CS 4001/7001 Cloud Computing
Spring 2015
Lab # 4: InterCloud Web Services for OpenStack-based Cloud Orchestration

Dr. Prasad Calyam & Ronny Bazan Antequera (Contact: calyamp@missouri.edu)

1. Purpose of the Lab

Deploy your own "personal cloud" on CloudLab (www.cloudlab.us) to understand how to program web services across multi-cloud platforms that are orchestrated using OpenStack cloud operating system.

2. References to guide lab work

[1] OpenStack - http://www.openstack.org
[2] OpenStack, User Guide - http://docs.openstack.org/user-guide/user-guide.pdf
[3] NSF CloudLab:
- Main Website - https://www.cloudlab.us
- CloudLab Resources - https://www.cloudlab.us/hardware.php
- CloudLab Manual - http://docs.cloudlab.us/manual.pdf

[4] Chapters 4 and 5, Distributed and Cloud Computing, Hwang, Fox & Dongarra

3. Lab Steps and output collection guidelines

Figure 1: Personal Cloud Infrastructure Deployment

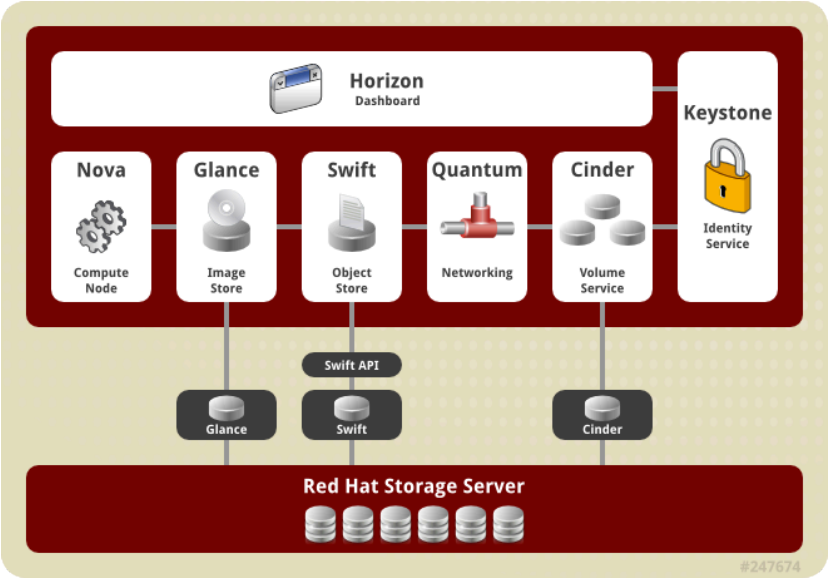
Figure 1 shows the required steps to successfully deploy your personal cloud infrastructure. You will be introduced to OpenStack services terminology [1], and you will follow instructions [2] that show how to customize and check status of your cloud resource instances and check status through GUI and command-line tools [3]. Lastly, you will configure web services to interact with external cloud services using RESTful APIs for tasks such as multi-cloud resource discovery and instance types.

3.1 OpenStack Overview

The OpenStack project is an open-source cloud operating system, which aims to be simple to implement, massively scalable, and feature rich. Developers and cloud computing technologists from around the world have joined together to create the OpenStack project.

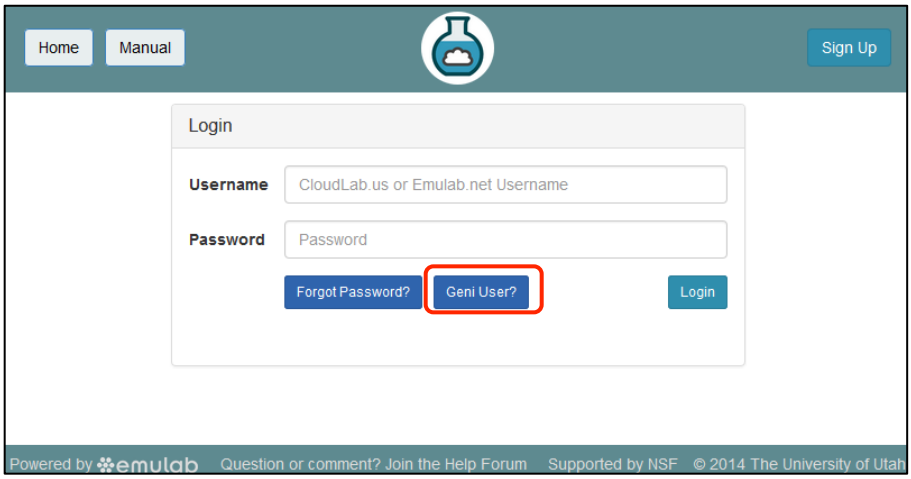
OpenStack provides an Infrastructure as a Service (IaaS) solution through a set of interrelated services. Each service offers an API that facilitates this integration. Depending on the cloud computing needs, you can install some or all services in your deployment. The following table describes select ***OpenStack Services*** that make up the overall OpenStack architecture.

Service Type	Name	Description
	Core Services	
Dashboard	Horizon	Provides a web-based self-service portal to interact with underlying OpenStack services, such as launching an instance, assigning IP addresses and configuring access controls.
Compute	Nova	Manages the lifecycle of compute instances in an OpenStack environment. Responsibilities include spawning, scheduling and decommissioning of machines on demand.
Networking	Neutron	Enables network connectivity as a service for other OpenStack services, such as OpenStack Compute. Provides an API for users to define networks and the attachments into them. Has a pluggable architecture that supports many networking vendors/technologies.
	Storage Services	
Object Storage	Swift	Stores and retrieves arbitrary unstructured data objects via a RESTful, HTTP based API. It is highly fault tolerant with its data replication and scale out architecture. Its implementation is not like a file server with mountable directories.
Block Storage	Cinder	Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the creation and management of block storage devices.
	Shared Services	
Identity Service	Keystone	Provides an authentication and authorization service for other OpenStack services. Provides a catalog of endpoints for all OpenStack services.
Image Service	Glance	Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.
Telemetry	Ceilometer	Monitors and meters the OpenStack cloud for billing, benchmarking, scalability, and statistical purposes.
	Higher-level Services	
Orchestration	Heat	Orchestrates multiple composite cloud applications by using either the native HOT template format or the AWS CloudFormation template format, through both an OpenStack-native REST API and a CloudFormation-compatible Query API.



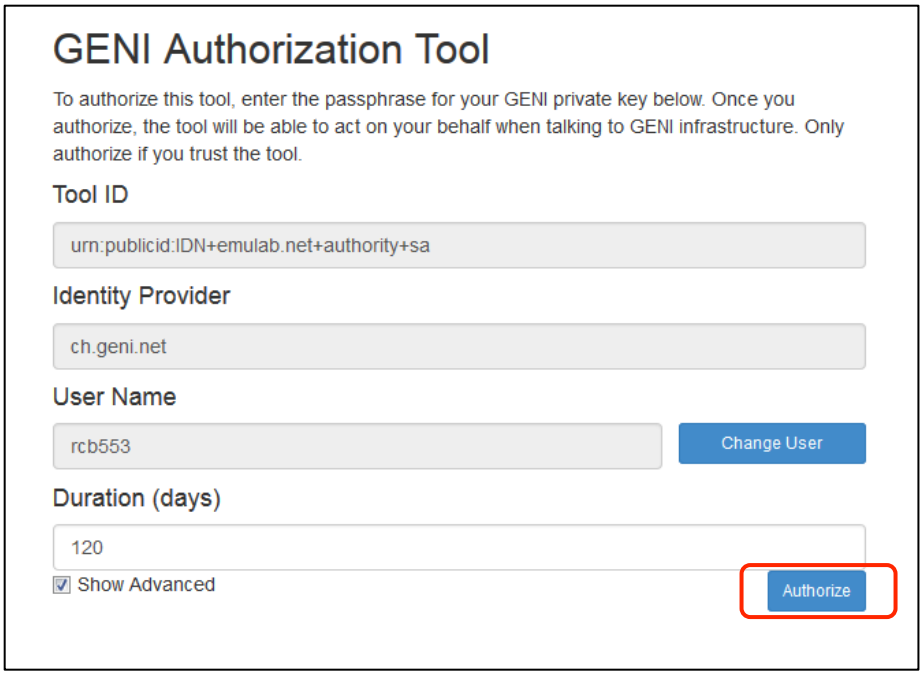
3.2. OpenStack Infrastructure Deployment in CloudLab

CloudLab allows EMULAB and GENI users to use their credentials, hence certificate and private key are stored in your web browser and your passphrase is never transmitted over the network. Open a web browser and enter https://www.cloudlab.us, select 'Geni User?' button.



