The OpenStack Compute service allows VMs to query metadata associated with a VM by making a web request to a special 169.254.169.254 address. OpenStack Networking supports proxying those requests to `nova-api`, even when the requests are made from isolated networks, or from multiple networks that use overlapping IP addresses.

To enable proxying the requests, you must update the following fields in `nova.conf`.

Table 4.45. nova.conf metadata settings

Item	Configuration
`service_neutron_metadata_proxy`	Update to `true`, otherwise `nova-api` will not properly respond to requests from the `neutron-metadata-agent`.
`neutron_metadata_proxy_shared_secret`	Update to a string "password" value. You must also configure the same value in the `metadata_agent.ini` file, to authenticate requests made for metadata. The default value of an empty string in both files will allow metadata to function, but will not be secure if any non-trusted entities have access to the metadata APIs exposed by `nova-api`.

Note

As a precaution, even when using `neutron_metadata_proxy_shared_secret`, it is recommended that you

do not expose metadata using the same `nova-api` instances that are used for tenants. Instead, you should run a dedicated set of `nova-api` instances for metadata that are available only on your management network. Whether a given `nova-api` instance exposes metadata APIs is determined by the value of `enabled_apis` in its `nova.conf`.

Example nova.conf (for nova-compute and nova-api)

Example values for the above settings, assuming a cloud controller node running OpenStack Compute and OpenStack Networking with an IP address of 192.168.1.2.

```
network_api_class=nova.network.neutronv2.api.API
neutron_url=http://192.168.1.2:9696
neutron_auth_strategy=keystone
neutron_admin_tenant_name=service
neutron_admin_username=neutron
neutron_admin_password=password
neutron_admin_auth_url=http://192.168.1.2:35357/v2.0

security_group_api=neutron
firewall_driver=nova.virt.firewall.NoopFirewallDriver

service_neutron_metadata_proxy=true
neutron_metadata_proxy_shared_secret=foo
```

Networking scenarios

This chapter describes two networking scenarios and how the Open vSwitch plug-in and the Linux bridging plug-in implement these scenarios.

Open vSwitch

This section describes how the Open vSwitch plug-in implements the OpenStack Networking abstractions.

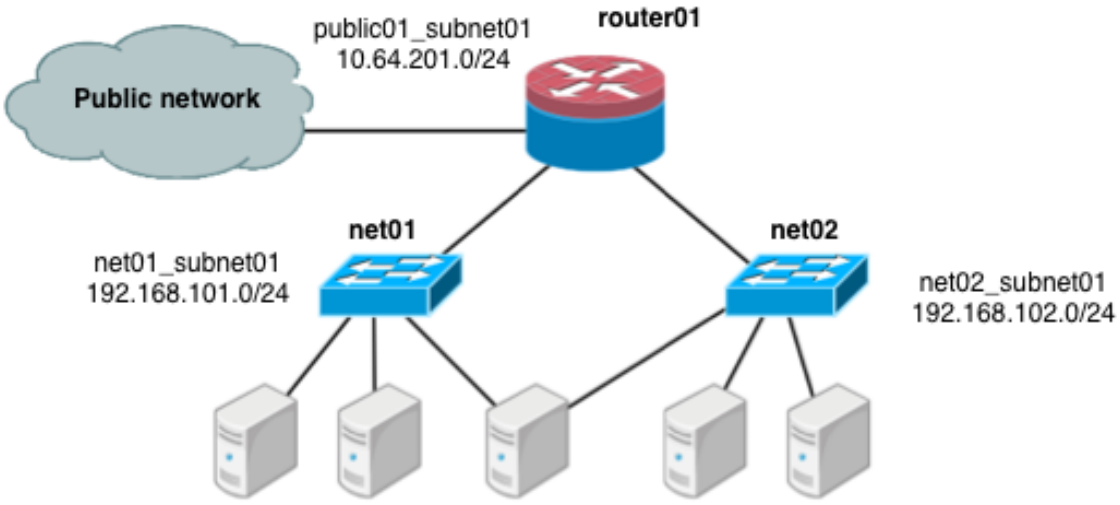
Configuration

This example uses VLAN isolation on the switches to isolate tenant networks. This configuration labels the physical network associated with the public network as `physnet1`, and the physical network associated with the data network as `physnet2`, which leads to the following configuration options in `ovs_neutron_plugin.ini`:

```
[ovs]
tenant_network_type = vlan
network_vlan_ranges = physnet2:100:110
integration_bridge = br-int
bridge_mappings = physnet2:br-eth1
```

Scenario 1: one tenant, two networks, one router

The first scenario has two private networks (`net01`, and `net02`), each with one subnet (`net01_subnet01`: 192.168.101.0/24, `net02_subnet01`, 192.168.102.0/24). Both private networks are attached to a router that connects them to the public network (10.64.201.0/24).



Under the `service` tenant, create the shared router, define the public network, and set it as the default gateway of the router

```
$ tenant=$(keystone tenant-list | awk '/service/ {print $2}')
$ neutron router-create router01
$ neutron net-create --tenant-id $tenant public01 \
          --provider:network_type flat \
          --provider:physical_network physnet1 \
          --router:external=True
$ neutron subnet-create --tenant-id $tenant --name public01_subnet01 \
          --gateway 10.64.201.254 public01 10.64.201.0/24 --disable-dhcp
$ neutron router-gateway-set router01 public01
```

Under the `demo` user tenant, create the private network `net01` and corresponding subnet, and connect it to the `router01` router. Configure it to use VLAN ID 101 on the physical switch.

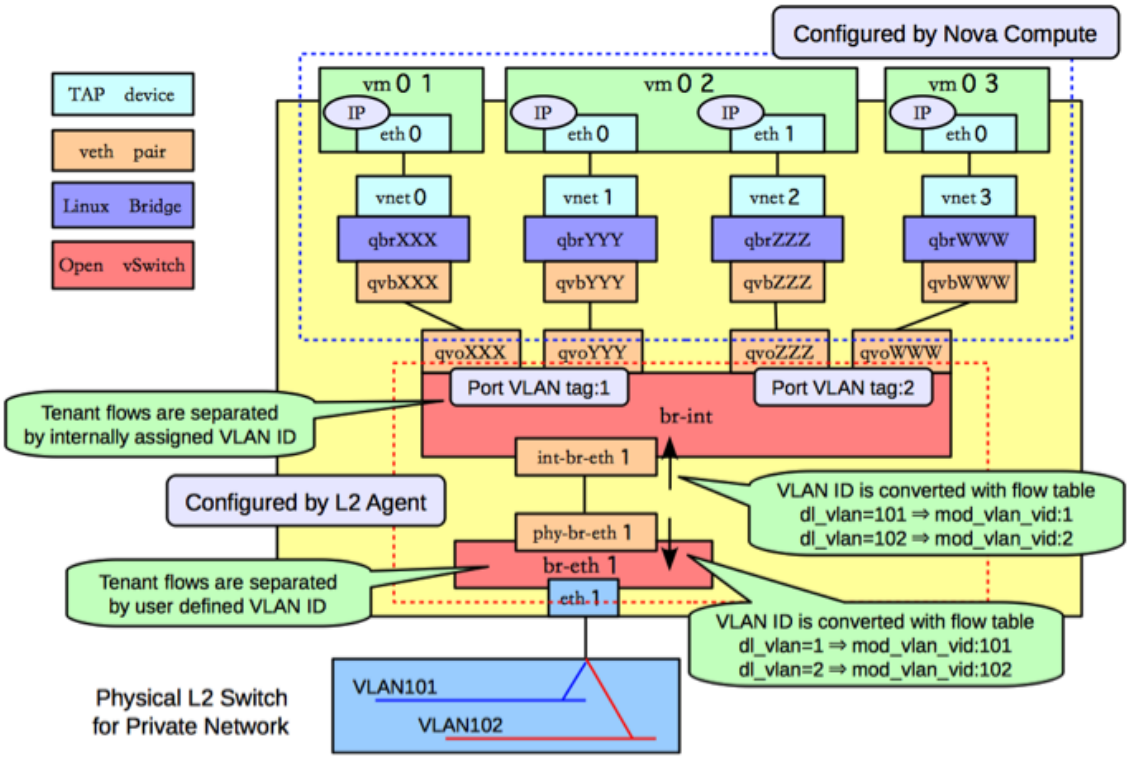
```
$ tenant=$(keystone tenant-list|awk '/demo/ {print $2}'
$ neutron net-create --tenant-id $tenant net01 \
          --provider:network_type vlan \
          --provider:physical_network physnet2 \
          --provider:segmentation_id 101
$ neutron subnet-create --tenant-id $tenant --name net01_subnet01 net01 192.
168.101.0/24
$ neutron router-interface-add router01 net01_subnet01
```

Similarly, for `net02`, using VLAN ID 102 on the physical switch:

```
$ neutron net-create --tenant-id $tenant net02 \
          --provider:network_type vlan \
          --provider:physical_network physnet2 \
          --provider:segmentation_id 102
$ neutron subnet-create --tenant-id $tenant --name net02_subnet01 net02 192.
168.102.0/24
$ neutron router-interface-add router01 net02_subnet01
```

Scenario 1: Compute host config

The following figure shows how to configure various Linux networking devices on the compute host:



Types of network devices

Note

There are four distinct type of virtual networking devices: TAP devices, veth pairs, Linux bridges, and Open vSwitch bridges. For an ethernet frame to travel from `eth0` of virtual machine `vm01` to the physical network, it must pass through nine devices inside of the host: TAP `vnet0`, Linux bridge `qbr`*nnn*, veth pair (`qvb`*nnn*, `qvo`*nnn*), Open vSwitch bridge `br-int`, veth pair (`int-br-eth1`, `phy-br-eth1`), and, finally, the physical network interface card `eth1`.

A *TAP device*, such as `vnet0` is how hypervisors such as KVM and Xen implement a virtual network interface card (typically called a VIF or vNIC). An ethernet frame sent to a TAP device is received by the guest operating system.

A *veth pair* is a pair of virtual network interfaces correctly directly together. An ethernet frame sent to one end of a veth pair is received by the other end of a veth pair. OpenStack networking makes use of veth pairs as virtual patch cables in order to make connections between virtual bridges.

A *Linux bridge* behaves like a hub: you can connect multiple (physical or virtual) network interfaces devices to a Linux bridge. Any ethernet frames that come in from one interface attached to the bridge is transmitted to all of the other devices.

An *Open vSwitch bridge* behaves like a virtual switch: network interface devices connect to Open vSwitch bridge's ports, and the ports can be configured much like a physical switch's ports, including VLAN configurations.

Integration bridge

The `br-int` OpenvSwitch bridge is the integration bridge: all of the guests running on the compute host connect to this bridge. OpenStack Networking implements isolation across these guests by configuring the `br-int` ports.

Physical connectivity bridge

The `br-eth1` bridge provides connectivity to the physical network interface card, `eth1`. It connects to the integration bridge by a veth pair: `(int-br-eth1, phy-br-eth1)`.

VLAN translation

In this example, net01 and net02 have VLAN ids of 1 and 2, respectively. However, the physical network in our example only supports VLAN IDs in the range 101 through 110. The Open vSwitch agent is responsible for configuring flow rules on `br-int` and `br-eth1` to do VLAN translation. When `br-eth1` receives a frame marked with VLAN ID 1 on the port associated with `phy-br-eth1`, it modifies the VLAN ID in the frame to 101. Similarly, when `br-int` receives a frame marked with VLAN ID 101 on the port associated with `int-br-eth1`, it modifies the VLAN ID in the frame to 1.

Security groups: iptables and Linux bridges

Ideally, the TAP device `vnet0` would be connected directly to the integration bridge, `br-int`. Unfortunately, this isn't possible because of how OpenStack security groups are currently implemented. OpenStack uses iptables rules on the TAP devices such as `vnet0` to implement security groups, and Open vSwitch is not compatible with iptables rules that are applied directly on TAP devices that are connected to an Open vSwitch port.

OpenStack Networking uses an extra Linux bridge and a veth pair as a workaround for this issue. Instead of connecting `vnet0` to an Open vSwitch bridge, it is connected to a Linux bridge, `qbr`*XXX*. This bridge is connected to the integration bridge, `br-int`, through the `(qvb`*XXX*`, qvo`*XXX*`)` veth pair.

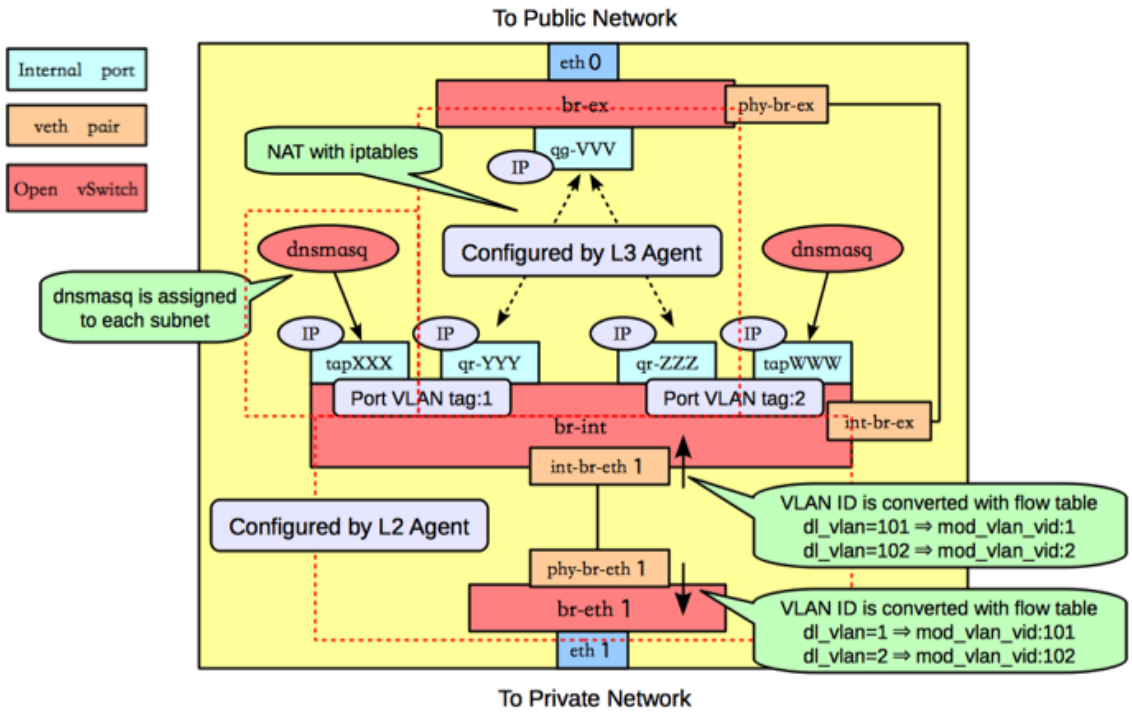
Scenario 1: Network host config

The network host runs the neutron-openvswitch-plugin-agent, the neutron-dhcp-agent, neutron-l3-agent, and neutron-metadata-agent services.

On the network host, assume that eth0 is connected to the external network, and eth1 is connected to the data network, which leads to the following configuration in the `ovs_neutron_plugin.ini` file:

```
[ovs]
tenant_network_type = vlan
network_vlan_ranges = physnet2:101:110
integration_bridge = br-int
bridge_mappings = physnet1:br-ex,physnet2:br-eth1
```

The following figure shows the network devices on the network host:



As on the compute host, there is an Open vSwitch integration bridge (`br-int`) and an Open vSwitch bridge connected to the data network (`br-eth1`), and the two are connected by a veth pair, and the neutron-openvswitch-plugin-agent configures the ports on both switches to do VLAN translation.

An additional Open vSwitch bridge, `br-ex`, connects to the physical interface that is connected to the external network. In this example, that physical interface is `eth0`.

Note

While the integration bridge and the external bridge are connected by a veth pair (`int-br-ex, phy-br-ex`), this example uses layer 3 connectivity to route packets from the internal networks to the public network: no packets traverse that veth pair in this example.

Open vSwitch internal ports

The network host uses Open vSwitch *internal ports*. Internal ports enable you to assign one or more IP addresses to an Open vSwitch bridge. In previous example, the `br-int` bridge has four internal ports: `tap`*XXX*, `qr-`*YYY*, `qr-`*ZZZ*, `tap`*WWW*. Each internal port has a separate IP address associated with it. An internal port, `qg-VVV`, is on the `br-ex` bridge.

DHCP agent

By default, The OpenStack Networking DHCP agent uses a program called dnsmasq to provide DHCP services to guests. OpenStack Networking must create an internal port for each network that requires DHCP services and attach a dnsmasq process to that port. In the previous example, the interface `tap`*XXX* is on subnet `net01_subnet01`, and the interface `tap`*WWW* is on `net02_subnet01`.

L3 agent (routing)

The OpenStack Networking L3 agent implements routing through the use of Open vSwitch internal ports and relies on the network host to route the packets across the interfaces. In this example: interfaceqr-YYY, which is on subnet net01_subnet01, has an IP address of 192.168.101.1/24, interface qr-*ZZZ*, which is on subnet net02_subnet01, has an IP address of 192.168.102.1/24, and interface qg-*VVV*, which has an IP address of 10.64.201.254/24. Because of each of these interfaces is visible to the network host operating system, it will route the packets appropriately across the interfaces, as long as an administrator has enabled IP forwarding.

The L3 agent uses iptables to implement floating IPs to do the network address translation (NAT).

Overlapping subnets and network namespaces

One problem with using the host to implement routing is that there is a chance that one of the OpenStack Networking subnets might overlap with one of the physical networks that the host uses. For example, if the management network is implemented on eth2 (not shown in the previous example), by coincidence happens to also be on the 192.168.101.0/24 subnet, then this will cause routing problems because it is impossible ot determine whether a packet on this subnet should be sent to qr-YYY or eth2. In general, if end-users are permitted to create their own logical networks and subnets, then the system must be designed to avoid the possibility of such collisions.

OpenStack Networking uses Linux *network namespaces* to prevent collisions between the physical networks on the network host, and the logical networks used by the virtual machines. It also prevents collisions across different logical networks that are not routed to each other, as you will see in the next scenario.

A network namespace can be thought of as an isolated environment that has its own networking stack. A network namespace has its own network interfaces, routes, and iptables rules. You can think of like a chroot jail, except for networking instead of a file system. As an aside, LXC (Linux containers) use network namespaces to implement networking virtualization.

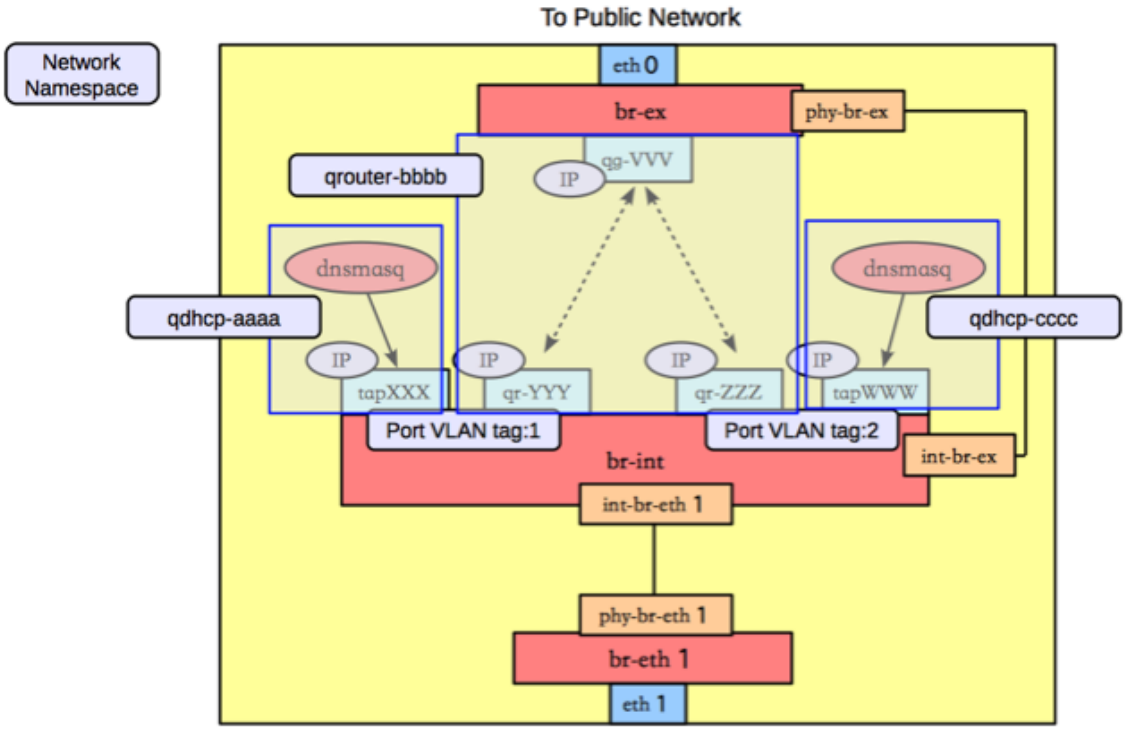
OpenStack Networking creates network namespaces on the network host in order to avoid subnet collisions.

Tn this example, there are three network namespaces, as depicted in the following figure.

- qdhcp-*aaa*: contains the tap*XXX* interface and the dnsmasq process that listens on that interface, to provide DHCP services for net01_subnet01. This allows overlapping IPs between net01_subnet01 and any other subnets on the network host.

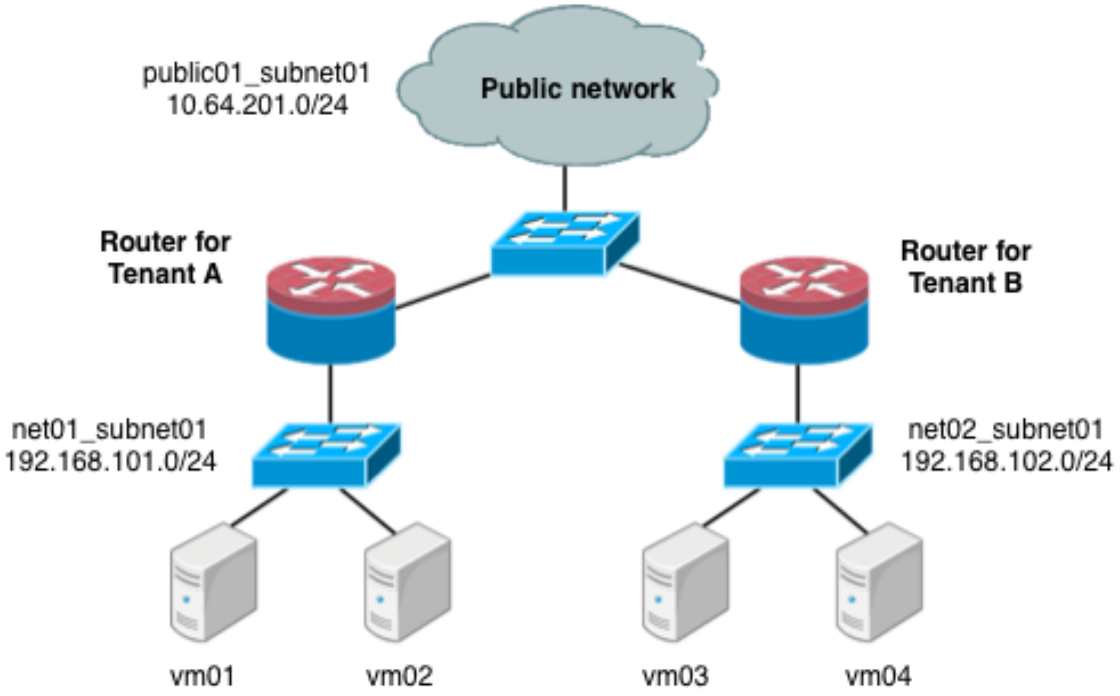
- qrouter-*bbbb*: contains the qr-*YYY*, qr-*ZZZ*, and qg-*VVV* interfaces, and the corresponding routes. This namespace implements router01 in our example.

- qdhcp-*ccc*: contains the tap*WWW* interface and the dnsmasq process that listens on that interface, to provide DHCP services for net02_subnet01. This allows overlapping IPs between net02_subnet01 and any other subnets on the network host.



Scenario 2: two tenants, two networks, two routers

In this scenario, tenant A and tenant B each have a network with one subnet and one router that connects the tenants to the public Internet.



Under the `service` tenant, define the public network:

```
$ tenant=$(keystone tenant-list | awk '/service/ {print $2}')
$ neutron net-create --tenant-id $tenant public01 \
          --provider:network_type flat \
          --provider:physical_network physnet1 \
          --router:external=True
$ neutron subnet-create --tenant-id $tenant --name public01_subnet01 \
          --gateway 10.64.201.254 public01 10.64.201.0/24 --disable-dhcp
```

Under the `tenantA` user tenant, create the tenant router and set its gateway for the public network.

```
$ tenant=$(keystone tenant-list|awk '/tenantA/ {print $2}')
$ neutron router-create --tenant-id $tenant router01
$ neutron router-gateway-set router01 public01
```

Then, define private network `net01` using VLAN ID 102 on the physical switch, along with its subnet, and connect it to the router.

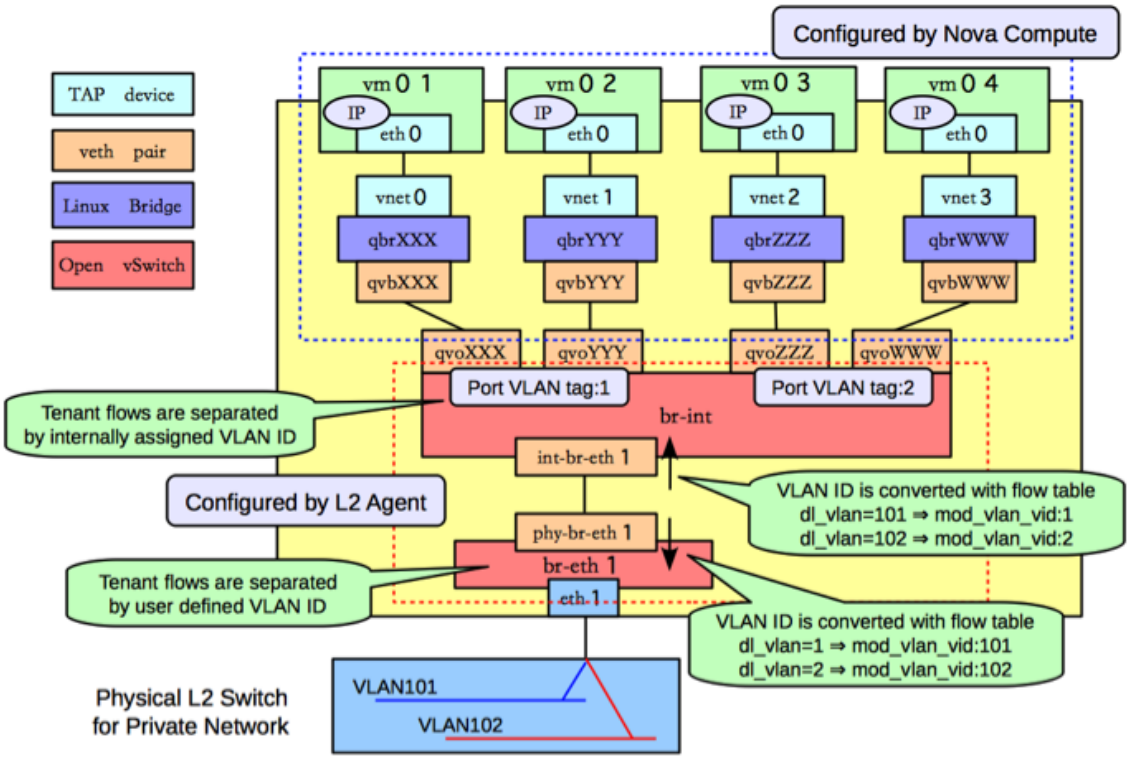
```
$ neutron net-create --tenant-id $tenant net01 \
          --provider:network_type vlan \
          --provider:physical_network physnet2 \
          --provider:segmentation_id 101
$ neutron subnet-create --tenant-id $tenant --name net01_subnet01 net01 192.
168.101.0/24
$ neutron router-interface-add router01 net01_subnet01
```

Similarly, for `tenantB`, create a router and another network, using VLAN ID 102 on the physical switch:

```
$ tenant=$(keystone tenant-list|awk '/tenantB/ {print $2}')
$ neutron router-create --tenant-id $tenant router02
$ neutron router-gateway-set router02 public01
$ neutron net-create --tenant-id $tenant net02 \
          --provider:network_type vlan \
          --provider:physical_network physnet2 \
          --provider:segmentation_id 102
$ neutron subnet-create --tenant-id $tenant --name net02_subnet01 net01 192.
168.101.0/24
$ neutron router-interface-add router02 net02_subnet01
```

Scenario 2: Compute host config

The following figure shows how to configure Linux networking devices on the Compute host:

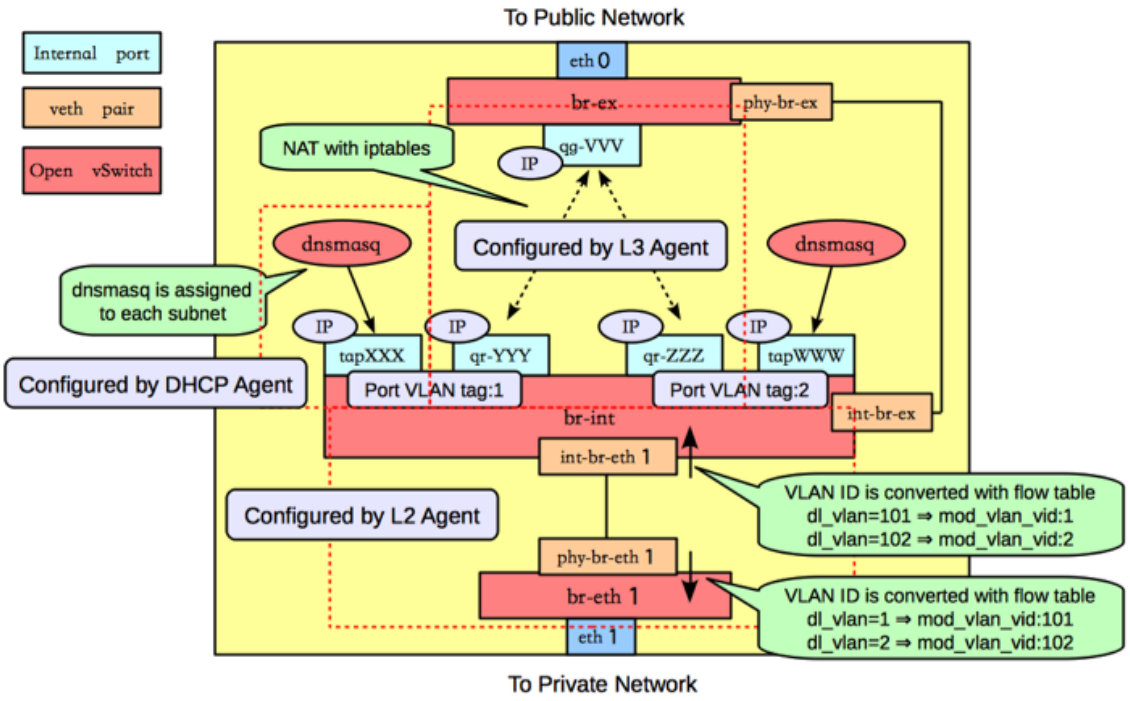


Note

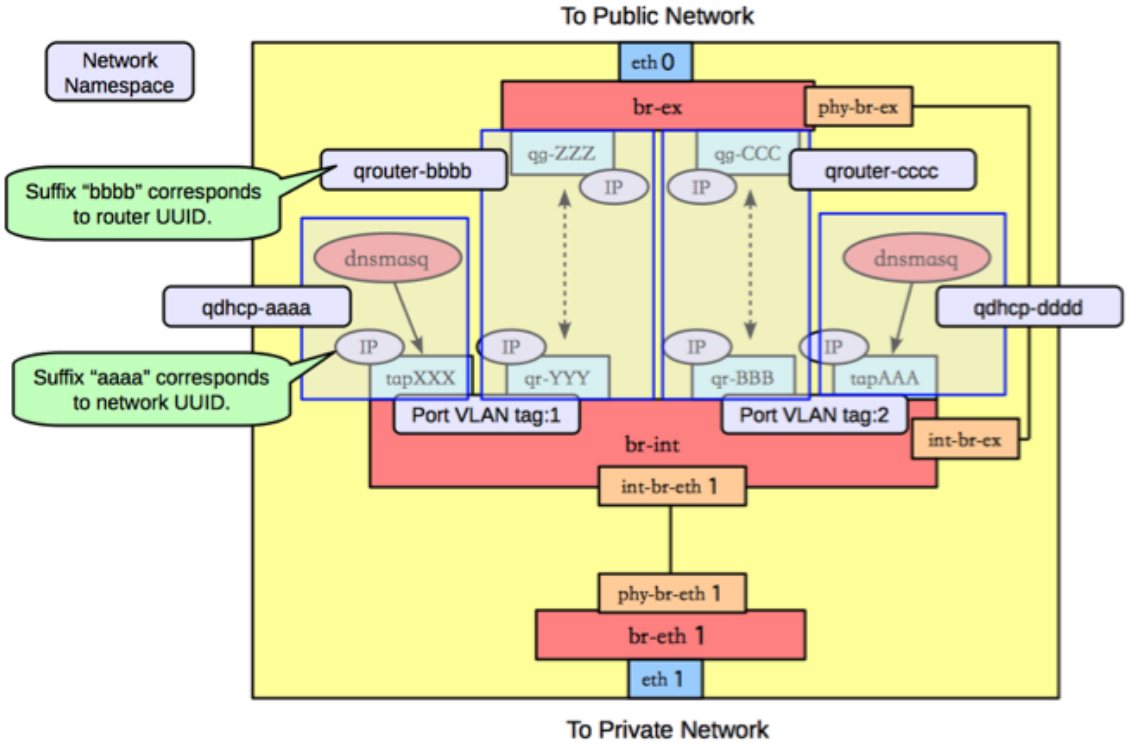
The Compute host configuration resembles the configuration in scenario 1. However, in scenario 1, a guest connects to two subnets while in this scenario, the subnets belong to different tenants.

Scenario 2: Network host config

The following figure shows the network devices on the network host for the second scenario.



In this configuration, the network namespaces are organized to isolate the two subnets from each other as shown in the following figure.



In this scenario, there are four network namespaces (`qhdcp-aaa`, `qrouter-bbbb`, `qrouter-cccc`, and `qhdcp-dddd`), instead of three. Since there is no connectivity between the two networks, and so each router is implemented by a separate namespace.

Linux Bridge

This section describes how the Linux Bridge plug-in implements the OpenStack Networking abstractions. For information about DHCP and L3 agents, see the section called "Scenario 1: one tenant, two networks, one router" [188].

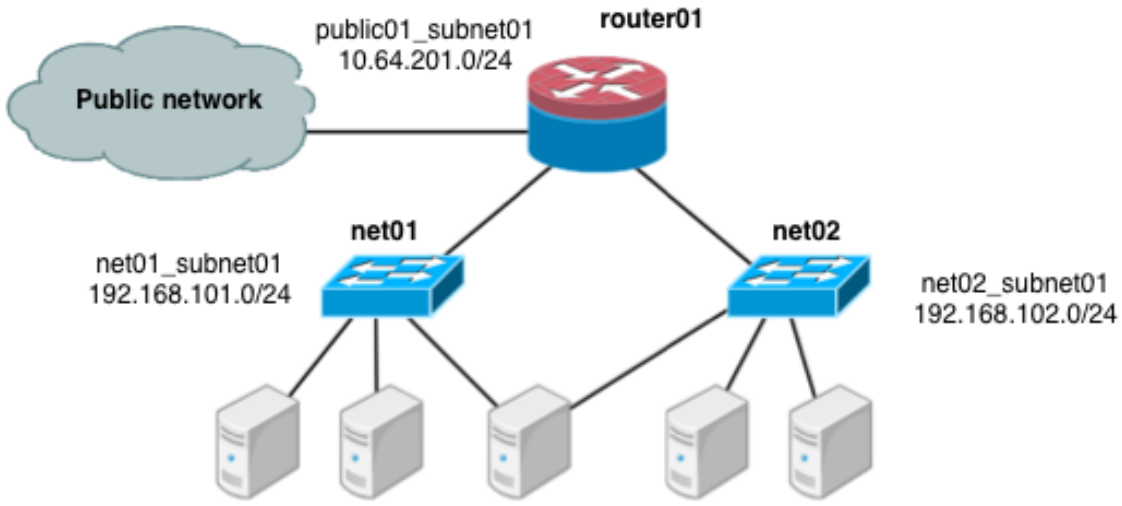
Configuration

This example uses VLAN isolation on the switches to isolate tenant networks. This configuration labels the physical network associated with the public network as `physnet1`, and the physical network associated with the data network as `physnet2`, which leads to the following configuration options in `linuxbridge_conf.ini`:

```
[vlans]
tenant_network_type = vlan
network_vlan_ranges = physnet2:100:110

[linux_bridge]
physical_interface_mappings = physnet2:eth1
```

Scenario 1: one tenant, two networks, one router

The first scenario has two private networks (`net01`, and `net02`), each with one subnet (`net01_subnet01`: 192.168.101.0/24, `net02_subnet01`, 192.168.102.0/24). Both private networks are attached to a router that contains them to the public network (10.64.201.0/24).



Under the `service` tenant, create the shared router, define the public network, and set it as the default gateway of the router

```
$ tenant=$(keystone tenant-list | awk '/service/ {print $2}')
$ neutron router-create router01
$ neutron net-create --tenant-id $tenant public01 \
```

```
          --provider:network_type flat \
          --provider:physical_network physnet1 \
          --router:external=True
$ neutron subnet-create --tenant-id $tenant --name public01_subnet01 \
          --gateway 10.64.201.254 public01 10.64.201.0/24 --disable-dhcp
$ neutron router-gateway-set router01 public01
```

Under the `demo` user tenant, create the private network `net01` and corresponding subnet, and connect it to the `router01` router. Configure it to use VLAN ID 101 on the physical switch.

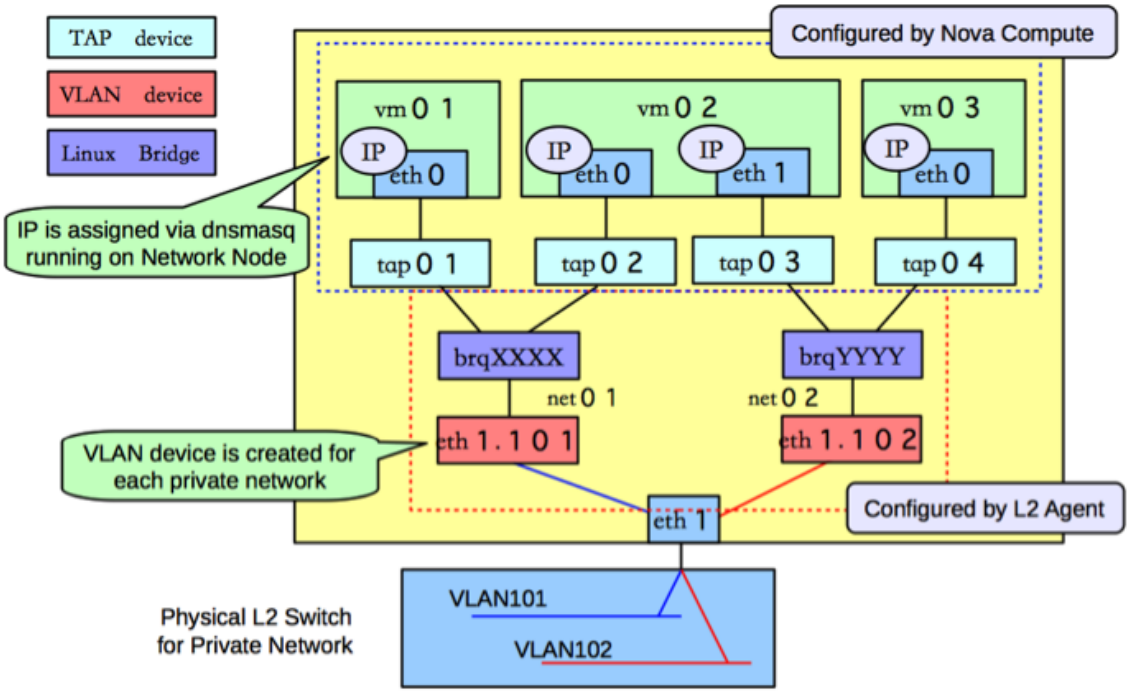
```
$ tenant=$(keystone tenant-list|awk '/demo/ {print $2}'
$ neutron net-create --tenant-id $tenant net01 \
          --provider:network_type vlan \
          --provider:physical_network physnet2 \
          --provider:segmentation_id 101
$ neutron subnet-create --tenant-id $tenant --name net01_subnet01 net01 192.
168.101.0/24
$ neutron router-interface-add router01 net01_subnet01
```

Similarly, for `net02`, using VLAN ID 102 on the physical switch:

```
$ neutron net-create --tenant-id $tenant net02 \
          --provider:network_type vlan \
          --provider:physical_network physnet2 \
          --provider:segmentation_id 102
$ neutron subnet-create --tenant-id $tenant --name net02_subnet01 net02 192.
168.102.0/24
$ neutron router-interface-add router01 net02_subnet01
```

Scenario 1: Compute host config

The following figure shows how to configure the various Linux networking devices on the compute host.



Types of network devices

Note

There are three distinct type of virtual networking devices: TAP devices, VLAN devices, and Linux bridges. For an ethernet frame to travel from `eth0` of virtual machine `vm01`, to the physical network, it must pass through four devices inside of the host: TAP `vnet0`, Linux bridge `brq`*XXX*, VLAN `eth1.101`), and, finally, the physical network interface card `eth1`.

