OBJECT STORAGE QUICK START

To use this guide, you must have executed the procedures in the 5-minute Quick Start guide first.

INSTALL APACHE AND FASTCGI

The Ceph object storage gateway runs on Apache and FastCGI. Install them on the server machine. Use the following procedure:

1. Install Apache and FastCGI on the server machine.

```
sudo apt-get update && sudo apt-get install apache2 libapache2-mod-fastcgi
```

2. Enable the URL rewrite modules for Apache and FastCGI.

```
sudo a2enmod rewrite
sudo a2enmod fastcgi
```

3. Add a line for the `ServerName` in the `/etc/apache2/httpd.conf` file. Provide the fully qualified domain name of the server machine.

   ServerName {fqdn}

4. Restart Apache so that the foregoing changes take effect.

```
sudo service apache2 restart
```

INSTALL RADOS GATEWAY

Once you have installed and configured Apache and FastCGI, you may install Ceph's RADOS Gateway.

```
sudo apt-get install radosgw
```

For details on the preceding steps, see RADOS Gateway Manual Install.

MODIFY THE CEPH CONFIGURATION FILE

On the server machine, perform the following steps:

1. Open the Ceph configuration file.

```
cd /etc/ceph
vim ceph.conf
```

2. Add the following settings to the Ceph configuration file:

```
[client.radosgw.gateway]
host = {host-name}
keyring = /etc/ceph/keyring.radosgw.gateway
rgw socket path = /tmp/radosgw.sock
log file = /var/log/ceph/radosgw.log
```

3. Go to the client machine and copy the configuration file from the server machine to `/etc/ceph/ceph.conf` on your client machine.

```
sudo scp {user}@{cluster-machine}:/etc/ceph/ceph.conf /etc/ceph/ceph.conf
```

Tip: Ensure the `ceph.conf` file has appropriate permissions set (e.g. `chmod 644`) on your client machine.

CREATE A DATA DIRECTORY

Create a data directory on the cluster server for the instance of `radosgw`.

```
sudo mkdir -p /var/lib/ceph/radosgw/ceph-radosgw.gateway
```

CREATE A GATEWAY CONFIGURATION FILE

The example configuration file will configure the gateway to operate with the Apache FastCGI module, a rewrite rule for OpenStack Swift, and paths for the log files. To add a configuration file for the Ceph Gateway, we suggest copying the contents of the example file below to an editor. Then, follow the steps below to modify it.

```
FastCgiExternalServer /var/www/s3gw.fcgi -socket /tmp/radosgw.sock


<VirtualHost *:80>
        ServerName {fqdn}
        ServerAdmin {email.address}
        DocumentRoot /var/www
</VirtualHost>

RewriteEngine On
RewriteRule ^/([a-zA-Z0-9-_.]*)([/]?.*) /s3gw.fcgi?page=$1&params=$2&%{QUERY_STRING} [E=HTTP_

<VirtualHost *:80>

        <IfModule mod_fastcgi.c>
                <Directory /var/www>
                        Options +ExecCGI
                        AllowOverride All
                        SetHandler fastcgi-script
                        Order allow,deny
                        Allow from all
                        AuthBasicAuthoritative Off
                </Directory>
        </IfModule>

        AllowEncodedSlashes On
        ErrorLog /var/log/apache2/error.log
        CustomLog /var/log/apache2/access.log combined
        ServerSignature Off
</VirtualHost>
```

1. Replace the `{fqdn}` entry with the fully-qualified domain name of the server server.
2. Replace the `{email.address}` entry with the email address for the server administrator.
3. Save the contents to the `/etc/apache2/sites-available` directory on the server machine.
4. Enable the site for `rgw.conf`.

   ```
   sudo a2ensite rgw.conf
   ```

5. Disable the default site.

   ```
   sudo a2dissite default
   ```

See Create rgw.conf for additional details.

ADD A FASTCGI SCRIPT

FastCGI requires a script for the S3-compatible interface. To create the script, execute the following procedures on the server machine.

1. Go to the `/var/www` directory.

```
cd /var/www
```

2. Open an editor with the file name `s3gw.fcgi`.

```
sudo vim s3gw.fcgi
```

3. Copy the following into the editor.

```
#!/bin/sh
exec /usr/bin/radosgw -c /etc/ceph/ceph.conf -n client.radosgw.gateway
```

4. Save the file.
5. Change the permissions on the file so that it is executable.

```
sudo chmod +x s3gw.fcgi
```

GENERATE A KEYRING AND KEY

Perform the following steps on the server machine.

