

LAB – OpenStack basic operations

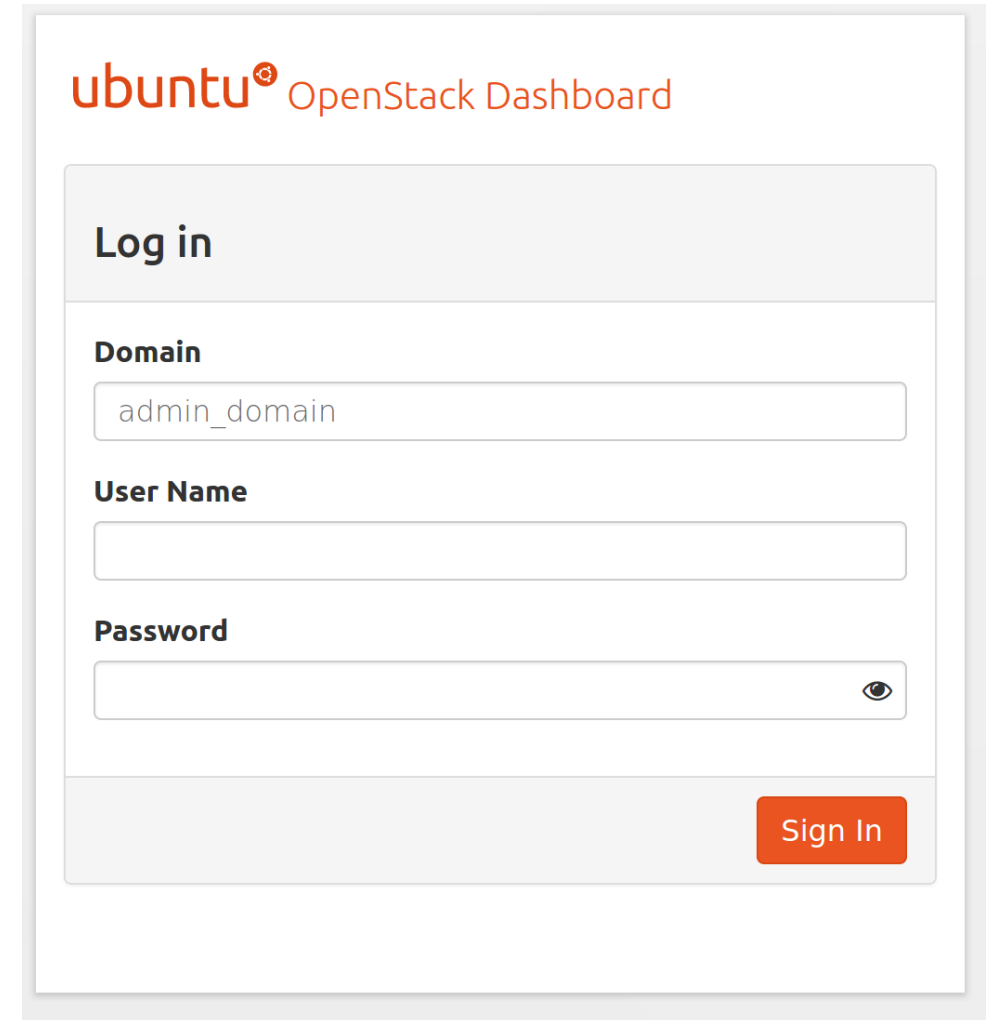
Hands on experience with OpenStack basic operations

References:

- OpenStack documentation

Connect to the dashboard

- Open the browser to the URL:
 - http://CONTROLLER_IP/horizon/auth/login/
- Domain: admin_domain
- User Name: admin
- Password: openstack



The screenshot shows the Ubuntu OpenStack Dashboard login interface. At the top, the logo "ubuntu" is in orange, followed by "OpenStack Dashboard" in grey. Below this is a "Log in" section with a light grey header. The form contains three fields: "Domain" with the value "admin_domain", "User Name" which is empty, and "Password" which is empty and has a toggle icon on the right. At the bottom right of the form is an orange "Sign In" button.

ubuntu[®] OpenStack Dashboard

Log in

Domain

User Name

Password

Sign In

Finalize the configuration

- A set of configurations are still missing for the platform to be functional
 - Network configuration
 - Flavors configuration
 - Image configuration

Network configuration

- A final configuration of the network is needed for the installation to be operational
- Specifically at least a couple of virtual networks has to be created to connect VMs among themselves and to connect the VMs to an external network (and eventually the Internet)
- The networks we want to create are:
 - An **external** network that is linked with a real external network to connect VMs with the outside world
 - An **internal** network (virtual) to link the VMs
- In general VMs will have a virtual NIC connected to the internal network, a router will be deployed to link the VMs with the external networks (through NAT as we will see)

Create the external network

- Admin -> Network -> Networks -> Create Network
- Create a new network with the following params:

This subnet is the external network, it will be bridged with physnet1 linked to vlan2 (as configured during installation)

