L103175 - Deploy Ceph Rados Gateway as a replacement for OpenStack Swift

Introduction

First of all, it's our pleasure to welcome you to Red Hat Summit 2017, here at the Boston Convention and Exhibition Centre! The past few years have been an exciting time for both Red Hat and the OpenStack community; we've seen unprecedented interest and development in this new revolutionary technology and we're proud to be at the heart of it all.

Red Hat is firmly committed to the future of OpenStack. Our goal is to continue to enhance the technology, make it more readily consumable and to enable our customers to be successful when using it.

This hands-on lab aims to get you understand how you can leverage Red Hat OpenStack Director to deploy Ceph Rados Gateway as a replacement for Openstack Swift.
In the lab you will:

- Learn how to Deploy Ceph Rados Gateway as a replacement for OpenStack Swift using Red Hat OpenStack Director
- Understand how Ceph Rados Gateway is configured to talk to OpenStack Keystone v3 API
- Acquire skills on basic Ceph Rados Gateway commands and interactions
- Gain knowledge on interacting with Ceph Rados Gateway using both Swift and Amazon S3 APIs.

At the end of this session, you will have a good understanding on how to take advantage of Object Storage with a single unified storage backend.

Openstack Lab Architecture

The lab's environment is entirely virtualized and consists of 6 VMs :

Ironic-name	Function	vCPU	vRAM	NIC	Disk
overcloud-ctrl01	Controller	2	8GB	1NIC Provisioning 2NICs data	1x60GB OS
overcloud-ceph01	Ceph OSD	4	4GB	1NIC Provisioning 2NICs data	1x60GB OS 1x60GB OSD
overcloud-ceph02	Ceph OSD	4	4GB	1NIC Provisioning 2NICs data	1x60GB
overcloud-ceph03	Ceph OSD	4	4GB	1NIC Provisioning 2NICs data	1x60GB
overcloud-compute01	Compute	4	8GB	1NIC Provisioning 2NICs data	1x60GB
overcloud-rgw01	RGW	2	4GB	1NIC Provisioning 2NICs data	1x60GB

Plus the Undercloud VM which is obviously not registered into Ironic (4vCPU, 16GB vRAM, 2xNICs (1 data, 1 provisioning), 1x 40GB Disk).

IMPORTANT: *You may have noticed that we only have one controller node, this is because we are limited by the lab's hardware. In real life deployments* ***we strongly recommend to deploy multiple controllers in a HA fashion.***

This lab focuses only on RadosGW implementation, therefore we provide a ready to use undercloud environment and set of templates that configures the following :

- Nodes already registered into ironic
    - Tagged nodes profiles (except for RGW)
- Network isolation, one VLAN per usage (internalAPI, storage, tenant, etc) except for the external network (untagged).
- NIC config that uses the first NIC for provisioning plus a bond on second and third NICs for Openstack traffic.
- Enable Ceph for Nova,Glance and Cinder

Network Name	CIDR	VLAN ID
ControlPlane	172.16.0.0/24	Untagged
InternalAPI	172.17.1.0/24	101
Tenant	172.17.2.0/24	201
Storage	172.17.3.0/24	301
StorageMgmt	172.17.4.0/24	401
External	192.168.122.0/24	Untagged

Prepare the environment

Check your lab VMs

All lab's VMs are tagged with the lab ID (L103175), double check they are all defined and shut-off.

```
hypervisor# virsh list --all | grep L103175
 -     L103175-undercloud             shut off
 -     overcloud-L103175-ceph01       shut off
 -     overcloud-L103175-ceph02       shut off
 -     overcloud-L103175-ceph03       shut off
 -     overcloud-L103175-compute01    shut off
 -     overcloud-L103175-ctrl01       shut off
```

Also check that the provisioning network is active

```
hypervisor# virsh net-list | grep provisioning
 provisioning         active     yes           yes
```

Access the undercloud

First you need to boot up the undercloud VM and connect to it.

```
hypervisor# virsh start L103175-undercloud
hypervisor# ssh stack@undercloud
[password : redhat]
undercloud$ source ~/stackrc
```

IMPORTANT : All the steps mentioned below **MUST** be executed as “stack” user.

