Private Cloud Setup Guide

Adapted from OpenStack Installation Manual

Advanced Sensing Computation and Control (ASCC) Lab
Oklahoma State University
May 19, 2016

Contents

1. Preface

This guide is prepared to serve as a manual for building a private cloud infrastructure with a minimal three node architecture. OpenStack Kilo project will be used on top of Ubuntu Trusty 14.04 LTS Server to setup the cloud.

Before proceeding with the installation, make sure your environment satisfies the hardware and network requirements presented in section 3 of this manual. This guide references the installation procedures on the OpenStack Kilo Installation Guide for building and configuring each service. The OpenStack Configuration Reference and Operations Guide can be used as a reference for customizing your deployment. Please follow the ASCC Cloud User manual to learn on how to use the cloud services. The links for the resources used to prepare this guide are provided in the resources section.

2. Overview

The *OpenStack* project is an open source cloud computing platform that supports all types of cloud environments. The project aims for simple implementation, massive scalability, and a rich set of features. OpenStack provides an Infrastructure-as-a-Service (*IaaS*) solution through a variety of complemental services. Each service offers an application programming interface (*API*) that facilitates this integration. The following table provides a list of the OpenStack services:

Service	**Project name**	**Description**
Dashboard	*Horizon*	Provides a web-based self-service portal to interact with underlying OpenStack services, such as launching an instance, assigning IP addresses and configuring access controls.
Compute	*Nova*	Manages the lifecycle of compute instances in an OpenStack environment. Responsibilities include spawning, scheduling and decommissioning of virtual machines on demand.
Networking	*Neutron*	Enables Network-Connectivity-as-a-Service for other OpenStack services, such as OpenStack Compute. Provides an API for users to define networks and the attachments into them. Has a pluggable architecture that supports many popular networking vendors and technologies.
		Storage
Object Storage	*Swift*	Stores and retrieves arbitrary unstructured data objects via a *RESTful*, HTTP based API. It is highly fault tolerant with its data replication and scale-out architecture. Its implementation is not like a file server with mountable directories. In this case, it writes objects and files to multiple drives, ensuring the data is replicated across a server cluster.
Block Storage	*Cinder*	Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.
		Shared services
Identity service	*Keystone*	Provides an authentication and authorization service for other OpenStack services. Provides a catalog of endpoints for all OpenStack services.
Image service	*Glance*	Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.
Telemetry	*Ceilometer*	Monitors and meters the OpenStack cloud for billing, benchmarking, scalability, and statistical purposes.
		Higher-level services
Orchestration	*Heat*	Orchestrates multiple composite cloud applications by using either the native *HOT* template format or the AWS CloudFormation template format, through both an OpenStack-native REST API and a CloudFormation-compatible Query API.
Database service	*Trove*	Provides scalable and reliable Cloud Database-as-a-Service functionality for both relational and non-relational database engines.
Data processing service	*Sahara*	Provides capabilties to provision and scale Hadoop clusters in OpenStack by specifying parameters like Hadoop version, cluster topology and nodes hardware details.

3. Architecture

OpenStack is highly configurable to meet different needs with various compute, networking, and storage options. This guide uses a Three-node architecture with legacy networking (nova-network). A controller node and two compute nodes are used in the configuration.

The ***controller node*** runs the Identity service, Image service, management portion of Compute, and the dashboard. It also includes supporting services such as a SQL database, *message queue*, and *Network Time Protocol (NTP)*.

In addition, the controller node also serves as the Block storage node which provides block storage services for virtual machine instances.

The ***compute nodes*** run the *hypervisor* portion of Compute that operates *tenant virtual machines* or instances. By default, Compute uses *KVM* as the *hypervisor*. Compute also provisions tenant networks and provides firewalling (*security groups*) services.

3.1. Minimal Architecture Hardware Requirements

For best performance, it is recommended that your environment meets or exceeds the following hardware requirements for each node.

