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 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:

* 1. 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]
  2. 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.
  3. The Python application can use Nova REST API to create additional “Cirros” instances whenever the above condition occurs.
  4. The auto scaling of the instances should be handled considering following requirements:

**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)

1. You can use the [Linux stress tool](https://www.tecmint.com/linux-cpu-load-stress-test-with-stress-ng-tool/) to raise the CPU utilization of an instance above the threshold.

What my code does

* + Create a ssh session with newly created instance
  + Check the current CPU utilization
  + Create a stress to increase CPU utilization
  + If CPU utilization is more than 20% then create new instances max limit 4, one at a time
  + If we stops the code in between with keyboard interrupt it will clean all the instances that has been created(cleanup Function)

# 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.

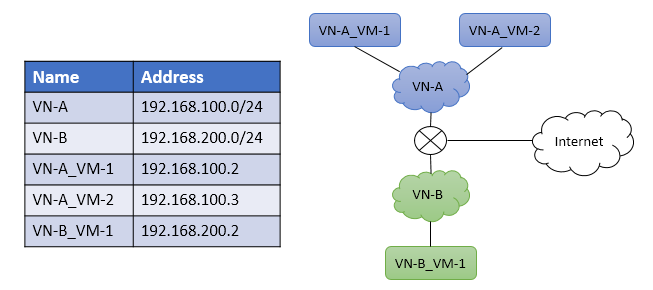


Figure 1. Final goal of Objective 3

# Section 1: Creating project, user, flavor and image

1. Within OpenStack UI Identity tag, create a project called lab2. Then create a user called lab2\_admin and attach it to the project lab2.  
   Step 1 – create a project with name lab2   
   Project information section add name and click on **Save**

A screenshot of a project

Description automatically generated

Now create a New User Under the project  
Create a password and add the project name that we just created above

A screenshot of a computer

Description automatically generated

1. 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

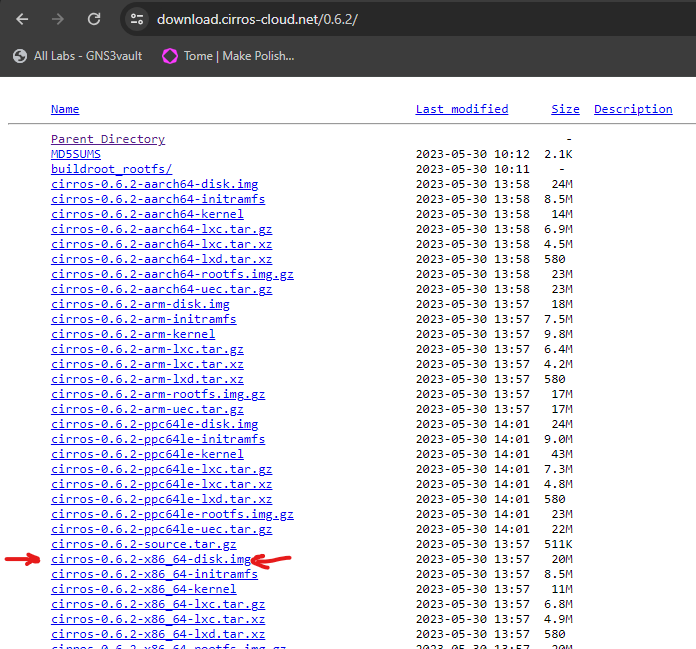
Swap Disk = 1GB

To create specifications for the VM instances we're planning to deploy, we need to define a flavor based on the provided specifications.

A screenshot of a computer

Description automatically generated

1. 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>
2. Iu8564rte54 or <https://docs.openstack.org/image-guide/obtain-images.html>.

Remember to make it public.  
Download below image to upload to create an instance  


* Now that we create flavor we need to create an image with uploading th file that we downloaded above.
* Select image type qcow

A screenshot of a computer

Description automatically generated

1. Before proceeding, logout and login with your newly created user lab2\_admin.

A screenshot of a computer

Description automatically generated

# Section 2: Setup Virtual Networks

1. Login back into OpenStack UI, within the Project tag, create a new Network called VN-A with network address 192.168.100.0/24.

Create two different networks for VM’s

Network on VN-A and subnet name we have to mention

Then enter the network address with subnet mask.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Reserve few ip’s for the DHCP pool and mention that range under Allocation pool separated by comma

A screenshot of a computer

Description automatically generated

1. Repeat the above steps to create a second network VN-B with network address 192.168.200.0/24.

Similarly we will do it for the VN-B

A screenshot of a computer

Description automatically generated

# Section 3: Launch VM instances

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

1. Launch VN-A\_VM-1 and VN-A\_VM-2 into virtual network VN-A.

Now that we have both the networks ready we will create VM’s under those 2 networks

A screenshot of a computer

Description automatically generated A screenshot of a computer

Description automatically generated

1. Launch VN-B\_VM-1 into virtual network VN-B.

A screenshot of a computer

Description automatically generated

# 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.

Ping from VN-A\_VM-1 to VN-A\_VM-2

A screenshot of a computer program

Description automatically generated

Ping from VN-A\_VM-2 to VN-A\_VM-1

A screenshot of a computer program

Description automatically generated

Ping from VN-B\_VM-1 to VN-A\_VM-1 and VN-A\_VM-2

A screenshot of a computer

Description automatically generated

1. Assign floating IP’s to the VM’s both in VN-A and VN-B, and test connectivity to the Internet.

A screenshot of a computer

Description automatically generated

Ping to Internet from VN-A\_VM-1

A screenshot of a computer program

Description automatically generated

Ping to Internet from VN-A\_VM-2

A screenshot of a computer program

Description automatically generated

Ping to Internet from VN-B\_VM-1

A screenshot of a computer program

Description automatically generated

# Deliverable (100 points):

## Individually complete all tasks in the lab

## Create a capstone group GitHub tutorial document about this lab

## Create a small guide explaining/demonstrating how to achieve each objective from the lab

## The individual objective guides should be divided between the team members evenly to be completed in GitHub

## Submit to Canvas:

## Individual Python Auto-scaling application

## Document (bulleted list) of what each member contributed to GitHub tutorial document

## Include a link to the GitHub page