Table of Contents

What is OpenStack?

*"Founded by **Rackspace** Hosting and **NASA** in **2010**, OpenStack has grown to be a global software community of developers collaborating on a standard and massively scalable open source cloud operating system*."

OpenStack is a set of software tools for building and managing cloud computing platforms for public and private clouds. Backed by some of the biggest companies in software development and hosting, as well as thousands of individual community members, many think that OpenStack is the future of cloud computing. OpenStack is managed by the OpenStack Foundation, a non-profit that oversees both development and community-building around the project

Why we need Open stack?

Simplicity: - IT administrators should be able to create their cloud on commodity hardware from a single physical server to hundreds of thousands of servers, without special skills.

Robust: - The software must be stable for long-term operation while allowing developers to add new useful functions.

Flexibility: - Besides the well-known Amazon EC2 model, the software should be able to implement new models that fit traditional enterprise needs. There are many organizations that don't want to lock themselves into a single vendor, regardless of how good or bad this vendor is.

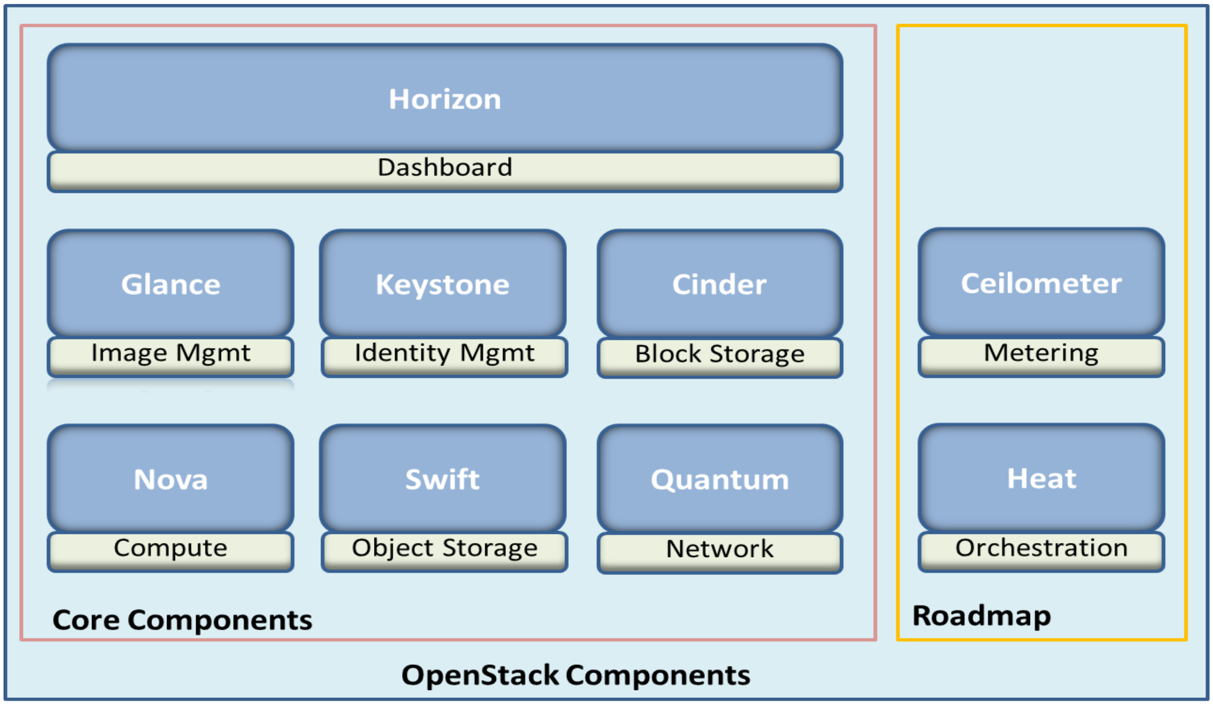
Control: - There are still many cases in which controlling the infrastructure will give you a business advantage in controlling your product margins. That is, using specialized infrastructure that is more optimized for the kind of workload and customers that you are serving, than just general purpose infrastructure.

Private cloud: - You can have your own private cloud in your premises with more security.

