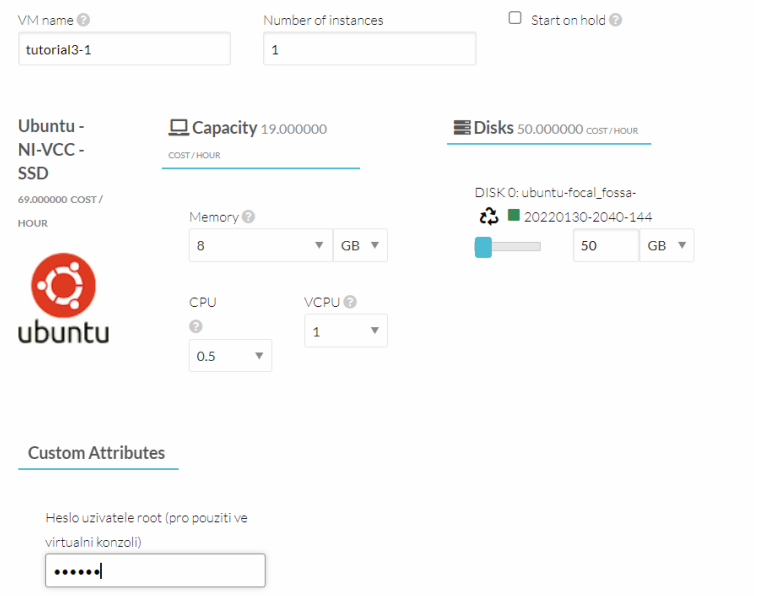
Tutorial 3

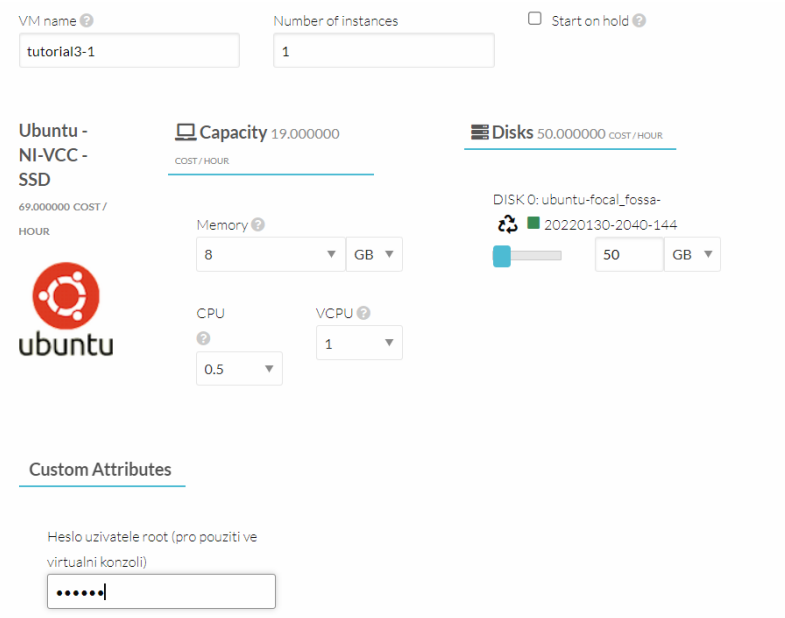
Deployment of OpenStack

Before the exercise, create 2 VMs, one 8 GB RAM and 0.5 CPUs, the other 2 GB RAM, 0.2 CPUs. Set up SSH access without a password from the first to itself and the second.

This is the first virtual machine.



This is the second virtual machine.



Sergio Teodoro Castellano Betancor

After creating the virtual machines, we have to add an user with the name of our personal computer username in my case “scast” and then in both virtual machines we have to add this new user to the sudo group with the command “`usermod -aG sudo scast`”.

Now from my home and with the vpn provided by the faculty we should be able to connect by ssh with the virtual machines.

```
root@tutorial3-1: /home/scast

C:\Users\scast>ssh 10.38.6.143
The authenticity of host '10.38.6.143 (10.38.6.143)' can't be established.
ECDSA key fingerprint is SHA256:z4S2DEEMW8wr0GhFkCPRka/uKMkNugCnCes1fRDN4JM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.38.6.143' (ECDSA) to the list of known hosts.
scast@10.38.6.143's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.4.0-96-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon 11 Apr 2022 06:17:01 PM UTC

  System load:  0.0               Processes:             111
  Usage of /:   5.4% of 48.96GB   Users logged in:       1
  Memory usage: 3%                IPv4 address for ens3: 10.38.6.143
  Swap usage:   0%


0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

scast@tutorial3-1:~$ sudo su
[sudo] password for scast:
root@tutorial3-1:/home/scast#
```