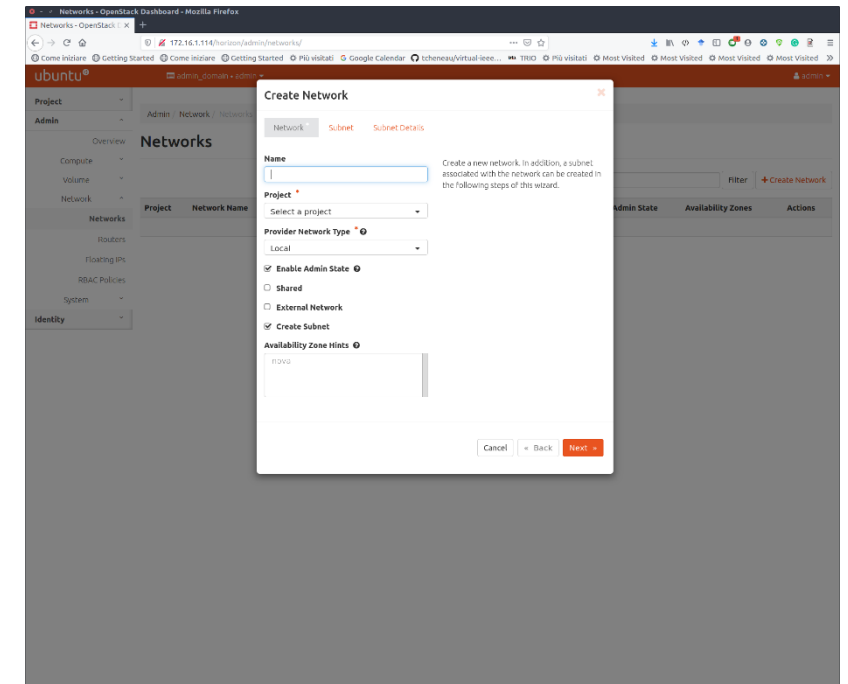
The type of network is flat, as it will be linked to a physical network

The network will be shared among all the users of the platform

The screenshot shows the 'Create Network' wizard in the OpenStack dashboard. The 'Network' tab is selected. The form contains the following fields and options:

- Name:** external
- Project:** admin
- Provider Network Type:** Flat
- Physical Network:** physnet1
- Enable Admin State:** ☒
- Shared:** ☒
- External Network:** ☒
- Create Subnet:** ☒
- Availability Zone Hints:** nova

At the bottom, there are buttons for 'Cancel', '< Back', and 'Next >'. A blue arrow points from the text 'This subnet is the external network...' to the 'Create Subnet' checkbox.



Create the subnet for the external network

This is the address of the network

This pool of IPs will be available to VMs that require external connection

This is the IP of the gateway. It is supposed to be a real gateway on the network (we will make one)

Create Network

Network * Subnet Subnet Details

Subnet Name

external

Network Address ?

192.168.1.0/24

IP Version

IPv4

Gateway IP ?

192.168.1.1

☐ Disable Gateway

Creates a subnet associated with the network. You need to enter a valid "Network Address" and "Gateway IP". If you did not enter the "Gateway IP", the first value of a network will be assigned by default. If you do not want gateway please check the "Disable Gateway" checkbox. Advanced configuration is available by clicking on the "Subnet Details" tab.

Cancel

<< Back

Next >>

Create Network

Network * Subnet Subnet Details

☒ Enable DHCP

Specify additional attributes for the subnet.

Allocation Pools ?

192.168.1.2,192.168.1.100

DNS Name Servers ?

8.8.8.8

Host Routes ?

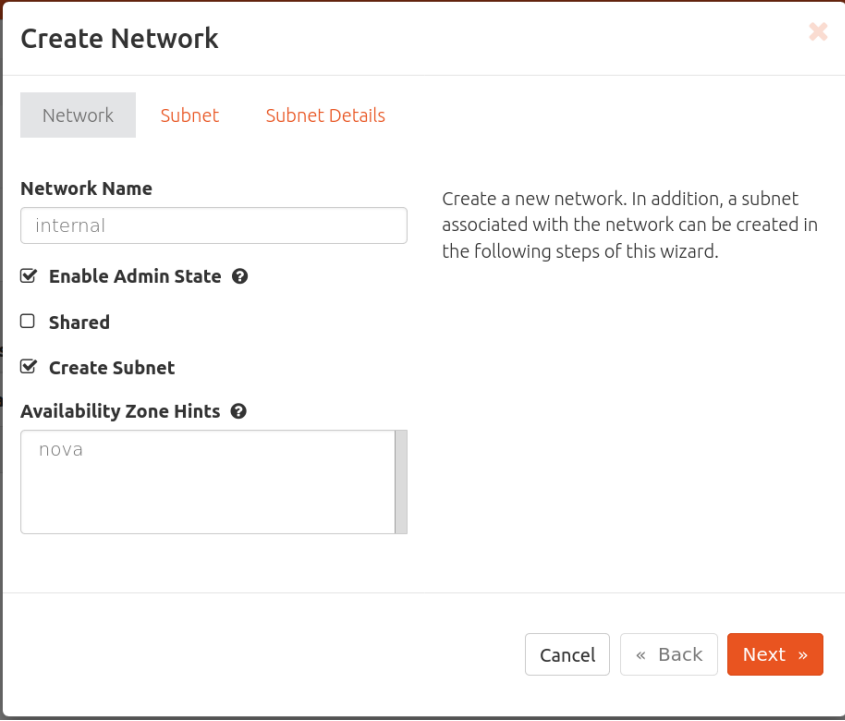
Cancel

<< Back

Create

Create an internal network

- Create an internal network for the VMs created by the admin user (each user of the platform can have its own virtual network)
- Project -> Network -> Networks -> Create Network