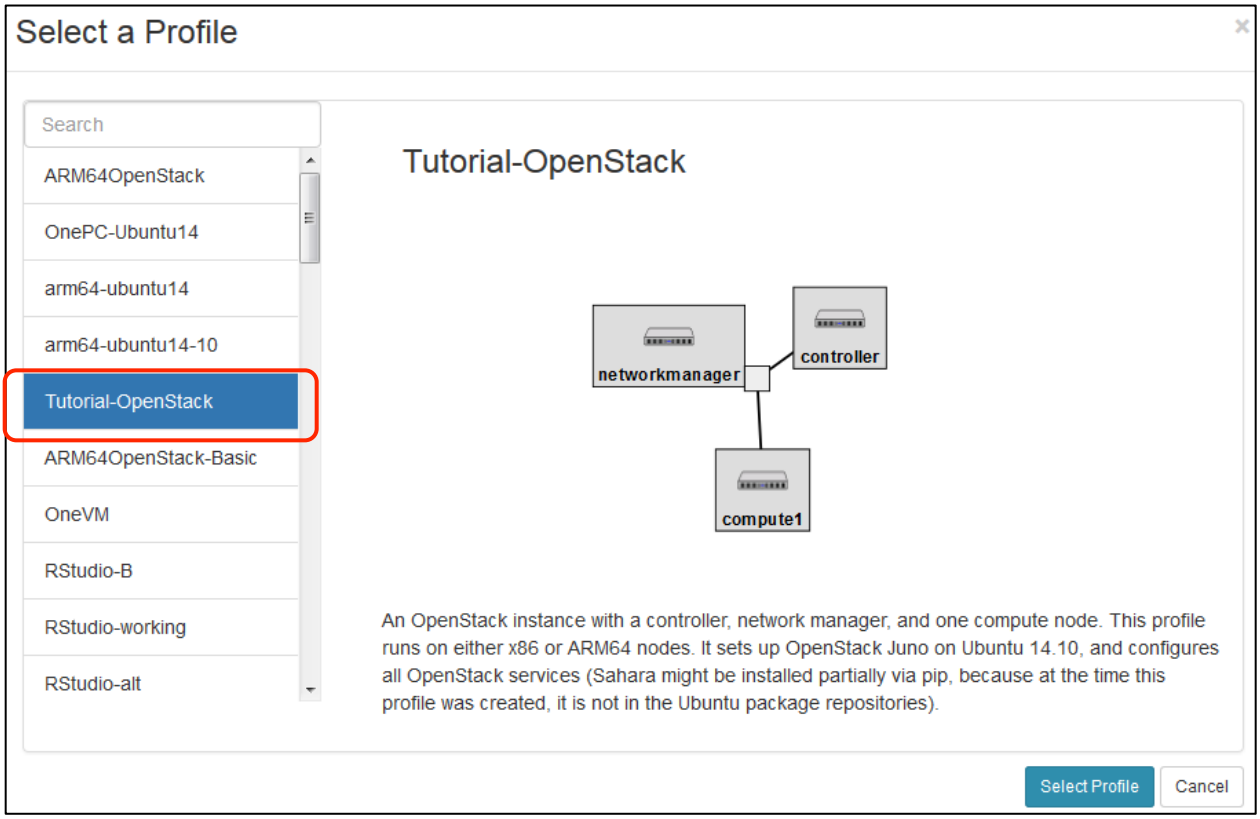
In the next screen select GENI logo and then click on the 'University of Missouri System' logo

Once you input your credential click on 'Authorize' button to allow CloudLab use your GENI credentials.

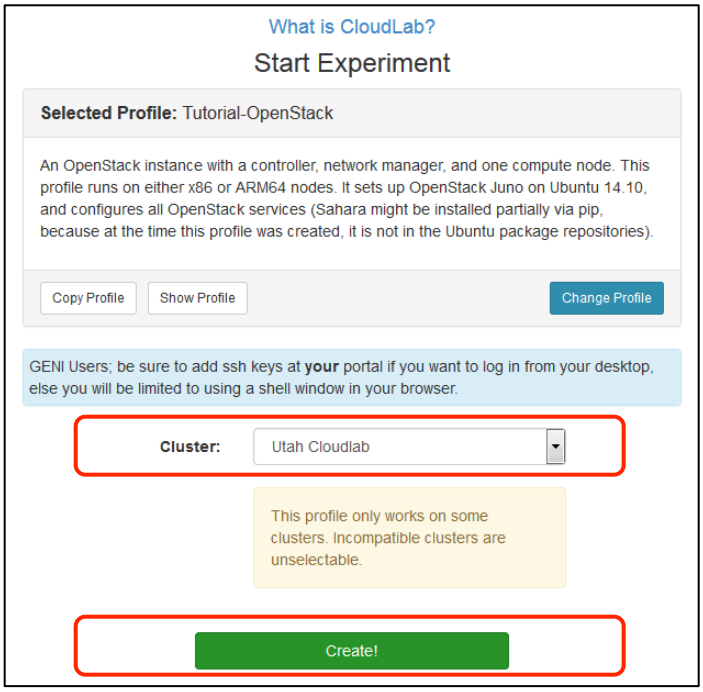


3.2.1 Run an experiment.

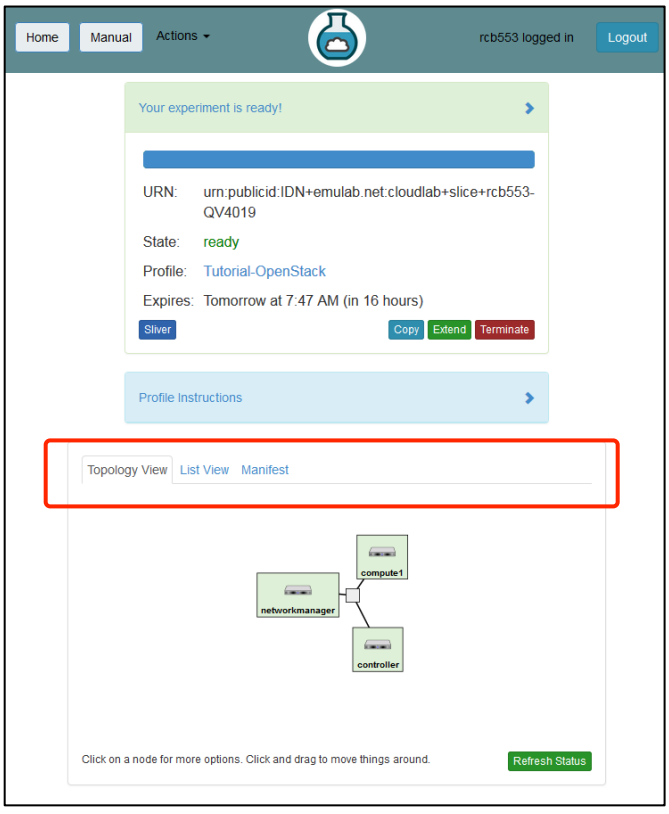
Select the 'Tutorial-OpenStack' profile that will deploy a controller, network manager and 1 compute node as shown in the figure below.



