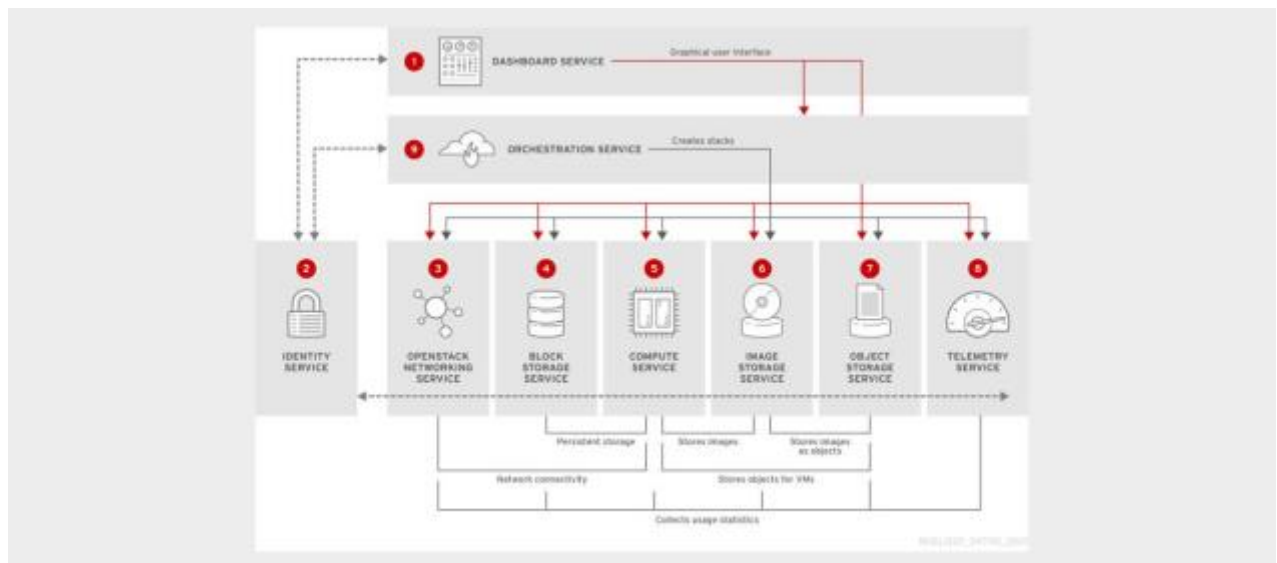


OPENSTACK 7th BIRTHDAY MEETUP DEMO/LAB

LAUNCHING AN INSTANCE USING THE OPENSTACK HORIZON DASHBOARD

INTRODUCTION

OPENSTACK ARCHITECTURE



The OpenStack platform is implemented as a collection of interacting services that control compute, storage, and networking resources.

OpenStack services and terminology:

- **Horizon (dashboard):** A web-based interface for managing OpenStack services. It provides a graphical user interface for operations such as launching instances, managing networking, and setting access controls.
- **Keystone (identity):** A centralized identity service that provides authentication and authorization for other services. Keystone also provides a central catalog of services running in a particular OpenStack cloud. It supports multiple forms of authentication, including user name and password credentials, token-based systems, and Amazon Web Services (AWS)-styled logins. Keystone acts as a single sign-on (SSO)

authentication service for users and components. This service is responsible for creating and managing users, roles, and projects/tenants

- **Users:** The Keystone service validates that incoming requests are made by legitimate users and may be assigned a token to access particular resources based on their role(s) in the project.
- **Projects:** A project in Keystone is equivalent to a tenant in previous versions of The OpenStack Platform. A project is a group of items (users, images, instances, network(s), volumes, etc.). It helps isolate or group identity objects. Depending on the service provider, a project (or tenant) can map to a customer, account, organization, or environment.
- **Neutron (network):** A software-defined networking service that helps to create networks, subnets, routers, and floating IP addresses. Administrators can create and attach interface devices managed by other OpenStack services to networks. OpenStack networking ships with plug-ins and agents for Cisco virtual and physical switches, Open vSwitch, and others. The common agents are L3 and Dynamic Host Configuration Protocol (DHCP), which provides DHCP IP addresses to instances. OpenStack networking enables projects (or tenants) to create advanced virtual network topologies, including services such as firewalls, load balancers, and virtual private networks (VPNs). The public and private networks, and floating IP address, were provided by the Neutron service.
- **Cinder (block storage):** A service that manages storage volumes for virtual machines. This is persistent block storage for the instances running in Nova. Snapshots can be taken for backing up data, either for restoring data or to be used to create new block storage volumes. This is often used in instances for storage such as database files.
- **Nova (compute):** A service that manages networks of virtual machines running on nodes, providing virtual machines on demand. Nova is a distributed component and interacts with Keystone for authentication, Glance for images, and Horizon for web interface. Nova is designed to scale horizontally on standard hardware, downloading images to launch instances as required. Nova compute uses libvirt, qemu, and kvm for the hypervisor.
- **Glance (image):** A service that acts as a registry for virtual machine images, allowing users to copy server images for immediate storage. These images can be used as templates when setting up new instances.
- **OpenStack Networking:** A service that provides connectivity between the interfaces of other OpenStack services, such as Nova. Due to OpenStack Networking's pluggable architecture, users can create their own networks, control traffic, and connect servers to other networks. Various networking technologies are supported.

