

CSCI 5380

Network Virtualization and Orchestration

Lab 2

OpenStack: Multi-tenants

University of Colorado Boulder
Department of Computer Science
Network Engineering

Professor Levi Perigo, Ph.D.

PART 1: OpenStack and multitenancy

Objective 1 - OpenStack: Overview

1. Explain the following components of OpenStack -

- a. Nova - Nova is the compute service in OpenStack that creates, manages, and deletes virtual machines.
- b. Swift - Swift is the object storage service used to store and retrieve large amounts of unstructured data like images and backups.
- c. Cinder - Cinder provides block storage volumes that can be attached to virtual machines like virtual hard disks.
- d. Neutron - Neutron manages networking in OpenStack, including networks, subnets, routers, and IP addresses.
- e. Glance - Glance stores and provides virtual machine images used to launch instances.
- f. Keystone - Keystone is the identity service that handles authentication, authorization, users, projects, and roles.
- g. Horizon - Horizon is the web-based dashboard that allows users to manage OpenStack resources through a GUI.

2. What is the difference between Users and Roles?

A user represents a person or service, while a role defines what actions that user is allowed to perform in a project.

3. What is a hypervisor and which hypervisors are supported in OpenStack?

A hypervisor is software that allows multiple virtual machines to run on a single physical machine, and OpenStack supports hypervisors such as KVM, QEMU, Hyper-V, and VMware ESXi.

4. Explain the meaning of 'flavor' in OpenStack.

A flavor defines the hardware configuration of a virtual machine, such as CPU, RAM, and disk size.

5. Create a new network of 64 IP addresses in the Network tab and enable DHCP for 32 of the IPs using either the GUI or the CLI.

Name	Network Address	IP Version	Gateway IP	Actions
lab2	192.168.1.0/26	IPv4	192.168.1.1	<button>Edit Subnet</button>

6. Create a router that connects this new network with the existing “public network” using either the GUI or the CLI.

Name	Fixed IPs	Status	Type	Admin State	Actions
(dcb3d904-b7f5)	• 192.168.1.1	Active	Internal Interface	UP	<button>Delete Interface</button>
(ec0c98b3-a7b0)	• 172.24.4.200 • 2001:db8::141	Active	External Gateway	UP	<button>Delete Interface</button>

7. Start two instances with the Cirros image present that connects to the new network of 64 IPs using either the GUI or the CLI.

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
c1-2	-	192.168.1.52	m1.nano	-	Build	nova	Block Device Mapping	No State	0 minutes	<button>Associate Floating IP</button>
c1-1	-	192.168.1.61	m1.nano	-	Build	nova	Block Device Mapping	No State	0 minutes	<button>Associate Floating IP</button>

Objective 2 – Auto-scaling application using Python

1. Scenario:

You are working in a cloud firm that has a single instance of an application running on OpenStack cloud platform. The firm is planning to add a functionality to the single running instance of the application that can autoscale/replicate itself to multiple instances whenever the compute capacity (eg. CPU cycles or memory) reaches a pre-defined threshold. Since you are familiar with the Python programming and REST API, you are being assigned a following task:

- a. Write a simple Python application that can ssh into the available “cirros” instance that was created in the above objective and extract the CPU utilization information. [As an alternative, you may use ceilometer service for retrieving this telemetry data] - Attached
- b. If the CPU utilization or memory usage exceeds a threshold value, for example 20%, spin up additional instances of cirros. The creation of cirros instances should be triggered whenever the usage of CPU or memory exceeds a predefined threshold. Select CPU or memory usage to your interest to define your condition to trigger the creation of additional instances. In order to collect the utilization data, you’ll have to monitor its usage using appropriate commands.

