Capstone Project Report Title:

OpenStack- Elastic Cloud Service Orchestration with Openstack Orchestrate Elastic Cloud Orchestration for Compute/Storage

Submitted By:

Fahaam Alam Hashmi

Under Supervision of:

Muhammad Durrani

Duration:

September 2016 - March 2017

ACKNOWLEDGEMENT

This final year project was possible because of the will and wish of Almighty ALLAH and we are grateful to Him. I express our sincere thanks to our supervisor Mr Mouhammad Durrani, for his encouraging comments and useful suggestions, which helped me in completing the project work, in time. I also feel great pleasure in acknowledging our heartiest gratitude to Mr Shah Nawaz Mir. This work could never have been completed without his perspicuous instruction, consistent help and invaluable support.

Abstract

There is undoubtedly no mendacity in the fact that the benefits of Cloud Computing has earned great attention of IT Industry today. Capability of providing on demand resources, contrast with elasticity and metered services in shared pool of hardware and software are some of the appealing characteristics of Cloud Computing. The technology become more advantageous, when it is not only open source, but also with global collaboration. Openstack is the service that is playing this role, with being open source. Openstacks increasing versatilities allures the organizations to build their own public and private cloud with openstack. Furthermore the provision of metering over cloud services known as ceilometer plays a fundamental role to support applications like ecommerce over cloud computing services. In this project we have two parts after implementation of openstack cloud. In the first part we used openstack ceilometer service, for the purpose of knowing the usage of resources being used by different tenants (users), by metering different instances in different period intervals. Provided records of multiple tenants, with monitoring different parameters. In the second part of the project elastic cloud computing service to tenant is provided through openstack heat services. Defining different policies in such a way to assure availability of resources provided to tenant, during peak usage time of resources.

TABLE OF CONTENTS

CHAPTER 1 OPENSTACK INTRODUCTION

1.1 Introduction to OpenStack

OpenStack provides Infrastructure as a Service solution. IaaS (Infrastructure as a Service solution) is the basic cloud computing service model, where virtual machines, load balancers, raw block storage, firewalls and networking services are provided on one platform, where everything interrelated. It provides control to compute, storage and network for a large pool. Compute Networking and Storage are the three major parts open stack is divided into. A collaboration of cloud computing service and developer private or public cloud computing, which is totally open source. Help's to manage large pools of public or private cloud data center on one screen in faster and cost effective way.

Open stack is the cloud computing self-service, this means it is a computing capability that provides Compute, network and servers services without any human interaction, which used to be achieved through large servers to install in data center, not only requires lots of space and power, but also requires more ram and space as increase in users. In cloud computing these services can are provided without any hardware setup. Furthermore, these resources are provided with the failover not only at the hardware level but are also provided on the application level. These services can be accessed through standard platforms. Multi-consumers are pooled through multi-tenant model by computing resources. The consumer has no idea where these resources are coming from, though these resources have elasticity can be scaled out or scale in. Moreover, they can be controlled or accessed worldwide anywhere on the internet. It has multi-tenant sharing, that is multiple tenants will be using one hardware as resources without any interfering each other. Telemetry is also one the characteristics of OpenStack that is used for monitoring and metering purposes not only by the service provider but also by the consumer.

Open stack software manages the large pool of resources such as compute, storage and network, everything through one dashboard. This is made possible through OpenStack API. It is the best way to reduce the cost. Buying storage devices, building up networks for those devices not only need a large place to set up the system but requires a large capital in contrast with. Open stack solves the whole of the problem, not only ease the way to setup everything but also provides the solution with less effort and no infrastructure.

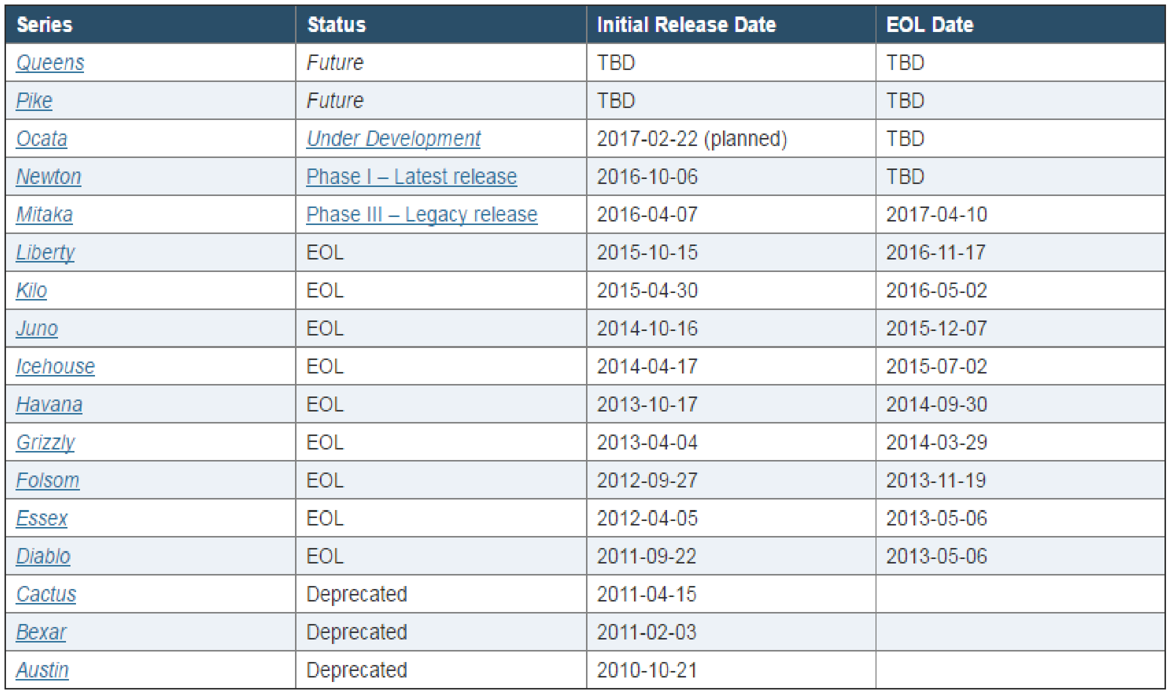
OpenStack is open source infrastructure that is available for everyone for free. The software consists of all components interrelated with each other as a whole data center. These components can be managed through graphical user interface (dashboard) or through the command line or through a RESTful API.

1.2 OpenStack History

OpenStack idea was built by the idea of AMAZON EC2 services though it was a cloud marketing that is developed for hosting, charging on an hourly basis. The idea behind this cloud infrastructure was work with applications that avoid hardware failure. In 2010 open stack the goal was the same as Amazon, but it was 2016 the idea gone far beyond. Clients were getting the same sort of performance as they built their own compute without any hardware setup.

The main aim of the OpenStack was to make huge Open Source Cloud Computing platform that will fulfill the needs of public and private cloud regardless of the size.

OpenStack project that was started in 2010 by Rackspace Hosting and NASA together, when NASA wanted to create their own cloud known as Nebula. The reason to create Nebula is to provide compute services to different departments with one location under one IT department. On other hands, Rackspace and Slice Host made open source cloud service to keep cloud files. The idea behind this is to combine all the technology and work together. Therefore NASA and Rackspace created a platform known as OpenStack. They took the Storage part from Rackspace and Nova Compute part from NASA, as the code for Nova Compute can scale way larger. Then furthermore companies such as HP, Dell, Redhat, AT&T and Intel also get into contributed further for investment. OpenStack every 6month releases the new version of the previous versions are stated in the following fig.1.1



Series	Status	Initial Release Date	EOL Date
Queens	Future	TBD	TBD
Pike	Future	TBD	TBD
Ocata	Under Development	2017-02-22 (planned)	TBD
Newton	Phase I – Latest release	2016-10-06	TBD
Mitaka	Phase III – Legacy release	2016-04-07	2017-04-10
Liberty	EOL	2015-10-15	2016-11-17
Kilo	EOL	2015-04-30	2016-05-02
Juno	EOL	2014-10-16	2015-12-07
Icehouse	EOL	2014-04-17	2015-07-02
Havana	EOL	2013-10-17	2014-09-30
Grizzly	EOL	2013-04-04	2014-03-29
Folsom	EOL	2012-09-27	2013-11-19
Essex	EOL	2012-04-05	2013-05-06
Diablo	EOL	2011-09-22	2013-05-06
Cactus	Deprecated	2011-04-15	
Bexar	Deprecated	2011-02-03	
Austin	Deprecated	2010-10-21	

Fig.1.1 (OpenStack releases)

OpenStack provides multiple services such as Nova, Swift, Glance, Keystone, Horizon Neutron, Cinder, Ceilometer and Heat are some of the main services that are provided by OpenStack.

1.3 Involvement of Companies in OpenStack

OpenStack is one of the most recognized technology in IT world today. About 8 huge names of industries such as AT&T, Ubuntu, HP, IBM, Intel, Rackspace, Redhat and Suse are platinum members in open stack project. These companies are sponsoring for empowering and promoting OpenStack community and software. Furthermore, there are about 24 gold members who are gold members of in OpenStack project who also take part in the funding of open stack project. As well as provide strategic alignment to the project. Infrastructure donors are also the developers of

OpenStack, they provided resources such as cloud for OpenStack. Corporate Sponsors are the additional support to this project that also helps in sponsoring, empowering and promotions. It was the interest of the companies that open stack was able to release five software in less than two years, with hundreds of contributors. The community meets every six months. During development phase the community collects the OpenStack designs, in order to provide facility to developers in assembling and to work on.

1.4 Compatibility of OpenStack with other Cloud Computing

The open stack was not compatible with every cloud computing APIs. There is some compatibility that is driven by the members of the project. EC2 API that provides compatibility with Amazon EC2 and GCE API project that provides compatibility with GOOGLE Compute service. Amazon EC2 helped users to have their own virtual system to run different other application or web services, whereas GCE API is a part of Google Cloud Platform that helps in running Google search engine and other services by Google. Compatibility with other compute services is one of the plus points of OpenStack.

In Amazon EC2, Google Cloud and Microsoft Azure. OpenStack is one of the more supportive services to work on. Since OpenStack is a platform where all companies are working together, with collaboration. Everything is available and open source. OpenStack is more compatible with. As well as release every six months provides OpenStack improvement, growth and options to do better.

CHAPTER 2 OPENSTACK ARCHITECTURE

2.1 How OpenStack Works

OpenStack provides multiple services such as Nova, Swift, Glance, Keystone, Horizon Neutron, Cinder, Ceilometer and Heat services. These are some of the main services of OpenStack. All these services are interconnected through APIs and play their role accordingly.

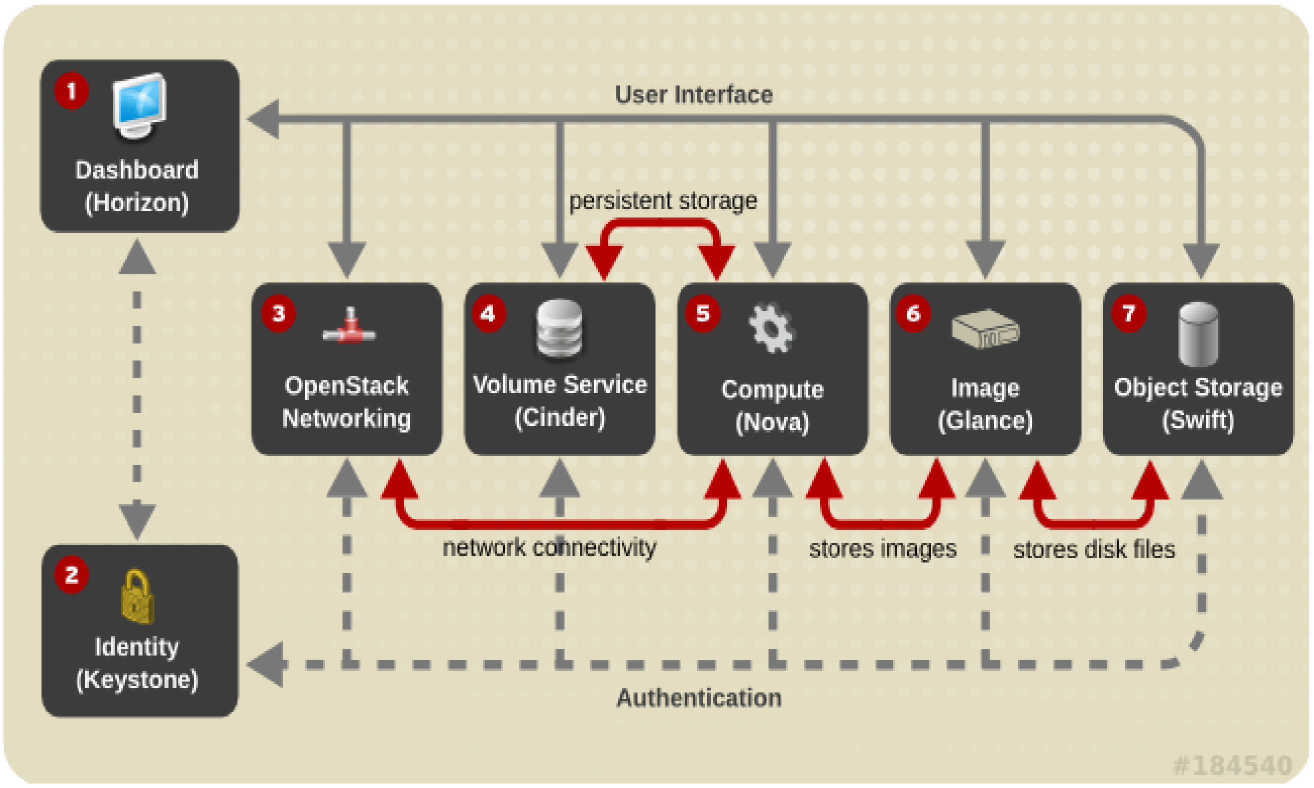


FIG 2.1 (OpenStack Architecture)

2.1.1 Horizon is the user interface or dashboard that is used to control every other service. It is the web interface. User access OpenStack services through the web interface. All OpenStack services meet up at the horizon. Every user gets the services according to their authentication. It also keeps information in cache until the service is running. It is the service that interacts all open stack services to other services

2.1.2 Keystone is the authentication part, known as identity service as well. It is the username and password that is given to every user, when this authentication is done the keystone provides the token to the user that consist of the services that are been allowed to the user to access, such as Glance, Keystone, and Swift. These all are connected through rest APIs. After passing through Keystone.

2.1.3 Nova responds user request into the virtual machine. Nova interacts with many other OpenStack services such as keystone for authentication, for web interfaces it communicates with the horizon and for image glance is there. Nova consists of different layers. Nova API accepts the user request and responds accordingly. It supports Open Stack API, EC-2 API and admin API for privilege administration work. Also initiates orchestration activities, running instance through hypervisor API is one of it. Nova Compute API converts the user request into the virtual machine. It accepts the request from the queue and perform actions accordingly and updating its database, with that there is nova-volume that manages creating, attaching and de-attaching of volumes. Nova network is used for networking task, it accepts the networking tasks such as changing bridging table or changing IP rules this service is replaced by the neutron. Nova scheduler decides what instances should run on which host, after accepting from the queue. A queue is a central place of communication, for all nova services. It works over rabbit MQ functionality. Nova database is used to store built time or runtime state for cloud infrastructure, these can be instance type that can be used and instance type that is used and network types in the projects.