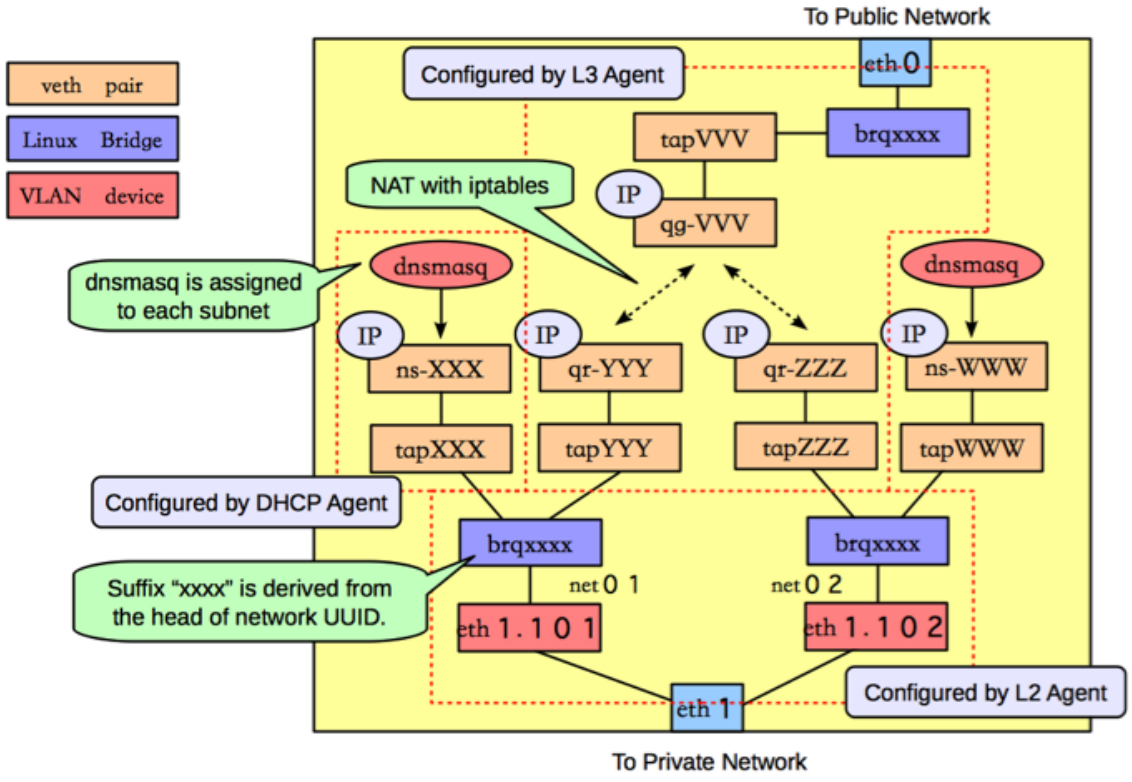
A *TAP device*, such as `vnet0` is how hypervisors such as KVM and Xen implement a virtual network interface card (typically called a VIF or vNIC). An ethernet frame sent to a TAP device is received by the guest operating system.

A *VLAN device* is associated with a VLAN tag attaches to an existing interface device and adds or removes VLAN tags. In the preceding example, VLAN device `eth1.101` is associated with VLAN ID 101 and is attached to interface `eth1`. Packets received from the outside by `eth1` with VLAN tag 101 will be passed to device `eth1.101`, which will then strip the tag. In the other direction, any ethernet frame sent directly to eth1.101 will have VLAN tag 101 added and will be forward to `eth1` for sending out to the network.

A *Linux bridge* behaves like a hub: you can connect multiple (physical or virtual) network interfaces devices to a Linux bridge. Any ethernet frames that come in from one interface attached to the bridge is transmitted to all of the other devices.

Scenario 1: Network host config

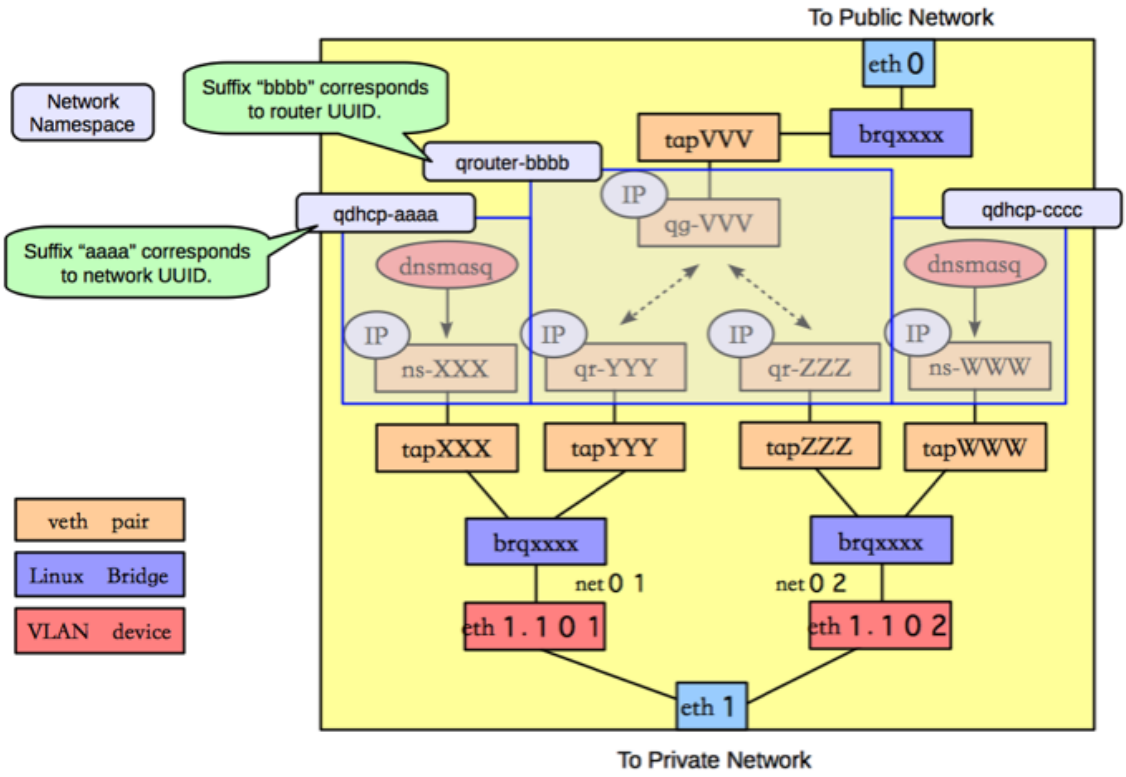
The following figure shows the network devices on the network host.



The following figure shows how the Linux Bridge plug-in uses network namespaces to provide isolation.

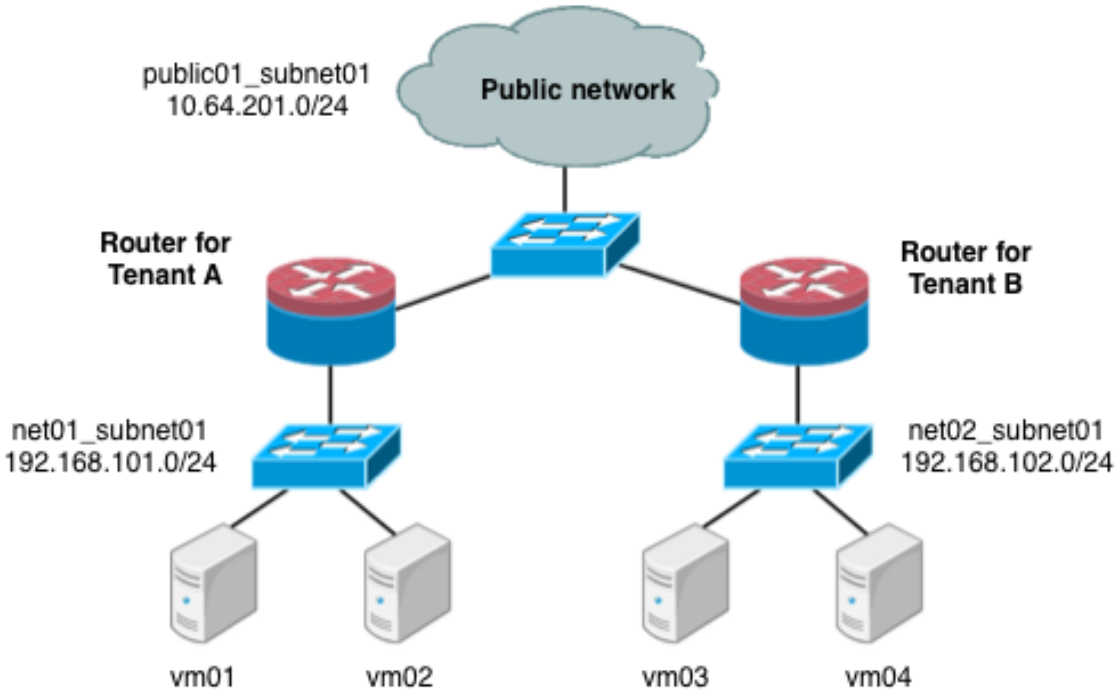
Note

veth pairs form connections between the Linux bridges and the network namespaces.



Scenario 2: two tenants, two networks, two routers

The second scenario has two tenants (A, B). Each tenant has a network with one subnet, and each one has a router that connects them to the public Internet.



Under the `service` tenant, define the public network:

```
$ tenant=$(keystone tenant-list | awk '/service/ {print $2}')
$ neutron net-create --tenant-id $tenant public01 \
         --provider:network_type flat \
         --provider:physical_network physnet1 \
         --router:external=True
$ neutron subnet-create --tenant-id $tenant --name public01_subnet01 \
         --gateway 10.64.201.254 public01 10.64.201.0/24 --disable-dhcp
```

Under the `tenantA` user tenant, create the tenant router and set its gateway for the public network.

```
$ tenant=$(keystone tenant-list|awk '/tenantA/ {print $2}')
$ neutron router-create --tenant-id $tenant router01
$ neutron router-gateway-set router01 public01
```

Then, define private network `net01` using VLAN ID 102 on the physical switch, along with its subnet, and connect it to the router.

```
$ neutron net-create --tenant-id $tenant net01 \
         --provider:network_type vlan \
         --provider:physical_network physnet2 \
         --provider:segmentation_id 101
$ neutron subnet-create --tenant-id $tenant --name net01_subnet01 net01 192.
168.101.0/24
$ neutron router-interface-add router01 net01_subnet01
```

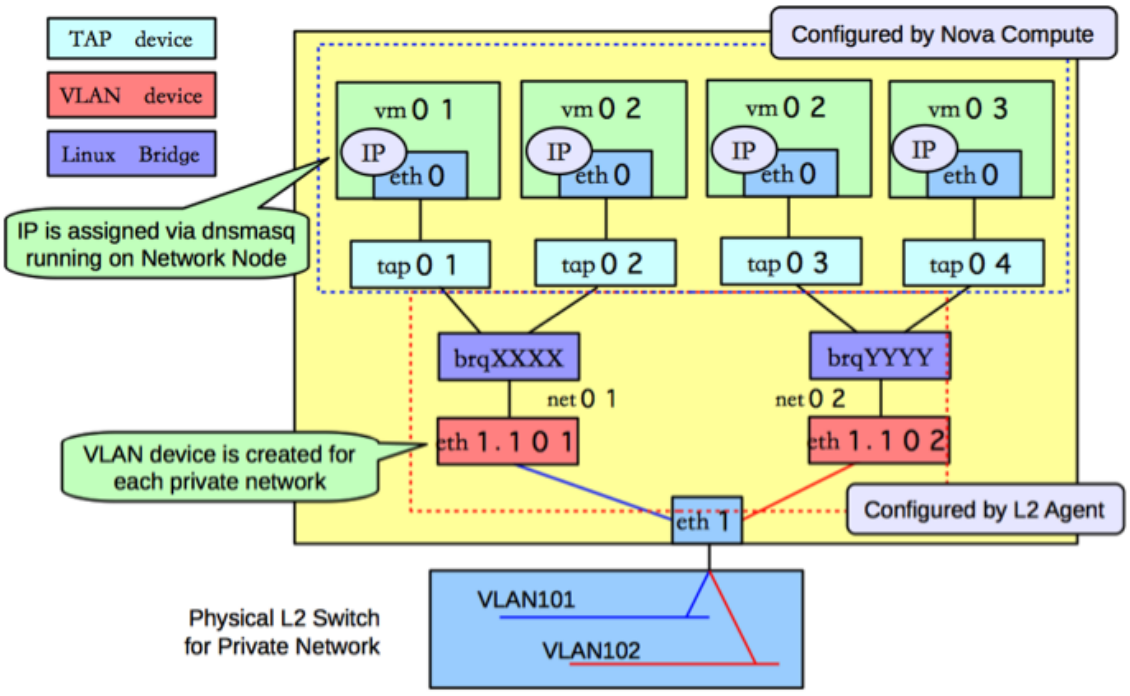
Similarly, for `tenantB`, create a router and another network, using VLAN ID 102 on the physical switch:

```
$ tenant=$(keystone tenant-list|awk '/tenantB/ {print $2}')
```

```
$ neutron router-create --tenant-id $tenant router02
$ neutron router-gateway-set router02 public01
$ neutron net-create --tenant-id $tenant net02 \
          --provider:network_type vlan \
          --provider:physical_network physnet2 \
          --provider:segmentation_id 102
$ neutron subnet-create --tenant-id $tenant --name net02_subnet01 net01 192.
168.101.0/24
$ neutron router-interface-add router02 net02_subnet01
```

Scenario 2: Compute host config

The following figure shows how the various Linux networking devices would be configured on the compute host under this scenario.

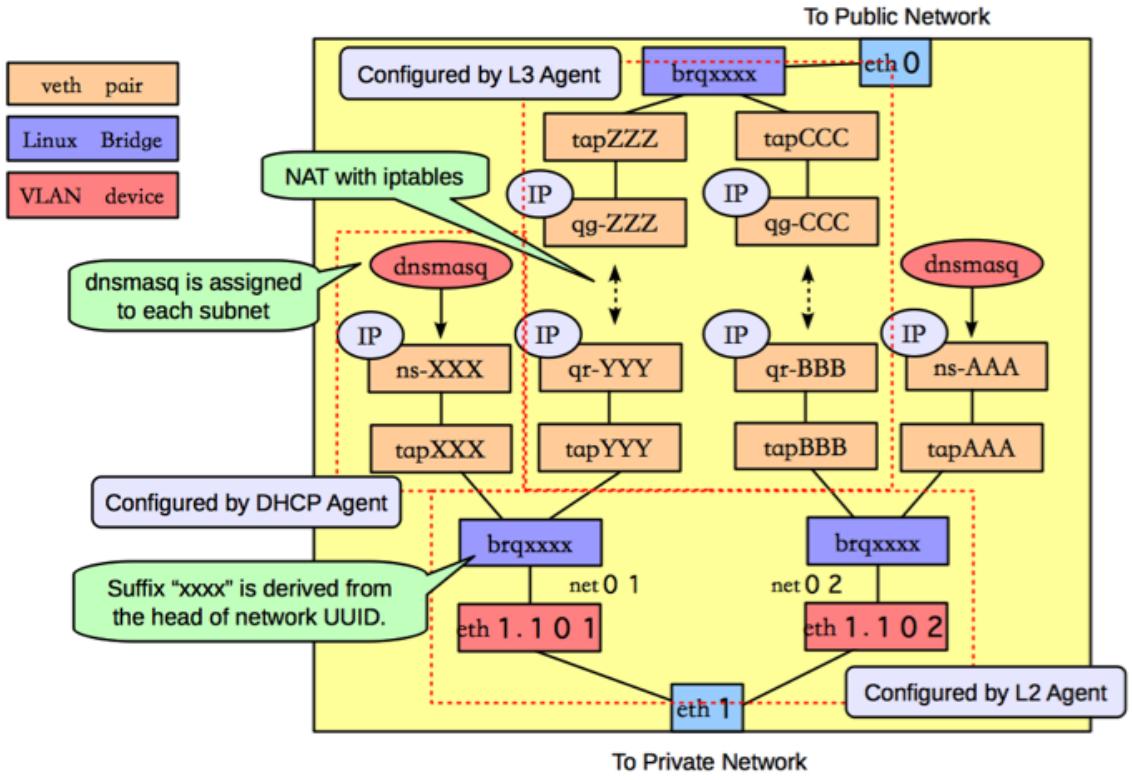


Note

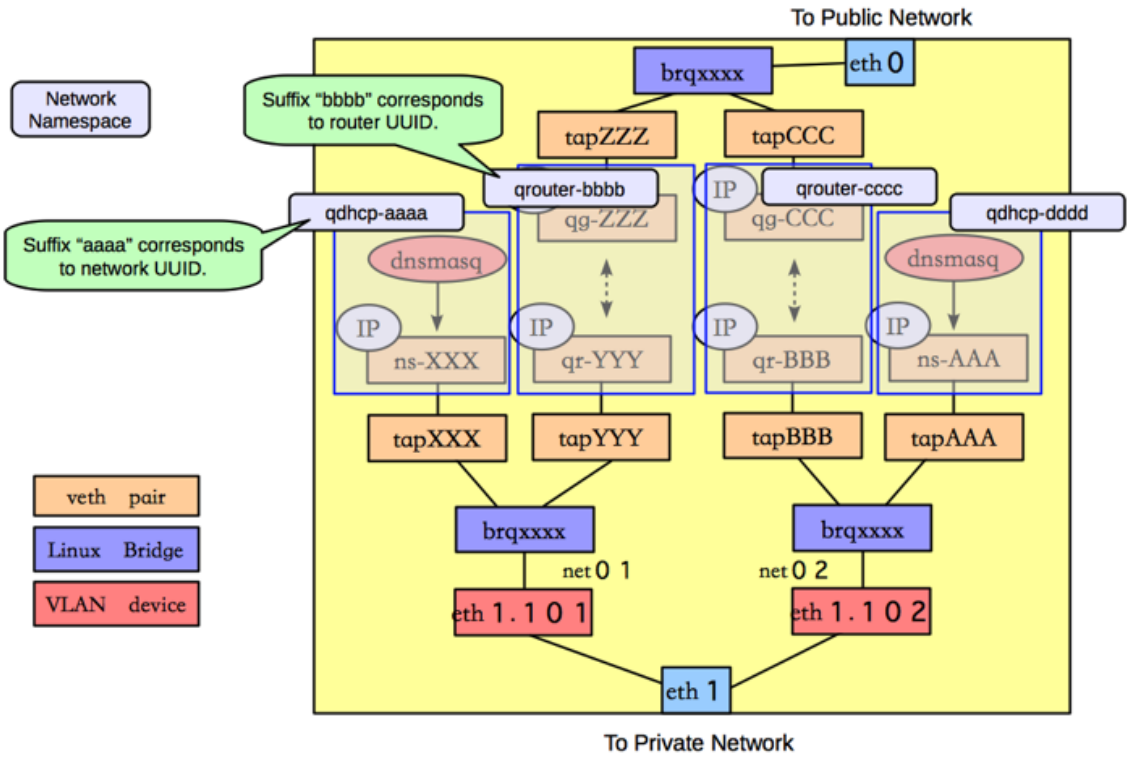
The configuration on the compute host is very similar to the configuration in scenario 1. The only real difference is that scenario 1 had a guest that was connected to two subnets, and in this scenario, the subnets belong to different tenants.

Scenario 2: Network host config

The following figure shows the network devices on the network host for the second scenario.



The main difference between the configuration in this scenario and the previous one is the organization of the network namespaces, in order to provide isolation across the two subnets, as shown in the following figure.



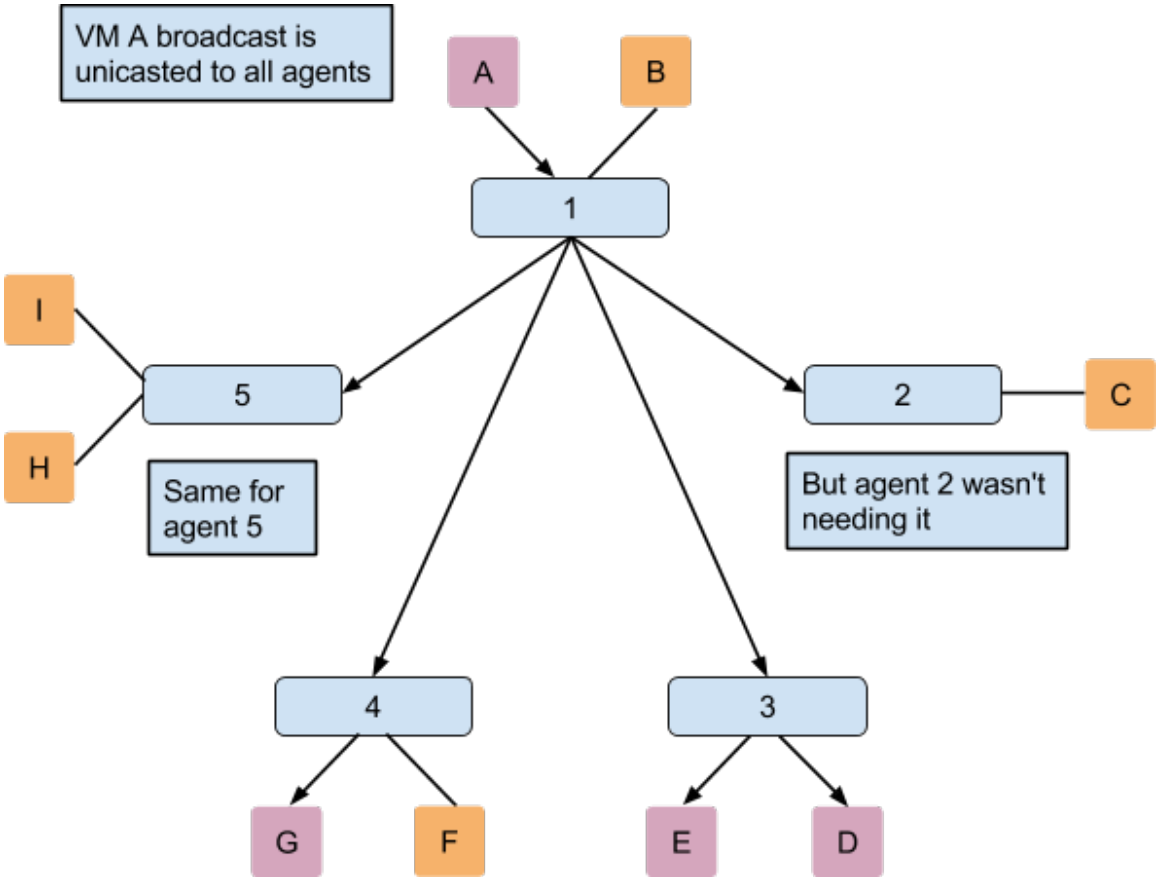
In this scenario, there are four network namespaces (`qhdcp-`*`aaa`*, `qrouter-`*`bbbb`*, `qrouter-`*`cccc`*, and `qhdcp-`*`dddd`*), instead of three. Since there is no connectivity between the two networks, and so each router is implemented by a separate namespace.

ML2

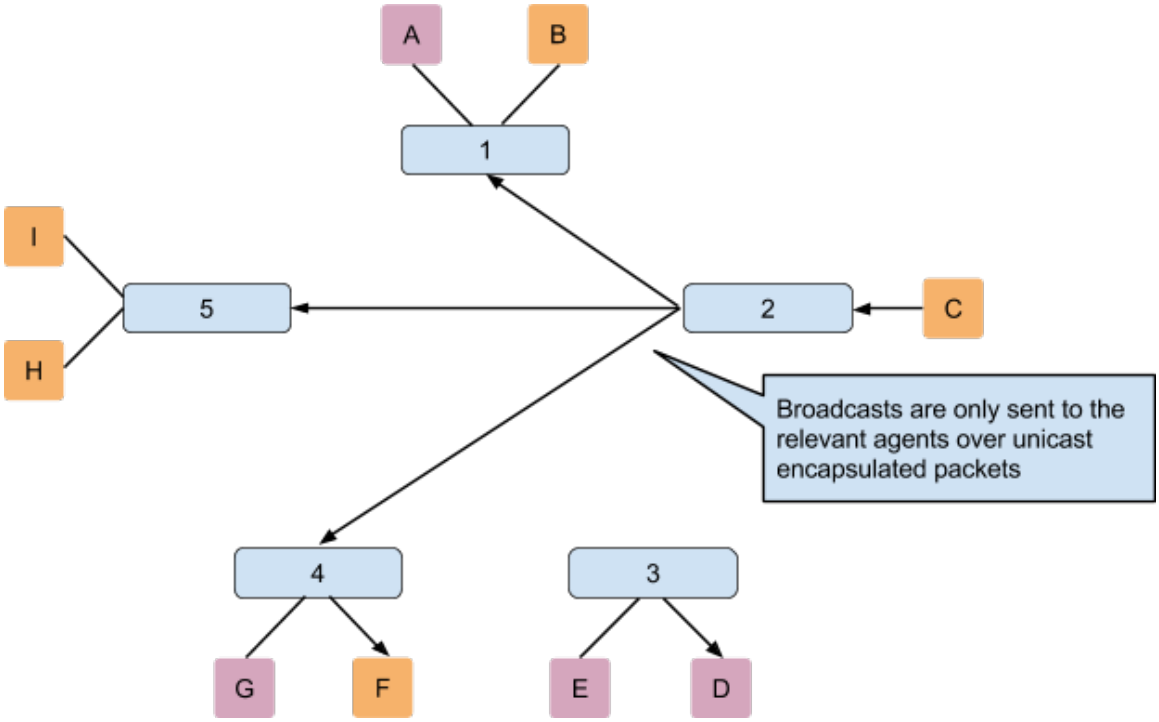
The Modular Layer 2 plugin allows OpenStack Networking to simultaneously utilize the variety of layer 2 networking technologies found in complex real-world data centers. It currently includes drivers for the local, flat, vlan, gre and vxlan network types and works with the existing *Open vSwitch*, *Linux Bridge*, and *HyperV* L2 agents. The *ML2* plug-in can be extended through mechanism drivers, multiple mechanisms can be used simultaneously. This section describes different *ML2* plug-in / agents configurations with different type drivers and mechanism drivers.

ML2 with L2 population mechanism driver

Current *Open vSwitch* and *Linux Bridge* tunneling implementations broadcast to every agent, even if they don't host the corresponding network as illustrated below.



As broadcast emulation on overlay is costly, it may be better to avoid its use for mac learning and ARP resolution. This supposes the use of proxy ARP on the agent to answer VM requests, and to populate forwarding table. Currently only the *Linux Bridge* Agent implements an ARP proxy. The prepopulation limits L2 broadcasts in overlay, however it may anyway be necessary to provide broadcast emulation. This is achieved by sending broadcasts packets over unicasts only to the relevant agents as illustrated below.



The partial-mesh is available with the *Open vSwitch* and the *Linux Bridge* agent. The following scenarios will use the L2 population mechanism driver with an *Open vSwitch* agent and a *Linux Bridge* agent. To enable the l2 population driver we have to add it in the list of mechanism drivers. We also need to have at least one tunneling type driver enabled, either GRE, VXLAN or both. Below configuration options that we have to set in `ml2_conf.ini`:

```
[ml2]
type_drivers = local,flat,vlan,gre,vxlan
mechanism_drivers = openvswitch,linuxbridge,l2population
```

Scenario 1: L2 population with Open vSwitch agent

We have to enable the l2 population extension on the *Open vSwitch* agent side and we also have to set the `local_ip` parameter and the `tunnel_types` `ml2_conf.ini`.

```
[ovs]
local_ip = 192.168.1.10

[agent]
tunnel_types = gre,vxlan
l2_population = True
```

Scenario 2: L2 population with *Linux Bridge* agent

We have to enable the l2 population extension on the agent side and we also have to set the local_ip parameter and enable VXLAN in `ml2_conf.ini`.

```
[vxlan]
enable_vxlan = True
local_ip = 192.168.1.10
l2_population = True
```

Enable security group API

Since the ML2 plugin can concurrently support different L2 agents (or other mechanisms) with different configuration files, the actual `firewall_driver` value in the `ml2_conf.ini` file does not matter in the server, but `firewall_driver` must be set to a non-default value in the ml2 configuration to enable the securitygroup extension. To enable securitygroup API, edit the `ml2_conf.ini` file:

```
[securitygroup]
  firewall_driver = dummy
```

Each L2 agent configuration file (such as `ovs_neutron_plugin.ini` or `linuxbridge_conf.ini`) should contain the appropriate `firewall_driver` value for that agent. To disable securitygroup API, edit the `ml2_conf.ini` file:

```
[securitygroup]
firewall_driver = neutron.agent.firewall.NoopFirewallDriver
```

Also, each L2 agent configuration file (such as `ovs_neutron_plugin.ini` or `linuxbridge_conf.ini`) should contain this value in `firewall_driver` parameter for that agent.

Advanced configuration options

This section describes advanced configuration options for various system components. For example, configuration options where the default works but that the user wants to customize options. After installing from packages, $NEUTRON_CONF_DIR is `/etc/neutron`.

OpenStack Networking server with plug-in

This is the web server that runs the OpenStack Networking API Web Server. It is responsible for loading a plug-in and passing the API calls to the plug-in for processing. The neutron-server should receive one of more configuration files as it its input, for example:

```
neutron-server --config-file <neutron config> --config-file <plugin config>
```

The neutron config contains the common neutron configuration parameters. The plug-in config contains the plug-in specific flags. The plug-in that is run on the service is loaded through the *`core_plugin`* configuration parameter. In some cases a plug-in might have an agent that performs the actual networking.

Most plug-ins require a SQL database. After you install and start the database server, set a password for the root account and delete the anonymous accounts:

```
$> mysql -u root
mysql> update mysql.user set password = password('iamroot') where user =
 'root';
mysql> delete from mysql.user where user = '';
```

Create a database and user account specifically for plug-in:

```
mysql> create database <database-name>;
mysql> create user '<user-name>'@'localhost' identified by '<user-name>';
mysql> create user '<user-name>'@'%' identified by '<user-name>';
```

```
mysql> grant all on <database-name>.* to '<user-name>'@'%';
```

Once the above is done you can update the settings in the relevant plug-in configuration files. The plug-in specific configuration files can be found at $NEUTRON_CONF_DIR/plugins.

Some plug-ins have a L2 agent that performs the actual networking. That is, the agent will attach the virtual machine NIC to the OpenStack Networking network. Each node should have an L2 agent running on it. Note that the agent receives the following input parameters:

```
neutron-plugin-agent --config-file <neutron config> --config-file <plugin
 config>
```

Two things need to be done prior to working with the plug-in:

1. Ensure that the core plug-in is updated.

2. Ensure that the database connection is correctly set.

The following table contains examples for these settings. Some Linux packages might provide installation utilities that configure these.

Table 4.46. Settings

Parameter	Value
Open vSwitch	
core_plugin ($NEUTRON_CONF_DIR/neutron.conf)	neutron.plugins.openvswitch.ovs_neutron_plugin.OVSNeutronPluginV2
connection (in the plugin configuration file, section `[database]`)	mysql://<username>:<password>@localhost/ovs_neutron?charset=utf8
Plug-in Configuration File	$NEUTRON_CONF_DIR/plugins/openvswitch/ovs_neutron_plugin.ini
Agent	neutron-openvswitch-agent
Linux Bridge	
core_plugin ($NEUTRON_CONF_DIR/neutron.conf)	neutron.plugins.linuxbridge.lb_neutron_plugin.LinuxBridgePluginV2
connection (in the plug-in configuration file, section `[database]`)	mysql://<username>:<password>@localhost/neutron_linux_bridge?charset=utf8
Plug-in Configuration File	$NEUTRON_CONF_DIR/plugins/linuxbridge/linuxbridge_conf.ini
Agent	neutron-linuxbridge-agent

All plug-in configuration files options can be found in the Appendix - Configuration File Options.

DHCP agent

There is an option to run a DHCP server that will allocate IP addresses to virtual machines running on the network. When a subnet is created, by default, the subnet has DHCP enabled.

The node that runs the DHCP agent should run:

```
neutron-dhcp-agent --config-file <neutron config>
--config-file <dhcp config>
```

Currently the DHCP agent uses dnsmasq to perform that static address assignment.

A driver needs to be configured that matches the plug-in running on the service.

Table 4.47. Basic settings

Parameter	Value
Open vSwitch	
interface_driver ($NEUTRON_CONF_DIR/dhcp_agent.ini)	neutron.agent.linux.interface.OVSInterfaceDriver
Linux Bridge	
interface_driver ($NEUTRON_CONF_DIR/dhcp_agent.ini)	neutron.agent.linux.interface.BridgeInterfaceDriver

Namespace

By default the DHCP agent makes use of Linux network namespaces in order to support overlapping IP addresses. Requirements for network namespaces support are described in the Limitations section.

If the Linux installation does not support network namespace, you must disable using network namespace in the DHCP agent config file (The default value of use_namespaces is True).

```
use_namespaces = False
```

L3 Agent

There is an option to run a L3 agent that will give enable layer 3 forwarding and floating IP support. The node that runs the L3 agent should run:

```
neutron-l3-agent --config-file <neutron config>
--config-file <l3 config>
```

A driver needs to be configured that matches the plug-in running on the service. The driver is used to create the routing interface.

Table 4.48. Basic settings

Parameter	Value
Open vSwitch	
interface_driver ($NEUTRON_CONF_DIR/l3_agent.ini)	neutron.agent.linux.interface.OVSInterfaceDriver
external_network_bridge ($NEUTRON_CONF_DIR/ l3_agent.ini)	br-ex
Linux Bridge	
interface_driver ($NEUTRON_CONF_DIR/l3_agent.ini)	neutron.agent.linux.interface.BridgeInterfaceDriver
external_network_bridge ($NEUTRON_CONF_DIR/ l3_agent.ini)	This field must be empty (or the bridge name for the external network).

The L3 agent communicates with the OpenStack Networking server via the OpenStack Networking API, so the following configuration is required:

1. OpenStack Identity authentication:

   ```
   auth_url="$KEYSTONE_SERVICE_PROTOCOL://$KEYSTONE_AUTH_HOST:
   $KEYSTONE_AUTH_PORT/v2.0"
   ```

   For example,

```
http://10.56.51.210:5000/v2.0
```

2. Admin user details:

```
admin_tenant_name $SERVICE_TENANT_NAME
admin_user $Q_ADMIN_USERNAME
admin_password $SERVICE_PASSWORD
```

Namespace

By default the L3 agent makes use of Linux network namespaces in order to support overlapping IP addresses. Requirements for network namespaces support are described in the Limitation section.

If the Linux installation does not support network namespace, you must disable using network namespace in the L3 agent config file (The default value of use_namespaces is True).

```
use_namespaces = False
```

When use_namespaces is set to False, only one router ID can be supported per node. This must be configured via the configuration variable *router_id*.

```
# If use_namespaces is set to False then the agent can only configure one
 router.
# This is done by setting the specific router_id.
router_id = 1064ad16-36b7-4c2f-86f0-daa2bcbd6b2a
```

To configure it, you need to run the OpenStack Networking service and create a router, and then set an ID of the router created to *router_id* in the L3 agent configuration file.

```
$ neutron router-create myrouter1
Created a new router:
+-----------------------+--------------------------------------+
| Field                 | Value                                |
+-----------------------+--------------------------------------+
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 338d42d7-b22e-42c5-9df6-f3674768fe75 |
| name                  | myrouter1                            |
| status                | ACTIVE                               |
| tenant_id             | 0c236f65baa04e6f9b4236b996555d56     |
+-----------------------+--------------------------------------+
```

Multiple floating IP pools

The L3 API in OpenStack Networking supports multiple floating IP pools. In OpenStack Networking, a floating IP pool is represented as an external network and a floating IP is allocated from a subnet associated with the external network. Since each L3 agent can be associated with at most one external network, we need to invoke multiple L3 agent to define multiple floating IP pools. **'gateway_external_network_id'** in L3 agent configuration file indicates the external network that the L3 agent handles. You can run multiple L3 agent instances on one host.

In addition, when you run multiple L3 agents, make sure that **handle_internal_only_routers** is set to **True** only for one L3 agent in an OpenStack

Networking deployment and set to **False** for all other L3 agents. Since the default value of this parameter is True, you need to configure it carefully.

Before starting L3 agents, you need to create routers and external networks, then update the configuration files with UUID of external networks and start L3 agents.

For the first agent, invoke it with the following l3_agent.ini where handle_internal_only_routers is True.

```
handle_internal_only_routers = True
gateway_external_network_id = 2118b11c-011e-4fa5-a6f1-2ca34d372c35
external_network_bridge = br-ex
```

```
python /opt/stack/neutron/bin/neutron-l3-agent
 --config-file /etc/neutron/neutron.conf
 --config-file=/etc/neutron/l3_agent.ini
```

For the second (or later) agent, invoke it with the following l3_agent.ini where handle_internal_only_routers is False.

```
handle_internal_only_routers = False
gateway_external_network_id = e828e54c-850a-4e74-80a8-8b79c6a285d8
external_network_bridge = br-ex-2
```

L3 Metering Agent

There is an option to run a L3 metering agent that will enable layer 3 traffic metering. In general case the metering agent should be launched on all nodes that run the L3 agent:

```
neutron-metering-agent --config-file <neutron config>
--config-file <l3 metering config>
```

A driver needs to be configured that matches the plug-in running on the service. The driver is used to add metering to the routing interface.

Table 4.49. Basic settings

Parameter	Value
Open vSwitch	
interface_driver ($NEUTRON_CONF_DIR/ metering_agent.ini)	neutron.agent.linux.interface.OVSInterfaceDriver
Linux Bridge	
interface_driver ($NEUTRON_CONF_DIR/ metering_agent.ini)	neutron.agent.linux.interface.BridgeInterfaceDriver

Namespace

The metering agent and the L3 agent have to have the same configuration regarding to the network namespaces setting.

Note

If the Linux installation does not support network namespace, you must disable using network namespace in the L3 metering config file (The default value of `use_namespaces` is `True`).

```
use_namespaces = False
```

L3 metering driver

A driver which implements the metering abstraction needs to be configured. Currently there is only one implementation which is based on iptables.

```
driver = neutron.services.metering.drivers.iptables.iptables_driver.
IptablesMeteringDriver
```

L3 metering service driver

To enable L3 metering you have to be sure to set the following parameter in `neutron.conf` on the host that runs `neutron-server`:

```
service_plugins = neutron.services.metering.metering_plugin.MeteringPlugin
```

Limitations

- *No equivalent for nova-network –multi_host flag:* Nova-network has a model where the L3, NAT, and DHCP processing happen on the compute node itself, rather than a dedicated networking node. OpenStack Networking now support running multiple l3-agent and dhcp-agents with load being split across those agents, but the tight coupling of that scheduling with the location of the VM is not supported in Grizzly. The Havana release is expected to include an exact replacement for the –multi_host flag in nova-network.

- *Linux network namespace required on nodes running `neutron-l3-agent` or `neutron-dhcp-agent` if overlapping IPs are in use:* . In order to support overlapping IP addresses, the OpenStack Networking DHCP and L3 agents use Linux network namespaces by default. The hosts running these processes must support network namespaces. To support network namespaces, the following are required:

  - Linux kernel 2.6.24 or newer (with CONFIG_NET_NS=y in kernel configuration) and

  - iproute2 utilities ('ip' command) version 3.1.0 (aka 20111117) or newer

  To check whether your host supports namespaces try running the following as root:

  ```
  # ip netns add test-ns
  # ip netns exec test-ns ifconfig
  ```

  If the preceding commands do not produce errors, your platform is likely sufficient to use the dhcp-agent or l3-agent with namespace. In our experience, Ubuntu 12.04 or later support namespaces as does Fedora 17 and new, but some older RHEL platforms do not by default. It may be possible to upgrade the iproute2 package on a platform that does not support namespaces by default.

  If you need to disable namespaces, make sure the `neutron.conf` used by neutron-server has the following setting:

  ```
  allow_overlapping_ips=False
  ```

  and that the dhcp_agent.ini and l3_agent.ini have the following setting:

```
use_namespaces=False
```

Note

If the host does not support namespaces then the `neutron-l3-agent` and `neutron-dhcp-agent` should be run on different hosts. This is due to the fact that there is no isolation between the IP addresses created by the L3 agent and by the DHCP agent. By manipulating the routing the user can ensure that these networks have access to one another.

If you run both L3 and DHCP services on the same node, you should enable namespaces to avoid conflicts with routes:

```
use_namespaces=True
```

- *No IPv6 support for L3 agent:* The `neutron-l3-agent`, used by many plug-ins to implement L3 forwarding, supports only IPv4 forwarding. Currently, there are no errors provided if you configure IPv6 addresses via the API.
- *ZeroMQ support is experimental*: Some agents, including `neutron-dhcp-agent`, `neutron-openvswitch-agent`, and `neutron-linuxbridge-agent` use RPC to communicate. ZeroMQ is an available option in the configuration file, but has not been tested and should be considered experimental. In particular, issues might occur with ZeroMQ and the dhcp agent.
- *MetaPlugin is experimental*: This release includes a MetaPlugin that is intended to support multiple plug-ins at the same time for different API requests, based on the content of those API requests. The core team has not thoroughly reviewed or tested this functionality. Consider this functionality to be experimental until further validation is performed.