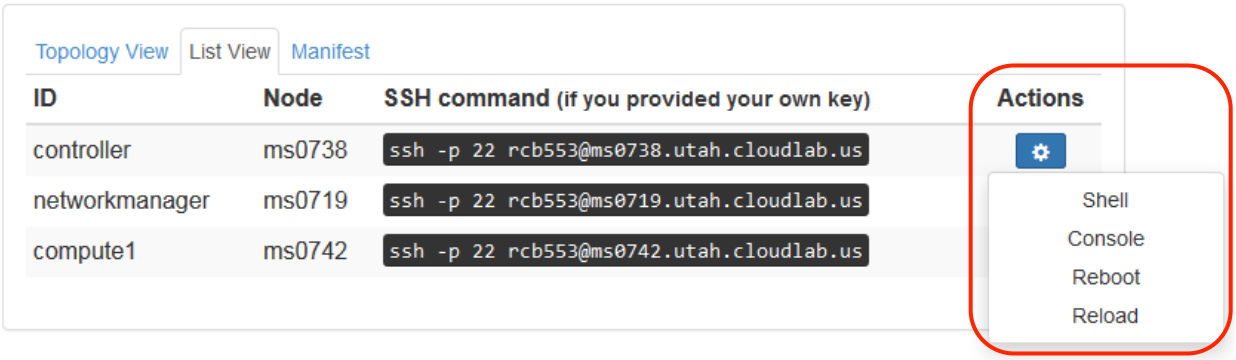
Select Utah CloudLab and click on ‘Create’ button to deploy your experiment.



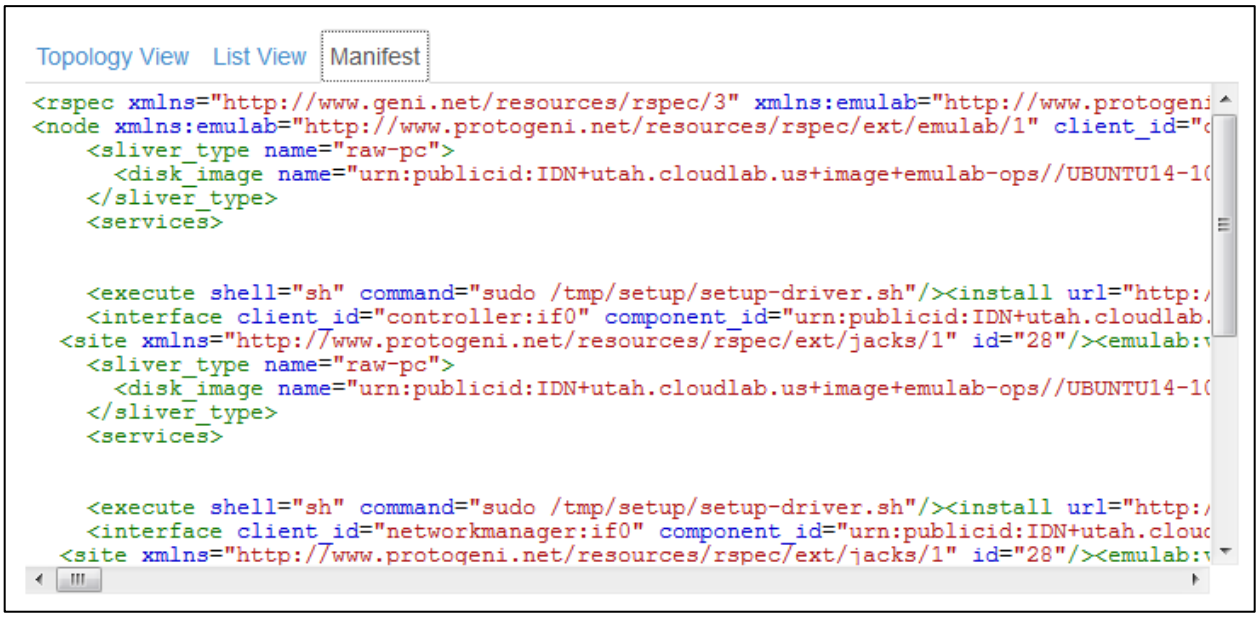
After a few minutes the state will change from ‘booting’ to ‘ready’. Once the cloud infrastructure is available an email will be sent to you.



In the bottom part you will find 3 tags for: Topology, List View and Manifest. Topology displays the graphic representation of the nodes just created. List View displays the nodes' SSH information. Also by clicking on the 'Actions' button you will display additional options for each node. Click on the 'Shell' option to access the various node terminals of the: 'contrøller', 'networkmanager' and 'compute1'.



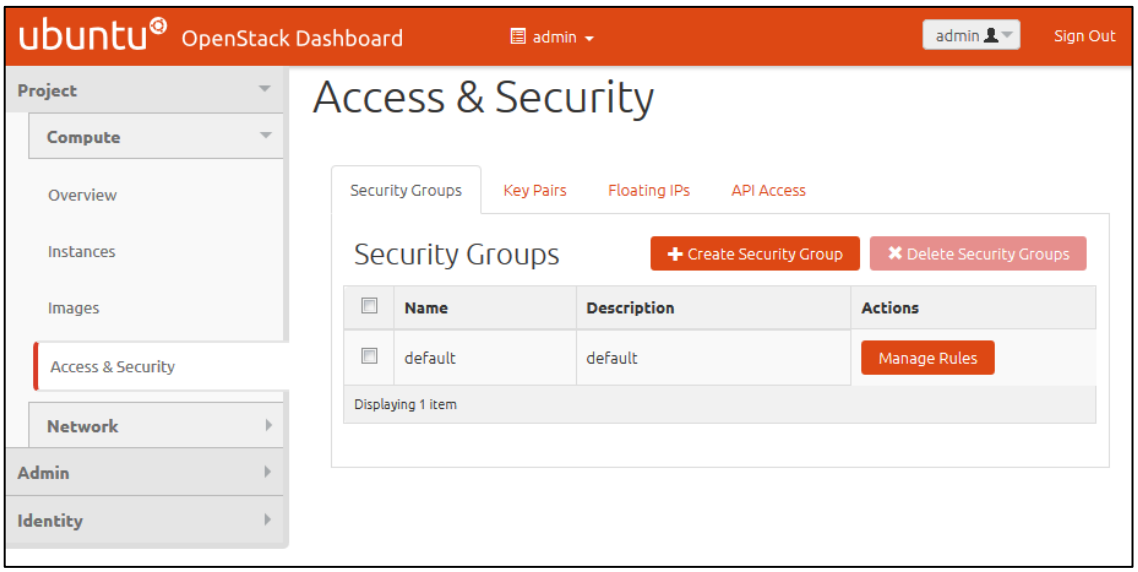
Finally, 'Manifest' tab displays the profile manifest that correspond to the deployed cloud.