2.1.4 Swift is a block storage service by OpenStack. This service is dominated by Rackspace in OpenStack, though this service is little changed due to the emergence of mobile web application and due to the emergence of software define storage, where huge storage are built with the involvement of hardware. Swift is the object storage system, with the combination of redundancy, availability, throughput and capacity. Swift has no delays with the fast transition of data, a large number of reads of storage data with the simultaneous transaction at the same time and can scale up to extremely large scale. Swift does not need high-level hardware. It can be deployed on commodity hardware and gain high performance. Swift is a distributed system that can easily be scale out, with the contrast of failover without compromising data information. As Nova, it has also consisted of different components. The swift proxy server accepts the received request through OpenStack object API. It accepts the modification, creation or uploads the file request. It also provides files or container listing to the web browser. The proxy server consists of accounting service that is the accounts defined for storage service. The container service is used for mapping of files and folder to the object storage, they have basically defined nodes to where data should go. The object storage manages the object storage that manages actual data files. Moreover for large data it also it runs housekeeping tests as well that can be replication service that can assure the availability, redundancy of the storage service. The proxy is the only part of the swift that deals with the external client as it accepts the requests. Furthermore, it also does external actions such as restore data to the website, data while online game or data retrieving from the backup. These all comes in a form of incoming request, where swifts check the origin of the request. Then responds the request according to the required action allowed for the user. These all request are based on HTTP request where Get, Put, Post, Delete and Head are the main requests in swift storage.

2.1.5 Glance is also storage service in OpenStack, but it keeps the images of the virtual disk's storage. It is image service, the main function of the glance is to provide discovery, registration and delivery service to server or disk image. The images those are stored in the service are used as the template, a number of backups can be made from these stored images. These images can be streamed for further more compute instances for the users. Users can create their own image by modifying the existing image. Furthermore this snapshot storage in glance use for quick backup restarting the virtual machine. Glance provide multiple of services such as image uploading, retrieving disk images. Not only this listing and querying virtual disk images and set virtual disks permissions are also included. The Glance service consists of three things glance API, glance registry, and glance database.

The image retrieval and discovery call for image API, all are accepted by glance API. Glance-registry is to store processes and retrieve metadata about the type or size of the images. Glance database store the metadata images or it can also store in swift storage.

2.1.6 Cinder services provide persistence block storage in OpenStack to compute instance. It is responsible for server attaching and de-attaching with block devices. They are logical the volumes, converted from raw volume, mounted on the virtual machines. They are the volumes or snapshot of volumes that can be created or deleted. Clones can be form from these volumes. Computer instances can be attached or detached, but only one instance can be attached at one time. This also works with the glance where we can copy images to the volume. Cinder also allows viewing the statistics of the volume.

2.1.7 Neutron is the network service in open stack. It provides networking between the interface devices managed by other OpenStack services such as nova. It permits the user to get connected to the server. It is very advantageous to the user to get commodity gear or vendor supported equipment through pluggable backend architecture. The extension allows additional network services, software or hardware to be integrated. Routers virtual switches and software defined networking controller are some of the features of the neutron, where router provides the gateway and virtual switches refer to the ports. These are instances or network services connected to the network consisting of mac address and IP address of interfaces plugged into them. Furthermore private and floating IP addresses are the address those are assigned to the instance. The private network that is dedicated to the tenant can only get private IP addresses. This isolates one tenant from another tenant. Neutron services also provide services from layer 1 to layer 3.As well as the load balancer. Neutron consist of the several services, the quantum server accepts API request and then route to the appropriate plugin. Quantum agents are used for plugging in and plugging out the ports. Queuing database helps to keep the record the status of the ports that are plugged in. Neutron manages the network and IP addresses. It makes sure that will not be the bottleneck in the cloud deployment.

.

2.1.8 Ceilometer is the monitoring service that is used to monitor other services that keep's the record of the usage or utilization. Ceilometer can be used for billing as well as alarm purposes both for the service provider as well as for the user. It collects the data by notifications from the services.

It is a part of billing model that consist of metering, rating, and billing.

Metering consist of continuous monitoring threshold, warnings and rate of collection, the central location of the data, and tenant details.

Rating is a process after the monitoring is done and utilization of data and values are achieved. It is a decision how billing is to done, prepaid , postpaid, discount , rate plan everything related with the result of output by monitoring and rating according to what is been monitored.

Billing is the final part where the transaction is decided and is decided how it will be charged with the contrast of service level agreement.

Ceilometer is just a part of the metering model, but it is the most important part. It finds the right query in Ceilometer API to extract the right information you need from monitoring. Metering is the process of gathering information about how much and what can be billed. It aims a point of the billing system to get all measurements they need to establish in customer billing. It not only provides the efficient collection of metering data in terms of CPU and Network but also helps in alarming processes. It allows to meter components directly or changing some of the components. Data can be gathered by monitoring components and service or by pulling infrastructure.

Monitoring is done through three particular aspects of resource usage Cumulative, Delta, and Gauge. Cumulative is monotonically increasing, whereas delta in interpreted as change from previous value and gauge standalone value relating only to the current duration

2.1.9 Heat is the main orchestration project. Orchestration engine launches multiple cloud applications and that orchestration engine is implemented by Heat. This application acts as code in text files. Heat provides native rest API as well and clouds information compatible query API. The orchestration is for infrastructure. Heat also provides compatibility with AWS API in template format. The orchestration is for infrastructure essentially what configuration management is for the software application. Heat is a declarative model that represents the infrastructure sources and relationship between them directly. Heat then perform a sequence of action to bring reality in line with your model. The model takes the form of heat template and the resulting template of infrastructure template is known as the stack. Orchestration allows you to allow you infrastructure as code. Therefore you can save your templates as version controlled system such as get to track changes, you update the stack with new template changes and then heat performs the necessary actions. Heat actually sits between the user and core API services. We can also interact with heat through the dashboard. The heat template describes the structure in the text file for the cloud. It is a readable and writable file. The infrastructure resources that can be described include servers, floating IPs, volumes, security groups, and users.

Heat is also integrated with ceilometer for auto-scaling services. Relationships between resources specify templates. This lets heat to interact with OpenStack API. Heat manages whole application processes. Changing the template and updating it on the stack, heat knows how to manage those changes. It also deletes all the resources when you are finished with the application.

Chapter 3 Basic Implementation of OpenStack

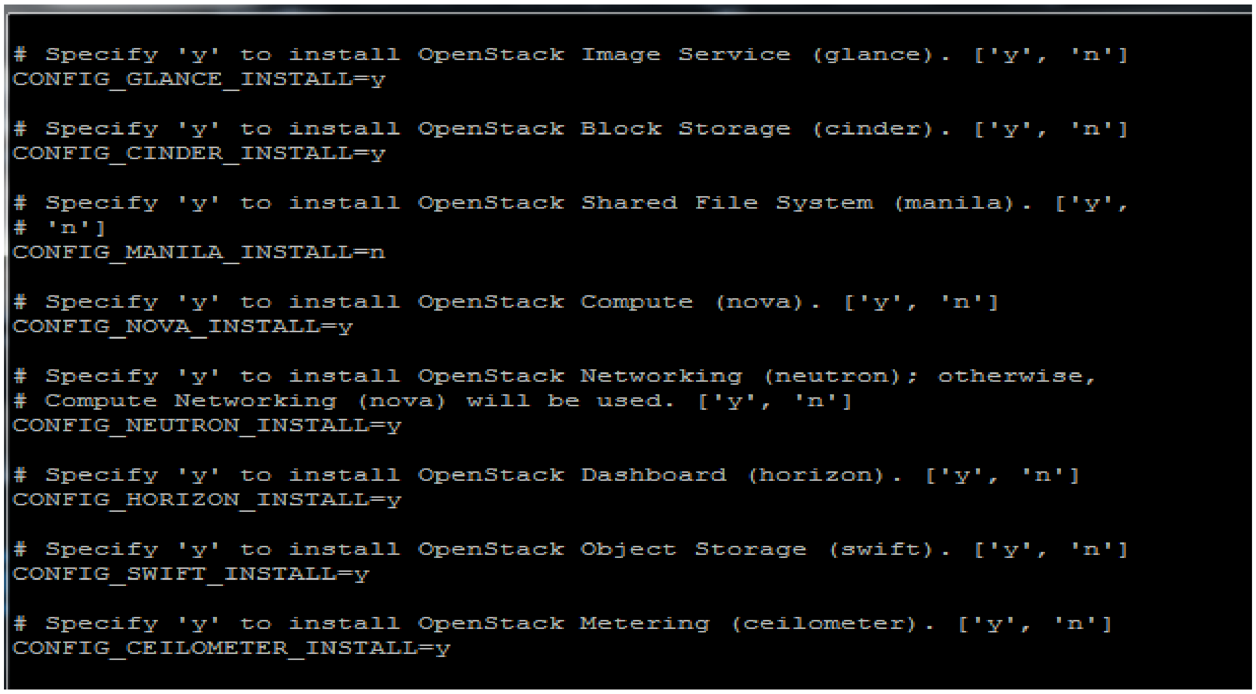
3.1 Basic Installation

In this project Open stack was installed through pack stack that is a repository module, help to implement various parts of OpenStack. **It** represents a utility which facilitates the deployment on multiple nodes for different components of **OpenStack** via **SSH**. This requires multiple pre-installed servers. In our case it is CentOS. There are two ways to work with packsack. As follows.

Packstack Interactively: Interactive way is the user commands are run at same time as he gives the command. The user enters the command

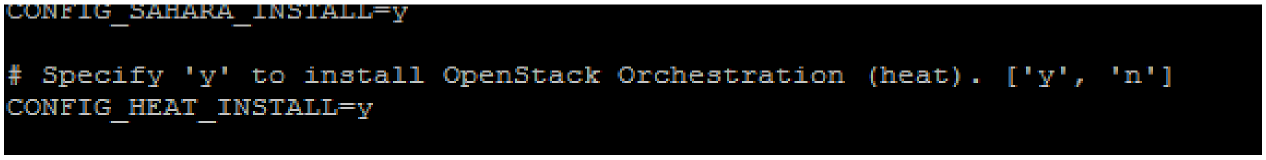
Packstack Non-Interactively: the Non-Interactive way is the way when the user updates the file known as answer file. Enables and customize the options in the answer file, that customize file is run for OpenStack.

In this project, Non interactive is used to install pack stack. Answer file will be applied on our controller node, as all service will work in controller node. The answer file is a file that pack stack takes in order to take decisions that we put in the script that we want to setup in OpenStack. Services will be enabled such as Glance, Cinder, Nova, Horizon, Swift, Ceilometer, and Heat as shown in FIG 3.1 and FIG 3.2



```
# Specify 'y' to install OpenStack Image Service (glance). ['y', 'n']
CONFIG_GLANCE_INSTALL=y

# Specify 'y' to install OpenStack Block Storage (cinder). ['y', 'n']
CONFIG_CINDER_INSTALL=y

# Specify 'y' to install OpenStack Shared File System (manila). ['y',
# 'n']
CONFIG_MANILA_INSTALL=n

# Specify 'y' to install OpenStack Compute (nova). ['y', 'n']
CONFIG_NOVA_INSTALL=y

# Specify 'y' to install OpenStack Networking (neutron); otherwise,
# Compute Networking (nova) will be used. ['y', 'n']
CONFIG_NEUTRON_INSTALL=y

# Specify 'y' to install OpenStack Dashboard (horizon). ['y', 'n']
CONFIG_HORIZON_INSTALL=y

# Specify 'y' to install OpenStack Object Storage (swift). ['y', 'n']
CONFIG_SWIFT_INSTALL=y

# Specify 'y' to install OpenStack Metering (ceilometer). ['y', 'n']
CONFIG_CEILOMETER_INSTALL=y
```

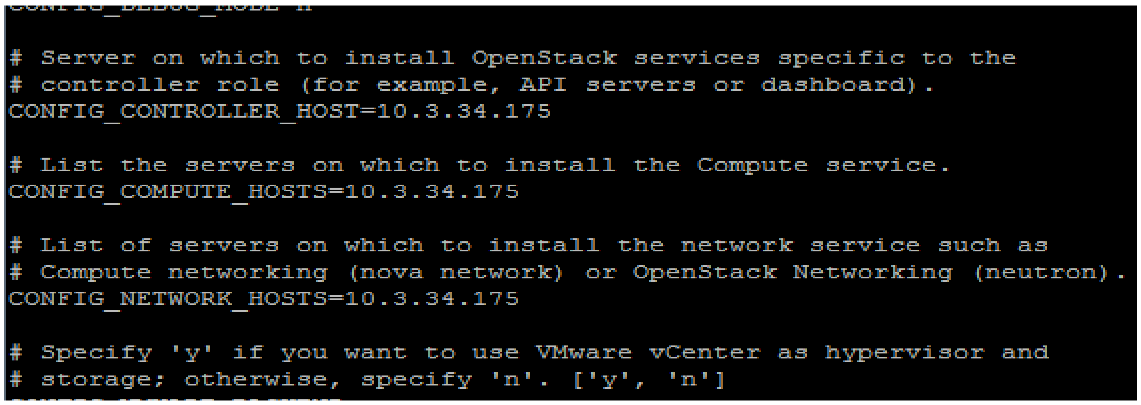
FIG.3.1 (Answer File with services)



```
CONFIG_SAHARA_INSTALL=y

# Specify 'y' to install OpenStack Orchestration (heat). ['y', 'n']
CONFIG_HEAT_INSTALL=y
```

FIG.3.2 (Answer File with heat services enabled)

Answer file also consists of IP addresses of the three main nodes, Compute, Controller, and Neutron. Compute node, in this project the three nodes are on one single node as shown in FIG.3.3. All these nodes should be interconnected.



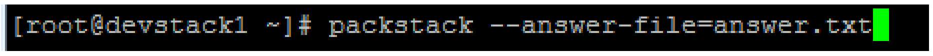
```
# Server on which to install OpenStack services specific to the
# controller role (for example, API servers or dashboard).
CONFIG_CONTROLLER_HOST=10.3.34.175

# List the servers on which to install the Compute service.
CONFIG_COMPUTE_HOSTS=10.3.34.175

# List of servers on which to install the network service such as
# Compute networking (nova network) or OpenStack Networking (neutron).
CONFIG_NETWORK_HOSTS=10.3.34.175

# Specify 'y' if you want to use VMware vCenter as hypervisor and
# storage; otherwise, specify 'n'. ['y', 'n']
```

FIG.3.3(Answer File with all IP address)

Moreover, the answer file also consists of keystone authentication user and password in plain text as well as the token that is discussed in the previous chapter as well. This token allows the user to the services that user is allowed to get access.