The screenshot shows a 'Create Network' wizard window. It has three tabs: 'Network' (selected), 'Subnet', and 'Subnet Details'. The 'Network' tab contains the following fields and options:

- Network Name:** A text input field containing the word 'internal'.
- Enable Admin State:** A checked checkbox with a help icon.
- Shared:** An unchecked checkbox.
- Create Subnet:** A checked checkbox.
- Availability Zone Hints:** A text input field containing the word 'nova'.

To the right of the 'Network Name' field, there is a descriptive text: 'Create a new network. In addition, a subnet associated with the network can be created in the following steps of this wizard.'

At the bottom of the window, there are three buttons: 'Cancel', '< Back', and 'Next >' (highlighted in orange).

Create an internal network

Create Network

Network Subnet Subnet Details

Subnet Name
internal

Network Address ⓘ
10.1.1.0/24

IP Version
IPv4

Gateway IP ⓘ
10.1.1.1

☐ Disable Gateway

Creates a subnet associated with the network. You need to enter a valid "Network Address" and "Gateway IP". If you did not enter the "Gateway IP", the first value of a network will be assigned by default. If you do not want gateway please check the "Disable Gateway" checkbox. Advanced configuration is available by clicking on the "Subnet Details" tab.

Cancel « Back Next »

This network will be the virtual one hosting the VMs

This gateway will be a virtual gateway

Create Network

Network Subnet Subnet Details

☒ Enable DHCP Specify additional attributes for the subnet.

Allocation Pools ⓘ
10.1.1.2, 10.1.1.100

DNS Name Servers ⓘ
8.8.8.8

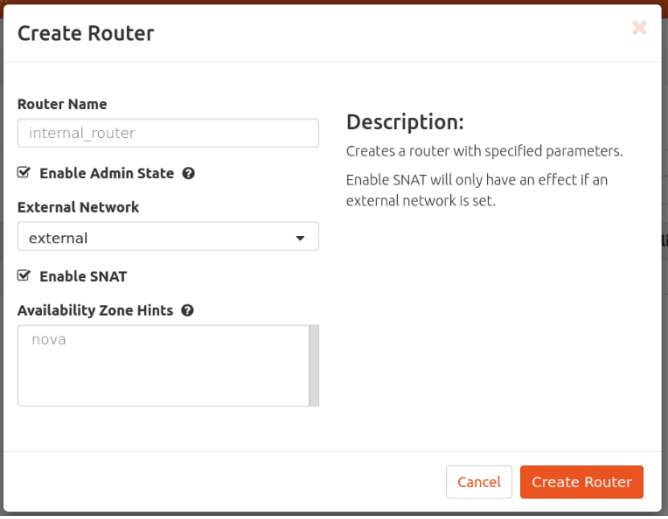
Host Routes ⓘ

Cancel « Back Create

Create a virtual router for the internal net

- Create a virtual router or gateway to link the virtual network with the external one
- Project -> Network -> Routers -> Create Router

The router will have an interface on the external network



The screenshot shows a 'Create Router' dialog box with the following fields and options:

- Router Name:** A text input field containing 'internal_router'.
- Enable Admin State:** A checked checkbox with a help icon.
- External Network:** A dropdown menu showing 'external'.
- Enable SNAT:** A checked checkbox.
- Availability Zone Hints:** A text input field containing 'nova'.
- Description:** A text area with the text: 'Creates a router with specified parameters. Enable SNAT will only have an effect if an external network is set.'
- Buttons:** 'Cancel' and 'Create Router' at the bottom right.

Connect router to the internal network

- The router needs to be explicitly connected to the internal network
- Enter in the router configuration, interfaces tab
- Then Add Interface:

Add Interface

Subnet *
internal: 10.1.1.0/24 (internal)

IP Address (optional) ?
10.1.1.1

Description:

You can connect a specified subnet to the router.

If you don't specify an IP address here, the gateway's IP address of the selected subnet will be used as the IP address of the newly created interface of the router. If the gateway's IP address is in use, you must use a different address which belongs to the selected subnet.