Having said that, I think that the most important point is that **OpenStack is not a product** and shouldn't be measured as such**. It's an ecosystem with strong foundations behind it**.

What are the components of Open Stack?

Open Stack is made up of many different moving parts. Because of its open nature, anyone can add additional components to OpenStack to help it to meet their needs. But the OpenStack community has collaboratively identified nine key components that are a part of the "core" of OpenStack, which are distributed as a part of any OpenStack system and officially maintained by the Open Stack community.



Nova: - is the primary computing engine behind OpenStack. It is used for deploying and managing large numbers of virtual machines and other instances to handle computing tasks.

Swift: - is a storage system for objects and files. Rather than the traditional idea of a referring to files by their location on a disk drive, developers can instead refer to a unique identifier referring to the file or piece of information and let OpenStack decide where to store this information. This makes scaling easy, as developers don’t have the worry about the capacity on a single system behind the software. It also allows the system, rather than the developer, to worry about how best to make sure that data is backed up in case of the failure of a machine or network connection.

Cinder: - is a block storage component, which is more analogous to the traditional notion of a computer being able to access specific locations on a disk drive. This more traditional way of accessing files might be important in scenarios in which data access speed is the most important consideration.

Neutron: - provides the networking capability for OpenStack. It helps to ensure that each of the components of an OpenStack deployment can communicate with one another quickly and efficiently.

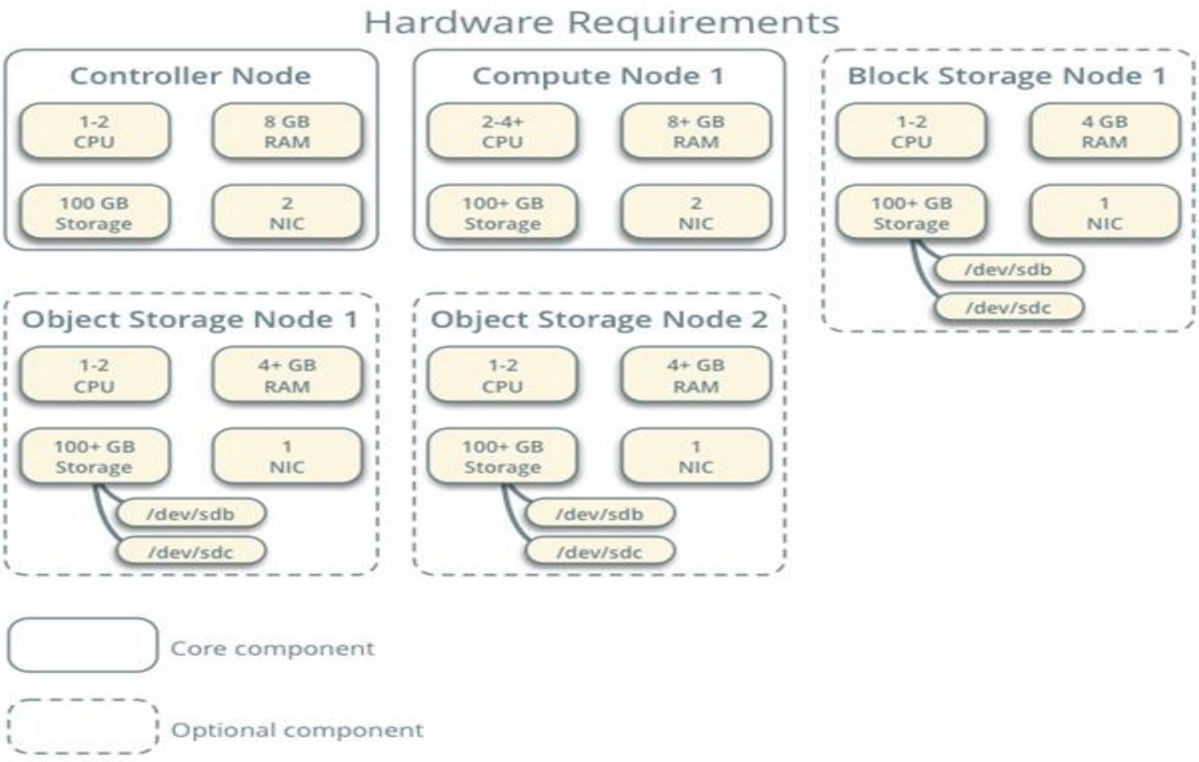
Horizon: - is the dashboard behind OpenStack. It is the only graphical interface to Open Stack, so for users wanting to give OpenStack a try, this may be the first component they actually “see.” Developers can access all of the components of OpenStack individually through an application programming interface (API), but the dashboard provides system administrators a look at what is going on in the cloud, and to manage it as needed.