After Answer file is modified the file is run through pack stack through the following in FIG3.4

.



```
[root@devstack1 ~]# packstack --answer-file=answer.txt
```

FIG.3.4 (Run Answer File over pack stack)

Then access the dashboard through GUI as it is successfully installed after as shown in FIG 3.5

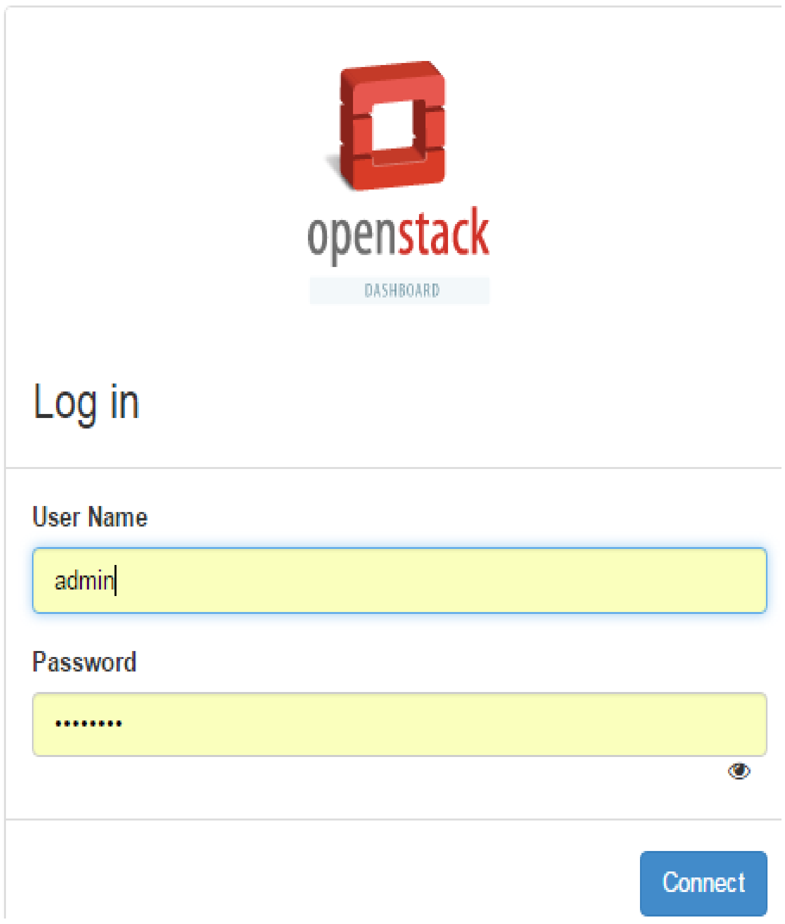
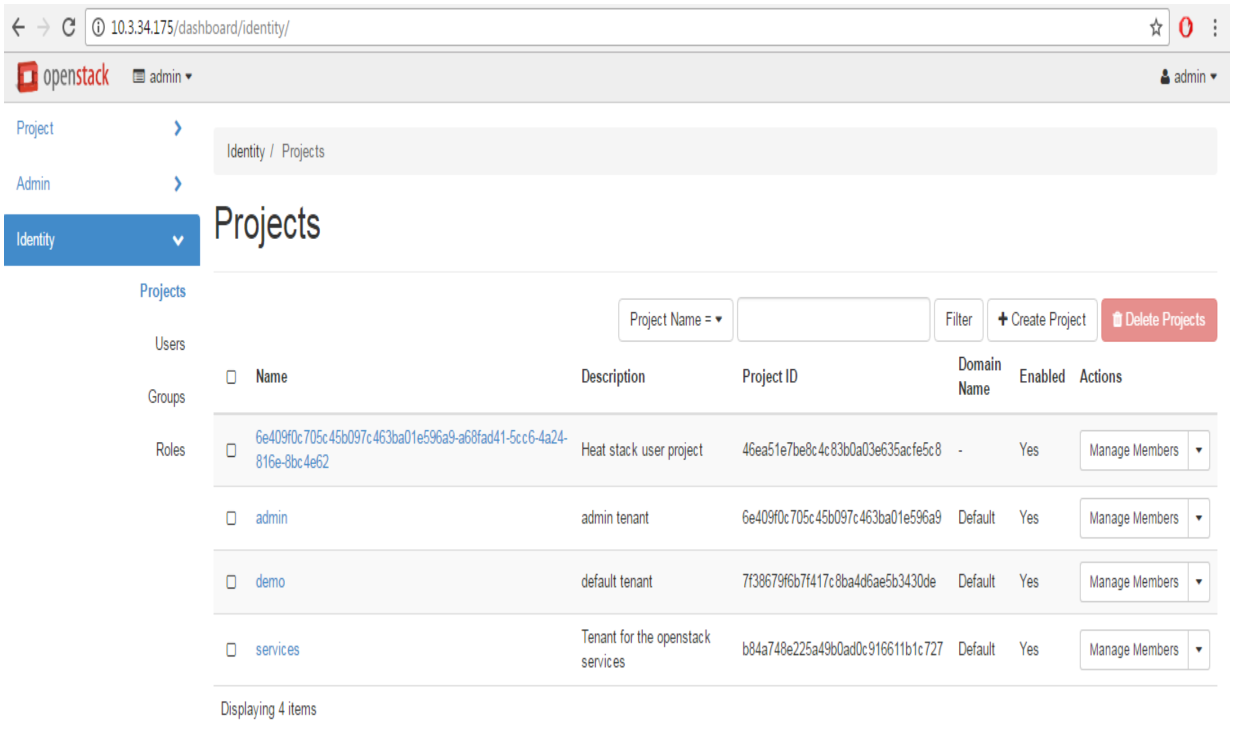


FIG.3.5 (User Authentication by open stack (Keystone))

As discussed before keystone plays the authentication role, known as identity service as well. It ask for username and password when this authentication is done the keystone provides the token to the user that consist of the services that are been allowed to the user to access

3.2 Implantation Projects and Other services



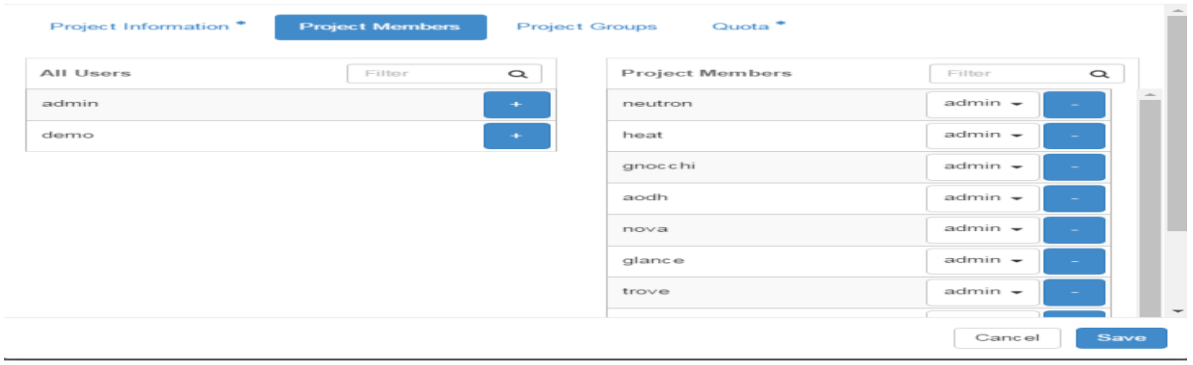
Projects

Name	Description	Project ID	Domain Name	Enabled	Actions
6e409f0c705c45b097c463ba01e596a9-a68fad41-5cc6-4a24-816e-8bc4e62	Heat stack user project	46ea51e7be8c4c83b0a03e635acfe5c8	-	Yes	Manage Members
admin	admin tenant	6e409f0c705c45b097c463ba01e596a9	Default	Yes	Manage Members
demo	default tenant	7f38679f6b7f417c8ba4d6ae5b3430de	Default	Yes	Manage Members
services	Tenant for the openstack services	b84a748e225a49b0ad0c916611b1c727	Default	Yes	Manage Members

Displaying 4 items

FIG.3.6 (Different projects of OpenStack)

After getting access we can see different projects. OpenStack can deal with different project at a time. Every project has different members as shown is FIG 3.7.A member can be a part of two different projects.

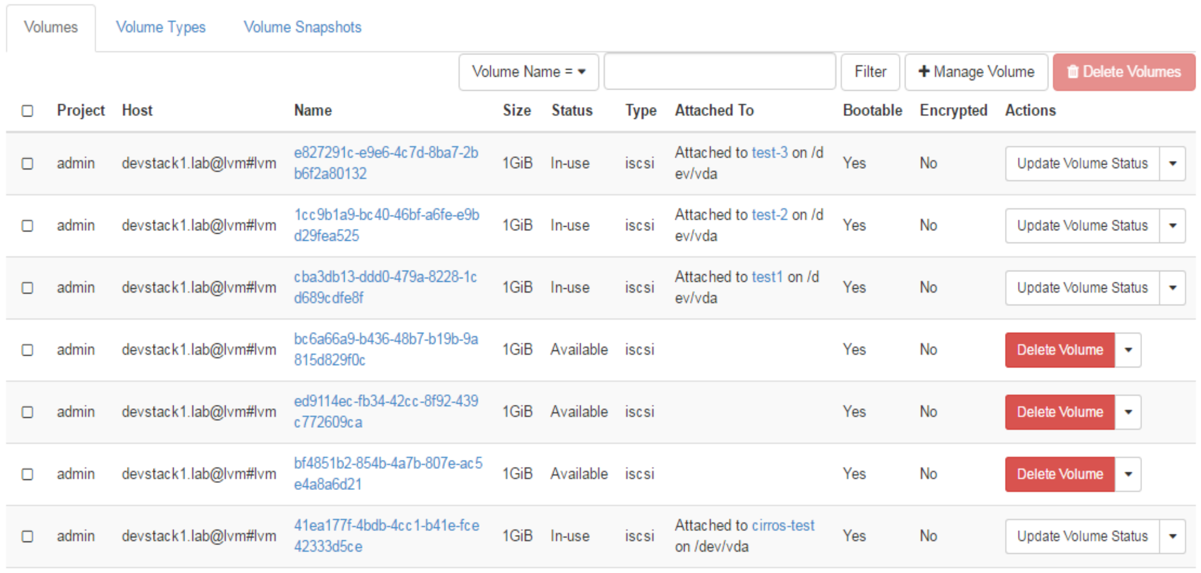


All Users
admin
demo

Project Members	Role
neutron	admin
heat	admin
gnocchi	admin
aodh	admin
nova	admin
glance	admin
trove	admin

FIG.3.7 (Project Members)

The cinder volume is the block storage that is provided to the user related to different projects. This is a logical storage to every user that can be attached or detached to the user. This memory can be reused for another user as user finished his work and memory is deleted. The user can be given different memory chunk according to the use.

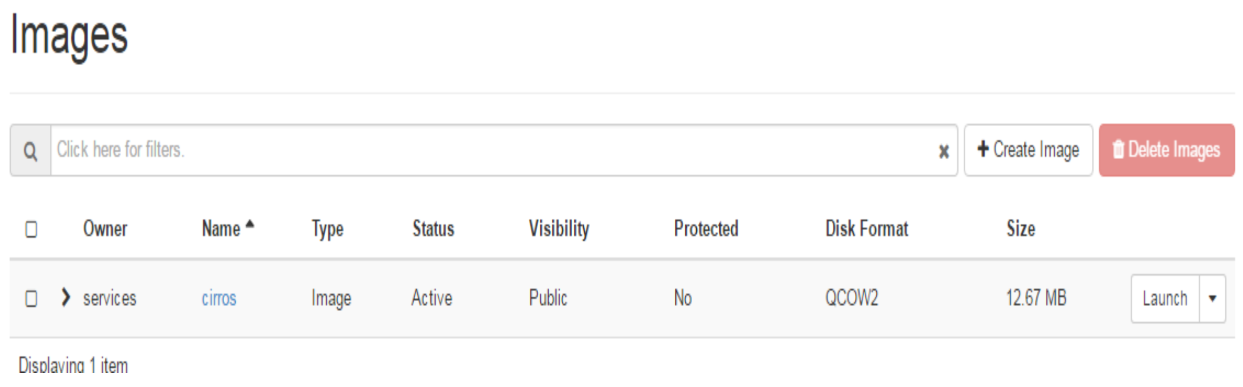


Project	Host	Name	Size	Status	Type	Attached To	Bootable	Encrypted	Actions
admin	devstack1.lab@lvm#lvm	e827291c-e9e6-4c7d-8ba7-2bb6f2a80132	1GiB	In-use	iscsi	Attached to test-3 on /dev/vda	Yes	No	Update Volume Status
admin	devstack1.lab@lvm#lvm	1cc9b1a9-bc40-46bf-a6fe-e9bd29fea525	1GiB	In-use	iscsi	Attached to test-2 on /dev/vda	Yes	No	Update Volume Status
admin	devstack1.lab@lvm#lvm	cba3db13-ddd0-479a-8228-1cd689cdfe8f	1GiB	In-use	iscsi	Attached to test1 on /dev/vda	Yes	No	Update Volume Status
admin	devstack1.lab@lvm#lvm	bc6a66a9-b436-48b7-b19b-9a815d829f0c	1GiB	Available	iscsi		Yes	No	Delete Volume
admin	devstack1.lab@lvm#lvm	ed9114ec-fb34-42cc-8f92-439c772609ca	1GiB	Available	iscsi		Yes	No	Delete Volume
admin	devstack1.lab@lvm#lvm	bf4851b2-854b-4a7b-807e-ac5e4a8a6d21	1GiB	Available	iscsi		Yes	No	Delete Volume
admin	devstack1.lab@lvm#lvm	41ea177f-4bdb-4cc1-b41e-fce42333d5ce	1GiB	In-use	iscsi	Attached to cirros-test on /dev/vda	Yes	No	Update Volume Status

FIG.3.8 (OpenStack Volumes attached to different users)

As in FIG.3.8 different volumes are attached to different users for the project “admin”, their IDs are showing as their name with size of their volume, with the contrast of their instances those are attached to.

The next service will be glance service in OpenStack picture. Glance is the storage service that is responsible for keeping images for different instances. All the instances will run through the image file that will be stored in Glance. There can be more than one image but in our scenario its just one image as shown in FIG.3.9



	Owner	Name	Type	Status	Visibility	Protected	Disk Format	Size
	services	cirros	Image	Active	Public	No	QCOW2	12.67 MB

Displaying 1 item

FIG.3.9 (Openstack Glance service)

3.3 Networks and Router

The network is the neutron service of the OpenStack that provides the interfaces for different user and services to connect with each other. Since OpenStack deals with cloud computing with private and public cloud, therefore there will be a big role to play for NATing. As shown in FIG 3.10. OpenStack not only provides compute and data storage service but also understands the role of networks to connect these services. In the following FIG it clearly shows the instances those are connected to a private cloud, on one side of the router with all private IP addresses, will be nated on the other side of the router through public cloud.

