ReScuE Documentation:

Creation, Management, and Operation

Developed By: Hisham Kanaan, Dong Han, Khalid Amen, Brendan Makar, Melissa Nichols, Nathan Torrez, Tianhuan Tu, Shuoying Zhao

Faculty Adviser: Anyi Liu & Huirong Fu

Department of Computer Science and Engineering

Oakland University

Rochester, MI

Table of Contents

Table of Figures

Introduction

In today's world, there is a vital need for the automated execution and benchmarking of experiments. The ability to consistently reproduce experiment environments and their results is crucial. Research in the field of engineering and computer science is no different to this premise. Empirical research based on detailed benchmarking results has been essential in driving value to the research question in terms of proving or negating the proposal. One particular topic of research has been the impact of the virtualization technology on changing the way everyone conducts business. It has been clear that the current market trend is headed towards a virtualized environment across the different layers of the technology. Since its original inception with the virtualization of end host instances and specialized servers, virtualization technology has come a long way in its maturity to be adopted in the networking field. The concept of an overlay network has been developed, with virtualized routers, switches, firewalls as well as a host of other cross layer virtualized security appliances. This has helped organizations cut down on the operation costs, increase their robustness and flexibility, decrease the mean time to recover from failures and build scalable topologies on the fly.

With the above being said, many initiatives were proposed as a collaboration between different organizations to introduce the virtualization technologies to the public. This would help drive forward the exploitation of the advantages of virtualization at a minimal cost to the users. In this document, we will discuss our efforts in building, and maintaining infrastructure instances built on OpenStack and Cloudlab. Cloudlab is the joint initiative between three universities: Wisconsin, Utah, and Clemson, and it provides the users a virtualized "slice" of their infrastructure for users to be able to experiment on. For more information on OpenStack and Cloudlab, please visit references [1], [2], [3] and [4].

Problem Statement

In various fields of interest, such as research and education, dynamic and accurate recreation of research results is essential. For example, in the field of education, conducting studies on a topic can be more involved within participants in a "Create-once-produce-many" implementation. In this case, the primary collaborators in the topic are able to produce a single instance on which they build their material. However, they need to be able to pass copies of this material to students for local deployment and a direct hands-on experience. In the field of research, being able to swiftly backup and restore your topologies is vital. This can help in a more detailed benchmarking as well as the ability to provide manageable copies of the researcher's environment. This can help others recreate, validate and further advance the topics of study based on the previous efforts.

Proposition

In this effort, we build on the virtualization capabilities that are provided by Cloudlab and generally through OpenStack. We aim at creating a framework through which users can manage their virtualized environments that are deployed on OpenStack instances in an automated and script-able manner. We also propose the automated profiling and restoration of entire instances with minimal user overhead.

This document will proceed with sections on creating OpenStack instances deployed on Cloudlab resources, and the general time-consuming manner through which networks are created. Then, we offer our own deployed scripts that provide the users with capabilities of automating various functions as well as easily creating profiles of their environments, and later we show the ease with which a profile can be recreated. We also include information on how to create a specific attack scenario experiment, and how to use the web client created for use with this experiment. The illustrating screenshots and recommendations included in this document are based on primary experience in the operation of OpenStack instances. Please consider entire framework we proposed as a continued work in progress as we have provided now the necessary and sufficient conditions to profile and reproduce environments and results in OpenStack and Cloudlab.

Individual Contributions

In this project, Hisham has tackled working with OpenStack and CloudLab. He studied the OpenStack software definitions and the SDK functionalities provided by OpenStack. Hisham was also responsible for finding solutions around the problems faced, as well as creating and validating the Python scripts that are produced herein.

Khalid tried out an introspection library, LibVMI that would help security researchers monitor their compute nodes and perform low level introspection to discover any compromise.

Brendan was operating with both focuses, and oversaw the website matters. In addition, Brendan was a co-author in this document, and a reviewer for its contents.

Melissa, a part of the second group who worked on this project, streamlined and added functionality to the previously created scripts, created images for use with attack scenario experiments, and wrote experiment documentation including the additions contributed by the second group to this document.

Nathan, also a part of the second group who worked on this project, streamlined and added functionality to the previously created scripts, added Snort rules to detect exploits with our attack scenario experiment, and helped to optimize the web client.

Tienhuan, also a part of the second group who worked on this project, developed a web client for usage with attack scenario experiments.

Part I: Creating an Account on CloudLab

To use CloudLab, you must create an account if you do not already have one. If you already have an account, you can skip to **Part II: Creating an OpenStack Instance on CloudLab**.

At www.cloudlab.us, click the "Request an Account" button to go to the account creation page.

Figure 1: Click "Request an Account" to create a new account

Fill out the *"Personal Information"* form (the SSH public key is optional at this point). You may either add your account to an existing project or start a new one. If you wish to work on a preexisting project, click the "*Join Existing Project"* button under the *"Project Information"* heading, then simply put its name in the box. You will be able to access the project once its administrator accepts your request to join. Otherwise, choose *"Start New Project".* Input some simple information, such as the class you are using this experiment for, and an explanation of the experiment (e.g., *"The purpose of this project is to use CloudLab to create a virtual environment for an educational penetration testing experiment."*).

You will receive an e-mail when your account is approved. Then, you will be able to login to CloudLab.

Part II: Creating an OpenStack Instance on CloudLab

After logging in to CloudLab, click on the tab labeled "*Experiments*" and choose "*Start Experiment*" from the drop-down menu.

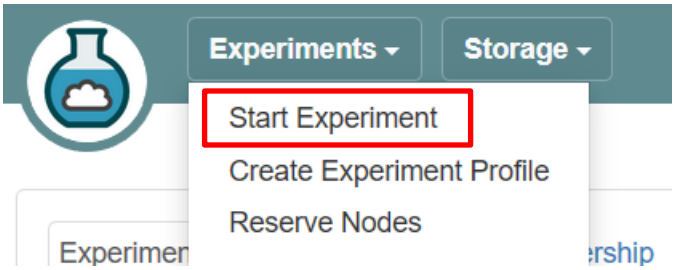

Figure 2: The Experiments drop-down menu

From there, you will be prompted to select a profile to start your experiment with. This changes the configuration that your experiment will use. This guide will primarily use the default OpenStack profile setting.

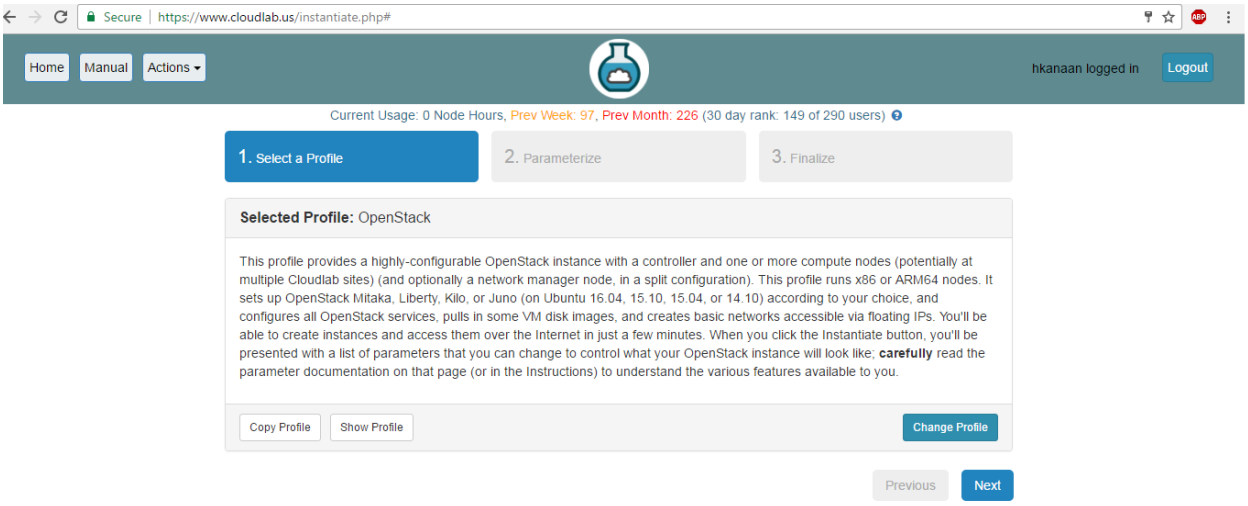

Figure 3: CloudLab initial OpenStack profile creation

After choosing "*next*", you will be prompted to configure the parameters of the OpenStack instance. There are a few important settings to consider:

- The **number of compute nodes at Site 1** refers to the number of physical PCs that will be obtained from the CloudLab clusters. The more resources (RAM, disk space, etc.) that you will need for your experiment, the more nodes you will need to assign to it.