OpenStack core services that are not part of today's lab:

- **Swift (object storage):** A service that provides object storage that allows users to store and retrieve files. Swift architecture is distributed to allow for horizontal scaling and to provide redundancy as failure-proofing.
- **Heat (orchestration):** A service to orchestrate multiple composite cloud applications using the AWS CloudFormation template format, through both a representational state transfer (REST) application programming interface (API) and a CloudFormation-compatible query API
- **Ceilometer (telemetry):** The OpenStack telemetry service provides user-level usage data, which can be used for customer billing, system monitoring, or alerts. It can collect data from notifications sent by OpenStack services like compute usage events, or by polling OpenStack infrastructure resources. Additionally, the service provides a plug-in system that can be used to add new monitoring metrics.

- **Gnocchi (time series database):** Gnocchi is a Time-series Database-as-a-Service (TDBaaS), which is used to store metrics and metadata for the resources collected by Ceilometer into this database. It uses Swift for its storage.
- **Trove (relational database):** Trove is a Database-as-a-Service (DBaaS). Its mission is to provide scalable and reliable relational database engines. The service provides isolation at high performance while automating complex database administrative tasks, including deployment, configuration, patching, backups, restores, and monitoring
- **Manila (shared file storage):** Manila is a secure file share as a service. It uses Network File System (NFS) and Common Internet File System (CIFS) protocols for sharing the files. It can be configured to run on a single-node back end or across multiple nodes.
- **Sahara (data processing):** Sahara aims to provide users with a simple means to provision a data processing cluster (such as Hadoop, Spark, and Storm) on OpenStack.
- **TripleO or OpenStack on OpenStack (OOO).** TripleO is used in installing, upgrading, and operating OpenStack clouds using OpenStack's own services as the foundations. It uses Nova, Neutron, and Heat, and other orchestration tools, like Chef or Puppet, to automate fleet management, including scaling up and down at datacenter scale.
- **Ironic (bare-metal provisioning):** Ironic is an OpenStack project that provisions physical hardware as opposed to virtual machines. It provides several drivers like PXE and IPMI to cover a wide range of hardware. It also allows vendor-specific drivers to be added.
- **Tempest (integration testing):** Tempest provides a set of integration tests to be run against a live OpenStack cluster. It has several tests for OpenStack API validation, scenarios, and other specific tests useful in validating an OpenStack deployment.

OPENSTACK TERMINOLOGY

OpenStack uses the following terminology:

- **Cloud controller:** The coordinating manager. All machines in the OpenStack cloud communicate with the cloud controller using the Advanced Message Queuing Protocol (AMQP).
- **Cloud types:** Cloud computing can be deployed in three forms: public, private, and hybrid clouds. These different forms, depending on the kind of data being worked with, each provide different levels of security and management.
- **Cloud models:** There are three models under which a cloud service can be categorized based on the service it delivers: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS). The OpenStack Platform is based on IaaS, which allows consumers to build their infrastructure for various deployments, whereas software like Red Hat OpenShift is PaaS, which offers a scalable platform to host applications.
- **Telemetry:** Resources can be monitored and metered by the service provider and the cloud consumer.

Traditional workloads vs. cloud workloads

Traditional workloads or datacenter virtualization has been common in the computer industry for many years. Traditional workloads use a client-server architecture; failover and scaling are built into the infrastructure. One configurable machine is built to handle the workload. When the workload increases, the machine scales up by adding more RAM, more CPUs, and more storage.

Cloud workloads require design changes to the application. The application uses a distributed architecture. Failover and scaling are built into the application. The application can scale out by adding more virtual instances to meet demand.