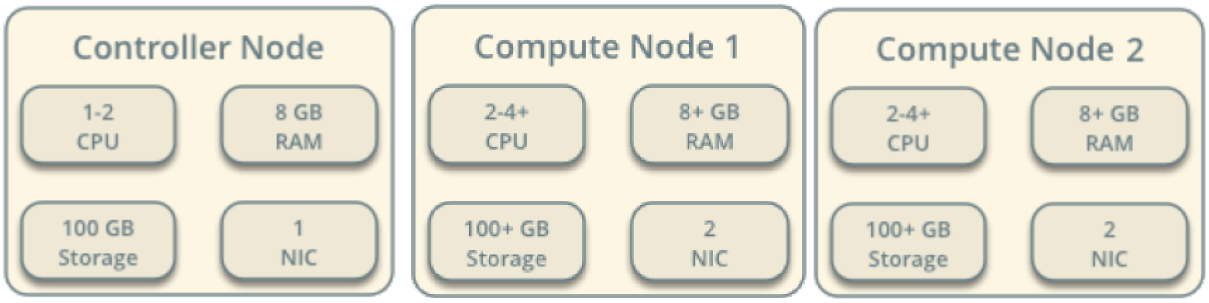


Fig.1. Minimal Architecture Hardware Requirements

To minimize clutter and provide more resources for OpenStack, a minimal installation of the Linux distribution is recommended. Also, it is strongly recommend that a 64-bit version of the distribution is installed on at least the compute nodes.

A single disk partition on each node works well for the minimal architecture in this guide. To achieve this we recommend the following RAID configurations.

Controller node

Note: This guide assumes 6 HDD storges of 2TB each on the controller node.

The first drive will be used for the controller and should be configured as a primary drive. The remaining drives should be configured in RAID-0 for use with Logical Volume Manager (LVM) for Block Storage service.

Configuration Summary

Drive 1 = 2 TB Primary

Drive 2 = RAID 0 (5 X 2TB) = 10TB Logical

Recent build of Ubuntu Server 14.04 Trusty amd-64 should be installed on Drive 1 and the 2TB primary storage can be partitioned as follows

- 250 MB /Boot partition
- 128 GB /Swap partition
- Remaining /root partition

The logical volume from Drive 2 will be used with LVM to setup block storage

Compute nodes

Note: This guide assumes 2 HDD storges of 2TB each on compute nodes.

The drives should be configured in RAID-0 for use as a primary drive

Configuration Summary

Drive 1 = RAID 0 (2 X 2TB) = 4TB

Recent build of Ubuntu Server 14.04 Trusty amd-64 should be installed on Drive 1 and the 4TB primary storage can be partitioned as follows

- 250 MB /Boot partition

- 128 GB /Swap partition
- Remaining /root partition

3.2. Minimal Architecture – Network Layout Legacy Networking (nova-network)

The provided architecture with legacy networking (nova-network) uses a controller node and two compute nodes. The controller node contains one network interface on the *management network*. The compute nodes contain one network interface on the management network and one on the *external network*. The architecture assumes use of the following networks:

Management on 10.0.0.0/24 with gateway 10.0.0.1

Note: This network requires a gateway to provide Internet access to all nodes for administrative purposes such as package installation, security updates, *DNS*, and *NTP*.

External on 203.0.113.0/24 with gateway 203.0.113.1

Note: This network requires a gateway to provide Internet access to instances in the OpenStack environment.

These ranges and gateways should be modified to work with your particular network infrastructure.

Note: Network interface names vary by distribution, use the following command to check your interface name before configuring.

```
$ hwinfo --short
```

This manual uses the interface naming “eth” followed by a sequential number as used by many platforms.