1. Create a keyring for the RADOS Gateway.

```
sudo ceph-authtool --create-keyring /etc/ceph/keyring.radosgw.gateway
sudo chmod +r /etc/ceph/keyring.radosgw.gateway
```

2. Create a key for the RADOS Gateway to authenticate with the cluster.

```
sudo ceph-authtool /etc/ceph/keyring.radosgw.gateway -n client.radosgw.gateway --gen-key
sudo ceph-authtool -n client.radosgw.gateway --cap osd 'allow rwx' --cap mon 'allow r' /e
```

3. Add the key to the Ceph keyring.

```
sudo ceph -k /etc/ceph/ceph.keyring auth add client.radosgw.gateway -i /etc/ceph/keyring.
```

CREATE A USER

To use the Gateway, you must create a Gateway user. First, create a gateway user for the S3-compatible interface; then, create a subuser for the Swift-compatible interface.

GATEWAY (S3) USER

First, create a Gateway user for the S3-compatible interface.

```
sudo radosgw-admin user create --uid="{username}" --display-name="{Display Name}"
```

For example:

```
radosgw-admin user create --uid=johndoe --display-name="John Doe" --email=john@example.com
```

```
{ "user_id": "johndoe",
  "rados_uid": 0,
  "display_name": "John Doe",
  "email": "john@example.com",
  "suspended": 0,
  "subusers": [],
  "keys": [
    { "user": "johndoe",
      "access_key": "QFAMEDSJP5DEKJO0DDXY",
      "secret_key": "iaSFLDVvDdQt6lkNzHyW4fPLZugBAI1g17LO0+87"}],
  "swift_keys": []
 }
```

Creating a user creates an access_key and secret_key entry for use with any S3 API-compatible client.

Important: Check the key output. Sometimes radosgw-admin generates a key with an escape () character, and some clients do not know how to handle escape characters. Remedies include removing the escape character (), encapsulating the string in quotes, or simply regenerating the key and ensuring that it does not have an escape character.

SUBUSER

Next, create a subuser for the Swift-compatible interface.

```
sudo radosgw-admin subuser create --uid=johndoe --subuser=johndoe:swift --access=full
```

```
{ "user_id": "johndoe",
  "rados_uid": 0,
  "display_name": "John Doe",
  "email": "john@example.com",
  "suspended": 0,
  "subusers": [
    { "id": "johndoe:swift",
      "permissions": "full-control"}],
  "keys": [
    { "user": "johndoe",
      "access_key": "QFAMEDSJP5DEKJO0DDXY",
      "secret_key": "iaSFLDVvDdQt6lkNzHyW4fPLZugBAI1g17LO0+87"}],
  "swift_keys": []}
```

```
sudo radosgw-admin key create --subuser=johndoe:swift --key-type=swift
```

```
{ "user_id": "johndoe",
  "rados_uid": 0,
  "display_name": "John Doe",
  "email": "john@example.com",
  "suspended": 0,
  "subusers": [
     { "id": "johndoe:swift",
       "permissions": "full-control"}],
  "keys": [
    { "user": "johndoe",
      "access_key": "QFAMEDSJP5DEKJO0DDXY",
      "secret_key": "iaSFLDVvDdQt6lkNzHyW4fPLZugBAI1g17LO0+87"}],
  "swift_keys": [
    { "user": "johndoe:swift",
      "secret_key": "E9T2rUZNu2gxUjcwUBO8n\/Ev4KX6\/GprEuH4qhu1"}]}
```

This step enables you to use any Swift client to connect to and use RADOS Gateway via the Swift-compatible API.

RGW's `user:subuser` tuple maps to the `tenant:user` tuple expected by Swift.

Note: RGW's Swift authentication service only supports built-in Swift authentication (`-V 1.0`) at this point. See RGW Configuration for Keystone integration details.

ENABLE SSL

Some REST clients use HTTPS by default. So you should consider enabling SSL for Apache on the server machine.

```
sudo a2enmod ssl
```

Once you enable SSL, you should generate an SSL certificate.

```
sudo mkdir /etc/apache2/ssl
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout /etc/apache2/ssl/apache.key
```

Then, restart Apache.

```
service apache2 restart
```