I used the ssh based login and top command to parse the CPU usage and trigger the auto VM scale –

```
(NVO) atul@csc15380-vm2-atan8167:~/Labs/Lab 2$ ./scale.py
2026-01-26 22:06:35.651 | INFO  | __main__:<module>:34 - Initializing OpenStack connection
2026-01-26 22:06:35.668 | INFO  | __main__:autoscale_controller:158 - Starting autoscaling controller
2026-01-26 22:06:35.668 | INFO  | __main__:read_instance_registry:81 - Loaded 1 registered instances
2026-01-26 22:06:35.831 | INFO  | __main__:get_remote_cpu_percent:59 - CPU usage from 172.24.4.223: 10.00%
2026-01-26 22:06:35.831 | INFO  | __main__:autoscale_controller:180 - lab (172.24.4.223) CPU: 10.00%
2026-01-26 22:06:35.831 | INFO  | __main__:autoscale_controller:201 - System stable - no scaling action
2026-01-26 22:06:40.832 | INFO  | __main__:read_instance_registry:81 - Loaded 1 registered instances
2026-01-26 22:06:41.042 | INFO  | __main__:get_remote_cpu_percent:59 - CPU usage from 172.24.4.223: 100.00%
2026-01-26 22:06:41.042 | INFO  | __main__:autoscale_controller:180 - lab (172.24.4.223) CPU: 100.00%
2026-01-26 22:06:41.042 | WARNING | __main__:autoscale_controller:185 - CPU threshold exceeded on lab
2026-01-26 22:06:41.042 | WARNING | __main__:autoscale_controller:194 - Autoscale event triggered
2026-01-26 22:06:41.042 | WARNING | __main__:provision_instance:121 - Provisioning new OpenStack instance
2026-01-26 22:06:54.319 | INFO  | __main__:provision_instance:138 - Instance auto_vm_2 is ACTIVE
2026-01-26 22:06:55.398 | INFO  | __main__:provision_instance:147 - Assigned floating IP 172.24.4.220 to auto_vm_2
2026-01-26 22:06:55.398 | INFO  | __main__:append_instance_registry:114 - Registered new instance auto_vm_2 (172.24.4.220)
2026-01-26 22:07:00.399 | INFO  | __main__:read_instance_registry:81 - Loaded 2 registered instances
2026-01-26 22:07:00.562 | INFO  | __main__:get_remote_cpu_percent:59 - CPU usage from 172.24.4.223: 0.00%
2026-01-26 22:07:00.562 | INFO  | __main__:autoscale_controller:180 - lab (172.24.4.223) CPU: 0.00%
```

- c. The Python application can use Nova REST API to create additional “Cirros” instances whenever the above condition occurs.

Additional instances are created and Floating IP is also attached in those instances

automatically –

The screenshot shows the OpenStack Instances page. At the top, there are navigation links: Project / Compute / Instances. Below the header, there is a search bar with 'Instance ID = ▾' and a 'Filter' button. To the right of the search bar are buttons for 'Launch Instance' (blue), 'Delete Instances' (red), and 'More Actions ▾'. The main table displays three items:

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
auto_vm_1	cirros-0.6.3-x86_64-disk	192.168.1.43	m1.nano	-	Active	nova	None	Running	0 minutes	<button>Create Snapshot ▾</button>
auto_vm_1	cirros-0.6.3-x86_64-disk	192.168.1.12, 172.24.4.159	m1.nano	-	Active	nova	None	Running	9 minutes	<button>Create Snapshot ▾</button>
lab	cirros-0.6.3-x86_64-disk	192.168.1.62, 172.24.4.223	m1.nano	-	Active	nova	None	Running	5 hours, 27 minutes	<button>Create Snapshot ▾</button>

Below the table, it says 'Displaying 3 items'.