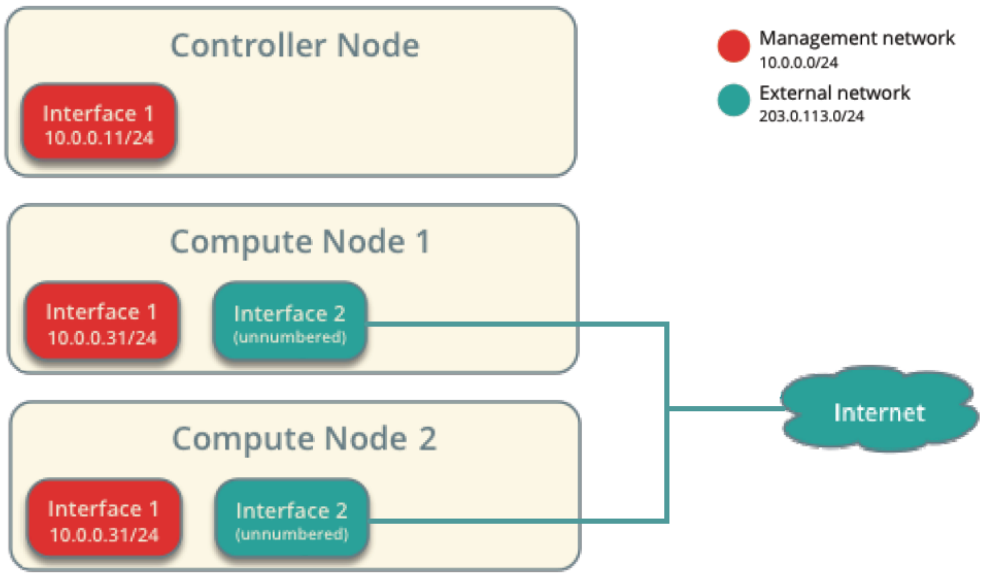


Fig. 2. Minimal Architecture Network Layout

Follow the OpenStack-kilo-install-guide pp 22 – 25 to configure network interfaces of the controller and compute nodes.

3.3. Minimal Architecture – Service Layout Legacy Networking (nova-network)

The Openstack service layout for the three-node minimal architecture using legacy network is as follows. All the database, management, API and Block Storage services are implemented on the controller node. The compute nodes run the compute hypervisor and compute networking services.

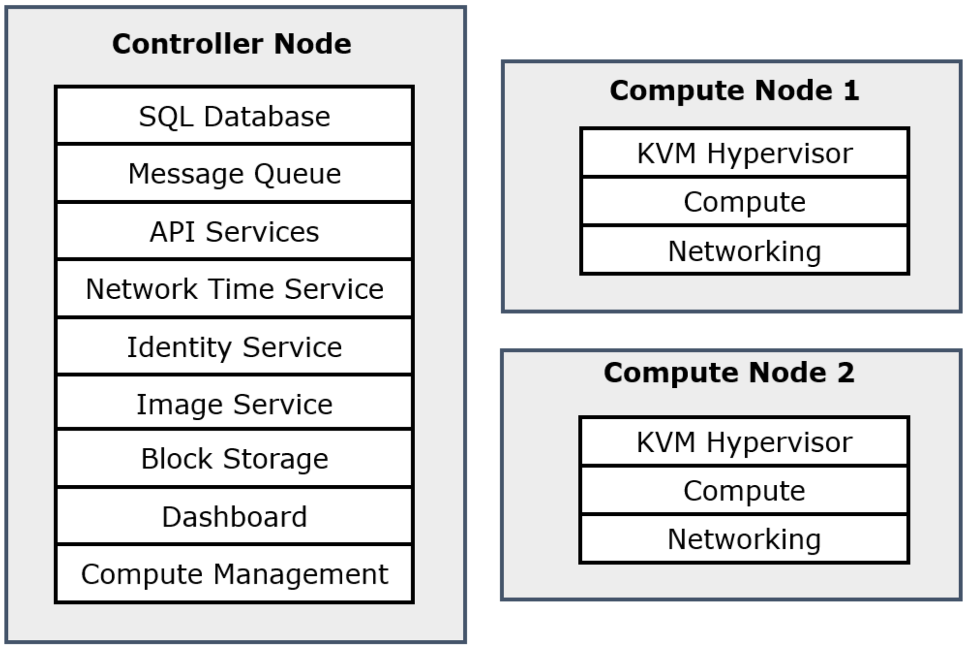


Fig. 3. Minimal Architecture Service Layout

4. Basic Environment

Follow the OpenStack-kilo-install-guide (pp 25 – 30) to install NTP service, OpenStack Packages, SQL database and Message Queue service

5. Identity Service

Follow the OpenStack-kilo-install-guide (pp 31 – 44) to install the OpenStack Identity service – Keystone.