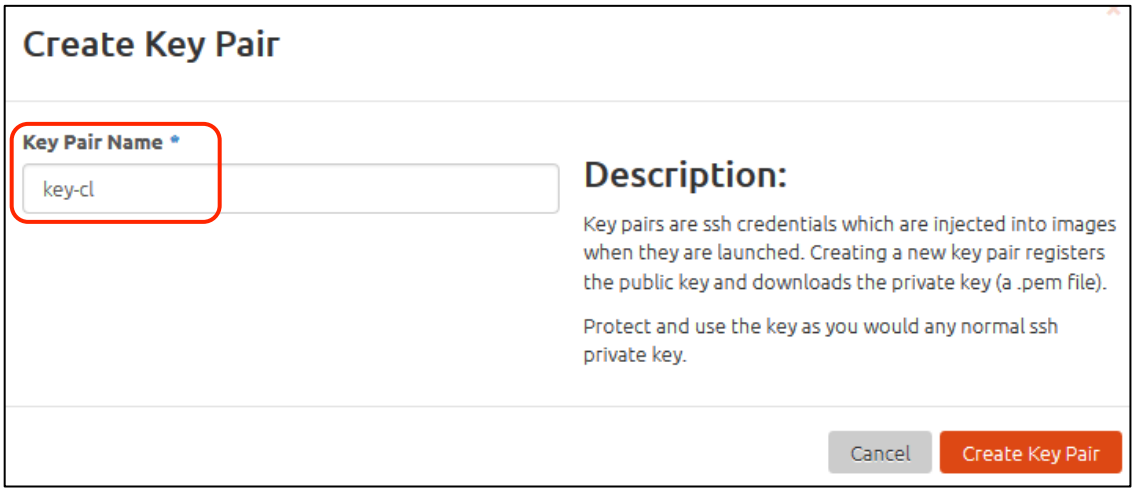
3.3 Accessing the OpenStack Dashboard

To access to the OpenStack dashboard, click on the link *'the OpenStack controller WWW interface'* under 'Profile Instructions' and type the user name and password described there, same password is valid for instances.

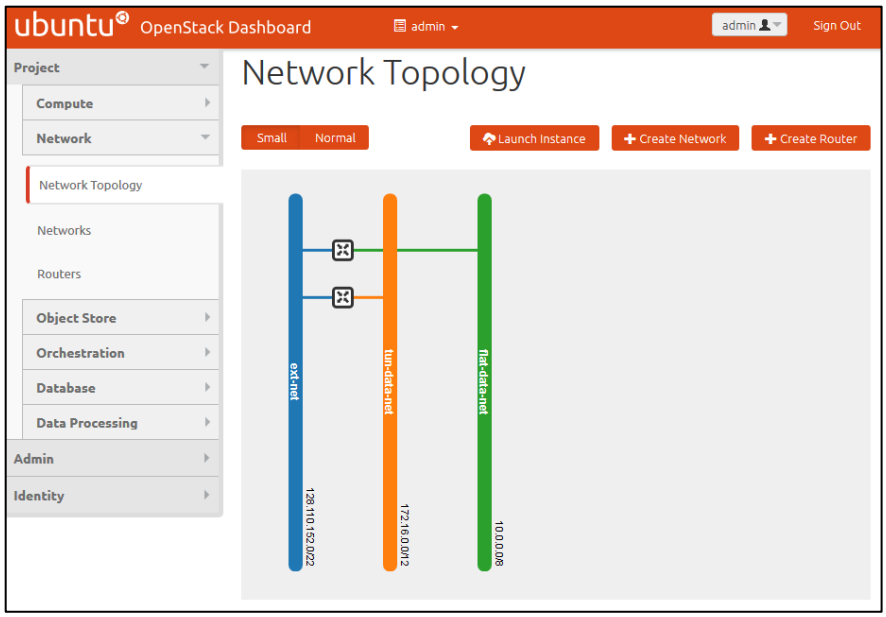
- Similar to AWS Amazon Lab01, first configure the Security Group and create your Key Pair.



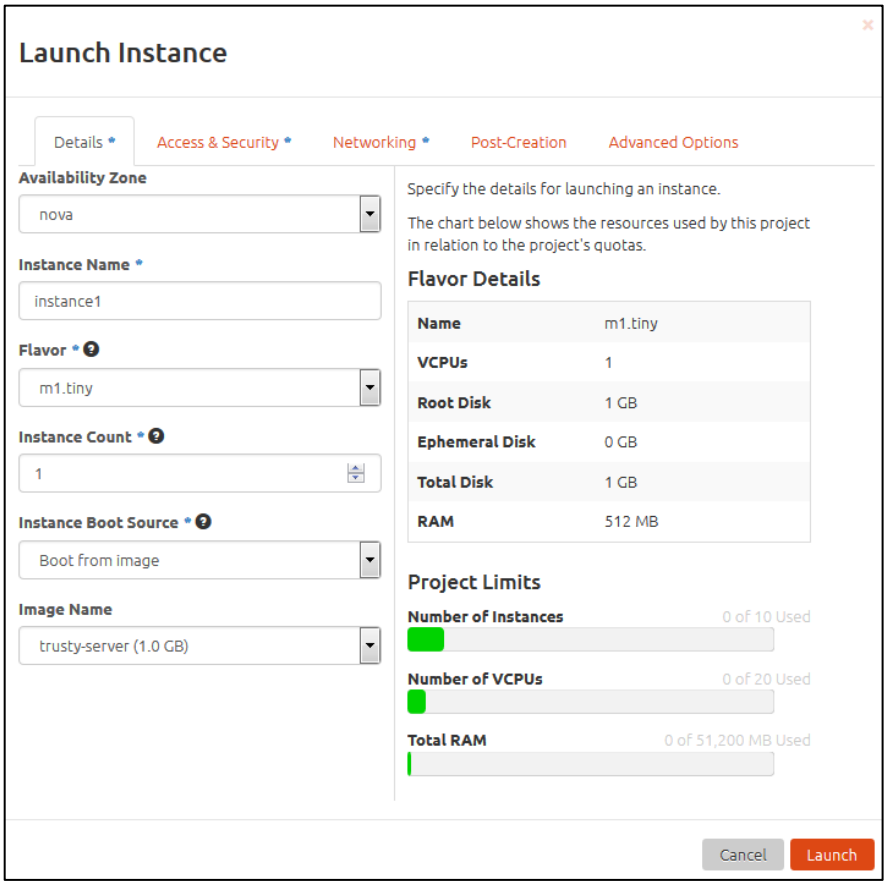
- Copy your key 'key-cl' to the 'networkmanager' node.



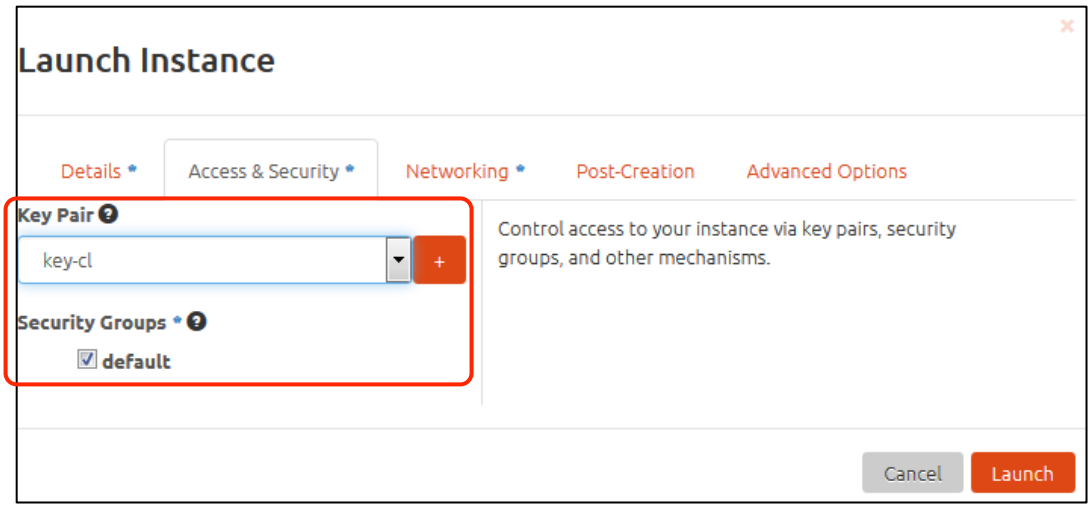
- In the Network Topology verify that you have 3 Network configurations. **We will take a screenshot of this topology for grading purposes.**