Scalable and highly available DHCP agents

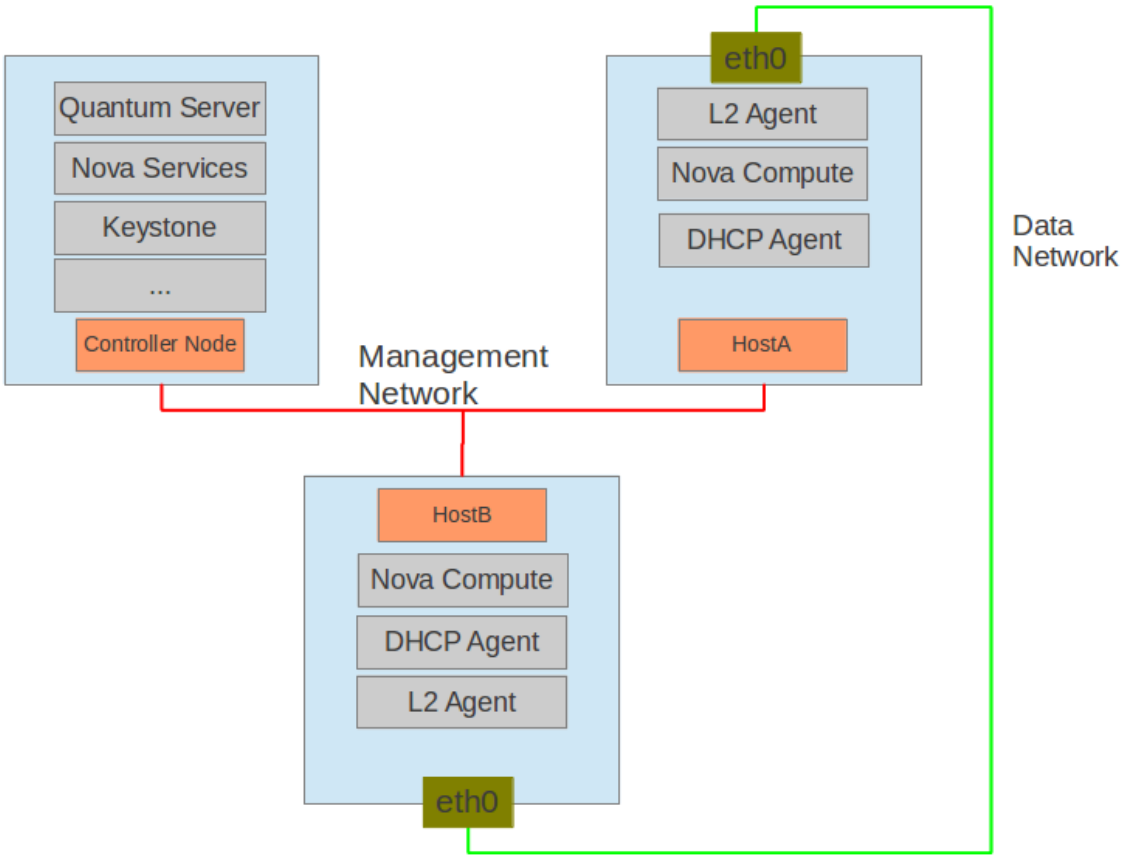
This section describes how to use the agent management (alias agent) and scheduler (alias agent_scheduler) extensions for DHCP agents scalability and HA.

Note

Use the **neutron ext-list** client command to check if these extensions are enabled:

```
$ neutron ext-list -c name -c alias
+-----------------+--------------------------+
| alias           | name                     |
+-----------------+--------------------------+
| agent_scheduler | Agent Schedulers         |
| binding         | Port Binding             |
| quotas          | Quota management support |
| agent           | agent                    |
| provider        | Provider Network         |
| router          | Neutron L3 Router        |
| lbaas           | LoadBalancing service    |
| extraroute      | Neutron Extra Route      |
```

```
+----------------+-------------------------+
```



There will be three hosts in the setup.

Table 4.50. Hosts for Demo

Host	Description
OpenStack Controller host - controlnode	Runs the Neutron, Keystone, and Nova services that are required to deploy VMs. The node must have at least one network interface that is connected to the Management Network. **Note** `nova-network` should not be running because it is replaced by Neutron.
HostA	Runs Nova compute, the Neutron L2 agent and DCHP agent
HostB	Same as HostA

Configuration

- **controlnode - Neutron Server**

  1. Neutron configuration file `/etc/neutron/neutron.conf`:

```
[DEFAULT]
core_plugin = neutron.plugins.linuxbridge.lb_neutron_plugin.
LinuxBridgePluginV2
rabbit_host = controlnode
allow_overlapping_ips = True
host = controlnode
agent_down_time = 5
```

2. Update the plug-in configuration file `/etc/neutron/plugins/linuxbridge/linuxbridge_conf.ini`:

```
[vlans]
tenant_network_type = vlan
network_vlan_ranges = physnet1:1000:2999
[database]
connection = mysql://root:root@127.0.0.1:3306/neutron_linux_bridge
retry_interval = 2
[linux_bridge]
physical_interface_mappings = physnet1:eth0
```

- **HostA and HostB - L2 Agent**

1. Neutron configuration file `/etc/neutron/neutron.conf`:

```
[DEFAULT]
rabbit_host = controlnode
rabbit_password = openstack
# host = HostB on hostb
host = HostA
```

2. Update the plug-in configuration file `/etc/neutron/plugins/linuxbridge/linuxbridge_conf.ini`:

```
[vlans]
tenant_network_type = vlan
network_vlan_ranges = physnet1:1000:2999
[database]
connection = mysql://root:root@127.0.0.1:3306/neutron_linux_bridge
retry_interval = 2
[linux_bridge]
physical_interface_mappings = physnet1:eth0
```

3. Update the nova configuration file `/etc/nova/nova.conf`:

```
[DEFAULT]
network_api_class=nova.network.neutronv2.api.API

neutron_admin_username=neutron
neutron_admin_password=servicepassword
neutron_admin_auth_url=http://controlnode:35357/v2.0/
neutron_auth_strategy=keystone
neutron_admin_tenant_name=servicetenant
neutron_url=http://100.1.1.10:9696/
firewall_driver=nova.virt.firewall.NoopFirewallDriver
```

- **HostA and HostB - DHCP Agent**

1. Update the DHCP configuration file `/etc/neutron/dhcp_agent.ini`:

```
[DEFAULT]
interface_driver = neutron.agent.linux.interface.BridgeInterfaceDriver
```

Commands in agent management and scheduler extensions

The following commands require the tenant running the command to have an admin role.

Note

Ensure that the following environment variables are set. These are used by the various clients to access Keystone.

```
export OS_USERNAME=admin
export OS_PASSWORD=adminpassword
export OS_TENANT_NAME=admin
export OS_AUTH_URL=http://controlnode:5000/v2.0/
```

- **Settings**

  To experiment, you need VMs and a neutron network:

```
$ nova list
+--------------------------------------+-----------+--------+---------------
+
| ID                                   | Name      | Status | Networks
 |
+--------------------------------------+-----------+--------+---------------
+
| c394fcd0-0baa-43ae-a793-201815c3e8ce | myserver1 | ACTIVE | net1=10.0.1.3
 |
| 2d604e05-9a6c-4ddb-9082-8a1fbdcc797d | myserver2 | ACTIVE | net1=10.0.1.4
 |
| c7c0481c-3db8-4d7a-a948-60ce8211d585 | myserver3 | ACTIVE | net1=10.0.1.5
 |
+--------------------------------------+-----------+--------+---------------
+

$ neutron net-list
+--------------------------------------+------
+--------------------------------------+
| id                                   | name | subnets
         |
+--------------------------------------+------
+--------------------------------------+
| 89dca1c6-c7d4-4f7a-b730-549af0fb6e34 | net1 | f6c832e3-9968-46fd-8e45-
d5cf646db9d1 |
+--------------------------------------+------
+--------------------------------------+
```

- **Manage agents in neutron deployment**

  Every agent which supports these extensions will register itself with the neutron server when it starts up.

  1. List all agents:

```
$ neutron agent-list
```

```
+--------------------------------------+--------------------+-------
+-------+---------------+
| id                                   | agent_type         | host  |
 alive | admin_state_up |
+--------------------------------------+--------------------+-------
+-------+---------------+
| 1b69828d-6a9b-4826-87cd-1757f0e27f31 | Linux bridge agent | HostA | :-)
  | True           |
| a0c1c21c-d4f4-4577-9ec7-908f2d48622d | DHCP agent         | HostA | :-)
  | True           |
| ed96b856-ae0f-4d75-bb28-40a47ffd7695 | Linux bridge agent | HostB | :-)
  | True           |
| f28aa126-6edb-4ea5-a81e-8850876bc0a8 | DHCP agent         | HostB | :-)
  | True           |
+--------------------------------------+--------------------+-------
+-------+---------------+
```

The output shows information for four agents. The `alive` field shows `:-)` if the agent reported its state within the period defined by the `agent_down_time` option in the `neutron.conf` file. Otherwise the `alive` is `xxx`.

2. List the DHCP agents that host a specified network

   In some deployments, one DHCP agent is not enough to hold all network data. In addition, you must have a backup for it even when the deployment is small. The same network can be assigned to more than one DHCP agent and one DHCP agent can host more than one network.

   List DHCP agents that a a specified network:

```
$ neutron dhcp-agent-list-hosting-net net1
+--------------------------------------+-------+----------------+-------+
| id                                   | host  | admin_state_up | alive |
+--------------------------------------+-------+----------------+-------+
| a0c1c21c-d4f4-4577-9ec7-908f2d48622d | HostA | True           | :-)   |
+--------------------------------------+-------+----------------+-------+
```

3. List the networks hosted by a given DHCP agent.

   This command is to show which networks a given dhcp agent is managing.

```
$ neutron net-list-on-dhcp-agent a0c1c21c-d4f4-4577-9ec7-908f2d48622d
+--------------------------------------+------
+-----------------------------------------------+
| id                                   | name | subnets
                      |
+--------------------------------------+------
+-----------------------------------------------+
| 89dca1c6-c7d4-4f7a-b730-549af0fb6e34 | net1 | f6c832e3-9968-46fd-8e45-
d5cf646db9d1  10.0.1.0/24 |
+--------------------------------------+------
+-----------------------------------------------+
```

4. Show agent details.

   The **agent-list** command shows details for a specified agent:

```
$ neutron agent-show  a0c1c21c-d4f4-4577-9ec7-908f2d48622d
+--------------------
+-------------------------------------------------------------+
| Field               | Value
      |
+--------------------
+-------------------------------------------------------------+
| admin_state_up      | True
      |
| agent_type          | DHCP agent
      |
| alive               | False
      |
| binary              | neutron-dhcp-agent
      |
| configurations      | {
      |
|                     |      "subnets": 1,
      |
|                     |      "use_namespaces": true,
      |
|                     |      "dhcp_driver": "neutron.agent.linux.dhcp.
Dnsmasq",  |
|                     |      "networks": 1,
      |
|                     |      "dhcp_lease_time": 120,
      |
|                     |      "ports": 3
      |
|                     | }
      |
| created_at          | 2013-03-16T01:16:18.000000
      |
| description         |
      |
| heartbeat_timestamp | 2013-03-17T01:37:22.000000
      |
| host                | HostA
      |
| id                  | 58f4ce07-6789-4bb3-aa42-ed3779db2b03
      |
| started_at          | 2013-03-16T06:48:39.000000
      |
| topic               | dhcp_agent
      |
+--------------------
+-------------------------------------------------------------+
```

In this output, `heartbeat_timestamp` is the time on the neutron server. You do not need to synchronize all agents to this time for this extension to run correctly. `configurations` describes the static configuration for the agent or run time data. This agent is a DHCP agent and it hosts one network, one subnet, and three ports.

Different types of agents show different details. The following output shows information for a Linux bridge agent:

```
$ neutron agent-show ed96b856-ae0f-4d75-bb28-40a47ffd7695
```

```
+--------------------+--------------------------------------+
| Field              | Value                                |
+--------------------+--------------------------------------+
| admin_state_up     | True                                 |
| binary             | neutron-linuxbridge-agent            |
| configurations     | {                                    |
|                    |      "physnet1": "eth0",             |
|                    |      "devices": "4"                  |
|                    | }                                    |
| created_at         | 2013-03-16T01:49:52.000000           |
| description        |                                      |
| disabled           | False                                |
| group              | agent                                |
| heartbeat_timestamp| 2013-03-16T01:59:45.000000           |
| host               | HostB                                |
| id                 | ed96b856-ae0f-4d75-bb28-40a47ffd7695 |
| topic              | N/A                                  |
| started_at         | 2013-03-16T06:48:39.000000           |
| type               | Linux bridge agent                   |
+--------------------+--------------------------------------+
```

The output shows `bridge-mapping` and the number of virtual network devices on this L2 agent.

- **Manage assignment of networks to DHCP agent**

Now that you have run the **net-list-on-dhcp-agent** and **dhcp-agent-list-hosting-net** commands, you can add a network to a DHCP agent and remove one from it.

1. Default scheduling.

   When you create a network with one port, you can schedule it to an active DHCP agent. If many active DHCP agents are running, select one randomly. You can design more sophisticated scheduling algorithms in the same way as `nova-schedule` later on.

```
$ neutron net-create net2
$ neutron subnet-create net2 9.0.1.0/24 --name subnet2
$ neutron port-create net2
$ neutron dhcp-agent-list-hosting-net net2
+--------------------------------------+-------+----------------+-------+
| id                                   | host  | admin_state_up | alive |
+--------------------------------------+-------+----------------+-------+
| a0c1c21c-d4f4-4577-9ec7-908f2d48622d | HostA | True           | :-)   |
+--------------------------------------+-------+----------------+-------+
```

   It is allocated to DHCP agent on HostA. If you want to validate the behavior through the **dnsmasq** command, you must create a subnet for the network because the DHCP agent starts the `dnsmasq` service only if there is a DHCP.

2. Assign a network to a given DHCP agent.

   To add another DHCP agent to host the network, run this command:

```
$ neutron dhcp-agent-network-add f28aa126-6edb-4ea5-a81e-8850876bc0a8 net2
Added network net2 to dhcp agent
$ neutron dhcp-agent-list-hosting-net net2
+--------------------------------------+-------+----------------+-------+
```

```
| id                                   | host  | admin_state_up | alive |
+--------------------------------------+-------+----------------+-------+
| a0c1c21c-d4f4-4577-9ec7-908f2d48622d | HostA | True           | :-)   |
| f28aa126-6edb-4ea5-a81e-8850876bc0a8 | HostB | True           | :-)   |
+--------------------------------------+-------+----------------+-------+
```

Both DHCP agents host the `net2` network.

3. Remove a network from a specified DHCP agent.

   This command is the sibling command for the previous one. Remove `net2` from the DHCP agent for HostA:

```
$ neutron dhcp-agent-network-remove a0c1c21c-d4f4-4577-9ec7-908f2d48622d
 net2
Removed network net2 to dhcp agent
$ neutron dhcp-agent-list-hosting-net net2
+--------------------------------------+-------+----------------+-------+
| id                                   | host  | admin_state_up | alive |
+--------------------------------------+-------+----------------+-------+
| f28aa126-6edb-4ea5-a81e-8850876bc0a8 | HostB | True           | :-)   |
+--------------------------------------+-------+----------------+-------+
```

You can see that only the DHCP agent for HostB is hosting the `net2` network.

- **HA of DHCP agents**

Boot a VM on net2. Let both DHCP agents host `net2`. Fail the agents in turn to see if the VM can still get the desired IP.

1. Boot a VM on net2.

```
$ neutron net-list
+--------------------------------------+------
+-----------------------------------------------+
| id                                   | name | subnets
                       |
+--------------------------------------+------
+-----------------------------------------------+
| 89dca1c6-c7d4-4f7a-b730-549af0fb6e34 | net1 | f6c832e3-9968-46fd-8e45-
d5cf646db9d1  10.0.1.0/24|
| 9b96b14f-71b8-4918-90aa-c5d705606b1a | net2 | 6979b71a-0ae8-448c-
aa87-65f68eedcaaa  9.0.1.0/24 |
+--------------------------------------+------
+-----------------------------------------------+
$ nova boot --image tty --flavor 1 myserver4 \
  --nic net-id=9b96b14f-71b8-4918-90aa-c5d705606b1a
$ nova list
+--------------------------------------+----------+--------
+--------------+
| ID                                   | Name     | Status | Networks
   |
+--------------------------------------+----------+--------
+--------------+
| c394fcd0-0baa-43ae-a793-201815c3e8ce | myserver1 | ACTIVE | net1=10.0.1.
3 |
| 2d604e05-9a6c-4ddb-9082-8a1fbdcc797d | myserver2 | ACTIVE | net1=10.0.1.
4 |
```

```
| c7c0481c-3db8-4d7a-a948-60ce8211d585 | myserver3 | ACTIVE | net1=10.0.1.
5 |
| f62f4731-5591-46b1-9d74-f0c901de567f | myserver4 | ACTIVE | net2=9.0.1.
2  |
+--------------------------------------+-----------+--------
+---------------+
```

2. Make sure both DHCP agents hosting 'net2'.

   Use the previous commands to assign the network to agents.

```
$ neutron dhcp-agent-list-hosting-net net2
+--------------------------------------+-------+---------------+-------+
| id                                   | host  | admin_state_up | alive |
+--------------------------------------+-------+---------------+-------+
| a0c1c21c-d4f4-4577-9ec7-908f2d48622d | HostA | True           | :-)   |
| f28aa126-6edb-4ea5-a81e-8850876bc0a8 | HostB | True           | :-)   |
+--------------------------------------+-------+---------------+-------+
```

3. **Procedure 4.2. To test the HA**

   1. Log in to the `myserver4` VM, and run `udhcpc`, `dhclient` or other DHCP client.
   2. Stop the DHCP agent on HostA. Besides stopping the `neutron-dhcp-agent` binary, you must stop the **dnsmasq** processes.
   3. Run a DHCP client in VM to see if it can get the wanted IP.
   4. Stop the DHCP agent on HostB too.
   5. Run **udhcpc** in the VM; it cannot get the wanted IP.
   6. Start DHCP agent on HostB. The VM gets the wanted IP again.

- Disable and remove an agent

  An administrator might want to disable an agent if a system hardware or software upgrade is planned. Some agents that support scheduling also support disabling and enabling agents, such as L3 and DHCP agents. After the agent is disabled, the scheduler does not schedule new resources to the agent. After the agent is disabled, you can safely remove the agent. Remove the resources on the agent before you delete the agent.

  To run the following commands, you must stop the DHCP agent on HostA.

```
$ neutron agent-update --admin-state-up False a0c1c21c-
d4f4-4577-9ec7-908f2d48622d
$ neutron agent-list
+--------------------------------------+--------------------+-------+-------
+---------------+
| id                                   | agent_type         | host  | alive
 | admin_state_up |
+--------------------------------------+--------------------+-------+-------
+---------------+
| 1b69828d-6a9b-4826-87cd-1757f0e27f31 | Linux bridge agent | HostA | :-)
 | True           |
| a0c1c21c-d4f4-4577-9ec7-908f2d48622d | DHCP agent         | HostA | :-)
 | False          |
```

```
| ed96b856-ae0f-4d75-bb28-40a47ffd7695 | Linux bridge agent | HostB | :-)
 | True           |
| f28aa126-6edb-4ea5-a81e-8850876bc0a8 | DHCP agent         | HostB | :-)
 | True           |
+--------------------------------------+--------------------+-------+-------
+----------------+
$ neutron agent-delete a0c1c21c-d4f4-4577-9ec7-908f2d48622d
Deleted agent: a0c1c21c-d4f4-4577-9ec7-908f2d48622d
$ neutron agent-list
+--------------------------------------+--------------------+-------+-------
+----------------+
| id                                   | agent_type         | host  | alive
 | admin_state_up |
+--------------------------------------+--------------------+-------+-------
+----------------+
| 1b69828d-6a9b-4826-87cd-1757f0e27f31 | Linux bridge agent | HostA | :-)
 | True           |
| ed96b856-ae0f-4d75-bb28-40a47ffd7695 | Linux bridge agent | HostB | :-)
 | True           |
| f28aa126-6edb-4ea5-a81e-8850876bc0a8 | DHCP agent         | HostB | :-)
 | True           |
+--------------------------------------+--------------------+-------+-------
+----------------+
```

After deletion, if you restart the DHCP agent, it appears on the agent list again.

5. Dashboard

Table of Contents

This chapter describes how to configure the OpenStack dashboard with Apache web server.

Configure the dashboard

You can configure the dashboard for a simple HTTP deployment.

You can configure the dashboard for a secured HTTPS deployment. While the standard installation uses a non-encrypted HTTP channel, you can enable SSL support for the dashboard.

Also, you can configure the size of the VNC window in the dashboard.

Configure the dashboard for HTTP

You can configure the dashboard for a simple HTTP deployment. The standard installation uses a non-encrypted HTTP channel.

1. Specify the host for your OpenStack Identity Service endpoint in the `/etc/openstack-dashboard/local_settings.py` file with the `OPENSTACK_HOST` setting.

   The following example shows this setting:

```
import os

from django.utils.translation import ugettext_lazy as _

DEBUG = False
TEMPLATE_DEBUG = DEBUG
PROD = True
USE_SSL = False

SITE_BRANDING = 'OpenStack Dashboard'

# Ubuntu-specific: Enables an extra panel in the 'Settings' section
# that easily generates a Juju environments.yaml for download,
# preconfigured with endpoints and credentials required for bootstrap
# and service deployment.
ENABLE_JUJU_PANEL = True

# Note: You should change this value
SECRET_KEY = 'elj1IWiLoWHgryYxFT6j7cM5fGOOxWY0'

# Specify a regular expression to validate user passwords.
# HORIZON_CONFIG = {
#     "password_validator": {
#         "regex": '.*',
#         "help_text": _("Your password does not meet the requirements.")
#     }
# }

LOCAL_PATH = os.path.dirname(os.path.abspath(__file__))

CACHES = {
```

```
 'default': {
  'BACKEND' : 'django.core.cache.backends.memcached.MemcachedCache',
  'LOCATION' : '127.0.0.1:11211'
 }
}

# Send email to the console by default
EMAIL_BACKEND = 'django.core.mail.backends.console.EmailBackend'
# Or send them to /dev/null
#EMAIL_BACKEND = 'django.core.mail.backends.dummy.EmailBackend'

# Configure these for your outgoing email host
# EMAIL_HOST = 'smtp.my-company.com'
# EMAIL_PORT = 25
# EMAIL_HOST_USER = 'djangomail'
# EMAIL_HOST_PASSWORD = 'top-secret!'

# For multiple regions uncomment this configuration, and add (endpoint, title).
# AVAILABLE_REGIONS = [
#     ('http://cluster1.example.com:5000/v2.0', 'cluster1'),
#     ('http://cluster2.example.com:5000/v2.0', 'cluster2'),
# ]

OPENSTACK_HOST = "127.0.0.1"
OPENSTACK_KEYSTONE_URL = "http://%s:5000/v2.0" % OPENSTACK_HOST
OPENSTACK_KEYSTONE_DEFAULT_ROLE = "Member"

# The OPENSTACK_KEYSTONE_BACKEND settings can be used to identify the
# capabilities of the auth backend for Keystone.
# If Keystone has been configured to use LDAP as the auth backend then set
# can_edit_user to False and name to 'ldap'.
#
# TODO(tres): Remove these once Keystone has an API to identify auth backend.
OPENSTACK_KEYSTONE_BACKEND = {
    'name': 'native',
    'can_edit_user': True
}

# OPENSTACK_ENDPOINT_TYPE specifies the endpoint type to use for the endpoints
# in the Keystone service catalog. Use this setting when Horizon is running
# external to the OpenStack environment. The default is 'internalURL'.
#OPENSTACK_ENDPOINT_TYPE = "publicURL"

# The number of Swift containers and objects to display on a single page before
# providing a paging element (a "more" link) to paginate results.
API_RESULT_LIMIT = 1000

# If you have external monitoring links, eg:
# EXTERNAL_MONITORING = [
#     ['Nagios','http://foo.com'],
#     ['Ganglia','http://bar.com'],
# ]

LOGGING = {
        'version': 1,
        # When set to True this will disable all logging except
        # for loggers specified in this configuration dictionary. Note that
        # if nothing is specified here and disable_existing_loggers is True,
        # django.db.backends will still log unless it is disabled explicitly.
        'disable_existing_loggers': False,
        'handlers': {
            'null': {
                'level': 'DEBUG',
                'class': 'django.utils.log.NullHandler',
                },
            'console': {
                # Set the level to "DEBUG" for verbose output logging.
                'level': 'INFO',
                'class': 'logging.StreamHandler',
                },
            },
        'loggers': {
            # Logging from django.db.backends is VERY verbose, send to null
            # by default.
            'django.db.backends': {
                'handlers': ['null'],
                'propagate': False,
                },
            'horizon': {
```

```
                'handlers': ['console'],
                'propagate': False,
            },
            'novaclient': {
                'handlers': ['console'],
                'propagate': False,
            },
            'keystoneclient': {
                'handlers': ['console'],
                'propagate': False,
            },
            'nose.plugins.manager': {
                'handlers': ['console'],
                'propagate': False,
            }
        }
}
```

The service catalog configuration in the Identity Service determines whether a service appears in the dashboard. For the full listing, see Horizon Settings and Configuration.

2. Restart Apache http server. For Ubuntu/Debian/SUSE:

```
# service apache2 restart
```

or for Fedora/RHEL/CentOS:

```
# service httpd restart
```

Next, restart memcached:

```
# service memcached restart
```

Configure the dashboard for HTTPS

You can configure the dashboard for a secured HTTPS deployment. While the standard installation uses a non-encrypted HTTP channel, you can enable SSL support for the dashboard.

The following example uses the domain, "http://openstack.example.com." Use a domain that fits your current setup.

1. In `/etc/openstack-dashboard/local_settings.py` update the following directives:

```
USE_SSL = True
CSRF_COOKIE_SECURE = True
SESSION_COOKIE_SECURE = True
SESSION_COOKIE_HTTPONLY = True
```

The first option is required to enable HTTPS. The other recommended settings defend against cross-site scripting and require HTTPS.

2. Edit `/etc/apache2/ports.conf` and add the following line:

```
NameVirtualHost *:443
```

3. Edit `/etc/apache2/conf.d/openstack-dashboard.conf:`

Before:

```
WSGIScriptAlias / /usr/share/openstack-dashboard/openstack_dashboard/wsgi/django.wsgi
WSGIDaemonProcess horizon user=www-data group=www-data processes=3 threads=10
Alias /static /usr/share/openstack-dashboard/openstack_dashboard/static/
<Directory /usr/share/openstack-dashboard/openstack_dashboard/wsgi>
# For Apache http server 2.2 and earlier:
Order allow,deny
Allow from all

# For Apache http server 2.4 and later:
# Require all granted
</Directory>
```

After:

```
<VirtualHost *:80>
ServerName openstack.example.com
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteCond %{HTTPS} off
RewriteRule (.*) https://%{HTTP_HOST}%{REQUEST_URI}
</IfModule>
<IfModule !mod_rewrite.c>
RedirectPermanent / https://openstack.example.com
</IfModule>
</VirtualHost>
<VirtualHost *:443>
ServerName openstack.example.com

SSLEngine On
# Remember to replace certificates and keys with valid paths in your environment
SSLCertificateFile /etc/apache2/SSL/openstack.example.com.crt
SSLCACertificateFile /etc/apache2/SSL/openstack.example.com.crt
SSLCertificateKeyFile /etc/apache2/SSL/openstack.example.com.key
SetEnvIf User-Agent ".*MSIE.*" nokeepalive ssl-unclean-shutdown

# HTTP Strict Transport Security (HSTS) enforces that all communications
# with a server go over SSL. This mitigates the threat from attacks such
# as SSL-Strip which replaces links on the wire, stripping away https prefixes
# and potentially allowing an attacker to view confidential information on the
# wire
Header add Strict-Transport-Security "max-age=15768000"

WSGIScriptAlias / /usr/share/openstack-dashboard/openstack_dashboard/wsgi/django.wsgi
WSGIDaemonProcess horizon user=www-data group=www-data processes=3 threads=10
Alias /static /usr/share/openstack-dashboard/openstack_dashboard/static/
<Directory /usr/share/openstack-dashboard/openstack_dashboard/wsgi>
# For Apache http server 2.2 and earlier:
Order allow,deny
Allow from all

# For Apache http server 2.4 and later:
# Require all granted
</Directory>
</VirtualHost>
```

In this configuration, Apache http server listens on the port 443 and redirects all the hits to the HTTPS protocol for all the non-secured requests. The secured section defines the private key, public key, and certificate to use.

4. Restart Apache http server. For Debian/Ubuntu/SUSE:

```
# service apache2 restart
```

Or for Fedora/RHEL/CentOS:

```
# service httpd restart
```

Next, restart memcached:

```
# service memcached restart
```

If you try to access the dashboard through HTTP, the browser redirects you to the HTTPS page.

Change the size of the dashboard VNC window

The `_detail_vnc.html` file defines the size of the VNC window. To change the window size, edit this file.

1. Edit `/usr/share/pyshared/horizon/dashboards/nova/instances/templates/instances/_detail_vnc.html`.

2. Modify the `width` and `height` parameters, as follows:

```
<iframe src="{{ vnc_url }}" width="720" height="430"></iframe>
```

Customize the dashboard

Adapted from How To Custom Brand The OpenStack "Horizon" Dashboard.

You install the OpenStack dashboard through the `openstack-dashboard` package. You can customize the dashboard with your own colors, logo, and site title through a CSS file.

Canonical also provides an `openstack-dashboard-ubuntu-theme` package that brands the Python-based Django interface.

1. Create a graphical logo with a transparent background. The text `TGen Cloud` in this example is rendered through `.png` files of multiple sizes created with a graphics program.

   Use a 200×27 for the logged-in banner graphic, and 365×50 for the login screen graphic.

2. Set the HTML title, which appears at the top of the browser window, by adding the following line to `/etc/openstack-dashboard/local_settings.py`:

   ```
   SITE_BRANDING = "Example, Inc. Cloud"
   ```

3. Upload your new graphic files to the following location: `/usr/share/openstack-dashboard/openstack_dashboard/static/dashboard/img/`

4. Create a CSS style sheet in the following directory: `/usr/share/openstack-dashboard/openstack_dashboard/static/dashboard/css/`

5. Edit your CSS file to override the Ubuntu customizations in the `ubuntu.css` file.

   Change the colors and image file names as appropriate, though the relative directory paths should be the same. The following example file shows you how to customize your CSS file:

```
/*
* New theme colors for dashboard that override the defaults:
*  dark blue: #355796 / rgb(53, 87, 150)
*  light blue: #BAD3E1 / rgb(186, 211, 225)
*
```

```
* By Preston Lee <plee@tgen.org>
*/
h1.brand {
background: #355796 repeat-x top left;
border-bottom: 2px solid #BAD3E1;
}
h1.brand a {
background: url(../img/my_cloud_logo_small.png) top left no-repeat;
}
#splash .login {
background: #355796 url(../img/my_cloud_logo_medium.png) no-repeat center 35px;
}
#splash .login .modal-header {
border-top: 1px solid #BAD3E1;
}
.btn-primary {
background-image: none !important;
background-color: #355796 !important;
border: none !important;
box-shadow: none;
}
.btn-primary:hover,
.btn-primary:active {
border: none;
box-shadow: none;
background-color: #BAD3E1 !important;
text-decoration: none;
}
```

6. Open the following HTML template in an editor: `/usr/share/openstack-dashboard/openstack_dashboard/templates/_stylesheets.html`

7. Add a line to include your `custom.css` file:

```
...
 <link href='{{ STATIC_URL }}bootstrap/css/bootstrap.min.css' media='screen' rel='stylesheet' />
 <link href='{{ STATIC_URL }}dashboard/css/{% choose_css %}' media='screen' rel='stylesheet' />
 <link href='{{ STATIC_URL }}dashboard/css/custom.css' media='screen' rel='stylesheet' />
 ...
```

8. Restart apache:

   On Ubuntu:

```
$ sudo service apache2 restart
```

   On Fedora, RHEL, CentOS:

```
$ sudo service httpd restart
```

   On openSUSE:

```
$ sudo service apache2 restart
```

9. Reload the dashboard in your browser to view your changes.

   Modify your CSS file as appropriate.

6. Object Storage

Table of Contents

OpenStack Object Storage uses multiple configuration files for multiple services and background daemons, and **paste.deploy** to manage server configurations. Default configuration options appear in the `[DEFAULT]` section. You can override the default values by setting values in the other sections.

Introduction to Object Storage

Object Storage is a robust, highly scalable and fault tolerant storage platform for unstructured data such as objects. Objects are stored bits, accessed through a RESTful, HTTP-based interface. You cannot access data at the block or file level. Object Storage is commonly used to archive and back up data, with use cases in virtual machine image, photo, video and music storage.

Object Storage provides a high degree of availability, throughput, and performance with its scale out architecture. Each object is replicated across multiple servers, residing within the same data center or across data centers, which mitigates the risk of network and hardware failure. In the event of hardware failure, Object Storage will automatically copy objects to a new location to ensure that there are always three copies available. Object Storage is an eventually consistent distributed storage platform; it sacrifices consistency for maximum availability and partition tolerance. Object Storage enables you to create a reliable platform by using commodity hardware and inexpensive storage.

For more information, review the key concepts in the developer documentation at docs.openstack.org/developer/swift/.

Object Storage general service configuration

Most Object Storage services fall into two categories, Object Storage's wsgi servers and background daemons.

Object Storage uses paste.deploy to manage server configurations. Read more at http://pythonpaste.org/deploy/.

Default configuration options are set in the `[DEFAULT]` section, and any options specified there can be overridden in any of the other sections when the syntax `set option_name = value` is in place.

Configuration for servers and daemons can be expressed together in the same file for each type of server, or separately. If a required section for the service trying to start is missing there will be an error. The sections not used by the service are ignored.

Consider the example of an object storage node. By convention configuration for the object-server, object-updater, object-replicator, and object-auditor exist in a single file `/etc/swift/object-server.conf`:

```
[DEFAULT]

[pipeline:main]
pipeline = object-server

[app:object-server]
use = egg:swift#object

[object-replicator]
reclaim_age = 259200

[object-updater]

[object-auditor]
```

Object Storage services expect a configuration path as the first argument:

```
$ swift-object-auditor
Usage: swift-object-auditor CONFIG [options]

Error: missing config path argument
```

If you omit the object-auditor section this file can not be used as the configuration path when starting the **swift-object-auditor** daemon:

```
$ swift-object-auditor /etc/swift/object-server.conf
Unable to find object-auditor config section in /etc/swift/object-server.conf
```

If the configuration path is a directory instead of a file all of the files in the directory with the file extension ".conf" will be combined to generate the configuration object which is delivered to the Object Storage service. This is referred to generally as "directory based configuration".

Directory based configuration leverages ConfigParser's native multi-file support. Files ending in ".conf" in the given directory are parsed in lexicographical order. File names starting with '.' are ignored. A mixture of file and directory configuration paths is not supported - if the configuration path is a file, only that file will be parsed.

The Object Storage service management tool `swift-init` has adopted the convention of looking for `/etc/swift/{type}-server.conf.d/` if the file `/etc/swift/{type}-server.conf` file does not exist.

When using directory based configuration, if the same option under the same section appears more than once in different files, the last value parsed is said to override previous occurrences. You can ensure proper override precedence by prefixing the files in the configuration directory with numerical values, as in the following example file layout:

```
/etc/swift/
    default.base
    object-server.conf.d/
        000_default.conf -> ../default.base
        001_default-override.conf
        010_server.conf
        020_replicator.conf
        030_updater.conf
        040_auditor.conf
```

You can inspect the resulting combined configuration object using the **swift-config** command line tool.

Object server configuration

Find an example object server configuration at `etc/object-server.conf-sample` in the source code repository.

The available configuration options are:

Table 6.1. Description of configuration options for [DEFAULT] in object-server.conf-sample

Configuration option=Default value	Description
bind_ip=0.0.0.0	IP Address for server to bind to
bind_port=6000	Port for server to bind to
bind_timeout=30	Seconds to attempt bind before giving up
backlog=4096	Maximum number of allowed pending TCP connections
user=swift	User to run as
swift_dir=/etc/swift	Swift configuration directory
devices=/srv/node	Parent directory of where devices are mounted
mount_check=true	Whether or not check if the devices are mounted to prevent accidentally writing to the root device
disable_fallocate=false	Disable "fast fail" fallocate checks if the underlying filesystem does not support it.
expiring_objects_container_divisor=86400	No help text available for this option
workers=auto	a much higher value, one can reduce the impact of slow file system operations in one request from negatively impacting other requests.
max_clients=1024	Maximum number of clients one worker can process simultaneously Lowering the number of clients handled per worker, and raising the number of workers can lessen the impact that a CPU intensive, or blocking, request can have on other requests served by the same worker. If the maximum number of clients is set to one, then a given worker will not perform another call while processing, allowing other workers a chance to process it.
log_name=swift	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
log_custom_handlers=	Comma-separated list of functions to call to setup custom log handlers.

Configuration option=Default value	Description
log_udp_host=	If not set, the UDB receiver for syslog is disabled.
log_udp_port=514	Port value for UDB receiver, if enabled.
log_statsd_host=localhost	If not set, the StatsD feature is disabled.
log_statsd_port=8125	Port value for the StatsD server.
log_statsd_default_sample_rate=1.0	Defines the probability of sending a sample for any given event or timing measurement.
log_statsd_sample_rate_factor=1.0	Not recommended to set this to a value less than 1.0, if frequency of logging is too high, tune the log_statsd_default_sample_rate instead.
log_statsd_metric_prefix=	Value will be prepended to every metric sent to the StatsD server.
eventlet_debug=false	If true, turn on debug logging for eventlet
fallocate_reserve=0	You can set fallocate_reserve to the number of bytes you'd like fallocate to reserve, whether there is space for the given file size or not. This is useful for systems that behave badly when they completely run out of space; you can make the services pretend they're out of space early. server. For most cases, this should be `egg:swift#object`.

Table 6.2. Description of configuration options for [app:object-server] in object-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#object	Entry point of paste.deploy in the server
set log_name=object-server	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_requests=true	Whether or not to log requests
set log_address=/dev/log	No help text available for this option
node_timeout=3	Request timeout to external services
conn_timeout=0.5	Connection timeout to external services
network_chunk_size=65536	Size of chunks to read/write over the network
disk_chunk_size=65536	Size of chunks to read/write to disk
max_upload_time=86400	Maximum time allowed to upload an object
slow=0	If > 0, Minimum time in seconds for a PUT or DELETE request to complete
keep_cache_size=5424880	Largest object size to keep in buffer cache
keep_cache_private=false	Allow non-public objects to stay in kernel's buffer cache
mb_per_sync=512	On PUT requests, sync file every n MB
allowed_headers=Content-Disposition, Content-Encoding, X-Delete-At, X-Object-Manifest, X-Static-Large-Object	Comma-separated list of headers that can be set in metadata of an object
auto_create_account_prefix=.	Prefix to use when automatically creating accounts
replication_server=false	If defined, tells server how to handle replication verbs in requests. When set to True (or 1), only replication verbs will be accepted. When set to False, replication verbs will be rejected. When undefined, server will accept any verb in the request.
threads_per_disk=0	Size of the per-disk thread pool used for performing disk I/O. The default of 0 means to not use a per-disk thread pool. It is recommended to keep this value small, as large

Configuration option=Default value	Description
	values can result in high read latencies due to large queue depths. A good starting point is 4 threads per disk.