Note: The ready to use templates mentioned earlier are stored in /home/stack/templates/

List the available ironic nodes and check the maintenance flag **is set to “False”** also verify the provisioning state is “available”

```
undercloud$ openstack baremetal node list --field  name power_state provision_state
maintenance
+---------------------+-------------+--------------------+-------------+
| Name                | Power State | Provisioning State | Maintenance |
+---------------------+-------------+--------------------+-------------+
| overcloud-ctrl01    | power off   | available          | False       |
| overcloud-compute01 | power off   | available          | False       |
| overcloud-ceph01    | power off   | available          | False       |
| overcloud-ceph02    | power off   | available          | False       |
| overcloud-ceph03    | power off   | available          | False       |
| overcloud-rgw       | power off   | available          | False       |
+---------------------+-------------+--------------------+-------------+
```

If any of your nodes has a maintenance flag set to “True” use this command to turn it off.

```
undercloud$ for node in $(ironic node-list --fields name maintenance | grep True | awk '{print
$2}'); do ironic node-set-maintenance $node false; done
```

Create the rados gateway role

As explained during our presentation, we need to create a Rados Gateway role which will be used by director to deploy our RGW node.

Below the list of required steps :

- Define the Rados Gateway Role
- Create the Rados Gateway flavor
- Tag the Ironic node
- Configure the heat templates
    - Role matching and number of nodes
    - Configure network ports
    - Configure NICs attribution.

Copy and edit the default role file to your home directory

```
undercloud$ cp /usr/share/openstack-tripleo-heat-templates/roles_data.yaml
templates/custom_roles.yaml
undercloud$ vi templates/custom_roles.yaml
```

By default the Rados Gateway service is defined into the controller role, as we want a dedicated RGW server, **remove or comment this line** from the Controller role section (line 27):

```
    - OS::TripleO::Services::CephRgw # DELETE OR COMMENT
```

Then add the following block at the end of the file, make sure to respect indentation

```
- name: RadosGW
  CountDefault: 1
  HostnameFormatDefault: '%stackname%-rgw-%index%'
  ServicesDefault:
    - OS::TripleO::Services::CACerts
    - OS::TripleO::Services::Kernel
    - OS::TripleO::Services::Ntp
    - OS::TripleO::Services::Timezone
    - OS::TripleO::Services::Snmp
    - OS::TripleO::Services::TripleoPackages
    - OS::TripleO::Services::TripleoFirewall
    - OS::TripleO::Services::SensuClient
    - OS::TripleO::Services::FluentdClient
    - OS::TripleO::Services::CephRgw
    - OS::TripleO::Services::VipHosts
    - OS::TripleO::Services::CephClient
```

That’s it ! Your role definition is complete. Now let’s configure the node attribution details.

Configure the rados gateway role

Next step is to configure director so **it picks the right Ironic node(s) for this role**. First create and configure a “rgw” nova flavor :

```
undercloud$ openstack flavor create --id auto --ram 4096 --disk 40 --vcpus 1 rgw
undercloud$ openstack flavor set  --property "capabilities:boot_option"="local" --property
"capabilities:profile"="rgw" rgw
undercloud$ openstack flavor show rgw | grep properties
| properties                 | capabilities:boot_option='local', capabilities:profile='rgw' |
```

Then tag “overcloud-rgw” Ironic node with the profile we just defined above. By default all our nodes except overcloud-rgw has a profile assigned, let’s assign it one !

```
undercloud$ openstack overcloud profiles list
+--------------------------------------+---------------------+-----------------+-----------------+-------------------+
| Node UUID                            | Node Name           | Provision State | Current Profile | Possible Profiles |
+--------------------------------------+---------------------+-----------------+-----------------+-------------------+
| b2c14304-7c0f-42a2-9546-74fb2d950776 | overcloud-ctrl01    | available       | control         |                   |
| 6821b53a-1c9c-409d-8b7d-6d252a74e32e | overcloud-compute01 | available       | compute         |                   |
| 86ba67ce-daa6-4128-abe8-0e05c8d1fc10 | overcloud-ceph01    | available       | ceph-storage    |                   |
| fac3d6bc-61a4-4546-afbc-56d4f38b3ba0 | overcloud-ceph02    | available       | ceph-storage    |                   |
| 56096f50-1b1f-4279-8818-43a432d77ab5 | overcloud-ceph03    | available       | ceph-storage    |                   |
| e2442bbc-0468-477b-81c2-cf99d8be3d62 | overcloud-rgw       | available       | None            |                   |
+--------------------------------------+---------------------+-----------------+-----------------+-------------------+
```