Benefits of OpenStack

The OpenStack Platform permits heterogeneous environments supporting cloud deployments to integrate with OpenStack technologies — which are based on open standards — all with the flexibility for future changes. As innovation is made within the OpenStack community and released by the OpenStack Foundation, The Foundation will ensure its platform maintains full interoperability for ongoing integration of all cloud infrastructure components in an environment. Perpetual interoperability coupled with the broad vendor ecosystem ensures companies that adopt the OpenStack platform can continue to purchase low-cost commodity hardware to meet demand and replace end-of-life equipment. With the open source platform, customers are not subject to cost and availability pitfalls that can happen with a proprietary approach. Some of the benefits of using OpenStack for building private clouds are:

- **Standardization in its foundation.** The promise of the cloud has greatly eliminated the days when large technology providers tried to one-up each other with giant, closed systems. But that promise is dependent on standardization. Thus, more than 800 companies supporting OpenStack are striving toward a flexible, standardized platform that works interchangeably with any infrastructure. This is extremely important, especially since many financial companies have spent years investing millions of dollars in IT.
- **Less cost and more innovation.** Most IT departments are focused on running and managing the infrastructure and not providing innovative solutions. The flexibility and low cost of OpenStack helps alleviate this by freeing up IT to focus on new applications, solutions, and service delivery rather than an inflexible underlying infrastructure. This allows for faster delivery of new features and products, such as online tools to help customers better manage their portfolios, and can help attract customers and increase retention.
- **Industry-wide support.** OpenStack receives widespread support from some of the most important players in the technology industry, all of which have come together to help companies break away from being locked in with a particular cloud vendor. While some of these players offer their own flavor of OpenStack, they still commit to the ideals of an open, standardized cloud. Therefore, unlike many technology purchases, it is not really about choosing the technology itself, but selecting a vendor with the richest ecosystem and support, knowing that support extends to virtually an entire industry.
- **Portability to other clouds.** Investments in open cloud development like OpenStack must be portable to other clouds. Portability takes a variety of forms, including programming languages and frameworks, data, and applications. If developing an application for one cloud, it should not need to be rewritten in a different language or use different APIs to move it somewhere else.
- **Deployable on the infrastructure of choice.** OpenStack provides an additional layer of abstraction above virtualization, physical servers, storage, networking, and public cloud providers. This requires that cloud management be independent of specific virtualization and other infrastructure technologies. OpenStack can work on libvirt, kvm, or qemu.

LAUNCHING AN INSTANCE WITH HORIZON WORKSHOP

In this lab, you will become familiar with the OpenStack UI project, Horizon. We will be using a Metacloud environment for the lab. Metacloud is a private cloud offering based on OpenStack that comes with a 99.99% SLA across the entire stack.

It makes use of the underlying unmodified OpenStack APIs. For the purpose of these labs, Metacloud is just being used as a robust highly available OpenStack cloud.

More information on Metacloud can be found at <http://www.cisco.com/go/metacloud>

OBJECTIVES

- Log into Horizon.
- Work with self-signed certificates.
- Describe a project and its associated users.
- Launch an instance in Horizon.
- Verify the instance launched.
- Verify the instance has connectivity.

Note: This workshop is just an introduction into OpenStack operations. Many courses exist that are a deep dive into administrating the OpenStack platform. Next steps for the novice user may be a course on how to:

- Deploy each of the OpenStack services manually.
- Manage users and projects.
- Deploy instances and use Heat to deploy and customize instances.
- Troubleshoot an operating OpenStack cloud
- Configure HA applications using hybrid and multi-cloud architectures

LAB

Logging into the Horizon Web Interface



The OpenStack dashboard is a web-based graphical user interface for managing OpenStack services.

This dashboard is also known as Horizon. It allows customizing the brand of the dashboard. Accessing the browser dashboard requires the host name and login password. It is accessible over http or https, for example: <https://23.246.115.210/>.

Work with self-signed certificates

Normally, a certificate authority (CA) would be used to generate the certificates used by OpenStack, including the web server certificate for Horizon. In the classroom, the default certificate mechanisms will be used, which will generate a self-signed certificate. Firefox/Chrome does not accept these self-signed certificates by default because of the security implications. To manually accept the certificate, follow these steps.

1. Browse to the Horizon URL: <https://23.246.115.210/>
2. Expand the I Understand the Risks menu.
3. Select the Add Exception... option.