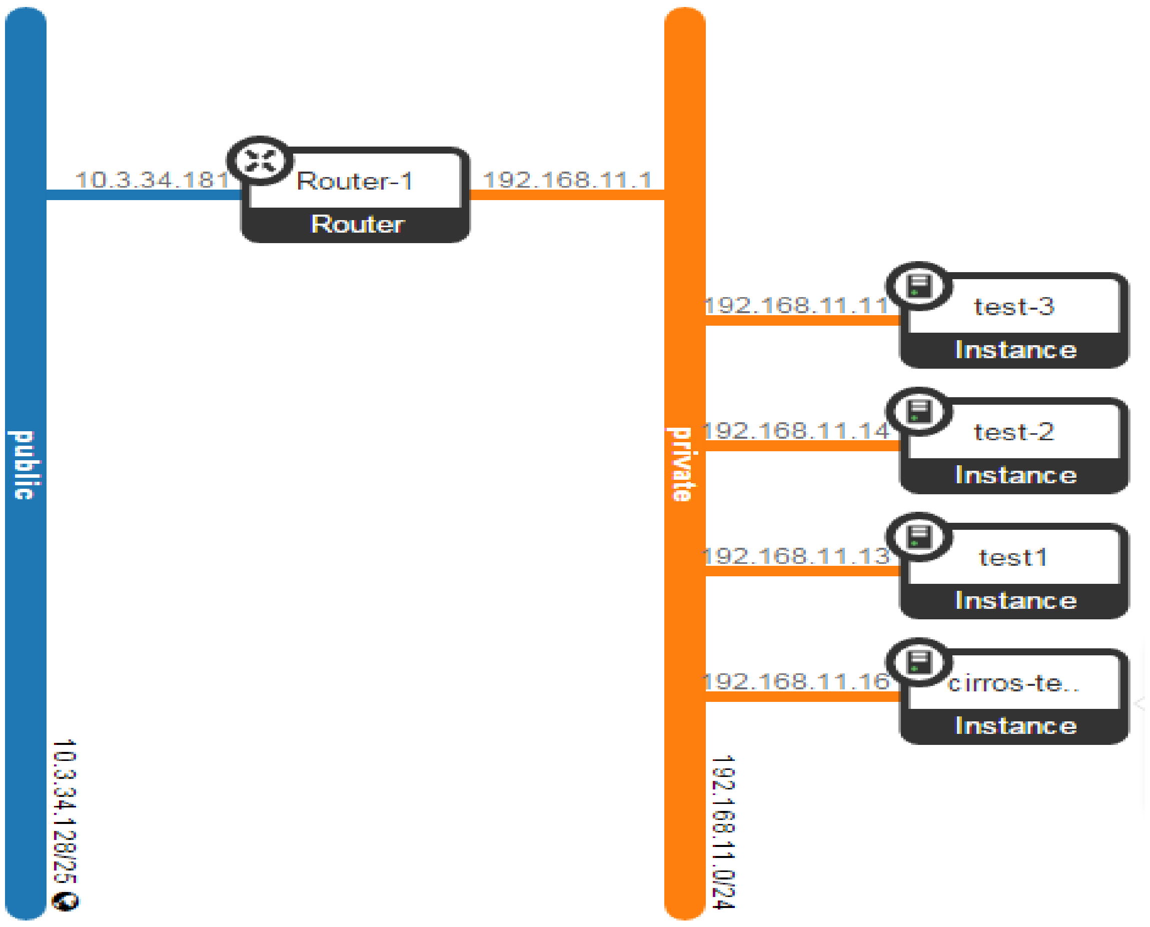
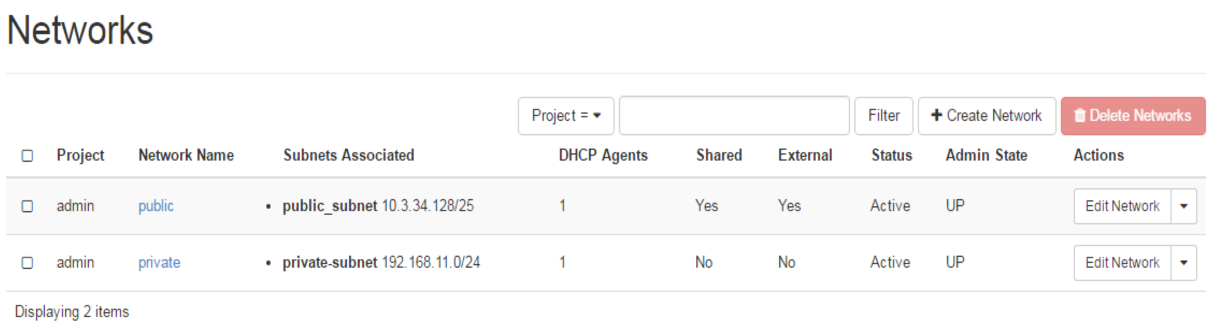


FIG.3.10 (OpenStack Network)

In other words, these private instances can also be accessed through external network. In this unit, this will be explained further.



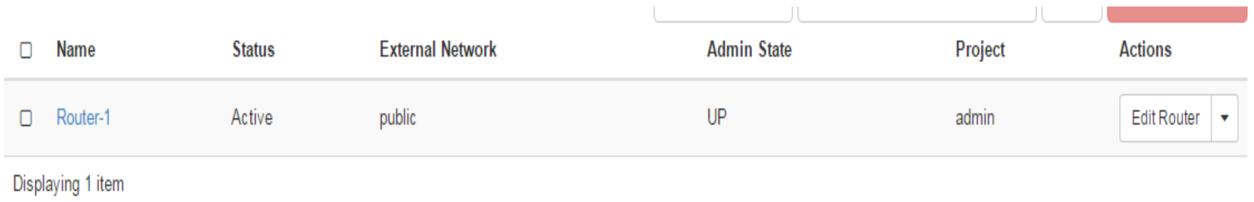
Networks

	Project	Network Name	Subnets Associated	DHCP Agents	Shared	External	Status	Admin State	Actions
☐	admin	public	• **public_subnet** 10.3.34.128/25	1	Yes	Yes	Active	UP	Edit Network
☐	admin	private	• **private-subnet** 192.168.11.0/24	1	No	No	Active	UP	Edit Network

Displaying 2 items

FIG.3.11 (OpenStack Network)

As shown in FIG.3.11 private and public IP addresses are shown that will be used for nating. Furthermore, router is also available is OpenStack networking services that not only help in nating but also manages the external and internal networks of OpenStack services.

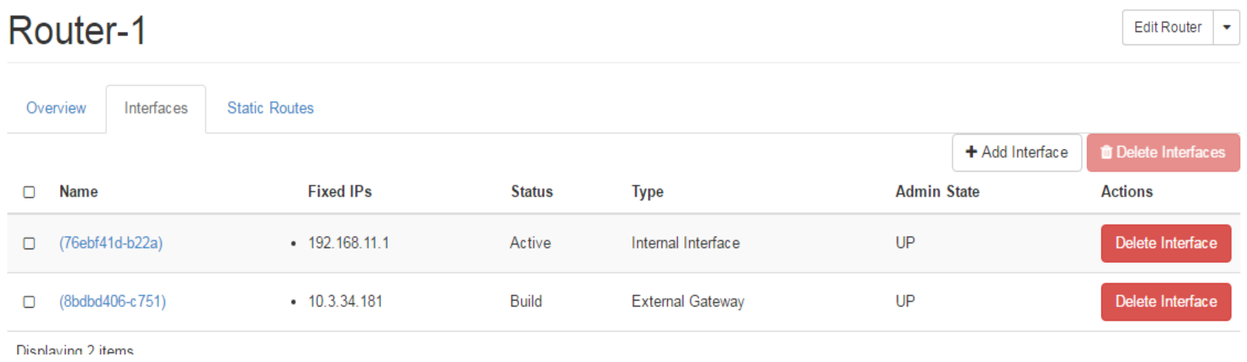


	Name	Status	External Network	Admin State	Project	Actions
☐	Router-1	Active	public	UP	admin	Edit Router

Displaying 1 item

FIG.3.12 (Openstack Router)

Unlike other vendor routers, it does not need so much of command configuration, which is the best part of OpenStack router. It is capable of doing all basic required routing functions. These all capabilities plays great role in OpenStack. As shown in FIG.3.13, FIG.3.14 and FIG.3.15 some of the routing capabilities.

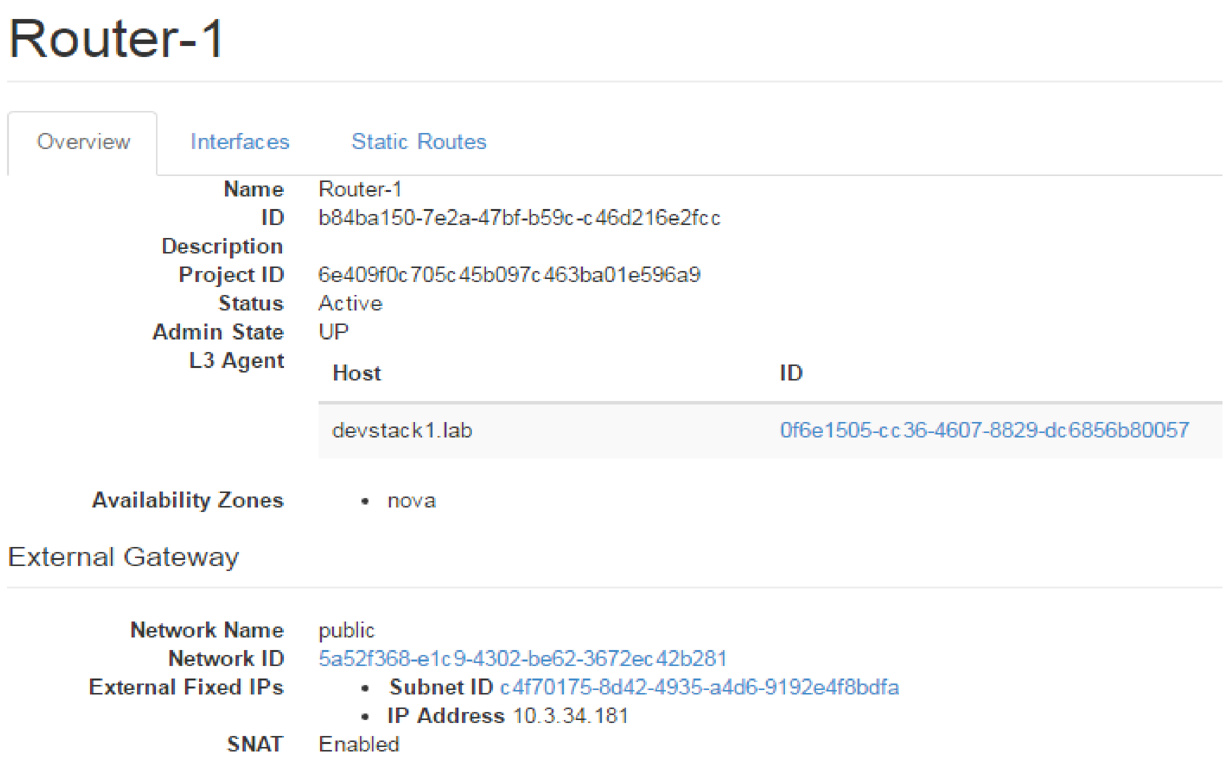


	Name	Fixed IPs	Status	Type	Admin State	Actions
☐	(76ebf41d-b22a)	• 192.168.11.1	Active	Internal Interface	UP	Delete Interface
☐	(8bdbd406-c751)	• 10.3.34.181	Build	External Gateway	UP	Delete Interface

Displaying 2 items

FIG.3.13 (Openstack Router)

In the FIG.3.13 you can easily identify the router ports, where the ports are assigned with the IP addresses, these are the IP addresses from the private and public cloud as shown in FIG.3.4. Moreover, this clarifies that NATing has done by the router, where one port is towards external gateway of the network. This is towards the internet in our scenario. The port is defined on internal or private network.


Router-1

Overview | Interfaces | Static Routes

Name	Router-1
ID	b84ba150-7e2a-47bf-b59c-c46d216e2fcc
Description	
Project ID	6e409f0c705c45b097c463ba01e596a9
Status	Active
Admin State	UP
L3 Agent	

Host	ID
devstack1.lab	0f6e1505-cc36-4607-8829-dc6856b80057

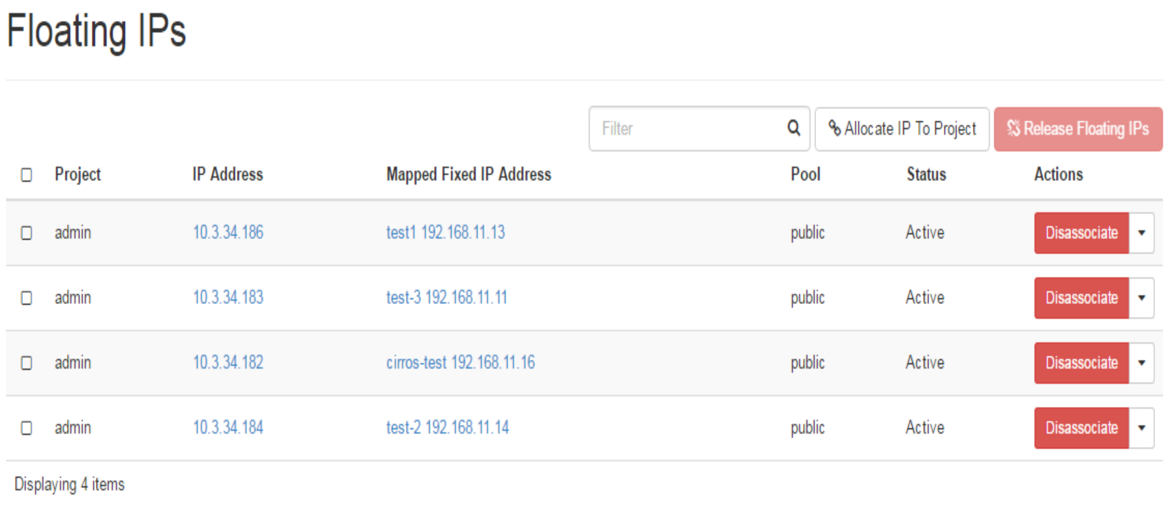
Availability Zones
- nova

External Gateway

Network Name	public
Network ID	5a52f368-e1c9-4302-be62-3672ec42b281
External Fixed IPs	• **Subnet ID** c4f70175-8d42-4935-a4d6-9192e4f8bdfa • **IP Address** 10.3.34.181
SNAT	Enabled

FIG.3.14 (Openstack Router)

Public IP addresses are defined as floating IP for the project that needs to be accessed from external network. In the given FIG 3.15 following floating public IP addresses are shown that are for the admin projects, where instances are also defined with their tagged names, as well as their status.



Floating IPs

Project	IP Address	Mapped Fixed IP Address	Pool	Status	Actions
admin	10.3.34.186	test1 192.168.11.13	public	Active	Disassociate
admin	10.3.34.183	test-3 192.168.11.11	public	Active	Disassociate
admin	10.3.34.182	cirros-test 192.168.11.16	public	Active	Disassociate
admin	10.3.34.184	test-2 192.168.11.14	public	Active	Disassociate

Displaying 4 items

FIG.3.15 (Floating IP addresses)

For any network it very important to have systematic way of deployments and organize the way of understanding the deployment of any network. This cannot be possible without a graphical representation of network topology diagram as shown in FIG.3.16, as well as in FIG.3.10 in the beginning.