- d. The auto scaling of the instances should be handled considering following requirements:

```
(NVO) atul@csci5380-vm2-atan8167:~/Labs/Lab 2$ ./scale.py
2026-01-26 22:06:35.651 | INFO | __main__: <module>:34 - Initializing OpenStack connection
2026-01-26 22:06:35.668 | INFO | __main__: autoscale_controller:158 - Starting autoscaling controller
2026-01-26 22:06:35.668 | INFO | __main__: read_instance_registry:81 - Loaded 1 registered instances
2026-01-26 22:06:35.831 | INFO | __main__: get_remote_cpu_percent:59 - CPU usage from 172.24.4.223: 10.00%
2026-01-26 22:06:35.831 | INFO | __main__: autoscale_controller:180 - lab (172.24.4.223) CPU: 10.00%
2026-01-26 22:06:35.831 | INFO | __main__: autoscale_controller:201 - System stable - no scaling action
2026-01-26 22:06:40.832 | INFO | __main__: read_instance_registry:81 - Loaded 1 registered instances
2026-01-26 22:06:41.042 | INFO | __main__: get_remote_cpu_percent:59 - CPU usage from 172.24.4.223: 100.00%
2026-01-26 22:06:41.042 | INFO | __main__: autoscale_controller:180 - lab (172.24.4.223) CPU: 100.00%
2026-01-26 22:06:41.042 | WARNING | __main__: autoscale_controller:185 - CPU threshold exceeded on lab
2026-01-26 22:06:41.042 | WARNING | __main__: autoscale_controller:194 - Autoscale event triggered
2026-01-26 22:06:41.042 | WARNING | __main__: provision_instance:121 - Provisioning new OpenStack instance
2026-01-26 22:06:54.319 | INFO | __main__: provision_instance:138 - Instance auto_vm_2 is ACTIVE
2026-01-26 22:06:55.398 | INFO | __main__: provision_instance:147 - Assigned floating IP 172.24.4.220 to auto_vm_2
2026-01-26 22:06:55.398 | INFO | __main__: append_instance_registry:114 - Registered new instance auto_vm_2 (172.24.4.220)
2026-01-26 22:07:00.399 | INFO | __main__: read_instance_registry:81 - Loaded 2 registered instances
2026-01-26 22:07:00.562 | INFO | __main__: get_remote_cpu_percent:59 - CPU usage from 172.24.4.223: 0.00%
2026-01-26 22:07:00.562 | INFO | __main__: autoscale_controller:180 - lab (172.24.4.223) CPU: 0.00%
```

Max scaling size: 4 (this value denotes the maximum number of instances that should be spun)

Increment size: 1 (this value denotes the number of instances that should be spun whenever CPU utilization exceeds threshold)

Evaluation period: 40 (this value denotes the time period in seconds for monitoring CPU usage)

2. You can use the [Linux stress tool](#) to raise the CPU utilization of an instance above the threshold.

I ran an infinite loop on the device which caused the CPU usage to go to the 100 percentage which triggered this autoscaling.

```
(NVO) atul@csc15380-vm2-atan8167:~/Labs/Lab 2$ cat instance_access.csv
name,ip,username,password
lab,172.24.4.223,cirros,gocubsgo

auto_vm_2,172.24.4.220,cirros,gocubsgo
(NVO) atul@csc15380-vm2-atan8167:~/Labs/Lab 2$
```

Objective 3: Multi-tenants

- In this objective, you are introduced to the function of basic tenant implementation and management with OpenStack.
- The goal is to create two virtual networks and three VMs as is shown in Figure 1.