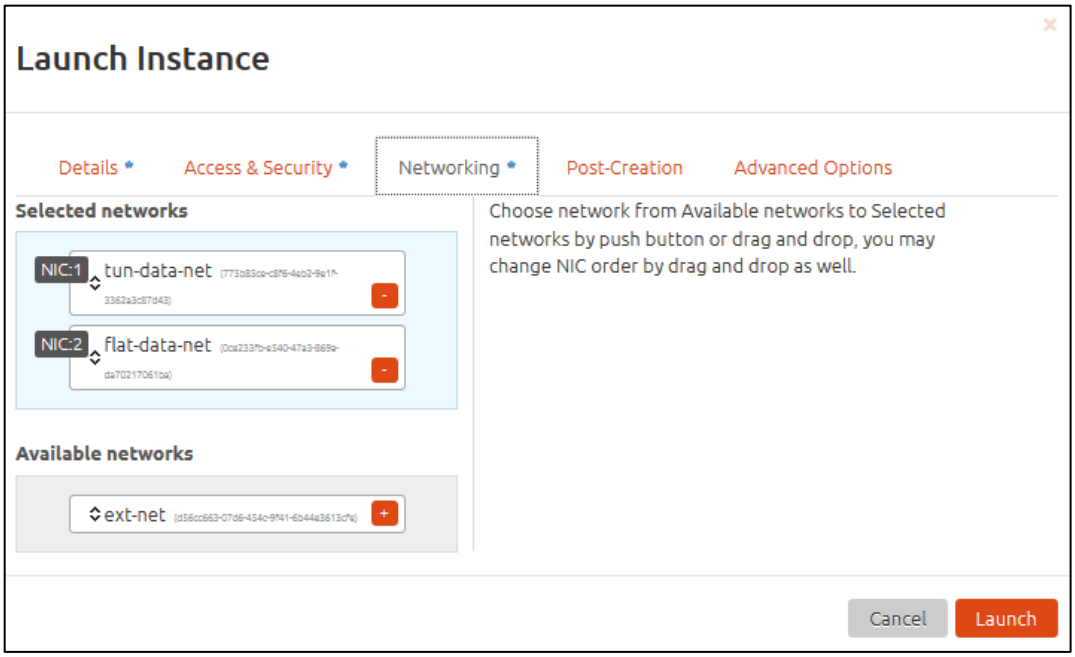
- Create a new instance by selecting the options similar to the graph below.



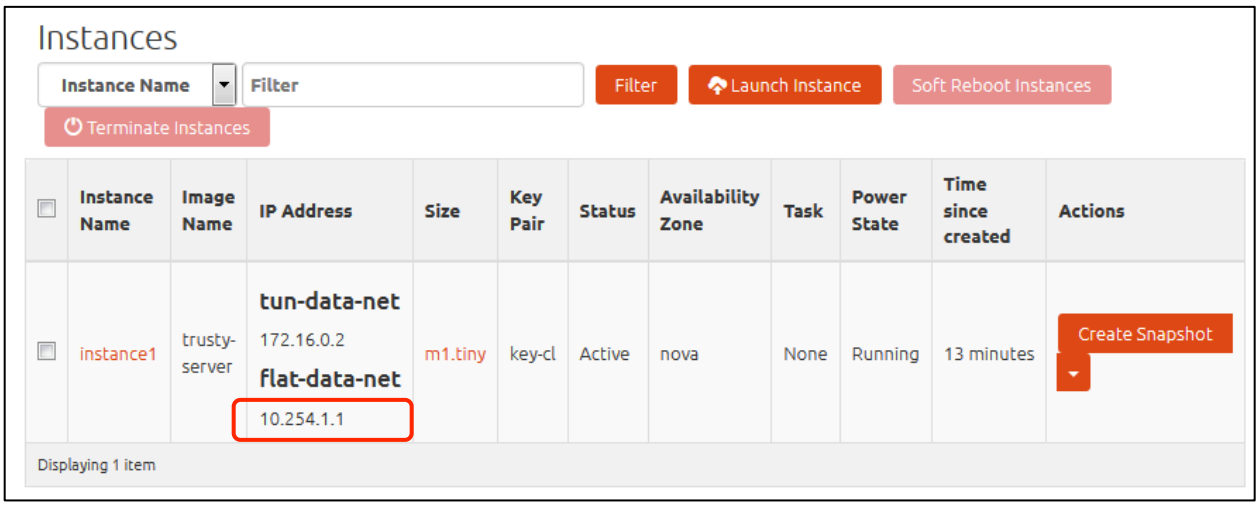
- Select your key and group



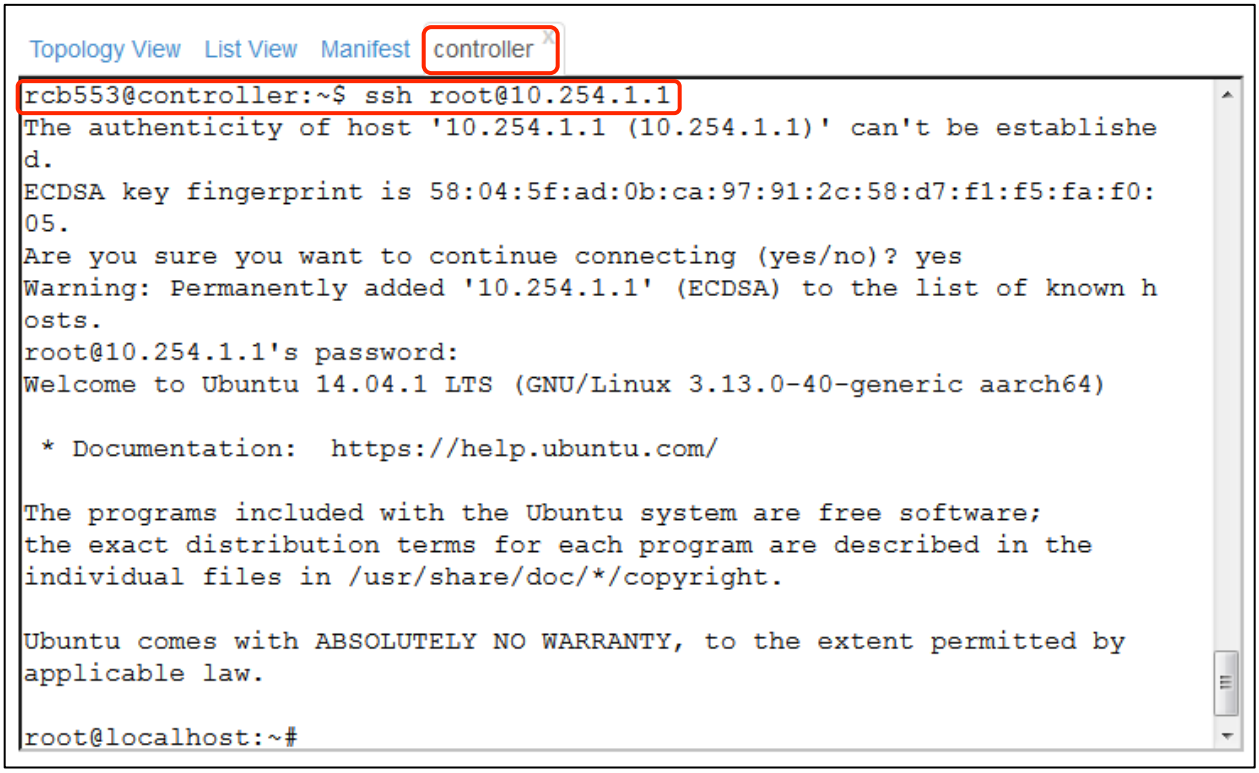
- Add ‘tun-data-net’ and ‘flat-data-net’ to the networking



- Once your instance is created, take note of the class A private IP.