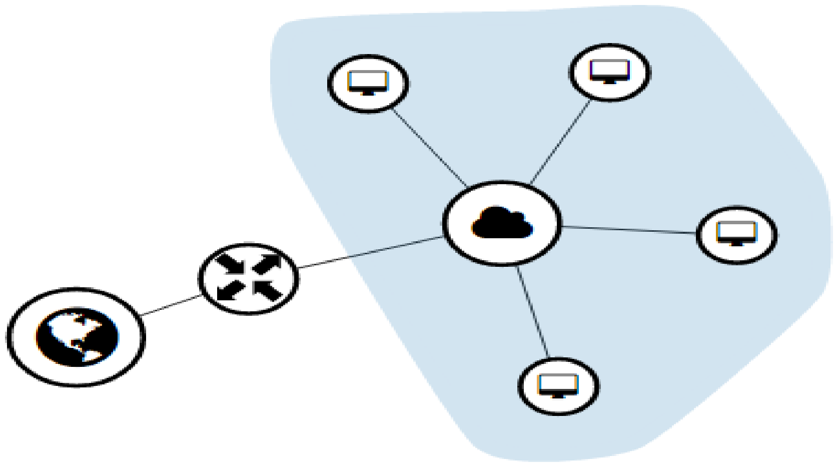
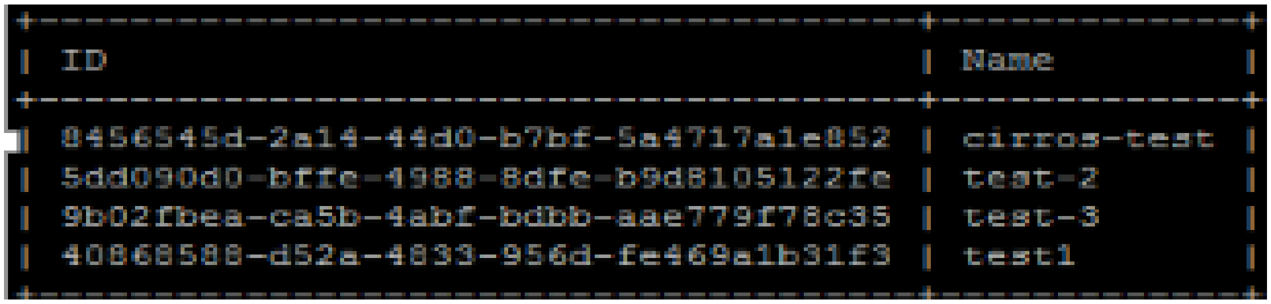


FIG.3.16 (Floating IP addresses)

Chapter 4 Ceilometer Services

4.1 *Implementation of Ceilometer*

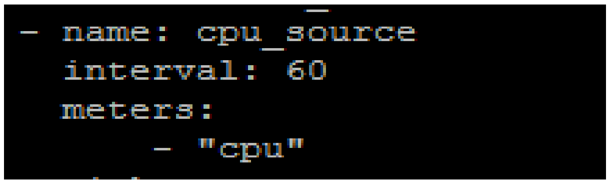
Ceilometer it the service by OpenStack is used to monitor different parameters for different instances, this monitoring can be for metering purposes as well as for alarm purpose as well. The given outputs are the monitoring result for four different tenants created on OpenStack network. The reason behind these outputs is to measure different parameters such as the volume, memory size and number if instances used by the tenant in one project. As mentioned before as in FIG 3.11 in the previous chapter there are 3 instances, outputs are taken for every single instance. Ceilometer monitors and displays every detail separately. Every instance has separate ID and name with their status as shown in FIG 4.1, through which it is identified and is metered accordingly through ceilometer process.



```
+--------------------------------------+-------------+
| ID                                   | Name        |
+--------------------------------------+-------------+
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cirros-test |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | test-2      |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | test-3      |
| 40868588-d52a-4833-956d-fe469a1b31f3 | test1       |
+--------------------------------------+-------------+
```

FIG.4.1 (Instance ID)

In the metering process, there are four separate outputs for four different instances. Every data is taken out for 60 min of the time period in a duration that is starting period to end the period in duration that is shown in the output which is defined in pipeline file known as pipeline.yaml file.

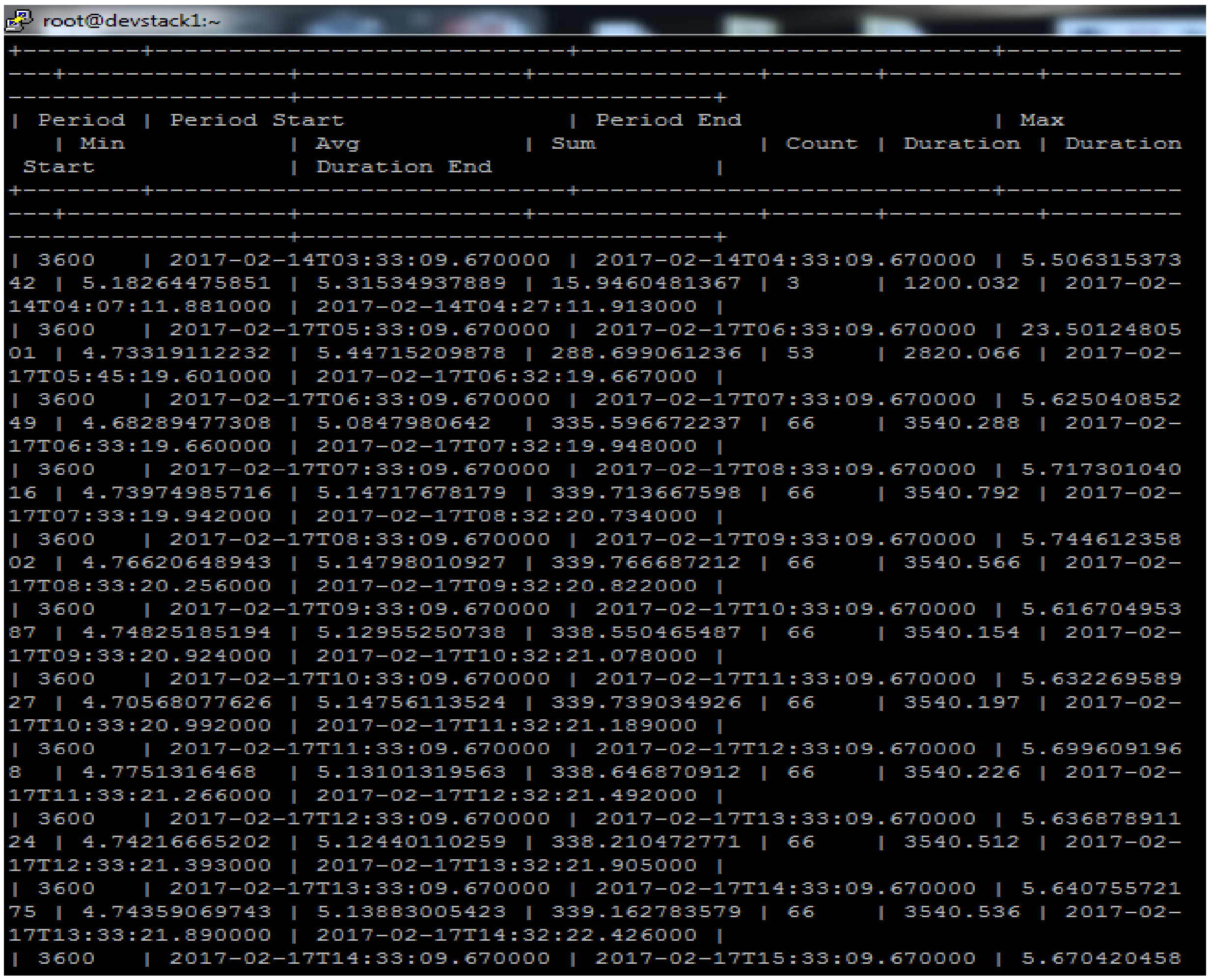


```
- name: cpu_source
  interval: 60
  meters:
      - "cpu"
```

FIG.4.2 (Period of interval)

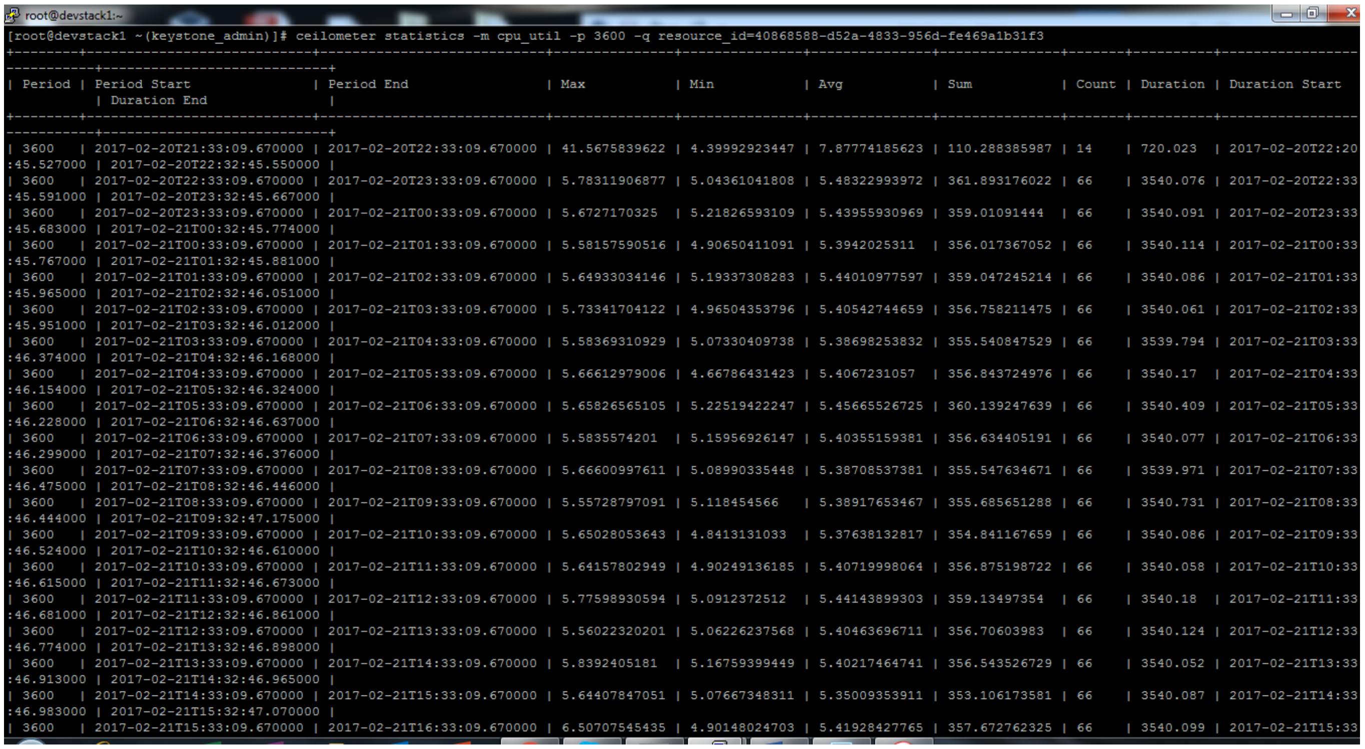
4.2 Expected outputs of Ceilometer

The data will be drawn for an hour and then statistics are taken out. The count tells how many times it is attempted to collect data during this time period. As shown in the output in FIG 4.3, FIG 4.4 and FIG 4.5 and FIG.4.6 the count were 3 in the first hour period. In the next hour period, the count is increased to 53 and later increased at 66 and become stable with 66.It also shows the maximum, minimum and average CPU utilization during that period of time.As shown in the output it is also showing the duration, which is in hour that means period and duration both are the same.



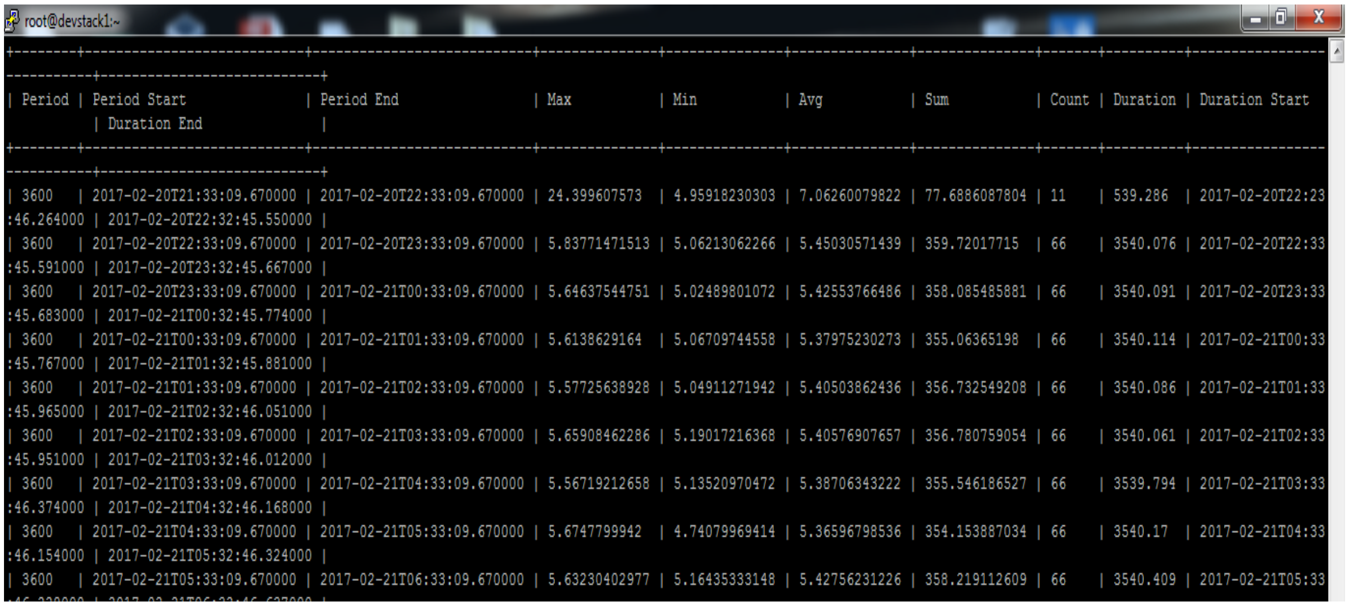
```
root@devstack1:~
+--------+-----------------------------+-----------------------------+------------
---+----------------+----------------+----------------+-------+----------+---------
-------------------+-----------------------------+
| Period | Period Start                | Period End                  | Max
   | Min            | Avg            | Sum            | Count | Duration | Duration
 Start              | Duration End                |
+--------+-----------------------------+-----------------------------+------------
---+----------------+----------------+----------------+-------+----------+---------
-------------------+-----------------------------+
| 3600   | 2017-02-14T03:33:09.670000 | 2017-02-14T04:33:09.670000 | 5.506315373
42 | 5.18264475851 | 5.31534937889 | 15.9460481367 | 3     | 1200.032 | 2017-02-
14T04:07:11.881000 | 2017-02-14T04:27:11.913000 |
| 3600   | 2017-02-17T05:33:09.670000 | 2017-02-17T06:33:09.670000 | 23.50124805
01 | 4.73319112232 | 5.44715209878 | 288.699061236 | 53    | 2820.066 | 2017-02-
17T05:45:19.601000 | 2017-02-17T06:32:19.667000 |
| 3600   | 2017-02-17T06:33:09.670000 | 2017-02-17T07:33:09.670000 | 5.625040852
49 | 4.68289477308 | 5.0847980642  | 335.596672237 | 66    | 3540.288 | 2017-02-
17T06:33:19.660000 | 2017-02-17T07:32:19.948000 |
| 3600   | 2017-02-17T07:33:09.670000 | 2017-02-17T08:33:09.670000 | 5.717301040
16 | 4.73974985716 | 5.14717678179 | 339.713667598 | 66    | 3540.792 | 2017-02-
17T07:33:19.942000 | 2017-02-17T08:32:20.734000 |
| 3600   | 2017-02-17T08:33:09.670000 | 2017-02-17T09:33:09.670000 | 5.744612358
02 | 4.76620648943 | 5.14798010927 | 339.766687212 | 66    | 3540.566 | 2017-02-
17T08:33:20.256000 | 2017-02-17T09:32:20.822000 |
| 3600   | 2017-02-17T09:33:09.670000 | 2017-02-17T10:33:09.670000 | 5.616704953
87 | 4.74825185194 | 5.12955250738 | 338.550465487 | 66    | 3540.154 | 2017-02-
17T09:33:20.924000 | 2017-02-17T10:32:21.078000 |
| 3600   | 2017-02-17T10:33:09.670000 | 2017-02-17T11:33:09.670000 | 5.632269589
27 | 4.70568077626 | 5.14756113524 | 339.739034926 | 66    | 3540.197 | 2017-02-
17T10:33:20.992000 | 2017-02-17T11:32:21.189000 |
| 3600   | 2017-02-17T11:33:09.670000 | 2017-02-17T12:33:09.670000 | 5.699609196
8  | 4.7751316468  | 5.13101319563 | 338.646870912 | 66    | 3540.226 | 2017-02-
17T11:33:21.266000 | 2017-02-17T12:32:21.492000 |
| 3600   | 2017-02-17T12:33:09.670000 | 2017-02-17T13:33:09.670000 | 5.636878911
24 | 4.74216665202 | 5.12440110259 | 338.210472771 | 66    | 3540.512 | 2017-02-
17T12:33:21.393000 | 2017-02-17T13:32:21.905000 |
| 3600   | 2017-02-17T13:33:09.670000 | 2017-02-17T14:33:09.670000 | 5.640755721
75 | 4.74359069743 | 5.13883005423 | 339.162783579 | 66    | 3540.536 | 2017-02-
17T13:33:21.890000 | 2017-02-17T14:32:22.426000 |
| 3600   | 2017-02-17T14:33:09.670000 | 2017-02-17T15:33:09.670000 | 5.670420458
```