Tag overcloud-rgw with the rgw profile

```
undercloud$ ironic node-update overcloud-rgw add
properties/capabilities='profile:rgw,boot_option:local'

undercloud$ openstack baremetal node show overcloud-rgw | grep properties
| properties             | {u'memory_mb': u'4096', u'cpu_arch': u'x86_64', u'local_gb': u'59',
u'cpus': u'2', u'capabilities': u'profile:rgw,boot_option:local'}

undercloud$  openstack overcloud profiles list
+--------------------------------------+---------------------+-----------------+-----------------+-------------------+
| Node UUID                            | Node Name           | Provision State | Current Profile | Possible Profiles |
+--------------------------------------+---------------------+-----------------+-----------------+-------------------+
| b2c14304-7c0f-42a2-9546-74fb2d950776 | overcloud-ctrl01    | active          | control         |                   |
| 6821b53a-1c9c-409d-8b7d-6d252a74e32e | overcloud-compute01 | active          | compute         |                   |
| 86ba67ce-daa6-4128-abe8-0e05c8d1fc10 | overcloud-ceph01    | active          | ceph-storage    |                   |
| fac3d6bc-61a4-4546-afbc-56d4f38b3ba0 | overcloud-ceph02    | active          | ceph-storage    |                   |
| 56096f50-1b1f-4279-8818-43a432d77ab5 | overcloud-ceph03    | active          | ceph-storage    |                   |
| e2442bbc-0468-477b-81c2-cf99d8be3d62 | overcloud-rgw       | active          | rgw             |                   |
+--------------------------------------+---------------------+-----------------+-----------------+-------------------+
```

Last step is to tell **director to pick a node that has the capability "rgw"** whenever a RGW role node needs to be deployed; we take this opportunity to configure the number of RGW node(s) we'd like to deploy, in our case just one.

```
undercloud$ cat << EOF > ~/templates/rgw-flavor.yaml
parameter_defaults:
  OvercloudRadosGWFlavor: rgw
  RadosGWCount: 1
EOF
```

Note: *Setting RadosGWCount to 1 is optional here as we already have CountDefault: 1 in the role file. This is only meant to demonstrate how to override the default value.*

Our new role definition is almost complete, the only part missing is **the network configuration**. We need to tell director on **which subnets this role needs to be connected and how to configure the node's NICs.**

Create a rgw-network.yaml file:

```
undercloud$ cat << EOF > ~/templates/rgw-network.yaml
resource_registry:
  OS::TripleO::RadosGW::Ports::ExternalPort:
/usr/share/openstack-tripleo-heat-templates/network/ports/noop.yaml
  OS::TripleO::RadosGW::Ports::InternalApiPort:
/usr/share/openstack-tripleo-heat-templates/network/ports/internal_api.yaml
  OS::TripleO::RadosGW::Ports::StoragePort:
/usr/share/openstack-tripleo-heat-templates/network/ports/storage.yaml
  OS::TripleO::RadosGW::Ports::StorageMgmtPort:
/usr/share/openstack-tripleo-heat-templates/network/ports/noop.yaml
  OS::TripleO::RadosGW::Ports::TenantPort:
/usr/share/openstack-tripleo-heat-templates/network/ports/noop.yaml

  OS::TripleO::RadosGW::Net::SoftwareConfig: /home/stack/templates/nic-configs/rgw.yaml
EOF
```

We ask director to connect our Rados Gateway Role (i.e nodes) to the following networks :

- Internal API
- Storage Access

Note: *Public Access to RGW is handled by HAProxy which by default is hosted on the controllers therefore there is no need to have a connection to the external network.*

Then the last line tells Director to use "/home/stack/templates/nic-configs/rgw.yaml" **to configure the NICs.**

There is no default pre-built nic config template shipped for RGW, for simplicity sake, we're going to use the swift-storage template and adjust it to our needs.

The only required change is to **remove the storage_mgmt interface** which is required for Swift but not for RGW.

Delete the appropriate lines so our RGW is not connected to the storage_mgmt network.

```
undercloud$ cd ~/templates/nic-configs/
undercloud$ sed -e '150,156d' swift-storage.yaml > rgw.yaml
```

If you're curious about which lines have been deleted, you can use diff to satisfy your curiosity

```
undercloud$ diff swift-storage.yaml rgw.yaml
150,156d149
<                   -
<                     type: vlan
<                     device: bond1
<                     vlan_id: {get_param: StorageMgmtNetworkVlanID}
<                     addresses:
<                       -
<                         ip_netmask: {get_param: StorageMgmtIpSubnet}
```

Note: *When doing such things in real life, we recommend to also update the description section at the beginning of the template file.*