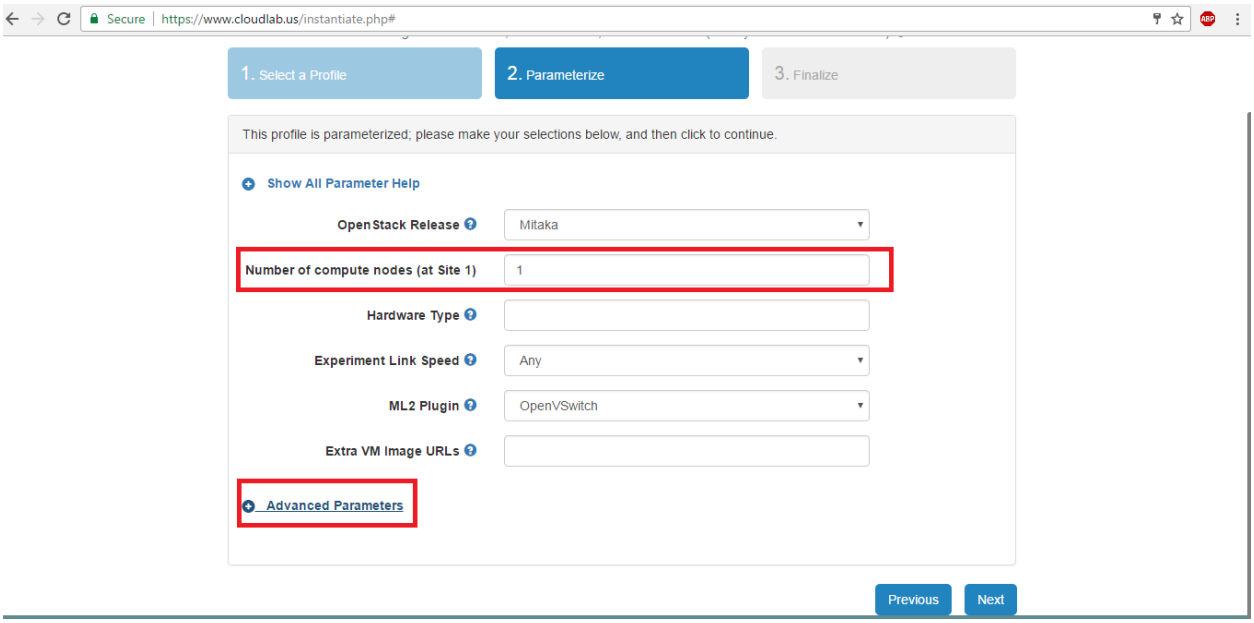

Figure 4: OpenStack parameters, with settings of interest highlighted

- The **number of public IP addresses** can be found under the *"Advanced Paramenters"* menu. You should always have more than two IP addresses assigned to your experiment, because two public IP addresses are assigned to the default networks at the beginning of the experiment. The importance of available public IP addresses will be covered more in **Part VI: Access the Created VM**.

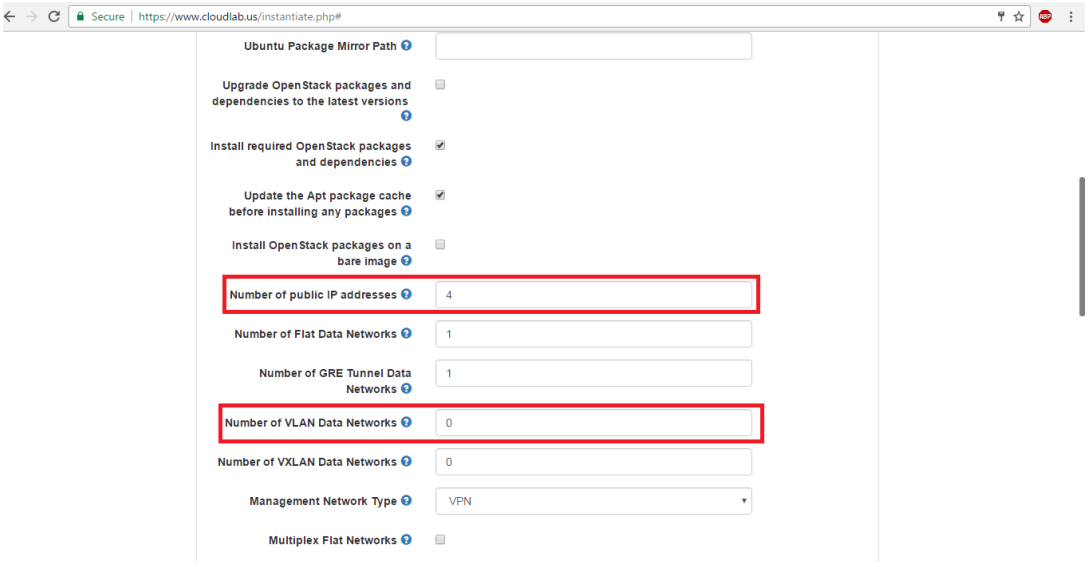

Figure 5: OpenStack parameters showing how to change number of IP addresses and VLANs

- If your experiment will require you to group together networks like a LAN would in a traditional network setup, you can also add **VLANs (virtual LANs)** from the *"Advanced Paramenters"* menu.

The next page is "*Finalize"*, which allows you to choose the cluster you wish to deploy your experiment on. The color of the dot next to the testbed name indicates how many nodes are free in that cluster: a green dot indicates that the testbed has a large amount of space available, while a yellow or red dot appears next to clusters with few resources open. Hovering your cursor over the dot will give a more detailed breakdown of the server availability for the cluster.

You may also name your experiment for easy identification. If you leave the "*Name"* box blank, a default name will be automatically generated based on your account name.

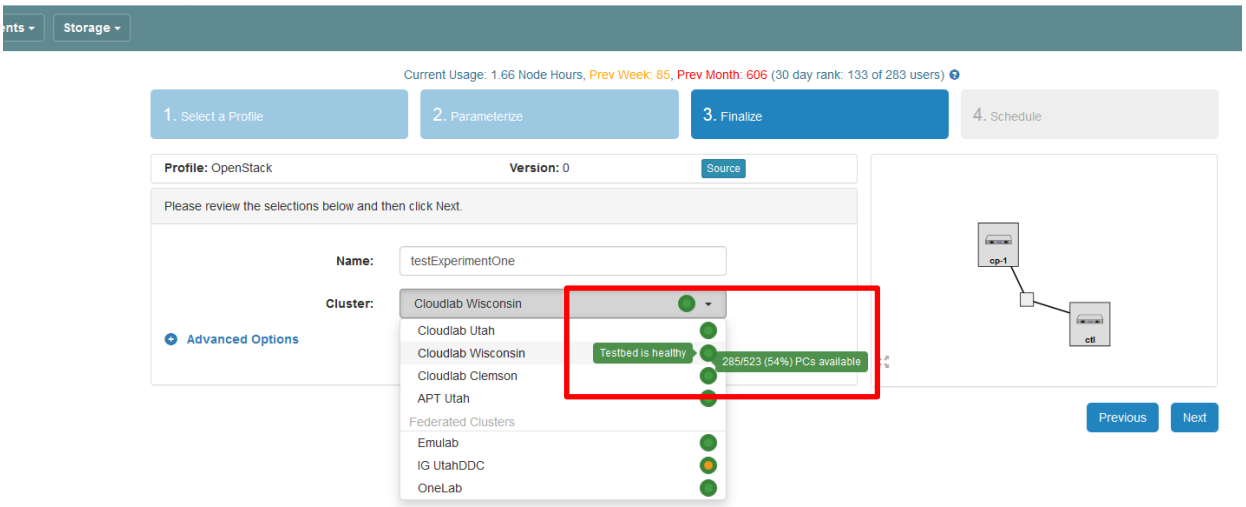

Figure 6: Choosing the cluster desired for implementation

The final page is the "*Schedule*" page, which allows you to change when the experiment will start and how long it will last. By default, the experiment will begin booting immediately and run for 16 hours. On this page, 16 hours is the maximum duration you can give your experiment, but if you need more time, it is possible to extend your time after creating your experiment. When you have scheduled your experiment, click "*Finish*".

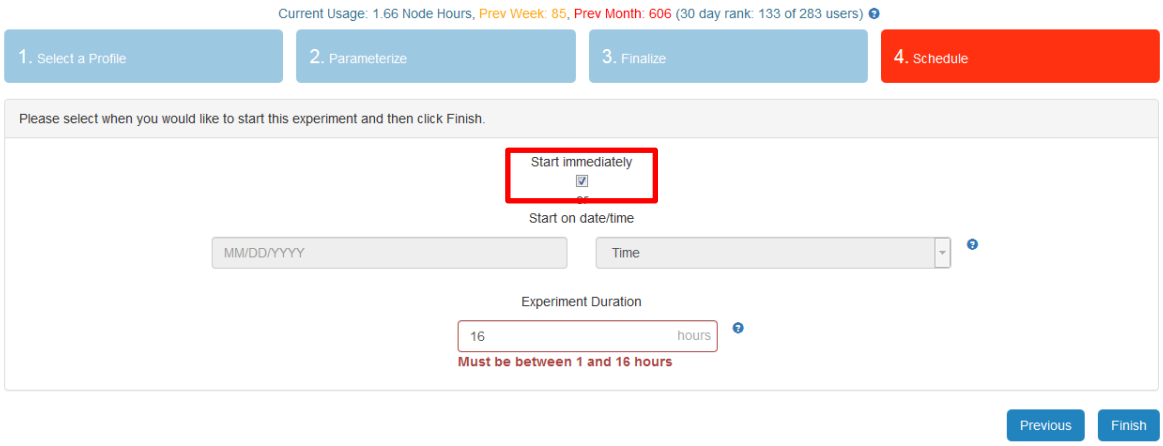

Figure 7: Change the start time and duration of your experiment

CloudLab will need about 10 to 15 minutes to completely boot up a new OpenStack experiment. The status "**booted**" means that not all services are yet activated. When the status changes to "**ready**", the experiment is fully prepared for use. You can extend the experiment's duration beyond your initial duration by clicking the "*Extend*" button. If you wish to extend your experiment past seven days, a justification for the extension must be supplied.

You can find information about the physical machines running your experiment at the bottom of this page. Clicking on the "*List View"* tab provides you with commands to use with an SSH client to log in to the nodes you are using. You can also control these computers by clicking the gear icon to open the "*Actions*" menu and create a console for the selected node in the browser.

The "*ctl"* node runs the OpenStack services and controls your experiment, while the additional nodes are used to provide computational power.