4. Select the Confirm Security Exception option to permanently store this exception.

Secure Connection Failed

An error occurred during a connection to demo.example.com. You have received an invalid certificate.

Please contact the server administrator or email correspondent and give them the following information:

Your certificate contains the same serial number as another certificate issued by the certificate authority.

Please get a new certificate containing a unique serial number.

(Error code: sec_error_reused_issuer_and_serial)

This error states that the serial number has been reused by the certificate authority. To remove the old certificate and certificate authority in Firefox, follow these steps:

12. Open the Firefox menu by pressing the Alt key.
12. Go to Edit > Preferences.
12. Go to the Advanced icon and select the Certificates tab.
12. Select the View Certificates option.
12. In the Authorities tab, scroll down to the openstack CA (if it exists). Highlight the server name and select the Delete or Distrust... option. Select OK to confirm. Do the same to any other CAs listed under the openstack list.
12. Go to the Servers tab. In the openstack accordion, highlight the server name and select the Delete option. Press OK to confirm. Repeat for any other servers listed under openstack.
12. Close the Preferences windows.
12. Open the menu again by pressing Alt. Go to History > Clear Recent History....
12. In Time range to clear, choose Everything. Select Cache and Active Logins check boxes. Select the Clear Now option.
12. Go to the Horizon dashboard URL and accept the self-signed certificate as outlined previously.

Into the Dashboard – User In A Project

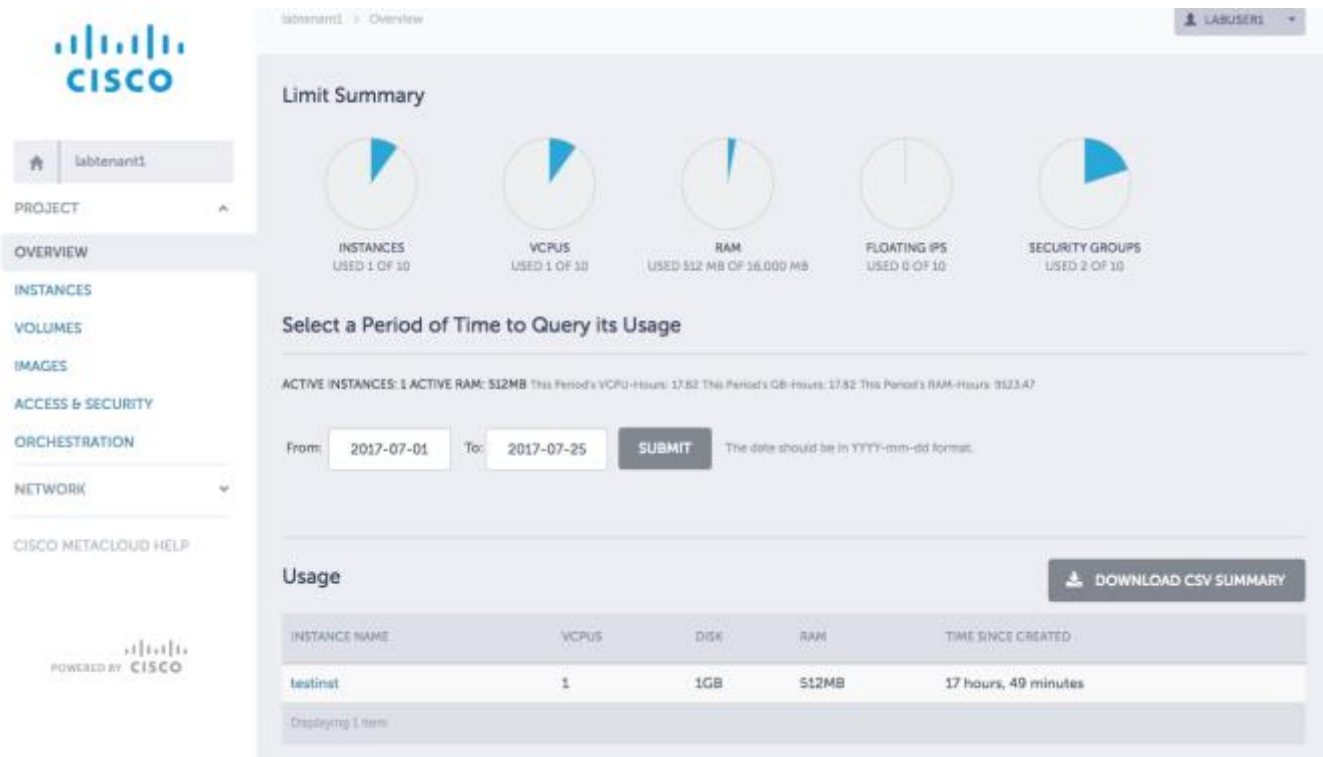
The *Project* tab provides an interface for viewing and managing projects and users.