Then I will copy the ssh keys of each machine on the other machine so they can communicate by shh without password.

```
root@tutorial3-1:~/.ssh# root@tutorial3-1:~/.ssh# cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDalAPN1Ph0gF6S6XtJz2obGItdMzJDpQyohQt6HyON3Xk2QyMLit/Pdp6dICNKlx10
mpjyldRIx0DnwbYt9t5k6W2Q5TxwS5BUyjTqPseI7UkjvDrOGCGGrcDMKKWYtfedJbEFw5TBCae7Yc7Z2PRCSvfHK0eOLxYwwhut58nr
Jd0N5uIb7fIhpvaOsOkDcszKaxshRvg3sevxksvVJhPqTbumE1DMP+4iu9DlhwEIk0HCyVAd3RgbSaF8sacpZyjdzD0OZgSFxZpfi5mK
L8+G105dzGj+jhUtQUsBiM09+YBTpe2556E59AsluCEq2PX5HRU76gYY7v5JujpnnIgNTp514n7D2IwHMTv/PPo05X9uLYmJRNUfznvl
wMaVxCdkhyOFm+aeHX4Ou2reLqRs1wlMT627DRN1SYD69iPirJujp1qR9Fsdw3nkJXQO+Z/h9qi7vCTrbjt/HOCHG/vA1OieFzPsdZOT
10UuqK/PrE1+w7pMG4q4Ls3vKBgamP0= scast@DESKTOP-JG58M3C
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDAXtUf2Sd04bwl0zD43Yjrjmfdb2Op/0pZStgYMRU8evXkZf4r1ZCqdlQyIgC6bg87
nmfYJJFt/BMeoA9yBWOY2iHxJ6KSEhhEKXgZdfgWChdPyo1Uy83QXFkRJy73/LcK4tUXpl53xlzvi3vmxFrDDQEBa77qNfYlQ2ChBjGx
oNiIGsLgetFVs7DLyUBoNxvy/iJ+ZNoMXDjBoPCHc9xkR6fbvjQ/SPbHnDvWSYHcpK+FrQ7j/7n45x12RTzsvcGHOnk0mJ8R15JdgCYM
DO5FKDKYfcko1Weh2JnYnAi7GxiTMcDVVtc9GewVkf1btigm8q2UOwBiJsByZBOusp/meU2SmK6Vq6MGOM0jk5tZMCdi4KPPUwxwrvf4
VJllrI7HxHO7YTcx+AAeWI2kY6W85xtfZSFNYhxT1wK+jy3joZBy6Negspuf3xSVQ8kFpeQ5hK1cAQA2l1nS1a88Io0OwDyiKWNx6Ad2
X1QvNBiCyIk4en0jhwXg/R9BiqYMl7E= root@tutorial3-2
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDjLsAxmOaYLkGs4e6F3OADc30dmgd+0OYe03Fm/nFxVN1d8J/ldXYPpijYGczt5wxi
lxH4UMKhSuBhgaFDDgDGwQuNREVo2eXez69b8qCCgWJEws/i1Ic6Eum4LKoYqVDIUaebV3ZhOJdqGhKL/YEZ25TvS7caNOcbDVcfRYc9
ubvGKRYF9y/janCgb3bmosOYoDqm6pNuT8xk6iHUBeLlQsZ0NTOFbNi4s+S9PzyIPrNxTocnbpIgzzhpEOFmDWhuVy4g+SeO+kJ8qtCH
tGbOhaBPmUHZ6FlOv/vjeZ8tt1OJ6Z/1UCS29XuRHsuwXo/u6qGV8+bpiXywjgyXPGs9SRjN+Bc9OQeDptvGfizby1UAR6zRts2EMQjY
ITscSC56olvA0yF7wfSe9GY6/jtvS5Hy6DVudqUvWozSbo/VmKXbL7dMtmfQMfYJmdRrM9FaaQm4qpkOAmBaPCkjQXT3h22rLM2dhRqt
jF2UEKdWIW70HKSOOzG/HLJE1OuSQZ8= root@tutorial3-1
```

```
root@tutorial3-2:~/.ssh# root@tutorial3-2:~/.ssh# cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDalAPN1Ph0gF6S6XtJz2obGItdMzJDpQyohQt6HyON3Xk2QyMLit/Pdp6dICNKlx10
mpjyldRIx0DnwbYt9t5k6W2Q5TxwS5BUyjTqPseI7UkjvDrOGCGGrcDMKKWYtfedJbEFw5TBCae7Yc7Z2PRCSvfHK0eOLxYwwhut58nr
Jd0N5uIb7fIhpvaOsOkDcszKaxshRvg3sevxksvVJhPqTbumE1DMP+4iu9DlhwEIk0HCyVAd3RgbSaF8sacpZyjdzD0OZgSFxZpfi5mK
L8+G105dzGj+jhUtQUsBiM09+YBTpe2556E59AsluCEq2PX5HRU76gYY7v5JujpnnIgNTp514n7D2IwHMTv/PPo05X9uLYmJRNUfznvl
wMaVxCdkhyOFm+aeHX4Ou2reLqRs1wlMT627DRN1SYD69iPirJujp1qR9Fsdw3nkJXQO+Z/h9qi7vCTrbjt/HOCHG/vA1OieFzPsdZOT
10UuqK/PrE1+w7pMG4q4Ls3vKBgamP0= scast@DESKTOP-JG58M3C
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQDjLsAxmOaYLkGs4e6F3OADc30dmgd+0OYe03Fm/nFxVN1d8J/ldXYPpijYGczt5wxi
lxH4UMKhSuBhgaFDDgDGwQuNREVo2eXez69b8qCCgWJEws/i1Ic6Eum4LKoYqVDIUaebV3ZhOJdqGhKL/YEZ25TvS7caNOcbDVcfRYc9
ubvGKRYF9y/janCgb3bmosOYoDqm6pNuT8xk6iHUBeLlQsZ0NTOFbNi4s+S9PzyIPrNxTocnbpIgzzhpEOFmDWhuVy4g+SeO+kJ8qtCH
tGbOhaBPmUHZ6FlOv/vjeZ8tt1OJ6Z/1UCS29XuRHsuwXo/u6qGV8+bpiXywjgyXPGs9SRjN+Bc9OQeDptvGfizby1UAR6zRts2EMQjY
ITscSC56olvA0yF7wfSe9GY6/jtvS5Hy6DVudqUvWozSbo/VmKXbL7dMtmfQMfYJmdRrM9FaaQm4qpkOAmBaPCkjQXT3h22rLM2dhRqt
jF2UEKdWIW70HKSOOzG/HLJE1OuSQZ8= root@tutorial3-1
```

Now we will continue by deleting firewall because it breaks open stack installation by executing the command "`apt purge firewall`" in both virtual machines.

```
root@tutorial3-1:~# apt purge firewalld
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  gir1.2-nm-1.0 ipset libipset13 libjansson4 libnftables1 libnm0 python3-decorator python3-firewall
  python3-nftables python3-selinux python3-slip python3-slip-dbus
Use 'sudo apt autoremove' to remove them.
```

```
root@tutorial3-2:~# apt purge firewalld
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  gir1.2-nm-1.0 ipset libipset13 libjansson4 libnftables1 libnm0 python3-decorator python3-firewall
```

(BOTH MACHINES)

Now we are going to start installing open stack with Kolla. First, we will install dependencies, for that we will run the next commands "`apt update`", "`apt install python3-dev libffi-dev gcc libssl-dev`" and "`apt install python3-pip`" in both virtual machines, after installing pip we need to make sure that we upgrade pip to the latest version executing "`pip3 install -U pip`".

```
Processing triggers for libc-bin (2.31-0ubuntu9.2) ...
root@tutorial3-1:~# pip3 install -U pip
Collecting pip
  Downloading pip-22.0.4-py3-none-any.whl (2.1 MB)
     |████████████████████████████████| 2.1 MB 9.4 MB/s
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 20.0.2
    Not uninstalling pip at /usr/lib/python3/dist-packages, outside environment /usr
    Can't uninstall 'pip'. No files were found to uninstall.
Successfully installed pip-22.0.4
```

The next step is to install ansible but we need to install it from pip by executing "`pip install -U 'ansible<3.0'`" **only on the first virtual machine.**

```
Successfully installed pip-22.0.4
root@tutorial3-1:~# pip install -U 'ansible<3.0'
Collecting ansible<3.0
  Downloading ansible-2.10.7.tar.gz (29.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 29.9/29.9 MB 20.5 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Collecting ansible-base<2.11,>=2.10.5
  Downloading ansible-base-2.10.17.tar.gz (6.1 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 6.1/6.1 MB 38.2 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: PyYAML in /usr/lib/python3/dist-packages (from ansible-base<2.11,>=2.10.5
->ansible<3.0) (5.3.1)
```

Now, we have to install Kolla ansible by executing "`pip3 install git+https://opendev.org/openstack/kolla-ansible@stable/xena`", then we have to create kola directory with "`mkdir -p /etc/kolla`" then, copy globals.yml and passwords.yml with "`cp -r /usr/local/share/kolla-ansible/etc_examples/kolla/* /etc/kolla`", then copy all-in-one and multinode inventory files to the current directory with "`cp /usr/local/share/kolla-ansible/ansible/inventory/* .`"

The next part is going to be to edit the configuration files,"mkdir /etc/ansible/" for this we will edit "`/etc/ansible/ansible.cfg`"

```
root@tutorial3-1: ~
[defaults]
host_key_checking=False
pipelining=True
forks=100
~
```

Now we need to customize the multimode file with the next aspects.