The OpenStack *Identity Service* performs the following functions:

- Tracking users and their permissions.
- Providing a catalog of available services with their API endpoints.

When installing OpenStack Identity service, each service must be registered in the Open-Stack installation. Identity service can then track which OpenStack services are installed, and where they are located on the network.

The following are key terms used in OpenStack Identity service:

Token: An alpha-numeric string of text used to access OpenStack APIs and resources. A token may be revoked at any time and is valid for a finite duration.
Tenant: A container used to group or isolate resources. Tenants also group or isolate identity objects. Depending on the service operator, a tenant may map to a customer, account, organization, or project.
Service: An OpenStack service, such as Compute (nova), Object Storage (swift), or Image service (glance). It provides one or more endpoints in which users can access resources and perform operations.
Endpoint: A network-accessible address where you access a service, usually a URL address. If you are using an extension for templates, an endpoint template can be created, which represents the templates of all the consumable services that are available across the regions.
Role: A personality with a defined set of user rights and privileges to perform a specific set of operations. In the Identity service, a token that is issued to a user includes the list of roles. Services that are being called by that user determine how they interpret the set of roles a user has and to which operations or resources each role grants access.
Keystone Client: A command line interface for the OpenStack Identity API. For example, users can run the **keystone service-create** and **keystone endpoint-create** commands to register services in their OpenStack installations.

6. Image Service

Follow the OpenStack-kilo-install-guide (pp 45 - 52) to install the OpenStack Image Service – Glance.

The OpenStack Image service (glance) enables users to discover, register, and retrieve virtual machine images. It offers a *REST* API that enables you to query virtual machine image metadata and retrieve an actual image. This guide describes configuring the Image service to use the file back-end, which uploads and stores in a directory on the controller node hosting the Image service. By default, this directory is /var/lib/glance/images/.

6.1. Obtain Images

The simplest way to obtain a virtual machine image that works with OpenStack is to download one that is already created. Most of the images contain the cloud-init package to support SSH key pair and user data injection. Because many of the images disable SSH password authentication by default, boot the image with an injected key pair. You can then SSH into the instance with the private key and default login account.

We recommend using the following cloud-images.

6.1.1. CirrOS (test) image

CirrOS is a minimal Linux distribution that was designed for use as a test image on clouds such as OpenStack Compute. You can download a CirrOS image in various formats from the CirrOS download page.

In a CirrOS image, the login account is “`cirros`” and the password is “`cubswin:)`”.

6.1.2. CentOS image

CentOS is a widely used enterprise server and official images can be downloaded from the following link CentOS 7 images .

In a CentOS cloud image, the login account is “`centos`”.

6.1.3. Ubuntu image

Ubuntu is another popular linux distribution based on debian. Recent builds for Ubuntu 14.04 Trusty Tahr LTS server can be downloaded from http://cloud-images.ubuntu.com/trusty/current/ .

In an Ubuntu cloud image, the login account is "`Ubuntu`".

7. Compute Service

Follow the OpenStack-kilo-install-guide (pp 53 - 65) to install the OpenStack Compute Service – Nova.

OpenStack Compute is used to host and manage cloud computing systems. OpenStack Compute is a major part of an Infrastructure-as-a-Service (IaaS) system. The main modules are implemented in Python.

OpenStack Compute interacts with OpenStack Identity for authentication, OpenStack Image service for disk and server images, and OpenStack dashboard for the user and administrative interface. Image access is limited by projects, and by users; quotas are limited per project (the number of instances, for example). OpenStack Compute can scale horizontally on standard hardware, and download images to launch instances.

8. Networking Service

Follow the OpenStack-kilo-install-guide (pp 91 - 93) to install the OpenStack Nova-Network Service.