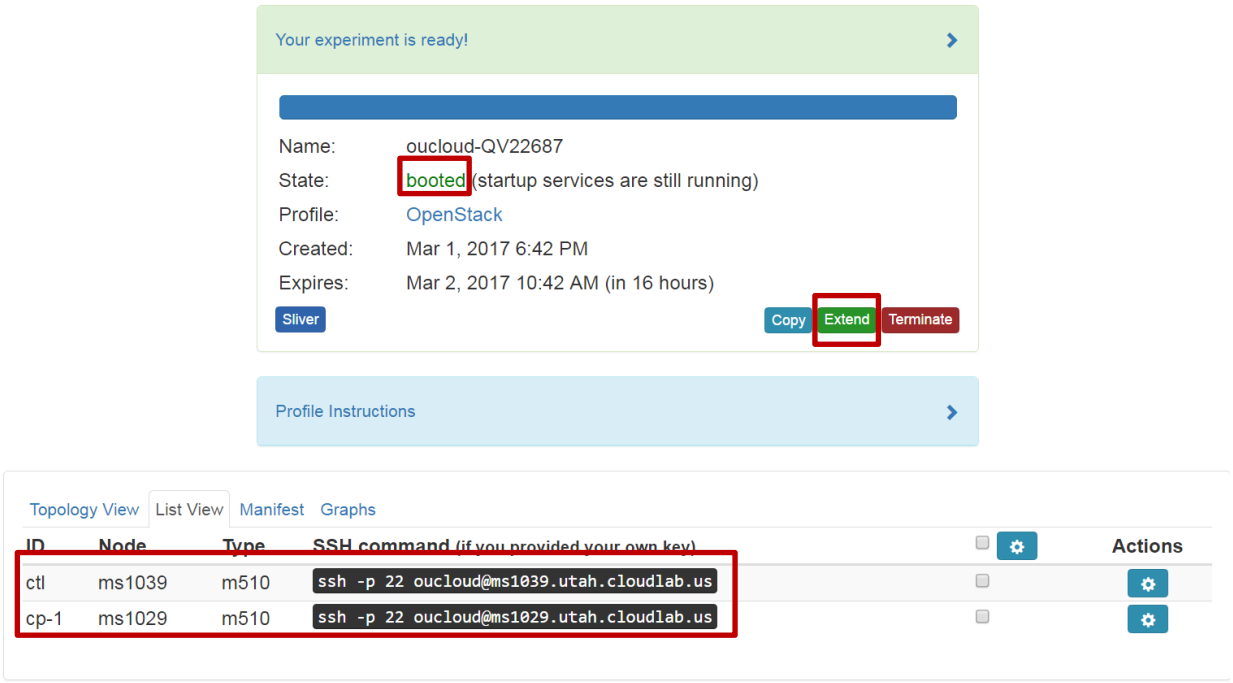

Figure 8: Cloudlab experiment homepage

Part II: Logging in to OpenStack and Profile Instructions Overview.

Once the OpenStack instance is created, the page will have a clickable label, namely "*Profile Instructions*". Clicking the label will display some instructions for the user containing the following information:

a) The default username of the OpenStack instance: **admin**

b) The domain of the OpenStack instance login: **default**

c) The default username for any virtual machine (VM) in your experiment: **ubuntu**

d) A randomly generated password used to log into the OpenStack instance as well as any VMs in your experiment. For instance, "*8e1cbe915ddf*" is the password in the example below.

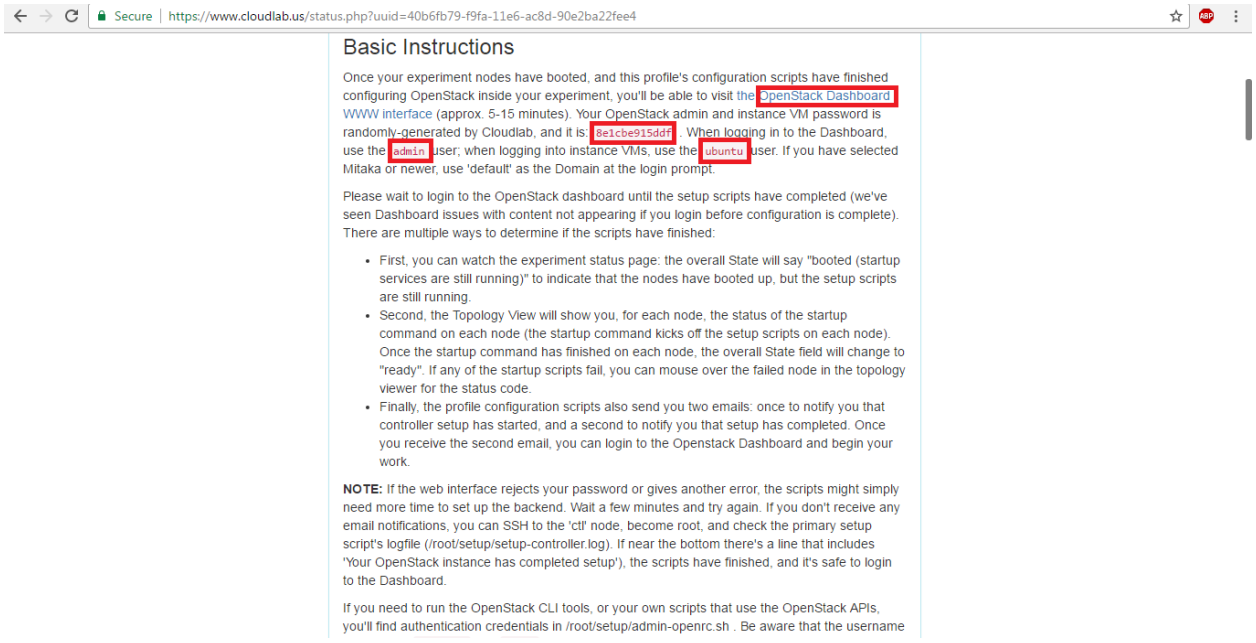

Figure 9: OpenStack instance details in CloudLab

To log in to the OpenStack instance, click on the hypertext link near the top of the first paragraph. This will take you to the OpenStack Dashboard (shown in *Figure 9*), where you will enter the login credentials.

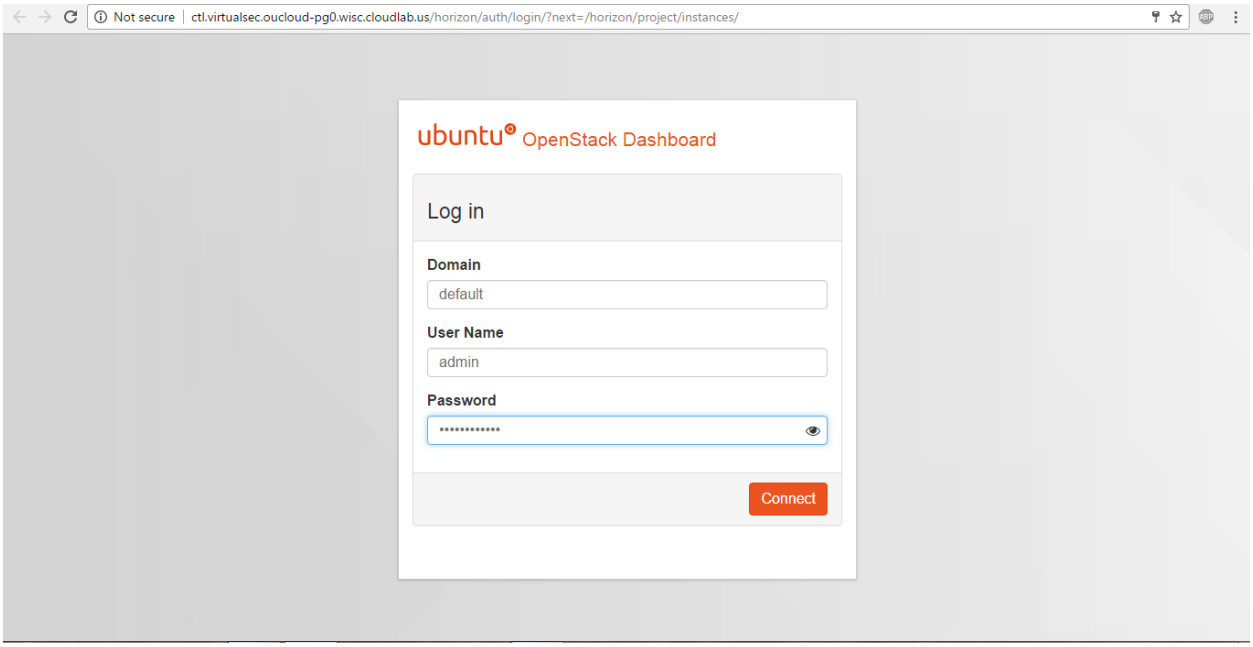

Figure 10: OpenStack Login page, using the credentials highlighted in Figure 8

After logging in to the OpenStack instance, you will be able to access the experiment.

Part III: Creating a Network

Once you have signed in to your the OpenStack instance, click the "*Network"* button in the menu bar to open the Network submenu. Then click on the "*Networks"* label. This displays a list of all the networks currently in your experiment. Click on the "*Create Network"* button to create a new network.