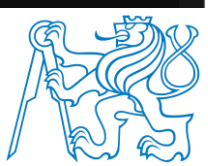
```
# These initial groups are the only groups required to be modified. The
# additional groups are for more control of the environment.
[control]
# These hostname must be resolvable from your deployment host
10.38.6.143

# The above can also be specified as follows:
#control[01:03]     ansible_user=kolla

# The network nodes are where your l3-agent and loadbalancers will run
# This can be the same as a host in the control group
[network:children]
control

[compute]
10.38.6.149

[monitoring:children]
control

# When compute nodes and control nodes use different interfaces,
# you need to comment out "api_interface" and other interfaces from the globals.yml
# and specify like below:
#compute01 neutron_external_interface=eth0 api_interface=em1 storage_interface=em1 tunnel_interface=em1

[storage:children]
compute
```

If we try now to ping with the command “`ansible -i multinode all -m ping`” we will see these three successes.

```
root@tutorial3-1:~# ansible -i multinode all -m ping
[WARNING]: Invalid characters were found in group names but not replaced, use -vvvv to see details
[DEPRECATION WARNING]: Distribution Ubuntu 20.04 on host localhost should use /usr/bin/python3, but is
using /usr/bin/python for backward compatibility with prior Ansible releases. A future Ansible release
will default to using the discovered platform python for this host. See
https://docs.ansible.com/ansible/2.10/reference_appendices/interpreter_discovery.html for more
information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
localhost | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
[DEPRECATION WARNING]: Distribution Ubuntu 20.04 on host 10.38.6.149 should use /usr/bin/python3, but
is using /usr/bin/python for backward compatibility with prior Ansible releases. A future Ansible
release will default to using the discovered platform python for this host. See
https://docs.ansible.com/ansible/2.10/reference_appendices/interpreter_discovery.html for more
information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
10.38.6.149 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
[DEPRECATION WARNING]: Distribution Ubuntu 20.04 on host 10.38.6.143 should use /usr/bin/python3, but
is using /usr/bin/python for backward compatibility with prior Ansible releases. A future Ansible
release will default to using the discovered platform python for this host. See
https://docs.ansible.com/ansible/2.10/reference_appendices/interpreter_discovery.html for more
information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by
setting deprecation_warnings=False in ansible.cfg.
10.38.6.143 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python"
    },
    "changed": false,
    "ping": "pong"
}
```

The next step will be to run Kolla first we have to execute “`kolla-genpwd`”, after this we will open the file /etc/kolla/globals.yml and edit the next lines:

```
# Valid options are ['centos', 'debian', 'rhel', 'ubuntu']
kolla_base_distro: "ubuntu"
```

```
# 'network_interface' as set in the Networking section below.
kolla_internal_vip_address: "10.38.6.143"
```

This is the ip address of the first virtual machine

```
# followed for other types of interfaces.
network_interface: "ens3"
```

```
# addresses for that reason.
neutron_external_interface: "veth0"
```

```
#enable_hacluster: "no"
enable_haproxy: "no"
```

Now we have to create virtual interface 0 for internet, so let's open the file /etc/network/interfaces on **both virtual machines** and apply the same changes

```
auto lo
iface lo inet loopback

auto ens3
iface ens3 inet static
  pre-up ip link add veth0 type veth peer name veth1
  pre-up ip link set veth0 up
  pre-up ip link set veth1 up
  address 10.38.6.208
  network 10.38.4.0
  netmask 255.255.252.0
  gateway 10.38.4.1

source /etc/network/interfaces.d/*.cfg
~
```

After this we must restart networking service by executing "`service networking restart`".

Then to check if everything is correct, we will execute the command "`ip link`" and we should get a response like this.

```
root@tutorial3-2:~# ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: ens3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT group default q
len 1000
    link/ether 02:00:0a:26:06:95 brd ff:ff:ff:ff:ff:ff
3: veth1@veth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group def
ault qlen 1000
    link/ether 0a:a4:54:2c:8c:b4 brd ff:ff:ff:ff:ff:ff
4: veth0@veth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP mode DEFAULT group def
ault qlen 1000
    link/ether be:91:c4:b1:b3:96 brd ff:ff:ff:ff:ff:ff
root@tutorial3-2:~#
```

After configuration is set, we can proceed to the deployment phase. Now **back in the first vm (ctl)** we need to setup basic host-level dependencies we have to run "`kolla-ansible -i ./multinode bootstrap-servers -e ansible_python_interpreter=/usr/bin/python3`" with the flag -e to specify the python interpreter.

```
skipping: [10.38.6.143]
skipping: [10.38.6.149]

TASK [baremetal : Update apt cache] ********************************************************************
*********************************************************************************************************
***
changed: [10.38.6.143]
changed: [10.38.6.149]

TASK [baremetal : Set firewall default policy] *********************************************************
*********************************************************************************************************
***
ok: [10.38.6.143]
ok: [10.38.6.149]
```

After bootstrap the next is prechecks so let's execute "`kolla-ansible -i ./multinode prechecks -e ansible_python_interpreter=/usr/bin/python3`"

```
PLAY [Apply role masakari] *****************************************************************************
skipping: no hosts matched

PLAY RECAP *********************************************************************************************
10.38.6.143                : ok=68   changed=0    unreachable=0    failed=0    skipped=60   rescued=0
 ignored=0
10.38.6.149                : ok=30   changed=0    unreachable=0    failed=0    skipped=30   rescued=0
 ignored=0
localhost                  : ok=11   changed=0    unreachable=0    failed=0    skipped=14   rescued=0
 ignored=0
```

And the last command "`kolla-ansible -i ./multinode deploy -e ansible_python_interpreter=/usr/bin/python3`"

```
TASK [common : Creating log volume] ********************************************************************
changed: [10.38.6.149]
changed: [10.38.6.143]

TASK [common : Link kolla_logs volume to /var/log/kolla] ***********************************************
changed: [10.38.6.149]
changed: [10.38.6.143]

RUNNING HANDLER [common : Restart fluentd container] ***************************************************
changed: [10.38.6.149]
changed: [10.38.6.143]

RUNNING HANDLER [common : Restart kolla-toolbox container] *********************************************
changed: [10.38.6.149]
changed: [10.38.6.143]
```

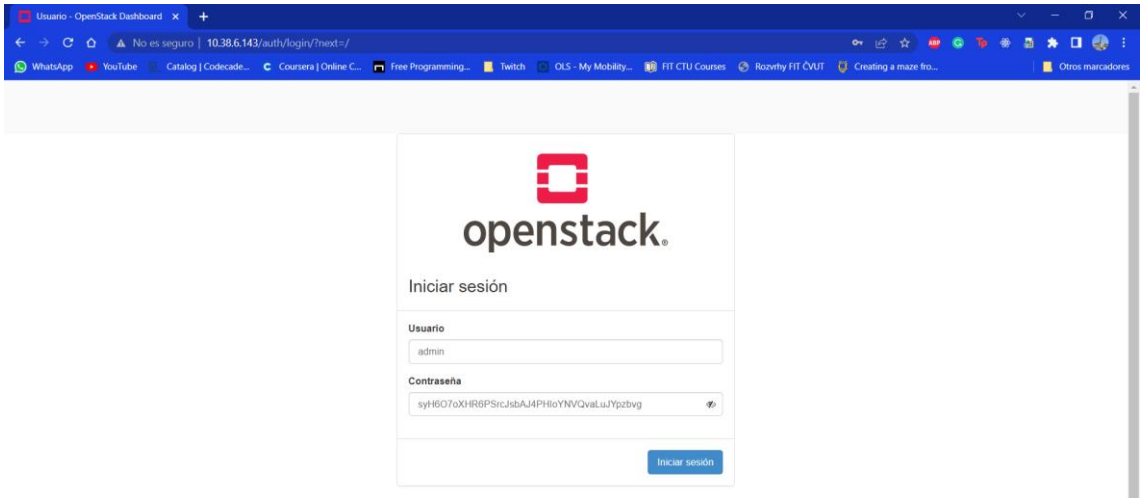
If we ssh on a second terminal and execute "`docker ps`" we can see that we are creating the containers for the database