The network controller with nova-network provides virtual networks to enable compute servers to interact with each other and with the public network. Compute with nova-network only supports Linux Bridge networking that allows virtual interfaces to connect to the outside network through the physical interface. Compute with nova-network supports the following network modes, which are implemented as Network Managers:

- Flat Network Manager
- Flat DHCP Network Manager
- VLAN Network Manager

A network manager defines the network topology for a given OpenStack deployment. FlatManager and FlatDHCPManager have lots in common. They both rely on the concept of bridged networking, with a single bridge device.

For each compute node, there is a single virtual bridge created, the name of which is specified in the Nova configuration file using this option:

```
flat_network_bridge = br100
```

All the VMs spawned by OpenStack get attached to this dedicated bridge.

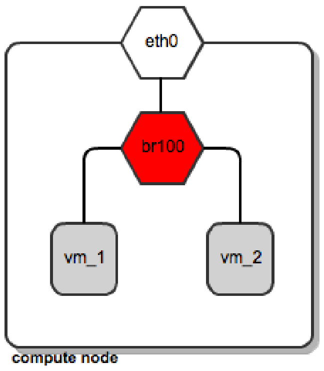


Fig. 4. Network Bridging on OpenStack Compute node

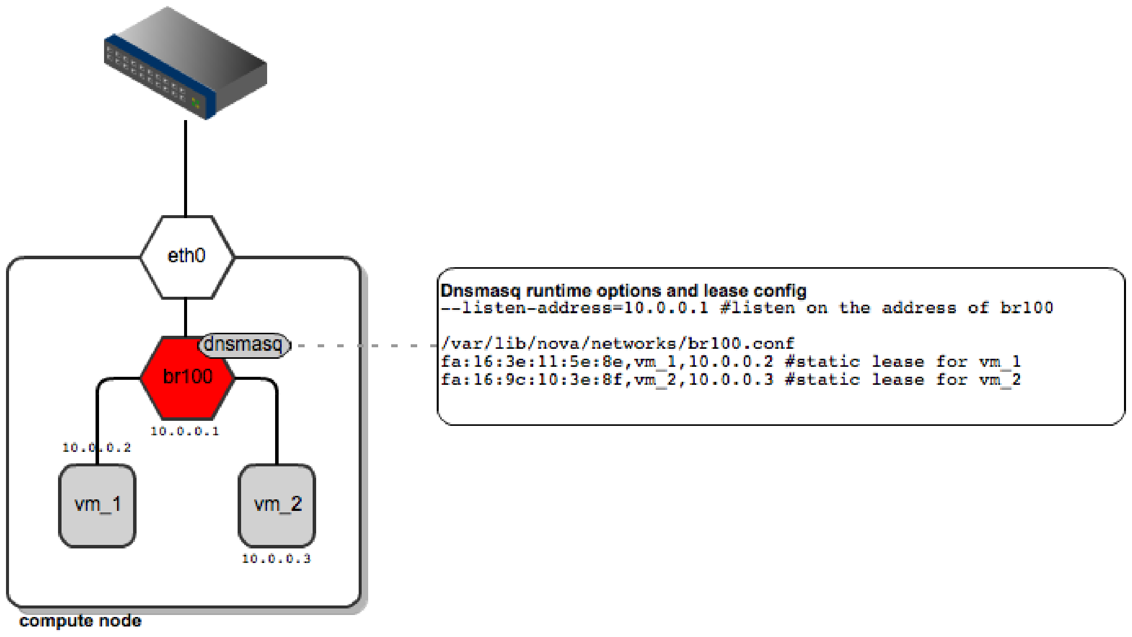
The idea behind FlatManager and FlatDHCPManager is to have one “flat” IP address pool defined throughout the cluster. This address space is shared among all user instances, regardless of which tenant they belong to. Each tenant is free to grab whatever address is available in the pool. We will be using FlatDHCPManager for the minimal architecture in this guide.

FlatDHCPManager plugs a given instance into the bridge, and on top of that provides a DHCP server to boot up from.