CancelSubmit

The screenshot shows the OpenStack Dashboard interface for configuring a router. The breadcrumb trail is Project / Network / Routers / internal_router. The 'internal_router' configuration page has three tabs: Overview, Interfaces (selected), and Static Routes. On the right side of the 'Interfaces' tab, there are buttons for '+ Add Interface' and 'Delete Interfaces'. Below these buttons is a table displaying the current interfaces. The table has columns for Name, Fixed IPs, Status, Type, Admin State, and Actions. One interface is listed: (631c740e-8faa) with Fixed IP 192.168.1.27, Status Active, Type External Gateway, and Admin State UP. The 'Delete Interface' button is visible in the Actions column for this interface.

| Name | Fixed IPs | Status | Type | Admin State | Actions |
|-----------------|--------------|--------|------------------|-------------|------------------|
| (631c740e-8faa) | 192.168.1.27 | Active | External Gateway | UP | Delete Interface |

Network configuration

- The network configuration is completed, and it has the following topology



Flavors

- A flavor is a configuration for a virtual machine
- It defines the amount of resources that are allocated to a VM at the time of creation
- The administrator must create at least one flavor
- Admin->Flavors->Create Flavor

Create Flavor

Flavor Information *

Flavor Access

Name *

ID ?

VCPUs *

RAM (MB) *

Root Disk (GB) *

Ephemeral Disk (GB)

Swap Disk (MB)

RX/TX Factor

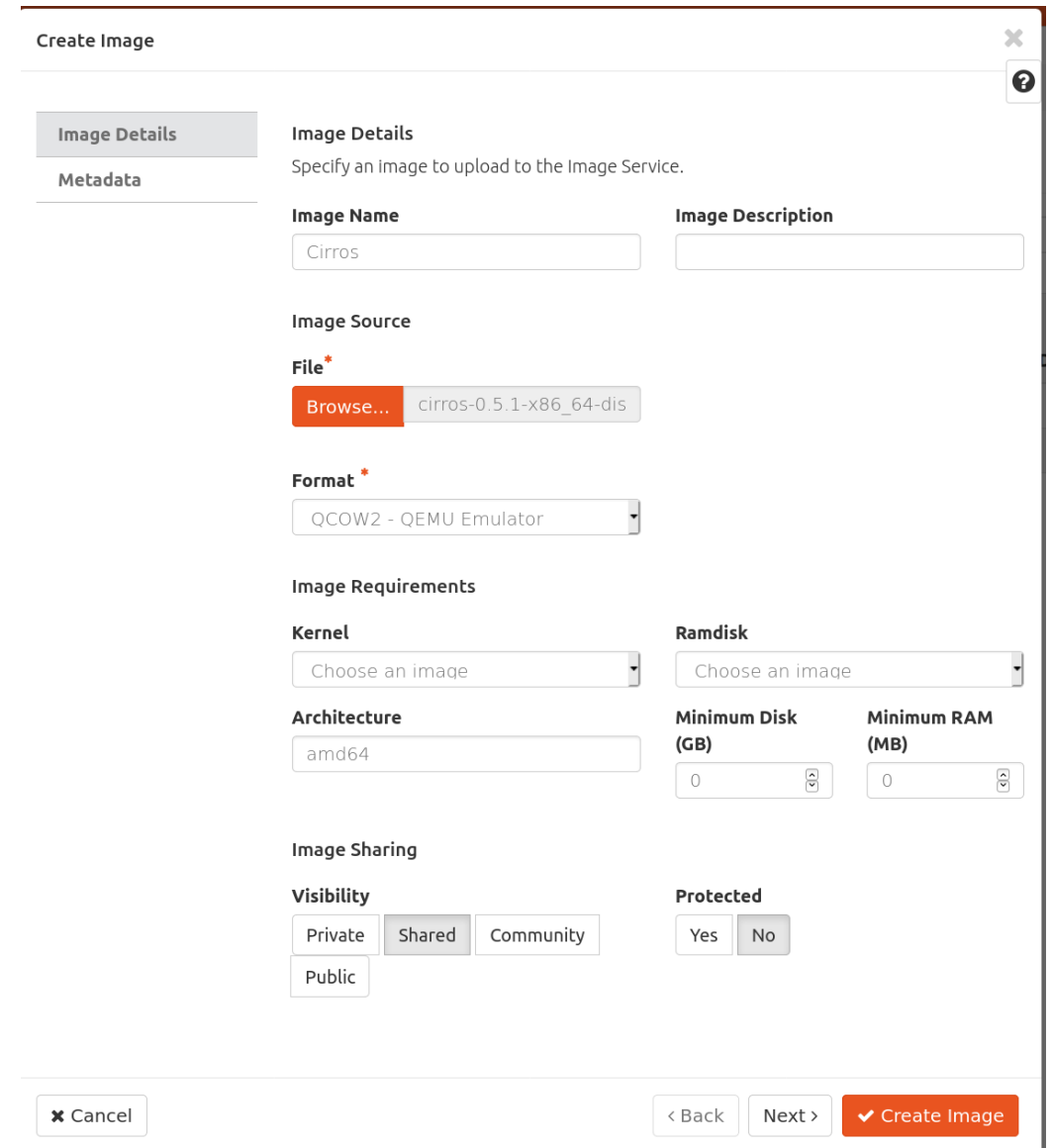
Flavors define the sizes for RAM, disk, number of cores, and other resources and can be selected when users deploy instances.