FIG.4.3 (CPU utilization statistics for Cirros Test)



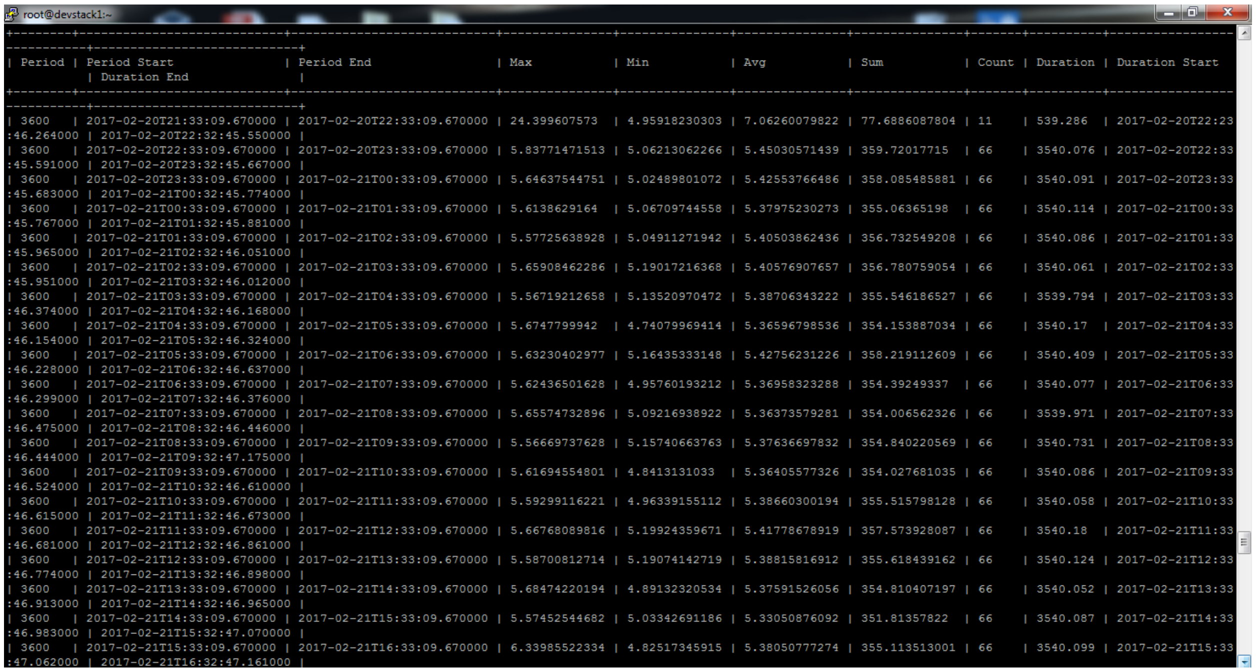
```
root@devstack1:~
[root@devstack1 ~(keystone_admin)]# ceilometer statistics -m cpu_util -p 3600 -q resource_id=40868588-d52a-4833-956d-fe469a1b31f3
+--------+----------------------------+----------------------------+---------------+---------------+---------------+---------------+-------+----------+-----------------
-----------+----------------------------+
| Period | Period Start               | Period End                 | Max           | Min           | Avg           | Sum           | Count | Duration | Duration Start
           | Duration End               |
+--------+----------------------------+----------------------------+---------------+---------------+---------------+---------------+-------+----------+-----------------
-----------+----------------------------+
| 3600   | 2017-02-20T21:33:09.670000 | 2017-02-20T22:33:09.670000 | 41.5675839622 | 4.39992923447 | 7.87774185623 | 110.288385987 | 14    | 720.023  | 2017-02-20T22:20
:45.527000 | 2017-02-20T22:32:45.550000 |
| 3600   | 2017-02-20T22:33:09.670000 | 2017-02-20T23:33:09.670000 | 5.78311906877 | 5.04361041808 | 5.48322993972 | 361.893176022 | 66    | 3540.076 | 2017-02-20T22:33
:45.591000 | 2017-02-20T23:32:45.667000 |
| 3600   | 2017-02-20T23:33:09.670000 | 2017-02-21T00:33:09.670000 | 5.6727170325  | 5.21826593109 | 5.43955930969 | 359.01091444  | 66    | 3540.091 | 2017-02-20T23:33
:45.683000 | 2017-02-21T00:32:45.774000 |
| 3600   | 2017-02-21T00:33:09.670000 | 2017-02-21T01:33:09.670000 | 5.58157590516 | 4.90650411091 | 5.3942025311  | 356.017367052 | 66    | 3540.114 | 2017-02-21T00:33
:45.767000 | 2017-02-21T01:32:45.881000 |
| 3600   | 2017-02-21T01:33:09.670000 | 2017-02-21T02:33:09.670000 | 5.64933034146 | 5.19337308283 | 5.44010977597 | 359.047245214 | 66    | 3540.086 | 2017-02-21T01:33
:45.965000 | 2017-02-21T02:32:46.051000 |
| 3600   | 2017-02-21T02:33:09.670000 | 2017-02-21T03:33:09.670000 | 5.73341704122 | 4.96504353796 | 5.40542744659 | 356.758211475 | 66    | 3540.061 | 2017-02-21T02:33
:45.951000 | 2017-02-21T03:32:46.012000 |
| 3600   | 2017-02-21T03:33:09.670000 | 2017-02-21T04:33:09.670000 | 5.58369310929 | 5.07330409738 | 5.38698253832 | 355.540847529 | 66    | 3539.794 | 2017-02-21T03:33
:46.374000 | 2017-02-21T04:32:46.168000 |
| 3600   | 2017-02-21T04:33:09.670000 | 2017-02-21T05:33:09.670000 | 5.66612979006 | 4.66786431423 | 5.4067231057  | 356.843724976 | 66    | 3540.17  | 2017-02-21T04:33
:46.154000 | 2017-02-21T05:32:46.324000 |
| 3600   | 2017-02-21T05:33:09.670000 | 2017-02-21T06:33:09.670000 | 5.65826565105 | 5.22519422247 | 5.45665526725 | 360.139247639 | 66    | 3540.409 | 2017-02-21T05:33
:46.228000 | 2017-02-21T06:32:46.637000 |
| 3600   | 2017-02-21T06:33:09.670000 | 2017-02-21T07:33:09.670000 | 5.5835574201  | 5.15956926147 | 5.40355159381 | 356.634405191 | 66    | 3540.077 | 2017-02-21T06:33
:46.299000 | 2017-02-21T07:32:46.376000 |
| 3600   | 2017-02-21T07:33:09.670000 | 2017-02-21T08:33:09.670000 | 5.66600997611 | 5.08990335448 | 5.38708537381 | 355.547634671 | 66    | 3539.971 | 2017-02-21T07:33
:46.475000 | 2017-02-21T08:32:46.446000 |
| 3600   | 2017-02-21T08:33:09.670000 | 2017-02-21T09:33:09.670000 | 5.55728797091 | 5.118454566   | 5.38917653467 | 355.685651288 | 66    | 3540.731 | 2017-02-21T08:33
:46.444000 | 2017-02-21T09:32:47.175000 |
| 3600   | 2017-02-21T09:33:09.670000 | 2017-02-21T10:33:09.670000 | 5.65028053643 | 4.8413131033  | 5.37638132817 | 354.841167659 | 66    | 3540.086 | 2017-02-21T09:33
:46.524000 | 2017-02-21T10:32:46.610000 |
| 3600   | 2017-02-21T10:33:09.670000 | 2017-02-21T11:33:09.670000 | 5.64157802949 | 4.90249136185 | 5.40719998064 | 356.875198722 | 66    | 3540.058 | 2017-02-21T10:33
:46.615000 | 2017-02-21T11:32:46.673000 |
| 3600   | 2017-02-21T11:33:09.670000 | 2017-02-21T12:33:09.670000 | 5.77598930594 | 5.0912372512  | 5.44143899303 | 359.13497354  | 66    | 3540.18  | 2017-02-21T11:33
:46.681000 | 2017-02-21T12:32:46.861000 |
| 3600   | 2017-02-21T12:33:09.670000 | 2017-02-21T13:33:09.670000 | 5.56022320201 | 5.06226237568 | 5.40463696711 | 356.70603983  | 66    | 3540.124 | 2017-02-21T12:33
:46.774000 | 2017-02-21T13:32:46.898000 |
| 3600   | 2017-02-21T13:33:09.670000 | 2017-02-21T14:33:09.670000 | 5.8392405181  | 5.16759399449 | 5.40217464741 | 356.543526729 | 66    | 3540.052 | 2017-02-21T13:33
:46.913000 | 2017-02-21T14:32:46.965000 |
| 3600   | 2017-02-21T14:33:09.670000 | 2017-02-21T15:33:09.670000 | 5.64407847051 | 5.07667348311 | 5.35009353911 | 353.106173581 | 66    | 3540.087 | 2017-02-21T14:33
:46.983000 | 2017-02-21T15:32:47.070000 |
| 3600   | 2017-02-21T15:33:09.670000 | 2017-02-21T16:33:09.670000 | 6.50707545435 | 4.90148024703 | 5.41928427765 | 357.672762325 | 66    | 3540.099 | 2017-02-21T15:33
```

FIG.4.4 (CPU utilization statistics for Test-1)



```
+--------+----------------------------+----------------------------+---------------+---------------+---------------+---------------+-------+----------+----------------
-----------+----------------------------+
| Period | Period Start               | Period End                 | Max           | Min           | Avg           | Sum           | Count | Duration | Duration Start
           | Duration End               |
+--------+----------------------------+----------------------------+---------------+---------------+---------------+---------------+-------+----------+----------------
-----------+----------------------------+
| 3600   | 2017-02-20T21:33:09.670000 | 2017-02-20T22:33:09.670000 | 24.399607573  | 4.95918230303 | 7.06260079822 | 77.6886087804 | 11    | 539.286  | 2017-02-20T22:23
:46.264000 | 2017-02-20T22:32:45.550000 |
| 3600   | 2017-02-20T22:33:09.670000 | 2017-02-20T23:33:09.670000 | 5.83771471513 | 5.06213062266 | 5.45030571439 | 359.72017715  | 66    | 3540.076 | 2017-02-20T22:33
:45.591000 | 2017-02-20T23:32:45.667000 |
| 3600   | 2017-02-20T23:33:09.670000 | 2017-02-21T00:33:09.670000 | 5.64637544751 | 5.02489801072 | 5.42553766486 | 358.085485881 | 66    | 3540.091 | 2017-02-20T23:33
:45.683000 | 2017-02-21T00:32:45.774000 |
| 3600   | 2017-02-21T00:33:09.670000 | 2017-02-21T01:33:09.670000 | 5.6138629164  | 5.06709744558 | 5.37975230273 | 355.06365198  | 66    | 3540.114 | 2017-02-21T00:33
:45.767000 | 2017-02-21T01:32:45.881000 |
| 3600   | 2017-02-21T01:33:09.670000 | 2017-02-21T02:33:09.670000 | 5.57725638928 | 5.04911271942 | 5.40503862436 | 356.732549208 | 66    | 3540.086 | 2017-02-21T01:33
:45.965000 | 2017-02-21T02:32:46.051000 |
| 3600   | 2017-02-21T02:33:09.670000 | 2017-02-21T03:33:09.670000 | 5.65908462286 | 5.19017216368 | 5.40576907657 | 356.780759054 | 66    | 3540.061 | 2017-02-21T02:33
:45.951000 | 2017-02-21T03:32:46.012000 |
| 3600   | 2017-02-21T03:33:09.670000 | 2017-02-21T04:33:09.670000 | 5.56719212658 | 5.13520970472 | 5.38706343222 | 355.546186527 | 66    | 3539.794 | 2017-02-21T03:33
:46.374000 | 2017-02-21T04:32:46.168000 |
| 3600   | 2017-02-21T04:33:09.670000 | 2017-02-21T05:33:09.670000 | 5.6747799942  | 4.74079969414 | 5.36596798536 | 354.153887034 | 66    | 3540.17  | 2017-02-21T04:33
:46.154000 | 2017-02-21T05:32:46.324000 |
| 3600   | 2017-02-21T05:33:09.670000 | 2017-02-21T06:33:09.670000 | 5.63230402977 | 5.16435333148 | 5.42756231226 | 358.219112609 | 66    | 3540.409 | 2017-02-21T05:33
```

FIG.4.5 (CPU utilization statistics for Test-2)