OpenStack deployment

Our configuration is now complete, time to kick off the deployment, we provided a deployment script (~stack/templates/overcloud-deploy.sh) which **we need to modify to reflect the RGW addition.**

Edit ~stack/templates/overcloud-deploy.sh and **add the following lines at the end of the file :**

```
        -e ~/templates/rgw-network.yaml \
        -e ~/templates/rgw-flavor.yaml \
        -e /usr/share/openstack-tripleo-heat-templates/environments/ceph-radosgw.yaml \
        -r ~/templates/custom_roles.yaml
```

We include our two RGW templates to the environment (rgw-network.yaml and rgw-flavor.yaml) as well as the custom role file we created earlier.

Swift being the default object storage backend in Red Hat OSP 10, RGW is de facto disabled (set to OS::Heat::None). **The ceph-radosgw.yaml template enables RGW and disable all Swift services,** inclusion of this template is mandatory to use RGW.

Your deployment script should now looks like this - **double check it is the same as below :**

```
undercloud$  cat ~/templates/overcloud-deploy.sh
#!/bin/bash

exec openstack overcloud deploy \
        --templates --ntp-server 10.16.255.1 \
        --control-flavor control --control-scale 1 \
        --compute-flavor compute --compute-scale 1 \
        --ceph-storage-flavor ceph-storage --ceph-storage-scale 3 \
        --neutron-tunnel-types vxlan --neutron-network-type vxlan \
        -e /usr/share/openstack-tripleo-heat-templates/environments/network-isolation.yaml \
        -e ~/templates/storage-environment.yaml \
        -e ~/templates/network-environment.yaml \
        -e ~/templates/rgw-network.yaml \
        -e ~/templates/rgw-flavor.yaml \
        -e /usr/share/openstack-tripleo-heat-templates/environments/ceph-radosgw.yaml \
        -r ~/templates/custom_roles.yaml
```

Kick off the deployment :

```
undercloud$ screen
undercloud$ source ~/stackrc
undercloud$ cd ~
undercloud$ sh templates/overcloud-deploy.sh
Started Mistral Workflow. Execution ID: xxxx
Plan updated
Deploying templates in the directory /tmp/tripleoclient-xxxx/tripleo-heat-templates
```

This process takes some time, while you wait for the deployment to complete you can have a look at the ironic nodes being booted up (create a new “screen” window or use another terminal).

```
undercloud$ source ~/stackrc
undercloud$ openstack baremetal node list
+------------------------------+--------------------+------------------------------+-------------+--------------------+-------------+
| UUID                         | Name               | Instance UUID                | Power State | Provisioning State | Maintenance |
+------------------------------+--------------------+------------------------------+-------------+--------------------+-------------+
| b2c14304-7c0f-               | overcloud-ctrl01   | 6719079a-bf30-4def-b1da-     | power on    | deploying          | False       |
| 42a2-9546-74fb2d950776       |                    | 5f179c876a63                 |             |                    |             |
| 6821b53a-1c9c-409d-8b7d-     | overcloud-compute01 | fb15a764-9f14-4772-aba1-0cba9 | power on    | deploying          | False       |
| 6d252a74e32e                 |                    | 6bdf6af                      |             |                    |             |
| 86ba67ce-                    | overcloud-ceph01   | 21859eb5-aea6-4ff0-9853-59591 | power on    | wait call-back     | False       |
| daa6-4128-abe8-0e05c8d1fc10  |                    | 4326a78                      |             |                    |             |
```

```
| fac3d6bc-61a4-4546-afbc-     | overcloud-ceph02    | 3c006ffd-7168-4815-8b3e-      | power on    | deploying          | False       |
| 56d4f38b3ba0                 |                     | 7c7c94a44724                  |             |                    |             |
| 56096f50-1b1f-4279-8818-43a4 | overcloud-ceph03    | 8d848c9d-bd3d-4639-89db-      | power on    | deploying          | False       |
| 32d77ab5                     |                     | adc25faef9fa                  |             |                    |             |
| e2442bbc-0468-477b-          | overcloud-rgw       | b327de8f-e8a2-4dac-           | power on    | deploying          | False       |
| 81c2-cf99d8be3d62            |                     | 8bf3-751fb2e13f7b             |             |                    |             |
+------------------------------+---------------------+-------------------------------+-------------+--------------------+-------------+
```