On each compute node:

- the network bridge is given an address from the “flat” IP pool
- a dnsmasq DHCP server process is spawned and listens on the bridge interface IP

- the bridge acts as the default gateway for all the instances running on the given compute node



Dnsmasq runtime options and lease config

```
--listen-address=10.0.0.1 #listen on the address of br100

/var/lib/nova/networks/br100.conf
fa:16:3e:11:5e:8e,vm_1,10.0.0.2 #static lease for vm_1
fa:16:9c:10:3e:8f,vm_2,10.0.0.3 #static lease for vm_2
```

Fig. 5. FlatDHCPManager Network Topology

As for dnsmasq, FlatDHCPManager creates a static lease file per compute node to guarantee the same IP address for the instance over time. The lease file is constructed based on instance data from the Nova database, namely MAC, IP and hostname. The dnsmasq server is supposed to hand out addresses only to instances running locally on the compute node. To achieve this, instance data to be put into DHCP lease file are filtered by the 'host' field from the 'instances' table. Also, the default gateway option in dnsmasq is set to the bridge's IP address. The diagram below shows the instance will be given a different default gateway depending on which compute node it lands.

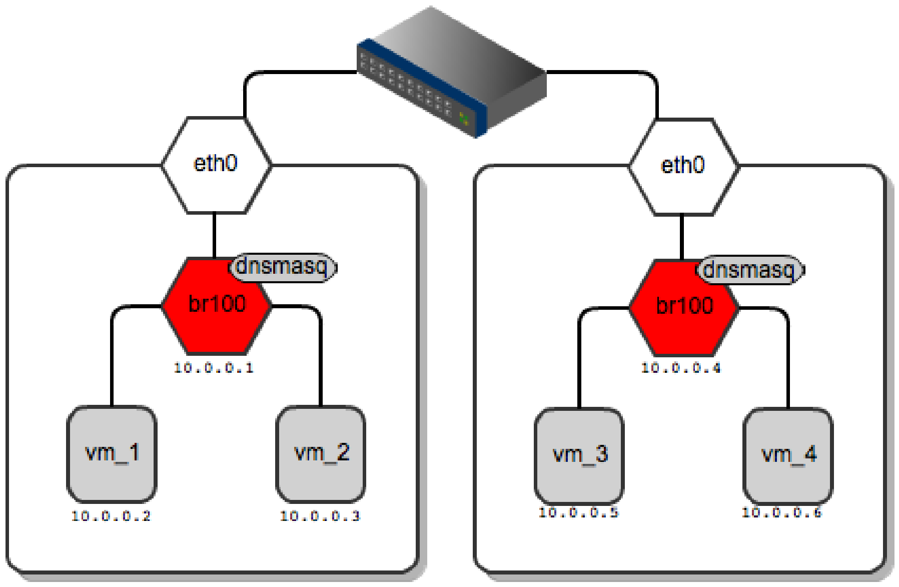


Fig. 6. Network Gateways for instances running on different compute nodes

Below I've shown the routing table from vm_1 and for vm_3 – each of them has a different default gateway:

```
root@vm_1:~# route -n
Kernel IP routing table
Destination     Gateway     Genmask Flags Metric Ref Use Iface
0.0.0.0         10.0.0.1     0.0.0.0 UG     0   0   0 eth0

root@vm_3:~# route -n
Kernel IP routing table
Destination     Gateway     Genmask Flags Metric Ref Use Iface
0.0.0.0         10.0.0.4     0.0.0.0 UG     0   0   0 eth0
```

By default, all the VMs in the "flat" network can see one another regardless of which tenant they belong to. Check the following flag in nova.conf to confirm instance visibility:

```
allow_same_net_traffic = True
```

If this is set to False, it configures IPtables policies to prevent any traffic between instances (even inside the same tenant), unless it is unblocked in a security group.

9. Dashboard

Follow the OpenStack-kilo-install-guide (pp 94 - 96) to install the OpenStack Dashboard Service Horizon.

9.1. Customize the dashboard

Once you have the dashboard installed you can customize the way it looks and feels to suit your own needs.