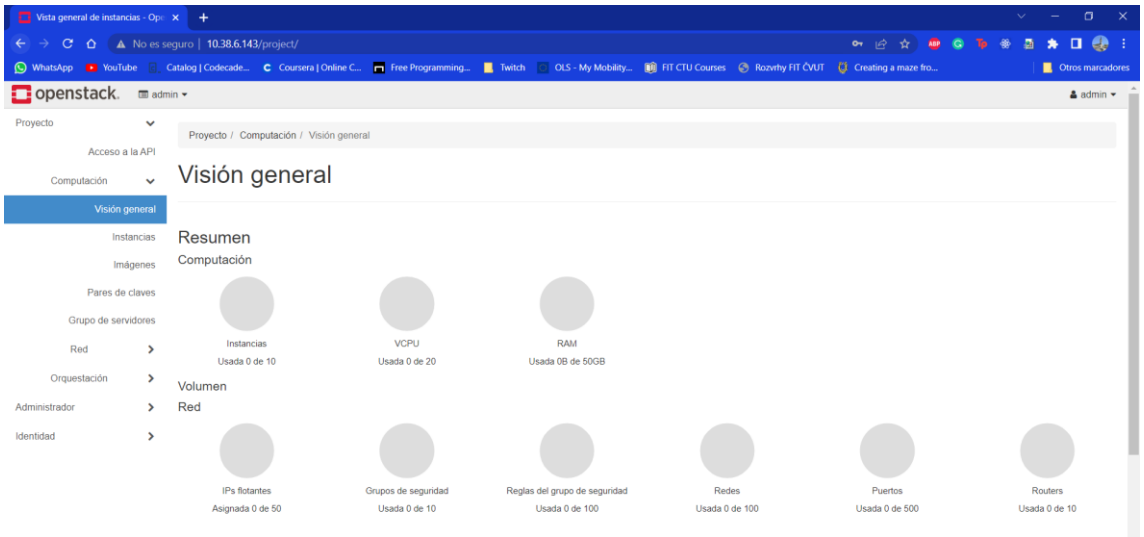
```
root@tutorial3-1:~# docker ps
CONTAINER ID   IMAGE                                                COMMAND                  CREATED
 STATUS              PORTS     NAMES
d8836a87929a   quay.io/openstack.kolla/ubuntu-source-mariadb-server:xena   "dumb-init -- kolla_…"   7 seconds ago
 Up 6 seconds                  mariadb
8be1f231592c   quay.io/openstack.kolla/ubuntu-source-cron:xena             "dumb-init --single-…"   45 seconds ago
 Up 45 seconds                 cron
c037105299ff   quay.io/openstack.kolla/ubuntu-source-kolla-toolbox:xena    "dumb-init --single-…"   About a minute ago
 Up About a minute             kolla_toolbox
27bdc2adba31   quay.io/openstack.kolla/ubuntu-source-fluentd:xena          "dumb-init --single-…"   About a minute ago
 Up About a minute             fluentd
root@tutorial3-1:~#
```

Now when the deploy is finished, let's install the openstack client with the command "`apt install python3-openstackclient`" and after that "`kolla-ansible post-deploy`" this will generate the file /etc/kolla/admin_openrc.sh

```
root@tutorial3-1:~# cat /etc/kolla/admin-openrc.sh
# Ansible managed

# Clear any old environment that may conflict.
for key in $( set | awk '{FS="="}  /^OS_/ {print $1}' ); do unset $key ; done
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_TENANT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=syH6O7oXHR6PSrcJsbAJ4PHIoYNVQvaLuJYpzbvg
export OS_AUTH_URL=http://10.38.6.143:35357/v3
export OS_INTERFACE=internal
export OS_ENDPOINT_TYPE=internalURL
export OS_IDENTITY_API_VERSION=3
export OS_REGION_NAME=RegionOne
export OS_AUTH_PLUGIN=password
root@tutorial3-1:~#
```

So now if we try to connect to the highlighted ip address via our browser we should be able to log in openstack graphical interface with user “admin” and the password is the one provided by the previous picture.



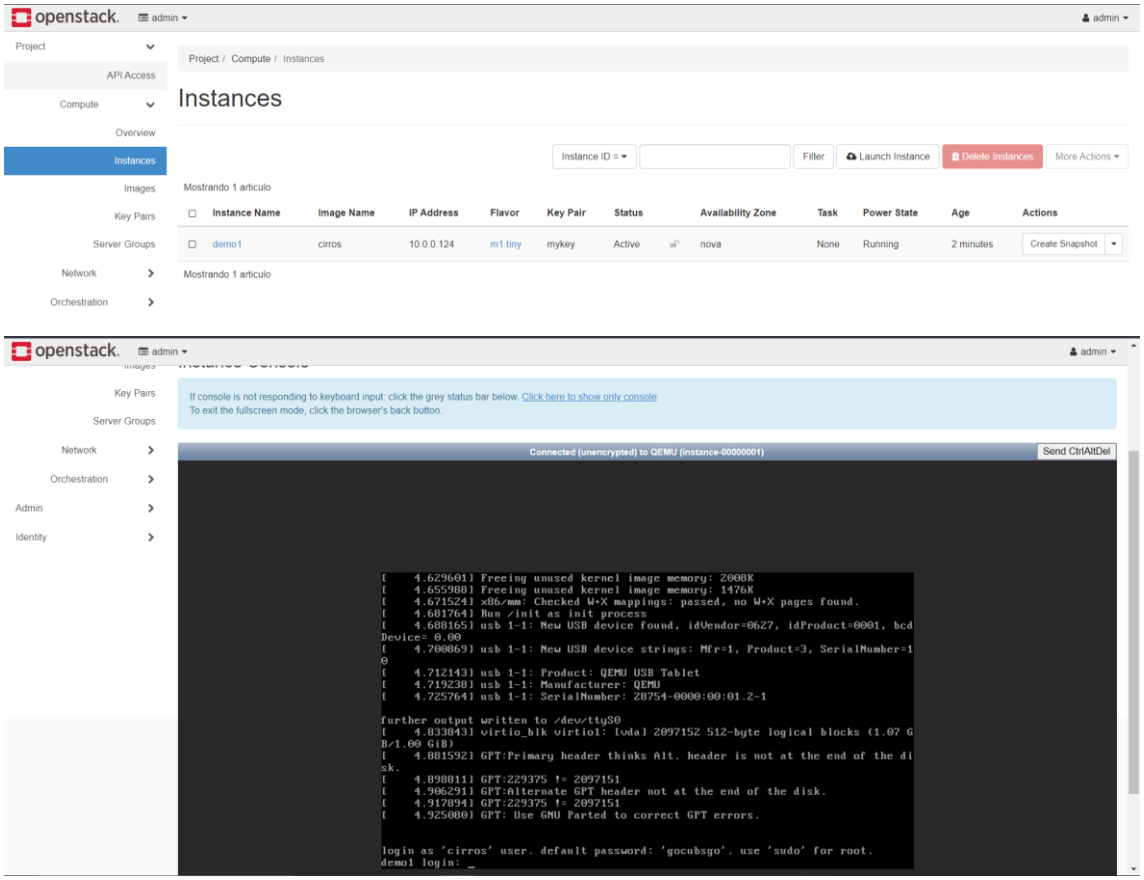


And then the last command “`/usr/local/share/kolla-ansible/init-runonce`”

```
Done.

To deploy a demo instance, run:

openstack server create \
    --image cirros \
    --flavor m1.tiny \
    --key-name mykey \
    --network demo-net \
    demo1
root@tutorial3-1:~# openstack server create \
>     --image cirros \
>     --flavor m1.tiny \
>     --key-name mykey \
>     --network demo-net \
>     demo1
```

When the script will be done it will show us how to deploy a demo instance of the machine and we will execute as is shown in the previous image.