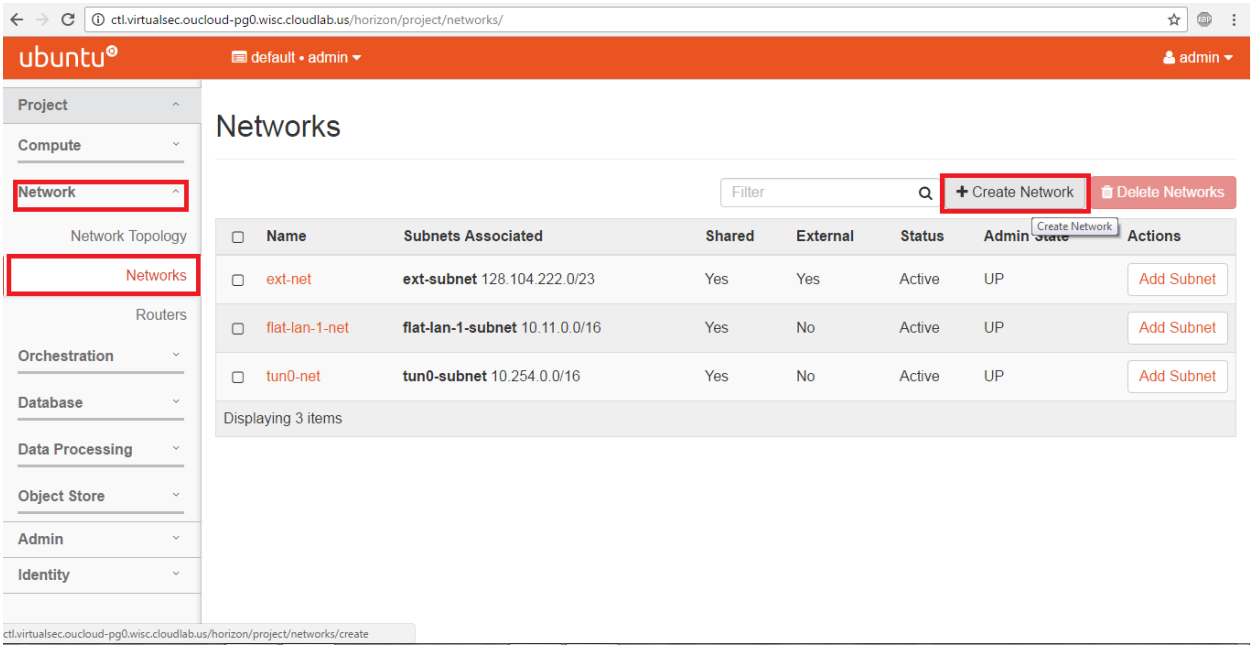

Figure 11: Creating a new network in OpenStack

It's important to note that the "*ext-net"* refers to the external (public) network interface of the instance. Connecting routers/hosts to this network makes them reachable over the public internet. The *"tun0-net"* is a network consisting of EGRE tunnels, and "*flat-lan-1-subnet"* is a generic network setup to allow you to easily SSH into any created VM instances through the physical nodes running the experiment.

When the network creation window opens, you can enter the details of your new network. You should make sure the "*shared*" checkbox is ticked in the "*Network*" tab. You can assign an IP address to your subnet under the "*subnet"* tab. The exact network address is unimportant, so long as it is a valid address that hasn't been assigned to any other networks in your experiment. In the example images, there is a network named "VictimNetwork" with a subnet named "VictimCorporation" holding the address of 10.0.0.0/24.

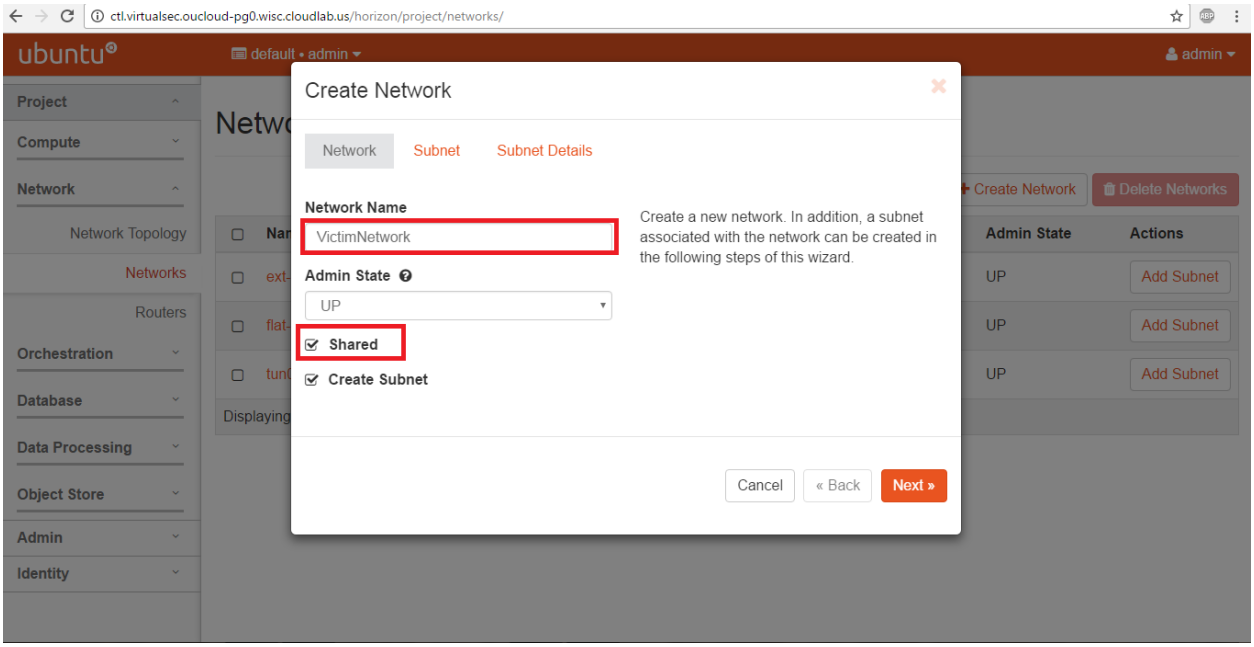

Figure 12: Input network parameters of name and nature in network creation

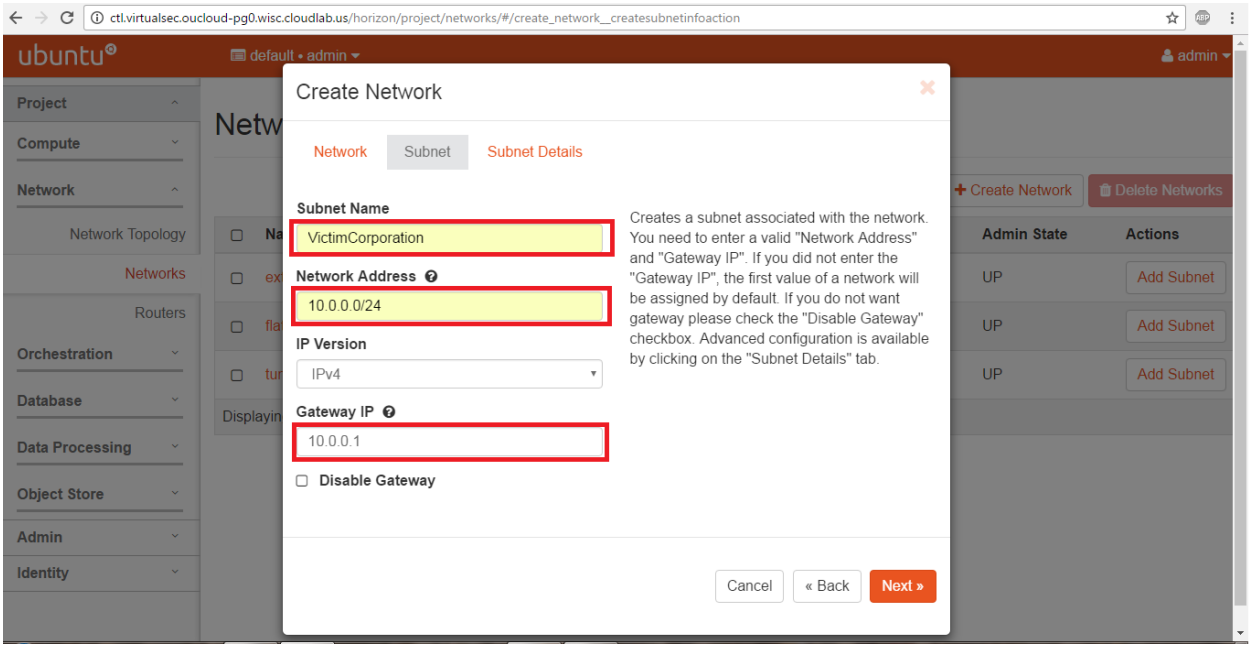

Figure 13: Input subnet details relevant to the created network

If needed, you can adjust the subnet's settings by clicking the "*Subnet Details*" tab.

Upon the completion of each dialog, click "*Next*". When it is done, click on "*Finish"* to save the settings.

Part IV: Creating a Router

You can configure a new router instance as well. To do that, in the ”*Networks*” submenu, first click “*Routers”*, then click “*Create Router*” as displayed below.

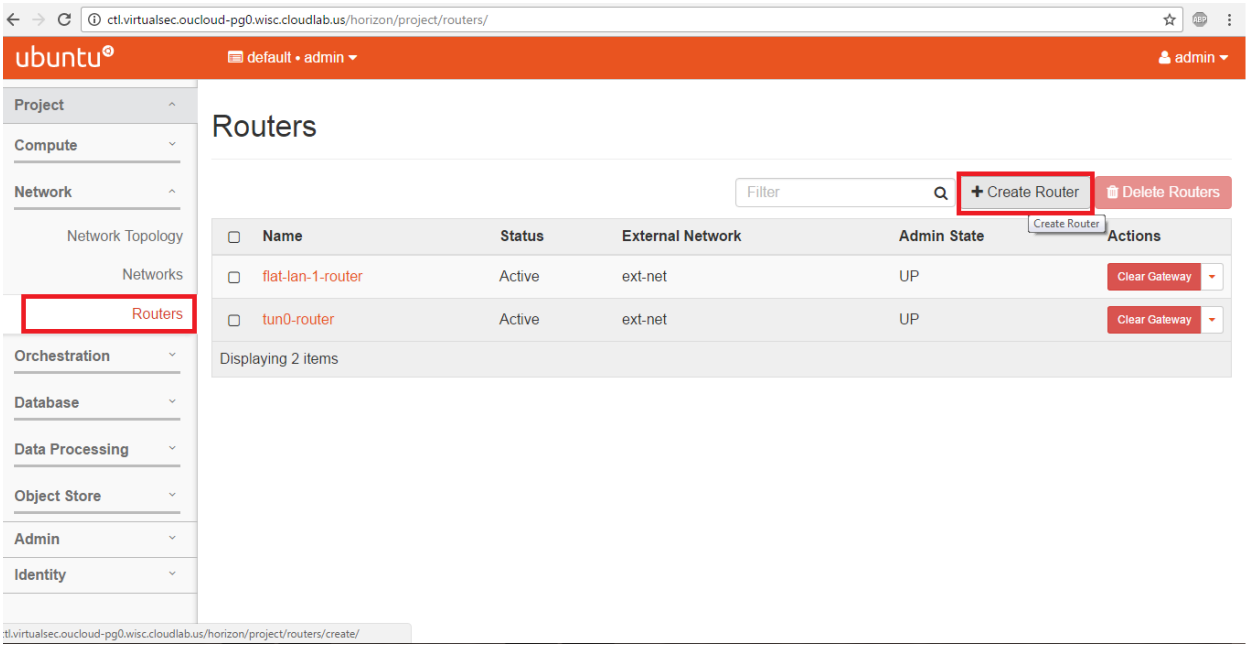

Figure 14: OpenStack creating a new router instance

In the new pop-up dialog, give the router a new name, and connect it to the public external network (“*ext-net”)*. This is so it is accessible over the public internet, allowing an SSH client to connect to the VM and control it from the public internet. More explanation can be found in **Part VI: Access the Created VM**.