To make an OpenStack cloud, multitenant *projects* are used to act as a container of resources owned by different users.

A set of resource quotas are preconfigured when a new project is created. The quotas include the number of instances, number of VCPUs, RAM, and floating IPs that can be assigned to instances in the project. Users can be assigned multiple projects, but one of the projects is designated the *primary project*. The *primary project* is simply the first project the user is associated with.

The Horizon dashboard or command-line tools can be used to create, modify, and delete projects, view project usage, and add or remove users as project members, modify quotas, and set an active project.

Using the *Horizon* dashboard or command line tools, users with the admin role associated in the admin project can view, create, edit, and delete users, and change user passwords. The *Users* tab displays only if logged in as a user with administrative privileges. While creating users, the Role needs to be specified apart from username, password, email address, and primary project. OpenStack comes with two predefined roles, *_member_*, and *admin*. Adding a user with *admin* role makes the user an administrative super user.



Components required to launch an instance

To launch an instance in OpenStack, the following components must be set up first:

Images. An image is a file that contains a virtual disk with a bootable operating system installed on it. Images can either be created or customized using openstack image create. Several prebaked images provided by various software vendors can also be imported. In a multitenant cloud environment, users can also share

their personal images with other projects. These images can be of various formats (RAW, QCOW2, ISO, VMDK, VHD, ARI, AKI and AMI), which can be imported into OpenStack.

Images for the current project can be found in the Horizon dashboard under Images. There is a Create Image option on the main Images page for creating new images.