Finally in the graphical interface we will have access to this instance.



Exercise 3: OpenStack which can run VMs on two nodes (either a second computing or compute service on the control node). The VMs must be able to ping each other.

Let's edit the multimode file and on the [compute] group let's add the ip address of the control node too.

root@tutorial3-ctl: ~

```
# These initial groups are the only groups required to be modified. The
# additional groups are for more control of the environment.
[control]
# These hostname must be resolvable from your deployment host
10.38.6.69

# The above can also be specified as follows:
#control[01:03]     ansible_user=kolla

# The network nodes are where your l3-agent and loadbalancers will run
# This can be the same as a host in the control group
[network:children]
control

[compute]
10.38.6.69
10.38.6.81
```

Now we have to make update the prechecks so the changes are reflected and the launch the deploy command for openstack again, that's why we will execute "`kolla-ansible -i ./multinode prechecks -e ansible_python_interpreter=/usr/bin/python3`"

```
PLAY [Apply role masakari] ******************************************************************************skipping: no hosts matched

PLAY RECAP **********************************************************************************************10.38.6.69                 : o
k=50   changed=0    unreachable=0    failed=0    skipped=82   rescued=0    ignored=0
10.38.6.81                 : ok=25   changed=0    unreachable=0    failed=0    skipped=31   rescued=0    ignored=0
localhost                  : ok=11   changed=0    unreachable=0    failed=0    skipped=14   rescued=0    ignored=0
```

and then "`kolla-ansible -i ./multinode deploy -e ansible_python_interpreter=/usr/bin/python3`"o

```
PLAY [Apply role masakari] ******************************************************************************skipping: no hosts matched

PLAY RECAP **********************************************************************************************10.38.6.69                 : ok=235  changed=29   unreachable=0    failed=0    skipped=123  rescued=0    ig
nored=0
10.38.6.81                 : ok=49   changed=1    unreachable=0    failed=0    skipped=43   rescued=0    ignored=0
localhost                  : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

root@tutorial3-ctl:~# kolla-ansible post-deploy
Post-Deploying Playbooks : ansible-playbook -e @/etc/kolla/globals.yml  -e @/etc/kolla/passwords.yml -e CONFIG_DIR=/etc/kolla  /usr/local/share/kolla-ansible/ansible/post-deploy.yml
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost
does not match 'all'

PLAY [Creating admin openrc file on the deploy node] ****************************************************
TASK [Gathering Facts] **********************************************************************************ok: [localhost]

TASK [Template out admin-openrc.sh] *********************************************************************ok: [localhost]

TASK [octavia : Template out octavia-openrc.sh] *********************************************************skipping: [localhost]

PLAY RECAP **********************************************************************************************localhost                  : ok=2    changed=0    unreachable=0    failed=0    skipped=1    rescued=0    ig
nored=0
```

Now if we execute the command "openstack host list" we will see both virtual machines as compute nodes.

```
root@tutorial3-ctl:~# openstack host list
+---------------+-----------+----------+
| Host Name     | Service   | Zone     |
+---------------+-----------+----------+
| tutorial3-ctl | conductor | internal |
| tutorial3-cmp | compute   | nova     |
| tutorial3-ctl | compute   | nova     |
| tutorial3-ctl | scheduler | internal |
+---------------+-----------+----------+
```

Now we will create a server specifying the compute node we want to use with the command "`openstack server create \`

`--image cirros \`

`--flavor m1.tiny \`

`--key-name mykey \`

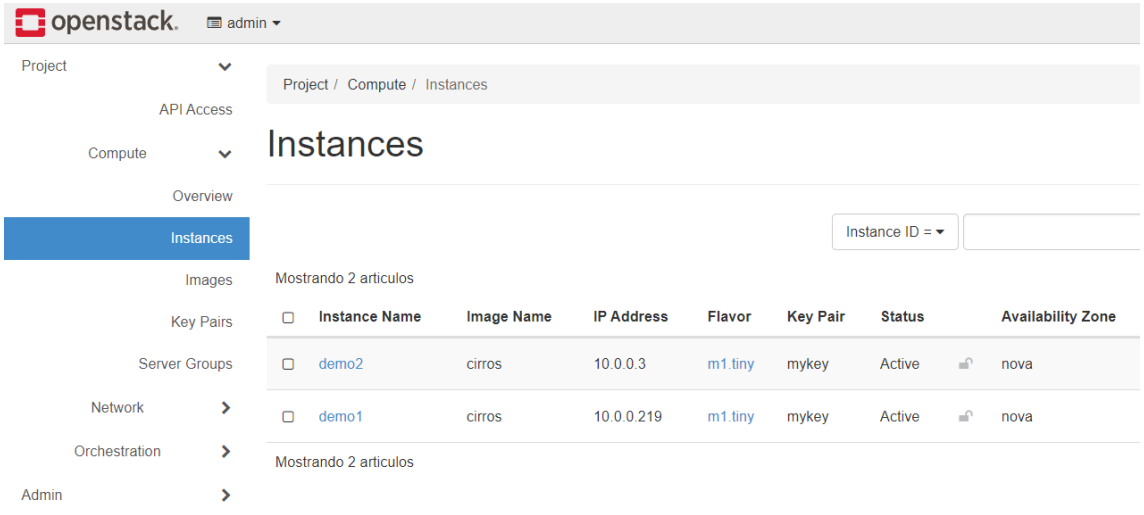
`--network demo-net \`

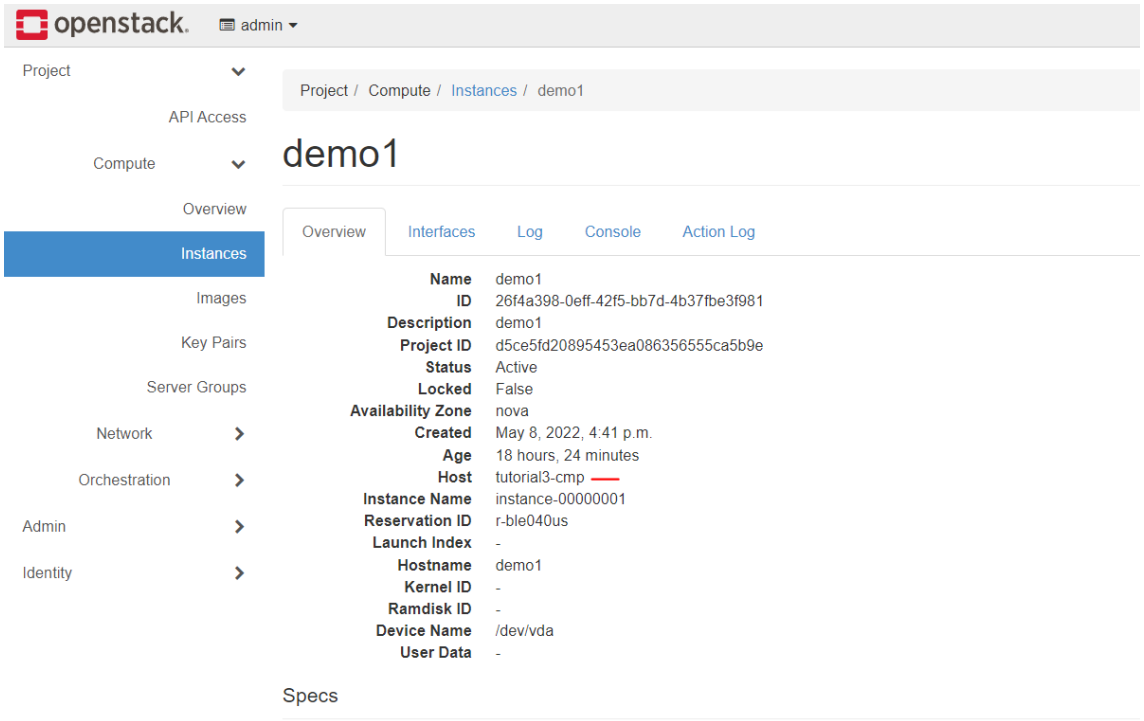
`--availability-zone nova:tutorial3-ctl \`