- Use the 'controller' to SSH into the node.



- You can also use your linux-based terminal (instead of the CloudLab portal) as shown in the screenshot below.

```
}rbazan@ubuntu:/opt/cloudlab$ ssh -p 22 rcb553@ms0735.utah.cloudlab.us
Welcome to Ubuntu 14.10 (GNU/Linux 3.13.0-40-generic aarch64)

 * Documentation:  https://help.ubuntu.com/

  Get cloud support with Ubuntu Advantage Cloud Guest:
    http://www.ubuntu.com/business/services/cloud


Last login: Thu Apr  2 09:33:39 2015 from mu-020038.dhcp.missouri.edu
rcb553@controller:~$ ssh root@10.254.1.1
root@10.254.1.1's password:
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-40-generic aarch64)

 * Documentation:  https://help.ubuntu.com/
Last login: Thu Jan  1 00:14:26 1970 from 10.10.1.1
root@localhost:~#
```

Display network configuration and take a screenshot with the MAC address clearly visible for grading purposes.

Topology View | List View | Manifest | controller

```
root@localhost:~# ifconfig
eth0      Link encap:Ethernet  HWaddr fa:16:3e:ab:04:d2
          inet addr:10.254.1.1  Bcast:10.255.255.255  Mask:255.0.0.0
          inet6 addr: fe80::f816:3eff:feab:4d2/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1454  Metric:1
          RX packets:555 errors:0 dropped:0 overruns:0 frame:0
          TX packets:411 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:53129 (53.1 KB)  TX bytes:45501 (45.5 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:600 (600.0 B)  TX bytes:600 (600.0 B)

root@localhost:~#
```

3.4 OpenStack Services, Command-line Interface Interaction

- Prepare your environment. An .sh file will be available for your experiment.

  SSH to 'controller' node and change to root user

  $ sudo –i
  $ source /root/setup/admin-openrc.sh

3.4.1. NOVA

- List instances on your cloud

```
root@controller:~# nova list
+--------------------------------------+-----------+--------+------------+-------------+----------------------
---------------------------+
| ID                                   | Name      | Status | Task State | Power State | Networks
                           |
+--------------------------------------+-----------+--------+------------+-------------+----------------------
---------------------------+
| b4ce4c4f-7ed9-4db7-99f2-d7db56be1a20 | instance2 | ACTIVE | -          | Running     | tun-data-net=172.16.0.
4; flat-data-net=10.254.1.3 |
+--------------------------------------+-----------+--------+------------+-------------+----------------------
---------------------------+
```

- List available images

```
root@controller:~# nova image-list
+--------------------------------------+---------------------------------+--------+--------+
| ID                                   | Name                            | Status | Server |
+--------------------------------------+---------------------------------+--------+--------+
| 7219db0a-5cdc-4665-9241-9676f045dbe9 | initrd-3.13.0-40-arm64-generic  | ACTIVE |        |
| 4b662afd-ae2b-40b8-b47c-798b3d21e8a1 | trusty-server                   | ACTIVE |        |
| 6eb0f9b9-037a-4167-af19-78222fa68142 | vmlinuz-3.13.0-40-arm64-generic | ACTIVE |        |
+--------------------------------------+---------------------------------+--------+--------+
```

- List flavors (compute nodes)

```
root@controller:~# nova flavor-list
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| 1  | m1.tiny   | 512       | 1    | 0         |      | 1     | 1.0         | True      |
| 2  | m1.small  | 2048      | 20   | 0         |      | 1     | 1.0         | True      |
| 3  | m1.medium | 4096      | 40   | 0         |      | 2     | 1.0         | True      |
| 4  | m1.large  | 8192      | 80   | 0         |      | 4     | 1.0         | True      |
| 5  | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1.0         | True      |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
```

Additional NOVA commands

$ nova show InstanceName
$ nova pause NAME
$ nova pause volumeTwoImage
$ nova unpause NAME
$ nova suspend NAME
$ nova resume NAME
$ nova stop NAME
$ nova keypair-add test > test.pem

3.4.2. *NETWORKING (NEUTRON)*

- Display nets

```
root@controller:~# neutron net-list
+--------------------------------------+--------------+------------------------------------------------------
+
| id                                   | name         | subnets
|
+--------------------------------------+--------------+------------------------------------------------------
+
| 775b85ce-c8f6-4eb2-9e1f-3362a3c87d43 | tun-data-net  | 62f34c4c-25de-4fa8-ba6b-471241a19ede 172.16.0.0/12
|
| d56cc663-07d6-454c-9f41-6b44e3613cfe | ext-net       | a0a89565-9e44-4575-acac-1a1efffea3c2 128.110.152.0/22
|
| 0ce233fb-e540-47a3-869a-da70217061ba | flat-data-net | bddef010-cd34-4a80-bcd6-8d6312dd8abc 10.0.0.0/8
|
+--------------------------------------+--------------+------------------------------------------------------
```

- Display net detail information

```
root@controller:~# neutron net-show 0ce233fb-e540-47a3-869a-da70217061ba
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 0ce233fb-e540-47a3-869a-da70217061ba |
| name                      | flat-data-net                        |
| provider:network_type     | flat                                 |
| provider:physical_network | data                                 |
| provider:segmentation_id  |                                      |
| router:external           | False                                |
| shared                    | True                                 |
| status                    | ACTIVE                               |
| subnets                   | bddef010-cd34-4a80-bcd6-8d6312dd8abc |
| tenant_id                 | a8b108fe509048f58f2e30cea58975fe     |
+---------------------------+--------------------------------------+
```

- Display subnets

```
root@controller:~# neutron subnet-list
+--------------------------------------+------------------+------------------+--------------------------------
----------------------+
| id                                   | name             | cidr             | allocation_pools
                      |
+--------------------------------------+------------------+------------------+--------------------------------
----------------------+
| 62f34c4c-25de-4fa8-ba6b-471241a19ede | tun-data-subnet  | 172.16.0.0/12    | {"start": "172.16.0.2", "end": "
172.31.255.254"}      |
| a0a89565-9e44-4575-acac-1a1efffea3c2 | ext-subnet       | 128.110.152.0/22 | {"start": "128.110.155.136", "en
d": "128.110.155.136"} |
|                                      |                  |                  | {"start": "128.110.155.137", "en
d": "128.110.155.137"} |
|                                      |                  |                  | {"start": "128.110.155.138", "en
d": "128.110.155.138"} |
|                                      |                  |                  | {"start": "128.110.155.139", "en
d": "128.110.155.139"} |
| bddef010-cd34-4a80-bcd6-8d6312dd8abc | flat-data-subnet | 10.0.0.0/8       | {"start": "10.254.1.1", "end": "
10.254.254.254"}      |
+--------------------------------------+------------------+------------------+--------------------------------
----------------------+
```

Additional NEUTRON commands

$ neutron net-create NAME
$ neutron subnet-create NETWORK_NAME CIDR
$ neutron subnet-create my-network 10.0.0.0/29

3.4.3. *IDENTITY SERVICE (KEYSTONE)*

- Display user list

```
root@controller:~# keystone user-list
+----------------------------------+------------+---------+------------------------+
|                id                |    name    | enabled |         email          |
+----------------------------------+------------+---------+------------------------+
| 522d29a2c9cd41deb2d07df1b6a04020 |   admin    |   True  | rcb553@mail.missouri.edu |
| 0ff23d664243424ab557a339a2abd2af | ceilometer |   True  |                        |
| 1f2c8f346c0348cda4cb3e5d69e07c1d |   cinder   |   True  |                        |
| 6c763f54339a4a42bd08722b5ad2b4f6 |   glance   |   True  |                        |
| 78ad0aae83e64dc087b23cc1a1dc47d8 |    heat    |   True  |                        |
| 4792e9e629294135b8c0d831e1c24403 |  neutron   |   True  |                        |
| bb27547f97fd4f3387d8a08aa9339ed9 |    nova    |   True  |                        |
| 45a8ad0ec7874bf68307fd0d0b3b2f3d |   sahara   |   True  |                        |
| c4c43e01f88248218754e2cf6f255744 |   swift    |   True  |                        |
| e82f6703e750487bafed61e2e0e6b3f6 |   trove    |   True  |                        |
+----------------------------------+------------+---------+------------------------+
```