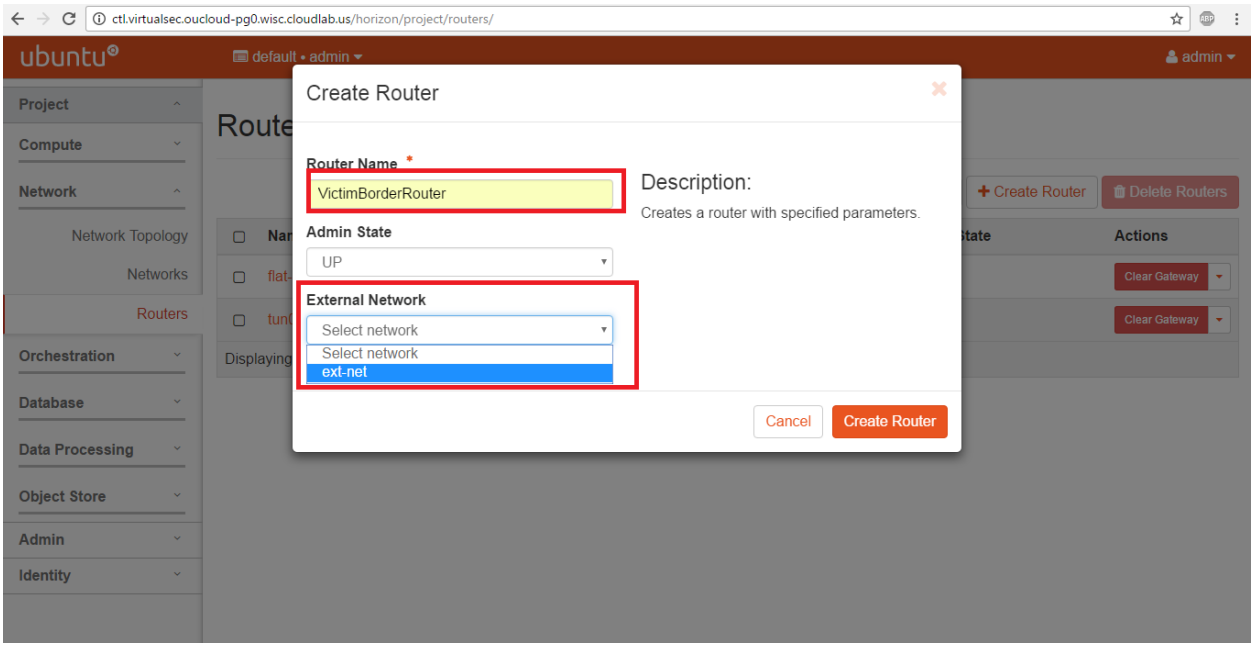

Figure 15: OpenStack modifying new router parameters

In order to connect the new router to a network, you need to add an interface to the router. Click on the desired router from the router list, select the "*Interfaces"* tab, and hit the "*Add Interface"* button, as displayed in *Figure 15*.

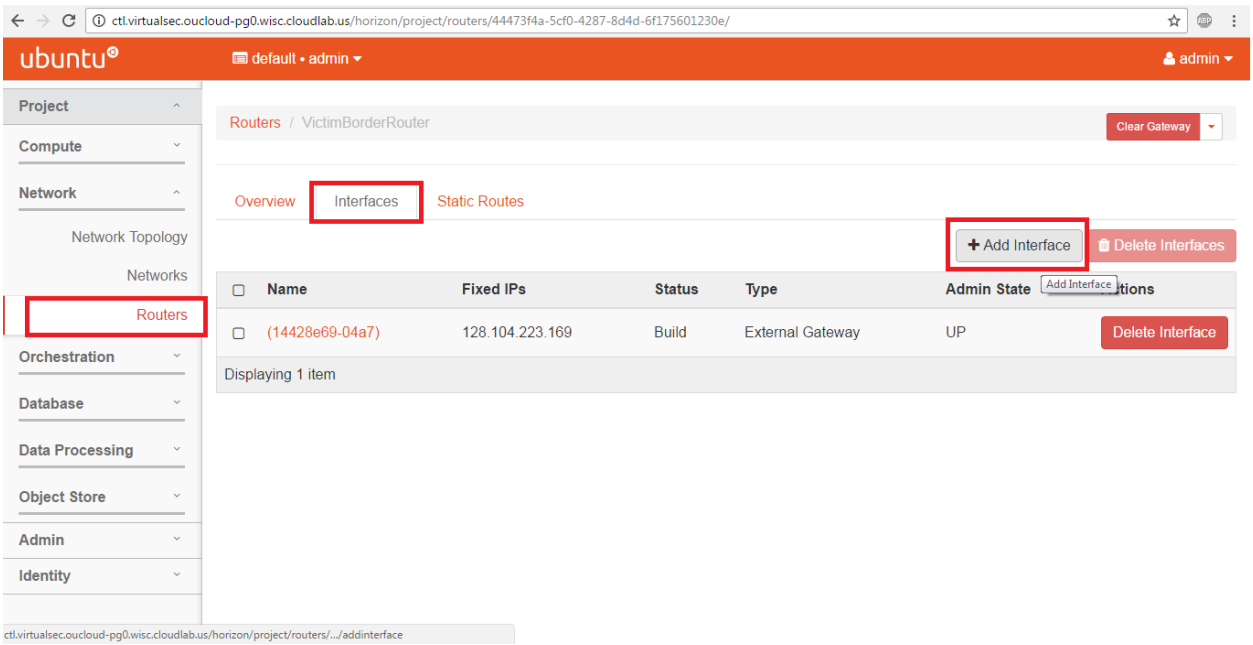

Figure 16:OpenStack connecting routers to new networks

In the new popup dialog, select which network would you like the new interface to connect to.

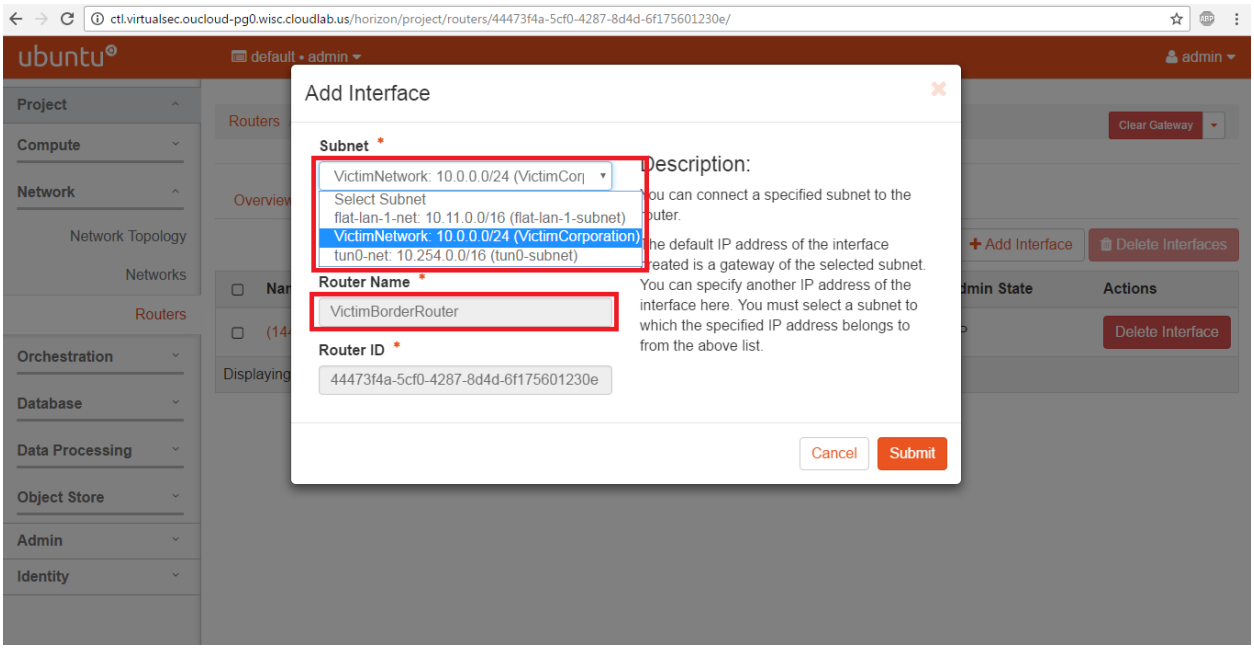

Figure 17: Choosing which new network router should connect to in OpenStack

In *Figure 16*, the new router is named "*VictimGateRouter*", and has two interfaces: one is connected to the public network, while the other is connected to the "*VictimNetwork*". Now, nodes inside of the "*VictimNetwork*" can interact with the public internet via the new router to the external network.

Part V: Creating a New VM

To create a new VM, click on *"Compute"* in the menu bar, followed by *"Instances"* in the sub-menu, then click *"Launch Instance"* as shown in *Figure 17*.