Keystone: - provides identity services for OpenStack. It is essentially a central list of all of the users of the OpenStack cloud, mapped against all of the services provided by the cloud, which they have permission to use. It provides multiple means of access, meaning developers can easily map their existing user access methods against Keystone.

Glance: - provides image services to OpenStack. In this case, "images" refers to images (or virtual copies) of hard disks. Glance allows these images to be used as templates when deploying new virtual machine instances.

Ceilometer: - provides telemetry services, which allow the cloud to provide billing services to individual users of the cloud. It also keeps a verifiable count of each user’s system usage of each of the various components of an OpenStack cloud. Think metering and usage reporting.

Hardware requirement for Open Stack?



How to Install OpenStack?

There are different ways to install Open stack some are listed below.

DevStack (Specially for Ubuntu)

A series of extensible scripts used to quickly bring up a complete OpenStack environment based on the latest versions of everything from git master. It is used interactively as a development environment and as the basis for much of the OpenStack project's functional testing.

The source is available at https://git.openstack.org/cgit/openstack-dev/devstack.

For installation refer to https://docs.openstack.org/developer/devstack/

If you do not have a preference, **Ubuntu 16.04** is the most tested, and will probably go the smoothest.

Mirantis

To gain experience with the installation and deployment process, most users start by installing OpenStack on their laptop or desktop computer. Mirantis recommends automated installation using OpenStack Fuel and Virtual Box.

Prerequisites A 64-bit host OS with at least 8 GB RAM and 300 GB of free space. Virtualization must be enabled in the BIOS. If you use Microsoft Windows, use Cygwin.

For installation refer to https://www.mirantis.com/how-to-install-openstack/

PackStack

A utility that uses Puppet modules to deploy various parts of OpenStack on multiple pre-installed servers over SSH automatically. Currently only CentOS, Red Hat Enterprise Linux (RHEL) and compatible derivatives of both are supported.

Prerequisites Machine with at least 4GB RAM, preferably 6GB RAM, processors with hardware virtualization extensions, and at least one network adapter.

Note - Here we will mainly focus on how to install OpenStack via PackStack.

Steps to Install PackStack: -

✓ Install Centos 7 with Virtual Box.

✓ Give it 6-8GB memory and 4-6 CPUs.

✓ 1 NIC .

✓ 20 GB disk space at least.

✓ If you plan on having external network access to the server and instances, this is a good moment to properly configure your network settings. A static IP address to your network card, and disabling Network Manager are good ideas.

```
$ sudo systemctl disable firewalld
$ sudo systemctl stop firewalld
$ sudo systemctl disable NetworkManager
$ sudo systemctl stop NetworkManager
$ sudo systemctl enable network
$ sudo systemctl start network
```

✓ If you are using non-English locale make sure your /etc/environment is populated:

```
LANG=en_US.utf-8
LC_ALL=en_US.utf-8
```

✓ Install the latest version of Opnstack release.

```
$ sudo yum install -y centos-release-openstack-ocata
$ sudo yum update -y
$ sudo yum install -y openstack-packstack
```

✓ Once all these packages installed you can go for all-in-one approach or you can install specific components for your environment. Below is all-in-one approach.

```
$ sudo packstack –allinone

   OR
$sudo packstat –gen-answer-file=answer.txt
```

This will create “answer.txt” file in your current directory.

```
$vim answer.txt
```

Here in “answer.txt” file you can choose which component you want you can add multiple compute nodes, network nodes, and storage nodes as per your requirement.
Once you are ready run the below command.

```
$sudo packstack –answer-file=answer.txt
```

Make sure you give 10GB to cinder volume you can give more default is 20GB.