Also have a look at the nova nodes being booted up

```
undercloud$ openstack server list
+--------------------------------------+-------------------------+--------+----------------------+----------------+
| ID                                   | Name                    | Status | Networks             | Image Name     |
+--------------------------------------+-------------------------+--------+----------------------+----------------+
| 21859eb5-aea6-4ff0-9853-595914326a78 | overcloud-cephstorage-2 | BUILD  | ctlplane=172.16.0.26 | overcloud-full |
| 8d848c9d-bd3d-4639-89db-adc25faef9fa | overcloud-cephstorage-0 | BUILD  | ctlplane=172.16.0.23 | overcloud-full |
| 6719079a-bf30-4def-b1da-5f179c876a63 | overcloud-controller-0  | BUILD  | ctlplane=172.16.0.27 | overcloud-full |
| 3c006ffd-7168-4815-8b3e-7c7c94a44724 | overcloud-cephstorage-1 | BUILD  | ctlplane=172.16.0.22 | overcloud-full |
| b327de8f-e8a2-4dac-8bf3-751fb2e13f7b | overcloud-rgw-0         | BUILD  | ctlplane=172.16.0.29 | overcloud-full |
| fb15a764-9f14-4772-aba1-0cba96bdf6af | overcloud-compute-0     | BUILD  | ctlplane=172.16.0.28 | overcloud-full |
+--------------------------------------+-------------------------+--------+----------------------+----------------+
```

When the deployment finishes, you should see something like :

```
Stack overcloud CREATE_COMPLETE

Started Mistral Workflow. Execution ID: xxxxx
/home/stack/.ssh/known_hosts updated.
Original contents retained as /home/stack/.ssh/known_hosts.old
Overcloud Endpoint: http://192.168.122.xxx:5000/v2.0
Overcloud Deployed
```

Check all nodes are up and deployed :

```
undercloud$ openstack baremetal node list
+------------------------------+---------------------+------------------------------+-------------+--------------------+-------------+
| UUID                         | Name                | Instance UUID                | Power State | Provisioning State | Maintenance |
+------------------------------+---------------------+------------------------------+-------------+--------------------+-------------+
| b2c14304-7c0f-               | overcloud-ctrl01    | 6719079a-bf30-4def-b1da-     | power on    | active             | False       |
| 42a2-9546-74fb2d950776       |                     | 5f179c876a63                 |             |                    |             |
| 6821b53a-1c9c-409d-8b7d-     | overcloud-compute01 | fb15a764-9f14-4772-aba1-0cba9 | power on    | active             | False       |
| 6d252a74e32e                 |                     | 6bdf6af                      |             |                    |             |
| 86ba67ce-                    | overcloud-ceph01    | 21859eb5-aea6-4ff0-9853-59591 | power on    | active             | False       |
| daa6-4128-abe8-0e05c8d1fc10  |                     | 4326a78                      |             |                    |             |
| fac3d6bc-61a4-4546-afbc-     | overcloud-ceph02    | 3c006ffd-7168-4815-8b3e-     | power on    | active             | False       |
| 56d4f38b3ba0                 |                     | 7c7c94a44724                 |             |                    |             |
| 56096f50-1b1f-4279-8818-43a4 | overcloud-ceph03    | 8d848c9d-bd3d-4639-89db-     | power on    | active             | False       |
| 32d77ab5                     |                     | adc25faef9fa                 |             |                    |             |
| e2442bbc-0468-477b-          | overcloud-rgw       | b327de8f-e8a2-4dac-          | power on    | active             | False       |
| 81c2-cf99d8be3d62            |                     | 8bf3-751fb2e13f7b            |             |                    |             |
+------------------------------+---------------------+------------------------------+-------------+--------------------+-------------+
```

```
undercloud$ openstack server list
+--------------------------------------+-------------------------+--------+----------------------+----------------+
| ID                                   | Name                    | Status | Networks             | Image Name     |
+--------------------------------------+-------------------------+--------+----------------------+----------------+
| 21859eb5-aea6-4ff0-9853-595914326a78 | overcloud-cephstorage-2 | ACTIVE | ctlplane=172.16.0.26 | overcloud-full |
| 8d848c9d-bd3d-4639-89db-adc25faef9fa | overcloud-cephstorage-0 | ACTIVE | ctlplane=172.16.0.23 | overcloud-full |
| 6719079a-bf30-4def-b1da-5f179c876a63 | overcloud-controller-0  | ACTIVE | ctlplane=172.16.0.27 | overcloud-full |
| 3c006ffd-7168-4815-8b3e-7c7c94a44724 | overcloud-cephstorage-1 | ACTIVE | ctlplane=172.16.0.22 | overcloud-full |
| b327de8f-e8a2-4dac-8bf3-751fb2e13f7b | overcloud-rgw-0         | ACTIVE | ctlplane=172.16.0.29 | overcloud-full |
```

```
| fb15a764-9f14-4772-aba1-0cba96bdf6af | overcloud-compute-0     | ACTIVE | ctlplane=172.16.0.28 | overcloud-full |
+--------------------------------------+-------------------------+--------+----------------------+----------------+
```