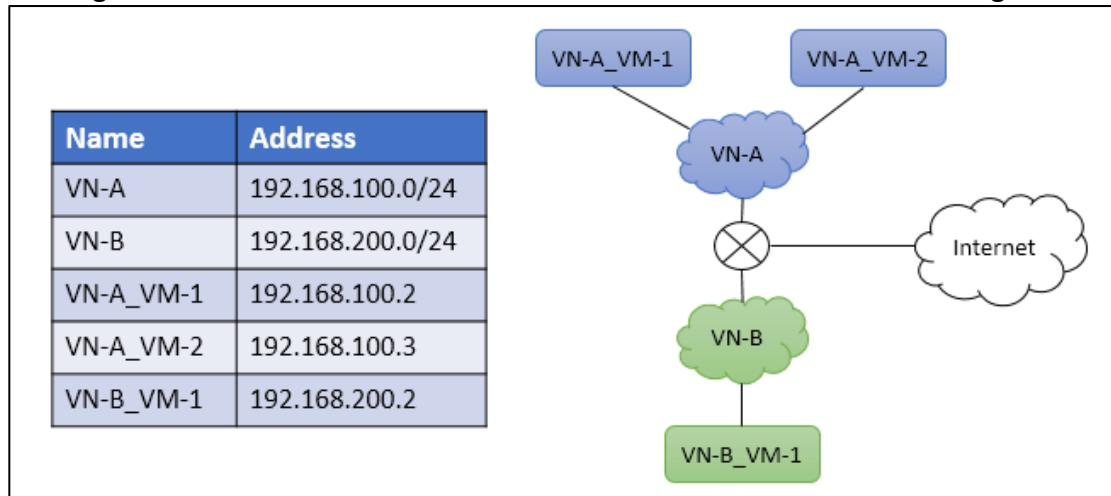


Figure 1. Final goal of Objective 3

Section 1: Creating project, user, flavor and image

- Within OpenStack UI Identity tag, create a project called lab2. Then create a user called lab2_admin and attach it to the project lab2.

Project Lab2 -

```
atul@csci5380-vm2-atan8167:~$ openstack project create lab2 --description "Lab2 project for NVO"
+-----+-----+
| Field      | Value          |
+-----+-----+
| description | Lab2 project for NVO |
| domain_id   | default         |
| enabled      | True            |
| id           | 0fe242c8501642eaa188950f99d55166 |
| is_domain    | False           |
| name         | lab2            |
| options      | {}              |
| parent_id    | default         |
| tags         | []              |
+-----+-----+
```

User Creation –

```
atul@csci5380-vm2-atan8167:~$ openstack user create lab2_admin --project lab2 --password atul
+-----+-----+
| Field      | Value          |
+-----+-----+
| default_project_id | 0fe242c8501642eaa188950f99d55166 |
| domain_id   | default         |
| email        | None            |
| enabled      | True            |
| id           | 21876a8232b1414e9acb62a8cac8b809 |
| name         | lab2_admin      |
| description   | None            |
| password_expires_at | None            |
| options      | {}              |
+-----+-----+
```

2. Within OpenStack UI Admin tag, create a VM Flavor called **ngn.tiny** with the following setting (or the setting that works for your VM image):

vCPU	= 1
RAM	= 128MB
Root Disk	= 1GB
Ephemeral Disk	= 1GB
Swap Disk	= 1GB

```
atul@csci5380-vm2-atan8167:~$ openstack flavor create ngn.tiny \
--vcpus 1 \
--ram 128 \
--disk 1 \
--ephemeral 1 \
--swap 1
+-----+-----+
| Field | Value |
+-----+-----+
| OS-FLV-DISABLED:disabled | False
| OS-FLV-EXT-DATA:ephemeral | 1
| description | None
| disk | 1
| id | ab91a233-9eae-4487-9564-4a9eae6259dc
| name | ngn.tiny
| os-flavor-access:is_public | True
| properties |
| ram | 128
| rxtx_factor | 1.0
| swap | 1
| vcpus | 1
+-----+-----+
```

Flavor List -

```
atul@csci5380-vm2-atan8167:~$ openstack flavor list
+-----+-----+-----+-----+-----+-----+-----+
| ID | Name | RAM | Disk | Ephemeral | VCPUs | Is Public |
+-----+-----+-----+-----+-----+-----+-----+
| 1 | m1.tiny | 512 | 1 | 0 | 1 | True |
| 2 | m1.small | 2048 | 20 | 0 | 1 | True |
| 3 | m1.medium | 4096 | 40 | 0 | 2 | True |
| 4 | m1.large | 8192 | 80 | 0 | 4 | True |
| 42 | m1.nano | 192 | 1 | 0 | 1 | True |
| 5 | m1.xlarge | 16384 | 160 | 0 | 8 | True |
| 84 | m1.micro | 256 | 1 | 0 | 1 | True |
| ab91a233-9eae-4487-9564-4a9eae6259dc | ngn.tiny | 128 | 1 | 1 | 1 | True |
| c1 | cirros256 | 256 | 1 | 0 | 1 | True |
| d1 | ds512M | 512 | 5 | 0 | 1 | True |
| d2 | ds1G | 1024 | 10 | 0 | 1 | True |
| d3 | ds2G | 2048 | 10 | 0 | 2 | True |
| d4 | ds4G | 4096 | 20 | 0 | 4 | True |
+-----+-----+-----+-----+-----+-----+-----+
```