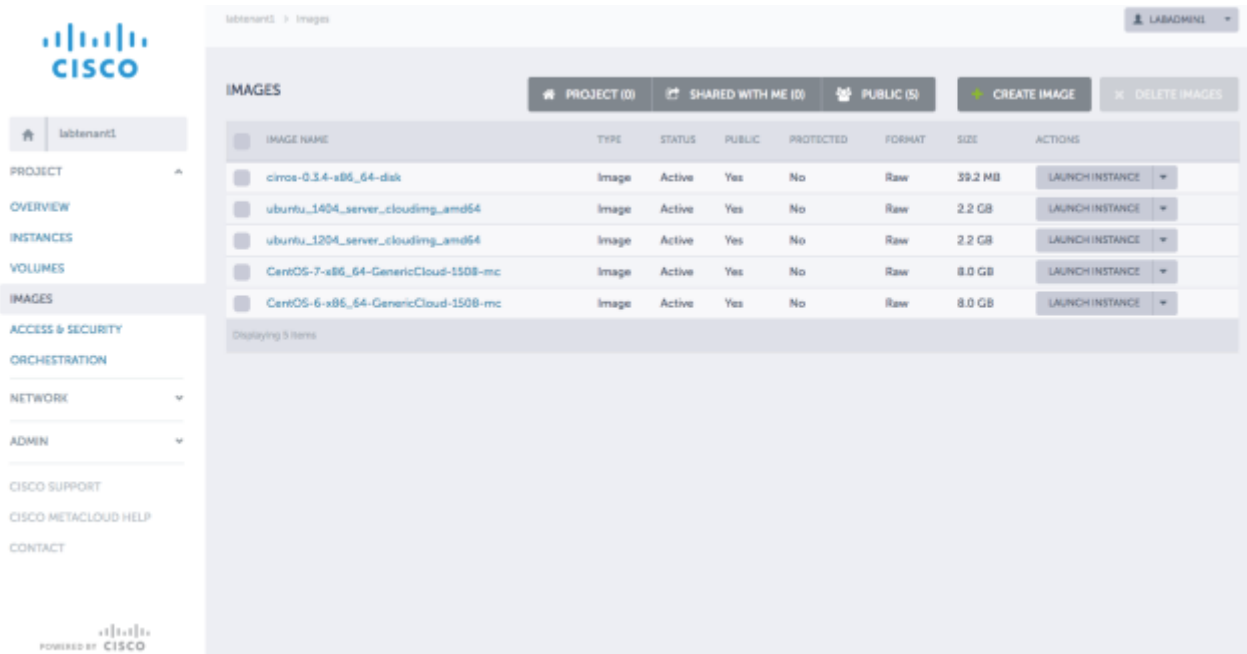


IMAGE NAME	TYPE	STATUS	PUBLIC	PROTECTED	FORMAT	SIZE	ACTIONS
cirros-0.3.4-x86_64-disk	Image	Active	Yes	No	Raw	39.2 MB	LAUNCH INSTANCE
ubuntu_1404_server_cloudimg_amd64	Image	Active	Yes	No	Raw	2.2 GB	LAUNCH INSTANCE
ubuntu_1204_server_cloudimg_amd64	Image	Active	Yes	No	Raw	2.2 GB	LAUNCH INSTANCE
CentOS-7-x86_64-GenericCloud-1508-mc	Image	Active	Yes	No	Raw	8.0 GB	LAUNCH INSTANCE
CentOS-6-x86_64-GenericCloud-1508-mc	Image	Active	Yes	No	Raw	8.0 GB	LAUNCH INSTANCE

Flavors. Virtual hardware templates are called *flavors* in OpenStack. It defines sizes for minimum RAM and disk and image format. The Horizon dashboard and command-line tools provide the ability to modify and delete an existing flavor and create a new one.

Flavors for the current project can be found in the Horizon dashboard while launching an instance by a minimal user, but only administrators have permission to edit or create new flavors. (The rights may also be delegated to other users.) An admin user can use the Admin tab, which provides a Flavors item.

FLAVORS

Filter

FLAVOR NAME	VCPUS	RAM	ROOT DISK	EPHEMERAL DISK	SWAP DISK	ID	PUBLIC	METADATA	INSTANCES	ACTIONS
<input type="checkbox"/> m1.tiny	1	512MB	1GB	0GB	0MB	1	Yes	No	2	<input type="button" value="EDIT FLAVOR"/>
<input type="checkbox"/> VCO_cisco_metapod_validation_flavor	1	2GB	10GB	0GB	0MB	0cad4244-2856-4087-b52a-7af5f8c11ad3	Yes	No	0	<input type="button" value="EDIT FLAVOR"/>
<input type="checkbox"/> m1.small	1	2GB	20GB	0GB	0MB	2	Yes	No	0	<input type="button" value="EDIT FLAVOR"/>
<input type="checkbox"/> m1.medium	2	4GB	40GB	0GB	0MB	3	Yes	No	0	<input type="button" value="EDIT FLAVOR"/>
<input type="checkbox"/> m1.large	4	8GB	80GB	0GB	0MB	4	Yes	No	0	<input type="button" value="EDIT FLAVOR"/>
<input type="checkbox"/> m1.xlarge	8	16GB	160GB	0GB	0MB	5	Yes	No	0	<input type="button" value="EDIT FLAVOR"/>

Displaying 6 items

Security groups. Security groups are sets of IP filter rules that are applied to an instance's network-ing. They are project-specific, and project members can edit the default rules and add new rule sets. All projects have the default security group pre-created, which is applied to instances that have no other security group defined. Unless edited, the default security group denies all incoming traffic.

Security groups for the current project can be found in the Horizon dashboard under Access & Security. There is a Create Security Group option on the main Access & Security page for creating new security

Key pair. To improve the security posture in the cloud, logging into the cloud instances with secure shell (SSH) is done by use of a public and private key pair instead of using a username and password.

These key pairs can be created or imported via the Horizon dashboard under Access & Security. There is a Create Security Group option on the main Access & Security page for creating or import-ing key pairs.