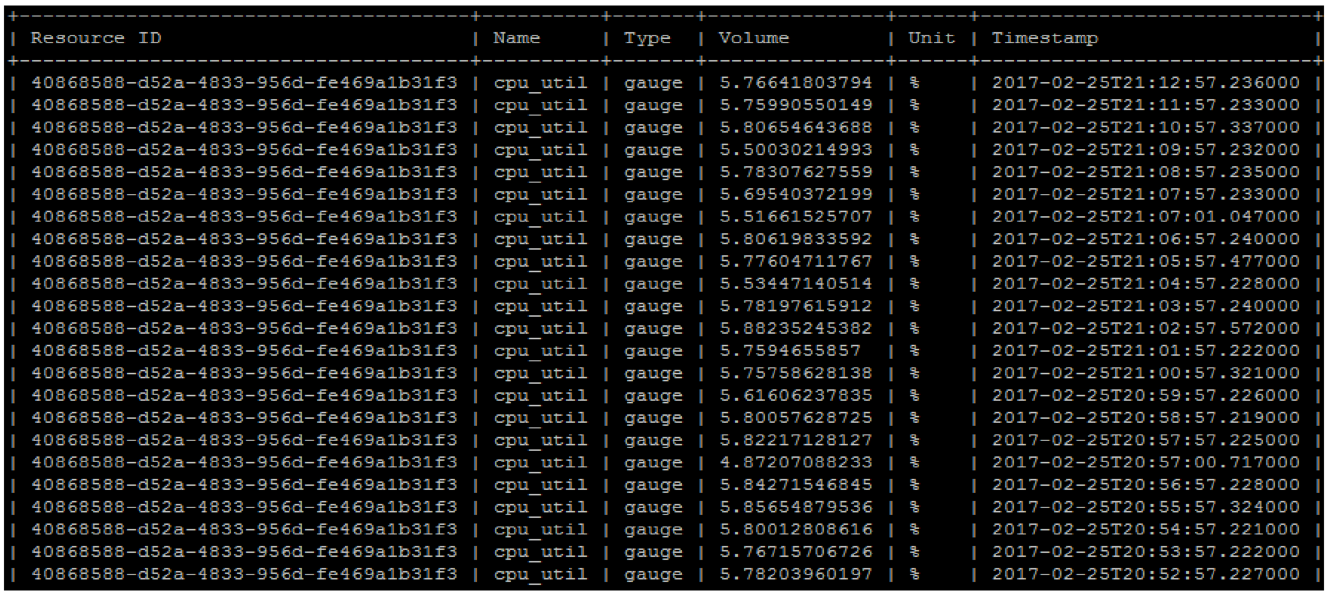


```
+--------+----------------------------+----------------------------+---------------+---------------+---------------+---------------+-------+----------+----------------
-----------+----------------------------+
| Period | Period Start               | Period End                 | Max           | Min           | Avg           | Sum           | Count | Duration | Duration Start
           | Duration End               |
+--------+----------------------------+----------------------------+---------------+---------------+---------------+---------------+-------+----------+----------------
-----------+----------------------------+
| 3600   | 2017-02-20T21:33:09.670000 | 2017-02-20T22:33:09.670000 | 24.399607573  | 4.95918230303 | 7.06260079822 | 77.6886087804 | 11    | 539.286  | 2017-02-20T22:23
:46.264000 | 2017-02-20T22:32:45.550000 |
| 3600   | 2017-02-20T22:33:09.670000 | 2017-02-20T23:33:09.670000 | 5.83771471513 | 5.06213062266 | 5.45030571439 | 359.72017715  | 66    | 3540.076 | 2017-02-20T22:33
:45.591000 | 2017-02-20T23:32:45.667000 |
| 3600   | 2017-02-20T23:33:09.670000 | 2017-02-21T00:33:09.670000 | 5.64637544751 | 5.02489801072 | 5.42553766486 | 358.085485881 | 66    | 3540.091 | 2017-02-20T23:33
:45.683000 | 2017-02-21T00:32:45.774000 |
| 3600   | 2017-02-21T00:33:09.670000 | 2017-02-21T01:33:09.670000 | 5.6138629164  | 5.06709744558 | 5.37975230273 | 355.06365198  | 66    | 3540.114 | 2017-02-21T00:33
:45.767000 | 2017-02-21T01:32:45.881000 |
| 3600   | 2017-02-21T01:33:09.670000 | 2017-02-21T02:33:09.670000 | 5.57725638928 | 5.04911271942 | 5.40503862436 | 356.732549208 | 66    | 3540.086 | 2017-02-21T01:33
:45.965000 | 2017-02-21T02:32:46.051000 |
| 3600   | 2017-02-21T02:33:09.670000 | 2017-02-21T03:33:09.670000 | 5.65908462286 | 5.19017216368 | 5.40576907657 | 356.780759054 | 66    | 3540.061 | 2017-02-21T02:33
:45.951000 | 2017-02-21T03:32:46.012000 |
| 3600   | 2017-02-21T03:33:09.670000 | 2017-02-21T04:33:09.670000 | 5.56719212658 | 5.13520970472 | 5.38706343222 | 355.546186527 | 66    | 3539.794 | 2017-02-21T03:33
:46.374000 | 2017-02-21T04:32:46.168000 |
| 3600   | 2017-02-21T04:33:09.670000 | 2017-02-21T05:33:09.670000 | 5.6747799942  | 4.74079969414 | 5.36596798536 | 354.153887034 | 66    | 3540.17  | 2017-02-21T04:33
:46.154000 | 2017-02-21T05:32:46.324000 |
| 3600   | 2017-02-21T05:33:09.670000 | 2017-02-21T06:33:09.670000 | 5.63230402977 | 5.16435333148 | 5.42756231226 | 358.219112609 | 66    | 3540.409 | 2017-02-21T05:33
:46.228000 | 2017-02-21T06:32:46.637000 |
| 3600   | 2017-02-21T06:33:09.670000 | 2017-02-21T07:33:09.670000 | 5.62436501628 | 4.95760193212 | 5.36958323288 | 354.39249337  | 66    | 3540.077 | 2017-02-21T06:33
:46.299000 | 2017-02-21T07:32:46.376000 |
| 3600   | 2017-02-21T07:33:09.670000 | 2017-02-21T08:33:09.670000 | 5.65574732896 | 5.09216938922 | 5.36373579281 | 354.006562326 | 66    | 3539.971 | 2017-02-21T07:33
:46.475000 | 2017-02-21T08:32:46.446000 |
| 3600   | 2017-02-21T08:33:09.670000 | 2017-02-21T09:33:09.670000 | 5.56669737628 | 5.15740663763 | 5.37636697832 | 354.840220569 | 66    | 3540.731 | 2017-02-21T08:33
:46.444000 | 2017-02-21T09:32:47.175000 |
| 3600   | 2017-02-21T09:33:09.670000 | 2017-02-21T10:33:09.670000 | 5.61694554801 | 4.8413131033  | 5.36405577326 | 354.027681035 | 66    | 3540.086 | 2017-02-21T09:33
:46.524000 | 2017-02-21T10:32:46.610000 |
| 3600   | 2017-02-21T10:33:09.670000 | 2017-02-21T11:33:09.670000 | 5.59299116221 | 4.96339155112 | 5.38660300194 | 355.515798128 | 66    | 3540.058 | 2017-02-21T10:33
:46.615000 | 2017-02-21T11:32:46.673000 |
| 3600   | 2017-02-21T11:33:09.670000 | 2017-02-21T12:33:09.670000 | 5.66768089816 | 5.19924359671 | 5.41778678919 | 357.573928087 | 66    | 3540.18  | 2017-02-21T11:33
:46.681000 | 2017-02-21T12:32:46.861000 |
| 3600   | 2017-02-21T12:33:09.670000 | 2017-02-21T13:33:09.670000 | 5.58700812714 | 5.19074142719 | 5.38815816912 | 355.618439162 | 66    | 3540.124 | 2017-02-21T12:33
:46.774000 | 2017-02-21T13:32:46.898000 |
| 3600   | 2017-02-21T13:33:09.670000 | 2017-02-21T14:33:09.670000 | 5.68474220194 | 4.89132320534 | 5.37591526056 | 354.810407197 | 66    | 3540.052 | 2017-02-21T13:33
:46.913000 | 2017-02-21T14:32:46.965000 |
| 3600   | 2017-02-21T14:33:09.670000 | 2017-02-21T15:33:09.670000 | 5.57452544682 | 5.03342691186 | 5.33050876092 | 351.81357822  | 66    | 3540.087 | 2017-02-21T14:33
:46.983000 | 2017-02-21T15:32:47.070000 |
| 3600   | 2017-02-21T15:33:09.670000 | 2017-02-21T16:33:09.670000 | 6.33985522334 | 4.82517345915 | 5.38050777274 | 355.113513001 | 66    | 3540.099 | 2017-02-21T15:33
:47.062000 | 2017-02-21T16:32:47.161000 |
```

FIG.4.6 (CPU utilization statistics for Test-3)

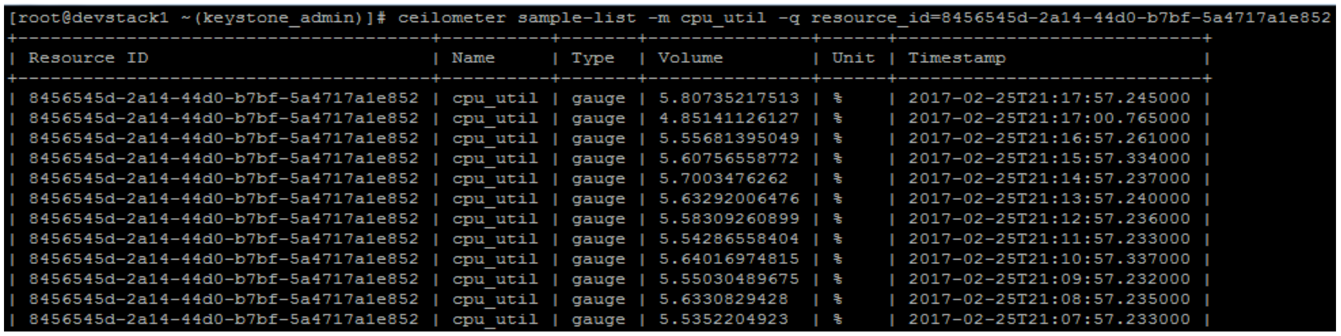
A sample list is also one of the ways to get the information about a different instance in ceilometer. The statistics show us the time period and time duration of the instance with the contrast of max, min, and an average of any perimeter. It also includes the number of the counts attempted to get the information. On other page sample list also tells about the total usage of the resource that is been used, as well as the type of metering the perimeter is, that can be Cumulative, Delta and Gauge as discussed before. Moreover, it also manages to give the system Id of the instance and the time stamp. These are some of the obvious fields that are shown in sample-list. The monitoring parameters may get change, with their units. These are defined by ceilometer in sample-list command. As shown in Fig.4.7, Fig.4.8, Fig.4.9 and Fig.4.10.

In the first group of outputs, CPU utilization is shown in form of percentage. The metering type is gauge that is discrete values and may change



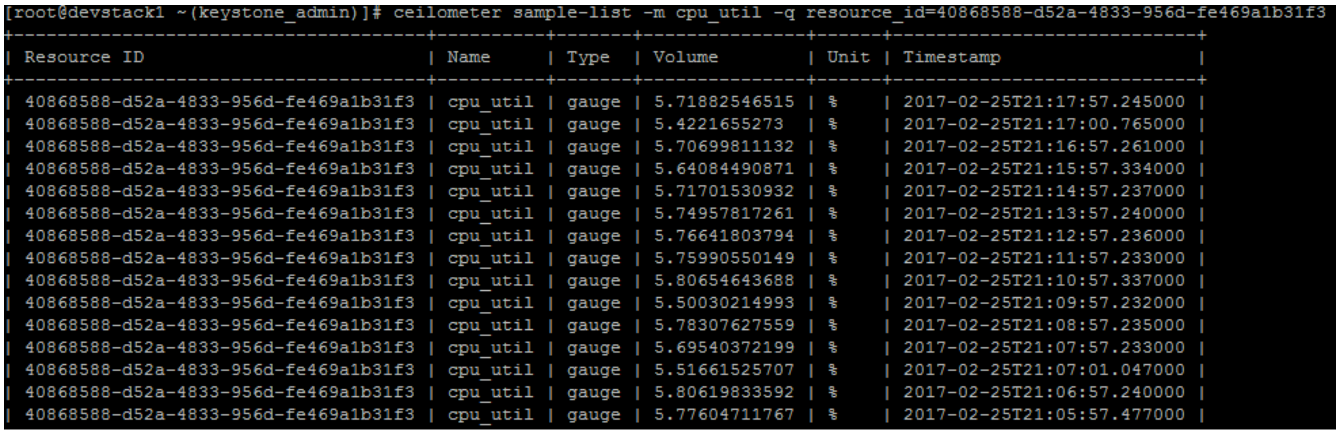
```
+--------------------------------------+----------+-------+----------------+------+----------------------------+
| Resource ID                          | Name     | Type  | Volume         | Unit | Timestamp                  |
+--------------------------------------+----------+-------+----------------+------+----------------------------+
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.76641803794  | %    | 2017-02-25T21:12:57.236000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.75990550149  | %    | 2017-02-25T21:11:57.233000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.80654643688  | %    | 2017-02-25T21:10:57.337000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.50030214993  | %    | 2017-02-25T21:09:57.232000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.78307627559  | %    | 2017-02-25T21:08:57.235000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.69540372199  | %    | 2017-02-25T21:07:57.233000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.51661525707  | %    | 2017-02-25T21:07:01.047000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.80619833592  | %    | 2017-02-25T21:06:57.240000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.77604711767  | %    | 2017-02-25T21:05:57.477000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.53447140514  | %    | 2017-02-25T21:04:57.228000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.78197615912  | %    | 2017-02-25T21:03:57.240000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.88235245382  | %    | 2017-02-25T21:02:57.572000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.7594655857   | %    | 2017-02-25T21:01:57.222000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.75758628138  | %    | 2017-02-25T21:00:57.321000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.61606237835  | %    | 2017-02-25T20:59:57.226000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.80057628725  | %    | 2017-02-25T20:58:57.219000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.82217128127  | %    | 2017-02-25T20:57:57.225000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 4.87207088233  | %    | 2017-02-25T20:57:00.717000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.84271546845  | %    | 2017-02-25T20:56:57.228000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.85654879536  | %    | 2017-02-25T20:55:57.324000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.80012808616  | %    | 2017-02-25T20:54:57.221000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.76715706726  | %    | 2017-02-25T20:53:57.222000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.78203960197  | %    | 2017-02-25T20:52:57.227000 |
```

FIG.4.7 (CPU utilization sample list for Cirros Test)