Note: The OpenStack dashboard by default on Ubuntu installs the `openstack-dashboard-ubuntu-theme` package. If you do not want to use this theme you can remove it and its dependencies using the following command:

```
# apt-get remove --auto-remove openstack-dashboard-ubuntu-theme
```

9.1.1. Logo and site colors

1. Create two logo files, png format, with transparent backgrounds using the following sizes:
   - Login screen: 365 x 50
   - Logged in banner: 216 x 35
2. Upload your new images to the following location: `/usr/share/openstack-dashboard/openstack_dashboard/static/dashboard/img/`
3. Create a CSS style sheet in the following directory: `/usr/share/openstack-dashboard/openstack_dashboard/static/dashboard/css/`
4. 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;
}
```

1. Open the following HTML template in an editor of your choice: `/usr/share/openstack-dashboard/openstack_dashboard/templates/_stylesheets.html`
2. Add a line to include your newly created style sheet. For example `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' />
    ...
```

Restart Apache:

```
# service apache2 restart
```

If the css file cannot update the logo, use the following commands to compress and update the logo,

```
apt-get install python-lesscpy
/usr/share/openstack-dashboard/manage.py compress
/usr/share/openstack-dashboard/manage.py collectstatic
```

9.1.2. HTML title

1. Set the HTML title, which appears at the top of the browser window, by adding the following line to `local_settings.py`:

   ```
   SITE_BRANDING = "Example, Inc. Cloud"
   ```

2. Restart Apache for this change to take effect.

9.1.3. Site Branding Link

1. The logo also acts as a hyperlink. The default behavior is to redirect to horizon: user_home. To change this, add the following attribute to `local_settings.py`

   ```
   SITE_BRANDING_LINK = http://example.com
   ```

2. Restart Apache for this change to take effect.

9.1.4. Help URL

1. By default the help URL points to `http://docs.openstack.org`. Change this by editing the following arritbute to the URL of your choice in `local_settings.py`

   ```
   'help_url': "http://openstack.mycompany.org",
   ```

2. Restart Apache for this change to take effect.

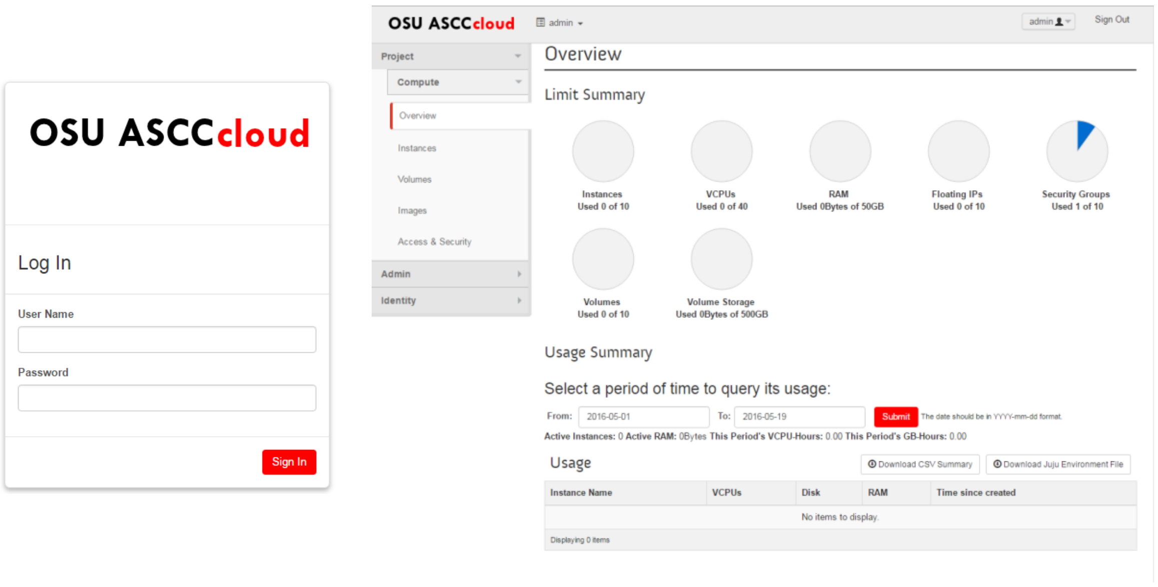


Fig. 7. Customized OpenStack Dashboard

10. Block Storage Service

Follow the OpenStack-kilo-install-guide (pp 97 - 107) to install the OpenStack Block Storage Service – Cinder.