- Display services

```
root@controller:~# keystone service-list
+----------------------------------+------------+----------------+----------------------------------+
|                id                |    name    |      type      |           description            |
+----------------------------------+------------+----------------+----------------------------------+
| 6b59d7926129425f945f566a144cbd8b | ceilometer |    metering    |   OpenStack Telemetry Service    |
| 0ed411bd08a4473e985bf48e6a32fc07 |   cinder   |     volume     | OpenStack Block Storage Service  |
| 9398a32024c44739b69deed723152b7b |  cinderv2  |    volumev2    | OpenStack Block Storage Service  |
| 8d787753af9842f7a21b4e2ec0933171 |   glance   |     image      |     OpenStack Image Service      |
| d13f10cd875e4a3f99a0e68762dd133a |    heat    | orchestration  | OpenStack Orchestration Service  |
| 98b2d40769ab48348c299b9b808b8a45 | heat-cfn   | cloudformation | OpenStack Orchestration Service  |
| e521076460f14ce285f73d42d07efc16 |  keystone  |    identity    |    OpenStack Identity Service    |
| cbf6c9ad88614c3cb633618c5c0ecb98 |  neutron   |    network     |   OpenStack Networking Service   |
| 9ff0108658bc49a899f1222e9ec4a969 |    nova    |    compute     |    OpenStack Compute Service     |
| 50772b51d10d4aab9d802206c559cdfa |   sahara   | data_processing | OpenStack Data Processing Service |
| 96e278caaa754699ae676554a1d17667 |   swift    |  object-store  | OpenStack Object Storage Service |
| 8ea92d5a28ef4b8bb81a0222ff845a76 |   trove    |    database    |    OpenStack Database Service    |
+----------------------------------+------------+----------------+----------------------------------+
```

Additional KEYSTONE commands

$ keystone catalog
$ keystone user-create --name NAME --tenant-id TENANT \
  --pass PASSWORD --email EMAIL --enabled BOOL
$ keystone tenant-create --name NAME --description "DESCRIPTION" \
  --enabled BOOL

3.4.4. *IMAGE SERVICE (GLANCE)*

$ glance image-list

```
root@controller:~# glance image-list
+--------------------------------------+---------------------------------+-------------+------------------+----
--------+--------+
| ID                                   | Name                            | Disk Format | Container Format | Siz
e       | Status |
+--------------------------------------+---------------------------------+-------------+------------------+----
--------+--------+
| 7219db0a-5cdc-4665-9241-9676f045dbe9 | initrd-3.13.0-40-arm64-generic  | ari         | ari              | 231
61439   | active |
| 4b662afd-ae2b-40b8-b47c-798b3d21e8a1 | trusty-server                   | ami         | ami              | 107
3741824 | active |
| 6eb0f9b9-037a-4167-af19-78222fa68142 | vmlinuz-3.13.0-40-arm64-generic | aki         | aki              | 936
0864    | active |
+--------------------------------------+---------------------------------+-------------+------------------+----
--------+--------+
```

Additional GLANCE commands

$ glance image-delete IMAGE
$ glance image-show IMAGE
$ glance image-update IMAGE
$ glance image-create --name "cirros-threepart-kernel" \
 --disk-format aki --container-format aki --is-public False \
 --file ~/images/cirros-0.3.1~pre4-x86_64-vmlinuz

3.4.5. *BLOCK STORAGE (CINDER)*

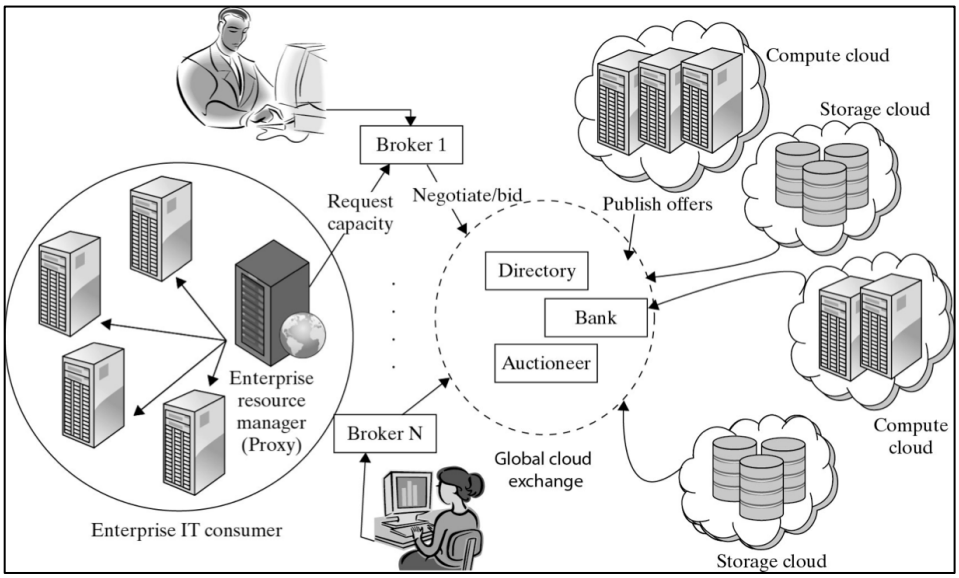
$ cinder create SIZE_IN_GB --display-name NAME
$ cinder create 1 --display-name MyFirstVolume
$ cinder list
$ nova volume-attach INSTANCE_ID VOLUME_ID auto
$ nova volume-attach MyVolumeInstance /dev/vdb auto

With above steps, you can see how your OpenStack environment is setup in your experiment. Next, we will study how to use this setup for Intercloud communications.

3.5. Intercloud Communication

On the basis of the below figure for multi-cloud resource brokering, we will create an Intercloud API to display deployed instances in your experiment that can be accessible from external cloud environments by using RESTful web services with Python and Flask.

We will start by creating a web application, then we will convert it to a RESTful service. The clients of the web service will ask the service to display the various available instances information. That information will be JSON data generated with jsonify function.



3.5.1. RESTful API with Python and Flask

Setting up your architecture design for web services and web APIs.

Create virtual environment

$ python –V
$ sudo apt-get install python-virtualenv
$ mkdir cloud-api
$ cd cloud-api
$ virtualenv flask
$ flask/bin/pip install flask
$ flask/bin/pip install flask-httpauth
$ source flask/bin/activate

```
rcb553@controller:~/cloud-api$ ls
flask
rcb553@controller:~/cloud-api$ source flask/bin/activate
(flask)rcb553@controller:~/cloud-api$
```

Create Intercloud API – code snippet

$vim intercloud.py

```
#flask/bin/python
from flask import Flask, jsonify, make_response, abort
from flask.ext.httpauth import HTTPBasicAuth

auth = HTTPBasicAuth()

@auth.get_password
def get_password(username):
   if username == 'clouduser':
      return 'EasyPassword15'
   return None

@auth.error_handler
def unauthorized():
   return make_response(jsonify({'error': 'Unauthorized access'}), 403)
app = Flask(__name__)

@app.errorhandler(404)
def not_found(error):
   return make_response(jsonify({'error': 'Not found'}), 403)

@auth.login_required
def get_tasks():
   return jsonify({'tasks': [make_public_task(task) for task in tasks]})

@app.route('/list', methods = ['GET'])
@auth.login_required
def get_list():
   import subprocess
   import os
   my_env = os.environ.copy()
   my_env['OS_TENANT_NAME'] = ' '
   my_env['OS_USERNAME'] = ' '
   my_env['OS_PASSWORD'] = ' '
   my_env['OS_AUTH_URL'] = ' '
   p = subprocess.Popen(['nova', 'list'], env=my_env, stdout=subprocess.PIPE)
   output, error = p.communicate()
   return jsonify({'output': output})

if __name__ == '__main__':
      app.run(debug=True,host="0.0.0.0",port=8090)
```