Cancel

Create Flavor

Images

- VMs are created from images, template of a virtual hard drive in which the OS is preinstalled
- At least one image must be imported
- Download a template in your PC using the following link:
 - http://download.cirros-cloud.net/0.4.0/cirros-0.4.0-x86_64-disk.img
- Admin->Images->Create Image
 - Select the downloaded image



The screenshot shows the 'Create Image' form in OpenStack. The form is divided into two tabs: 'Image Details' (selected) and 'Metadata'. The 'Image Details' tab contains the following sections:

- Image Details:** A section with the instruction 'Specify an image to upload to the Image Service.' It contains two input fields: 'Image Name' (with the value 'Cirros') and 'Image Description' (empty).
- Image Source:** A section with a 'File' label and a red asterisk. It includes a 'Browse...' button and a text input field containing 'cirros-0.5.1-x86_64-disk'.
- Format:** A section with a red asterisk and a dropdown menu showing 'QCOW2 - QEMU Emulator'.
- Image Requirements:** A section with four sub-sections:
 - Kernel:** A dropdown menu showing 'Choose an image'.
 - Ramdisk:** A dropdown menu showing 'Choose an image'.
 - Architecture:** A text input field containing 'amd64'.
 - Minimum Disk (GB):** A numeric input field with a value of '0'.
 - Minimum RAM (MB):** A numeric input field with a value of '0'.
- Image Sharing:** A section with two sub-sections:
 - Visibility:** A group of buttons: 'Private', 'Shared' (selected), 'Community', and 'Public'.
 - Protected:** A group of buttons: 'Yes' and 'No'.

At the bottom of the form, there are three buttons: 'Cancel', '< Back', and 'Next >', followed by a large orange 'Create Image' button.

Instantiate the first VM

- Project->Compute->Instances->Launch Instance

Launch Instance

Details

Please provide the initial hostname for the instance, the availability zone where it will be deployed, and the instance count. Increase the Count to create multiple instances with the same settings.

Source

Flavor

Networks

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Instance Name

Prova

Description

Availability Zone

nova

Count

1

Total Instances (10 Max)

10%

0 Current Usage

1 Added

9 Remaining

< Back

Next >

Launch Instance

Launch Instance

Details

Source

Flavor

Networks

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Instance source is the template used to create an instance. You can use an image, a snapshot of an instance (image snapshot), a volume or a volume snapshot (if enabled). You can also choose to use persistent storage by creating a new volume.

Select Boot Source

Image

Create New Volume

Yes

No

Volume Size (GB)

1

Delete Volume on Instance Delete

Yes

No

Allocated

| Name | Updated | Size | Type | Visibility |
|--------|-----------------|----------|-------|------------|
| Cirros | 4/21/20 3:41 PM | 15.58 MB | qcow2 | Shared |

Available

Click here for filters or full text search

| Name | Updated | Size | Type | Visibility |
|--------------------|---------|------|------|------------|
| No available items | | | | |

< Back

Next >

Launch Instance

Launch Instance

Details

Source

Flavor

Networks

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Flavors manage the sizing for the compute, memory and storage capacity of the instance.

Allocated

| Name | VCPUS | RAM | Total Disk | Root Disk | Ephemeral Disk | Public |
|-------|-------|-------|------------|-----------|----------------|--------|
| basic | 1 | 64 MB | 1 GB | 1 GB | 0 GB | Yes |

Available

Click here for filters or full text search

| Name | VCPUS | RAM | Total Disk | Root Disk | Ephemeral Disk | Public |
|------|-------|-----|------------|-----------|----------------|--------|
|------|-------|-----|------------|-----------|----------------|--------|

< Back

Next >

Launch Instance

Launch Instance

Details

Source

Flavor

Networks

Network Ports

Security Groups

Key Pair

Configuration

Server Groups

Scheduler Hints

Metadata

Networks provide the communication channels for instances in the cloud.

Allocated

Select networks from those listed below.

| Network | Subnets Associated | Shared | Admin State | Status | |
|---------|--------------------|----------|-------------|--------|--------|
| 1 | internal | internal | No | Up | Active |

Available

Select at least one network

Click here for filters or full text search

| Network | Subnets Associated | Shared | Admin State | Status | |
|----------|--------------------|--------|-------------|--------|--------|
| external | external | | Yes | Up | Active |