Table 6.3. Description of configuration options for [pipeline:main] in object-server.conf-sample

Configuration option=Default value	Description
pipeline=healthcheck recon object-server	No help text available for this option

Table 6.4. Description of configuration options for [object-replicator] in object-server.conf-sample

Configuration option=Default value	Description
log_name=object-replicator	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
vm_test_mode=no	Indicates that you are using a VM environment
daemonize=on	Whether or not to run replication as a daemon
run_pause=30	Time in seconds to wait between replication passes
concurrency=1	Number of replication workers to spawn
stats_interval=300	Interval in seconds between logging replication statistics
rsync_timeout=900	Max duration (seconds) of a partition rsync
rsync_bwlimit=0	No help text available for this option
rsync_io_timeout=30	Passed to rsync for a max duration (seconds) of an I/O op
http_timeout=60	Maximum duration for an HTTP request
lockup_timeout=1800	Attempts to kill all workers if nothing replications for lockup_timeout seconds
reclaim_age=604800	Time elapsed in seconds before an object can be reclaimed
ring_check_interval=15	How often (in seconds) to check the ring
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored
rsync_error_log_line_length=0	No help text available for this option

Table 6.5. Description of configuration options for [object-updater] in object-server.conf-sample

Configuration option=Default value	Description
log_name=object-updater	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
interval=300	Minimum time for a pass to take
concurrency=1	Number of replication workers to spawn
node_timeout=10	Request timeout to external services
conn_timeout=0.5	Connection timeout to external services
slowdown=0.01	Time in seconds to wait between objects
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored

Table 6.6. Description of configuration options for [object-auditor] in object-server.conf-sample

Configuration option=Default value	Description
log_name=object-auditor	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
files_per_second=20	Maximum files audited per second. Should be tuned according to individual system specs. 0 is unlimited.
bytes_per_second=10000000	Maximum bytes audited per second. Should be tuned according to individual system specs. 0 is unlimited. mounted to prevent accidentally writing to the root device process simultaneously (it will actually accept(2) N + 1). Setting this to one (1) will only handle one request at a time, without accepting another request concurrently. By increasing the number of workers to a much higher value, one can reduce the impact of slow file system operations in one request from negatively impacting other requests. underlying filesystem does not support it. to setup custom log handlers. bytes you'd like fallocate to reserve, whether there is space for the given file size or not. This is useful for systems that behave badly when they completely run out of space; you can make the services pretend they're out of space early. container server. For most cases, this should be `egg:swift#container`.
log_time=3600	Frequency of status logs in seconds.
zero_byte_files_per_second=50	Maximum zero byte files audited per second.
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored
object_size_stats=	No help text available for this option

Table 6.7. Description of configuration options for [filter:healthcheck] in object-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#healthcheck	Entry point of paste.deploy in the server
disable_path=	No help text available for this option

Table 6.8. Description of configuration options for [filter:recon] in object-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#recon	Entry point of paste.deploy in the server
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored
recon_lock_path=/var/lock	No help text available for this option

Sample object server configuration file

```
[DEFAULT]
# bind_ip = 0.0.0.0
# bind_port = 6000
# bind_timeout = 30
# backlog = 4096
# user = swift
# swift_dir = /etc/swift
```

```
# devices = /srv/node
# mount_check = true
# disable_fallocate = false
# expiring_objects_container_divisor = 86400
# expiring_objects_account_name = expiring_objects
#
# Use an integer to override the number of pre-forked processes that will
# accept connections.
# workers = auto
#
# Maximum concurrent requests per worker
# max_clients = 1024
#
# You can specify default log routing here if you want:
# log_name = swift
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# comma separated list of functions to call to setup custom log handlers.
# functions get passed: conf, name, log_to_console, log_route, fmt, logger,
# adapted_logger
# log_custom_handlers =
#
# If set, log_udp_host will override log_address
# log_udp_host =
# log_udp_port = 514
#
# You can enable StatsD logging here:
# log_statsd_host = localhost
# log_statsd_port = 8125
# log_statsd_default_sample_rate = 1.0
# log_statsd_sample_rate_factor = 1.0
# log_statsd_metric_prefix =
#
# eventlet_debug = false
#
# You can set fallocate_reserve to the number of bytes you'd like fallocate to
# reserve, whether there is space for the given file size or not.
# fallocate_reserve = 0
#
# Time to wait while attempting to connect to another backend node.
# conn_timeout = 0.5
# Time to wait while sending each chunk of data to another backend node.
# node_timeout = 3
# Time to wait while receiving each chunk of data from a client or another
# backend node.
# client_timeout = 60
#
# network_chunk_size = 65536
# disk_chunk_size = 65536

[pipeline:main]
pipeline = healthcheck recon object-server

[app:object-server]
use = egg:swift#object
# You can override the default log routing for this app here:
# set log_name = object-server
# set log_facility = LOG_LOCAL0
```

```
# set log_level = INFO
# set log_requests = true
# set log_address = /dev/log
#
# max_upload_time = 86400
# slow = 0
#
# Objects smaller than this are not evicted from the buffercache once read
# keep_cache_size = 5424880
#
# If true, objects for authenticated GET requests may be kept in buffer cache
# if small enough
# keep_cache_private = false
#
# on PUTs, sync data every n MB
# mb_per_sync = 512
#
# Comma separated list of headers that can be set in metadata on an object.
# This list is in addition to X-Object-Meta-* headers and cannot include
# Content-Type, etag, Content-Length, or deleted
# allowed_headers = Content-Disposition, Content-Encoding, X-Delete-At, X-
Object-Manifest, X-Static-Large-Object
#
# auto_create_account_prefix = .
#
# A value of 0 means "don't use thread pools". A reasonable starting point is
# 4.
# threads_per_disk = 0
#
# Configure parameter for creating specific server
# To handle all verbs, including replication verbs, do not specify
# "replication_server" (this is the default). To only handle replication,
# set to a True value (e.g. "True" or "1"). To handle only non-replication
# verbs, set to "False". Unless you have a separate replication network, you
# should not specify any value for "replication_server".
# replication_server = false
#
# Set to restrict the number of concurrent incoming REPLICATION requests
# Set to 0 for unlimited
# Note that REPLICATION is currently an ssync only item
# replication_concurrency = 4
#
# Restricts incoming REPLICATION requests to one per device,
# replication_currency above allowing. This can help control I/O to each
# device, but you may wish to set this to False to allow multiple REPLICATION
# requests (up to the above replication_concurrency setting) per device.
# replication_one_per_device = True
#
# Number of seconds to wait for an existing replication device lock before
# giving up.
# replication_lock_timeout = 15
#
# These next two settings control when the REPLICATION subrequest handler will
# abort an incoming REPLICATION attempt. An abort will occur if there are at
# least threshold number of failures and the value of failures / successes
# exceeds the ratio. The defaults of 100 and 1.0 means that at least 100
# failures have to occur and there have to be more failures than successes for
# an abort to occur.
# replication_failure_threshold = 100
# replication_failure_ratio = 1.0
```

```
[filter:healthcheck]
use = egg:swift#healthcheck
# An optional filesystem path, which if present, will cause the healthcheck
# URL to return "503 Service Unavailable" with a body of "DISABLED BY FILE"
# disable_path =

[filter:recon]
use = egg:swift#recon
#recon_cache_path = /var/cache/swift
#recon_lock_path = /var/lock

[object-replicator]
# You can override the default log routing for this app here (don't use set!):
# log_name = object-replicator
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# vm_test_mode = no
# daemonize = on
# run_pause = 30
# concurrency = 1
# stats_interval = 300
#
# The sync method to use; default is rsync but you can use ssync to try the
# EXPERIMENTAL all-swift-code-no-rsync-callouts method. Once ssync is verified
# as having performance comparable to, or better than, rsync, we plan to
# deprecate rsync so we can move on with more features for replication.
# sync_method = rsync
#
# max duration of a partition rsync
# rsync_timeout = 900
#
# bandwidth limit for rsync in kB/s. 0 means unlimited
# rsync_bwlimit = 0
#
# passed to rsync for io op timeout
# rsync_io_timeout = 30
#
# node_timeout = <whatever's in the DEFAULT section or 10>
# max duration of an http request; this is for REPLICATE finalization calls
 and
# so should be longer than node_timeout
# http_timeout = 60
#
# attempts to kill all workers if nothing replicates for lockup_timeout
 seconds
# lockup_timeout = 1800
#
# The replicator also performs reclamation
# reclaim_age = 604800
#
# ring_check_interval = 15
# recon_cache_path = /var/cache/swift
#
# limits how long rsync error log lines are
# 0 means to log the entire line
# rsync_error_log_line_length = 0
#
```

```
# handoffs_first and handoff_delete are options for a special case
# such as disk full in the cluster. These two options SHOULD NOT BE
# CHANGED, except for such an extreme situations. (e.g. disks filled up
# or are about to fill up. Anyway, DO NOT let your drives fill up)
# handoffs_first is the flag to replicate handoffs prior to canonical
# partitions. It allows to force syncing and deleting handoffs quickly.
# If set to a True value(e.g. "True" or "1"), partitions
# that are not supposed to be on the node will be replicated first.
# handoffs_first = False
#
# handoff_delete is the number of replicas which are ensured in swift.
# If the number less than the number of replicas is set, object-replicator
# could delete local handoffs even if all replicas are not ensured in the
# cluster. Object-replicator would remove local handoff partition directories
# after syncing partition when the number of successful responses is greater
# than or equal to this number. By default(auto), handoff partitions will be
# removed  when it has successfully replicated to all the canonical nodes.
# handoff_delete = auto

[object-updater]
# You can override the default log routing for this app here (don't use set!):
# log_name = object-updater
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# interval = 300
# concurrency = 1
# node_timeout = <whatever's in the DEFAULT section or 10>
# slowdown will sleep that amount between objects
# slowdown = 0.01
#
# recon_cache_path = /var/cache/swift

[object-auditor]
# You can override the default log routing for this app here (don't use set!):
# log_name = object-auditor
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# files_per_second = 20
# bytes_per_second = 10000000
# log_time = 3600
# zero_byte_files_per_second = 50
# recon_cache_path = /var/cache/swift

# Takes a comma separated list of ints. If set, the object auditor will
# increment a counter for every object whose size is <= to the given break
# points and report the result after a full scan.
# object_size_stats =
```

Container server configuration

Find an example container server configuration at `etc/container-server.conf-sample` in the source code repository.

The available configuration options are:

Table 6.9. Description of configuration options for [DEFAULT] in container-server.conf-sample

Configuration option=Default value	Description
bind_ip=0.0.0.0	IP Address for server to bind to
bind_port=6001	Port for server to bind to
bind_timeout=30	Seconds to attempt bind before giving up
backlog=4096	Maximum number of allowed pending TCP connections
user=swift	User to run as
swift_dir=/etc/swift	Swift configuration directory
devices=/srv/node	Parent directory of where devices are mounted
mount_check=true	Whether or not check if the devices are mounted to prevent accidentally writing to the root device
disable_fallocate=false	Disable "fast fail" fallocate checks if the underlying filesystem does not support it.
workers=auto	a much higher value, one can reduce the impact of slow file system operations in one request from negatively impacting other requests.
max_clients=1024	Maximum number of clients one worker can process simultaneously Lowering the number of clients handled per worker, and raising the number of workers can lessen the impact that a CPU intensive, or blocking, request can have on other requests served by the same worker. If the maximum number of clients is set to one, then a given worker will not perform another call while processing, allowing other workers a chance to process it.
allowed_sync_hosts=127.0.0.1	No help text available for this option
log_name=swift	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
log_custom_handlers=	Comma-separated list of functions to call to setup custom log handlers.
log_udp_host=	If not set, the UDB receiver for syslog is disabled.
log_udp_port=514	Port value for UDB receiver, if enabled.
log_statsd_host=localhost	If not set, the StatsD feature is disabled.
log_statsd_port=8125	Port value for the StatsD server.
log_statsd_default_sample_rate=1.0	Defines the probability of sending a sample for any given event or timing measurement.
log_statsd_sample_rate_factor=1.0	Not recommended to set this to a value less than 1.0, if frequency of logging is too high, tune the log_statsd_default_sample_rate instead.
log_statsd_metric_prefix=	Value will be prepended to every metric sent to the StatsD server.
db_preallocation=off	If you don't mind the extra disk space usage in overhead, you can turn this on to preallocate disk space with SQLite databases to decrease fragmentation. underlying filesystem does not support it. to setup custom log handlers. bytes you'd like fallocate to reserve, whether there is space for the given file size or not. This is useful for systems that behave badly when they completely run out of space; you can make the services pretend they're out of space early. server. For most cases, this should be

Configuration option=Default value	Description
	`egg:swift#account`. replication passes account can be reclaimed
eventlet_debug=false	If true, turn on debug logging for eventlet
fallocate_reserve=0	You can set fallocate_reserve to the number of bytes you'd like fallocate to reserve, whether there is space for the given file size or not. This is useful for systems that behave badly when they completely run out of space; you can make the services pretend they're out of space early. server. For most cases, this should be `egg:swift#object`.

Table 6.10. Description of configuration options for [app:container-server] in container-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#container	Entry point of paste.deploy in the server
set log_name=container-server	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_requests=true	Whether or not to log requests
set log_address=/dev/log	No help text available for this option
node_timeout=3	Request timeout to external services
conn_timeout=0.5	Connection timeout to external services
allow_versions=false	Enable/Disable object versioning feature
auto_create_account_prefix=.	Prefix to use when automatically creating accounts
replication_server=false	If defined, tells server how to handle replication verbs in requests. When set to True (or 1), only replication verbs will be accepted. When set to False, replication verbs will be rejected. When undefined, server will accept any verb in the request.

Table 6.11. Description of configuration options for [pipeline:main] in container-server.conf-sample

Configuration option=Default value	Description
pipeline=healthcheck recon container-server	No help text available for this option

Table 6.12. Description of configuration options for [container-replicator] in container-server.conf-sample

Configuration option=Default value	Description
log_name=container-replicator	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
vm_test_mode=no	Indicates that you are using a VM environment
per_diff=1000	Limit number of items to get per diff
max_diffs=100	Caps how long the replicator spends trying to sync a database per pass
concurrency=8	Number of replication workers to spawn
interval=30	Minimum time for a pass to take
node_timeout=10	Request timeout to external services

Configuration option=Default value	Description
conn_timeout=0.5	Connection timeout to external services
reclaim_age=604800	Time elapsed in seconds before an object can be reclaimed
run_pause=30	Time in seconds to wait between replication passes
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored

Table 6.13. Description of configuration options for [container-updater] in container-server.conf-sample

Configuration option=Default value	Description
log_name=container-updater	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
interval=300	Minimum time for a pass to take
concurrency=4	Number of replication workers to spawn
node_timeout=3	Request timeout to external services
conn_timeout=0.5	Connection timeout to external services
slowdown=0.01	Time in seconds to wait between objects
account_suppression_time=60	Seconds to suppress updating an account that has generated an error (timeout, not yet found, etc.)
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored

Table 6.14. Description of configuration options for [container-auditor] in container-server.conf-sample

Configuration option=Default value	Description
log_name=container-auditor	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
interval=1800	Minimum time for a pass to take
containers_per_second=200	Maximum containers audited per second. Should be tuned according to individual system specs. 0 is unlimited. mounted to prevent accidentally writing to the root device process simultaneously (it will actually accept(2) N + 1). Setting this to one (1) will only handle one request at a time, without accepting another request concurrently. By increasing the number of workers to a much higher value, one can reduce the impact of slow file system operations in one request from negatively impacting other requests.
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored

Table 6.15. Description of configuration options for [container-sync] in container-server.conf-sample

Configuration option=Default value	Description
log_name=container-sync	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to

Configuration option=Default value	Description
sync_proxy=http://127.0.0.1:8888	If you need to use an HTTP proxy, set it here. Defaults to no proxy.
interval=300	Minimum time for a pass to take
container_time=60	Maximum amount of time to spend syncing each container

Table 6.16. Description of configuration options for `[filter:healthcheck]` in `container-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#healthcheck	Entry point of paste.deploy in the server
disable_path=	No help text available for this option

Table 6.17. Description of configuration options for `[filter:recon]` in `container-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#recon	Entry point of paste.deploy in the server
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored

Sample container server configuration file

```
[DEFAULT]
# bind_ip = 0.0.0.0
# bind_port = 6001
# bind_timeout = 30
# backlog = 4096
# user = swift
# swift_dir = /etc/swift
# devices = /srv/node
# mount_check = true
# disable_fallocate = false
#
# Use an integer to override the number of pre-forked processes that will
# accept connections.
# workers = auto
#
# Maximum concurrent requests per worker
# max_clients = 1024
#
# This is a comma separated list of hosts allowed in the X-Container-Sync-To
# field for containers. This is the old-style of using container sync. It is
# strongly recommended to use the new style of a separate
# container-sync-realms.conf -- see container-sync-realms.conf-sample
# allowed_sync_hosts = 127.0.0.1
#
# You can specify default log routing here if you want:
# log_name = swift
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# comma separated list of functions to call to setup custom log handlers.
# functions get passed: conf, name, log_to_console, log_route, fmt, logger,
# adapted_logger
# log_custom_handlers =
```

```
#
# If set, log_udp_host will override log_address
# log_udp_host =
# log_udp_port = 514
#
# You can enable StatsD logging here:
# log_statsd_host = localhost
# log_statsd_port = 8125
# log_statsd_default_sample_rate = 1.0
# log_statsd_sample_rate_factor = 1.0
# log_statsd_metric_prefix =
#
# If you don't mind the extra disk space usage in overhead, you can turn this
# on to preallocate disk space with SQLite databases to decrease
 fragmentation.
# db_preallocation = off
#
# eventlet_debug = false
#
# You can set fallocate_reserve to the number of bytes you'd like fallocate to
# reserve, whether there is space for the given file size or not.
# fallocate_reserve = 0

[pipeline:main]
pipeline = healthcheck recon container-server

[app:container-server]
use = egg:swift#container
# You can override the default log routing for this app here:
# set log_name = container-server
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_requests = true
# set log_address = /dev/log
#
# node_timeout = 3
# conn_timeout = 0.5
# allow_versions = false
# auto_create_account_prefix = .
#
# Configure parameter for creating specific server
# To handle all verbs, including replication verbs, do not specify
# "replication_server" (this is the default). To only handle replication,
# set to a True value (e.g. "True" or "1"). To handle only non-replication
# verbs, set to "False". Unless you have a separate replication network, you
# should not specify any value for "replication_server".
# replication_server = false

[filter:healthcheck]
use = egg:swift#healthcheck
# An optional filesystem path, which if present, will cause the healthcheck
# URL to return "503 Service Unavailable" with a body of "DISABLED BY FILE"
# disable_path =

[filter:recon]
use = egg:swift#recon
#recon_cache_path = /var/cache/swift

[container-replicator]
# You can override the default log routing for this app here (don't use set!):
```

```
# log_name = container-replicator
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# vm_test_mode = no
# per_diff = 1000
# max_diffs = 100
# concurrency = 8
# interval = 30
# node_timeout = 10
# conn_timeout = 0.5
#
# The replicator also performs reclamation
# reclaim_age = 604800
#
# Time in seconds to wait between replication passes
# run_pause = 30
#
# recon_cache_path = /var/cache/swift

[container-updater]
# You can override the default log routing for this app here (don't use set!):
# log_name = container-updater
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# interval = 300
# concurrency = 4
# node_timeout = 3
# conn_timeout = 0.5
#
# slowdown will sleep that amount between containers
# slowdown = 0.01
#
# Seconds to suppress updating an account that has generated an error
# account_suppression_time = 60
#
# recon_cache_path = /var/cache/swift

[container-auditor]
# You can override the default log routing for this app here (don't use set!):
# log_name = container-auditor
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# Will audit each container at most once per interval
# interval = 1800
#
# containers_per_second = 200
# recon_cache_path = /var/cache/swift

[container-sync]
# You can override the default log routing for this app here (don't use set!):
# log_name = container-sync
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
```

```
#
# If you need to use an HTTP Proxy, set it here; defaults to no proxy.
# You can also set this to a comma separated list of HTTP Proxies and they
 will
# be randomly used (simple load balancing).
# sync_proxy = http://10.1.1.1:8888,http://10.1.1.2:8888
#
# Will sync each container at most once per interval
# interval = 300
#
# Maximum amount of time to spend syncing each container per pass
# container_time = 60
```

Account server configuration

Find an example account server configuration at `etc/account-server.conf-sample` in the source code repository.

The available configuration options are:

Table 6.18. Description of configuration options for `[DEFAULT]` in `account-server.conf-sample`

Configuration option=Default value	Description
bind_ip=0.0.0.0	IP Address for server to bind to
bind_port=6002	Port for server to bind to
bind_timeout=30	Seconds to attempt bind before giving up
backlog=4096	Maximum number of allowed pending TCP connections
user=swift	User to run as
swift_dir=/etc/swift	Swift configuration directory
devices=/srv/node	Parent directory of where devices are mounted
mount_check=true	Whether or not check if the devices are mounted to prevent accidentally writing to the root device
disable_fallocate=false	Disable "fast fail" fallocate checks if the underlying filesystem does not support it.
workers=auto	a much higher value, one can reduce the impact of slow file system operations in one request from negatively impacting other requests.
max_clients=1024	Maximum number of clients one worker can process simultaneously Lowering the number of clients handled per worker, and raising the number of workers can lessen the impact that a CPU intensive, or blocking, request can have on other requests served by the same worker. If the maximum number of clients is set to one, then a given worker will not perform another call while processing, allowing other workers a chance to process it.
log_name=swift	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
log_custom_handlers=	Comma-separated list of functions to call to setup custom log handlers.
log_udp_host=	If not set, the UDB receiver for syslog is disabled.

Configuration option=Default value	Description
log_udp_port=514	Port value for UDB receiver, if enabled.
log_statsd_host=localhost	If not set, the StatsD feature is disabled.
log_statsd_port=8125	Port value for the StatsD server.
log_statsd_default_sample_rate=1.0	Defines the probability of sending a sample for any given event or timing measurement.
log_statsd_sample_rate_factor=1.0	Not recommended to set this to a value less than 1.0, if frequency of logging is too high, tune the log_statsd_default_sample_rate instead.
log_statsd_metric_prefix=	Value will be prepended to every metric sent to the StatsD server.
db_preallocation=off	If you don't mind the extra disk space usage in overhead, you can turn this on to preallocate disk space with SQLite databases to decrease fragmentation. underlying filesystem does not support it. to setup custom log handlers. bytes you'd like fallocate to reserve, whether there is space for the given file size or not. This is useful for systems that behave badly when they completely run out of space; you can make the services pretend they're out of space early. server. For most cases, this should be `egg:swift#account`. replication passes account can be reclaimed
eventlet_debug=false	If true, turn on debug logging for eventlet
fallocate_reserve=0	You can set fallocate_reserve to the number of bytes you'd like fallocate to reserve, whether there is space for the given file size or not. This is useful for systems that behave badly when they completely run out of space; you can make the services pretend they're out of space early. server. For most cases, this should be `egg:swift#object`.

Table 6.19. Description of configuration options for [app:account-server] in account-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#account	Entry point of paste.deploy in the server
set log_name=account-server	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_requests=true	Whether or not to log requests
set log_address=/dev/log	No help text available for this option
auto_create_account_prefix=.	Prefix to use when automatically creating accounts
replication_server=false	If defined, tells server how to handle replication verbs in requests. When set to True (or 1), only replication verbs will be accepted. When set to False, replication verbs will be rejected. When undefined, server will accept any verb in the request.

Table 6.20. Description of configuration options for [pipeline:main] in account-server.conf-sample

Configuration option=Default value	Description
pipeline=healthcheck recon account-server	No help text available for this option

Table 6.21. Description of configuration options for [account-replicator] in account-server.conf-sample

Configuration option=Default value	Description
log_name=account-replicator	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
vm_test_mode=no	Indicates that you are using a VM environment
per_diff=1000	Limit number of items to get per diff
max_diffs=100	Caps how long the replicator spends trying to sync a database per pass
concurrency=8	Number of replication workers to spawn
interval=30	Minimum time for a pass to take
error_suppression_interval=60	Time in seconds that must elapse since the last error for a node to be considered no longer error limited
error_suppression_limit=10	Error count to consider a node error limited
node_timeout=10	Request timeout to external services
conn_timeout=0.5	Connection timeout to external services
reclaim_age=604800	Time elapsed in seconds before an object can be reclaimed
run_pause=30	Time in seconds to wait between replication passes
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored

Table 6.22. Description of configuration options for [account-auditor] in account-server.conf-sample

Configuration option=Default value	Description
log_name=account-auditor	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
interval=1800	Minimum time for a pass to take
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
accounts_per_second=200	Maximum accounts audited per second. Should be tuned according to individual system specs. 0 is unlimited.
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored

Table 6.23. Description of configuration options for [account-reaper] in account-server.conf-sample

Configuration option=Default value	Description
log_name=account-reaper	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
concurrency=25	Number of replication workers to spawn
interval=3600	Minimum time for a pass to take
node_timeout=10	Request timeout to external services

Configuration option=Default value	Description
conn_timeout=0.5	Connection timeout to external services
delay_reaping=0	Normally, the reaper begins deleting account information for deleted accounts immediately; you can set this to delay its work however. The value is in seconds, 2592000 = 30 days, for example. bind to giving up worker can process simultaneously (it will actually accept(2) N + 1). Setting this to one (1) will only handle one request at a time, without accepting another request concurrently. By increasing the number of workers to a much higher value, one can reduce the impact of slow file system operations in one request from negatively impacting other requests.
reap_warn_after=2592000	No help text available for this option

Table 6.24. Description of configuration options for `[filter:healthcheck]` in `account-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#healthcheck	Entry point of paste.deploy in the server
disable_path=	No help text available for this option

Table 6.25. Description of configuration options for `[filter:recon]` in `account-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#recon	Entry point of paste.deploy in the server
recon_cache_path=/var/cache/swift	Directory where stats for a few items will be stored

Sample account server configuration file

```
[DEFAULT]
# bind_ip = 0.0.0.0
# bind_port = 6002
# bind_timeout = 30
# backlog = 4096
# user = swift
# swift_dir = /etc/swift
# devices = /srv/node
# mount_check = true
# disable_fallocate = false
#
# Use an integer to override the number of pre-forked processes that will
# accept connections.
# workers = auto
#
# Maximum concurrent requests per worker
# max_clients = 1024
#
# You can specify default log routing here if you want:
# log_name = swift
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# comma separated list of functions to call to setup custom log handlers.
# functions get passed: conf, name, log_to_console, log_route, fmt, logger,
# adapted_logger
```

```
# log_custom_handlers =
#
# If set, log_udp_host will override log_address
# log_udp_host =
# log_udp_port = 514
#
# You can enable StatsD logging here:
# log_statsd_host = localhost
# log_statsd_port = 8125
# log_statsd_default_sample_rate = 1.0
# log_statsd_sample_rate_factor = 1.0
# log_statsd_metric_prefix =
#
# If you don't mind the extra disk space usage in overhead, you can turn this
# on to preallocate disk space with SQLite databases to decrease
 fragmentation.
# db_preallocation = off
#
# eventlet_debug = false
#
# You can set fallocate_reserve to the number of bytes you'd like fallocate to
# reserve, whether there is space for the given file size or not.
# fallocate_reserve = 0

[pipeline:main]
pipeline = healthcheck recon account-server

[app:account-server]
use = egg:swift#account
# You can override the default log routing for this app here:
# set log_name = account-server
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_requests = true
# set log_address = /dev/log
#
# auto_create_account_prefix = .
#
# Configure parameter for creating specific server
# To handle all verbs, including replication verbs, do not specify
# "replication_server" (this is the default). To only handle replication,
# set to a True value (e.g. "True" or "1"). To handle only non-replication
# verbs, set to "False". Unless you have a separate replication network, you
# should not specify any value for "replication_server".
# replication_server = false

[filter:healthcheck]
use = egg:swift#healthcheck
# An optional filesystem path, which if present, will cause the healthcheck
# URL to return "503 Service Unavailable" with a body of "DISABLED BY FILE"
# disable_path =

[filter:recon]
use = egg:swift#recon
# recon_cache_path = /var/cache/swift

[account-replicator]
# You can override the default log routing for this app here (don't use set!):
# log_name = account-replicator
# log_facility = LOG_LOCAL0
```

```
# log_level = INFO
# log_address = /dev/log
#
# vm_test_mode = no
# per_diff = 1000
# max_diffs = 100
# concurrency = 8
# interval = 30
#
# How long without an error before a node's error count is reset. This will
# also be how long before a node is reenabled after suppression is triggered.
# error_suppression_interval = 60
#
# How many errors can accumulate before a node is temporarily ignored.
# error_suppression_limit = 10
#
# node_timeout = 10
# conn_timeout = 0.5
#
# The replicator also performs reclamation
# reclaim_age = 604800
#
# Time in seconds to wait between replication passes
# run_pause = 30
#
# recon_cache_path = /var/cache/swift

[account-auditor]
# You can override the default log routing for this app here (don't use set!):
# log_name = account-auditor
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# Will audit each account at most once per interval
# interval = 1800
#
# log_facility = LOG_LOCAL0
# log_level = INFO
# accounts_per_second = 200
# recon_cache_path = /var/cache/swift

[account-reaper]
# You can override the default log routing for this app here (don't use set!):
# log_name = account-reaper
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_address = /dev/log
#
# concurrency = 25
# interval = 3600
# node_timeout = 10
# conn_timeout = 0.5
#
# Normally, the reaper begins deleting account information for deleted
 accounts
# immediately; you can set this to delay its work however. The value is in
# seconds; 2592000 = 30 days for example.
# delay_reaping = 0
#
```

```
# If the account fails to be be reaped due to a persistent error, the
# account reaper will log a message such as:
#     Account <name> has not been reaped since <date>
# You can search logs for this message if space is not being reclaimed
# after you delete account(s).
# Default is 2592000 seconds (30 days). This is in addition to any time
# requested by delay_reaping.
# reap_warn_after = 2592000
```

Proxy server configuration

Find an example proxy server configuration at `etc/proxy-server.conf-sample` in the source code repository.

The available configuration options are:

Table 6.26. Description of configuration options for [DEFAULT] in proxy-server.conf-sample

Configuration option=Default value	Description
bind_ip=0.0.0.0	IP Address for server to bind to
bind_port=80	Port for server to bind to
bind_timeout=30	Seconds to attempt bind before giving up
backlog=4096	Maximum number of allowed pending TCP connections
swift_dir=/etc/swift	Swift configuration directory
user=swift	User to run as
workers=auto	a much higher value, one can reduce the impact of slow file system operations in one request from negatively impacting other requests.
max_clients=1024	Maximum number of clients one worker can process simultaneously Lowering the number of clients handled per worker, and raising the number of workers can lessen the impact that a CPU intensive, or blocking, request can have on other requests served by the same worker. If the maximum number of clients is set to one, then a given worker will not perform another call while processing, allowing other workers a chance to process it.
cert_file=/etc/swift/proxy.crt	to the ssl .crt. This should be enabled for testing purposes only.
key_file=/etc/swift/proxy.key	to the ssl .key. This should be enabled for testing purposes only.
expiring_objects_container_divisor=86400	No help text available for this option
log_name=swift	Label used when logging
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_headers=false	No help text available for this option
log_address=/dev/log	Location where syslog sends the logs to
trans_id_suffix=	No help text available for this option
log_custom_handlers=	Comma-separated list of functions to call to setup custom log handlers.
log_udp_host=	If not set, the UDB receiver for syslog is disabled.
log_udp_port=514	Port value for UDB receiver, if enabled.
log_statsd_host=localhost	If not set, the StatsD feature is disabled.

Configuration option=Default value	Description
log_statsd_port=8125	Port value for the StatsD server.
log_statsd_default_sample_rate=1.0	Defines the probability of sending a sample for any given event or timing measurement.
log_statsd_sample_rate_factor=1.0	Not recommended to set this to a value less than 1.0, if frequency of logging is too high, tune the log_statsd_default_sample_rate instead.
log_statsd_metric_prefix=	Value will be prepended to every metric sent to the StatsD server.
cors_allow_origin=	is a list of hosts that are included with any CORS request by default and returned with the Access-Control-Allow-Origin header in addition to what the container has set. to call to setup custom log handlers. for eventlet the proxy server. For most cases, this should be `egg:swift#proxy`. request whenever it has to failover to a handoff node
client_timeout=60	Timeout to read one chunk from a client external services
eventlet_debug=false	If true, turn on debug logging for eventlet

Table 6.27. Description of configuration options for [app:proxy-server] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#proxy	Entry point of paste.deploy in the server
set log_name=proxy-server	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_address=/dev/log	No help text available for this option
log_handoffs=true	No help text available for this option
recheck_account_existence=60	Cache timeout in seconds to send memcached for account existence
recheck_container_existence=60	Cache timeout in seconds to send memcached for container existence
object_chunk_size=8192	Chunk size to read from object servers
client_chunk_size=8192	Chunk size to read from clients
node_timeout=10	Request timeout to external services
conn_timeout=0.5	Connection timeout to external services
error_suppression_interval=60	Time in seconds that must elapse since the last error for a node to be considered no longer error limited
error_suppression_limit=10	Error count to consider a node error limited
allow_account_management=false	Whether account PUTs and DELETEs are even callable
object_post_as_copy=true	Set object_post_as_copy = false to turn on fast posts where only the metadata changes are stored anew and the original data file is kept in place. This makes for quicker posts; but since the container metadata isn't updated in this mode, features like container sync won't be able to sync posts.
account_autocreate=false	If set to 'true' authorized accounts that do not yet exist within the Swift cluster will be automatically created.
max_containers_per_account=0	If set to a positive value, trying to create a container when the account already has at least this maximum containers will result in a 403 Forbidden. Note: This is a soft limit, meaning a user might exceed the cap for recheck_account_existence before the 403s kick in.

Configuration option=Default value	Description
max_containers_whitelist=	is a comma separated list of account names that ignore the max_containers_per_account cap.
deny_host_headers=	No help text available for this option
auto_create_account_prefix=.	Prefix to use when automatically creating accounts
put_queue_depth=10	No help text available for this option
rate_limit_after_segment=10	Rate limit the download of large object segments after this segment is downloaded.
rate_limit_segments_per_sec=1	Rate limit large object downloads at this rate. contact for a normal request. You can use '* replicas' at the end to have it use the number given times the number of replicas for the ring being used for the request. paste.deploy to use for auth. To use tempauth set to: `egg:swift#tempauth` each request
sorting_method=shuffle	No help text available for this option
timing_expiry=300	No help text available for this option
allow_static_large_object=true	No help text available for this option
max_large_object_get_time=86400	No help text available for this option
request_node_count=2 * replicas	* replicas Set to the number of nodes to contact for a normal request. You can use '* replicas' at the end to have it use the number given times the number of replicas for the ring being used for the request. conf file for values will only be shown to the list of swift_owners. The exact default definition of a swift_owner is headers> up to the auth system in use, but usually indicates administrative responsibilities. paste.deploy to use for auth. To use tempauth set to: `egg:swift#tempauth` each request
read_affinity=r1z1=100, r1z2=200, r2=300	No help text available for this option
read_affinity=	No help text available for this option
write_affinity=r1, r2	No help text available for this option
write_affinity=	No help text available for this option
write_affinity_node_count=2 * replicas	No help text available for this option
swift_owner_headers=x-container-read, x-container-write, x-container-sync-key, x-container-sync-to, x-account-meta-temp-url-key, x-account-meta-temp-url-key-2	the sample These are the headers whose conf file for values will only be shown to the list of swift_owners. The exact default definition of a swift_owner is headers> up to the auth system in use, but usually indicates administrative responsibilities. paste.deploy to use for auth. To use tempauth set to: `egg:swift#tempauth` each request

Table 6.28. Description of configuration options for `[pipeline:main]` in `proxy-server.conf-sample`

Configuration option=Default value	Description
pipeline=catch_errors healthcheck proxy-logging cache bulk slo ratelimit tempauth container-quotas account-quotas proxy-logging proxy-server	No help text available for this option

Table 6.29. Description of configuration options for `[filter:account-quotas]` in `proxy-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#account_quotas	Entry point of paste.deploy in the server

Table 6.30. Description of configuration options for [filter:authtoken] in proxy-server.conf-sample

Configuration option=Default value	Description
auth_host=keystonehost	No help text available for this option
auth_port=35357	No help text available for this option
auth_protocol=http	No help text available for this option
auth_uri=http://keystonehost:5000/	No help text available for this option
admin_tenant_name=service	No help text available for this option
admin_user=swift	No help text available for this option
admin_password=password	No help text available for this option
delay_auth_decision=1	No help text available for this option
cache=swift.cache	No help text available for this option

Table 6.31. Description of configuration options for [filter:cache] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#memcache	Entry point of paste.deploy in the server
set log_name=cache	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_headers=false	If True, log headers in each request
set log_address=/dev/log	No help text available for this option
memcache_servers=127.0.0.1:11211	Comma separated list of memcached servers ip:port services
memcache_serialization_support=2	No help text available for this option

Table 6.32. Description of configuration options for [filter:catch_errors] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#catch_errors	Entry point of paste.deploy in the server
set log_name=catch_errors	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_headers=false	If True, log headers in each request
set log_address=/dev/log	No help text available for this option

Table 6.33. Description of configuration options for [filter:healthcheck] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#healthcheck	Entry point of paste.deploy in the server
disable_path=	No help text available for this option

Table 6.34. Description of configuration options for [filter:keystoneauth] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#keystoneauth	Entry point of paste.deploy in the server
operator_roles=admin, swiftoperator	No help text available for this option

Table 6.35. Description of configuration options for [filter:list-endpoints] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#list_endpoints	Entry point of paste.deploy in the server
list_endpoints_path=/endpoints/	No help text available for this option

Table 6.36. Description of configuration options for [filter:proxy-logging] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#proxy_logging	Entry point of paste.deploy in the server
access_log_name=swift	No help text available for this option
access_log_facility=LOG_LOCAL0	No help text available for this option
access_log_level=INFO	No help text available for this option
access_log_address=/dev/log	No help text available for this option
access_log_udp_host=	No help text available for this option
access_log_udp_port=514	No help text available for this option
access_log_statsd_host=localhost	No help text available for this option
access_log_statsd_port=8125	No help text available for this option
access_log_statsd_default_sample_rate=1.0	No help text available for this option
access_log_statsd_sample_rate_factor=1.0	No help text available for this option
access_log_statsd_metric_prefix=	No help text available for this option
access_log_headers=false	No help text available for this option
logged with access_log_headers=True.	No help text available for this option
reveal_sensitive_prefix=8192	The X-Auth-Token is sensitive data. If revealed to an unauthorised person, they can now make requests against an account until the token expires. Set reveal_sensitive_prefix to the number of characters of the token that are logged. For example reveal_sensitive_prefix=12 so only first 12 characters of the token are logged. Or, set to 0 to completely remove the token.
log_statsd_valid_http_methods=GET,HEAD,POST,PUT,DELETE,COPY,OPTIONS	No help text available for this option

Table 6.37. Description of configuration options for [filter:tempauth] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#tempauth	Entry point of paste.deploy in the server
set log_name=tempauth	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_headers=false	If True, log headers in each request

Configuration option=Default value	Description
set log_address=/dev/log	No help text available for this option
reseller_prefix=AUTH	The naming scope for the auth service. Swift
auth_prefix=/auth/	The HTTP request path prefix for the auth service. Swift itself reserves anything beginning with the letter `v`.
token_life=86400	The number of seconds a token is valid.
allow_overrides=true	No help text available for this option
storage_url_scheme=default	Scheme to return with storage urls: http, https, or default (chooses based on what the server is running as) This can be useful with an SSL load balancer in front of a non-SSL server.
user_admin_admin=admin .admin .reseller_admin	No help text available for this option
user_test_tester=testing .admin	No help text available for this option
user_test2_tester2=testing2 .admin	No help text available for this option
user_test_tester3=testing3	No help text available for this option