Note- It will take 30-40 minutes to install. For more info on installation refer to https://www.rdoproject.org/install/quickstart/

✓ After installation finish you can see file “keystone_admin” in your current working directory. This file will have dashboard url and admin user password.

```
$sudo source keyston_admin
$sudo nova list
```

✓ Make changes in the **/etc/libvirt/qemu.conf** file set these parameters as mentioned below. (Find and change as below).

```
user=root
group=root
clear_emulator_capabilities=0
```

✓ Run the below commands:

```
#systemctl restart libvirtd
#chown –R nova:nova /var/lib/nova/instances
#mount –t devpts devpts /dev/pts
```

And vim **/etc/fstab** and append below line.

```
devpts /dev/pts devpts gid=5,mode=620 0 0
```

Configure Networking on local host machine: -

- On your base machine simply copy your ifcfg-eth0 to ifcfg-br-ex and make /etc/sysconfig/network-scripts/ifcfg-br-ex resemble:

```
DEVICE=br-ex
DEVICETYPE=ovs
TYPE=OVSBridge
BOOTPROTO=static
IPADDR=192.168.122.212
NETMASK=255.255.255.0  # your netmask
GATEWAY=192.168.122.1  # your gateway
DNS1=192.168.122.1     # your nameserver
ONBOOT=yes
```

- Make /etc/sysconfig/network-scripts/ifcfg-eth0 resemble (no BOOTPROTO!):

  Note: if on Centos7, the file could be /etc/sysconfig/network-scripts/ifcfg-enp2s0 and DEVICE should be enp2s0

```
DEVICE=eth0
TYPE=OVSPort
DEVICETYPE=ovs
OVS_BRIDGE=br-ex
ONBOOT=yes
```

Note- It is also possible to use a bond. In that case /etc/sysconfig/network-scripts/ifcfg-bond0 may look like this:

```
DEVICE=bond0
DEVICETYPE=ovs
TYPE=OVSPort
OVS_BRIDGE=br-ex
ONBOOT=yes
BONDING_MASTER=yes
BONDING_OPTS="mode=802.3ad"
```

- This means, we will bring up the interface and plug it into br-ex OVS bridge as a port, providing the uplink connectivity.

  Restart the network service

```
# reboot
```

or, alternatively:

```
# service network restart
```

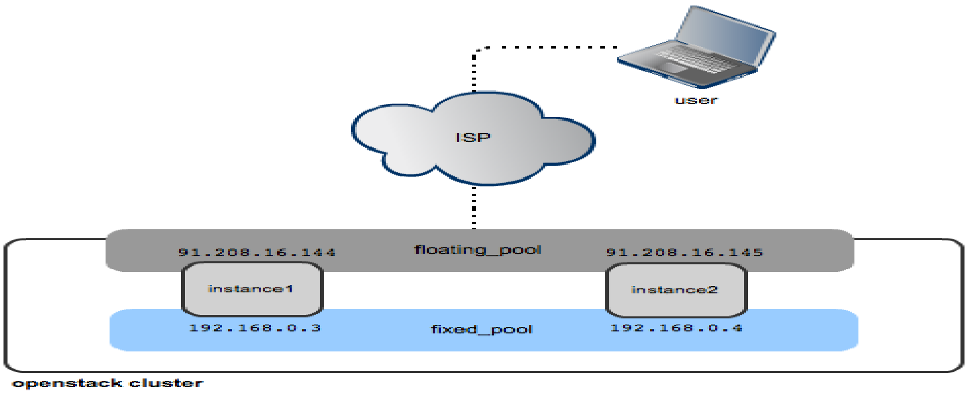
Networking for open stack instance: -

We need two network for the instances and a router to join them.

1. Private network to communicate internally.

2. External network to access the world.

3. Router to join these two networks.



Create a public network : -

1. Log in to the dashboard.
2. Select the appropriate project from the drop down menu at the top left.
3. On the Project tab, open the Admin tab and click Networks category.
4. Click Create Network.

5. In the Create Network dialog box, specify the following values.
   a. Network tab
   b. Network Name: Public.
   c. Shared: Share the network with other projects. Non admin users are not allowed to set shared option.

6. Create Subnet: Select this check box to create a subnet
   a. Subnet tab
   b. Subnet Name: public_subnet
   c. Network Address: 192.168.114/0/24.
   d. IP Version: Select IPv4
   e. Gateway IP: 192.168.114.1. This parameter is optional.
   f. Disable Gateway: Do not select this check box to disable a gateway IP address.