For simplicity sake, populate the undercloud hosts file so we can access the nodes by their names.

```
undercloud$ nova list --fields name,networks | awk '/overcloud/ { gsub("ctlplane=",""); print $6" "$4; }' | sudo tee -a /etc/hosts
```

Ping one of the overcloud's node for example the controller

```
undercloud$ ping overcloud-controller-0
PING overcloud-controller-0 (172.16.0.x) 56(84) bytes of data.
64 bytes from overcloud-controller-0 (172.16.0.x): icmp_seq=1 ttl=64 time=0.232 ms
```

What next ?

Check everything is properly deployed

OpenStack

Let's now verify that our overcloud is operational, first we are going to quickly check the core OpenStack services then focus on RGW.

List all running services :

```
undercloud$ source ~/overcloudrc
undercloud$ openstack service list
+----------------------------------+------------+----------------+
| ID                               | Name       | Type           |
+----------------------------------+------------+----------------+
| 21456c69580640de9a14b573c7de9a59 | heat-cfn   | cloudformation |
| 5f4f1c37a83541f59be3432cf4b172a4 | neutron    | network        |
| 623e4395e4854353872999fcaf8d8017 | nova       | compute        |
| 78a0809fdf6f4353ac2430a5f599a012 | cinder     | volume         |
| 8875fe6e065c477db787c2cd53f7362f | swift      | object-store   |
| 91777fc71ce949d49bddd3cd27a5ab4f | cinderv2   | volumev2       |
| 978f404f32fc496fabfe3fdd8db8cb6f | glance     | image          |
| bd63c0a7bca747f695772714a240f8ee | gnocchi    | metric         |
| c52bc9dbff174da0b07f41e76fc8b8f9 | ceilometer | metering       |
| c61af82e7613463d86bc0fcd43dda666 | cinderv3   | volumev3       |
| c72b6bf38e5647c8b86b48ab1ab78346 | keystone   | identity       |
| cd53b7a1262e48a39dcce221480d6188 | aodh       | alarming       |
```

```
| d22e8b9d1c0e4efe9e8d7761322172fa | heat       | orchestration  |
+----------------------------------+------------+----------------+
```

List all OpenStack endpoints :

```
undercloud$ openstack catalog list
+------------+----------------+-------------------------------------------------------------------
---------------+
| Name       | Type           | Endpoints
|
+------------+----------------+-------------------------------------------------------------------
---------------+
| nova       | compute        | regionOne
|
|            |                |   publicURL: http://192.168.122.107:8774/v2.1
|
|            |                |   internalURL: http://172.17.1.13:8774/v2.1
|
|            |                |   adminURL: http://172.17.1.13:8774/v2.1
|
```

(...)

Rados Gateway

In this section we will have a look at how Rados Gateway has been deployed.
Connect to the controller.

```
undercloud$ ssh heat-admin@overcloud-controller-0
```

In a traditional production deployment, HAProxy configuration balances VIPs 8080/tcp requests to all controller nodes on 8080/tcp internal IPs. In our case we only have one controller.

```
overcloud-controller-0$ sudo -i
overcloud-controller-0# grep "listen ceph_rgw" /etc/haproxy/haproxy.cfg -A3
listen ceph_rgw
  bind 172.17.3.13:8080 transparent
  bind 192.168.122.109:8080 transparent
  server overcloud-rgw-0.storage.localdomain 172.17.3.15:8080 check fall 5 inter 2000 rise 2
```

We can confirm that with :

```
overcloud-controller-0# ss -lnp | grep ":8080"
tcp    LISTEN     0      128    192.168.122.109:8080                 *:*
users:(("haproxy",pid=149466,fd=10))
```

```
tcp    LISTEN     0      128    172.17.3.13:8080                 *:*
users:(("haproxy",pid=149466,fd=9))
```

Log Out and connect to the rados gateway node:

```
undercloud$ ssh heat-admin@overcloud-rgw-0
```