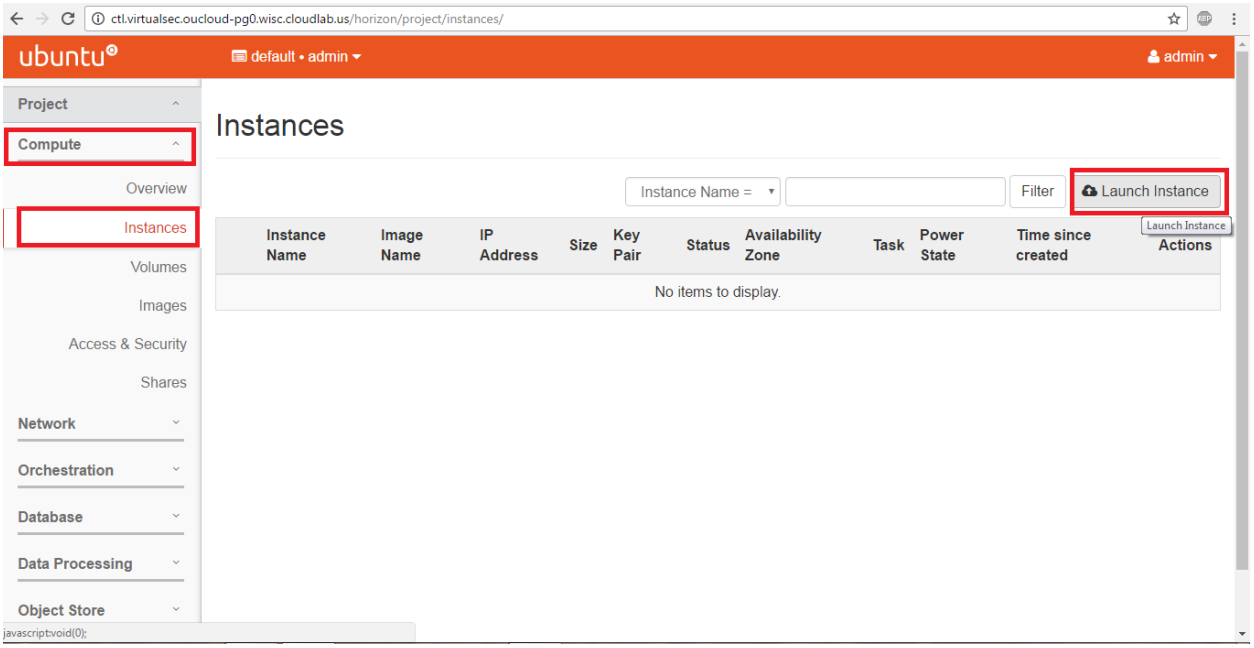

Figure 18: OpenStack launching a new VM instance

The new instance will either use a VM image of Ubuntu14 ("*trusty-server*") or Ubuntu12 ("*manilla-service-image"*).

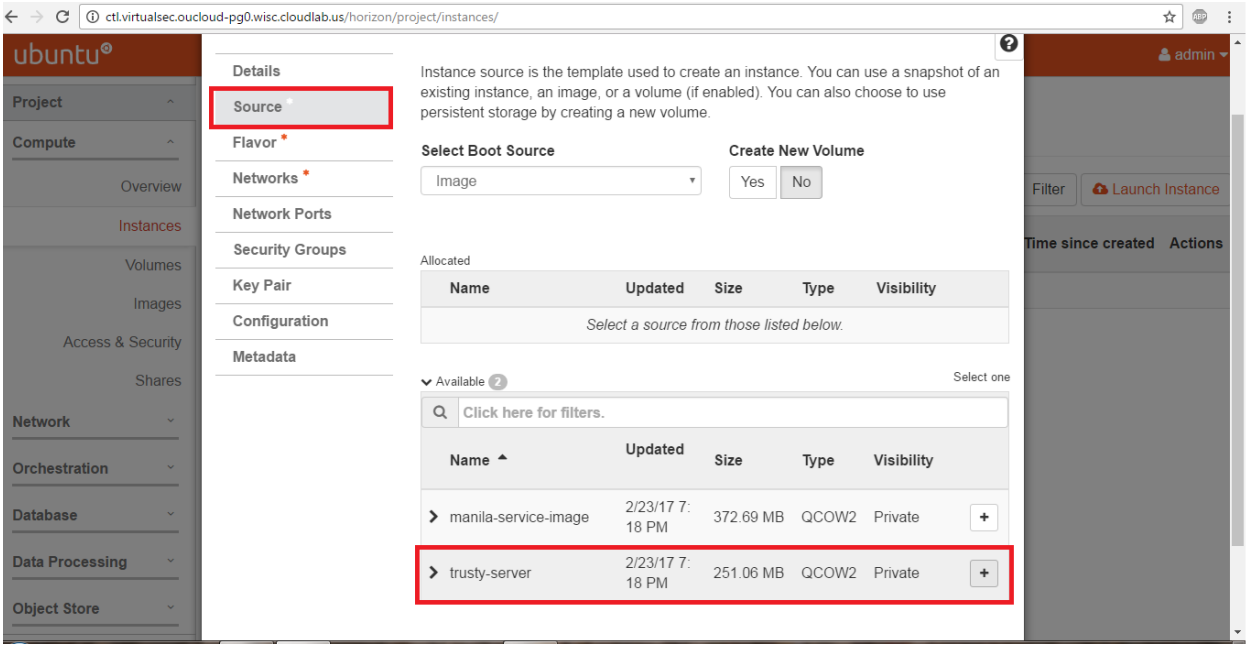

Figure 19: Modifying image source for new VM

Now, you will need to select the "flavor", or size, of the operating system you wish to install. This determines the amount of available memory, storage, and computing power the instance will have. We recommend the user to select any size other than "*tiny*", as you may face errors due to size mismatch with the disk space, and will not be able to boot your VM. You can choose your desired image size by clicking on the "+" button on the right edge of the option. With custom images, you may have to choose a larger size. Medium is the recommended size for custom images.

In *Figure 19*, the VM runs the "*trusty"* image with the "*small*" flavor.

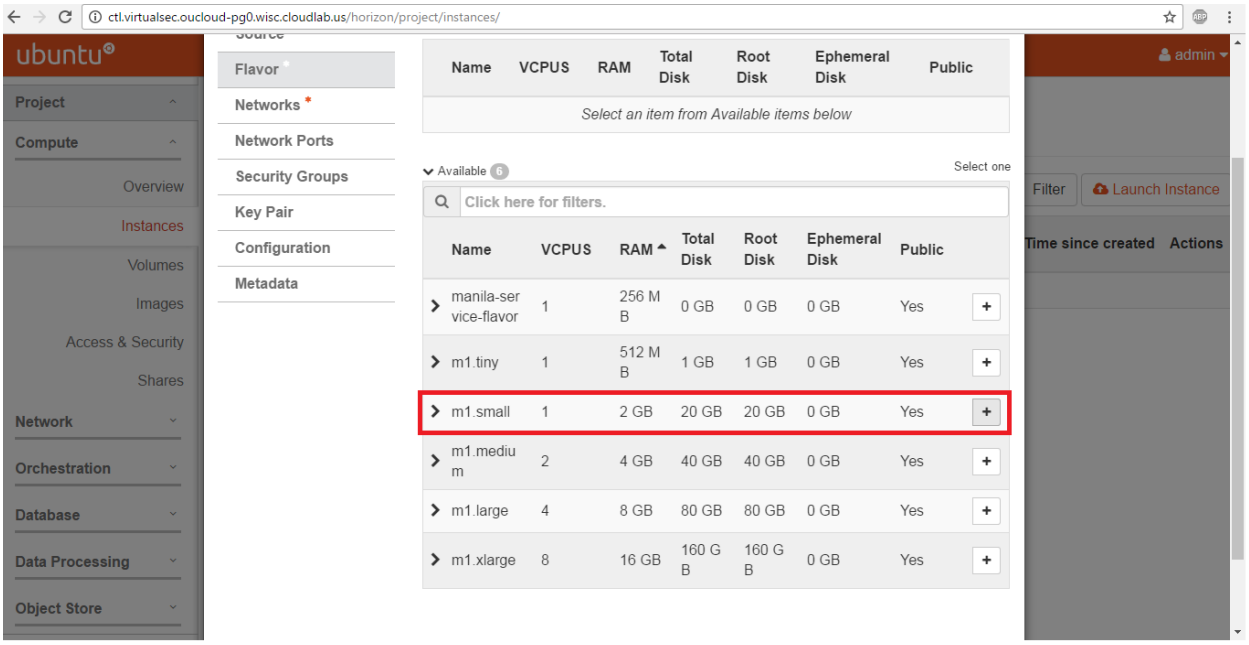

Figure 20: Choosing the flavor of the image for the new VM

As illustrated in *Figure 20*, the third item on the menu is the "*Networks"* option, which allows you to connect the VM to a network in your experiment. Press the "+" button next to add the VM to the network(s).