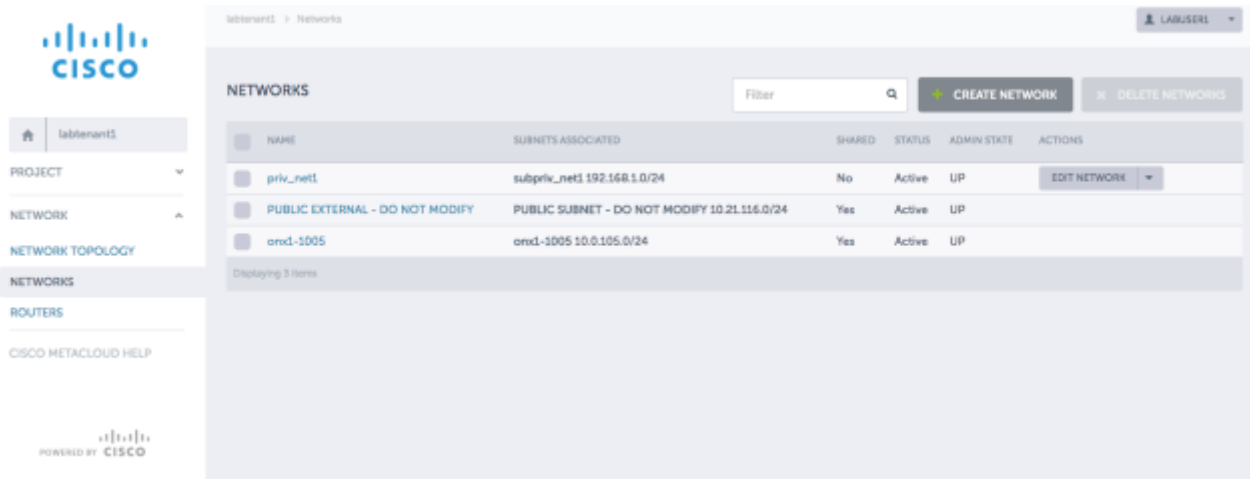
Key Pairs

Filter

KEY PAIR NAME	FINGERPRINT	ACTIONS
<input type="checkbox"/> labpair	97:58:b4:79:32:b5:42:ab:80:2d:1d:10:87:ee:52:2e	<input type="button" value="DELETE KEY PAIR"/>

Displaying 1 item

Networks. Networks define the network routes to be added to an instance. Based on the network the instance belongs to and the route table for the network, the instance can be public-facing or private-facing.



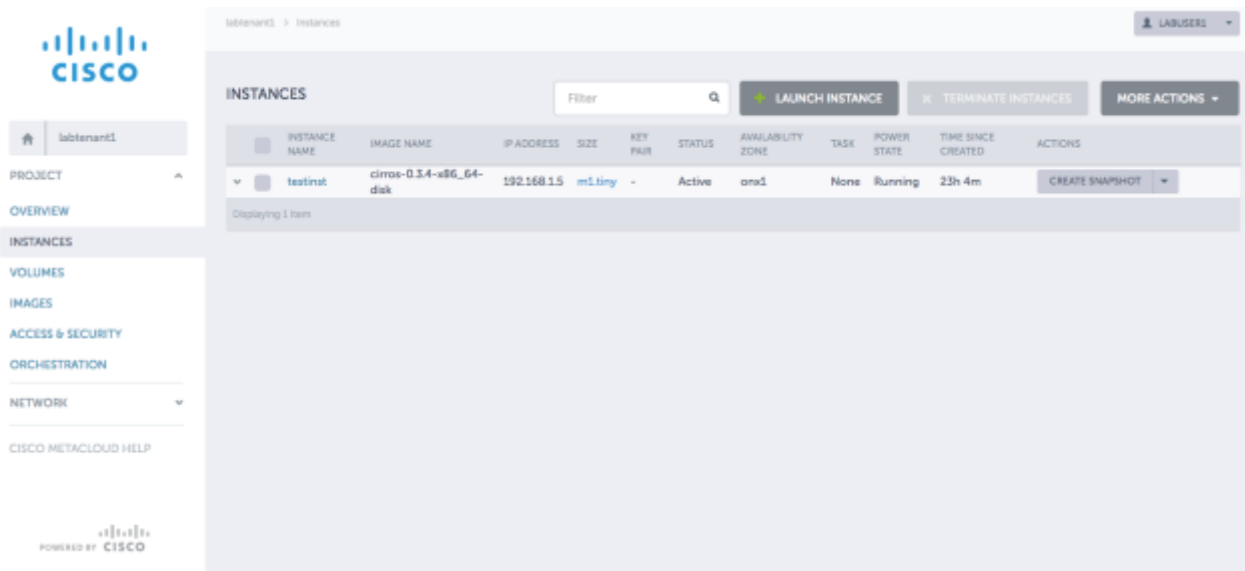
NAME	SUBNETS ASSOCIATED	SHARED	STATUS	ADMIN STATE	ACTIONS
priv_net1	subpriv_net1 192.168.1.0/24	No	Active	UP	EDIT NETWORK
PUBLIC EXTERNAL - DO NOT MODIFY	PUBLIC SUBNET - DO NOT MODIFY 10.21.116.0/24	Yes	Active	UP	
ond-1005	ond1-1005 10.0.105.0/24	Yes	Active	UP	

A *floating IP* address can be associated with the private-facing interface to make it public-facing and potentially reachable outside the OpenStack network. Administrators can configure rich network topologies by creating and configuring networks and subnets, and then instructing other OpenStack services to attach virtual devices to ports on these networks. In this lab we will not be assigning floating IPs and the network setup has already been completed to route traffic out to the external network. Feel free to delete the network setup and attempt to rebuild it, as a project member, you have that ability. You can also create a new private network with a new subnet range and place an instance on it.