< Back

Next >

Launch Instance

Check the image running

- Admin->Compute->Instances
- The image runs on a certain host and has a certain IP in the internal network

```
root@DVQM5IJDG1MQD8B: ~
top - 13:51:21 up 1 day, 2:37, 1 user, load average: 1.27, 0.80, 0.50
Tasks: 170 total, 1 running, 113 sleeping, 0 stopped, 0 zombie
%Cpu(s): 50.0 us, 2.2 sy, 0.0 ni, 47.1 id, 0.7 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem : 7688188 total, 2730100 free, 2249252 used, 2708836 buff/cache
KiB Swap: 999420 total, 991628 free, 7792 used, 5785704 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR   S  %CPU  %MEM    TIME+  COMMAND
 787927 libvirt+  20   0  755248 118408 19560 S 100.7   1.5   3:04.96 qemu-system-x86
    974 root       20   0  680268   3136   1204 S   0.7   0.0   5:13.81 lxcfs
121078 164045    20   0 1089212 641884 28552 S   0.7   8.3   7:18.82 ceph-mon
122470 164045    20   0  953564 236648 35820 S   0.3   3.1   2:45.91 ceph_mgr
125709 ceph       20   0  860856  84132 32976 S   0.3   1.1   2:58.26 ceph-osd
745584 neutron   20   0 331580 129028 15364 S   0.3   1.7   0:12.89 /usr/bin/python
789509 root       20   0   45852   3992   3288 R   0.3   0.1   0:00.03 top
    1 root       20   0 225988    768    464 S   0.0   0.1   3:02.24 systemd
    2 root       20   0      0      0      0 S   0.0   0.0   0:00.06 kthreadd
    4 root       0 -20      0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H
    6 root       0 -20      0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
    7 root       20   0      0      0      0 S   0.0   0.0   0:09.37 ksoftirqd/0
    8 root       20   0      0      0      0 I   0.0   0.0   0:53.60 rcu_sched
    9 root       20   0      0      0      0 I   0.0   0.0   0:00.00 rcu_bh
   10 root       rt   0      0      0      0 S   0.0   0.0   0:00.27 migration/0
   11 root       rt   0      0      0      0 S   0.0   0.0   0:00.28 watchdog/0
   12 root       20   0      0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
   13 root       20   0      0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
   14 root       rt   0      0      0      0 S   0.0   0.0   0:00.25 watchdog/1
```

Instances

Project Name = Filter Delete Instances

Displaying 1 item

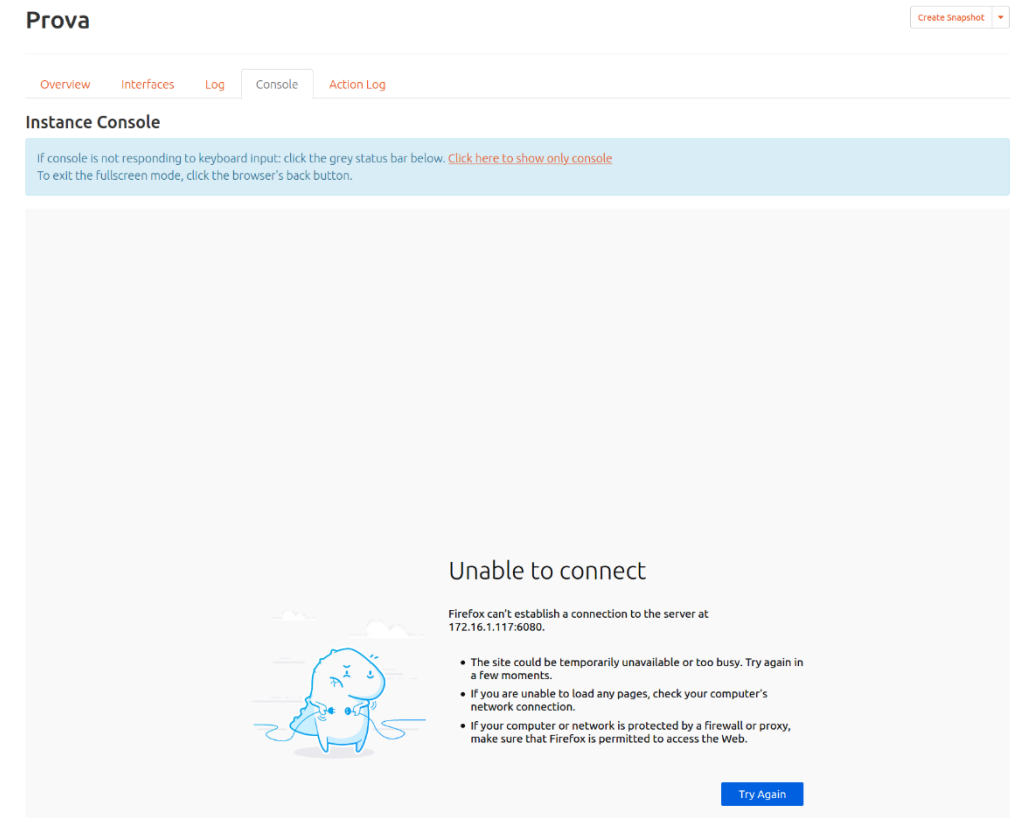
| <input type="checkbox"/> | Project | Host | Name | Image Name | IP Address | Flavor | Status | Task | Power State | Age | Actions |
|--------------------------|---------|-----------------|-------|------------|------------|--------|--------|------|-------------|-----------|--|
| <input type="checkbox"/> | admin | DVQM5IJDG1MQD8B | Prova | Cirros | 10.1.1.56 | basic | Active | None | Running | 2 minutes | Rescue Instance |