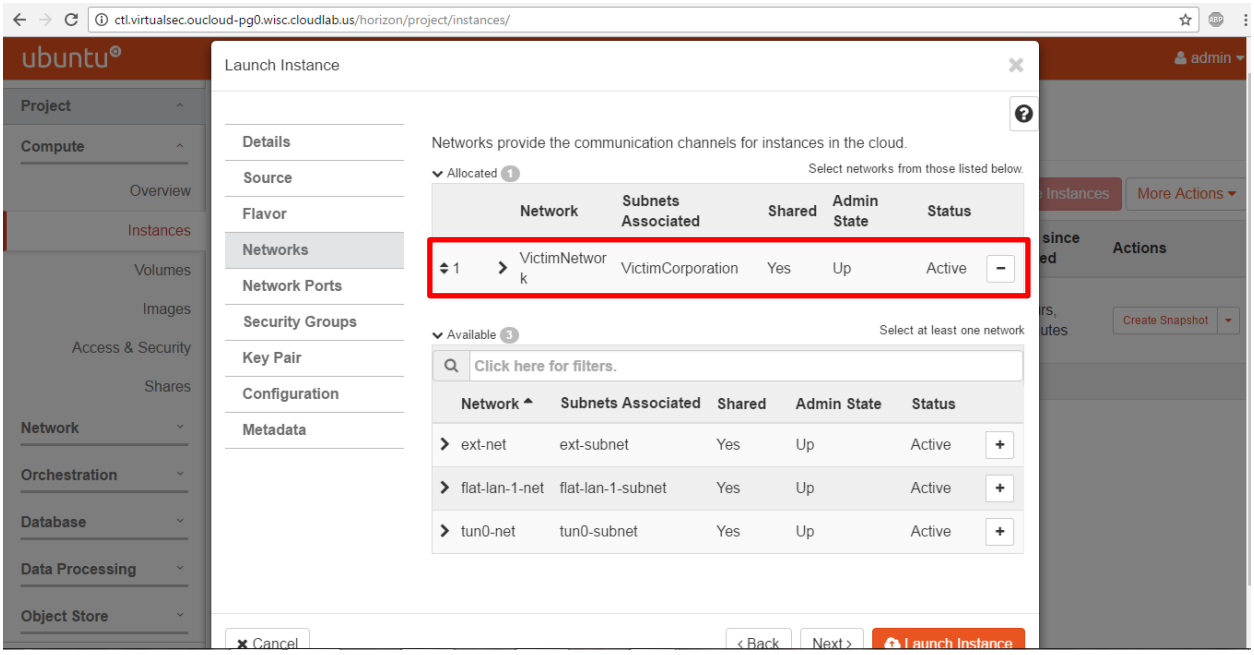

Figure 21: Attaching new VM to desired network

Once everything is done, click "*Launch Instance*". In a couple of minutes, your VM should be ready to log in to.

You can snapshot the VM by clicking the "*Create Snapshot*" button.

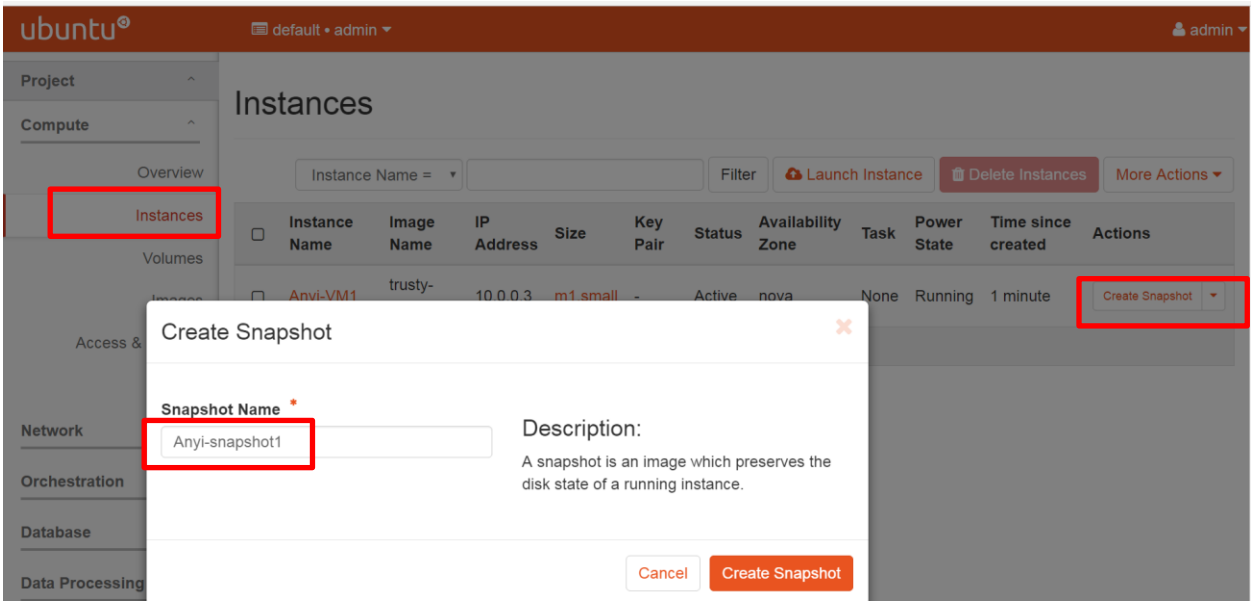

Figure 22: Creating an image snapshot

Part VI: Access the Created VM

In CloudLab, you can use different ways to access a VM. We list just two possible ways below.

Option A: Managing through the default console (not recommended)

The default method deployed by OpenStack to control your VM is through the "*Console*" tab, found by clicking on the instance you wish to work on from the "*Instances*" page under the "*Computer*" submenu. The corresponding screenshot is displayed in *Figure 22*.

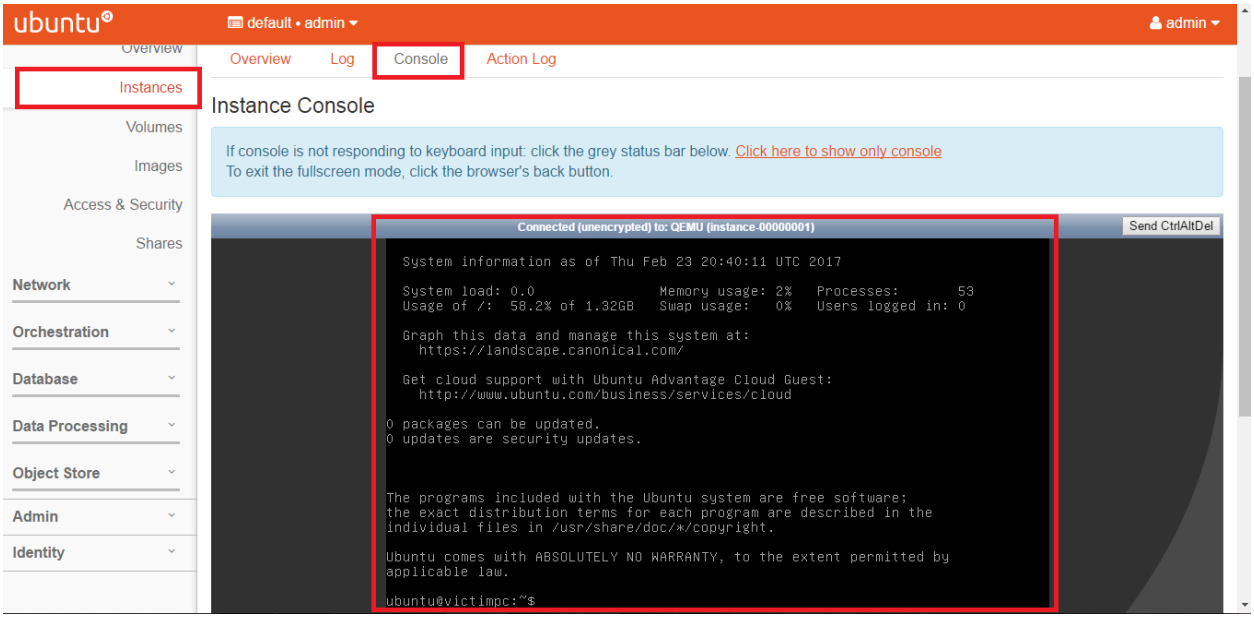

Figure 23: Viewing a VM console in OpenStack

However, accessing and modifying the VM through the console is very slow. Commands will often take a long time to execute. The console session also suffers from frequent hanging, and refreshing the page or opening a new Dashboard instance in the browser is required to use the console again. In addition, depending on your firewalls and settings, you may not be able to directly access the console at all.

Usually, it is better to bypass the in-browser console by connecting to the VM with your own computer through SSH.

Option B: Managing through SSH over VM Public IP

As mentioned before, we recommend you acquire more than two floating IP addresses. The additional IPs will be useful in this step, which allow us to associate them with the virtual machines. Associate a public IP address to the VM, and then we can connect to that IP address from an SSH client (such as PuTTY, SecureCRT, or MobaXterm).

To associate a public IP to the VM, we start by clicking on "*Instances"* in the "*Compute"* submenu. Then, click the drop down menu next to the VM you wish to connect to, and choose "*Associate Floating IP*". *Figure 23* shows this step.

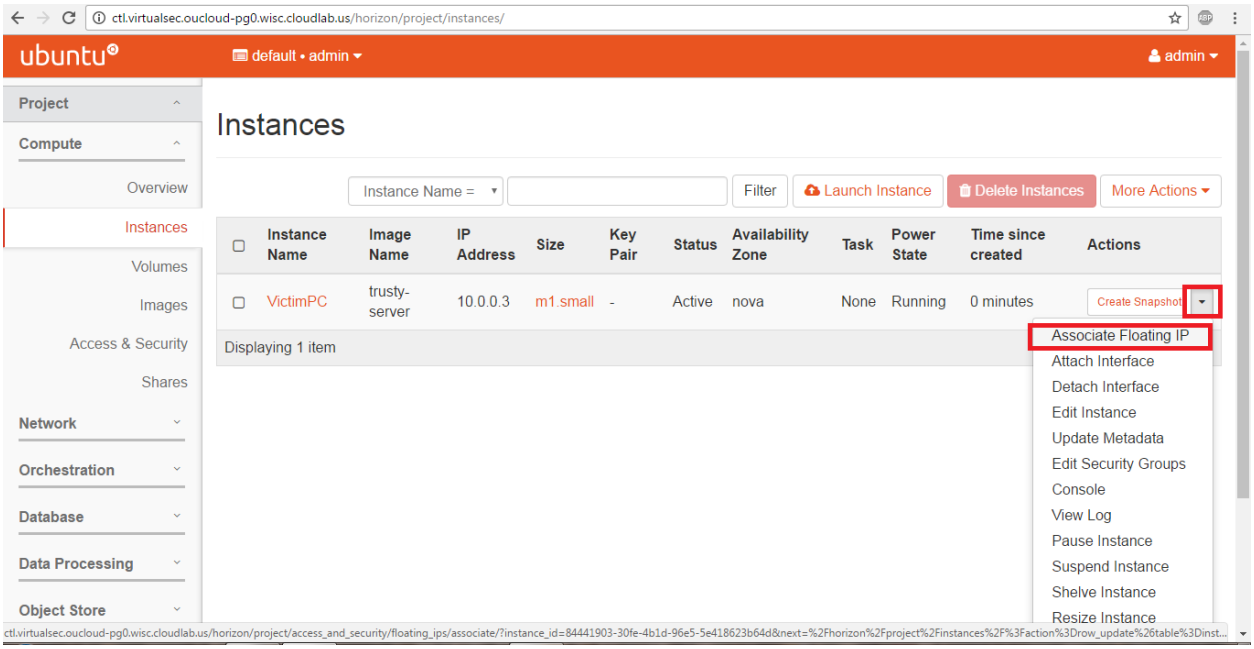

Figure 24: Associating a Floating IP for a VM using OpenStack GUI

A popup window will appear, allowing you to configure your floating IP. If no floating IP addresses have yet been assigned to this instance, you will need to create it by pressing the "+" button.