Have a look at the RGW process :

```
overcloud-rgw-0$ sudo -i
overcloud-rgw-0# ps aux | grep radosgw
ceph       37012  0.0  0.7 2742588 30540 ?       Ssl  Apr13   1:09 /usr/bin/radosgw -f
--cluster ceph --name client.radosgw.gateway --setuser ceph --setgroup ceph
```

We have RGW running with client.radosgw.gateway as auth key :

```
overcloud-rgw-0# cat /etc/ceph/ceph.client.radosgw.gateway.keyring
[client.radosgw.gateway]
               key = AQCHtvtXAAAAABAAP9TMC7mc7HL/ovNmtqojZA==
               caps mon = "allow *"
               caps osd = "allow *"
```

RGW related configuration in /etc/ceph/ceph.conf :

```
[client.radosgw.gateway]
user = apache
rgw_frontends = civetweb port=172.17.3.15:8080
log_file = /var/log/ceph/radosgw.log
host = overcloud-rgw-0
keyring = /etc/ceph/ceph.client.radosgw.gateway.keyring
rgw_keystone_token_cache_size = 500
rgw_keystone_url = http://172.16.0.24:35357
rgw_s3_auth_use_keystone = True
rgw_keystone_admin_token = QD3Gw4dWGDUZuV92f7Rm2ZABz
rgw_keystone_accepted_roles = admin,_member_,Member
```

Create a testing environment

Openstack

The best way to quickly check if our Openstack and RGW are operational is to deploy a testing environment.

We start by importing our SSH public key into nova :

```
undercloud$ source ~/overcloudrc
undercloud$ openstack keypair create --public-key ~/.ssh/id_rsa.pub rhsummit
undercloud$ openstack keypair list
+----------+-------------------------------------------------+
| Name     | Fingerprint                                     |
+----------+-------------------------------------------------+
| rhsummit | 20:da:1e:c4:8f:ba:01:b4:89:0c:72:cb:7c:e6:de:86 |
+----------+-------------------------------------------------+
```

Create the “admin stack”, this stack creates, networks, flavor, image, user, etc

```
undercloud$ openstack stack create -t ~/heat/admin.yaml stack_admin --wait
```

Once the stack has successfully terminated, list the newly created subnets.

```
undercloud$ openstack network list
+--------------------------------------+----------+--------------------------------------+
| ID                                   | Name     | Subnets                              |
+--------------------------------------+----------+--------------------------------------+
| d29b743c-3a20-47e0-a70a-05abe573f5d1 | external | fa72ad95-78b2-4dbf-a337-056b7fab6016 |
| d5656f5f-d492-4be3-9268-fb9f1e33d73c | net1     | be7a34fb-3887-4512-ba29-eadef75008d9 |
+--------------------------------------+----------+--------------------------------------+
```

Here we have

- One external subnet (no tunneling / used for floating IPs)
- One tenant network - “net1” (VXLAN tunneling / private instance network)

Check our flavor type has been created

```
undercloud$ openstack flavor list
+--------------------------------------+----------+------+------+-----------+-------+-----------+
| ID                                   | Name     |  RAM | Disk | Ephemeral | VCPUs | Is Public |
+--------------------------------------+----------+------+------+-----------+-------+-----------+
| 88d59257-9a42-48d2-95e9-4608b77814ac | m1.small | 1024 |    0 |         1 |     1 | True      |
+--------------------------------------+----------+------+------+-----------+-------+-----------+
```

Create the “user stack”, this stack creates, one instance and attach a floating IP to it.

```
undercloud$ openstack stack create -t ~/heat/user.yaml stack_user --wait
```

Note: *In order to reduce the image size of the lab we use a qcow2 image which slows down the instance boot process. On live environments you should use raw images if you use Ceph as backend.*

```
undercloud$ openstack server list
+--------------------------------------+------+--------+----------------------------------+------------+
| ID                                   | Name | Status | Networks                         | Image Name |
+--------------------------------------+------+--------+----------------------------------+------------+
| 4f91fbc9-e4c7-429f-b1f3-96871229b9a5 | vm1  | ACTIVE | net1=192.168.1.20, 192.168.122.161 | centos7    |
+--------------------------------------+------+--------+----------------------------------+------------+
```

Test the instance’s connectivity by pinging its floating IP (192.168.122.x)

```
undercloud$ ping 192.168.122.x
```

Ceph Rados Gateway

In this section we will show you how to use RadosGW from an end-user perspective. We will cover both Swift and S3 APIs.