OpenStack networking supports multiple private networks for each project and allows projects to choose their own IP addressing scheme, even if IP addresses overlap with those used by other projects.

The network topology can be viewed via the Horizon dashboard under Project by selecting the **Network Topology** menu item under the Network menu.

Launch an instance in Horizon. Horizon can be used to launch and manage instances after they have spawned. Before launching an instance, it will be necessary to create and upload an image (which constitutes the operating system and software for the instance), configure a security group (to open certain ports to the instance through the firewall), create an SSH key pair (to be able to connect to the instance), and associate a floating IP address (to be able to access the instance from outside). The following demonstration will walk through the steps of launching the first instance.



Lab: Launching an instance in Horizon

Launching an instance in Horizon

- On workstation, open Firefox and browse to <https://23.246.115.210/metacloud/>
- Log into the Horizon dashboard using labuserX as the username and the password as p455word
- The domain is: default
- When you log in, make sure your project IE: labtenant1 is listed in the upper left hand corner as the active project.

Launch an instance named lab_instance using m1.small flavor, the cirros image, leave key pair blank, the labsecgroup security group (uncheck default), and the priv_netX network.

1. Go to the Instances subtab under the Project tab.
2. Select the Launch Instance option.
3. In the Details tab, leave the availability zone as onx1 and enter lab_instance as the Instance Name.
4. In the Details tab, choose m1.small as the Flavor.
5. In the Details tab, choose Boot from image in the Instance Boot Source dropdown menu.
6. In the Details tab, select cirros-0.3.4-x86_64-disk as the Image Name.
7. In the Access & Security tab, no key pair is selected. Select the labsecgroupX security group.

8. In the Networking tab, ensure that priv_netX network is under selected networks. If it's not selected, click the + button next to the priv_netX network to select it. Select the Launch option to launch the demo_instance.
9. If you are not in the Instance subtab under Project, click on Project and go to instance subtab.
10. Inside the Instance subtab in lab_instance row, open the dropdown menu under Actions.
11. Select the menu item View Log. Examine the log for any errors. If you see lab_instance login: as the last line, then the server is up and ready for you to console in.

Note:

Due to the way the lab is connected for this session, the console may not load. To get the console to load, do the following:

Mac:

1. Open a terminal window on your Mac and elevate your privileges to root while opening the hosts file:

```
# sudo "/Applications/TextEdit.app/Contents/MacOS/TextEdit" /etc/hosts
```

2. Now edit your hosts file by placing the line below under "::1 localhost":

```
23.246.115.210 dashboard-onx1.client.metacloud.net
```

3. Save your file in TextEdit and Quit TextEdit

4. Now flush your DNS cache:

```
# sudo killall -HUP mDNSResponder
```

5. Close the terminal window and try the console again.

Windows 8 and 10:

1. Press the Windows key.
 2. Type Notepad in the search field.
 3. In the search results, right-click Notepad and select Run as administrator.
 4. From Notepad, open the following file: c:\Windows\System32\Drivers\etc\hosts.
Add 23.246.115.210 dashboard-onx1.client.metacloud.net
 5. Click File > Save to save your changes.
12. Login to the console (you may have to click on the black area and hit enter several times). If you don't get a prompt, right click on the link named "Click here to show only console" and open the console in a new tab. The login is "cirros" and the password is cubswin:)
 13. To see if you have a working instance, try running "curl <http://www.example.com>" and if you get output back, your instance is connected to the Internet.

Level Up (other things to try)*:

* Don't worry, you can't break anything other than your own project 😊

- Delete your subnet, router default gateway, router, subnet and finally network. Then put it all back. Alternatively you can simply create a second private network, subnet, and router. Call it, "private_[yourfirstname]."
- Create another instance from another image, see if you can ping it from the first instance.
- Create a ssh key, use it on a new instance and ssh to it from the console of another instance.
- Delete all of your security groups and create a new one. Leave out ssh port, see what happens when you try and ssh out or in.
- Create a volume, attach it to an instance. Log into that instance and see if it is available under the OS.
 - Hint: `# ls /dev/disk/virtio*` in your VM
 - Mkfs it and attach it like any other hardware based disk