7. Subnet Details tab
   a. Enable DHCP: Select this check box to enable DHCP.
   b. Allocation Pools: Specify IP address pools. This is optional.
   c. DNS Name Servers: Specify a name for the DNS server. This is optional.
   d. Host Routes: Specify the IP address of host routes. This is optional.
8. Click Create.

   The dashboard shows the network on the Networks tab.

Create a private network: -

1. Log in to the dashboard.
2. Select the appropriate project from the drop down menu at the top left.
3. On the Project tab, open the Network tab and click Networks category.
4. Click Create Network.
5. In the Create Network dialog box, specify the following values.
   a. Network tab
   b. Network Name: Private.
   c. Shared: Share the network with other projects. Non admin users are not allowed to set shared option.

6. Create Subnet: Select this check box to create a subnet
   a. Subnet tab
   b. Subnet Name: private_subnet
   c. Network Address: 10.10.1.0/24.
   d. IP Version: Select IPv4
   e. Gateway IP: 10.10.1.1. This parameter is optional.
   f. Disable Gateway: Do not select this check box to disable a gateway IP address.

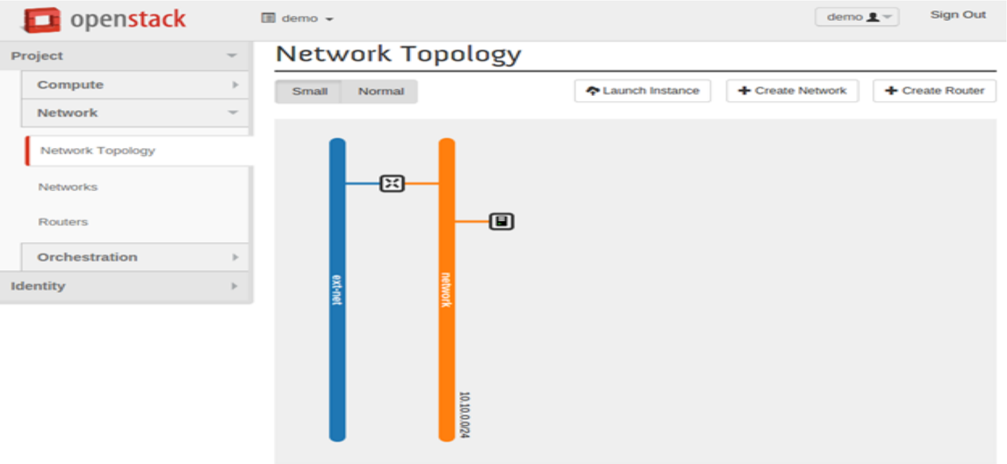
7. Subnet Details tab
   a. Enable DHCP: Select this check box to enable DHCP.
   b. Allocation Pools: Specify IP address pools. This is optional.
   c. DNS Name Servers: Specify a name for the DNS server. This is optional.
   d. Host Routes: Specify the IP address of host routes. This is optional.
8. Click Create.
   The dashboard shows the network on the Networks tab.

Create router : -

1. Log in to the dashboard.
2. Select the appropriate project from the drop down menu at the top left.
3. On the Project tab, open the Network tab and click Routers category.
4. Click Create Router.
5. In the Create Router dialog box, specify a name (Give any logical name) for the router and External Network, and click Create Router.
6. The new router is now displayed in the Routers tab.
7. To connect a private network to the newly created router, perform the following steps:
   a. On the Routers tab, click the name of the router.
   b. On the Router Details page, click the Interfaces tab, then click Add Interface.
   c. In the Add Interface dialog box, select a Subnet.
   d. Add both public and private network.
   e. If you choose not to set the IP Address value, then by default OpenStack Networking uses the first host IP address in the subnet.