Sample proxy server configuration file

```
[DEFAULT]
# bind_ip = 0.0.0.0
# bind_port = 80
# bind_timeout = 30
# backlog = 4096
# swift_dir = /etc/swift
# user = swift

# Enables exposing configuration settings via HTTP GET /info.
# expose_info = true

# Key to use for admin calls that are HMAC signed.  Default is empty,
# which will disable admin calls to /info.
# admin_key = secret_admin_key
#
# Allows the ability to withhold sections from showing up in the public
# calls to /info.  The following would cause the sections 'container_quotas'
# and 'tempurl' to not be listed.  Default is empty, allowing all registered
# fetures to be listed via HTTP GET /info.
# disallowed_sections = container_quotas, tempurl

# Use an integer to override the number of pre-forked processes that will
# accept connections.  Should default to the number of effective cpu
# cores in the system.  It's worth noting that individual workers will
# use many eventlet co-routines to service multiple concurrent requests.
# workers = auto
#
# Maximum concurrent requests per worker
# max_clients = 1024
#
# Set the following two lines to enable SSL. This is for testing only.
# cert_file = /etc/swift/proxy.crt
# key_file = /etc/swift/proxy.key
#
# expiring_objects_container_divisor = 86400
# expiring_objects_account_name = expiring_objects
#
# You can specify default log routing here if you want:
# log_name = swift
```

```
# log_facility = LOG_LOCAL0
# log_level = INFO
# log_headers = false
# log_address = /dev/log
#
# This optional suffix (default is empty) that would be appended to the swift
 transaction
# id allows one to easily figure out from which cluster that X-Trans-Id
 belongs to.
# This is very useful when one is managing more than one swift cluster.
# trans_id_suffix =
#
# comma separated list of functions to call to setup custom log handlers.
# functions get passed: conf, name, log_to_console, log_route, fmt, logger,
# adapted_logger
# log_custom_handlers =
#
# If set, log_udp_host will override log_address
# log_udp_host =
# log_udp_port = 514
#
# You can enable StatsD logging here:
# log_statsd_host = localhost
# log_statsd_port = 8125
# log_statsd_default_sample_rate = 1.0
# log_statsd_sample_rate_factor = 1.0
# log_statsd_metric_prefix =
#
# Use a comma separated list of full url (http://foo.bar:1234,https://foo.bar)
# cors_allow_origin =
#
# client_timeout = 60
# eventlet_debug = false

[pipeline:main]
pipeline = catch_errors gatekeeper healthcheck proxy-logging cache
 container_sync bulk tempurl slo dlo ratelimit tempauth container-quotas
 account-quotas proxy-logging proxy-server

[app:proxy-server]
use = egg:swift#proxy
# You can override the default log routing for this app here:
# set log_name = proxy-server
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_address = /dev/log
#
# log_handoffs = true
# recheck_account_existence = 60
# recheck_container_existence = 60
# object_chunk_size = 8192
# client_chunk_size = 8192
#
# How long the proxy server will wait on responses from the a/c/o servers.
# node_timeout = 10
#
# How long the proxy server will wait for an initial response and to read a
# chunk of data from the object servers while serving GET / HEAD requests.
# Timeouts from these requests can be recovered from so setting this to
# something lower than node_timeout would provide quicker error recovery
```

```
# while allowing for a longer timeout for non-recoverable requests (PUTs).
# Defaults to node_timeout, should be overriden if node_timeout is set to a
# high number to prevent client timeouts from firing before the proxy server
# has a chance to retry.
# recoverable_node_timeout = node_timeout
#
# conn_timeout = 0.5
#
# How long to wait for requests to finish after a quorum has been established.
# post_quorum_timeout = 0.5
#
# How long without an error before a node's error count is reset. This will
# also be how long before a node is reenabled after suppression is triggered.
# error_suppression_interval = 60
#
# How many errors can accumulate before a node is temporarily ignored.
# error_suppression_limit = 10
#
# If set to 'true' any authorized user may create and delete accounts; if
# 'false' no one, even authorized, can.
# allow_account_management = false
#
# Set object_post_as_copy = false to turn on fast posts where only the
 metadata
# changes are stored anew and the original data file is kept in place. This
# makes for quicker posts; but since the container metadata isn't updated in
# this mode, features like container sync won't be able to sync posts.
# object_post_as_copy = true
#
# If set to 'true' authorized accounts that do not yet exist within the Swift
# cluster will be automatically created.
# account_autocreate = false
#
# If set to a positive value, trying to create a container when the account
# already has at least this maximum containers will result in a 403 Forbidden.
# Note: This is a soft limit, meaning a user might exceed the cap for
# recheck_account_existence before the 403s kick in.
# max_containers_per_account = 0
#
# This is a comma separated list of account hashes that ignore the
# max_containers_per_account cap.
# max_containers_whitelist =
#
# Comma separated list of Host headers to which the proxy will deny requests.
# deny_host_headers =
#
# Prefix used when automatically creating accounts.
# auto_create_account_prefix = .
#
# Depth of the proxy put queue.
# put_queue_depth = 10
#
# Storage nodes can be chosen at random (shuffle), by using timing
# measurements (timing), or by using an explicit match (affinity).
# Using timing measurements may allow for lower overall latency, while
# using affinity allows for finer control. In both the timing and
# affinity cases, equally-sorting nodes are still randomly chosen to
# spread load.
# The valid values for sorting_method are "affinity", "shuffle", and "timing".
# sorting_method = shuffle
```

```
#
# If the "timing" sorting_method is used, the timings will only be valid for
# the number of seconds configured by timing_expiry.
# timing_expiry = 300
#
# The maximum time (seconds) that a large object connection is allowed to
 last.
# max_large_object_get_time = 86400
#
# Set to the number of nodes to contact for a normal request. You can use
# '* replicas' at the end to have it use the number given times the number of
# replicas for the ring being used for the request.
# request_node_count = 2 * replicas
#
# Which backend servers to prefer on reads. Format is r<N> for region
# N or r<N>z<M> for region N, zone M. The value after the equals is
# the priority; lower numbers are higher priority.
#
# Example: first read from region 1 zone 1, then region 1 zone 2, then
# anything in region 2, then everything else:
# read_affinity = r1z1=100, r1z2=200, r2=300
# Default is empty, meaning no preference.
# read_affinity =
#
# Which backend servers to prefer on writes. Format is r<N> for region
# N or r<N>z<M> for region N, zone M. If this is set, then when
# handling an object PUT request, some number (see setting
# write_affinity_node_count) of local backend servers will be tried
# before any nonlocal ones.
#
# Example: try to write to regions 1 and 2 before writing to any other
# nodes:
# write_affinity = r1, r2
# Default is empty, meaning no preference.
# write_affinity =
#
# The number of local (as governed by the write_affinity setting)
# nodes to attempt to contact first, before any non-local ones. You
# can use '* replicas' at the end to have it use the number given
# times the number of replicas for the ring being used for the
# request.
# write_affinity_node_count = 2 * replicas
#
# These are the headers whose values will only be shown to swift_owners. The
# exact definition of a swift_owner is up to the auth system in use, but
# usually indicates administrative responsibilities.
# swift_owner_headers = x-container-read, x-container-write, x-container-sync-
key, x-container-sync-to, x-account-meta-temp-url-key, x-account-meta-temp-
url-key-2, x-account-access-control


[filter:tempauth]
use = egg:swift#tempauth
# You can override the default log routing for this filter here:
# set log_name = tempauth
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_headers = false
# set log_address = /dev/log
#
```

```
# The reseller prefix will verify a token begins with this prefix before even
# attempting to validate it. Also, with authorization, only Swift storage
# accounts with this prefix will be authorized by this middleware. Useful if
# multiple auth systems are in use for one Swift cluster.
# reseller_prefix = AUTH
#
# The auth prefix will cause requests beginning with this prefix to be routed
# to the auth subsystem, for granting tokens, etc.
# auth_prefix = /auth/
# token_life = 86400
#
# This allows middleware higher in the WSGI pipeline to override auth
# processing, useful for middleware such as tempurl and formpost. If you know
# you're not going to use such middleware and you want a bit of extra
 security,
# you can set this to false.
# allow_overrides = true
#
# This specifies what scheme to return with storage urls:
# http, https, or default (chooses based on what the server is running as)
# This can be useful with an SSL load balancer in front of a non-SSL server.
# storage_url_scheme = default
#
# Lastly, you need to list all the accounts/users you want here. The format
 is:
#   user_<account>_<user> = <key> [group] [group] [...] [storage_url]
# or if you want underscores in <account> or <user>, you can base64 encode
 them
# (with no equal signs) and use this format:
#   user64_<account_b64>_<user_b64> = <key> [group] [group] [...]
 [storage_url]
# There are special groups of:
#   .reseller_admin = can do anything to any account for this auth
#   .admin = can do anything within the account
# If neither of these groups are specified, the user can only access
 containers
# that have been explicitly allowed for them by a .admin or .reseller_admin.
# The trailing optional storage_url allows you to specify an alternate url to
# hand back to the user upon authentication. If not specified, this defaults
 to
# $HOST/v1/<reseller_prefix>_<account> where $HOST will do its best to resolve
# to what the requester would need to use to reach this host.
# Here are example entries, required for running the tests:
user_admin_admin = admin .admin .reseller_admin
user_test_tester = testing .admin
user_test2_tester2 = testing2 .admin
user_test_tester3 = testing3

# To enable Keystone authentication you need to have the auth token
# middleware first to be configured. Here is an example below, please
# refer to the keystone's documentation for details about the
# different settings.
#
# You'll need to have as well the keystoneauth middleware enabled
# and have it in your main pipeline so instead of having tempauth in
# there you can change it to: authtoken keystoneauth
#
# [filter:authtoken]
# paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory
# auth_host = keystonehost
```

```
# auth_port = 35357
# auth_protocol = http
# auth_uri = http://keystonehost:5000/
# admin_tenant_name = service
# admin_user = swift
# admin_password = password
# delay_auth_decision = 1
# cache = swift.cache
# include_service_catalog = False
#
# [filter:keystoneauth]
# use = egg:swift#keystoneauth
# Operator roles is the role which user would be allowed to manage a
# tenant and be able to create container or give ACL to others.
# operator_roles = admin, swiftoperator
# The reseller admin role has the ability to create and delete accounts
# reseller_admin_role = ResellerAdmin

[filter:healthcheck]
use = egg:swift#healthcheck
# An optional filesystem path, which if present, will cause the healthcheck
# URL to return "503 Service Unavailable" with a body of "DISABLED BY FILE".
# This facility may be used to temporarily remove a Swift node from a load
# balancer pool during maintenance or upgrade (remove the file to allow the
# node back into the load balancer pool).
# disable_path =

[filter:cache]
use = egg:swift#memcache
# You can override the default log routing for this filter here:
# set log_name = cache
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_headers = false
# set log_address = /dev/log
#
# If not set here, the value for memcache_servers will be read from
# memcache.conf (see memcache.conf-sample) or lacking that file, it will
# default to the value below. You can specify multiple servers separated with
# commas, as in: 10.1.2.3:11211,10.1.2.4:11211
# memcache_servers = 127.0.0.1:11211
#
# Sets how memcache values are serialized and deserialized:
# 0 = older, insecure pickle serialization
# 1 = json serialization but pickles can still be read (still insecure)
# 2 = json serialization only (secure and the default)
# If not set here, the value for memcache_serialization_support will be read
# from /etc/swift/memcache.conf (see memcache.conf-sample).
# To avoid an instant full cache flush, existing installations should
# upgrade with 0, then set to 1 and reload, then after some time (24 hours)
# set to 2 and reload.
# In the future, the ability to use pickle serialization will be removed.
# memcache_serialization_support = 2
#
# Sets the maximum number of connections to each memcached server per worker
# memcache_max_connections = 2

[filter:ratelimit]
use = egg:swift#ratelimit
# You can override the default log routing for this filter here:
```

```
# set log_name = ratelimit
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_headers = false
# set log_address = /dev/log
#
# clock_accuracy should represent how accurate the proxy servers' system
 clocks
# are with each other. 1000 means that all the proxies' clock are accurate to
# each other within 1 millisecond.  No ratelimit should be higher than the
# clock accuracy.
# clock_accuracy = 1000
#
# max_sleep_time_seconds = 60
#
# log_sleep_time_seconds of 0 means disabled
# log_sleep_time_seconds = 0
#
# allows for slow rates (e.g. running up to 5 sec's behind) to catch up.
# rate_buffer_seconds = 5
#
# account_ratelimit of 0 means disabled
# account_ratelimit = 0

# these are comma separated lists of account names
# account_whitelist = a,b
# account_blacklist = c,d

# with container_limit_x = r
# for containers of size x limit write requests per second to r.  The
 container
# rate will be linearly interpolated from the values given. With the values
# below, a container of size 5 will get a rate of 75.
# container_ratelimit_0 = 100
# container_ratelimit_10 = 50
# container_ratelimit_50 = 20

# Similarly to the above container-level write limits, the following will
 limit
# container GET (listing) requests.
# container_listing_ratelimit_0 = 100
# container_listing_ratelimit_10 = 50
# container_listing_ratelimit_50 = 20

[filter:domain_remap]
use = egg:swift#domain_remap
# You can override the default log routing for this filter here:
# set log_name = domain_remap
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_headers = false
# set log_address = /dev/log
#
# storage_domain = example.com
# path_root = v1
# reseller_prefixes = AUTH

[filter:catch_errors]
use = egg:swift#catch_errors
# You can override the default log routing for this filter here:
```

```
# set log_name = catch_errors
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_headers = false
# set log_address = /dev/log

[filter:cname_lookup]
# Note: this middleware requires python-dnspython
use = egg:swift#cname_lookup
# You can override the default log routing for this filter here:
# set log_name = cname_lookup
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_headers = false
# set log_address = /dev/log
#
# Specify the storage_domain that match your cloud, multiple domains
# can be specified separated by a comma
# storage_domain = example.com
#
# lookup_depth = 1

# Note: Put staticweb just after your auth filter(s) in the pipeline
[filter:staticweb]
use = egg:swift#staticweb

# Note: Put tempurl before dlo, slo and your auth filter(s) in the pipeline
[filter:tempurl]
use = egg:swift#tempurl
# The methods allowed with Temp URLs.
# methods = GET HEAD PUT
#
# The headers to remove from incoming requests. Simply a whitespace delimited
# list of header names and names can optionally end with '*' to indicate a
# prefix match. incoming_allow_headers is a list of exceptions to these
# removals.
# incoming_remove_headers = x-timestamp
#
# The headers allowed as exceptions to incoming_remove_headers. Simply a
# whitespace delimited list of header names and names can optionally end with
# '*' to indicate a prefix match.
# incoming_allow_headers =
#
# The headers to remove from outgoing responses. Simply a whitespace delimited
# list of header names and names can optionally end with '*' to indicate a
# prefix match. outgoing_allow_headers is a list of exceptions to these
# removals.
# outgoing_remove_headers = x-object-meta-*
#
# The headers allowed as exceptions to outgoing_remove_headers. Simply a
# whitespace delimited list of header names and names can optionally end with
# '*' to indicate a prefix match.
# outgoing_allow_headers = x-object-meta-public-*

# Note: Put formpost just before your auth filter(s) in the pipeline
[filter:formpost]
use = egg:swift#formpost

# Note: Just needs to be placed before the proxy-server in the pipeline.
[filter:name_check]
```

```
use = egg:swift#name_check
# forbidden_chars = '"`<>
# maximum_length = 255
# forbidden_regexp = /\./|/\.\./|/\.$|/\.\.$

[filter:list-endpoints]
use = egg:swift#list_endpoints
# list_endpoints_path = /endpoints/

[filter:proxy-logging]
use = egg:swift#proxy_logging
# If not set, logging directives from [DEFAULT] without "access_" will be used
# access_log_name = swift
# access_log_facility = LOG_LOCAL0
# access_log_level = INFO
# access_log_address = /dev/log
#
# If set, access_log_udp_host will override access_log_address
# access_log_udp_host =
# access_log_udp_port = 514
#
# You can use log_statsd_* from [DEFAULT] or override them here:
# access_log_statsd_host = localhost
# access_log_statsd_port = 8125
# access_log_statsd_default_sample_rate = 1.0
# access_log_statsd_sample_rate_factor = 1.0
# access_log_statsd_metric_prefix =
# access_log_headers = false
#
# If access_log_headers is True and access_log_headers_only is set only
# these headers are logged. Multiple headers can be defined as comma separated
# list like this: access_log_headers_only = Host, X-Object-Meta-Mtime
# access_log_headers_only =
#
# By default, the X-Auth-Token is logged. To obscure the value,
# set reveal_sensitive_prefix to the number of characters to log.
# For example, if set to 12, only the first 12 characters of the
# token appear in the log. An unauthorized access of the log file
# won't allow unauthorized usage of the token. However, the first
# 12 or so characters is unique enough that you can trace/debug
# token usage. Set to 0 to suppress the token completely (replaced
# by '...' in the log).
# Note: reveal_sensitive_prefix will not affect the value
# logged with access_log_headers=True.
# reveal_sensitive_prefix = 8192
#
# What HTTP methods are allowed for StatsD logging (comma-sep); request
 methods
# not in this list will have "BAD_METHOD" for the <verb> portion of the
 metric.
# log_statsd_valid_http_methods = GET,HEAD,POST,PUT,DELETE,COPY,OPTIONS
#
# Note: The double proxy-logging in the pipeline is not a mistake. The
# left-most proxy-logging is there to log requests that were handled in
# middleware and never made it through to the right-most middleware (and
# proxy server). Double logging is prevented for normal requests. See
# proxy-logging docs.

# Note: Put before both ratelimit and auth in the pipeline.
[filter:bulk]
```

```
use = egg:swift#bulk
# max_containers_per_extraction = 10000
# max_failed_extractions = 1000
# max_deletes_per_request = 10000
# max_failed_deletes = 1000

# In order to keep a connection active during a potentially long bulk request,
# Swift may return whitespace prepended to the actual response body. This
# whitespace will be yielded no more than every yield_frequency seconds.
# yield_frequency = 10

# Note: The following parameter is used during a bulk delete of objects and
# their container. This would frequently fail because it is very likely
# that all replicated objects have not been deleted by the time the middleware
 got a
# successful response. It can be configured the number of retries. And the
# number of seconds to wait between each retry will be 1.5**retry

# delete_container_retry_count = 0

# Note: Put after auth in the pipeline.
[filter:container-quotas]
use = egg:swift#container_quotas

# Note: Put before both ratelimit and auth in the pipeline.
[filter:slo]
use = egg:swift#slo
# max_manifest_segments = 1000
# max_manifest_size = 2097152
# min_segment_size = 1048576
# Start rate-limiting SLO segment serving after the Nth segment of a
# segmented object.
# rate_limit_after_segment = 10
#
# Once segment rate-limiting kicks in for an object, limit segments served
# to N per second. 0 means no rate-limiting.
# rate_limit_segments_per_sec = 0
#
# Time limit on GET requests (seconds)
# max_get_time = 86400

# Note: Put before both ratelimit and auth in the pipeline, but after
# gatekeeper, catch_errors, and proxy_logging (the first instance).
# If you don't put it in the pipeline, it will be inserted for you.
[filter:dlo]
use = egg:swift#dlo
# Start rate-limiting DLO segment serving after the Nth segment of a
# segmented object.
# rate_limit_after_segment = 10
#
# Once segment rate-limiting kicks in for an object, limit segments served
# to N per second. 0 means no rate-limiting.
# rate_limit_segments_per_sec = 1
#
# Time limit on GET requests (seconds)
# max_get_time = 86400

[filter:account-quotas]
use = egg:swift#account_quotas
```

```
[filter:gatekeeper]
use = egg:swift#gatekeeper
# You can override the default log routing for this filter here:
# set log_name = gatekeeper
# set log_facility = LOG_LOCAL0
# set log_level = INFO
# set log_headers = false
# set log_address = /dev/log

[filter:container_sync]
use = egg:swift#container_sync
# Set this to false if you want to disallow any full url values to be set for
# any new X-Container-Sync-To headers. This will keep any new full urls from
# coming in, but won't change any existing values already in the cluster.
# Updating those will have to be done manually, as knowing what the true realm
# endpoint should be cannot always be guessed.
# allow_full_urls = true
```

Configure Object Storage features

Object Storage zones

In OpenStack Object Storage, data is placed across different tiers of failure domains. First, data is spread across regions, then zones, then servers, and finally across drives. Data is placed to get the highest failure domain isolation. If you deploy multiple regions, the Object Storage service places the data across the regions. Within a region, each replica of the data should be stored in unique zones, if possible. If there is only one zone, data should be placed on different servers. And if there is only one server, data should be placed on different drives.

Regions are widely separated installations with a high-latency or otherwise constrained network link between them. Zones are arbitrarily assigned, and it is up to the administrator of the Object Storage cluster to choose an isolation level and attempt to maintain the isolation level through appropriate zone assignment. For example, a zone may be defined as a rack with a single power source. Or a zone may be a DC room with a common utility provider. Servers are identified by a unique IP/port. Drives are locally attached storage volumes identified by mount point.

In small clusters (five nodes or fewer), everything is normally in a single zone. Larger Object Storage deployments may assign zone designations differently; for example, an entire cabinet or rack of servers may be designated as a single zone to maintain replica availability if the cabinet becomes unavailable (for example, due to failure of the top of rack switches or a dedicated circuit). In very large deployments, such as service provider level deployments, each zone might have an entirely autonomous switching and power infrastructure, so that even the loss of an electrical circuit or switching aggregator would result in the loss of a single replica at most.

Rackspace zone recommendations

For ease of maintenance on OpenStack Object Storage, Rackspace recommends that you set up at least five nodes. Each node will be assigned its own zone (for a total of five zones), which will give you host level redundancy. This allows you to take down a single zone for maintenance and still guarantee object availability in the event that another zone fails during your maintenance.

You could keep each server in its own cabinet to achieve cabinet level isolation, but you may wish to wait until your swift service is better established before developing cabinet-level isolation. OpenStack Object Storage is flexible; if you later decide to change the isolation level, you can take down one zone at a time and move them to appropriate new homes.

RAID controller configuration

OpenStack Object Storage does not require RAID. In fact, most RAID configurations cause significant performance degradation. The main reason for using a RAID controller is the battery-backed cache. It is very important for data integrity reasons that when the operating system confirms a write has been committed that the write has actually been committed to a persistent location. Most disks lie about hardware commits by default, instead writing to a faster write cache for performance reasons. In most cases, that write cache exists only in non-persistent memory. In the case of a loss of power, this data may never actually get committed to disk, resulting in discrepancies that the underlying file system must handle.

OpenStack Object Storage works best on the XFS file system, and this document assumes that the hardware being used is configured appropriately to be mounted with the **nobarriers** option. For more information, refer to the XFS FAQ: http://xfs.org/index.php/XFS_FAQ

To get the most out of your hardware, it is essential that every disk used in OpenStack Object Storage is configured as a standalone, individual RAID 0 disk; in the case of 6 disks, you would have six RAID 0s or one JBOD. Some RAID controllers do not support JBOD or do not support battery backed cache with JBOD. To ensure the integrity of your data, you must ensure that the individual drive caches are disabled and the battery backed cache in your RAID card is configured and used. Failure to configure the controller properly in this case puts data at risk in the case of sudden loss of power.

You can also use hybrid drives or similar options for battery backed up cache configurations without a RAID controller.

Throttle resources through rate limits

Rate limiting in OpenStack Object Storage is implemented as a pluggable middleware that you configure on the proxy server. Rate limiting is performed on requests that result in database writes to the account and container SQLite databases. It uses memcached and is dependent on the proxy servers having highly synchronized time. The rate limits are limited by the accuracy of the proxy server clocks.

Configure rate limiting

All configuration is optional. If no account or container limits are provided there will be no rate limiting. Available configuration options include:

Table 6.38. Description of configuration options for `[filter:ratelimit]` in `proxy-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#ratelimit	Entry point of paste.deploy in the server

Configuration option=Default value	Description
set log_name=ratelimit	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_headers=false	If True, log headers in each request
set log_address=/dev/log	No help text available for this option
clock_accuracy=1000	Represents how accurate the proxy servers' system clocks are with each other. 1000 means that all the proxies' clock are accurate to each other within 1 millisecond. No ratelimit should be higher than the clock accuracy.
max_sleep_time_seconds=60	App will immediately return a 498 response if the necessary sleep time ever exceeds the given max_sleep_time_seconds.
log_sleep_time_seconds=0	To allow visibility into rate limiting set this value > 0 and all sleeps greater than the number will be logged.
rate_buffer_seconds=5	Number of seconds the rate counter can drop and be allowed to catch up (at a faster than listed rate). A larger number will result in larger spikes in rate but better average accuracy.
account_ratelimit=0	If set, will limit PUT and DELETE requests to /account_name/container_name. Number is in requests per second.
account_whitelist=a,b	Comma separated lists of account names that will not be rate limited.
account_blacklist=c,d	Comma separated lists of account names that will not be allowed. Returns a 497 response. r: for containers of size x, limit requests per second to r. Will limit PUT, DELETE, and POST requests to /a/c/o. container_listing_ratelimit_x = r: for containers of size x, limit listing requests per second to r. Will limit GET requests to /a/c.
with container_limit_x=r	No help text available for this option
container_ratelimit_0=100	No help text available for this option
container_ratelimit_10=50	No help text available for this option
container_ratelimit_50=20	No help text available for this option
container_listing_ratelimit_0=100	No help text available for this option
container_listing_ratelimit_10=50	No help text available for this option
container_listing_ratelimit_50=20	No help text available for this option

The container rate limits are linearly interpolated from the values given. A sample container rate limiting could be:

container_ratelimit_100 = 100

container_ratelimit_200 = 50

container_ratelimit_500 = 20

This would result in:

Table 6.39. Values for Rate Limiting with Sample Configuration Settings

Container Size	Rate Limit
0-99	No limiting

100	100
150	75
500	20
1000	20

Health check

Provides an easy way to monitor whether the swift proxy server is alive. If you access the proxy with the path `/healthcheck`, it respond OK in the response body, which monitoring tools can use.

Table 6.40. Description of configuration options for `[filter:healthcheck]` in `account-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#healthcheck	Entry point of paste.deploy in the server
disable_path=	No help text available for this option

Domain remap

Middleware that translates container and account parts of a domain to path parameters that the proxy server understands.

Table 6.41. Description of configuration options for `[filter:domain_remap]` in `proxy-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#domain_remap	Entry point of paste.deploy in the server
set log_name=domain_remap	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_headers=false	If True, log headers in each request
set log_address=/dev/log	No help text available for this option
storage_domain=example.com	Domain to use for remap
path_root=v1	Root path
reseller_prefixes=AUTH	Reseller prefix

CNAME lookup

Middleware that translates an unknown domain in the host header to something that ends with the configured storage_domain by looking up the given domain's CNAME record in DNS.

Table 6.42. Description of configuration options for `[filter:cname_lookup]` in `proxy-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#cname_lookup	Entry point of paste.deploy in the server

Configuration option=Default value	Description
set log_name=cname_lookup	Label to use when logging
set log_facility=LOG_LOCAL0	Syslog log facility
set log_level=INFO	Log level
set log_headers=false	If True, log headers in each request
set log_address=/dev/log	No help text available for this option
storage_domain=example.com	Domain to use for remap
lookup_depth=1	As CNAMES can be recursive, how many levels to search through

Temporary URL

Allows the creation of URLs to provide temporary access to objects. For example, a website may wish to provide a link to download a large object in Swift, but the Swift account has no public access. The website can generate a URL that will provide GET access for a limited time to the resource. When the web browser user clicks on the link, the browser will download the object directly from Swift, obviating the need for the website to act as a proxy for the request. If the user were to share the link with all his friends, or accidentally post it on a forum, the direct access would be limited to the expiration time set when the website created the link.

A temporary URL is the typical URL associated with an object, with two additional query parameters:

`temp_url_sig` A cryptographic signature

`temp_url_expires` An expiration date, in Unix time.

An example of a temporary URL:

```
    https://swift-cluster.example.com/v1/AUTH_a422b2-91f3-2f46-74b7-
d7c9e8958f5d30/container/object?
    temp_url_sig=da39a3ee5e6b4b0d3255bfef95601890afd80709&
    temp_url_expires=1323479485
```

To create temporary URLs, first set the `X-Account-Meta-Temp-URL-Key` header on your Swift account to an arbitrary string. This string will serve as a secret key. For example, to set a key of `b3968d0207b54ece87cccc06515a89d4` using the **swift** command-line tool:

```
$ swift post -m "Temp-URL-Key:b3968d0207b54ece87cccc06515a89d4"
```

Next, generate an HMAC-SHA1 (RFC 2104) signature to specify:

- Which HTTP method to allow (typically GET or PUT)
- The expiry date as a Unix timestamp
- the full path to the object
- The secret key set as the `X-Account-Meta-Temp-URL-Key`

Here is code generating the signature for a GET for 24 hours on `/v1/AUTH_account/container/object`:

```
import hmac
from hashlib import sha1
from time import time
method = 'GET'
duration_in_seconds = 60*60*24
expires = int(time() + duration_in_seconds)
path = '/v1/AUTH_a422b2-91f3-2f46-74b7-d7c9e8958f5d30/container/object'
key = 'mykey'
hmac_body = '%s\n%s\n%s' % (method, expires, path)
sig = hmac.new(key, hmac_body, sha1).hexdigest()
s = 'https://{host}/{path}?temp_url_sig={sig}&temp_url_expires={expires}'
url = s.format(host='swift-cluster.example.com', path=path, sig=sig, expires=
expires)
```

Any alteration of the resource path or query arguments results in a 401 Unauthorized error. Similarly, a PUT where GET was the allowed method returns a 401. HEAD is allowed if GET or PUT is allowed. Using this in combination with browser form post translation middleware could also allow direct-from-browser uploads to specific locations in Swift. Note that

Note

Changing the `X-Account-Meta-Temp-URL-Key` will invalidate any previously generated temporary URLs within 60 seconds (the memcache time for the key). Swift supports up to two keys, specified by `X-Account-Meta-Temp-URL-Key` and `X-Account-Meta-Temp-URL-Key-2`. Signatures are checked against both keys, if present. This is to allow for key rotation without invalidating all existing temporary URLs.

Swift includes a script called **swift-temp-url** that will generate the query parameters automatically:

```
$ bin/swift-temp-url GET 3600 /v1/AUTH_account/container/object mykey
/v1/AUTH_account/container/object?
temp_url_sig=5c4cc8886f36a9d0919d708ade98bf0cc71c9e91&
temp_url_expires=1374497657
```

Because this command only returns the path, you must prefix the Swift storage hostname (for example, `https://swift-cluster.example.com`).

With GET Temporary URLs, a `Content-Disposition` header will be set on the response so that browsers will interpret this as a file attachment to be saved. The filename chosen is based on the object name, but you can override this with a `filename` query parameter. The following example specifies a filename of `My Test File.pdf`:

```
https://swift-cluster.example.com/v1/AUTH_a422b2-91f3-2f46-74b7-
d7c9e8958f5d30/container/object?
temp_url_sig=da39a3ee5e6b4b0d3255bfef95601890afd80709&
temp_url_expires=1323479485&
filename=My+Test+File.pdf
```

To enable Temporary URL functionality, edit `/etc/swift/proxy-server.conf` to add `tempurl` to the `pipeline` variable defined in the `[pipeline:main]` section. The

`tempurl` entry should appear immediately before the authentication filters in the pipeline, such as `authtoken`, `tempauth` or `keystoneauth`. For example:

```
[pipeline:main]
pipeline = pipeline = healthcheck cache tempurl authtoken keystoneauth proxy-server
```

Table 6.43. Description of configuration options for [filter:tempurl] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#tempurl	Entry point of paste.deploy in the server. You should not ever need to change this.
methods=GET HEAD PUT	HTTP methods allowed with Temporary URLs
incoming_remove_headers=x-timestamp	Headers to remove from incoming requests. Simply a whitespace delimited list of header names and names can optionally end with '*' to indicate a prefix match.
incoming_allow_headers=	Headers allowed as exceptions to incoming_remove_headers. Simply a whitespace delimited list of header names and names can optionally end with '*' to indicate a prefix match.
outgoing_remove_headers=x-object-meta-*	Headers to remove from outgoing responses. Simply a whitespace delimited list of header names and names can optionally end with '*' to indicate a prefix match.
outgoing_allow_headers=x-object-meta-public-*	Headers allowed as exceptions to outgoing_allow_headers. Simply a whitespace delimited list of header names and names can optionally end with '*' to indicate a prefix match.

Name check filter

Name Check is a filter that disallows any paths that contain defined forbidden characters or that exceed a defined length.

Table 6.44. Description of configuration options for [filter:name_check] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#name_check	Entry point of paste.deploy in the server
forbidden_chars='"`<>	Characters that are not allowed in a name
maximum_length=255	Maximum length of a name
forbidden_regexp=/\./|/\.\./|/\.$\|/\.\.$	Substrings to forbid, using regular expression syntax

Constraints

To change the OpenStack Object Storage internal limits, update the values in the `swift-constraints` section in the `swift.conf` file. Use caution when you update these values because they affect the performance in the entire cluster.

Table 6.45. Description of configuration options for [swift-constraints] in swift.conf-sample

Configuration option=Default value	Description
max_file_size=5368709122	The largest normal object that can be saved in the cluster. This is also the limit on the size of each segment of a large

Configuration option=Default value	Description
	object when using the large object manifest support. This value is set in bytes. Setting it to lower than 1MiB will cause some tests to fail. It is STRONGLY recommended to leave this value at the default (5 * 2**30 + 2).
max_meta_name_length=128	The maximum number of bytes in the utf8 encoding of the name portion of a metadata header.
max_meta_value_length=256	The max number of bytes in the utf8 encoding of a metadata value.
max_meta_count=90	The maximum number of metadata keys that can be stored on a single account, container, or object.
max_meta_overall_size=4096	The maximum number of bytes in the utf8 encoding of the metadata (keys + values).
max_header_size=8192	The maximum number of bytes in the utf8 encoding of each header.
max_object_name_length=1024	The maximum number of bytes in the utf8 encoding of an object name.
container_listing_limit=10000	The default (and maximum) number of items returned for a container listing request.
account_listing_limit=10000	The default (and maximum) number of items returned for an account listing request.
max_account_name_length=256	The maximum number of bytes in the utf8 encoding of an account name.
max_container_name_length=256	The maximum number of bytes in the utf8 encoding of a container name.

Cluster health

Use the **swift-dispersion-report** tool to measure overall cluster health. This tool checks if a set of deliberately distributed containers and objects are currently in their proper places within the cluster. For instance, a common deployment has three replicas of each object. The health of that object can be measured by checking if each replica is in its proper place. If only 2 of the 3 is in place the object's health can be said to be at 66.66%, where 100% would be perfect. A single object's health, especially an older object, usually reflects the health of that entire partition the object is in. If you make enough objects on a distinct percentage of the partitions in the cluster,you get a good estimate of the overall cluster health. In practice, about 1% partition coverage seems to balance well between accuracy and the amount of time it takes to gather results. The first thing that needs to be done to provide this health value is create a new account solely for this usage. Next, you need to place the containers and objects throughout the system so that they are on distinct partitions. The swift-dispersion-populate tool does this by making up random container and object names until they fall on distinct partitions. Last, and repeatedly for the life of the cluster, you need to run the **swift-dispersion-report** tool to check the health of each of these containers and objects. These tools need direct access to the entire cluster and to the ring files (installing them on a proxy server will probably do). Both **swift-dispersion-populate** and **swift-dispersion-report** use the same configuration file, `/etc/swift/dispersion.conf`. Example dispersion.conf file:

```
[dispersion]
auth_url = http://localhost:8080/auth/v1.0
auth_user = test:tester
auth_key = testing
```

There are also options for the conf file for specifying the dispersion coverage (defaults to 1%), retries, concurrency, etc. though usually the defaults are fine. Once the configuration is in place, run swift-dispersion-populate to populate the containers and objects throughout the cluster. Now that those containers and objects are in place, you can run swift-dispersion-report to get a dispersion report, or the overall health of the cluster. Here is an example of a cluster in perfect health:

```
$ swift-dispersion-report
Queried 2621 containers for dispersion reporting, 19s, 0 retries
100.00% of container copies found (7863 of 7863)
Sample represents 1.00% of the container partition space

Queried 2619 objects for dispersion reporting, 7s, 0 retries
100.00% of object copies found (7857 of 7857)
Sample represents 1.00% of the object partition space
```

Now, deliberately double the weight of a device in the object ring (with replication turned off) and rerun the dispersion report to show what impact that has:

```
$ swift-ring-builder object.builder set_weight d0 200
$ swift-ring-builder object.builder rebalance
...
$ swift-dispersion-report
Queried 2621 containers for dispersion reporting, 8s, 0 retries
100.00% of container copies found (7863 of 7863)
Sample represents 1.00% of the container partition space

Queried 2619 objects for dispersion reporting, 7s, 0 retries
There were 1763 partitions missing one copy.
77.56% of object copies found (6094 of 7857)
Sample represents 1.00% of the object partition space
```

You can see the health of the objects in the cluster has gone down significantly. Of course, this test environment has just four devices, in a production environment with many devices the impact of one device change is much less. Next, run the replicators to get everything put back into place and then rerun the dispersion report:

```
... start object replicators and monitor logs until they're caught up ...
$ swift-dispersion-report
Queried 2621 containers for dispersion reporting, 17s, 0 retries
100.00% of container copies found (7863 of 7863)
Sample represents 1.00% of the container partition space

Queried 2619 objects for dispersion reporting, 7s, 0 retries
100.00% of object copies found (7857 of 7857)
Sample represents 1.00% of the object partition space
```

Alternatively, the dispersion report can also be output in json format. This allows it to be more easily consumed by third party utilities:

```
$ swift-dispersion-report -j
{"object": {"retries:": 0, "missing_two": 0, "copies_found": 7863,
 "missing_one": 0,
"copies_expected": 7863, "pct_found": 100.0, "overlapping": 0, "missing_all":
 0}, "container":
{"retries:": 0, "missing_two": 0, "copies_found": 12534, "missing_one": 0,
 "copies_expected":
12534, "pct_found": 100.0, "overlapping": 15, "missing_all": 0}}
```

Table 6.46. Description of configuration options for [dispersion] in dispersion.conf-sample

Configuration option=Default value	Description
auth_url=http://localhost:8080/auth/v1.0	Endpoint for auth server, such as keystone
auth_user=test:tester	Default user for dispersion in this context
auth_key=testing	No help text available for this option
auth_url=http://saio:5000/v2.0/	Endpoint for auth server, such as keystone
auth_user=test:tester	Default user for dispersion in this context
auth_key=testing	No help text available for this option
auth_version=2.0	Indicates which version of auth
endpoint_type=publicURL	Indicates whether endpoint for auth is public or internal
keystone_api_insecure=no	No help text available for this option
swift_dir=/etc/swift	Swift configuration directory
dispersion_coverage=1.0	No help text available for this option
retries=5	No help text available for this option
concurrency=25	Number of replication workers to spawn
container_report=yes	No help text available for this option
object_report=yes	No help text available for this option
dump_json=no	No help text available for this option

Static Large Object (SLO) support

This feature is very similar to Dynamic Large Object (DLO) support in that it allows the user to upload many objects concurrently and afterwards download them as a single object. It is different in that it does not rely on eventually consistent container listings to do so. Instead, a user defined manifest of the object segments is used.

Table 6.47. Description of configuration options for [filter:slo] in proxy-server.conf-sample

Configuration option=Default value	Description
use=egg:swift#slo	Entry point of paste.deploy in the server
max_manifest_segments=1000	No help text available for this option
max_manifest_size=2097152	No help text available for this option
min_segment_size=1048576	No help text available for this option

Container quotas

The container_quotas middleware implements simple quotas that can be imposed on swift containers by a user with the ability to set container metadata, most likely the account administrator. This can be useful for limiting the scope of containers that are delegated to non-admin users, exposed to formpost uploads, or just as a self-imposed sanity check.

Any object PUT operations that exceed these quotas return a 413 response (request entity too large) with a descriptive body.

Quotas are subject to several limitations: eventual consistency, the timeliness of the cached container_info (60 second ttl by default), and it is unable to reject chunked transfer

uploads that exceed the quota (though once the quota is exceeded, new chunked transfers will be refused).

Quotas are set by adding meta values to the container, and are validated when set:

- X-Container-Meta-Quota-Bytes: Maximum size of the container, in bytes.
- X-Container-Meta-Quota-Count: Maximum object count of the container.

Table 6.48. Description of configuration options for `[filter:container-quotas]` in `proxy-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#container_quotas	Entry point of paste.deploy in the server

Account quotas

The *`x-account-meta-quota-bytes`* metadata entry must be requests (PUT, POST) if a given account quota (in bytes) is exceeded while DELETE requests are still allowed.

The x-account-meta-quota-bytes metadata entry must be set to store and enable the quota. Write requests to this metadata entry are only permitted for resellers. There is no account quota limitation on a reseller account even if x-account-meta-quota-bytes is set.

Any object PUT operations that exceed the quota return a 413 response (request entity too large) with a descriptive body.

The following command uses an admin account that own the Reseller role to set a quota on the test account:

```
$ swift -A http://127.0.0.1:8080/auth/v1.0 -U admin:admin -K admin \
--os-storage-url=http://127.0.0.1:8080/v1/AUTH_test post -m quota-bytes:10000
```

Here is the stat listing of an account where quota has been set:

```
$ swift -A http://127.0.0.1:8080/auth/v1.0 -U test:tester -K testing stat
Account: AUTH_test
Containers: 0
Objects: 0
Bytes: 0
Meta Quota-Bytes: 10000
X-Timestamp: 1374075958.37454
X-Trans-Id: tx602634cf478546a39b1be-0051e6bc7a
```

The command below removes the account quota:

```
$ swift -A http://127.0.0.1:8080/auth/v1.0 -U admin:admin -K admin --os-
storage-url=http://127.0.0.1:8080/v1/AUTH_test post -m quota-bytes:
```

Bulk delete

Will delete multiple files from their account with a single request. Responds to DELETE requests with a header 'X-Bulk-Delete: true_value'. The body of the DELETE request will be a newline separated list of files to delete. The files listed must be URL encoded and in the form:

```
/container_name/obj_name
```

If all files were successfully deleted (or did not exist) will return an HTTPOk. If any files failed to delete will return an HTTPBadGateway. In both cases the response body is a json dictionary specifying in the number of files successfully deleted, not found, and a list of the files that failed.

Table 6.49. Description of configuration options for `[filter:bulk]` in `proxy-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#bulk	Entry point of paste.deploy in the server
max_containers_per_extraction=10000	No help text available for this option
max_failed_extractions=1000	No help text available for this option
max_deletes_per_request=10000	No help text available for this option
yield_frequency=60	No help text available for this option

Configure Object Storage with the S3 API

The Swift3 middleware emulates the S3 REST API on top of Object Storage.

The following operations are currently supported:

- GET Service
- DELETE Bucket
- GET Bucket (List Objects)
- PUT Bucket
- DELETE Object
- GET Object
- HEAD Object
- PUT Object
- PUT Object (Copy)

To use this middleware, first download the latest version from its repository to your proxy server(s).

```
$ git clone https://github.com/fujita/swift3.git
```

Optional: To use this middleware with Swift 1.7.0 and previous versions, you'll need to use the v1.7 tag of the fujita/swift3 repository. Clone the repo as above and then:

```
$ cd swift3; git checkout v1.7
```

Then, install it using standard python mechanisms, such as:

```
$ sudo python setup.py install
```

Alternatively, if you have configured the Ubuntu Cloud Archive, you may use:

```
$ sudo apt-get install swift-python-s3
```

To add this middleware to your configuration, add the swift3 middleware in front of the auth middleware, and before any other middleware that look at swift requests (like rate limiting).

Ensure that your proxy-server.conf file contains swift3 in the pipeline and the `[filter:swift3]` section, as shown below:

```
[pipeline:main]
pipeline = healthcheck cache swift3 swauth proxy-server

[filter:swift3]
use = egg:swift3#swift3
```

Next, configure the tool that you use to connect to the S3 API. For S3curl, for example, you'll need to add your host IP information by adding y our host IP to the @endpoints array (line 33 in s3curl.pl):

```
my @endpoints = ( '1.2.3.4');
```

Now you can send commands to the endpoint, such as:

```
$ ./s3curl.pl - 'myacc:myuser' -key mypw -get - -s -v http://1.2.3.4:8080
```

To set up your client, the access key will be the concatenation of the account and user strings that should look like test:tester, and the secret access key is the account password. The host should also point to the Swift storage node's hostname. It also will have to use the old-style calling format, and not the hostname-based container format. Here is an example client setup using the Python boto library on a locally installed all-in-one Swift installation.

```
connection = boto.s3.Connection(
    aws_access_key_id='test:tester',
    aws_secret_access_key='testing',
    port=8080,
    host='127.0.0.1',
    is_secure=False,
    calling_format=boto.s3.connection.OrdinaryCallingFormat())
```

Drive audit

The `swift-drive-audit` configuration items reference a script that can be run by using **cron** to watch for bad drives. If errors are detected, it will unmount the bad drive, so that OpenStack Object Storage can work around it. It takes the following options:

Table 6.50. Description of configuration options for `[drive-audit]` in `drive-audit.conf-sample`

Configuration option=Default value	Description
device_dir=/srv/node	Directory devices are mounted under

Configuration option=Default value	Description
log_facility=LOG_LOCAL0	Syslog log facility
log_level=INFO	Logging level
log_address=/dev/log	Location where syslog sends the logs to
minutes=60	Number of minutes to look back in `/var/log/kern.log`
error_limit=1	Number of errors to find before a device is unmounted
log_file_pattern=/var/log/kern*	Location of the log file with globbing pattern to check against device errors locate device blocks with errors in the log file
regex_pattern_1=\berror\b.*\b(dm-[0-9]{1,2}\d?)\b	No help text available for this option

Form post

Middleware that provides the ability to upload objects to a cluster using an HTML form POST. The format of the form is:

```
<![CDATA[
<form action="<swift-url>" method="POST"
      enctype="multipart/form-data">
  <input type="hidden" name="redirect" value="<redirect-url>" />
  <input type="hidden" name="max_file_size" value="<bytes>" />
  <input type="hidden" name="max_file_count" value="<count>" />
  <input type="hidden" name="expires" value="<unix-timestamp>" />
  <input type="hidden" name="signature" value="<hmac>" />
  <input type="file" name="file1" /><br />
  <input type="submit" />
</form>]]>
```

The `swift-url` is the URL to the Swift destination, such as: `https://swift-cluster.example.com/v1/AUTH_account/container/object_prefix` The name of each file uploaded is appended to the specified `swift-url`. So, you can upload directly to the root of container with a url like: `https://swift-cluster.example.com/v1/AUTH_account/container/` Optionally, you can include an object prefix to better separate different users' uploads, such as: `https://swift-cluster.example.com/v1/AUTH_account/container/object_prefix`

Note the form method must be POST and the enctype must be set as `multipart/form-data`.