3. Within OpenStack UI Admin tag, upload a VM image into OpenStack. You can use

this URL: <http://tinycorelinux.net/7.x/x86/release/Core-current.iso> or

<https://docs.openstack.org/image-guide/obtain-images.html>.

Remember to make it public.

Images

Click here for filters or full text search. ✖ + Create Image Delete Images

Displaying 2 items

<input type="checkbox"/>	Owner	Name ^	Type	Status	Visibility	Protected	Disk Format	Size	
<input type="checkbox"/>	> admin	cirros-0.6.3-x86_64-disk	Image	Active	Public	No	QCOW2	20.69 MB	Launch ▾
<input type="checkbox"/>	> admin	core	Image	Active	Public	No	ISO	10.60 MB	Launch ▾

Displaying 2 items

- Before proceeding, logout and login with your newly created user lab2_admin.

Section 2: Setup Virtual Networks

- Login back into OpenStack UI, within the Project tag, create a new Network called VN-A with network address 192.168.100.0/24. – attached
- Repeat the above steps to create a second network VN-B with network address 192.168.200.0/24.

	Name	Subnets Associated	Shared	External	Status	Admin State	Availability Zones	Actions
<input type="checkbox"/>	shared	shared-subnet 192.168.233.0/24	Yes	No	Active	UP	-	Edit Network ▾
<input type="checkbox"/>	VN-A	VN-A 192.168.100.0/24	No	No	Active	UP	-	Edit Network ▾
<input type="checkbox"/>	VN-B	VN-B 192.168.200.0/24	No	No	Active	UP	-	Edit Network ▾

Section 3: Launch VM instances

Launch the following VMs using the flavor and image created in Section 1.

- Launch VN-A_VM-1 and VN-A_VM-2 into virtual network VN-A.
- Launch VN-B_VM-1 into virtual network VN-B.

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age	Actions
VN-A_VM-1	core	192.168.100.216	gn.tiny	-	Active	nova	None	Running	1 minute	<button>Create Snapshot</button>
VN-A_VM-2	core	192.168.100.120	gn.tiny	-	Active	nova	None	Running	0 minutes	<button>Create Snapshot</button>
VN-B_VM-1	-	192.168.200.219	gn.tiny	-	Active	nova	None	Running	0 minutes	<button>Create Snapshot</button>

Section 4: Ping testing

1. Use the console within OpenStack UI to test if VMs in VN-A can ping each other, while the VM in VN-B cannot reach VMs in VN-A.

VN-A_VM-1 to VN-A_VM-2-

```

tc@box:~$ ping 192.168.100.120
PING 192.168.100.120 (192.168.100.120): 56 data bytes
64 bytes from 192.168.100.120: seq=0 ttl=64 time=1.479 ms
64 bytes from 192.168.100.120: seq=1 ttl=64 time=0.567 ms
64 bytes from 192.168.100.120: seq=2 ttl=64 time=0.448 ms
^C
--- 192.168.100.120 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.448/0.831/1.479 ms

```

VN-B_VM1 to VM's in VN-A-

```

( ' >')
( /) TC (\ Core is distributed with ABSOLUTELY NO WARRANTY.
( \---_--\ www.tinycorelinux.net

tc@box:~$ ping 192.168.100.120
PING 192.168.100.120 (192.168.100.120): 56 data bytes
^C
--- 192.168.100.120 ping statistics ---
6 packets transmitted, 0 packets received, 100% packet loss
tc@box:~$ ping 192.168.100.216
PING 192.168.100.216 (192.168.100.216): 56 data bytes
^C
--- 192.168.100.216 ping statistics ---
3 packets transmitted, 0 packets received, 100% packet loss
tc@box:~$ 