Note: the controller node also serves as a storage node in this minimal architecture, therefore all block storage services should both be implemented on the controller node.

Note: Remember to change the storage node management interface in the OpenStack-kilo-install-guide to the management interface of the controller node.

The following subsection explains how a Logical Volume Manager LVM works, which is going to be used for setting up the block storage.

10.1. Linux LVM - Logical Volume Manager

Multiple physical volumes can be created on a physical hard disk. A volume group can then be created from these physical volumes and finally multiple logical volumes can be created using the logical volume manager depending on the need. The following figure shows the partitioning hierarchy.

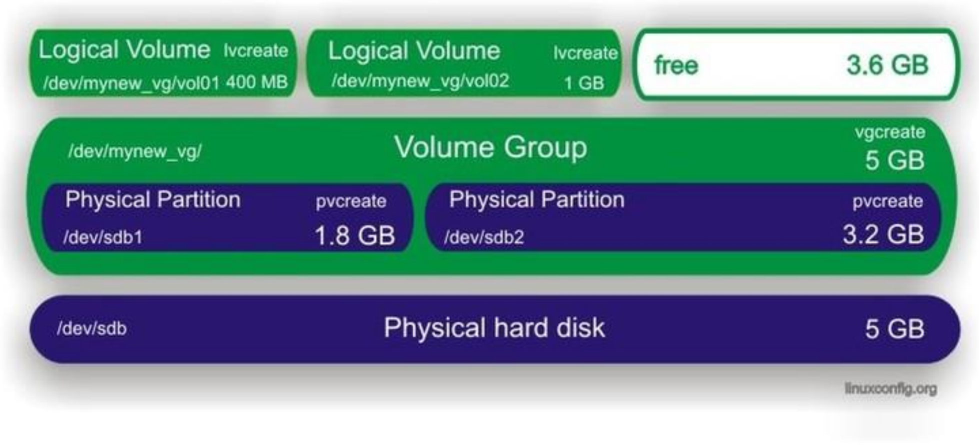


Fig. 8. Linux Logical Volume Manager Partitioning

Before creating partitions check your disk configuration using the following command

```
# fdisk –l
```

This should display the partitions on your disks. On the controller `/dev/sda` is the primary drive for the controller node and `dev/sdb` is the volume that we setup for block storage in section 2. `/dev/sdb` should have a single partition `/dev/sdb1` with the full storage size.

Create a physical volume and volume group as shown in the installation guide.

```
# pvcreate /dev/sdb1
Physical volume "/dev/sdb1" successfully created

# vgcreate cinder-volumes /dev/sdb1
Volume group "cinder-volumes" successfully created
```

Remember to use the naming “cinder-volumes” for the volume group, otherwise the naming configuration must be updated in the cinder.conf file.

Launch an Instance

Follow the OpenStack-kilo-install-guide (pp 153 - 159) to launch an instance using the command line.

11. Resources

OpenStack Kilo Installation Guide

http://docs.openstack.org/kilo/install-guide/install/apt/content/

OpenStack Kilo Configuration Reference

http://docs.openstack.org/kilo/config-reference/content/

OpenStack Kilo Operations Guide

http://docs.openstack.org/openstack-ops/content/

OpenStack Nova Network Flatmanager and FlatDHCPmanager

https://www.mirantis.com/blog/openstack-networking-flatmanager-and-flatdhcpmanager/

Linux Logical Volume Manager

https://linuxconfig.org/linux-lvm-logical-volume-manager

Custom Brand OpenStack Horizon Dashboard

https://www.prestonlee.com/2012/05/09/how-to-custom-brand-the-openstack-horizon-dashboard/