The redirect attribute is the URL to redirect the browser to after the upload completes. The URL will have status and message query parameters added to it, indicating the HTTP status code for the upload (2xx is success) and a possible message for further information if there was an error (such as "`max_file_size exceeded`").

The `max_file_size` attribute must be included and indicates the largest single file upload that can be done, in bytes.

The `max_file_count` attribute must be included and indicates the maximum number of files that can be uploaded with the form. Include additional `<![CDATA[<input type="file" name="filexx"/>]]>` attributes if desired.

The expires attribute is the Unix timestamp before which the form must be submitted before it is invalidated.

The signature attribute is the HMAC-SHA1 signature of the form. This sample Python code shows how to compute the signature:

```
import hmac
from hashlib import sha1
from time import time
path = '/v1/account/container/object_prefix'
redirect = 'https://myserver.com/some-page'
max_file_size = 104857600
max_file_count = 10
expires = int(time() + 600)
key = 'mykey'
hmac_body = '%s\n%s\n%s\n%s\n%s' % (path, redirect,
    max_file_size, max_file_count, expires)
signature = hmac.new(key, hmac_body, sha1).hexdigest()
```

The key is the value of the `X-Account-Meta-Temp-URL-Key` header on the account.

Be certain to use the full path, from the `/v1/` onward.

The command line tool **swift-form-signature** may be used (mostly just when testing) to compute expires and signature.

The file attributes must appear after the other attributes to be processed correctly. If attributes come after the file, they are not sent with the sub-request because on the server side, all attributes in the file cannot be parsed unless the whole file is read into memory and the server does not have enough memory to service these requests. So, attributes that follow the file are ignored.

Table 6.51. Description of configuration options for `[filter:formpost]` in `proxy-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#formpost	Entry point of paste.deploy in the server

Static web sites

When configured, this middleware serves container data as a static web site with index file and error file resolution and optional file listings. This mode is normally only active for anonymous requests.

Table 6.52. Description of configuration options for `[filter:staticweb]` in `proxy-server.conf-sample`

Configuration option=Default value	Description
use=egg:swift#staticweb	Entry point of paste.deploy in the server

7. Block Storage

Table of Contents

The OpenStack Block Storage Service works with many different storage drivers that you can configure by using these instructions.

Introduction to the Block Storage Service

The Openstack Block Storage Service provides persistent block storage resources that OpenStack Compute instances can consume. This includes secondary attached storage similar to the Amazon Elastic Block Storage (EBS) offering. In addition, you can write images to a Block Storage device for Compute to use as a bootable persistent instance.

The Block Storage Service differs slightly from the Amazon EBS offering. The Block Storage Service does not provide a shared storage solution like NFS. With the Block Storage Service, you can attach a device to only one instance.

The Block Storage Service provides:

- `cinder-api`. A WSGI app that authenticates and routes requests throughout the Block Storage Service. It supports the OpenStack APIs only, although there is a translation that can be done through Compute's EC2 interface, which calls in to the cinderclient.
- `cinder-scheduler`. Schedules and routes requests to the appropriate volume service. As of Grizzly; depending upon your configuration this may be simple round-robin scheduling to the running volume services, or it can be more sophisticated through the use of the Filter Scheduler. The Filter Scheduler is the default in Grizzly and enables filters on things like Capacity, Availability Zone, Volume Types, and Capabilities as well as custom filters.
- `cinder-volume`. Manages Block Storage devices, specifically the back-end devices themselves.
- `cinder-backup` Provides a means to back up a Block Storage Volume to OpenStack Object Store (SWIFT).

The Block Storage Service contains the following components:

- **Back-end Storage Devices**. The Block Storage Service requires some form of back-end storage that the service is built on. The default implementation is to use LVM on a local volume group named "cinder-volumes." In addition to the base driver implementation, the Block Storage Service also provides the means to add support for other storage devices to be utilized such as external Raid Arrays or other storage appliances. These

back-end storage devices may have custom block sizes when using KVM or QEMU as the hypervisor.

- **Users and Tenants (Projects)**. The Block Storage Service is designed to be used by many different cloud computing consumers or customers, basically tenants on a shared system, using role-based access assignments. Roles control the actions that a user is allowed to perform. In the default configuration, most actions do not require a particular role, but this is configurable by the system administrator editing the appropriate `policy.json` file that maintains the rules. A user's access to particular volumes is limited by tenant, but the username and password are assigned per user. Key pairs granting access to a volume are enabled per user, but quotas to control resource consumption across available hardware resources are per tenant.

  For tenants, quota controls are available to limit:

  - The number of volumes that can be created
  - The number of snapshots that can be created
  - The total number of GBs allowed per tenant (shared between snapshots and volumes)

  You can revise the default quota values with the cinder CLI, so the limits placed by quotas are editable by admin users.

- **Volumes, Snapshots, and Backups**. The basic resources offered by the Block Storage Service are volumes and snapshots, which are derived from volumes, and backups:

  - **Volumes**. Allocated block storage resources that can be attached to instances as secondary storage or they can be used as the root store to boot instances. Volumes are persistent R/W block storage devices most commonly attached to the Compute node through iSCSI.
  - **Snapshots**. A read-only point in time copy of a volume. The snapshot can be created from a volume that is currently in use (through the use of '–force True') or in an available state. The snapshot can then be used to create a new volume through create from snapshot.
  - **Backups**. An archived copy of a volume currently stored in OpenStack Object Storage (Swift).

`cinder.conf` configuration file

The `cinder.conf` file is installed in `/etc/cinder` by default. When you manually install the Block Storage Service, the options in the `cinder.conf` file are set to default values.

This example shows a typical `cinder.conf` file:

```
[DEFAULT]
rootwrap_config=/etc/cinder/rootwrap.conf
sql_connection = mysql://cinder:openstack@192.168.127.130/cinder
api_paste_config = /etc/cinder/api-paste.ini

iscsi_helper=tgtadm
volume_name_template = volume-%s
```

```
volume_group = cinder-volumes
verbose = True
auth_strategy = keystone
#osapi_volume_listen_port=5900

# Add these when not using the defaults.
rabbit_host = 10.10.10.10
rabbit_port = 5672
rabbit_userid = rabbit
rabbit_password = secure_password
rabbit_virtual_host = /nova
```

Volume drivers

To use different volume drivers for the `cinder-volume` service, use the parameters described in these sections.

The volume drivers are included in the Block Storage repository (https://github.com/openstack/cinder). To set a volume driver, use the `volume_driver` flag. The default is:

```
volume_driver=cinder.volume.driver.ISCSIDriver
iscsi_helper=tgtadm
```

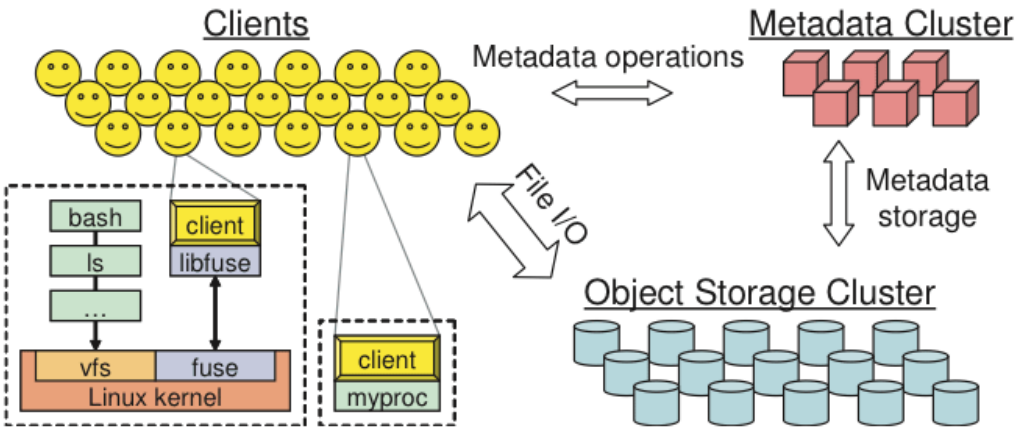
Ceph RADOS Block Device (RBD)

By Sebastien Han from http://www.sebastien-han.fr/blog/2012/06/10/introducing-ceph-to-openstack/

If you use KVM or QEMU as your hypervisor, you can configure the Compute service to use Ceph RADOS block devices (RBD) for volumes.

Ceph is a massively scalable, open source, distributed storage system. It is comprised of an object store, block store, and a POSIX-compliant distributed file system. The platform can auto-scale to the exabyte level and beyond. It runs on commodity hardware, is self-healing and self-managing, and has no single point of failure. Ceph is in the Linux kernel and is integrated with the OpenStack cloud operating system. Due to its open source nature, you can install and use this portable storage platform in public or private clouds.

Figure 7.1. Ceph architecture



RADOS?

You can easily get confused by the naming: Ceph? RADOS?

RADOS: Reliable Autonomic Distributed Object Store is an object store. RADOS distributes objects across the storage cluster and replicates objects for fault tolerance. RADOS contains the following major components:

- *Object Storage Device (ODS)*. The storage daemon - RADOS service, the location of your data. You must run this daemon on each server in your cluster. For each OSD, you can have an associated hard drive disks. For performance purposes, pool your hard drive disk with raid arrays, logical volume management (LVM) or B-tree file system (Btrfs) pooling. By default, the following pools are created: data, metadata, and RBD.

- *Meta-Data Server (MDS)*. Stores metadata. MDSs build a POSIX file system on top of objects for Ceph clients. However, if you do not use the Ceph file system, you do not need a metadata server.

- *Monitor (MON)*. This lightweight daemon handles all communications with external applications and clients. It also provides a consensus for distributed decision making in a Ceph/RADOS cluster. For instance, when you mount a Ceph shared on a client, you point to the address of a MON server. It checks the state and the consistency of the data. In an ideal setup, you must run at least three `ceph-mon` daemons on separate servers.

Ceph developers recommend that you use Btrfs as a file system for storage. XFS might be a better alternative for production environments. Neither Ceph nor Btrfs is ready for production and it could be risky to use them in combination. XFS is an excellent alternative to Btrfs. The ext4 file system is also compatible but does not exploit the power of Ceph.

Note

Currently, configure Ceph to use the XFS file system. Use Btrfs when it is stable enough for production.

See ceph.com/docs/master/rec/file system/ for more information about usable file systems.

Ways to store, use, and expose data

To store and access your data, you can use the following storage systems:

- *RADOS*. Use as an object, default storage mechanism.

- *RBD*. Use as a block device. The Linux kernel RBD (rados block device) driver allows striping a Linux block device over multiple distributed object store data objects. It is compatible with the kvm RBD image.

- *CephFS*. Use as a file, POSIX-compliant file system.

Ceph exposes its distributed object store (RADOS). You can access it through the following interfaces:

- *RADOS Gateway*. Swift and Amazon-S3 compatible RESTful interface. See RADOS_Gateway for more information.

- *librados*, and the related C/C++ bindings.
- *rbd and QEMU-RBD*. Linux kernel and QEMU block devices that stripe data across multiple objects.

For detailed installation instructions and benchmarking information, see http://www.sebastien-han.fr/blog/2012/06/10/introducing-ceph-to-openstack/.

Driver options

The following table contains the configuration options supported by the Ceph RADOS Block Device driver.

Table 7.1. Description of configuration options for storage_ceph

Configuration option=Default value	Description
rbd_ceph_conf=	(StrOpt) path to the ceph configuration file to use
rbd_flatten_volume_from_snapshot=False	(BoolOpt) flatten volumes created from snapshots to remove dependency
rbd_max_clone_depth=5	(IntOpt) maximum number of nested clones that can be taken of a volume before enforcing a flatten prior to next clone. A value of zero disables cloning
rbd_pool=rbd	(StrOpt) the RADOS pool in which rbd volumes are stored
rbd_secret_uuid=None	(StrOpt) the libvirt uuid of the secret for the rbd_uservolumes
rbd_user=None	(StrOpt) the RADOS client name for accessing rbd volumes - only set when using cephx authentication
volume_tmp_dir=None	(StrOpt) where to store temporary image files if the volume driver does not write them directly to the volume

Coraid AoE driver configuration

Coraid storage appliances can provide block-level storage to OpenStack instances. Coraid storage appliances use the low-latency ATA-over-Ethernet (ATA) protocol to provide high-bandwidth data transfer between hosts and data on the network.

Once configured for OpenStack, you can:

- Create, delete, attach, and detach block storage volumes.
- Create, list, and delete volume snapshots.
- Create a volume from a snapshot, copy an image to a volume, copy a volume to an image, clone a volume, and get volume statistics.

This document describes how to configure the OpenStack Block Storage Service for use with Coraid storage appliances.

Terminology

These terms are used in this section:

Term	Definition
AoE	ATA-over-Ethernet protocol

Term	Definition
EtherCloud Storage Manager (ESM)	ESM provides live monitoring and management of EtherDrive appliances that use the AoE protocol, such as the SRX and VSX.
Fully-Qualified Repository Name (FQRN)	The FQRN is the full identifier of a storage profile. FQRN syntax is: *performance_class*-*availability_class*:*profile_name*:*repository_name*
SAN	Storage Area Network
SRX	Coraid EtherDrive SRX block storage appliance
VSX	Coraid EtherDrive VSX storage virtualization appliance

Requirements

To support the OpenStack Block Storage Service, your SAN must include an SRX for physical storage, a VSX running at least CorOS v2.0.6 for snapshot support, and an ESM running at least v2.1.1 for storage repository orchestration. Ensure that all storage appliances are installed and connected to your network before you configure OpenStack volumes.

So that the node can communicate with the SAN, you must install the Coraid AoE Linux driver on each compute node on the network that runs an OpenStack instance.

Overview

To configure the OpenStack Block Storage for use with Coraid storage appliances, perform the following procedures:

1. Download and install the Coraid Linux AoE driver.
2. Create a storage profile by using the Coraid ESM GUI.
3. Create a storage repository by using the ESM GUI and record the FQRN.
4. Configure the `cinder.conf` file.
5. Create and associate a block storage volume type.

Install the Coraid AoE driver

Install the Coraid AoE driver on every compute node that will require access to block storage.

The latest AoE drivers will always be located at http://support.coraid.com/support/linux/.

To download and install the AoE driver, follow the instructions below, replacing “aoeXXX” with the AoE driver file name:

1. Download the latest Coraid AoE driver.

```
# wget http://support.coraid.com/support/linux/aoeXXX.tar.gz
```

2. Unpack the AoE driver.

3. Install the AoE driver.

```
# cd aoeXXX

# make

# make install
```

4. Initialize the AoE driver.

```
# modprobe aoe
```

5. Optionally, specify the Ethernet interfaces that the node can use to communicate with the SAN.

   The AoE driver may use every Ethernet interface available to the node unless limited with the `aoe_iflist` parameter. For more information about the `aoe_iflist` parameter, see the `aoe readme` file included with the AoE driver.

```
# modprobe aoe_iflist="eth1 eth2 ..."
```

Create a storage profile

To create a storage profile using the ESM GUI:

1. Log in to the ESM.
2. Click **Storage Profiles** in the **SAN Domain** pane.
3. Choose **Menu > Create Storage Profile**. If the option is unavailable, you might not have appropriate permissions. Make sure you are logged in to the ESM as the SAN administrator.
4. Use the storage class selector to select a storage class.

   Each storage class includes performance and availability criteria (see the Storage Classes topic in the ESM Online Help for information on the different options).
5. Select a RAID type (if more than one is available) for the selected profile type.
6. Type a **Storage Profile** name.

   The name is restricted to alphanumeric characters, underscore (_), and hyphen (-), and cannot exceed 32 characters.
7. Select the drive size from the drop-down menu.
8. Select the number of drives to be initialized for each RAID (LUN) from the drop-down menu (if the selected RAID type requires multiple drives).
9. Type the number of RAID sets (LUNs) you want to create in the repository by using this profile.
10. Click **Next**.

Create a storage repository and get the FQRN

Create a storage repository and get its fully qualified repository name (FQRN):

1. Access the **Create Storage Repository** dialog box.
2. Type a Storage Repository name.

   The name is restricted to alphanumeric characters, underscore (_), hyphen (-), and cannot exceed 32 characters.
3. Click **Limited** or **Unlimited** to indicate the maximum repository size.

   Limited sets the amount of space that can be allocated to the repository. Specify the size in TB, GB, or MB.

   When the difference between the reserved space and the space already allocated to LUNs is less than is required by a LUN allocation request, the reserved space is increased until the repository limit is reached.

   ### Note

   The reserved space does not include space used for parity or space used for mirrors. If parity and/or mirrors are required, the actual space allocated to the repository from the SAN is greater than that specified in reserved space.

   Unlimited—Unlimited means that the amount of space allocated to the repository is unlimited and additional space is allocated to the repository automatically when space is required and available.

   ### Note

   Drives specified in the associated Storage Profile must be available on the SAN in order to allocate additional resources.
4. Check the **Resizeable LUN** box.

   This is required for OpenStack volumes.

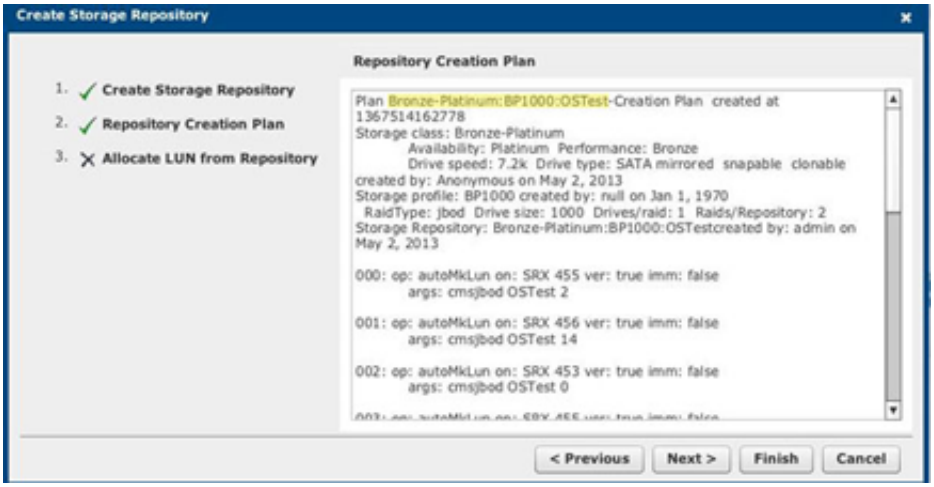
   ### Note

   If the Storage Profile associated with the repository has platinum availability, the Resizeable LUN box is automatically checked.
5. Check the **Show Allocation Plan API calls** box. Click **Next**.
6. Record the FQRN and click **Finish**.

   The FQRN is located in the first line of output following the `Plan` keyword in the **Repository Creation Plan** window. The FQRN syntax is *`performance_class`*-*`availability_class`*:*`profile_name`*:*`repository_name`*.

In this example, the FQRN is `Bronze-Platinum:BP1000:OSTest`, and is highlighted.

Figure 7.2. Repository Creation Plan screen



Record the FQRN; it is a required parameter later in the configuration procedure.

Configure options in the cinder.conf file

Edit or add the following lines to the file `/etc/cinder/cinder.conf`:

```
volume_driver = cinder.volume.drivers.coraid.CoraidDriver
coraid_esm_address = ESM_IP_address
coraid_user = username
coraid_group = Access_Control_Group_name
coraid_password = password
coraid_repository_key = coraid_repository_key
```

Table 7.2. Description of configuration options for coraid

Configuration option=Default value	Description
coraid_esm_address=	(StrOpt) IP address of Coraid ESM
coraid_group=admin	(StrOpt) Name of group on Coraid ESM to which coraid_user belongs (must have admin privilege)
coraid_password=password	(StrOpt) Password to connect to Coraid ESM
coraid_repository_key=coraid_repository	(StrOpt) Volume Type key name to store ESM Repository Name
coraid_user=admin	(StrOpt) User name to connect to Coraid ESM

Access to storage devices and storage repositories can be controlled using Access Control Groups configured in ESM. Configuring `cinder.conf` to log on to ESM as the SAN administrator (user name `admin`), will grant full access to the devices and repositories configured in ESM.

Optionally, you can configure an ESM Access Control Group and user. Then, use the `cinder.conf` file to configure access to the ESM through that group, and user limits access from the OpenStack instance to devices and storage repositories that are defined in the group.

To manage access to the SAN by using Access Control Groups, you must enable the Use Access Control setting in the **ESM System Setup** > **Security** screen.

For more information, see the ESM Online Help.

Create and associate a volume type

Create and associate a volume with the ESM storage repository.

1. Restart Cinder.

```
# service openstack-cinder-api restart

# service openstack-cinder-scheduler restart

# service openstack-cinder-volume restart
```

2. Create a volume.

```
# cinder type-create 'volume_type_name'
```

where *volume_type_name* is the name you assign the volume. You will see output similar to the following:

```
+--------------------------------------+-------------+
|                  ID                  |     Name    |
+--------------------------------------+-------------+
| 7fa6b5ab-3e20-40f0-b773-dd9e16778722 | JBOD-SAS600 |
+--------------------------------------+-------------+
```

Record the value in the ID field; you use this value in the next step.

3. Associate the volume type with the Storage Repository.

```
#cinder type-key UUID set coraid_repository_key='FQRN'
```

Variable	Description
UUID	The ID returned from the **cinder type-create** command. You can use the **cinder type-list** command to recover the ID.
coraid_repository_key	The key name used to associate the Cinder volume type with the ESM in the `cinder.conf` file. If no key name was defined, this is default value for `coraid_repository`.
FQRN	The FQRN recorded during the Create Storage Repository process.

Dell EqualLogic volume driver

The Dell EqualLogic volume driver interacts with configured EqualLogic arrays and supports various operations, such as volume creation and deletion, volume attachment and detachment, snapshot creation and deletion, and clone creation.

To configure and use a Dell EqualLogic array with Block Storage, modify your `cinder.conf` as follows.

Set the `volume_driver` option to the Dell EqualLogic volume driver:

```
volume_driver=cinder.volume.drivers.eqlx.DellEQLSanISCSIDriver
```

Set the `san_ip` option to the IP address to reach the EqualLogic Group through SSH:

```
san_ip=10.10.72.53
```

Set the `san_login` option to the user name to login to the Group manager:

```
san_login=grpadmin
```

Set the `san_password` option to the password to login the Group manager with:

```
san_password=password
```

Optionally set the `san_thin_provision` option to false to disable creation of thin-provisioned volumes:

```
san_thin_provision=false
```

The following table describes additional options that the driver supports:

Table 7.3. Description of configuration options for eqlx

Configuration option=Default value	Description
eqlx_chap_login=admin	(StrOpt) Existing CHAP account name
eqlx_chap_password=password	(StrOpt) Password for specified CHAP account name
eqlx_cli_max_retries=5	(IntOpt) Maximum retry count for reconnection
eqlx_cli_timeout=30	(IntOpt) Timeout for the Group Manager cli command execution
eqlx_group_name=group-0	(StrOpt) Group name to use for creating volumes
eqlx_pool=default	(StrOpt) Pool in which volumes will be created
eqlx_use_chap=False	(BoolOpt) Use CHAP authentificaion for targets?

EMC SMI-S iSCSI driver

The EMC SMI-S iSCSI driver, which is based on the iSCSI driver, can create, delete, attach, and detach volumes, create and delete snapshots, and so on.

The EMC SMI-S iSCSI driver runs volume operations by communicating with the back-end EMC storage. It uses a CIM client in Python called PyWBEM to perform CIM operations over HTTP.

The EMC CIM Object Manager (ECOM) is packaged with the EMC SMI-S provider. It is a CIM server that enables CIM clients to perform CIM operations over HTTP by using SMI-S in the back-end for EMC storage operations.

The EMC SMI-S Provider supports the SNIA Storage Management Initiative (SMI), an ANSI standard for storage management. It supports VMAX and VNX storage systems.

System requirements

EMC SMI-S Provider V4.5.1 and higher is required. You can download SMI-S from the EMC Powerlink web site. See the EMC SMI-S Provider release notes for installation instructions.

EMC storage VMAX Family and VNX Series are supported.

Supported operations

VMAX and VNX arrays support these operations:

- Create volume
- Delete volume
- Attach volume
- Detach volume
- Create snapshot
- Delete snapshot
- Create cloned volume
- Copy image to volume
- Copy volume to image

Only VNX supports these operations:

- Create volume from snapshot

Only thin provisioning is supported.

Task flow

Procedure 7.1. To set up the EMC SMI-S iSCSI driver

1. Install the python-pywbem package for your distribution. See the section called "Install the python-pywbem package" [293].
2. Download SMI-S from PowerLink and install it. Add your VNX/VMAX arrays to SMI-S.

   For information, see the section called "Set up SMI-S" [294] and the SMI-S release notes.
3. Register with VNX. See the section called "Register with VNX" [294].
4. Create a masking view on VMAX. See the section called "Create a masking view on VMAX" [295].

Install the python-pywbem package

- Install the python-pywbem package for your distribution:
  - On Ubuntu:

```
$ sudo apt-get install python-pywbem
```

- On openSUSE:

```
$ zypper install python-pywbem
```

- On Fedora:

```
$ yum install pywbem
```

Set up SMI-S

You can install SMI-S on a non-OpenStack host. Supported platforms include different flavors of Windows, Red Hat, and SUSE Linux. The host can be either a physical server or VM hosted by an ESX server. See the EMC SMI-S Provider release notes for supported platforms and installation instructions.

Note

You must discover storage arrays on the SMI-S server before you can use the Cinder driver. Follow instructions in the SMI-S release notes.

SMI-S is usually installed at `/opt/emc/ECIM/ECOM/bin` on Linux and `C:\Program Files\EMC\ECIM\ECOM\bin` on Windows. After you install and configure SMI-S, go to that directory and type **TestSmiProvider.exe**.

Use **addsys** in **TestSmiProvider.exe** to add an array. Use **dv** and examine the output after the array is added. Make sure that the arrays are recognized by the SMI-S server before using the EMC Cinder driver.

Register with VNX

To export a VNX volume to a Compute node, you must register the node with VNX.

On the Compute node `1.1.1.1`, run these commands (assume `10.10.61.35` is the iscsi target):

```
$ sudo /etc/init.d/open-iscsi start
```

```
$ sudo iscsiadm -m discovery -t st -p 10.10.61.35
```

```
$ cd /etc/iscsi
```

```
$ sudo more initiatorname.iscsi
```

```
$ iscsiadm -m node
```

Log in to VNX from the Compute node by using the target corresponding to the SPA port:

```
$ sudo iscsiadm -m node -T iqn.1992-04.com.emc:cx.apm01234567890.a0 -p 10.10.
61.35 -l
```

Assume `iqn.1993-08.org.debian:01:1a2b3c4d5f6g` is the initiator name of the Compute node. Log in to Unisphere, go to `VNX00000`->Hosts->Initiators, refresh and wait

until initiator `iqn.1993-08.org.debian:01:1a2b3c4d5f6g` with SP Port `A-8v0` appears.

Click **Register**, select **CLARiiON/VNX**, and enter the `myhost1` host name and `myhost1` IP address. Click **Register**. Now the `1.1.1.1` host appears under **Hosts Host List** as well.

Log out of VNX on the Compute node:

```
$ sudo iscsiadm -m node -u
```

Log in to VNX from the Compute node using the target corresponding to the SPB port:

```
$ sudo iscsiadm -m node -T iqn.1992-04.com.emc:cx.apm01234567890.b8 -p 10.10.
10.11 -l
```

In Unisphere, register the initiator with the SPB port.

Log out:

```
$ sudo iscsiadm -m node -u
```

Create a masking view on VMAX

For VMAX, you must set up the Unisphere for VMAX server. On the Unisphere for VMAX server, create initiator group, storage group, and port group and put them in a masking view. Initiator group contains the initiator names of the OpenStack hosts. Storage group must have at least six gatekeepers.

`cinder.conf` configuration file

Make the following changes in `/etc/cinder/cinder.conf`.

For VMAX, add the following entries, where `10.10.61.45` is the IP address of the VMAX iscsi target:

```
iscsi_target_prefix = iqn.1992-04.com.emc
iscsi_ip_address = 10.10.61.45
volume_driver = cinder.volume.drivers.emc.emc_smis_iscsi.EMCSMISISCSIDriver
cinder_emc_config_file = /etc/cinder/cinder_emc_config.xml
```

For VNX, add the following entries, where `10.10.61.35` is the IP address of the VNX iscsi target:

```
iscsi_target_prefix = iqn.2001-07.com.vnx
iscsi_ip_address = 10.10.61.35
volume_driver = cinder.volume.drivers.emc.emc_smis_iscsi.EMCSMISISCSIDriver
cinder_emc_config_file = /etc/cinder/cinder_emc_config.xml
```

Restart the `cinder-volume` service.

`cinder_emc_config.xml` configuration file

Create the file `/etc/cinder/cinder_emc_config.xml`. You do not need to restart the service for this change.

For VMAX, add the following lines to the XML file:

```
<?xml version='1.0' encoding='UTF-8'?>
<EMC>
<StorageType>xxxx</StorageType>
<MaskingView>xxxx</MaskingView>
<EcomServerIp>x.x.x.x</EcomServerIp>
<EcomServerPort>xxxx</EcomServerPort>
<EcomUserName>xxxxxxxx</EcomUserName>
<EcomPassword>xxxxxxxx</EcomPassword>
</EMC<
```

For VNX, add the following lines to the XML file:

```
<?xml version='1.0' encoding='UTF-8'?>
<EMC>
<StorageType>xxxx</StorageType>
<EcomServerIp>x.x.x.x</EcomServerIp>
<EcomServerPort>xxxx</EcomServerPort>
<EcomUserName>xxxxxxxx</EcomUserName>
<EcomPassword>xxxxxxxx</EcomPassword>
</EMC<
```

To attach VMAX volumes to an OpenStack VM, you must create a Masking View by using Unisphere for VMAX. The Masking View must have an Initiator Group that contains the initiator of the OpenStack compute node that hosts the VM.

StorageType is the thin pool where user wants to create the volume from. Only thin LUNs are supported by the plug-in. Thin pools can be created using Unisphere for VMAX and VNX.

EcomServerIp and EcomServerPort are the IP address and port number of the ECOM server which is packaged with SMI-S. EcomUserName and EcomPassword are credentials for the ECOM server.

GlusterFS driver

GlusterFS is an open-source scalable distributed filesystem that is able to grow to petabytes and beyond in size. More information can be found on Gluster's homepage.

This driver enables use of GlusterFS in a similar fashion as the NFS driver. It supports basic volume operations, and like NFS, does not support snapshot/clone.

Note

You must use a Linux kernel of version 3.4 or greater (or version 2.6.32 or greater in RHEL/CentOS 6.3+) when working with Gluster-based volumes. See Bug 1177103 for more information.

To use Cinder with GlusterFS, first set the `volume_driver` in `cinder.conf`:

```
volume_driver=cinder.volume.drivers.glusterfs.GlusterfsDriver
```

The following table contains the configuration options supported by the GlusterFS driver.

Table 7.4. Description of configuration options for storage_glusterfs

Configuration option=Default value	Description
glusterfs_disk_util=df	(StrOpt) Use du or df for free space calculation
glusterfs_mount_point_base=$state_path/mnt	(StrOpt) Base dir containing mount points for gluster shares.
glusterfs_qcow2_volumes=False	(BoolOpt) Create volumes as QCOW2 files rather than raw files.
glusterfs_shares_config=/etc/cinder/glusterfs_shares	(StrOpt) File with the list of available gluster shares
glusterfs_sparsed_volumes=True	(BoolOpt) Create volumes as sparsed files which take no space.If set to False volume is created as regular file.In such case volume creation takes a lot of time.

HDS iSCSI volume driver

This Cinder volume driver provides iSCSI support for HUS (Hitachi Unified Storage) arrays such as, HUS-110, HUS-130, and HUS-150.

System requirements

Use the HDS **hus-cmd** command to communicate with an HUS array. You can download this utility package from the HDS support site (https://HDSSupport.hds.com.

Platform: Ubuntu 12.04LTS or newer.

Supported Cinder operations

These operations are supported:

- Create volume
- Delete volume
- Attach volume
- Detach volume
- Clone volume
- Extend volume
- Create snapshot
- Delete snapshot
- Copy image to volume
- Copy volume to image
- Create volume from snapshot
- get_volume_stats

Thin provisioning, also known as Hitachi Dynamic Pool (HDP), is supported for volume or snapshot creation. Cinder volumes and snapshots do not have to reside in the same pool.

Configuration

The HDS driver supports the concept of differentiated services, where volume type can be associated with the fine-tuned performance characteristics of HDP—the the dynamic pool where volumes shall be created[1]. For instance, an HDP can consist of fast SSDs to provide speed. HDP can provide a certain reliability based on things like its RAID level characteristics. HDS driver maps volume type to the `volume_type` option in its configuration file.

Configuration is read from an XML-format file. Examples are shown for single and multi back-end cases.

Note

- Configuration is read from an XML file. This example shows the configuration for single back-end and for multi-back-end cases.
- It is not recommended to manage a HUS array simultaneously from multiple Cinder instances or servers. [2]

Table 7.5. Description of configuration options for hds

Configuration option=Default value	Description
hds_cinder_config_file=/opt/hds/hus/cinder_hus_conf.xml	(StrOpt) configuration file for HDS cinder plugin for HUS

Single back-end

In a single back-end deployment, only one Cinder instance runs on the Cinder server and controls one HUS array: this setup requires these configuration files:

1. Set the `hds_cinder_config_file` option in the `/etc/cinder/cinder.conf` file to use the HDS volume driver. This option points to a configuration file.[3]

```
volume_driver = cinder.volume.drivers.hds.hds.HUSDriver
hds_cinder_config_file = /opt/hds/hus/cinder_hds_conf.xml
```

2. Configure `hds_cinder_config_file` at the location specified previously. For example, `/opt/hds/hus/cinder_hds_conf.xml`:

```
<?xml version="1.0" encoding="UTF-8" ?>
 <config>
 <mgmt_ip0>172.17.44.16</mgmt_ip0>
 <mgmt_ip1>172.17.44.17</mgmt_ip1>
 <username>system</username>
 <password>manager</password>
 <svc_0>
 <volume_type>default</volume_type>
```

[1] Do not confuse differentiated services with the Cinder volume service.
[2] It is okay to manage multiple HUS arrays by using multiple Cinder instances (or servers).
[3] The configuration file location is not fixed.

```
<iscsi_ip>172.17.39.132</iscsi_ip>
<hdp>9</hdp>
</svc_0>
<snapshot>
<hdp>13</hdp>
</snapshot>
<lun_start>
3000
</lun_start>
<lun_end>
4000
</lun_end>
</config>
```

Multi back-end

In a multi back-end deployment, more than one Cinder instance runs on the same server. In this example, two HUS arrays are used, possibly providing different storage performance:

1. Configure `/etc/cinder/cinder.conf`: the `hus1` `hus2` configuration blocks are created. Set the `hds_cinder_config_file` option to point to an unique configuration file for each block. Set the `volume_driver` option for each back-end to `cinder.volume.drivers.hds.hds.HUSDriver`

```
enabled_backends=hus1,hus2
[hus1]
volume_driver = cinder.volume.drivers.hds.hds.HUSDriver
hds_cinder_config_file = /opt/hds/hus/cinder_hus1_conf.xml
volume_backend_name=hus-1
[hus2]
volume_driver = cinder.volume.drivers.hds.hds.HUSDriver
hds_cinder_config_file = /opt/hds/hus/cinder_hus2_conf.xml
volume_backend_name=hus-2
```

2. Configure `/opt/hds/hus/cinder_hus1_conf.xml`:

```
<?xml version="1.0" encoding="UTF-8" ?>
<config>
<mgmt_ip0>172.17.44.16</mgmt_ip0>
<mgmt_ip1>172.17.44.17</mgmt_ip1>
<username>system</username>
<password>manager</password>
<svc_0>
<volume_type>regular</volume_type>
<iscsi_ip>172.17.39.132</iscsi_ip>
<hdp>9</hdp>
</svc_0>
<snapshot>
<hdp>13</hdp>
</snapshot>
<lun_start>
3000
</lun_start>
<lun_end>
4000
</lun_end>
```

```
</config>
```

3. Configure the `/opt/hds/hus/cinder_hus2_conf.xml` file:

```
<?xml version="1.0" encoding="UTF-8" ?>
<config>
<mgmt_ip0>172.17.44.20</mgmt_ip0>
<mgmt_ip1>172.17.44.21</mgmt_ip1>
<username>system</username>
<password>manager</password>
<svc_0>
<volume_type>platinum</volume_type>
<iscsi_ip>172.17.30.130</iscsi_ip>
<hdp>2</hdp>
</svc_0>
<snapshot>
<hdp>3</hdp>
</snapshot>
<lun_start>
2000
</lun_start>
<lun_end>
3000
</lun_end>
</config>
```

Type extra specs: `volume_backend` and volume type

If you use volume types, you must configure them in the configuration file and set the `volume_backend_name` option to the appropriate back-end. In the previous multi back-end example, the `platinum` volume type is served by hus-2, and the `regular` volume type is served by hus-1.

```
cinder type-key regular set volume_backend_name=hus-1
cinder type-key platinum set volume_backend_name=hus-2
```

Non differentiated deployment of HUS arrays

You can deploy multiple Cinder instances that each control a separate HUS array. Each instance has no volume type associated with it. The Cinder filtering algorithm selects the HUS array with the largest available free space. In each configuration file, you must define the `default volume_type` in the service labels.

HDS iSCSI volume driver configuration options

These details apply to the XML format configuration file that is read by HDS volume driver. These differentiated service labels are predefined: `svc_0`, `svc_1`, `svc_2`, and `svc_3`[4]. Each respective service label associates with these parameters and tags:

1. `volume-types`: A create_volume call with a certain volume type shall be matched up with this tag. `default` is special in that any service associated with this type is used to create volume when no other labels match. Other labels are case sensitive and should

[4]There is no relative precedence or weight among these four labels.

exactly match. If no configured volume_types match the incoming requested type, an error occurs in volume creation.

2. HDP, the pool ID associated with the service.

3. An iSCSI port dedicated to the service.

Typically a Cinder volume instance has only one such service label. For example, any `svc_0`, `svc_1`, `svc_2`, or `svc_3` can be associated with it. But any mix of these service labels can be used in the same instance [5].

Table 7.6. Configuration options

Option	Type	Default	Description
`mgmt_ip0`	Required		Management Port 0 IP address
`mgmt_ip1`	Required		Management Port 1 IP address
`username`	Optional		Username is required only if secure mode is used
`password`	Optional		Password is required only if secure mode is used
`svc_0, svc_1, svc_2, svc_3`	Optional	(at least one label has to be defined)	Service labels: these four predefined names help four different sets of configuration options – each can specify iSCSI port address, HDP and an unique volume type.
`snapshot`	Required		A service label which helps specify configuration for snapshots, such as, HDP.
`volume_type`	Required		`volume_type` tag is used to match volume type. `Default` meets any type of `volume_type`, or if it is not specified. Any other volume_type is selected if exactly matched during `create_volume`.
`iscsi_ip`	Required		iSCSI port IP address where volume attaches for this volume type.
`hdp`	Required		HDP, the pool number where volume, or snapshot should be created.
`lun_start`	Optional	0	LUN allocation starts at this number.
`lun_end`	Optional	4096	LUN allocation is up to, but not including, this number.

HP 3PAR Fibre Channel and iSCSI drivers

The `HP3PARFCDriver` and `HP3PARISCSIDriver` drivers, which are based on the Block Storage Service (Cinder) plug-in architecture, run volume operations by communicating with the HP 3PAR storage system over HTTP, HTTPS, and SSH connections. The HTTP and HTTPS communications use hp3parclient, which is part of the Python standard library.

For information about how to manage HP 3PAR storage systems, see the HP 3PAR user documentation.

System requirements

To use the HP 3PAR drivers, install the following software and components on the HP 3PAR storage system:

- HP 3PAR Operating System software version 3.1.2 (MU2) or higher

[5] get_volume_stats() always provides the available capacity based on the combined sum of all the HDPs that are used in these services labels.

- HP 3PAR Web Services API Server must be enabled and running
- One Common Provisioning Group (CPG)
- Additionally, you must install the hp3parclient version 2.0 or newer from the Python standard library on the system with the enabled Block Storage Service volume drivers.

Supported operations

- Create volumes.
- Delete volumes.
- Extend volumes.
- Attach volumes.
- Detach volumes.
- Create snapshots.
- Delete snapshots.
- Create volumes from snapshots.
- Create cloned volumes.
- Copy images to volumes.
- Copy volumes to images.

Volume type support for both HP 3PAR drivers includes the ability to set the following capabilities in the OpenStack Cinder API `cinder.api.contrib.types_extra_specs` volume type extra specs extension module:

- `hp3par:cpg`
- `hp3par:snap_cpg`
- `hp3par:provisioning`
- `hp3par:persona`
- `hp3par:vvs`
- `qos:maxBWS`
- `qos:maxIOPS`

To work with the default filter scheduler, the key values are case sensitive and scoped with `hp3par:` or `qos:`. For information about how to set the key-value pairs and associate them with a volume type, run the following command:

```
$
```

```
cinder help type-key
```

Note

Volumes that are cloned only support extra specs keys cpg, snap_cpg, provisioning and vvs. The others are ignored. In addition the comments section of the cloned volume in the HP 3PAR StoreServ storage array is not populated.

The following keys require that the HP 3PAR StoreServ storage array has a Priority Optimization license installed.