8. The Router Name and Router ID fields are automatically updated.
9. Click Add Interface.

10. You have successfully created the router. You can view the new topology from the Network Topology tab.

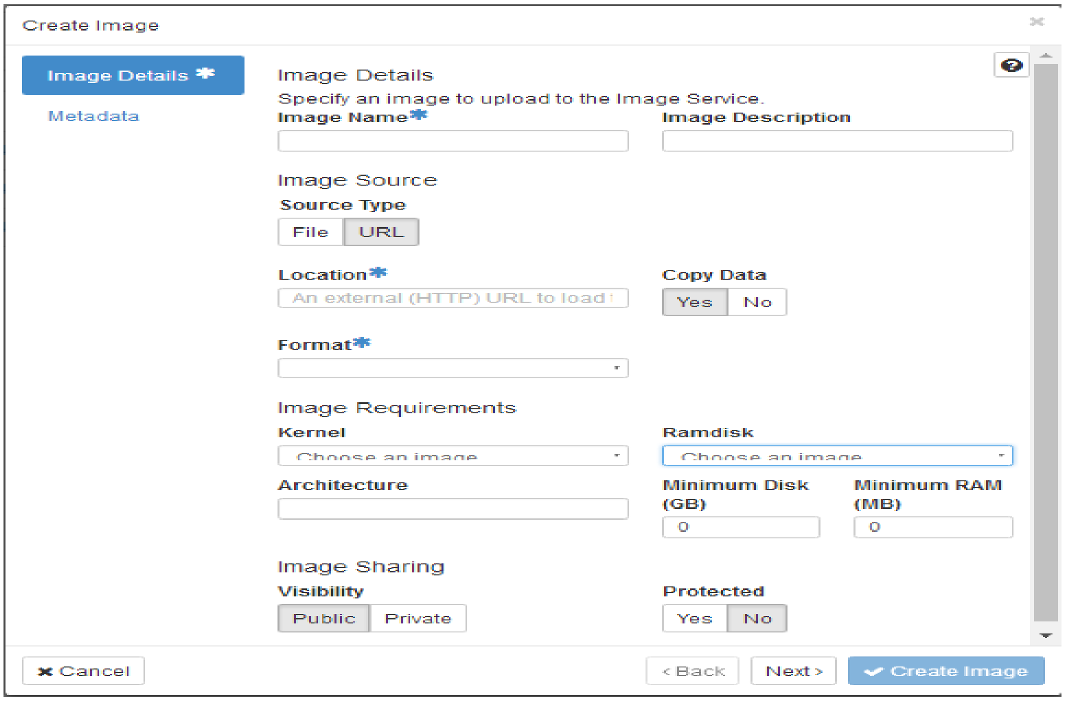


Download/Add image to the environment: -

Now to spin the instances you need to have a image/template to deploy as VM/instance. In Open stack this is taken care by **Glance.**

Follow this procedure to upload an image to a project:

1. Log in to the dashboard.
2. Select the appropriate project from the drop down menu at the top left.
3. On the Project tab, open the Compute tab and click Images category.
4. Click Create Image.
5. The Create an Image dialog box appears.



Create Image

Image Details

Specify an image to upload to the Image Service.

Image Name*

Image Description

Image Source

Source Type: File | URL

Location*: An external (HTTP) URL to load t

Copy Data: Yes | No

Format*

Image Requirements

Kernel: Choose an image

Ramdisk: Choose an image

Architecture

Minimum Disk (GB): 0

Minimum RAM (MB): 0

Image Sharing

Visibility: Public | Private

Protected: Yes | No

6. You can locally download the image and upload OR you can give URL and it will be downloaded automatically.

Manage flavor: -

In OpenStack, flavors define the compute, memory, and storage capacity of nova computing instances. To put it simply, a flavor is an available hardware configuration for a server. It defines the size of a virtual server that can be launched. By default there will be some flavor created in advance, If you want you can create a new one as per your requirement.

From command line you can check flavor by the below command.

```
#openstack flavor list
```

OR if you want to create by command line then:-

```
#openstack flavor create FLAVOR_NAME --id FLAVOR_ID --ram RAM_IN_MB --disk
ROOT_DISK_IN_GB --vcpus NUMBER_OF_VCPUS
```

To delete a flavor:-

```
#openstack flavor delete FLAVOR_ID
```

Create SSH Key-pair: -

1. You can generate a key pair or upload an existing public key.

To generate a key pair, run the following command.

```
# openstack keypair create KEY_NAME > MY_KEY.pem
```

Note- This command generates a key pair with the name that you specify for KEY_NAME, writes the private key to the .pem file that you specify, and registers the public key to the Nova database.