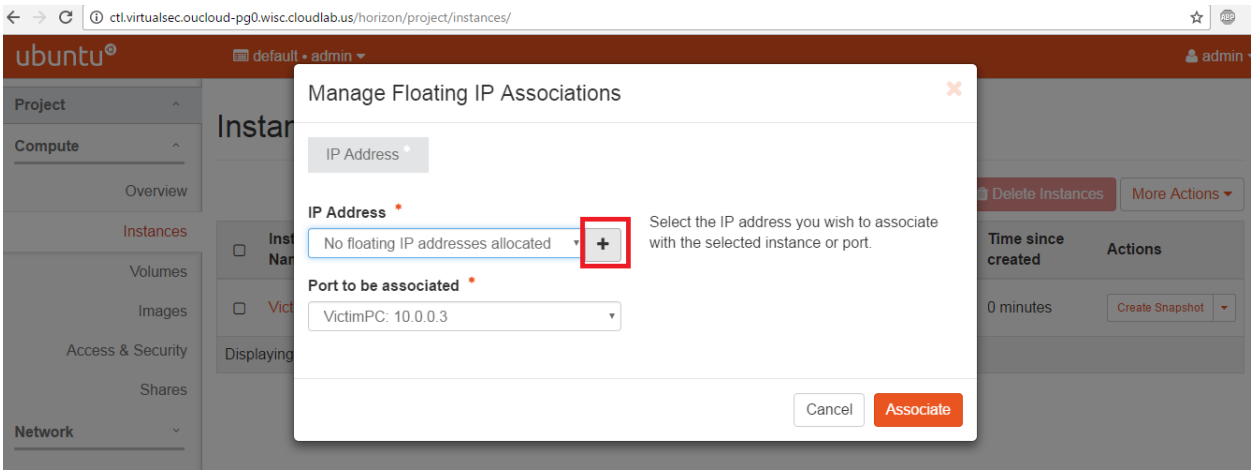

Figure 25: Creating a new Floating IP using OpenStack GUI

Once a new public IP has been allocated, make sure it is selected from the "*IP Address"* dropdown menu, and choose which port it should connect with. After clicking "*Associate",* the floating IP will now appear under the internal IP address in the "*Instances"* tab.

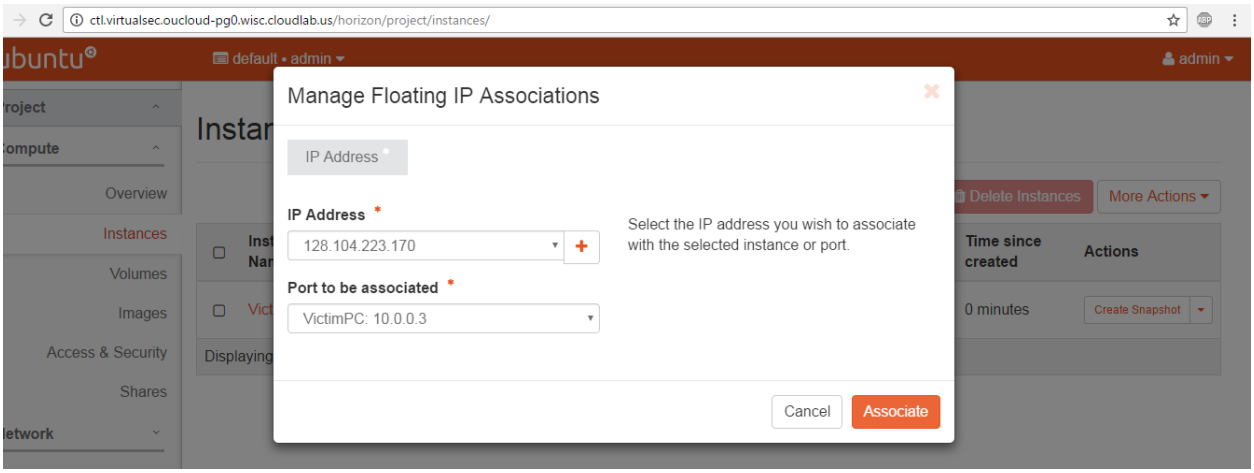

Figure 26: Assign the Floating IP to VM in OpenStack GUI

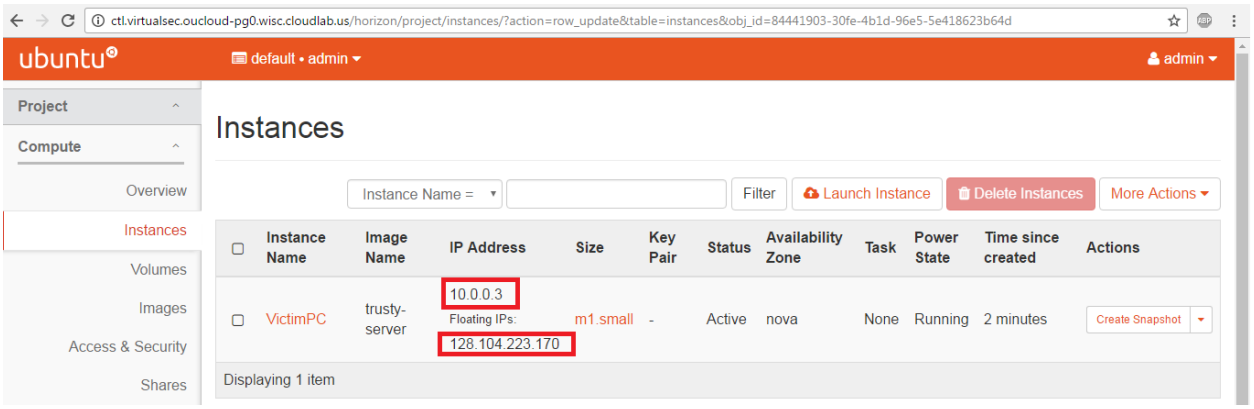

Figure 27: VM status if Floating IP association is successful.

Now you can connect to the public IP of the VM with an SSH client to configure it. This approach is generally much better than controlling the VM via the online console. The only downside is that if the associated interface of the VM fails, then the SSH session will fail correspondingly, and the only access is back via the in-browser console.

Once all the above steps are done, you have configured a simple network of a router and a virtual machine, connected to the public internet via the gateway, and mimicking a connection to the "VictimPC" VM. The topology can be viewed by clicking on "*Network*", followed by "*Network Topology*".

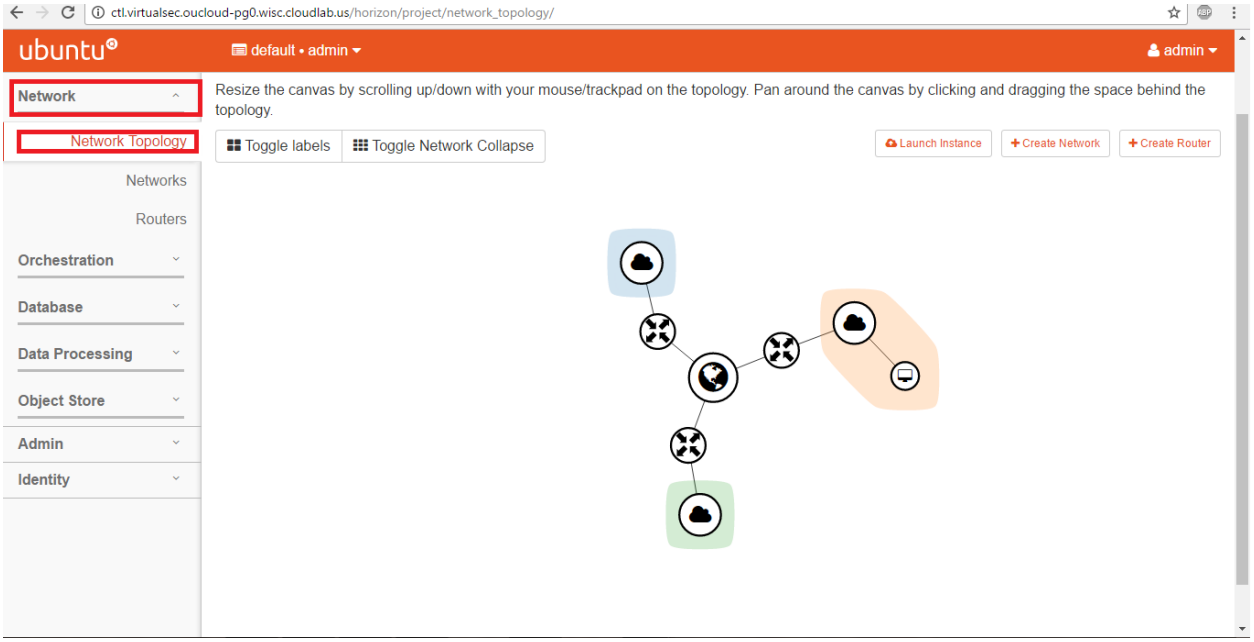

Figure 28: Network Topology view using OpenStack GUI