Swift API

First take note of your overcloud external API URL, you will need it in few moments

```
undercloud$  grep OS_AUTH_URL overcloudrc
export OS_AUTH_URL=http://192.168.122.xxx:5000/v2.0
```

Next connect to the instance :

```
undercloud$ source ~/overcloudrc
undercloud$ openstack server list
+--------------------------------------+------+--------+----------------------------------+------------+
| ID                                   | Name | Status | Networks                         | Image Name |
+--------------------------------------+------+--------+----------------------------------+------------+
| 4f91fbc9-e4c7-429f-b1f3-96871229b9a5 | vm1  | ACTIVE | net1=192.168.1.20, 192.168.122.161 | centos7    |
+--------------------------------------+------+--------+----------------------------------+------------+
undercloud$ ssh centos@192.168.122.x
```

Get root privileges and jump into the swift directory and edit the `userrc` file:

```
vm1$ sudo -i
vm1$ cd swift
vm1$ vi userrc
```

And replace `KEYSTONE_ENDPOINT_ADDRESS` with the IP address of the Keystone public endpoint. Look at the beginning of this section on how to find it.

Then execute the `swift-exercise.sh` script:

```
vm1$ ./swift-exercise.sh
```

This will execute a bunch of commands against the Ceph Rados Gateway using the Swift API. Have a look at the script, it contains commands along with comments on what does what !

Now, logout from your guest virtual machine:

```
vm1$ logout
```

S3 API

Since we want to test the S3 API from Ceph Rados Gateway, we need to create a user on it, for this log on the Ceph Rados Gateway machine and create the user:

```
undercloud$ ssh heat-admin@overcloud-rgw-0

overcloud-rgw-0$ sudo -i

overcloud-rgw-0# radosgw-admin user create --uid=user1 --display-name="Ceph demo user"
--access-key=G1EZ5R4K6IJ7XUQKMAED --secret-key=cNmUrqpBKjCMzcfqG8fg4Qk07Xkoyau52OmvnSsz

{
    "user_id": "user1",
    "display_name": "Ceph demo user",
    "email": "",
    "suspended": 0,
    "max_buckets": 1000,
    "auid": 0,
    "subusers": [],
    "keys": [
        {
            "user": "user1",
            "access_key": "G1EZ5R4K6IJ7XUQKMAED",
            "secret_key": "cNmUrqpBKjCMzcfqG8fg4Qk07Xkoyau52OmvnSsz"
```

```
        }
    ],
    "swift_keys": [],
    "caps": [],
    "op_mask": "read, write, delete",
    "default_placement": "",
    "placement_tags": [],
    "bucket_quota": {
        "enabled": false,
        "max_size_kb": -1,
        "max_objects": -1
    },
    "user_quota": {
        "enabled": false,
        "max_size_kb": -1,
        "max_objects": -1
    },
    "temp_url_keys": []
}
```

Logout back to your undercloud and take note of the address of your Swift API, using your `overcloudrc` credentials:

```
undercloud$ source ~/overcloudrc
undercloud$ openstack catalog list | grep swift | grep public
 publicURL: http://192.168.122.109:8080/swift/v1
```

Go back into your guest virtual machine again and become root:

```
undercloud$ ssh centos@192.168.122.x
vm1$ sudo -i
```

Let's test the S3 API now, edit the .s3cfg file

```
vm1# vi .s3cfg
```

Change these two options with the Swift `publicURL` we just got above:

```
host_base = SWIFT_ENDPOINT_ADDRESS:8080
host_bucket = SWIFT_ENDPOINT_ADDRESS:8080
```

Now, change directory and execute the `s3-exercise.sh` :

```
vm1# cd s3/
vm1# ./s3-exercise.sh
```

Have a look at the script, it contains commands along with comments on what does what !

Thank you !

That’s all ! Thanks a lot for attending our session!

We hope it met your expectations and you learnt how you can use Red Hat OpenStack director to leverage Red Hat Ceph Storage as a replacement for Swift!

Don’t hesitate to come meet us and ask anything.

Have a great day and enjoy summit !

Sebastien Han - Red Hat Ceph Storage Engineering
Gregory Charot - Red Hat Openstack Platform Field Product Management
Cyril Lopez - Red Hat Consulting

https://www.redhat.com/en/topics/openstack
https://www.redhat.com/en/technologies/linux-platforms/openstack-platform
https://www.redhat.com/en/technologies/storage/ceph