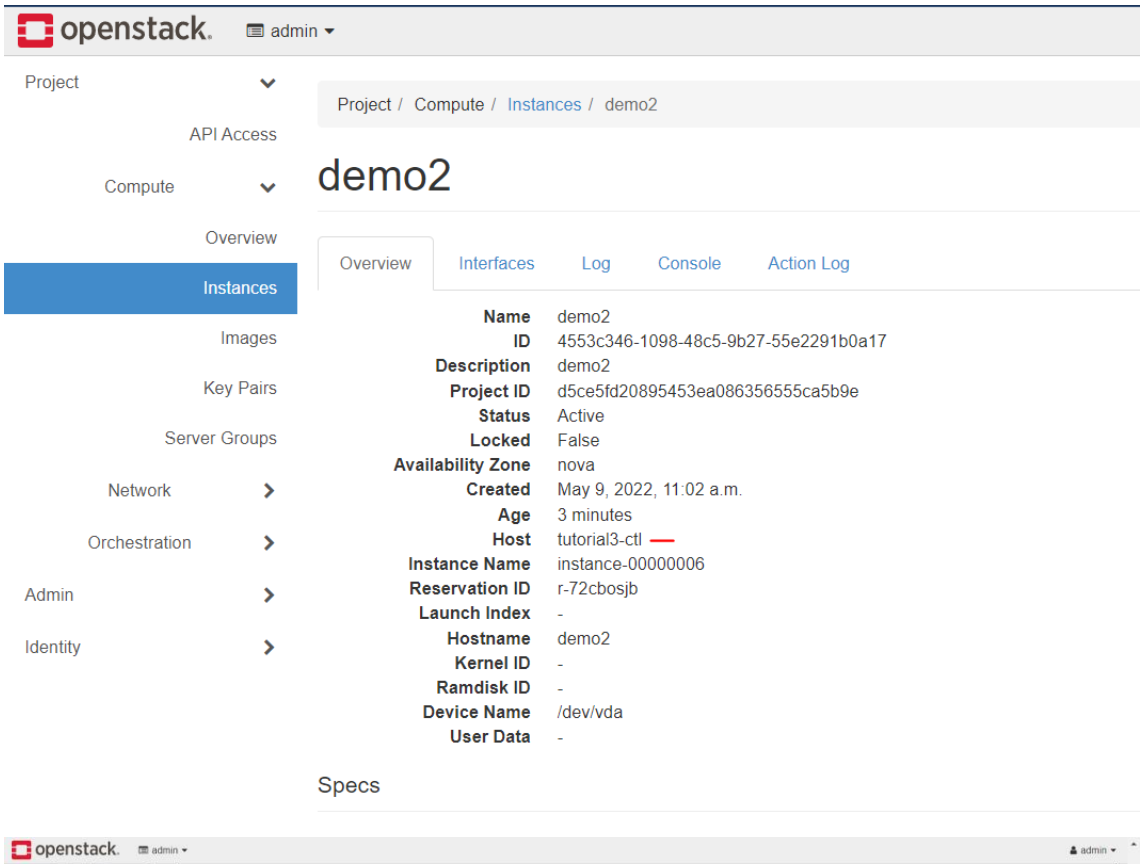
`demo2`"

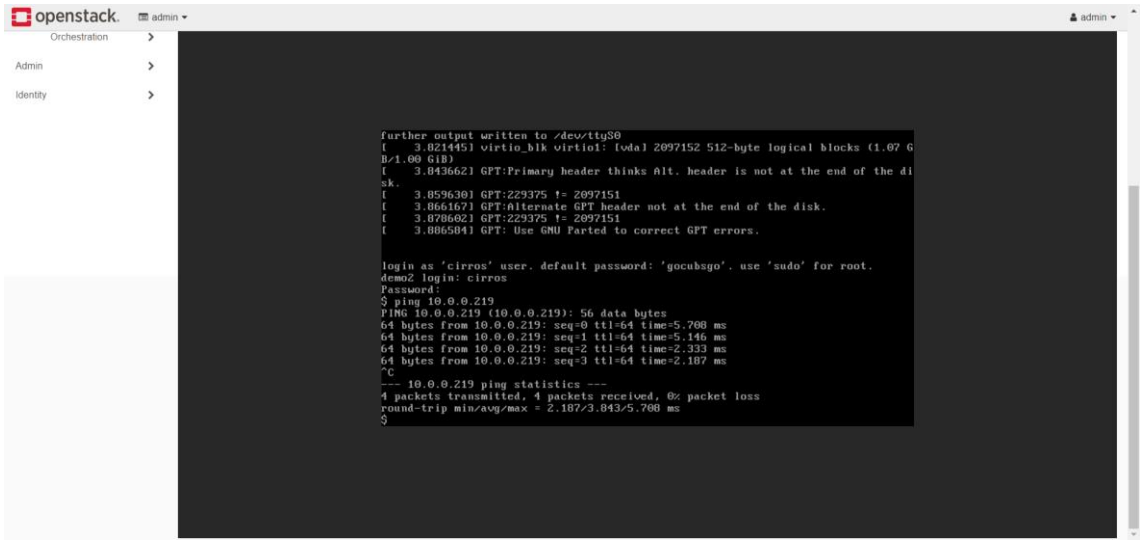
```
root@tutorial3-ctl:~# openstack server create \
>     --image cirros \
>     --flavor m1.tiny \
>     --key-name mykey \
>     --network demo-net \
>     --availability-zone nova:tutorial3-ctl \
>     demo2
+-------------------------------------+----------------------------------------------+
| Field                               | Value                                        |
+-------------------------------------+----------------------------------------------+
| OS-DCF:diskConfig                   | MANUAL                                       |
| OS-EXT-AZ:availability_zone         | nova                                         |
| OS-EXT-SRV-ATTR:host                | None                                         |
| OS-EXT-SRV-ATTR:hypervisor_hostname | None                                         |
| OS-EXT-SRV-ATTR:instance_name       |                                              |
| OS-EXT-STS:power_state              | NOSTATE                                      |
| OS-EXT-STS:task_state               | scheduling                                   |
| OS-EXT-STS:vm_state                 | building                                     |
| OS-SRV-USG:launched_at              | None                                         |
| OS-SRV-USG:terminated_at            | None                                         |
```

Finally, to check that everything has been made correctly if in the GUI we check the nodes of the instances we will see that they have different nodes, and they can ping between each other across the network.