```

2. Assign floating IPs to the VM's both in VN-A and VN-B, and test connectivity to the Internet.

Router connected to Both the network and with internet –

Project / Network / Floating IPs

Floating IPs

Floating IP Address = Filter % Allocate IP To Project

Displaying 3 items

<input type="checkbox"/>	IP Address	Description	DNS Name	DNS Domain	Mapped Fixed IP Address	Pool	Status	Actions
<input type="checkbox"/>	172.24.4.126	VN-B_VM-A			VN-B_VM-1 192.168.200.219	public	Active	<input type="button" value="Disassociate"/>
<input type="checkbox"/>	172.24.4.122	VN-A_VM-B			VN-A_VM-2 192.168.100.120	public	Active	<input type="button" value="Disassociate"/>
<input type="checkbox"/>	172.24.4.35	VN-A_VM-A			VN_A_VM-1 192.168.100.216	public	Active	<input type="button" value="Disassociate"/>

Displaying 3 items

VN-B_VM-1 –Ping Success –

Project / Compute / Instances / VN-B_VM-1

VN-B_VM-1

Create Snapshot

Overview Interfaces Log Action Log

Instance Console

If console is not responding to keyboard input: click the grey status bar below. [Click here to show only console](#)
To exit the fullscreen mode, click the browser's back button.

Connected to QEMU (Instance-00000003)

```
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: seq=0 ttl=111 time=5.151 ms
64 bytes from 8.8.8.8: seq=1 ttl=111 time=3.033 ms
64 bytes from 8.8.8.8: seq=2 ttl=111 time=4.700 ms
64 bytes from 8.8.8.8: seq=3 ttl=111 time=3.909 ms
```

Similarly, Other VM's in VN-A also can ping internet.

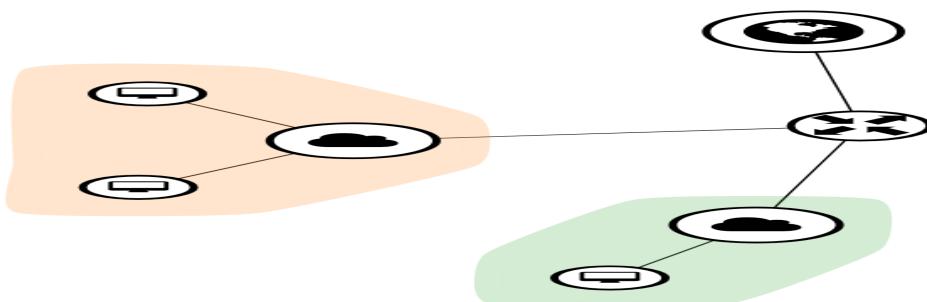
Objective 4 – Network policies in OpenStack

Summary:

In this objective, you will manage network policies within OpenStack.

Section 1: VM and Virtual Network Setup

First you will need to create the configuration as is shown in Figure 1 above.



Section 2: Achieve Inter-VN Communication

The default policy allows only intra-VN communication.

- Figure out how to ping between VM's in VN-A and VM's in VN-B.

I created a new security group called as lab which is mapped to all the instances – This allowed me to allow the ping as well as any interesting traffic of my choice. This way I was able to ping from VN-B to VN-A as attached below.

ID	IP Protocol	Ethertype	IP Range	Port Range	Direction	Remote Security Group	Remote Address Group
6a93a027-0cb9-4efe-bd6b-98670971eca0	None	IPv4	0.0.0.0/0		egress	None	None
adf7ff9d-5bd0-4809-8205-cbcbe212afae	icmp	IPv4	0.0.0.0/0		ingress	None	None
dc53140e-826c-4178-9a86-668c32b32d84	None	IPv6	::/0		egress	None	None
e1204296-2d2e-403e-9f7a-8cbe3891233c	tcp	IPv4	0.0.0.0/0	22:22	ingress	None	None

VN-B_VM-1

Overview Interfaces Log **Console** Action Log

Instance Console

If console is not responding to keyboard input: click the grey status bar below. [Click here to show only console](#)
To exit the fullscreen mode, click the browser's back button.