Basic secure user authentication

Error handler function defined

Basic secure user authentication

Fill out the blank spaces with your environment information
$sudo cat /root/setup/admin-openrc.sh

Subprocess with 'nova list' command

Port 8090 configured

Run the application

$ python intercloud.py

```
(flask)rcb553@controller:~/cloud-api$ python intercloud.py
 * Running on http://0.0.0.0:8090/ (Press CTRL+C to quit)
 * Restarting with stat
```

3.6.2. Accessing the Intercloud Service

With the intercloud.py script running, our Intercloud API is ready to be accessed from an external cloud service. You can use either Option 1 (curl) or Option 2 (RESTclient for browser) at the external cloud client side or at a user's private cloud side.

Option 1. Using curl command.

Login to the "controller" and note the public IP associated to the external bridge.

```
br-ex     Link encap:Ethernet  HWaddr 14:58:d0:57:0f:b2
          inet addr:128.110.153.53  Bcast:128.110.155.255  Mask:255.255.252.0
          inet6 addr: fe80::2cc5:1bff:fe11:ea26/64 Scope:Link
          UP BROADCAST RUNNING  MTU:1500  Metric:1
          RX packets:1509373 errors:0 dropped:0 overruns:0 frame:0
          TX packets:898091 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:384391610 (384.3 MB)  TX bytes:181891046 (181.8 MB)
```

From a linux-based system (our systems or an EC2 instance in Amazon) input the following command to display instances in our cloud environment:

$ curl -u clouduser:EasyPassword15 -i http://[IP]:8090/list

```
rbazan@ubuntu:/opt/cloudlab$ curl -u clouduser:EasyPassword15 -i http://128.110.153.53:8090/list
HTTP/1.0 200 OK
Content-Type: application/json
Content-Length: 728
Server: Werkzeug/0.10.4 Python/2.7.8
Date: Thu, 02 Apr 2015 20:05:51 GMT

{
  "output": "+--------------------------------------+-----------+--------+------------+-------------+----------
----------------------------------------+\n| ID                                   | Name      | Status | Task
State | Power State | Networks                                 |\n+--------------------------------------
---+-----------+--------+------------+-------------+----------------------------------------------------+\n| bc8
8cc80-787e-47f4-b2df-7cf947ea1118 | instance1 | ACTIVE | -          | Running     | tun-data-net=172.16.0.2; fl
at-data-net=10.254.1.1 |\n+--------------------------------------+-----------+--------+------------+-----------
--+----------------------------------------------------+\n"
```

An "Unauthorized access" message is displayed whenever we omit credentials

$ curl -i http://[IP]:8090/list

```
}rbazan@ubuntu:/opt/cloudlab$ curl -i http://128.110.153.53:8090/list
HTTP/1.0 401 UNAUTHORIZED
Content-Type: application/json
Content-Length: 36
WWW-Authenticate: Basic realm="Authentication Required"
Server: Werkzeug/0.10.4 Python/2.7.8
Date: Thu, 02 Apr 2015 20:08:31 GMT

{
  "error": "Unauthorized access"
```

And a “Not Found” error is displayed whenever we typed an invalid address

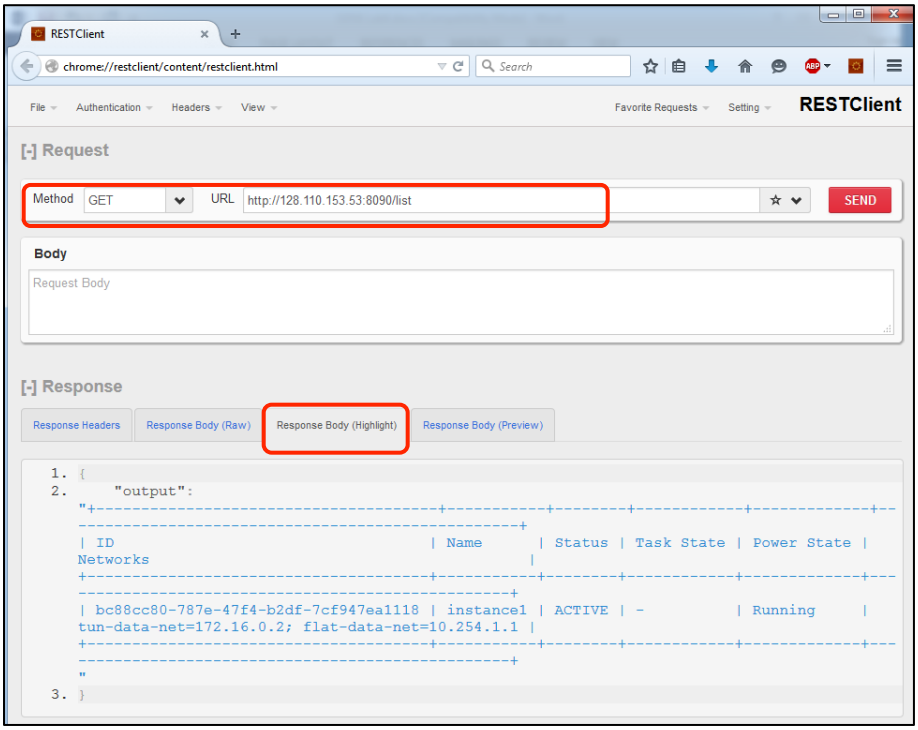
```
}rbazan@ubuntu:/opt/cloudlab$ curl -u clouduser:EasyPassword15 -i http://128.110.153.53:8090/invalid_address
HTTP/1.0 404 NOT FOUND
Content-Type: application/json
Content-Length: 26
Server: Werkzeug/0.10.4 Python/2.7.8
Date: Thu, 02 Apr 2015 20:09:05 GMT

{
  "error": "Not found"
```

At the cloud environment side, the controller node handles 200, 401 and 403 messages accordingly.

```
(flask)rcb553@controller:~/cloud-api$ python intercloud.py
 * Running on http://0.0.0.0:8090/ (Press CTRL+C to quit)
 * Restarting with stat
128.206.20.38 - - [02/Apr/2015 14:08:21] "GET /list HTTP/1.1" 200 -
128.206.20.38 - - [02/Apr/2015 14:08:31] "GET /list HTTP/1.1" 403 -
128.206.20.38 - - [02/Apr/2015 14:09:05] "GET /invalid_address HTTP/1.1" 404 -
```

Option 2. Using “Advanced REST client” plugin for Chrome or “RESTclient” for Firefox.



4. What to turn in for Grading?

1. Provide screenshot of the “Network Topology” after a new instance is created. Explain the graph.

2. Provide screenshot of the ‘controller’ node with the MAC Address clearly displayed.

3. List in detail the resources available for the deployed cloud infrastructure (vCPUs, RAM, Floating IPs, Security Groups, and Volumes)

4. List the necessary changes in the profile file to add an extra compute node, and submit a revised RSpec.

5. Extend the Intercloud API to display user list (KEYSTONE) as:

curl -u clouduser:EasyPassword15 -i http://[IP]:8090/**list_user**

Provide screenshot of the output.

6. By using your AWS instance setup in AWS Lab-2, you should write a web service client (use any language of your preference) to request and display the cloud information available in the JSON file in a simple web site. Include the Amazon DNS link and the code in your submission report.