openstack.
admin
Project
API Access
Compute
Overview
Instances
Images
Key Pairs
Server Groups
Network
Orchestration
Admin
Project / Compute / Instances
Instances
Instance ID =
Mostrando 2 articulos
Instance Name
Image Name
IP Address
Flavor
Key Pair
Status
Availability Zone
demo2
cirros
10.0.0.3
m1.tiny
mykey
Active
nova
demo1
cirros
10.0.0.219
m1.tiny
mykey
Active
nova
Mostrando 2 articulos


openstack.
admin
Project
API Access
Compute
Overview
Instances
Images
Key Pairs
Server Groups
Network
Orchestration
Admin
Identity
Project / Compute / Instances / demo1
demo1
Overview
Interfaces
Log
Console
Action Log
Name demo1
ID 26f4a398-0eff-42f5-bb7d-4b37fbe3f981
Description demo1
Project ID d5ce5fd20895453ea086356555ca5b9e
Status Active
Locked False
Availability Zone nova
Created May 8, 2022, 4:41 p.m.
Age 18 hours, 24 minutes
Host tutorial3-cmp
Instance Name instance-00000001
Reservation ID r-ble040us
Launch Index -
Hostname demo1
Kernel ID -
Ramdisk ID -
Device Name /dev/vda
User Data -
Specs


openstack.
admin
Project
API Access
Compute
Overview
Instances
Images
Key Pairs
Server Groups
Network
Orchestration
Admin
Identity
Project / Compute / Instances / demo2
demo2
Overview
Interfaces
Log
Console
Action Log
Name demo2
ID 4553c346-1098-48c5-9b27-55e2291b0a17
Description demo2
Project ID d5ce5fd20895453ea086356555ca5b9e
Status Active
Locked False
Availability Zone nova
Created May 9, 2022, 11:02 a.m.
Age 3 minutes
Host tutorial3-ctl
Instance Name instance-00000006
Reservation ID r-72cbosjb
Launch Index -
Hostname demo2
Kernel ID -
Ramdisk ID -
Device Name /dev/vda
User Data -
Specs


openstack.
admin
Orchestration
Admin
Identity
further output written to /dev/ttyS0
[3.021445] virtio_blk virtio1: [vda] 2097152 512-byte logical blocks (1.07 G
B/1.00 GiB)
[3.043662] GPT:Primary header thinks Alt. header is not at the end of the di
sk.
[3.059630] GPT:229375 != 2097151
[3.066167] GPT:Alternate GPT header not at the end of the disk.
[3.078602] GPT:229375 != 2097151
[3.086504] GPT: Use GNU Parted to correct GPT errors.
login as 'cirros' user. default password: 'gocubsgo'. use 'sudo' for root.
demo2 login: cirros
Password:
$ ping 10.0.0.219
PING 10.0.0.219 (10.0.0.219): 56 data bytes
64 bytes from 10.0.0.219: seq=0 ttl=64 time=5.708 ms
64 bytes from 10.0.0.219: seq=1 ttl=64 time=5.146 ms
64 bytes from 10.0.0.219: seq=2 ttl=64 time=2.333 ms
64 bytes from 10.0.0.219: seq=3 ttl=64 time=2.187 ms
^C
--- 10.0.0.219 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 2.187/3.843/5.708 ms
$

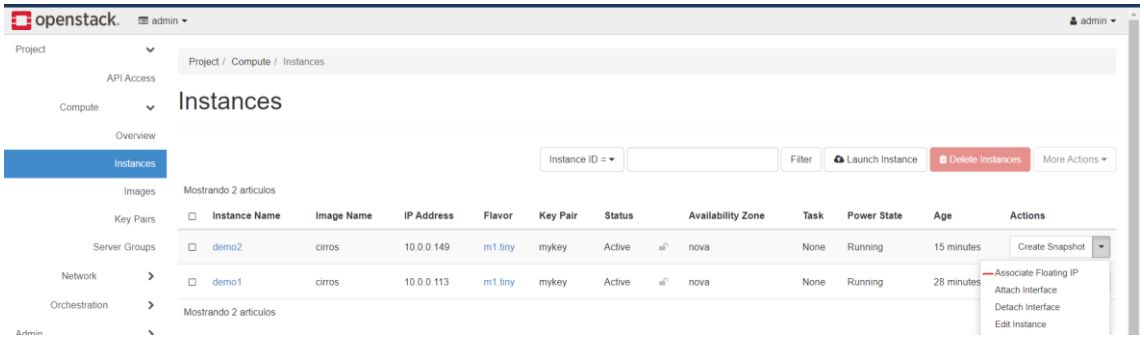
Exercise 4: As in the previous point, plus functional access from the VM to the Internet using NAT.

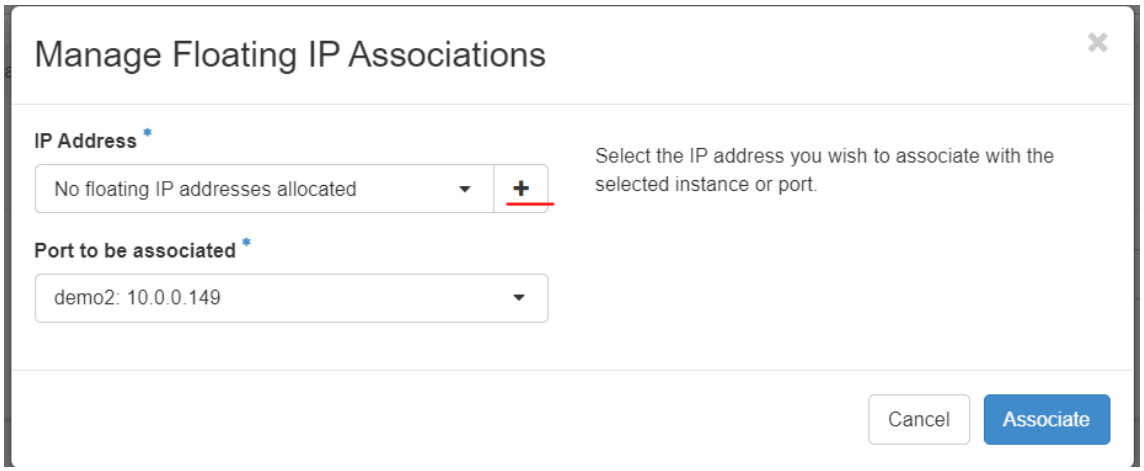
First, I will allow nat on iptables of the network node with the next rules:

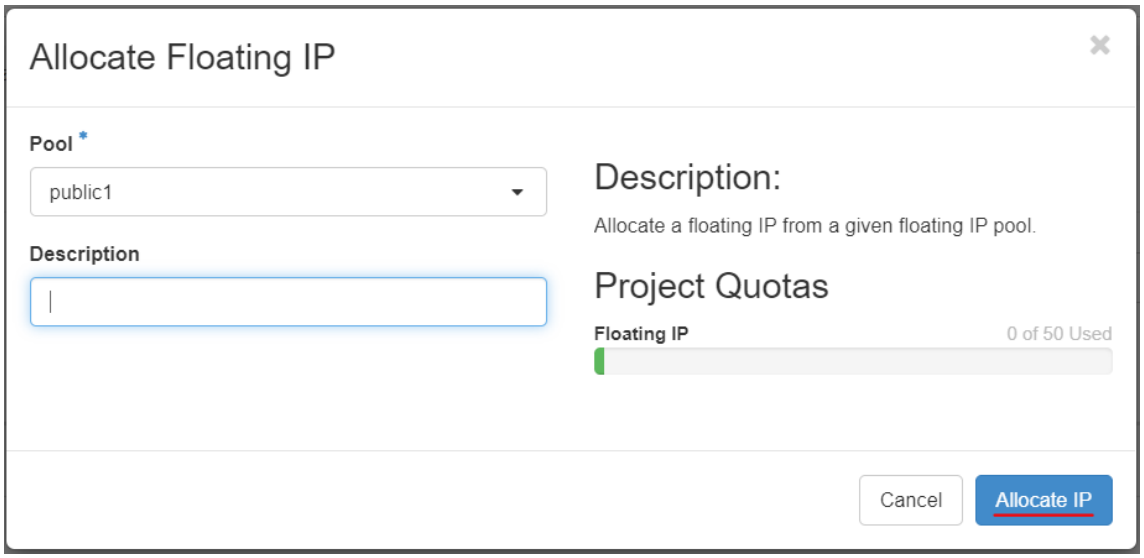
iptables -t nat -A POSTROUTING -o ens3 -j MASQUERADE