To set the permissions of the .pem file so that only you can read and write to it, run the following command.

```
# chmod 600 MY_KEY.pem
```

2. If you have already generated a key pair and the public key is located at ~/.ssh/id_rsa.pub, run the following command to upload the public key.

```
# openstack keypair create --public-key ~/.ssh/id_rsa.pub KEY_NAME
```

Note- This command registers the public key at the Nova database and names the key pair the name that you specify for KEY_NAME.

To ensure that the key pair has been successfully imported, list key pairs as follows:

```
$ openstack keypair list
```

Security Group management: -

Security group are like IPTABLES rules in Linux, We can modify them by CMD and GUI as per you feel good.

Go to **Project>Network>Security Access** And you can manage them.

Create/Launch an Instance: -

1. Log in to the dashboard.
2. Select the appropriate project from the drop down menu at the top left.
3. On the Project tab, open the Compute tab and click Instances category.
4. Click Launch Instance.
5. In the Launch Instance dialog box, specify the following values:
   a) Instance Name
   b) Availability Zone

By default, this value is set to the availability zone given by the cloud provider (for example, us-west or apac-south). For some cases, it could be nova.

Note: The name you assign here becomes the initial host name of the server. If the name is longer than 63 characters, the Compute service truncates it automatically to ensure dnsmasq works correctly.

c) **Count**: To launch multiple instances, enter a value greater than 1. The default is 1.
d) Source tab
e) Instance Boot Source

Your options are:

i. **Boot from image**: If you choose this option, a new field for Image Name displays. You can select the image from the list.
ii. **Boot from snapshot**: If you choose this option, a new field for Instance Snapshot displays. You can select the snapshot from the list.
iii. **Boot from volume:** If you choose this option, a new field for Volume displays. You can select the volume from the list.
iv. **Boot from image** (creates a new volume): With this option, you can boot from an image and create a volume by entering the Device Size and Device Name for your volume. Click the Delete Volume on Instance Delete option to delete the volume on deleting the instance.
v. **Boot from volume snapshot** (creates a new volume): Using this option, you can boot from a volume snapshot and create a new volume by choosing Volume Snapshot from a list and adding a Device Name for your volume. Click the Delete Volume on Instance Delete option to delete the volume on deleting the instance.

6. **Image Name:** This field changes based on your previous selection. If you have chosen to launch an instance using an image, the Image Name field displays. Select the image name from the dropdown list.
7. **Volume:** This field changes based on your previous selection. If you have chosen to launch an instance using a volume, the Volume field displays. Select the volume name from the dropdown list. If you want to delete the volume on instance delete, check the Delete Volume on Instance Delete option.
8. **Flavor tab**
   a) **Flavor**: Specify the size of the instance to launch.

Note- The flavor is selected based on the size of the image selected for launching an instance. For example, while creating an image, if you have entered the value in the Minimum RAM (MB) field as 2048, then on selecting the image, the default flavor is m1.small.

9. **Networks tab**

   Selected Networks

   To add a network to the instance, click the + in the Available field.

10. **Network Ports tab**
    a) Ports: Activate the ports that you want to assign to the instance.
11. **Security Groups tab**
    a) Security Groups:Activate the security groups that you want to assign to the instance.

Note- Security groups are a kind of cloud firewall that define which incoming network traffic is forwarded to instances. If you have not created any security groups, you can assign only the default security group to the instance.

12. **Key Pair tab**
    a) Specify a key pair.

Note- If the image uses a static root password or a static key set (neither is recommended), you do not need to provide a key pair to launch the instance.

13. **Configuration tab**
    a) Specify a customization script that runs after your instance launches.
14. **Metadata tab**
    a) Add Metadata items to your instance.
15. **Click Launch Instance.**

The instance starts on a compute node in the cloud.

Note- If you did not provide a key pair, security groups, or rules, users can access the instance only from inside the cloud through VNC. Even pinging the instance is not possible without an ICMP rule configured. You can also launch an instance from the Images or Volumes category when you launch an instance from an image or a volume respectively. When you launch an instance from an image, OpenStack creates a local copy of the image on the compute node where the instance starts.