Connected to QEMU (instance-00000003)

```
PING 192.168.100.120 (192.168.100.120): 56 data bytes
64 bytes from 192.168.100.120: seq=0 ttl=63 time=4.446 ms
64 bytes from 192.168.100.120: seq=1 ttl=63 time=0.989 ms
64 bytes from 192.168.100.120: seq=2 ttl=63 time=0.486 ms
^C
--- 192.168.100.120 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.486/1.973/4.446 ms
tcebox:~$ ping 192.168.100.216
PING 192.168.100.216 (192.168.100.216): 56 data bytes
64 bytes from 192.168.100.216: seq=0 ttl=63 time=2.548 ms
64 bytes from 192.168.100.216: seq=1 ttl=63 time=1.121 ms
^C
```

- Figure out how to ping from the VM's out to the Internet.

Custom SG allowed to ensure rule changes by limiting the blast radius from Network.

Section 3: Network Policy Management

Now manage the network policy inside OpenStack, such that:

- VN-A_VM-1 can ping VN-B_VM-1, but VN-A_VM-2 cannot ping VN-B_VM-1.
- VN-B_VM-1 can go out to the Internet, but VN-A_VM-1 and VN-A_VM-2 cannot.

I created Security group to ensure only VM-1 can ping the VM1 in VN-B and same group

restricted access to internet for VN-A also created another group for Vn-b where it restricted ingress rule from VN-A_VM-2 to ensure no ICMP reachability.

vn-a-no-internet for VN-A VMs

ID	IP Protocol	Ethertype	IP Range	Port Range	Direction	Remote Security Group	Remote Address Group
7bfffcea-58a8-40e9-a316-4fc91fb57df1	icmp	IPv4	192.168.200.219/32		egress	None	None
f10193a7-e94e-4f57-a6ce-62b34cc3e0dd	None	IPv4	0.0.0.0/0		ingress	None	None
fec950f9-540c-4afa-975e-8d697b9d920c	None	IPv4	192.168.100.1/24		egress	None	None

vn-b group for VN-B_hosts –

ID	IP Protocol	Ethertype	IP Range	Port Range	Direction	Remote Security Group	Remote Address Group
b16ae9e7-28f4-4b8b-9790-6fa8396f0b92	icmp	IPv4	192.168.100.216/32		ingress	None	None
dbb1b898-31f9-4671-a4c3-69bb89368527	None	IPv4	0.0.0.0/0		egress	None	None

Project / Network / Security Groups

Security Groups

Displaying 4 items					
<input type="checkbox"/>	Name	Security Group ID	Description	Shared	Actions
<input type="checkbox"/>	default	1a588586-aa6e-45ea-9262-1bbcceb175b8e	Default security group	-	<button>Manage Rules</button>
<input type="checkbox"/>	lab	65cbef0c-e31b-43f8-b836-709d51aa8060		-	<button>Manage Rules</button>
<input type="checkbox"/>	vn-a-no-internet	b6bdd1a3-5b33-46c9-8fec-3aff6faf9fd6	vn-a-no-internet	-	<button>Manage Rules</button>
<input type="checkbox"/>	vn-b-policy	adb45fd2-f480-4a7d-adaa-bd715ab8585c	vn-b-policy	-	<button>Manage Rules</button>

Deliverable (100 points):

1. Individually complete all tasks in the lab - **Completed**
2. Create a capstone group GitHub tutorial document about this lab
 - a. Create a small guide explaining/demonstrating how to achieve each objective from the lab
 - i. The individual objective guides should be divided between the team members evenly to be completed in GitHub
3. Submit to Canvas:
 - a. Individual Python Auto-scaling application - Attached
 - b. Document (bulleted list) of what each member contributed to GitHub tutorial document – Not applicable
 - c. Include a link to the GitHub page

<https://github.com/atulanand25/NVO/>