Displaying 1 item

Connect to the VM

- Project -> Compute -> Instances
- Select the instance and then the Console tab

VNC interface is provided by the nova controller, we need to configure a port forwarding to forward requests from the IP address of the web interface to the 6080 port of the nova controller



Forward VNC requests

- Retrieve the internal IP address of the container that hosts the nova controller
- Run `juju status` (on the manager) and retrieve the IP address of nova-cloud-controller

```
neutron-api/0* active idle 0/lxd/0 252.2.117.127 9090/tcp Unit is ready
neutron-gateway/0* active idle 0 172.16.2.117 Unit is ready
nova-cloud-controller/0* active idle 0/lxd/4 252.2.117.160 8774/tcp,8775/tcp Unit is ready
nova-compute/0* active idle 1 172.16.2.151 Unit is ready
  neutron-openvswitch/1 active idle 172.16.2.151 Unit is ready
nova-compute/1 active idle 2 172.16.2.152 Unit is ready
nova-compute/2 active idle 3 172.16.2.153 Unit is ready
```

- Install the following rules on the controller

```
iptables -t nat -A PREROUTING -i eth0 -p tcp -m tcp --dport
6080 -j DNAT --to-destination 252.2.117.160:6080
```

```
iptables -t nat -A POSTROUTING -d 252.2.117.160/32 -o eth0 -j
MASQUERADE
```

If the configuration is wrong...

- A module can be reconfigured as follows:

```
juju config --file dashboard.yaml openstack-dashboard
```

Connect to the VM

```
$
login as 'cirros' user. default password: 'gocubsgo'. use 'sudo' for root.
dsdfs login:
login as 'cirros' user. default password: 'gocubsgo'. use 'sudo' for root.
dsdfs login:
login as 'cirros' user. default password: 'gocubsgo'. use 'sudo' for root.
dsdfs login:
login as 'cirros' user. default password: 'gocubsgo'. use 'sudo' for root.
dsdfs login: _
```

Connected (unencrypted) to: QEMU (instance-00000000b)

```
^K64 bytes from 10.1.1.1: seq=1 ttl=64 time=1.271 ms
^C
--- 10.1.1.1 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 1.271/9.001/16.731 ms
$ ifconfig
eth0      Link encap:Ethernet  HWaddr FA:16:3E:40:EB:66
          inet addr:10.1.1.70  Bcast:10.1.1.255  Mask:255.255.255.0
          inet6 addr: fe80::f816:3eff:fe40:eb66/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1458  Metric:1
          RX packets:88 errors:0 dropped:0 overruns:0 frame:0
          TX packets:119 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9086 (8.8 KiB)  TX bytes:10524 (10.2 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

$
```

Connected (unencrypted) to: QEMU (instance-00000000b)

```
inet addr:127.0.0.1  Mask:255.0.0.0
inet6 addr: ::1/128 Scope:Host
UP LOOPBACK RUNNING  MTU:65536  Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1
RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

$ ping 10.1.1.1
PING 10.1.1.1 (10.1.1.1): 56 data bytes
64 bytes from 10.1.1.1: seq=0 ttl=64 time=1.723 ms
64 bytes from 10.1.1.1: seq=1 ttl=64 time=0.837 ms
64 bytes from 10.1.1.1: seq=2 ttl=64 time=0.719 ms
^K64 bytes from 10.1.1.1: seq=3 ttl=64 time=0.697 ms
64 bytes from 10.1.1.1: seq=4 ttl=64 time=0.719 ms
^C
--- 10.1.1.1 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 0.697/0.939/1.723 ms
$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8): 56 data bytes
^C
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 0 packets received, 100% packet loss
$ _
```

Connect the VM to the internet

- VMs cannot connect to the internet, the physical gateway with address 192.168.1.1 that reroutes traffic from/to VMs to/from the internet does not exist on vlan2 network (the virtual network that we created to mimic a missing real public network)
- Let us configure the network node (the controller in our deployment) to be that node
- Configure the bridge br-ex (the bridge that collects all the traffic towards external networks) on the controller node with 192.168.1.1 address:

ethernets:

br-ex:

addresses:

- 192.168.1.1/24

```
root@J6Y5KZ00WNX7C42:~# ifconfig br-ex
br-ex: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.1 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::21d:d8ff:feb7:257b prefixlen 64 scopeid 0x20<link>
    ether 00:1d:d8:b7:25:7b txqueuelen 1000 (Ethernet)
    RX packets 65 bytes 3086 (3.0 KB)
    RX errors 0 dropped 215 overruns 0 frame 0
    TX packets 44 bytes 3496 (3.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Apply the configuration to /etc/netplan/01-netcfg.yaml and run **netplan apply**

Floating IPs

- VMs can be assigned a public IP address to connect to the internet via Floating IP addresses
- A Floating IP is an IP address that is assigned to a VM to access an external network
- The IP address is not actually assigned to the VM, instead, NAT is performed by the router
- To assign a floating IP address go to the menu of the instance and select 'Associate Floating IP'

| | | | | |
|--------------------------|-------|--------|----------------------------|-------|
| <input type="checkbox"/> | Prova | Cirros | 10.1.1.70, 192.168.1.87 | basic |
|--------------------------|-------|--------|----------------------------|-------|

The image shows two screenshots of the 'Manage Floating IP Associations' dialog box. The left screenshot shows the 'Request a new' button highlighted with a blue arrow. The right screenshot shows the 'IP Address' dropdown set to '192.168.1.87'.

Manage Floating IP Associations

Request a new

IP Address *
No floating IP addresses alloc... +
Select the IP address you wish to associate with the selected instance or port.

Port to be associated *
Prova: 10.1.1.70

Cancel Associate

Manage Floating IP Associations

IP Address *
192.168.1.87 +
Select the IP address you wish to associate with the selected instance or port.

Port to be associated *
Prova: 10.1.1.70

Cancel Associate

Test connectivity towards the gateway

- Run ping 192.168.1.1 on the VM

```
$  
$  
$  
$  
$ ping 192.168.1.1  
PING 192.168.1.1 (192.168.1.1): 56 data bytes  
64 bytes from 192.168.1.1: seq=0 ttl=63 time=3.501 ms  
64 bytes from 192.168.1.1: seq=1 ttl=63 time=0.923 ms  
64 bytes from 192.168.1.1: seq=2 ttl=63 time=0.708 ms  
64 bytes from 192.168.1.1: seq=3 ttl=63 time=0.704 ms  
^K64 bytes from 192.168.1.1: seq=4 ttl=63 time=0.885 ms  
^C  
--- 192.168.1.1 ping statistics ---  
5 packets transmitted, 5 packets received, 0% packet loss  
round-trip min/avg/max = 0.704/1.344/3.501 ms  
$
```

Configure the gateway

- The next step is to configure the controller to forward the traffic from br-ex to eth0 and to perform NAT to allow all the VMs on the platform to send and receive data via the controller
- IPTABLES configuration

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
```

Allow ingress traffic

- By default ingress traffic to VMs is disabled
- To enable some type of traffic:
 - Network->Security Groups->Manage Rules->Add Rule

Add Rule

Rule *

Custom TCP Rule

Description ?

Direction

Ingress

Open Port *

Port

Port * ?

Remote * ?

CIDR

CIDR * ?

0.0.0.0/0

Description:

Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

Rule:

You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

Open Port/Port Range:

For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

Remote:

You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

Cancel

Add

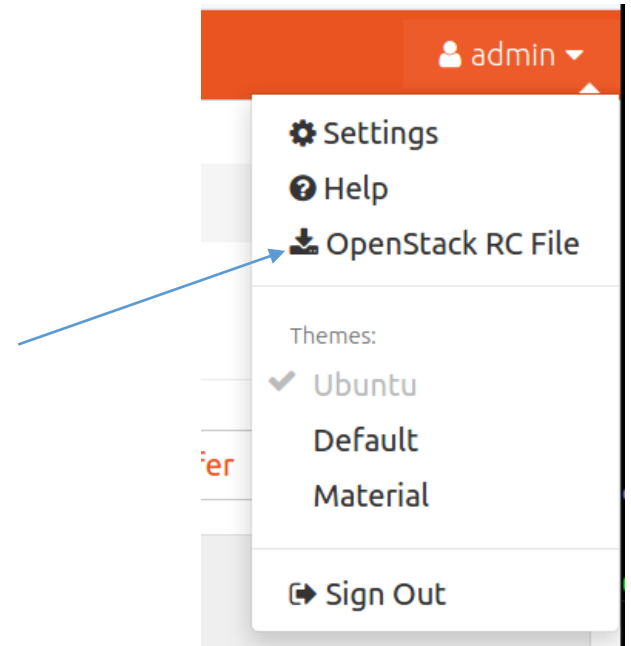
OpenStack SDK

- OpenStack provides an SDK to create applications that interact with the platform, one SDK version is available for python3
- To install it (e.g. on the controller node):

```
apt-get install python3-pip  
pip3 install openstacksdk
```

- Download the authentication file and authenticate
- Upload the file on the controller and activate

```
source openstack.sh
```



Simple Application

```
import openstack
```

```
# Connect
```

```
conn=openstack.connect()
```

```
# list images
```

```
for image in conn.compute.images():  
    print(image)
```

```
# list VMs
```

```
print("List Servers:")
```

```
for server in conn.compute.servers():  
    print(server)
```

Command line

- OpenStack offers a command line interface alternative to the web one
`snap install --classic openstackclients`

```
wget https://cloud-images.ubuntu.com/bionic/current/bionic-server-cloudimg-amd64.img
```

```
source admin.sh
```

```
openstack image create --public --disk-format qcow2 --  
container-format bare --file bionic-server-cloudimg-  
amd64.img Ubuntu
```

Other examples

- <https://github.com/openstack/openstacksdk/blob/master/examples/>