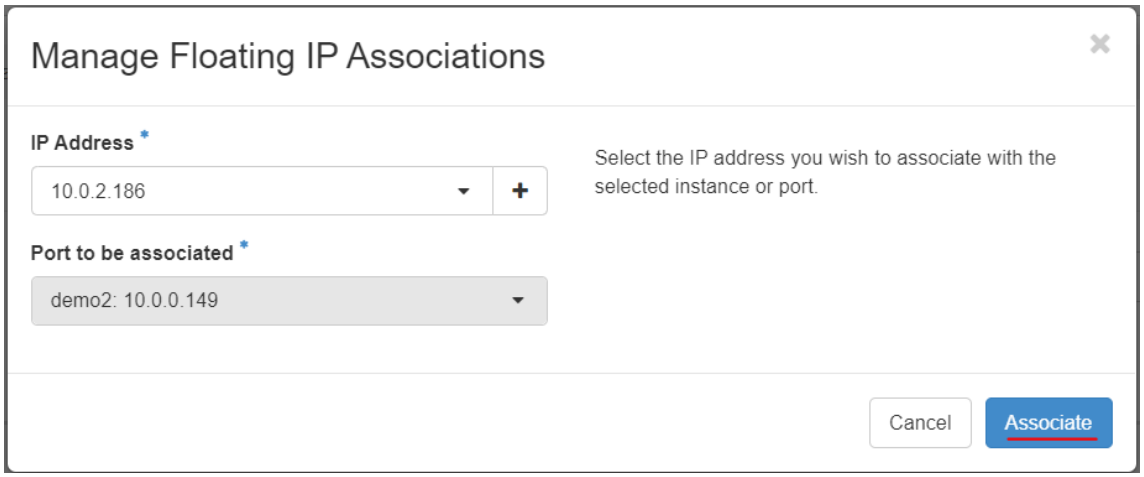
```
root@tutorial3-ctl:~# iptables -t nat -A POSTROUTING -o ens3 -j MASQUERADE
root@tutorial3-ctl:~# iptables -L -t nat
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE  all  --  anywhere             anywhere
```

Then we need to add floating ip to each instance.









Add the ip of the gateway of the external subnet to the interface veth1 with

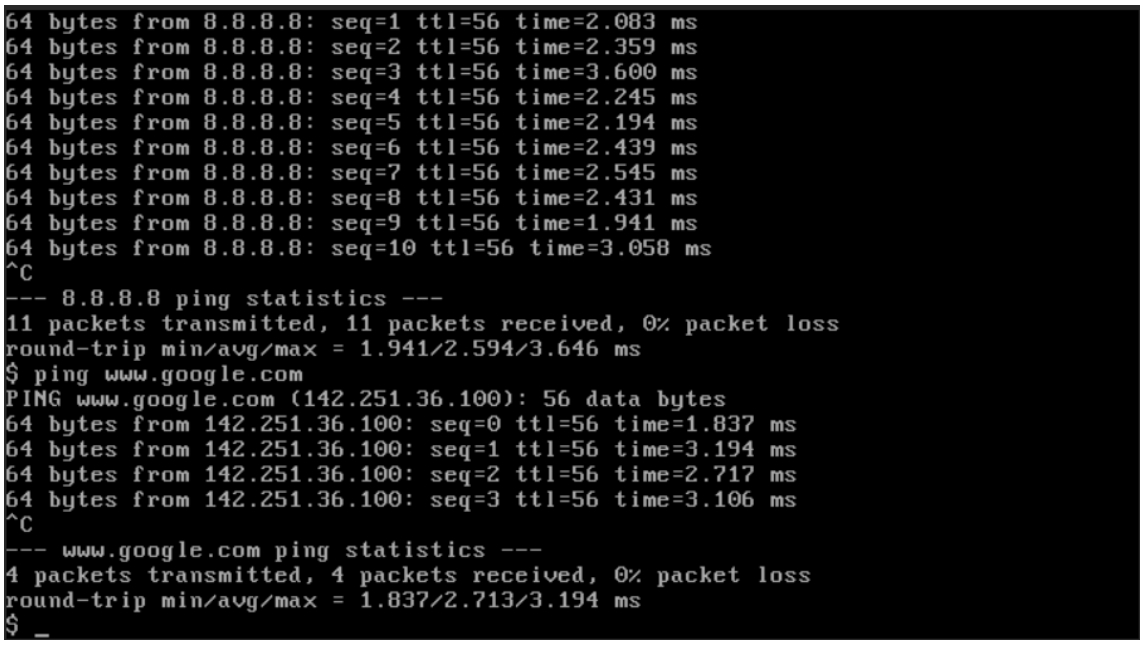
"ip addr add 10.0.2.1/24 dev veth1"

```
root@tuorial3-ctl:~# ip addr add 10.0.2.1/24 dev veth1
```

And finally, to activate the forwarding of packages in the control node let's execute "echo 1 > /proc/sys/net/ipv4/ip_forward"

```
root@tuorial3-ctl:~# echo 1 > /proc/sys/net/ipv4/ip_forward
```

And now if we ping from one of our instances to google for example (8.8.8.8) we should get a reply.


64 bytes from 8.8.8.8: seq=1 ttl=56 time=2.083 ms
64 bytes from 8.8.8.8: seq=2 ttl=56 time=2.359 ms
64 bytes from 8.8.8.8: seq=3 ttl=56 time=3.600 ms
64 bytes from 8.8.8.8: seq=4 ttl=56 time=2.245 ms
64 bytes from 8.8.8.8: seq=5 ttl=56 time=2.194 ms
64 bytes from 8.8.8.8: seq=6 ttl=56 time=2.439 ms
64 bytes from 8.8.8.8: seq=7 ttl=56 time=2.545 ms
64 bytes from 8.8.8.8: seq=8 ttl=56 time=2.431 ms
64 bytes from 8.8.8.8: seq=9 ttl=56 time=1.941 ms
64 bytes from 8.8.8.8: seq=10 ttl=56 time=3.058 ms
^C
--- 8.8.8.8 ping statistics ---
11 packets transmitted, 11 packets received, 0% packet loss
round-trip min/avg/max = 1.941/2.594/3.646 ms
$ ping www.google.com
PING www.google.com (142.251.36.100): 56 data bytes
64 bytes from 142.251.36.100: seq=0 ttl=56 time=1.837 ms
64 bytes from 142.251.36.100: seq=1 ttl=56 time=3.194 ms
64 bytes from 142.251.36.100: seq=2 ttl=56 time=2.717 ms
64 bytes from 142.251.36.100: seq=3 ttl=56 time=3.106 ms
^C
--- www.google.com ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 1.837/2.713/3.194 ms
$ _