- `hp3par:vvs` - The virtual volume set name that has been predefined by the Administrator with Quality of Service (QoS) rules associated to it. If you specify `hp3par:vvs`, the `qos:maxIOPS` and `qos:maxBWS` settings are ignored.
- `qos:maxBWS` - The QoS I/O issue count rate limit in MBs. If not set, the I/O issue bandwidth rate has no limit.
- `qos:maxIOPS` - The QoS I/O issue count rate limit. If not set, the I/O issue count rate has no limit.

If volume types are not used or a particular key is not set for a volume type, the following defaults are used.

- `hp3par:cpg` - Defaults to the `hp3par_cpg` setting in the `cinder.conf` file.
- `hp3par:snap_cpg` - Defaults to the `hp3par_snap` setting in the `cinder.conf` file. If `hp3par_snap` is not set, it defaults to the `hp3par_cpg` setting.
- `hp3par:provisioning` - Defaults to thin provisioning, the valid values are `thin` and `full`.
- `hp3par:persona` - Defaults to the `1 – Generic` persona. The valid values are, `1 – Generic`, `2 - Generic-ALUA`, `6 - Generic-legacy`, `7 - HPUX-legacy`, `8 - AIX-legacy`, `9 – EGENERA`, `10 - ONTAP-legacy`, `11 – VMware`, and `12 - OpenVMS`.

Enable the HP 3PAR Fibre Channel and iSCSI drivers

The `HP3PARFCDriver` and `HP3PARISCSIDriver` are installed with the OpenStack software.

1. Install the `hp3parclient` Python package on the OpenStack Block Storage system.

   ```
   $sudo pip install hp3parclient
   ```

2. Verify that the HP 3PAR Web Services API server is enabled and running on the HP 3PAR storage system.

   a. Log onto the HP 3PAR storage system with administrator access.

   ```
   #ssh 3paradm@<HP 3PAR IP Address>
   ```

   b. View the current state of the Web Services API Server.

```
#showwsapi

-Service- -State- -HTTP_State-
       HTTP_Port -HTTPS_State- HTTPS_Port -Version-

Enabled   Active Enabled       8008
       Enabled       8080         1.1
```

c. If the Web Services API Server is disabled, start it.

```
#startwsapi
```

3. If the HTTP or HTTPS state is disabled, enable one of them.

```
#setwsapi -http enable
```

or

```
#setwsapi -https enable
```

Note

To stop the Web Services API Server, use the stopwsapi command. For other options run the **setwsapi –h** command.

4. If you are not using an existing CPG, create a CPG on the HP 3PAR storage system to be used as the default location for creating volumes.

5. Make the following changes in the `/etc/cinder/cinder.conf` file.

```
                    ## REQUIRED SETTINGS
# 3PAR WS API Server URL
  hp3par_api_url=https://10.10.0.141:8080/api/v1

# 3PAR Super user username
  hp3par_username=3paradm

# 3PAR Super user password
  hp3par_password=3parpass

# 3PAR domain to use - DEPRECATED
  hp3par_domain=None

# 3PAR CPG to use for volume creation
  hp3par_cpg=OpenStackCPG_RAID5_NL

# IP address of SAN controller for SSH access to the array
  san_ip=10.10.22.241

# Username for SAN controller for SSH access to the array
  san_login=3paradm

# Password for SAN controller for SSH access to the array
  san_password=3parpass

# FIBRE CHANNEL(uncomment the next line to enable the FC driver)
  # volume_driver=cinder.volume.drivers.san.hp.hp_3par_fc.HP3PARFCDriver
```

```
# iSCSI (uncomment the next line to enable the iSCSI driver and
# hp3par_iscsi_ips or iscsi_ip_address)
  #volume_driver=cinder.volume.drivers.san.hp.hp_3par_iscsi.
HP3PARISCSIDriver

# iSCSI multiple port configuration
  # hp3par_iscsi_ips=10.10.220.253:3261,10.10.222.234

# Still available for single port iSCSI configuration
  #iscsi_ip_address=10.10.220.253
                        ## OPTIONAL SETTINGS
# Enable HTTP debugging to 3PAR
  hp3par_debug=False

# The CPG to use for Snapshots for volumes. If empty hp3par_cpg will be
 used.
  hp3par_snap_cpg=OpenStackSNAP_CPG

# Time in hours to retain a snapshot. You can't delete it before this
 expires.
  hp3par_snapshot_retention=48

# Time in hours when a snapshot expires and is deleted. This must be
 larger than retention.
  hp3par_snapshot_expiration=72
```

Note

You can enable only one driver on each cinder instance unless you enable multiple back-end support. See the Cinder multiple back-end support instructions to enable this feature.

Note

You can configure one or more iSCSI addresses by using the `hp3par_iscsi_ips` option. When you configure multiple addresses, the driver selects the iSCSI port with the fewest active volumes at attach time. The IP address might include an IP port by using a colon (`:`) to separate the address from port. If you do not define an IP port, the default port 3260 is used. Separate IP addresses with a comma (`,`). The `iscsi_ip_address/iscsi_port` options might be used as an alternative to `hp3par_iscsi_ips` for single port iSCSI configuration.

6. Save the changes to the `cinder.conf` file and restart the `cinder-volume` service.

The HP 3PAR Fibre Channel and iSCSI drivers are now enabled on your OpenStack system. If you experience problems, review the Block Storage Service log files for errors.

HP / LeftHand SAN

HP/LeftHand SANs are optimized for virtualized environments with VMware ESX & Microsoft Hyper-V, though the OpenStack integration provides additional support to various other virtualized environments, such as Xen, KVM, and OpenVZ, by exposing the volumes through ISCSI to connect to the instances.

The HpSanISCSIDriver enables you to use a HP/Lefthand SAN that supports the Cliq interface. Every supported volume operation translates into a cliq call in the back-end.

To use Cinder with HP/Lefthand SAN, you must set the following parameters in the `cinder.conf` file:

- Set *`volume_driver=cinder.volume.drivers.san.HpSanISCSIDriver`*.
- Set *`san_ip`* flag to the hostname or VIP of your Virtual Storage Appliance (VSA).
- Set *`san_login`* and *`san_password`* to the user name and password of the ssh user with all necessary privileges on the appliance.
- Set `san_ssh_port=16022`. The default is 22. However, the default for the VSA is usually 16022.
- Set `san_clustername` to the name of the cluster where the associated volumes are created.

The following optional parameters have the following default values:

- `san_thin_provision=True`. To disable thin provisioning, set to `False`.
- `san_is_local=False`. Typically, this parameter is set to `False` for this driver. To configure the cliq commands to run locally instead of over ssh, set this parameter to `True`.

In addition to configuring the `cinder-volume` service, you must configure the VSA to function in an OpenStack environment.

Procedure 7.2. To configure the VSA

1. Configure Chap on each of the `nova-compute` nodes.
2. Add Server associations on the VSA with the associated Chap and initiator information. The name should correspond to the *'hostname'* of the `nova-compute` node. For Xen, this is the hypervisor host name. To do this, use either Cliq or the Centralized Management Console.

Huawei storage driver

Huawei driver supports the iSCSI and Fibre Channel connections and enables OceanStor T series unified storage, OceanStor Dorado high-performance storage, and OceanStor HVS high-end storage to provide block storage services for OpenStack.

Supported operations

OceanStor T series unified storage supports the following operations:

- Create volume
- Delete volume
- Attach volume

- Detach volume
- Create snapshot
- Delete snapshot
- Create volume from snapshot
- Create clone volume
- Copy image to volume
- Copy volume to image

OceanStor Dorado5100 supports the following operations:

- Create volume
- Delete volume
- Attach volume
- Detach volume
- Create snapshot
- Delete snapshot
- Copy image to volume
- Copy volume to image

OceanStor Dorado2100 G2 supports the following operations:

- Create volume
- Delete volume
- Attach volume
- Detach volume
- Copy image to volume
- Copy volume to image

OceanStor HVS supports the following operations:

- Create volume
- Delete volume
- Attach volume
- Detach volume

- Create snapshot
- Delete snapshot
- Create volume from snapshot
- Create clone volume
- Copy image to volume
- Copy volume to image

Configure Cinder nodes

In `/etc/cinder`, create the driver configuration file named `cinder_huawei_conf.xml`.

You must configure `Product` and `Protocol` to specify a storage system and link type. The following uses the iSCSI driver as an example. The driver configuration file of OceanStor T series unified storage is shown as follows:

```
<?xml version='1.0' encoding='UTF-8'?>
<config>
    <Storage>
        <Product>T</Product>
        <Protocol>iSCSI</Protocol>
        <ControllerIP0>x.x.x.x</ControllerIP0>
        <ControllerIP1>x.x.x.x</ControllerIP1>
        <UserName>xxxxxxxx</UserName>
        <UserPassword>xxxxxxxx</UserPassword>
    </Storage>
    <LUN>
        <LUNType>Thick</LUNType>
        <StripUnitSize>64</StripUnitSize>
        <WriteType>1</WriteType>
        <MirrorSwitch>1</MirrorSwitch>
        <Prefetch Type="3" value="0"/>
        <StoragePool Name="xxxxxxxx"/>
        <StoragePool Name="xxxxxxxx"/>
    </LUN>
    <iSCSI>
        <DefaultTargetIP>x.x.x.x</DefaultTargetIP>
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
    </iSCSI>
    <Host OSType="Linux" HostIP="x.x.x.x, x.x.x.x"/>
</config>
```

The driver configuration file of OceanStor Dorado5100 is shown as follows:

```
<?xml version='1.0' encoding='UTF-8'?>
<config>
    <Storage>
        <Product>Dorado</Product>
        <Protocol>iSCSI</Protocol>
        <ControllerIP0>x.x.x.x</ControllerIP0>
        <ControllerIP1>x.x.x.x</ControllerIP1>
        <UserName>xxxxxxxx</UserName>
        <UserPassword>xxxxxxxx</UserPassword>
```

```
    </Storage>
    <LUN>
        <StripUnitSize>64</StripUnitSize>
        <WriteType>1</WriteType>
        <MirrorSwitch>1</MirrorSwitch>
        <StoragePool Name="xxxxxxxx"/>
        <StoragePool Name="xxxxxxxx"/>
    </LUN>
    <iSCSI>
        <DefaultTargetIP>x.x.x.x</DefaultTargetIP>
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
    </iSCSI>
    <Host OSType=”Linux” HostIP=”x.x.x.x, x.x.x.x”/>
</config>
```

The driver configuration file of OceanStor Dorado2100 G2 is shown as follows:

```
<?xml version='1.0' encoding='UTF-8'?>
<config>
    <Storage>
        <Product>Dorado</Product>
        <Protocol>iSCSI</Protocol>
        <ControllerIP0>x.x.x.x</ControllerIP0>
        <ControllerIP1>x.x.x.x</ControllerIP1>
        <UserName>xxxxxxxx</UserName>
        <UserPassword>xxxxxxxx</UserPassword>
    </Storage>
    <LUN>
        <LUNType>Thick</LUNType>
        <WriteType>1</WriteType>
        <MirrorSwitch>1</MirrorSwitch>
    </LUN>
    <iSCSI>
        <DefaultTargetIP>x.x.x.x</DefaultTargetIP>
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
    </iSCSI>
    <Host OSType=”Linux” HostIP=”x.x.x.x, x.x.x.x”/>
</config>
```

The driver configuration file of OceanStor HVS is shown as follows:

```
<?xml version='1.0' encoding='UTF-8'?>
<config>
    <Storage>
        <Product>HVS</Product>
        <Protocol>iSCSI</Protocol>
        <HVSURL>https://x.x.x.x:8088/deviceManager/rest/</HVSURL>
        <UserName>xxxxxxxx</UserName>
        <UserPassword>xxxxxxxx</UserPassword>
    </Storage>
    <LUN>
        <LUNType>Thick</LUNType>
        <WriteType>1</WriteType>
        <MirrorSwitch>1</MirrorSwitch>
        <StoragePool>xxxxxxxx</StoragePool>
    </LUN>
    <iSCSI>
        <DefaultTargetIP>x.x.x.x</DefaultTargetIP>
```

```
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
    </iSCSI>
    <Host OSType="Linux" HostIP="x.x.x.x, x.x.x.x"/>
</config>
```

Note

You do not need to configure the iSCSI target IP address for the Fibre Channel driver. In the prior example, delete the iSCSI configuration:

```
    <iSCSI>
        <DefaultTargetIP>x.x.x.x</DefaultTargetIP>
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
        <Initiator Name="xxxxxxxx" TargetIP="x.x.x.x"/>
    </iSCSI>
```

To add `volume_driver` and `cinder_huawei_conf_file` items, you can modify the `cinder.conf` configuration file as follows:

```
volume_driver = cinder.volume.drivers.huawei.HuaweiVolumeDriver
cinder_huawei_conf_file = /etc/cinder/cinder_huawei_conf.xml
```

You can configure multiple Huawei back-end storages as follows:

```
enabled_backends = t_iscsi, dorado5100_iscsi
[t_iscsi]
volume_driver = cinder.volume.drivers.huawei.HuaweiVolumeDriver
cinder_huawei_conf_file = /etc/cinder/cinder_huawei_conf_t_iscsi.xml
volume_backend_name = HuaweiTISCSIDriver
[dorado5100_iscsi]
volume_driver = cinder.volume.drivers.huawei.HuaweiVolumeDriver
cinder_huawei_conf_file = /etc/cinder/cinder_huawei_conf_dorado5100_iscsi.xml
volume_backend_name = HuaweiDorado5100ISCSIDriver
```

OceanStor HVS storage system supports the QoS function. You must create a QoS policy for the HVS storage system and create the volume type to enable QoS as follows:

```
Create volume type: QoS_high
cinder type-create QoS_high
Configure extra_specs for QoS_high:
cinder type-key QoS_high set capabilities:QoS_support="<is> True"
 drivers:flow_strategy=OpenStack_QoS_high drivers:io_priority=high
```

Note

`OpenStack_QoS_high` is a QoS policy created by a user for the HVS storage system. `QoS_high` is the self-defined volume type. Set the `io_priority` option to `high`, `normal`, or `low`.

OceanStor HVS storage system supports the SmartTier function. SmartTier has three tiers. You can create the volume type to enable SmartTier as follows:

```
Create volume type: Tier_high
cinder type-create Tier_high
Configure extra_specs for Tier_high:
cinder type-key Tier_high set capabilities:Tier_support="<is> True"
 drivers:distribute_policy=high drivers:transfer_strategy=high
```

Note

`distribute_policy` and `transfer_strategy` can only be set to `high`, `normal`, or `low`.

Configuration file details

This table describes the Huawei storage driver configuration options:

Table 7.7. Huawei storage driver configuration options

Flag name	Type	Default	Description
`Product`	Required		Type of a storage product. Valid values are `T`, `Dorado`, or `HVS`.
`Protocol`	Required		Type of a protocol. Valid values are `iSCSI` or `FC`.
`ControllerIP0`	Required		IP address of the primary controller (not required for the HVS)
`ControllerIP1`	Required		IP address of the secondary controller (not required for the HVS)
`HVSURL`	Required		Access address of the Rest port (required only for the HVS)
`UserName`	Required		User name of an administrator
`UserPassword`	Required		Password of an administrator
`LUNType`	Optional	Thin	Type of a created LUN. Valid values are `Thick` or `Thin`.
`StripUnitSize`	Optional	64	Stripe depth of a created LUN. The value is expressed in KB. **Note** This flag is not valid for a thin LUN.
`WriteType`	Optional	1	Cache write method. The method can be write back, write through, or Required write back. The default value is `1`, indicating write back.
`MirrorSwitch`	Optional	1	Cache mirroring policy. The default value is `1`, indicating that a mirroring policy is used.
`Prefetch Type`	Optional	3	Cache prefetch strategy. The strategy can be constant prefetch, variable prefetch, or intelligent prefetch. Default value is 3, which indicates intelligent prefetch and is not required for the HVS.
`Prefetch Value`	Optional	0	Cache prefetch value.
`StoragePool`	Required		Name of a storage pool that you want to use. Not required for the Dorado2100 G2.
`DefaultTargetIP`	Optional		Default IP address of the iSCSI port provided for compute nodes.
`Initiator Name`	Optional		Name of a compute node initiator.
`Initiator TargetIP`	Optional		IP address of the iSCSI port provided for Compute nodes.
`OSType`	Optional	Linux	The OS type for a Compute node.

Flag name	Type	Default	Description
`HostIP`	Optional		The IPs for Compute nodes.

Note

1. You can configure one iSCSI target port for each or all Compute nodes. The driver checks whether a target port IP address is configured for the current Compute node. If not, select `DefaultTargetIP`.
2. You can configure multiple storage pools in one configuration file, which supports the use of multiple storage pools in a storage system. (HVS allows configuration of only one storage pool.)
3. For details about LUN configuration information, see the **createlun** command in the command-line interface (CLI) documentation or run the **help -c createlun** on the storage system CLI.
4. After the driver is loaded, the storage system obtains any modification of the driver configuration file in real time and you do not need to restart the `cinder-volume` service.

IBM XIV/DS8K volume driver

There is a unified volume back-end for IBM XIV and DS8K storage. Set the following in your `cinder.conf`, and use the following options to configure it.

```
volume_driver=cinder.volume.drivers.xiv_ds8k.XIVDS8KDriver
```

Table 7.8. Description of configuration options for xiv

Configuration option=Default value	Description
xiv_ds8k_connection_type=iscsi	(StrOpt) Connection type to the IBM Storage Array (fibre_channel|iscsi)
xiv_ds8k_proxy=xiv_ds8k_openstack.nova_proxy.XIVDS8KNovaProxy	(StrOpt) Proxy driver that connects to the IBM Storage Array

IBM GPFS volume driver

IBM General Parallel File System (GPFS) is a cluster file system that provides concurrent access to file systems from multiple nodes. The storage provided by these nodes can be direct attached, network attached, SAN attached, or a combination of these methods. GPFS provides many features beyond common data access, including data replication, policy based storage management, and space efficient file snapshot and clone operations.

How the GPFS driver works

The GPFS driver enables the use of GPFS in a fashion similar to that of the NFS driver. With the GPFS driver, instances do not actually access a storage device at the block level. Instead, volume backing files are created in a GPFS file system and mapped to instances, which emulate a block device.

Note

GPFS software must be installed and running on nodes where Block Storage and Compute services run in the OpenStack environment. A GPFS file system must also be created and mounted on these nodes before starting the `cinder-volume` service. The details of these GPFS specific steps are covered in *GPFS: Concepts, Planning, and Installation Guide* and *GPFS: Administration and Programming Reference*.

Optionally, the Image Service can be configured to store images on a GPFS file system. When a Block Storage volume is created from an image, if both image data and volume data reside in the same GPFS file system, the data from image file is moved efficiently to the volume file using copy-on-write optimization strategy.

Enable the GPFS driver

To use the Block Storage Service with the GPFS driver, first set the `volume_driver` in `cinder.conf`:

```
volume_driver = cinder.volume.drivers.gpfs.GPFSDriver
```

The following table contains the configuration options supported by the GPFS driver.

Table 7.9. Description of configuration options for storage_gpfs

Configuration option=Default value	Description
gpfs_images_dir=None	(StrOpt) Specifies the path of the Image service repository in GPFS. Leave undefined if not storing images in GPFS.
gpfs_images_share_mode=None	(StrOpt) Specifies the type of image copy to be used. Set this when the Image service repository also uses GPFS so that image files can be transferred efficiently from the Image service to the Block Storage service. There are two valid values: "copy" specifies that a full copy of the image is made; "copy_on_write" specifies that copy-on-write optimization strategy is used and unmodified blocks of the image file are shared efficiently.
gpfs_max_clone_depth=0	(IntOpt) Specifies an upper limit on the number of indirections required to reach a specific block due to snapshots or clones. A lengthy chain of copy-on-write snapshots or clones can have a negative impact on performance, but improves space utilization. 0 indicates unlimited clone depth.
gpfs_mount_point_base=None	(StrOpt) Specifies the path of the GPFS directory where Block Storage volume and snapshot files are stored.
gpfs_sparse_volumes=True	(BoolOpt) Specifies that volumes are created as sparse files which initially consume no space. If set to False, the volume is created as a fully allocated file, in which case, creation may take a significantly longer time.

Note

The `gpfs_images_share_mode` flag is only valid if the Image Service is configured to use GPFS with the `gpfs_images_dir` flag. When the value of this flag is `copy_on_write`, the paths specified by the `gpfs_mount_point_base` and `gpfs_images_dir` flags must both reside in the same GPFS file system and in the same GPFS file set.

Volume creation options

It is possible to specify additional volume configuration options on a per-volume basis by specifying volume metadata. The volume is created using the specified options. Changing the metadata after the volume is created has no effect. The following table lists the volume creation options supported by the GPFS volume driver.

Table 7.10. Volume Create Options for GPFS Volume Drive

Metadata Item Name	Description
`fstype`	Specifies whether to create a file system or a swap area on the new volume. If `fstype=swap` is specified, the mkswap command is used to create a swap area. Otherwise the mkfs command is passed the specified file system type, for example ext3, ext4 or ntfs.
`fslabel`	Sets the file system label for the file system specified by `fstype` option. This value is only used if `fstype` is specified.
`data_pool_name`	Specifies the GPFS storage pool to which the volume is to be assigned. Note: The GPFS storage pool must already have been created.
`replicas`	Specifies how many copies of the volume file to create. Valid values are 1, 2, and, for GPFS V3.5.0.7 and later, 3. This value cannot be greater than the value of the `MaxDataReplicas` attribute of the file system.
`dio`	Enables or disables the Direct I/O caching policy for the volume file. Valid values are `yes` and `no`.
`write_affinity_depth`	Specifies the allocation policy to be used for the volume file. Note: This option only works if `allow-write-affinity` is set for the GPFS data pool.
`block_group_factor`	Specifies how many blocks are laid out sequentially in the volume file to behave as a single large block. Note: This option only works if `allow-write-affinity` is set for the GPFS data pool.
`write_affinity_failure_group`	Specifies the range of nodes (in GPFS shared nothing architecture) where replicas of blocks in the volume file are to be written. See *GPFS: Administration and Programming Reference* for more details on this option.

Example: Volume creation options

This example shows the creation of a 50GB volume with an ext4 file system labeled `newfs`and direct IO enabled:

```
$cinder create --metadata fstype=ext4 fslabel=newfs dio=yes --display-name
 volume_1 50
```

Operational notes for GPFS driver

Snapshots and clones

Volume snapshots are implemented using the GPFS file clone feature. Whenever a new snapshot is created, the snapshot file is efficiently created as a read-only clone parent of the volume, and the volume file uses copy-on-write optimization strategy to minimize data movement.

Similarly when a new volume is created from a snapshot or from an existing volume, the same approach is taken. The same approach is also used when a new volume is created from a Glance image, if the source image is in raw format, and `gpfs_images_share_mode` is set to `copy_on_write`.

IBM Storwize family and SVC volume driver

The volume management driver for Storwize family and SAN Volume Controller (SVC) provides OpenStack Compute instances with access to IBM Storwize family or SVC storage systems.

Configure the Storwize family and SVC system

Network configuration

The Storwize family or SVC system must be configured for iSCSI, Fibre Channel, or both.

If using iSCSI, each Storwize family or SVC node should have at least one iSCSI IP address. The IBM Storwize/SVC driver uses an iSCSI IP address associated with the volume's preferred node (if available) to attach the volume to the instance, otherwise it uses the first available iSCSI IP address of the system. The driver obtains the iSCSI IP address directly from the storage system; there is no need to provide these iSCSI IP addresses directly to the driver.

Note

If using iSCSI, ensure that the compute nodes have iSCSI network access to the Storwize family or SVC system.

Note

OpenStack Nova's Grizzly version supports iSCSI multipath. Once this is configured on the Nova host (outside the scope of this documentation), multipath is enabled.

If using Fibre Channel (FC), each Storwize family or SVC node should have at least one WWPN port configured. If the `storwize_svc_multipath_enabled` flag is set to True in the Cinder configuration file, the driver uses all available WWPNs to attach the volume to the instance (details about the configuration flags appear in the next section). If the flag is not set, the driver uses the WWPN associated with the volume's preferred node (if available), otherwise it uses the first available WWPN of the system. The driver obtains the WWPNs directly from the storage system; there is no need to provide these WWPNs directly to the driver.

Note

If using FC, ensure that the compute nodes have FC connectivity to the Storwize family or SVC system.

iSCSI CHAP authentication

If using iSCSI for data access and the `storwize_svc_iscsi_chap_enabled` is set to `True`, the driver will associate randomly-generated CHAP secrets with all hosts on the

Storwize family system. OpenStack compute nodes use these secrets when creating iSCSI connections.

Note

CHAP secrets are added to existing hosts as well as newly-created ones. If the CHAP option is enabled, hosts will not be able to access the storage without the generated secrets.

Note

Not all OpenStack Compute drivers support CHAP authentication. Please check compatibility before using.

Note

CHAP secrets are passed from OpenStack Block Storage to Compute in clear text. This communication should be secured to ensure that CHAP secrets are not discovered.

Configure storage pools

Each instance of the IBM Storwize/SVC driver allocates all volumes in a single pool. The pool should be created in advance and be provided to the driver using the `storwize_svc_volpool_name` configuration flag. Details about the configuration flags and how to provide the flags to the driver appear in the next section.

Configure user authentication for the driver

The driver requires access to the Storwize family or SVC system management interface. The driver communicates with the management using SSH. The driver should be provided with the Storwize family or SVC management IP using the `san_ip` flag, and the management port should be provided by the `san_ssh_port` flag. By default, the port value is configured to be port 22 (SSH).

Note

Make sure the compute node running the nova-volume management driver has SSH network access to the storage system.

To allow the driver to communicate with the Storwize family or SVC system, you must provide the driver with a user on the storage system. The driver has two authentication methods: password-based authentication and SSH key pair authentication. The user should have an Administrator role. It is suggested to create a new user for the management driver. Please consult with your storage and security administrator regarding the preferred authentication method and how passwords or SSH keys should be stored in a secure manner.

Note

When creating a new user on the Storwize or SVC system, make sure the user belongs to the Administrator group or to another group that has an Administrator role.

If using password authentication, assign a password to the user on the Storwize or SVC system. The driver configuration flags for the user and password are `san_login` and `san_password`, respectively.

If you are using the SSH key pair authentication, create SSH private and public keys using the instructions below or by any other method. Associate the public key with the user by uploading the public key: select the "choose file" option in the Storwize family or SVC management GUI under "SSH public key". Alternatively, you may associate the SSH public key using the command line interface; details can be found in the Storwize and SVC documentation. The private key should be provided to the driver using the `san_private_key` configuration flag.

Create a SSH key pair with OpenSSH

You can create an SSH key pair using OpenSSH, by running:

```
$ ssh-keygen -t rsa
```

The command prompts for a file to save the key pair. For example, if you select 'key' as the filename, two files are created: `key` and `key.pub`. The `key` file holds the private SSH key and `key.pub` holds the public SSH key.

The command also prompts for a pass phrase, which should be empty.

The private key file should be provided to the driver using the `san_private_key` configuration flag. The public key should be uploaded to the Storwize family or SVC system using the storage management GUI or command line interface.

Note

Ensure that Cinder has read permissions on the private key file.

Configure the Storwize family and SVC driver

Enable the Storwize family and SVC driver

Set the volume driver to the Storwize family and SVC driver by setting the `volume_driver` option in `cinder.conf` as follows:

```
volume_driver = cinder.volume.drivers.storwize_svc.StorwizeSVCDriver
```

Storwize family and SVC driver options in cinder.conf

The following options specify default values for all volumes. Some can be over-ridden using volume types, which are described below.

Table 7.11. List of configuration flags for Storwize storage and SVC driver

Flag name	Type	Default	Description
`san_ip`	Required		Management IP or host name
`san_ssh_port`	Optional	22	Management port
`san_login`	Required		Management login username
`san_password`	Required [a]		Management login password

Flag name	Type	Default	Description
`san_private_key`	Required [a]		Management login SSH private key
`storwize_svc_volpool_name`	Required		Default pool name for volumes
`storwize_svc_vol_rsize`	Optional	2	Initial physical allocation (percentage) [b]
`storwize_svc_vol_warning`	Optional	0 (disabled)	Space allocation warning threshold (percentage) [b]
`storwize_svc_vol_autoexpand`	Optional	True	Enable or disable volume auto expand [c]
`storwize_svc_vol_grainsize`	Optional	256	Volume grain size [b] in KB
`storwize_svc_vol_compression`	Optional	False	Enable or disable Real-time Compression [d]
`storwize_svc_vol_easytier`	Optional	True	Enable or disable Easy Tier [e]
`storwize_svc_vol_iogrp`	Optional	0	The I/O group in which to allocate vdisks
`storwize_svc_flashcopy_timeout`	Optional	120	FlashCopy timeout threshold [f] (seconds)
`storwize_svc_connection_protocol`	Optional	iSCSI	Connection protocol to use (currently supports 'iSCSI' or 'FC')
`storwize_svc_iscsi_chap_enabled`	Optional	True	Configure CHAP authentication for iSCSI connections
`storwize_svc_multipath_enabled`	Optional	False	Enable multipath for FC connections [g]
`storwize_svc_multihost_enabled`	Optional	True	Enable mapping vdisks to multiple hosts [h]

[a]The authentication requires either a password (`san_password`) or SSH private key (`san_private_key`). One must be specified. If both are specified, the driver uses only the SSH private key.

[b]The driver creates thin-provisioned volumes by default. The `storwize_svc_vol_rsize` flag defines the initial physical allocation percentage for thin-provisioned volumes, or if set to `-1`, the driver creates full allocated volumes. More details about the available options are available in the Storwize family and SVC documentation.

[c]Defines whether thin-provisioned volumes can be auto expanded by the storage system, a value of `True` means that auto expansion is enabled, a value of `False` disables auto expansion. Details about this option can be found in the `–autoexpand` flag of the Storwize family and SVC command line interface `mkvdisk` command.

[d]Defines whether Real-time Compression is used for the volumes created with OpenStack. Details on Real-time Compression can be found in the Storwize family and SVC documentation. The Storwize or SVC system must have compression enabled for this feature to work.

[e]Defines whether Easy Tier is used for the volumes created with OpenStack. Details on EasyTier can be found in the Storwize family and SVC documentation. The Storwize or SVC system must have Easy Tier enabled for this feature to work.

[f]The driver wait timeout threshold when creating an OpenStack snapshot. This is actually the maximum amount of time that the driver waits for the Storwize family or SVC system to prepare a new FlashCopy mapping. The driver accepts a maximum wait time of 600 seconds (10 minutes).

[g]Multipath for iSCSI connections requires no storage-side configuration and is enabled if the compute host has multipath configured.

[h]This option allows the driver to map a vdisk to more than one host at a time. This scenario occurs during migration of a virtual machine with an attached volume; the volume is simultaneously mapped to both the source and destination compute hosts. If your deployment does not require attaching vdisks to multiple hosts, setting this flag to False will provide added safety.

Table 7.12. Description of configuration options for storwize

Configuration option=Default value	Description
storwize_svc_connection_protocol=iSCSI	(StrOpt) Connection protocol (iSCSI/FC)
storwize_svc_flashcopy_timeout=120	(IntOpt) Maximum number of seconds to wait for FlashCopy to be prepared. Maximum value is 600 seconds (10 minutes)
storwize_svc_iscsi_chap_enabled=True	(BoolOpt) Configure CHAP authentication for iSCSI connections (Default: Enabled)
storwize_svc_multihostmap_enabled=True	(BoolOpt) Allows vdisk to multi host mapping
storwize_svc_multipath_enabled=False	(BoolOpt) Connect with multipath (FC only; iSCSI multipath is controlled by Nova)
storwize_svc_vol_autoexpand=True	(BoolOpt) Storage system autoexpand parameter for volumes (True/False)
storwize_svc_vol_compression=False	(BoolOpt) Storage system compression option for volumes

Configuration option=Default value	Description
storwize_svc_vol_easytier=True	(BoolOpt) Enable Easy Tier for volumes
storwize_svc_vol_grainsize=256	(IntOpt) Storage system grain size parameter for volumes (32/64/128/256)
storwize_svc_vol_iogrp=0	(IntOpt) The I/O group in which to allocate volumes
storwize_svc_volpool_name=volpool	(StrOpt) Storage system storage pool for volumes
storwize_svc_vol_rsize=2	(IntOpt) Storage system space-efficiency parameter for volumes (percentage)
storwize_svc_vol_warning=0	(IntOpt) Storage system threshold for volume capacity warnings (percentage)

Placement with volume types

The IBM Storwize/SVC driver exposes capabilities that can be added to the `extra specs` of volume types, and used by the filter scheduler to determine placement of new volumes. Make sure to prefix these keys with `capabilities:` to indicate that the scheduler should use them. The following `extra specs` are supported:

- capabilities:volume_back-end_name - Specify a specific back-end where the volume should be created. The back-end name is a concatenation of the name of the IBM Storwize/SVC storage system as shown in `lssystem`, an underscore, and the name of the pool (mdisk group). For example:

```
capabilities:volume_back-end_name=myV7000_openstackpool
```

- capabilities:compression_support - Specify a back-end according to compression support. A value of `True` should be used to request a back-end that supports compression, and a value of `False` will request a back-end that does not support compression. If you do not have constraints on compression support, do not set this key. Note that specifying `True` does not enable compression; it only requests that the volume be placed on a back-end that supports compression. Example syntax:

```
capabilities:compression_support='<is> True'
```

- capabilities:easytier_support - Similar semantics as the `compression_support` key, but for specifying according to support of the Easy Tier feature. Example syntax:

```
capabilities:easytier_support='<is> True'
```

- capabilities:storage_protocol - Specifies the connection protocol used to attach volumes of this type to instances. Legal values are `iSCSI` and `FC`. This `extra specs` value is used for both placement and setting the protocol used for this volume. In the example syntax, note <in> is used as opposed to <is> used in the previous examples.

```
capabilities:storage_protocol='<in> FC'
```

Configure per-volume creation options

Volume types can also be used to pass options to the IBM Storwize/SVC driver, which override the default values set in the configuration file. Contrary to the previous examples where the "capabilities" scope was used to pass parameters to the Cinder scheduler, options can be passed to the IBM Storwize/SVC driver with the "drivers" scope.

The following `extra specs` keys are supported by the IBM Storwize/SVC driver:

- rsize
- warning
- autoexpand
- grainsize
- compression
- easytier
- multipath
- iogrp

These keys have the same semantics as their counterparts in the configuration file. They are set similarly; for example, `rsize=2` or `compression=False`.

Example: Volume types

In the following example, we create a volume type to specify a controller that supports iSCSI and compression, to use iSCSI when attaching the volume, and to enable compression:

```
$ cinder type-create compressed
$ cinder type-key compressed set capabilities:storage_protocol='<in> iSCSI'
 capabilities:compression_support='<is> True' drivers:compression=True
```

We can then create a 50GB volume using this type:

```
$ cinder create --display-name "compressed volume" --volume-type compressed 50
```

Volume types can be used, for example, to provide users with different

- performance levels (such as, allocating entirely on an HDD tier, using Easy Tier for an HDD-SDD mix, or allocating entirely on an SSD tier)
- resiliency levels (such as, allocating volumes in pools with different RAID levels)
- features (such as, enabling/disabling Real-time Compression)

Operational notes for the Storwize family and SVC driver

Migrate volumes

In the context of OpenStack Block Storage's volume migration feature, the IBM Storwize/SVC driver enables the storage's virtualization technology. When migrating a volume from one pool to another, the volume will appear in the destination pool almost immediately, while the storage moves the data in the background.

Note

To enable this feature, both pools involved in a given volume migration must have the same values for `extent_size`. If the pools have different values for `extent_size`, the data will still be moved directly between the pools (not host-side copy), but the operation will be synchronous.

Extend volumes

The IBM Storwize/SVC driver allows for extending a volume's size, but only for volumes without snapshots.

Snapshots and clones

Snapshots are implemented using FlashCopy with no background copy (space-efficient). Volume clones (volumes created from existing volumes) are implemented with FlashCopy, but with background copy enabled. This means that volume clones are independent, full copies. While this background copy is taking place, attempting to delete or extend the source volume will result in that operation waiting for the copy to complete.

NetApp unified driver

NetApp unified driver is a block storage driver that supports multiple storage families and storage protocols. The storage family corresponds to storage systems built on different technologies like 7-Mode and clustered Data ONTAP®. The storage protocol refers to the protocol used to initiate data storage and access operations on those storage systems like iSCSI and NFS. NetApp unified driver can be configured to provision and manage OpenStack volumes on a given storage family for the specified storage protocol. The OpenStack volumes can then be used for accessing and storing data using the storage protocol on the storage family system. NetApp unified driver is an extensible interface that can support new storage families and storage protocols.

NetApp clustered Data ONTAP storage family

The NetApp clustered Data ONTAP storage family represents a configuration group which provides OpenStack compute instances access to clustered Data ONTAP storage systems. At present it can be configured in cinder to work with iSCSI and NFS storage protocols.

NetApp iSCSI configuration for clustered Data ONTAP

The NetApp iSCSI configuration for clustered Data ONTAP is an interface from OpenStack to clustered Data ONTAP storage systems for provisioning and managing the SAN block storage entity, that is, NetApp LUN which can be accessed using iSCSI protocol.

The iSCSI configuration for clustered Data ONTAP is a direct interface from OpenStack to clustered Data ONTAP and it does not require additional management software to achieve the desired functionality. It uses NetApp APIs to interact with the clustered Data ONTAP.

Configuration options for clustered Data ONTAP family with iSCSI protocol

Set the volume driver, storage family and storage protocol to NetApp unified driver, clustered Data ONTAP and iSCSI respectively by setting the `volume_driver`, `netapp_storage_family` and `netapp_storage_protocol` options in `cinder.conf` as follows:

```
volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
netapp_storage_family=ontap_cluster
netapp_storage_protocol=iscsi
```

See OpenStack NetApp community for detailed information on available configuration options.

NetApp NFS configuration for clustered Data ONTAP

The NetApp NFS configuration for clustered Data ONTAP is an interface from OpenStack to clustered Data ONTAP system for provisioning and managing OpenStack volumes on NFS exports provided by the clustered Data ONTAP system which can then be accessed using NFS protocol.

The NFS configuration for clustered Data ONTAP does not require any additional management software to achieve the desired functionality. It uses NetApp APIs to interact with the clustered Data ONTAP.

Configuration options for the clustered Data ONTAP family with NFS protocol

Set the volume driver, storage family and storage protocol to NetApp unified driver, clustered Data ONTAP and NFS respectively by setting the `volume_driver`, `netapp_storage_family` and `netapp_storage_protocol` options in `cinder.conf` as follows:

```
volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
netapp_storage_family=ontap_cluster
netapp_storage_protocol=nfs
```

See OpenStack NetApp community for detailed information on available configuration options.

NetApp 7-Mode Data ONTAP storage family

The NetApp 7-Mode Data ONTAP storage family represents a configuration group which provides OpenStack compute instances access to 7-Mode storage systems. At present it can be configured in cinder to work with iSCSI and NFS storage protocols.

NetApp iSCSI configuration for 7-Mode storage controller

The NetApp iSCSI configuration for 7-Mode Data ONTAP is an interface from OpenStack to 7-Mode storage systems for provisioning and managing the SAN block storage entity, that is, NetApp LUN which can be accessed using iSCSI protocol.

The iSCSI configuration for 7-Mode Data ONTAP is a direct interface from OpenStack to 7-Mode storage system and it does not require additional management software to achieve the desired functionality. It uses NetApp APIs to interact with the 7-Mode storage system.

Configuration options for the 7-Mode Data ONTAP storage family with iSCSI protocol

Set the volume driver, storage family and storage protocol to NetApp unified driver, 7-Mode Data ONTAP and iSCSI respectively by setting the `volume_driver`, `netapp_storage_family` and `netapp_storage_protocol` options in `cinder.conf` as follows:

```
volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
netapp_storage_family=ontap_7mode
netapp_storage_protocol=iscsi
```

See OpenStack NetApp community for detailed information on available configuration options.

NetApp NFS configuration for 7-Mode Data ONTAP

The NetApp NFS configuration for 7-Mode Data ONTAP is an interface from OpenStack to 7-Mode storage system for provisioning and managing OpenStack volumes on NFS exports provided by the 7-Mode storage system which can then be accessed using NFS protocol.

The NFS configuration for 7-Mode Data ONTAP does not require any additional management software to achieve the desired functionality. It uses NetApp APIs to interact with the 7-Mode storage system.

Configuration options for the 7-Mode Data ONTAP family with NFS protocol

Set the volume driver, storage family and storage protocol to NetApp unified driver, 7-Mode Data ONTAP and NFS respectively by setting the `volume_driver`, `netapp_storage_family` and `netapp_storage_protocol` options in `cinder.conf` as follows:

```
volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
netapp_storage_family=ontap_7mode
netapp_storage_protocol=nfs
```

See OpenStack NetApp community for detailed information on available configuration options.

Driver options

Table 7.13. Description of configuration options for netapp

Configuration option=Default value	Description
expiry_thres_minutes=720	(IntOpt) Threshold minutes after which cache file can be cleaned.
netapp_login=None	(StrOpt) Login user name for the storage 7-Mode controller/clustered Data ONTAP management.
netapp_password=None	(StrOpt) Login password for the 7-Mode controller/ clustered Data ONTAP management.
netapp_server_hostname=None	(StrOpt) The management IP address for the 7-Mode controller or clustered Data ONTAP.
netapp_server_port=80	(IntOpt) The 7-Mode controller/Clustered ONTAP data port to use for communication. As a custom, 80 is used for HTTP and 443 is used for HTTPS communication. The defaults should be changed if other ports are used for ONTAPI.
netapp_size_multiplier=1.2	(FloatOpt) When creating volumes, the quantity to be multiplied to the requested OpenStack volume size to ensure enough space is available on the 7-Mode Controller/Clustered Data ONTAP Vserver.

Configuration option=Default value	Description
netapp_storage_family=ontap_cluster	(StrOpt) Storage family type. Valid values are ontap_7mode for using a 7-Mode controller or ontap_cluster for a Clustered Data ONTAP.
netapp_storage_protocol=None	(StrOpt) The storage protocol to be used. Valid options are nfs or iscsi, but we recommended that you consult the detailed explanation. If None is selected, nfs will be used.
netapp_transport_type=http	(StrOpt) Transport protocol for communicating with the 7-Mode controller or Clustered Data ONTAP. Supported protocols include http and https.
netapp_vfiler=None	(StrOpt) The vFiler unit to be use for provisioning OpenStack Volumes. Use this only if using MultiStore®.
netapp_volume_list=None	(StrOpt) Comma separated list of NetApp volumes to be used for provisioning on 7-Mode controller. This option is used to restrict provisioning to the specified NetApp controller volumes. In case this option is not specified all NetApp controller volumes except the controller root volume are used for provisioning OpenStack volumes.
netapp_vserver=None	(StrOpt) The Vserver on the cluster on which provisioning of OpenStack volumes occurs. If using netapp_storage_protocol=nfs, it is a mandatory parameter for storage service catalog support. If specified the exports belonging to the vserver will only be used for provisioning in future. OpenStack volumes on exports not belonging to the vserver will continue to function in a normal manner and recieve Block Storage operations like snapshot creation etc.
thres_avl_size_perc_start=20	(IntOpt) Threshold available percent to start cache cleaning.
thres_avl_size_perc_stop=60	(IntOpt) Threshold available percent to stop cache cleaning.

Upgrading NetApp drivers to Havana

NetApp has introduced a new unified driver in Havana for configuring different storage families and storage protocols. This requires defining upgrade path for NetApp drivers which existed in a previous release like Grizzly. This section covers the upgrade configuration for NetApp drivers and lists deprecated NetApp drivers.

Upgraded NetApp drivers

This section shows upgrade configuration in Havana for NetApp drivers in Grizzly.

Driver upgrade configuration

1. NetApp iSCSI direct driver for clustered Data ONTAP in Grizzly

```
 volume_driver=cinder.volume.drivers.netapp.iscsi.
NetAppDirectCmodeISCSIDriver
```

NetApp Unified Driver configuration

```
 volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
 netapp_storage_family=ontap_cluster
 netapp_storage_protocol=iscsi
```

2. NetApp NFS direct driver for clustered Data ONTAP in Grizzly

```
volume_driver=cinder.volume.drivers.netapp.nfs.NetAppDirectCmodeNfsDriver
```

NetApp Unified Driver configuration

```
volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
netapp_storage_family=ontap_cluster
netapp_storage_protocol=nfs
```

3. NetApp iSCSI direct driver for 7-Mode storage controller in Grizzly

```
volume_driver=cinder.volume.drivers.netapp.iscsi.
NetAppDirect7modeISCSIDriver
```

NetApp Unified Driver configuration

```
volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
netapp_storage_family=ontap_7mode
netapp_storage_protocol=iscsi
```

4. NetApp NFS direct driver for 7-Mode storage controller in Grizzly

```
volume_driver=cinder.volume.drivers.netapp.nfs.NetAppDirect7modeNfsDriver
```

NetApp Unified Driver configuration

```
volume_driver=cinder.volume.drivers.netapp.common.NetAppDriver
netapp_storage_family=ontap_7mode
netapp_storage_protocol=nfs
```

Deprecated NetApp drivers

This section lists the NetApp drivers in Grizzly that are deprecated in Havana.

1. NetApp iSCSI driver for clustered Data ONTAP.

```
volume_driver=cinder.volume.drivers.netapp.iscsi.NetAppCmodeISCSIDriver
```

2. NetApp NFS driver for clustered Data ONTAP.

```
volume_driver=cinder.volume.drivers.netapp.nfs.NetAppCmodeNfsDriver
```

3. NetApp iSCSI driver for 7-Mode storage controller.

```
volume_driver=cinder.volume.drivers.netapp.iscsi.NetAppISCSIDriver
```

4. NetApp NFS driver for 7-Mode storage controller.

```
volume_driver=cinder.volume.drivers.netapp.nfs.NetAppNFSDriver
```

Note

See OpenStack NetApp community for information on supporting deprecated NetApp drivers in Havana.

Nexenta drivers

NexentaStor Appliance is NAS/SAN software platform designed for building reliable and fast network storage arrays. The Nexenta Storage Appliance uses ZFS as a disk management system. NexentaStor can serve as a storage node for the OpenStack and for the virtual servers through iSCSI and NFS protocols.

With the NFS option, every Compute volume is represented by a directory designated to be its own file system in the ZFS file system. These file systems are exported using NFS.

With either option some minimal setup is required to tell OpenStack which NexentaStor servers are being used, whether they are supporting iSCSI and/or NFS and how to access each of the servers.

Typically the only operation required on the NexentaStor servers is to create the containing directory for the iSCSI or NFS exports. For NFS this containing directory must be explicitly exported via NFS. There is no software that must be installed on the NexentaStor servers; they are controlled using existing management plane interfaces.

Nexenta iSCSI driver

The Nexenta iSCSI driver allows you to use NexentaStor appliance to store Compute volumes. Every Compute volume is represented by a single zvol in a predefined Nexenta namespace. For every new volume the driver creates a iSCSI target and iSCSI target group that are used to access it from compute hosts.

The Nexenta iSCSI volume driver should work with all versions of NexentaStor. The NexentaStor appliance must be installed and configured according to the relevant Nexenta documentation. A pool and an enclosing namespace must be created for all iSCSI volumes to be accessed through the volume driver. This should be done as specified in the release specific NexentaStor documentation.

The NexentaStor Appliance iSCSI driver is selected using the normal procedures for one or multiple back-end volume drivers. The following items will need to be configured for each NexentaStor appliance that the iSCSI volume driver will control:

Enable the Nexenta iSCSI driver and related options

The following table contains the options supported by the Nexenta iSCSI driver.

Table 7.14. Description of configuration options for storage_nexenta_iscsi

Configuration option=Default value	Description
nexenta_blocksize=	(StrOpt) block size for volumes (blank=default,8KB)
nexenta_host=	(StrOpt) IP address of Nexenta SA
nexenta_iscsi_target_portal_port=3260	(IntOpt) Nexenta target portal port
nexenta_password=nexenta	(StrOpt) Password to connect to Nexenta SA
nexenta_rest_port=2000	(IntOpt) HTTP port to connect to Nexenta REST API server
nexenta_rest_protocol=auto	(StrOpt) Use http or https for REST connection (default auto)
nexenta_sparse=False	(BoolOpt) flag to create sparse volumes
nexenta_target_group_prefix=cinder/	(StrOpt) prefix for iSCSI target groups on SA
nexenta_target_prefix=iqn.1986-03.com.sun:02:cinder-	(StrOpt) IQN prefix for iSCSI targets
nexenta_user=admin	(StrOpt) User name to connect to Nexenta SA
nexenta_volume=cinder	(StrOpt) pool on SA that will hold all volumes

To use Compute with the Nexenta iSCSI driver, first set the `volume_driver`:

```
volume_driver=cinder.volume.drivers.nexenta.iscsi.NexentaISCSIDriver
```

Then set value for `nexenta_host` and other parameters from table if needed.

Nexenta NFS driver

The Nexenta NFS driver allows you to use NexentaStor appliance to store Compute volumes via NFS. Every Compute volume is represented by a single NFS file within a shared directory.

While the NFS protocols standardize file access for users, they do not standardize administrative actions such as taking snapshots or replicating file systems. The Openstack Volume Drivers bring a common interface to these operations. The Nexenta NFS driver implements these standard actions using the ZFS management plane that already is deployed on NexentaStor appliances.

The Nexenta NFS volume driver should work with all versions of NexentaStor. The NexentaStor appliance must be installed and configured according to the relevant Nexenta documentation. A single parent file system must be created for all virtual disk directories supported for OpenStack. This directory must be created and exported on each NexentaStor appliance. This should be done as specified in the release specific NexentaStor documentation.

Enable the Nexenta NFS driver and related options

To use Compute with the Nexenta NFS driver, first set the `volume_driver`:

```
volume_driver = cinder.volume.drivers.nexenta.nfs.NexentaNfsDriver
```

The following table contains the options supported by the Nexenta NFS driver.

Table 7.15. Description of configuration options for storage_nexenta_nfs

Configuration option=Default value	Description
nexenta_mount_options=None	(StrOpt) Mount options passed to the nfs client. See section of the nfs man page for details

Configuration option=Default value	Description
nexenta_mount_point_base=$state_path/mnt	(StrOpt) Base dir containing mount points for nfs shares
nexenta_oversub_ratio=1.0	(FloatOpt) This will compare the allocated to available space on the volume destination. If the ratio exceeds this number, the destination will no longer be valid.
nexenta_shares_config=/etc/cinder/nfs_shares	(StrOpt) File with the list of available nfs shares
nexenta_sparsed_volumes=True	(BoolOpt) Create volumes as sparsed files which take no space.If set to False volume is created as regular file.In such case volume creation takes a lot of time.
nexenta_used_ratio=0.95	(FloatOpt) Percent of ACTUAL usage of the underlying volume before no new volumes can be allocated to the volume destination.
nexenta_volume_compression=on	(StrOpt) Default compression value for new ZFS folders.

Add your list of Nexenta NFS servers to the file you specified with the `nexenta_shares_config` option. For example, if the value of this option was set to `/etc/cinder/nfs_shares`, then:

```
# cat /etc/cinder/nfs_shares
192.168.1.200:/storage http://admin:nexenta@192.168.1.200:2000
192.168.1.201:/storage http://admin:nexenta@192.168.1.201:2000
192.168.1.202:/storage http://admin:nexenta@192.168.1.202:2000
```

Comments are allowed in this file. They begin with a #.

Each line in this file represents a NFS share. The first part of the line is the NFS share URL, the second is the connection URL to the NexentaStor Appliance.

NFS driver

The Network File System (NFS) is a distributed file system protocol originally developed by Sun Microsystems in 1984. An NFS server *exports* one or more of its file systems, known as *shares*. An NFS client can mount these exported shares on its own file system. You can perform file actions on this mounted remote file system as if the file system were local.

How the NFS driver works

The NFS driver, and other drivers based off of it, work quite differently than a traditional block storage driver.

The NFS driver does not actually allow an instance to access a storage device at the block level. Instead, files are created on an NFS share and mapped to instances, which emulates a block device. This works in a similar way to QEMU, which stores instances in the `/var/lib/nova/instances` directory.

Enable the NFS driver and related options

To use Cinder with the NFS driver, first set the `volume_driver` in `cinder.conf`:

```
volume_driver=cinder.volume.drivers.nfs.NfsDriver
```

The following table contains the options supported by the NFS driver.

Table 7.16. Description of configuration options for storage_nfs

Configuration option=Default value	Description
nfs_mount_options=None	(StrOpt) Mount options passed to the nfs client. See section of the nfs man page for details.
nfs_mount_point_base=$state_path/mnt	(StrOpt) Base dir containing mount points for nfs shares.
nfs_oversub_ratio=1.0	(FloatOpt) This will compare the allocated to available space on the volume destination. If the ratio exceeds this number, the destination will no longer be valid.
nfs_shares_config=/etc/cinder/nfs_shares	(StrOpt) File with the list of available nfs shares
nfs_sparsed_volumes=True	(BoolOpt) Create volumes as sparsed files which take no space.If set to False volume is created as regular file.In such case volume creation takes a lot of time.
nfs_used_ratio=0.95	(FloatOpt) Percent of ACTUAL usage of the underlying volume before no new volumes can be allocated to the volume destination.

How to use the NFS driver

1. Access to one or more NFS servers. Creating an NFS server is outside the scope of this document. This example assumes access to the following NFS servers and mount points:

   - `192.168.1.200:/storage`
   - `192.168.1.201:/storage`
   - `192.168.1.202:/storage`

   This example demonstrates the use of with this driver with multiple NFS servers. Multiple servers are not required. One is usually enough.

2. Add your list of NFS servers to the file you specified with the `nfs_shares_config` option. For example, if the value of this option was set to `/etc/cinder/shares.txt`, then:

```
# cat /etc/cinder/shares.txt
192.168.1.200:/storage
192.168.1.201:/storage
192.168.1.202:/storage
```

   Comments are allowed in this file. They begin with a #.

3. Configure the `nfs_mount_point_base` option. This is a directory where `cinder-volume` mounts all NFS shares stored in `shares.txt`. For this example, `/var/lib/cinder/nfs` is used. You can, of course, use the default value of `$state_path/mnt`.

4. Start the `cinder-volume` service. `/var/lib/cinder/nfs` should now contain a directory for each NFS share specified in `shares.txt`. The name of each directory is a hashed name:

```
# ls /var/lib/cinder/nfs/
...
46c5db75dc3a3a50a10bfd1a456a9f3f
...
```

5. You can now create volumes as you normally would:

```
# nova volume-create --display-name=myvol 5
# ls /var/lib/cinder/nfs/46c5db75dc3a3a50a10bfd1a456a9f3f
volume-a8862558-e6d6-4648-b5df-bb84f31c8935
```

This volume can also be attached and deleted just like other volumes. However, snapshotting is *not* supported.

NFS driver notes

- `cinder-volume` manages the mounting of the NFS shares as well as volume creation on the shares. Keep this in mind when planning your OpenStack architecture. If you have one master NFS server, it might make sense to only have one `cinder-volume` service to handle all requests to that NFS server. However, if that single server is unable to handle all requests, more than one `cinder-volume` service is needed as well as potentially more than one NFS server.
- Because data is stored in a file and not actually on a block storage device, you might not see the same IO performance as you would with a traditional block storage driver. Please test accordingly.
- Despite possible IO performance loss, having volume data stored in a file might be beneficial. For example, backing up volumes can be as easy as copying the volume files.

Note

Regular IO flushing and syncing still stands.

SolidFire

The SolidFire Cluster is a high performance all SSD iSCSI storage device that provides massive scale out capability and extreme fault tolerance. A key feature of the SolidFire cluster is the ability to set and modify during operation specific QoS levels on a volume for volume basis. The SolidFire cluster offers this along with de-duplication, compression, and an architecture that takes full advantage of SSDs.

To configure the use of a SolidFire cluster with Block Storage, modify your `cinder.conf` file as follows:

```
volume_driver=cinder.volume.drivers.solidfire.SolidFire
san_ip=172.17.1.182         # the address of your MVIP
san_login=sfadmin           # your cluster admin login
san_password=sfpassword     # your cluster admin password
sf_account_prefix=''        # prefix for tenant account creation on solidfire
 cluster (see warning below)
```

Warning

The SolidFire driver creates a unique account prefixed with `$cinder-volume-service-hostname-$tenant-id` on the SolidFire cluster for each tenant that accesses the cluster through the Volume API. Unfortunately, this account formation results in issues for High Availability (HA) installations and installations where the `cinder-volume` service can move to a new node.

HA installations can return an Account Not Found error because the call to the SolidFire cluster is not always going to be sent from the same node. In installations where the `cinder-volume` service moves to a new node, the same issue can occur when you perform operations on existing volumes, such as clone, extend, delete, and so on.

Tip

Set the `sf_account_prefix` option to an empty string ('') in the `cinder.conf` file. This setting results in unique accounts being created on the SolidFire cluster, but the accounts are prefixed with the tenant-id or any unique identifier that you choose and are independent of the host where the `cinder-volume` service resides.

Table 7.17. Description of configuration options for solidfire

Configuration option=Default value	Description
sf_account_prefix=docwork	(StrOpt) Create SolidFire accounts with this prefix
sf_allow_tenant_qos=False	(BoolOpt) Allow tenants to specify QOS on create
sf_api_port=443	(IntOpt) SolidFire API port. Useful if the device api is behind a proxy on a different port.
sf_emulate_512=True	(BoolOpt) Set 512 byte emulation on volume creation;

VMware VMDK driver

The VMware VMDK driver enables management of OpenStack Block Storage volumes on vCenter-managed datastores. Volumes are backed by VMDK files on datastores using any VMware compatible storage technology (such as, NFS, iSCSI, FiberChannel, and vSAN).

Configuration

The recommended OpenStack Block Storage volume driver is the VMware vCenter VMDK driver. When configuring the driver, you must match it with the appropriate OpenStack Compute driver from VMware and both drivers must point to the same server.

For example, in the `nova.conf` file, the volume driver setting is:

```
compute_driver=vmwareapi.VMwareVCDriver
```

In the `cinder.conf` file, the volume driver setting is:

```
volume_driver=cinder.volume.drivers.vmware.vmdk.VMwareVcVmdkDriver
```

The following table lists various options that the drivers support for the OpenStack Block Storage configuration (`cinder.conf`):

Table 7.18. Description of configuration options for vmware

Configuration option=Default value	Description
vmware_api_retry_count=10	(IntOpt) Number of times VMware ESX/VC server API must be retried upon connection related issues.
vmware_host_ip=None	(StrOpt) IP address for connecting to VMware ESX/VC server.

Configuration option=Default value	Description
vmware_host_password=None	(StrOpt) Password for authenticating with VMware ESX/VC server.
vmware_host_username=None	(StrOpt) Username for authenticating with VMware ESX/VC server.
vmware_image_transfer_timeout_secs=7200	(IntOpt) Timeout in seconds for VMDK volume transfer between Cinder and Glance.
vmware_max_objects_retrieval=100	(IntOpt) Max number of objects to be retrieved per batch. Query results will be obtained in batches from the server and not in one shot. Server may still limit the count to something less than the configured value.
vmware_task_poll_interval=5	(IntOpt) The interval used for polling remote tasks invoked on VMware ESX/VC server.
vmware_volume_folder=cinder-volumes	(StrOpt) Name for the folder in the VC datacenter that will contain cinder volumes.
vmware_wsdl_location=None	(StrOpt) Optional VIM service WSDL Location e.g http://<server>/vimService.wsdl. Optional over-ride to default location for bug work-arounds.

VMDK disk type

The VMware VMDK drivers support creating VMDK disk files of type: `thin`, `thick` and `eagerZeroedThick`. The VMDK disk file type is specified using the `vmware:vmdk_type` extra spec key with the appropriate value. The following table captures the mapping between the extra spec entry and the VMDK disk file type:

Table 7.19. Extra spec entry to VMDK disk file type mapping

Disk file type	Extra spec key	Extra spec value
thin	vmware:vmdk_type	thin
thick	vmware:vmdk_type	thick
eagerZeroedThick	vmware:vmdk_type	eagerZeroedThick

If no `vmdk_type` extra spec entry is specified, the default disk file type is `thin`.

The example below shows how to create a `thick` VMDK volume using the appropriate `vmdk_type`:

```
$ cinder type-create thick_volume
$ cinder type-key thick_volume set vmware:vmdk_type=thick
$ cinder create --volume-type thick_volume --display-name volume1 1
```

Clone type

With the VMware VMDK drivers, you can create a volume from another source volume or from a snapshot point. The VMware vCenter VMDK driver supports clone types `full` and `linked/fast`. The clone type is specified using the `vmware:clone_type` extra spec key with the appropriate value. The following table captures the mapping for clone types:

Table 7.20. Extra spec entry to clone type mapping

Clone type	Extra spec key	Extra spec value
full	vmware:clone_type	full

Clone type	Extra spec key	Extra spec value
linked/fast	vmware:clone_type	linked

If not specified, the default clone type is `full`.

The following is an example of linked cloning from another source volume:

```
$ cinder type-create fast_clone
$ cinder type-key fast_clone set vmware:clone_type=linked
$ cinder create --volume-type fast_clone --source-volid
25743b9d-3605-462b-b9eb-71459fe2bb35 --display-name volume1 1
```

Note: The VMware ESX VMDK driver ignores the extra spec entry and always creates a `full` clone.

Supported operations

The following operations are supported by the VMware vCenter and ESX VMDK drivers:

- Create volume
- Create volume from another source volume. (Supported only if source volume is not attached to an instance.)
- Create volume from snapshot
- Create volume from glance image
- Attach volume (When a volume is attached to an instance, a reconfigure operation is performed on the instance to add the volume's VMDK to it. The user must manually rescan and mount the device from within the guest operating system.)
- Detach volume
- Create snapshot (Allowed only if volume is not attached to an instance.)
- Delete snapshot (Allowed only if volume is not attached to an instance.)
- Upload as image to glance (Allowed only if volume is not attached to an instance.)

Note

Although the VMware ESX VMDK driver supports these operations, it has not been extensively tested.

Datastore selection

When creating a volume, the driver chooses a datastore that has sufficient free space and has the highest `freespace/totalspace` metric value.

When a volume is attached to an instance, the driver attempts to place the volume under the instance's ESX host on a datastore that is selected using the strategy above.

Windows

There is a volume back-end for Windows. Set the following in your `cinder.conf`, and use the options below to configure it.

```
volume_driver=cinder.volume.drivers.windows.WindowsDriver
```

Table 7.21. Description of configuration options for windows

Configuration option=Default value	Description
windows_iscsi_lun_path=C:\iSCSIVirtualDisks	(StrOpt) Path to store VHD backed volumes

XenAPINFS

XenAPINFS is a Block Storage (Cinder) driver that uses an NFS share through the XenAPI Storage Manager to store virtual disk images and expose those virtual disks as volumes.

This driver does not access the NFS share directly. It accesses the share only through XenAPI Storage Manager. Consider this driver as a reference implementation for use of the XenAPI Storage Manager in OpenStack (present in XenServer and XCP).

Requirements

- A XenServer/XCP installation that acts as Storage Controller. This hypervisor is known as the storage controller.
- Use XenServer/XCP as your hypervisor for Compute nodes.
- An NFS share that is configured for XenServer/XCP. For specific requirements and export options, see the administration guide for your specific XenServer version. The NFS share must be accessible by all XenServers components within your cloud.
- To create volumes from XenServer type images (vhd tgz files), XenServer Nova plug-ins are also required on the storage controller.

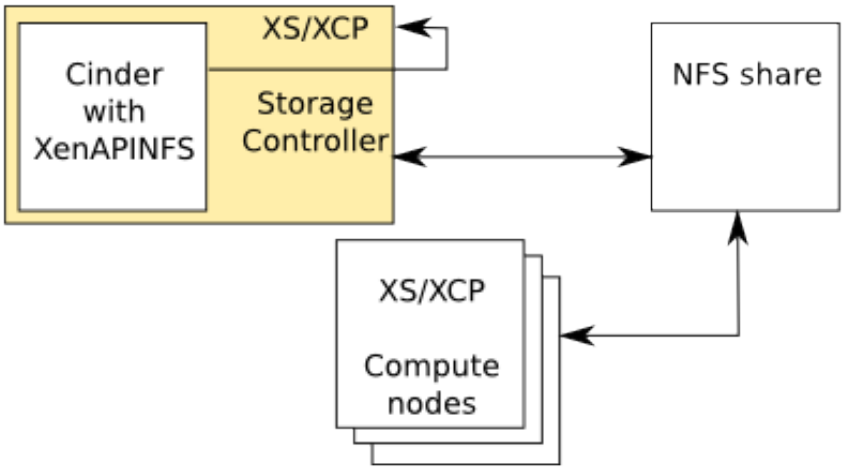
Note

You can use a XenServer as a storage controller and Compute node at the same time. This minimal configuration consists of a XenServer/XCP box and an NFS share.

Configuration patterns

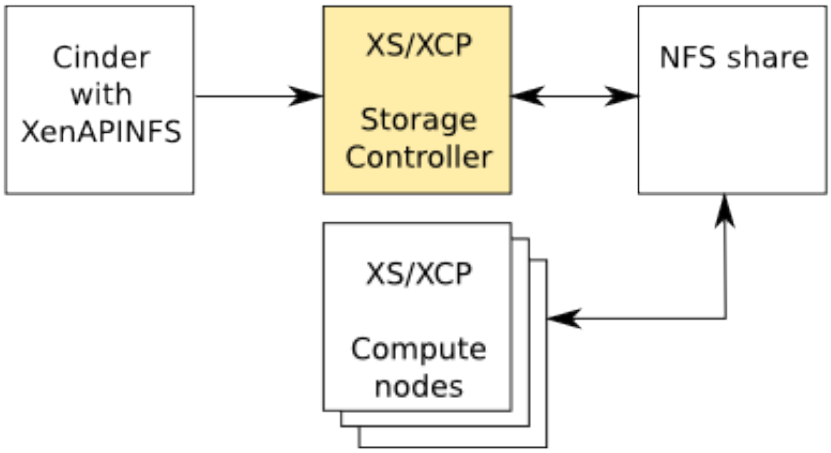
- Local configuration (Recommended): The driver runs in a virtual machine on top of the storage controller. With this configuration, you can create volumes from `qemu-img`-supported formats.

Figure 7.3. Local configuration



- Remote configuration: The driver is not a guest VM of the storage controller. With this configuration, you can only use XenServer vhd-type images to create volumes.

Figure 7.4. Remote configuration



Configuration options

Assuming the following setup:

- XenServer box at `10.2.2.1`
- XenServer password is `r00tme`
- NFS server is `nfs.example.com`
- NFS export is at `/volumes`

To use XenAPINFS as your cinder driver, set these configuration options in the `cinder.conf` file:

```
volume_driver = cinder.volume.drivers.xenapi.sm.XenAPINFSDriver
xenapi_connection_url = http://10.2.2.1
xenapi_connection_username = root
xenapi_connection_password = r00tme
xenapi_nfs_server = nfs.example.com
xenapi_nfs_serverpath = /volumes
```

The following table shows the configuration options that the XenAPINFS driver supports:

Table 7.22. Description of configuration options for storage_xen

Configuration option=Default value	Description
xenapi_connection_password=None	(StrOpt) Password for XenAPI connection
xenapi_connection_url=None	(StrOpt) URL for XenAPI connection
xenapi_connection_username=root	(StrOpt) Username for XenAPI connection
xenapi_nfs_server=None	(StrOpt) NFS server to be used by XenAPINFSDriver
xenapi_nfs_serverpath=None	(StrOpt) Path of exported NFS, used by XenAPINFSDriver
xenapi_sr_base_path=/var/run/sr-mount	(StrOpt) Base path to the storage repository

XenAPI Storage Manager volume driver

The Xen Storage Manager volume driver (xensm) is a XenAPI hypervisor specific volume driver, and can be used to provide basic storage functionality, including volume creation and destruction, on a number of different storage back-ends. It also enables the capability of using more sophisticated storage back-ends for operations like cloning/snapshots, etc. The list below shows some of the storage plug-ins already supported in Citrix XenServer and Xen Cloud Platform (XCP):

1. NFS VHD: Storage repository (SR) plug-in that stores disks as Virtual Hard Disk (VHD) files on a remote Network File System (NFS).

2. Local VHD on LVM: SR plug-in tjat represents disks as VHD disks on Logical Volumes (LVM) within a locally-attached Volume Group.

3. HBA LUN-per-VDI driver: SR plug-in that represents Logical Units (LUs) as Virtual Disk Images (VDIs) sourced by host bus adapters (HBAs). For example, hardware-based iSCSI or FC support.

4. NetApp: SR driver for mapping of LUNs to VDIs on a NETAPP server, providing use of fast snapshot and clone features on the filer.

5. LVHD over FC: SR plug-in that represents disks as VHDs on Logical Volumes within a Volume Group created on an HBA LUN. For example, hardware-based iSCSI or FC support.

6. iSCSI: Base ISCSI SR driver, provides a LUN-per-VDI. Does not support creation of VDIs but accesses existing LUNs on a target.

7. LVHD over iSCSI: SR plug-in that represents disks as Logical Volumes within a Volume Group created on an iSCSI LUN.

8. EqualLogic: SR driver for mapping of LUNs to VDIs on a EQUALLOGIC array group, providing use of fast snapshot and clone features on the array.

Design and operation

Definitions

- **Back-end:** A term for a particular storage back-end. This could be iSCSI, NFS, Netapp etc.
- **Back-end-config:** All the parameters required to connect to a specific back-end. For example, for NFS, this would be the server, path, and so on.
- **Flavor:** This term is equivalent to volume "types". A user friendly term to specify some notion of quality of service. For example, "gold" might mean that the volumes use a back-end where backups are possible. A flavor can be associated with multiple back-ends. The volume scheduler, with the help of the driver, decides which back-end is used to create a volume of a particular flavor. Currently, the driver uses a simple "first-fit" policy, where the first back-end that can successfully create this volume is the one that is used.

Operation

The admin uses the nova-manage command detailed below to add flavors and back-ends.

One or more `cinder-volume` service instances are deployed for each availability zone. When an instance is started, it creates storage repositories (SRs) to connect to the back-ends available within that zone. All `cinder-volume` instances within a zone can see all the available back-ends. These instances are completely symmetric and hence should be able to service any `create_volume` request within the zone.

On XenServer, PV guests required

Note that when using XenServer you can only attach a volume to a PV guest.

Configure XenAPI Storage Manager

Prerequisites

1. xensm requires that you use either Citrix XenServer or XCP as the hypervisor. The NetApp and EqualLogic back-ends are not supported on XCP.
2. Ensure all **hosts** running volume and compute services have connectivity to the storage system.

Configuration

- **Set the following configuration options for the nova volume service: (`nova-compute` also requires the volume_driver configuration option.)**

```
--volume_driver="nova.volume.xensm.XenSMDriver"
--use_local_volumes=False
```

- **The back-end configurations that the volume driver uses need to be created before starting the volume service.**

```
$ nova-manage sm flavor_create <label> <description>

$ nova-manage sm flavor_delete <label>

$ nova-manage sm backend_add <flavor label> <SR type> [config connection
 parameters]

Note: SR type and config connection parameters are in keeping with the
 XenAPI Command Line Interface. http://support.citrix.com/article/CTX124887

$ nova-manage sm backend_delete <back-end-id>
```

Example: For the NFS storage manager plug-in, the steps below may be used.

```
$ nova-manage sm flavor_create gold "Not all that glitters"

$ nova-manage sm flavor_delete gold

$ nova-manage sm backend_add gold nfs name_label=myback-end server=myserver
 serverpath=/local/scratch/myname

$ nova-manage sm backend_remove 1
```

- **Start `cinder-volume` and `nova-compute` with the new configuration options.**

Create and access the volumes from VMs

Currently, the flavors have not been tied to the volume types API. As a result, we simply end up creating volumes in a "first fit" order on the given back-ends.

The standard euca-* or OpenStack API commands (such as volume extensions) should be used for creating, destroying, attaching, or detaching volumes.

Zadara

There is a volume back-end for Zadara. Set the following in your `cinder.conf`, and use the following options to configure it.

```
volume_driver=cinder.volume.drivers.zadara.ZadaraVPSAISCSIDriver
```

Table 7.23. Description of configuration options for zadara

Configuration option=Default value	Description
zadara_default_striping_mode=simple	(StrOpt) Default striping mode for volumes
zadara_password=None	(StrOpt) Password for the VPSA
zadara_user=None	(StrOpt) User name for the VPSA
zadara_vol_encrypt=False	(BoolOpt) Default encryption policy for volumes
zadara_vol_name_template=OS_%s	(StrOpt) Default template for VPSA volume names
zadara_vol_thin=True	(BoolOpt) Default thin provisioning policy for volumes
zadara_vpsa_allow_nonexistent_delete=True	(BoolOpt) Don't halt on deletion of non-existing volumes

Configuration option=Default value	Description
zadara_vpsa_auto_detach_on_delete=True	(BoolOpt) Automatically detach from servers on volume delete
zadara_vpsa_ip=None	(StrOpt) Management IP of Zadara VPSA
zadara_vpsa_poolname=None	(StrOpt) Name of VPSA storage pool for volumes
zadara_vpsa_port=None	(StrOpt) Zadara VPSA port number
zadara_vpsa_use_ssl=False	(BoolOpt) Use SSL connection

Backup drivers

This section describes how to configure the `cinder-backup` service and its drivers.

The volume drivers are included with the Block Storage repository (https://github.com/openstack/cinder). To set a backup driver, use the `backup_driver` flag. By default there is no backup driver enabled.

Ceph backup driver

The Ceph backup driver backs up volumes of any type to a Ceph back-end store. The driver can also detect whether the volume to be backed up is a Ceph RBD volume, and if so, it tries to perform incremental and differential backups.

For source Ceph RBD volumes, you can perform backups within the same Ceph pool (not recommended) and backups between different Ceph pools and between different Ceph clusters.

At the time of writing, differential backup support in Ceph/librbd was quite new. This driver attempts a differential backup in the first instance. If the differential backup fails, the driver falls back to full backup/copy.

If incremental backups are used, multiple backups of the same volume are stored as snapshots so that minimal space is consumed in the backup store. It takes far less time to restore a volume than to take a full copy.

Note

Block Storage Service enables you to:

- Restore to a new volume, which is the default and recommended action.
- Restore to the original volume from which the backup was taken. The restore action takes a full copy because this is the safest action.

To enable the Ceph backup driver, include the following option in the `cinder.conf` file:

```
backup_driver=cinder.backup.driver.ceph
```

The following configuration options are available for the Ceph backup driver.

Table 7.24. Description of configuration options for backups_ceph

Configuration option=Default value	Description
backup_ceph_chunk_size=134217728	(IntOpt) the chunk size in bytes that a backup will be broken into before transfer to backup store

Configuration option=Default value	Description
backup_ceph_conf=/etc/ceph/ceph.conf	(StrOpt) Ceph config file to use.
backup_ceph_pool=backups	(StrOpt) the Ceph pool to backup to
backup_ceph_stripe_count=0	(IntOpt) RBD stripe count to use when creating a backup image
backup_ceph_stripe_unit=0	(IntOpt) RBD stripe unit to use when creating a backup image
backup_ceph_user=cinder	(StrOpt) the Ceph user to connect with
restore_discard_excess_bytes=True	(BoolOpt) If True, always discard excess bytes when restoring volumes.

This example shows the default options for the Ceph backup driver.

```
backup_ceph_conf=/etc/ceph/ceph.conf
backup_ceph_user=cinder
backup_ceph_chunk_size=134217728
backup_ceph_pool=backups
backup_ceph_stripe_unit=0
backup_ceph_stripe_count=0
```

IBM Tivoli Storage Manager backup driver

The IBM Tivoli Storage Manager (TSM) backup driver enables performing volume backups to a TSM server.

The TSM client should be installed and configured on the machine running the `cinder-backup` service. See the *IBM Tivoli Storage Manager Backup-Archive Client Installation and User's Guide* for details on installing the TSM client.

To enable the IBM TSM backup driver, include the following option in `cinder.conf`:

```
backup_driver=cinder.backup.driver.tsm
```

The following configuration options are available for the TSM backup driver.

Table 7.25. Description of configuration options for backups_tsm

Configuration option=Default value	Description
backup_tsm_compression=True	(BoolOpt) Enable or Disable compression for backups
backup_tsm_password=password	(StrOpt) TSM password for the running username
backup_tsm_volume_prefix=backup	(StrOpt) Volume prefix for the backup id when backing up to TSM

This example shows the default options for the TSM backup driver.

```
backup_tsm_volume_prefix = backup
backup_tsm_password = password
backup_tsm_compression = True
```

Swift backup driver

The backup driver for Swift back-end performs a volume backup to a Swift object storage system.

To enable the Swift backup driver, include the following option in the `cinder.conf` file:

```
backup_driver=cinder.backup.driver.swift
```

The following configuration options are available for the Swift back-end backup driver.

Table 7.26. Description of configuration options for backups_swift

Configuration option=Default value	Description
backup_swift_auth=per_user	(StrOpt) Swift authentication mechanism
backup_swift_container=volumebackups	(StrOpt) The default Swift container to use
backup_swift_key=None	(StrOpt) Swift key for authentication
backup_swift_object_size=52428800	(IntOpt) The size in bytes of Swift backup objects
backup_swift_retry_attempts=3	(IntOpt) The number of retries to make for Swift operations
backup_swift_retry_backoff=2	(IntOpt) The backoff time in seconds between Swift retries
backup_swift_url=http://localhost:8080/v1/AUTH_	(StrOpt) The URL of the Swift endpoint
backup_swift_user=None	(StrOpt) Swift user name

This example shows the default options for the Swift back-end backup driver.

```
backup_swift_url=http://localhost:8080/v1/AUTH
backup_swift_auth=per_user
backup_swift_user=<None>
backup_swift_key=<None>
backup_swift_container=volumebackups
backup_swift_object_size=52428800
backup_swift_retry_attempts=3
backup_swift_retry_backoff=2
backup_compression_algorithm=zlib
```

Appendix A. Community support

Table of Contents

Many resources are available to help you run and use OpenStack. Members of the OpenStack community can answer questions and help with bug suspicions. We are constantly improving and adding to the main features of OpenStack, but if you have any problems, do not hesitate to ask. Use the following resources to get OpenStack support and troubleshoot your existing installations.

Documentation

For the available OpenStack documentation, see docs.openstack.org.

To provide feedback on documentation, join and use the <openstack-docs@lists.openstack.org> mailing list at OpenStack Documentation Mailing List, or report a bug.

The following books explain how to install an OpenStack cloud and its associated components:

- *Installation Guide for Debian 7.0*
- *Installation Guide for openSUSE and SUSE Linux Enterprise Server*
- *Installation Guide for Red Hat Enterprise Linux, CentOS, and Fedora*
- *Installation Guide for Ubuntu 12.04 (LTS)*

The following books explain how to configure and run an OpenStack cloud:

- *Cloud Administrator Guide*
- *Configuration Reference*
- *Operations Guide*
- *High Availability Guide*
- *Security Guide*

- *Virtual Machine Image Guide*

The following books explain how to use the OpenStack dashboard and command-line clients:

- *API Quick Start*
- *End User Guide*
- *Admin User Guide*

The following documentation provides reference and guidance information for the OpenStack APIs:

- OpenStack API Reference
- *OpenStack Block Storage Service API v2 Reference*
- *OpenStack Compute API v2 and Extensions Reference*
- *OpenStack Identity Service API v2.0 Reference*
- *OpenStack Image Service API v2 Reference*
- *OpenStack Networking API v2.0 Reference*
- *OpenStack Object Storage API v1 Reference*

ask.openstack.org

During the set up or testing of OpenStack, you might have questions about how a specific task is completed or be in a situation where a feature does not work correctly. Use the ask.openstack.org site to ask questions and get answers. When you visit the http://ask.openstack.org site, scan the recently asked questions to see whether your question has already been answered. If not, ask a new question. Be sure to give a clear, concise summary in the title and provide as much detail as possible in the description. Paste in your command output or stack traces, links to screen shots, and so on.

OpenStack mailing lists

A great way to get answers and insights is to post your question or problematic scenario to the OpenStack mailing list. You can learn from and help others who might have similar issues. To subscribe or view the archives, go to http://lists.openstack.org/cgi-bin/mailman/listinfo/openstack. You might be interested in the other mailing lists for specific projects or development, which you can find on the wiki. A description of all mailing lists is available at http://wiki.openstack.org/MailingLists.

The OpenStack wiki

The OpenStack wiki contains a broad range of topics but some of the information can be difficult to find or is a few pages deep. Fortunately, the wiki search feature enables you to search by title or content. If you search for specific information, such as about networking

or nova, you can find lots of relevant material. More is being added all the time, so be sure to check back often. You can find the search box in the upper right corner of any OpenStack wiki page.

The Launchpad Bugs area

The OpenStack community values your set up and testing efforts and wants your feedback. To log a bug, you must sign up for a Launchpad account at https://launchpad.net/+login. You can view existing bugs and report bugs in the Launchpad Bugs area. Use the search feature to determine whether the bug has already been reported or even better, already fixed. If it still seems like your bug is unreported, fill out a bug report.

Some tips:

- Give a clear, concise summary!
- Provide as much detail as possible in the description. Paste in your command output or stack traces, links to screen shots, and so on.
- Be sure to include the software and package versions that you are using, especially if you are using a development branch, such as, `"Grizzly release" vs git commit bc79c3ecc55929bac585d04a03475b72e06a3208`.
- Any deployment specific information is helpful, such as Ubuntu 12.04 or multi-node install.

The Launchpad Bugs areas are available here:

- Bugs: OpenStack Compute (nova)
- Bugs : OpenStack Object Storage (swift)
- Bugs : OpenStack Image Service (glance)
- Bugs : OpenStack Identity (keystone)
- Bugs : OpenStack Dashboard (horizon)
- Bugs : OpenStack Networking (neutron)
- Bugs: Orchestration (heat)
- Bugs: Telemetry (ceilometer)
- Bugs : OpenStack Documentation (docs.openstack.org)
- Bugs : OpenStack API Documentation (api.openstack.org)

The OpenStack IRC channel

The OpenStack community lives and breathes in the #openstack IRC channel on the Freenode network. You can hang out, ask questions, or get immediate feedback for urgent and pressing issues. To install an IRC client or use a browser-based client, go to http://

webchat.freenode.net/. You can also use Colloquy (Mac OS X, http://colloquy.info/), mIRC (Windows, http://www.mirc.com/), or XChat (Linux). When you are in the IRC channel and want to share code or command output, the generally accepted method is to use a Paste Bin. The OpenStack project has one at http://paste.openstack.org. Just paste your longer amounts of text or logs in the web form and you get a URL you can paste into the channel. The OpenStack IRC channel is: `#openstack` on `irc.freenode.net`. You can find a list of all OpenStack-related IRC channels at https://wiki.openstack.org/wiki/IRC.

Documentation feedback

To provide feedback on documentation, join and use the `<openstack-docs@lists.openstack.org>` mailing list at OpenStack Documentation Mailing List, or report a bug.

OpenStack distribution packages

The following Linux distributions provide community-supported packages for OpenStack:

- **Debian:** http://wiki.debian.org/OpenStack
- **CentOS, Fedora, and Red Hat Enterprise Linux:** http://openstack.redhat.com/
- **openSUSE and SUSE Linux Enterprise Server:** http://en.opensuse.org/Portal:OpenStack
- **Ubuntu:** https://wiki.ubuntu.com/ServerTeam/CloudArchive