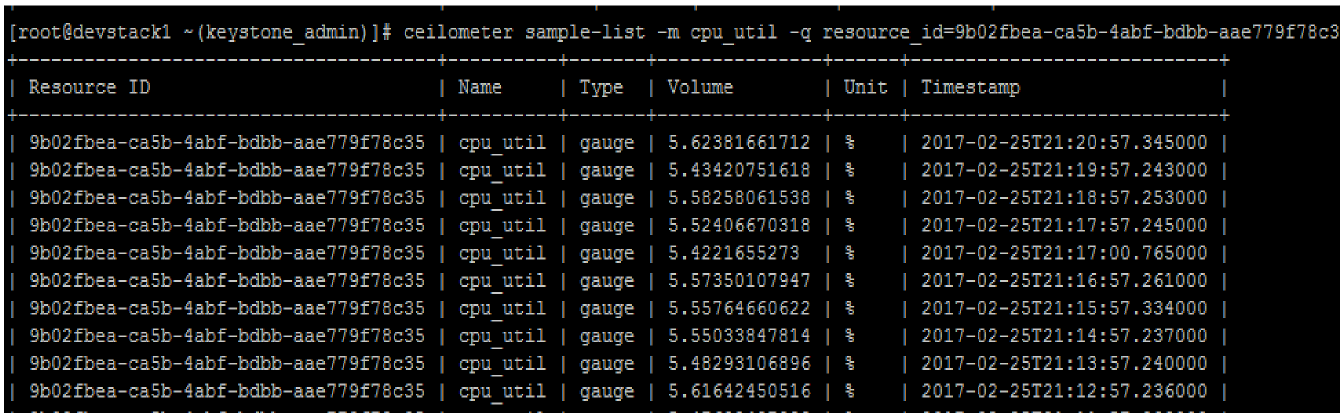
```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m cpu_util -q resource_id=8456545d-2a14-44d0-b7bf-5a4717a1e852
+--------------------------------------+----------+-------+---------------+------+----------------------------+
| Resource ID                          | Name     | Type  | Volume        | Unit | Timestamp                  |
+--------------------------------------+----------+-------+---------------+------+----------------------------+
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.80735217513 | %    | 2017-02-25T21:17:57.245000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 4.85141126127 | %    | 2017-02-25T21:17:00.765000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.55681395049 | %    | 2017-02-25T21:16:57.261000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.60756558772 | %    | 2017-02-25T21:15:57.334000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.7003476262  | %    | 2017-02-25T21:14:57.237000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.63292006476 | %    | 2017-02-25T21:13:57.240000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.58309260899 | %    | 2017-02-25T21:12:57.236000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.54286558404 | %    | 2017-02-25T21:11:57.233000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.64016974815 | %    | 2017-02-25T21:10:57.337000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.55030489675 | %    | 2017-02-25T21:09:57.232000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.6330829428  | %    | 2017-02-25T21:08:57.235000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu_util | gauge | 5.5352204923  | %    | 2017-02-25T21:07:57.233000 |
```

FIG.4.8 (CPU utilization sample list for Test-1)



```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m cpu_util -q resource_id=40868588-d52a-4833-956d-fe469a1b31f3
+--------------------------------------+----------+-------+---------------+------+----------------------------+
| Resource ID                          | Name     | Type  | Volume        | Unit | Timestamp                  |
+--------------------------------------+----------+-------+---------------+------+----------------------------+
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.71882546515 | %    | 2017-02-25T21:17:57.245000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.4221655273  | %    | 2017-02-25T21:17:00.765000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.70699811132 | %    | 2017-02-25T21:16:57.261000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.64084490871 | %    | 2017-02-25T21:15:57.334000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.71701530932 | %    | 2017-02-25T21:14:57.237000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.74957817261 | %    | 2017-02-25T21:13:57.240000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.76641803794 | %    | 2017-02-25T21:12:57.236000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.75990550149 | %    | 2017-02-25T21:11:57.233000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.80654643688 | %    | 2017-02-25T21:10:57.337000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.50030214993 | %    | 2017-02-25T21:09:57.232000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.78307627559 | %    | 2017-02-25T21:08:57.235000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.69540372199 | %    | 2017-02-25T21:07:57.233000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.51661525707 | %    | 2017-02-25T21:07:01.047000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.80619833592 | %    | 2017-02-25T21:06:57.240000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu_util | gauge | 5.77604711767 | %    | 2017-02-25T21:05:57.477000 |
```

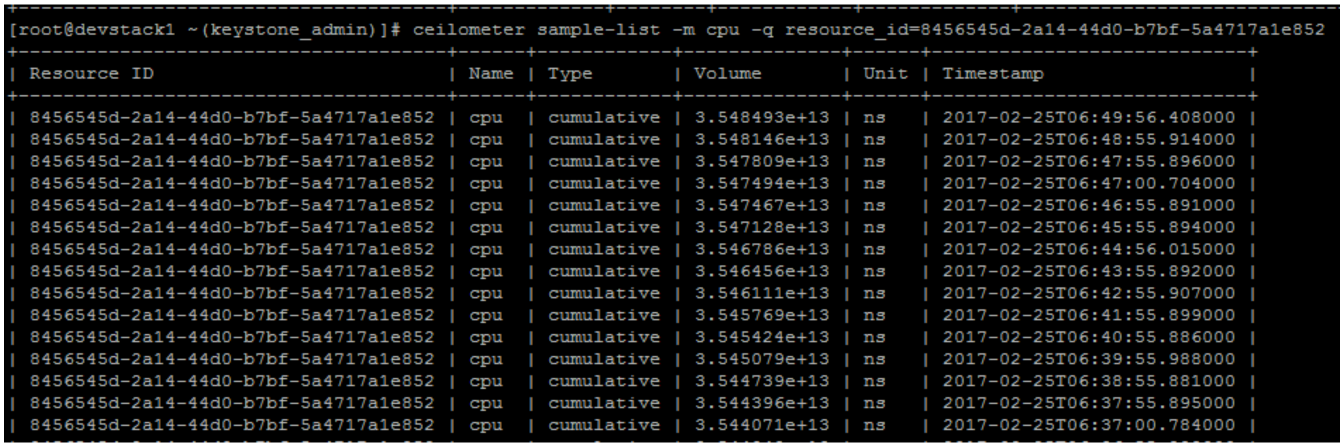
FIG.4.9 (CPU utilization sample list for Test-2)



```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m cpu_util -q resource_id=9b02fbea-ca5b-4abf-bdbb-aae779f78c3
+--------------------------------------+----------+-------+---------------+------+----------------------------+
| Resource ID                          | Name     | Type  | Volume        | Unit | Timestamp                  |
+--------------------------------------+----------+-------+---------------+------+----------------------------+
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.62381661712 | %    | 2017-02-25T21:20:57.345000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.43420751618 | %    | 2017-02-25T21:19:57.243000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.58258061538 | %    | 2017-02-25T21:18:57.253000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.52406670318 | %    | 2017-02-25T21:17:57.245000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.4221655273  | %    | 2017-02-25T21:17:00.765000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.57350107947 | %    | 2017-02-25T21:16:57.261000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.55764660622 | %    | 2017-02-25T21:15:57.334000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.55033847814 | %    | 2017-02-25T21:14:57.237000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.48293106896 | %    | 2017-02-25T21:13:57.240000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu_util | gauge | 5.61642450516 | %    | 2017-02-25T21:12:57.236000 |
```

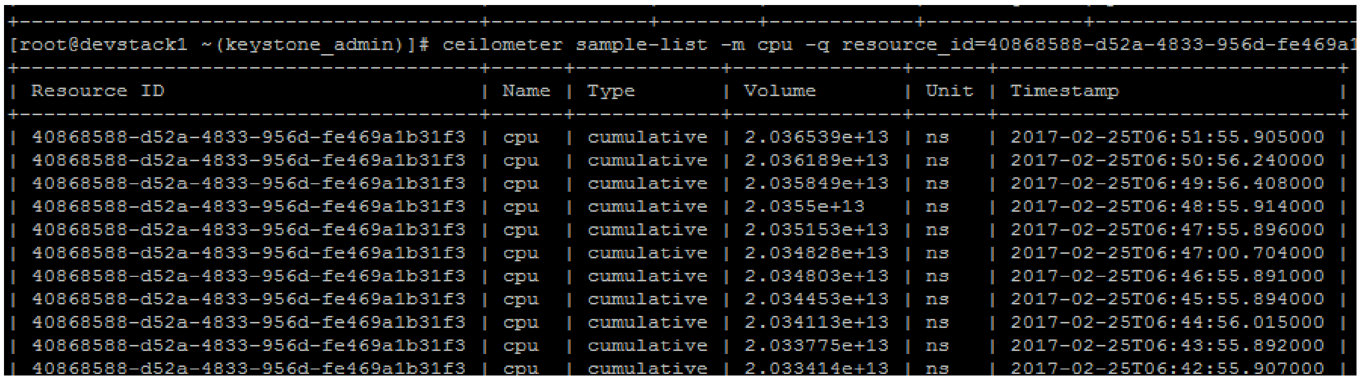
FIG.4.10 (CPU utilization sample list for Test-3)

In next set of outputs in FIG.4.11, FIG.4.12, FIG.4.13 and FIG.4.14 the parameter remains the same, with a contrast of different unit. This shows multiple ways that ceilometer monitors different parameters and show them in a presentable format. The Percentage is now changed to nS, hence metering type is changed to cumulative, now it is showing the volume in the units of Nanoseconds.



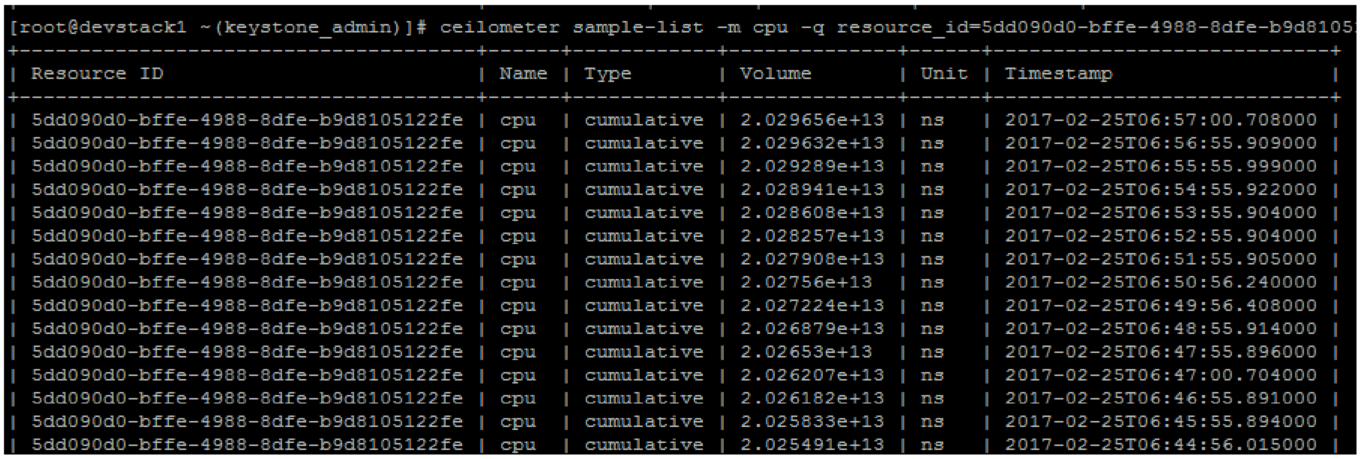
```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m cpu -q resource_id=8456545d-2a14-44d0-b7bf-5a4717a1e852
+--------------------------------------+------+------------+--------------+------+----------------------------+
| Resource ID                          | Name | Type       | Volume       | Unit | Timestamp                  |
+--------------------------------------+------+------------+--------------+------+----------------------------+
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.548493e+13 | ns   | 2017-02-25T06:49:56.408000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.548146e+13 | ns   | 2017-02-25T06:48:55.914000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.547809e+13 | ns   | 2017-02-25T06:47:55.896000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.547494e+13 | ns   | 2017-02-25T06:47:00.704000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.547467e+13 | ns   | 2017-02-25T06:46:55.891000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.547128e+13 | ns   | 2017-02-25T06:45:55.894000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.546786e+13 | ns   | 2017-02-25T06:44:56.015000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.546456e+13 | ns   | 2017-02-25T06:43:55.892000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.546111e+13 | ns   | 2017-02-25T06:42:55.907000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.545769e+13 | ns   | 2017-02-25T06:41:55.899000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.545424e+13 | ns   | 2017-02-25T06:40:55.886000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.545079e+13 | ns   | 2017-02-25T06:39:55.988000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.544739e+13 | ns   | 2017-02-25T06:38:55.881000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.544396e+13 | ns   | 2017-02-25T06:37:55.895000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | cpu  | cumulative | 3.544071e+13 | ns   | 2017-02-25T06:37:00.784000 |
```

FIG.4.11 (CPU sample list for Cirros Test)



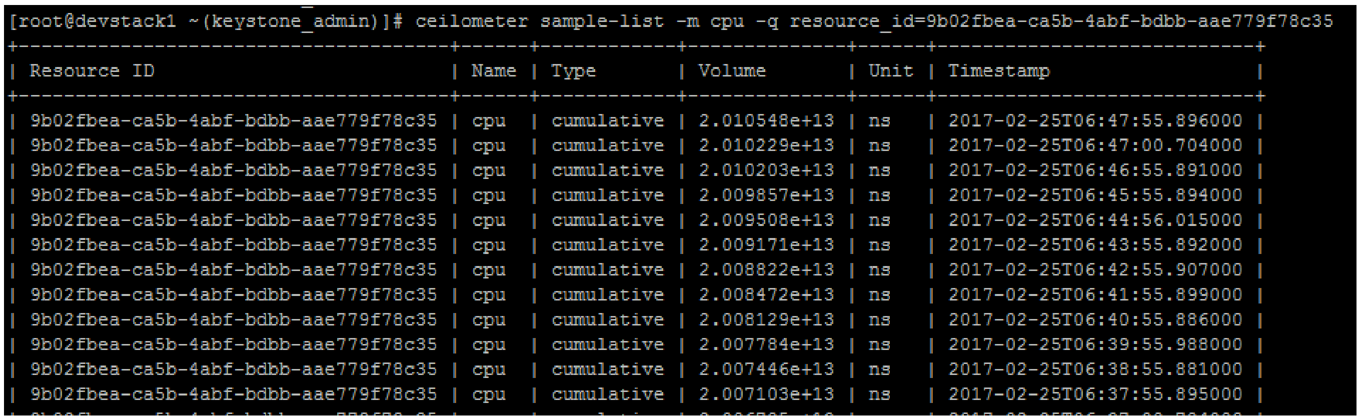
```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m cpu -q resource_id=40868588-d52a-4833-956d-fe469a1
+--------------------------------------+------+------------+--------------+------+----------------------------+
| Resource ID                          | Name | Type       | Volume       | Unit | Timestamp                  |
+--------------------------------------+------+------------+--------------+------+----------------------------+
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.036539e+13 | ns   | 2017-02-25T06:51:55.905000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.036189e+13 | ns   | 2017-02-25T06:50:56.240000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.035849e+13 | ns   | 2017-02-25T06:49:56.408000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.0355e+13   | ns   | 2017-02-25T06:48:55.914000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.035153e+13 | ns   | 2017-02-25T06:47:55.896000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.034828e+13 | ns   | 2017-02-25T06:47:00.704000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.034803e+13 | ns   | 2017-02-25T06:46:55.891000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.034453e+13 | ns   | 2017-02-25T06:45:55.894000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.034113e+13 | ns   | 2017-02-25T06:44:56.015000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.033775e+13 | ns   | 2017-02-25T06:43:55.892000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | cpu  | cumulative | 2.033414e+13 | ns   | 2017-02-25T06:42:55.907000 |
```

FIG.4.12 (CPU sample list for Test-1)



```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m cpu -q resource_id=5dd090d0-bffe-4988-8dfe-b9d8105
+--------------------------------------+------+------------+--------------+------+----------------------------+
| Resource ID                          | Name | Type       | Volume       | Unit | Timestamp                  |
+--------------------------------------+------+------------+--------------+------+----------------------------+
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.029656e+13 | ns   | 2017-02-25T06:57:00.708000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.029632e+13 | ns   | 2017-02-25T06:56:55.909000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.029289e+13 | ns   | 2017-02-25T06:55:55.999000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.028941e+13 | ns   | 2017-02-25T06:54:55.922000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.028608e+13 | ns   | 2017-02-25T06:53:55.904000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.028257e+13 | ns   | 2017-02-25T06:52:55.904000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.027908e+13 | ns   | 2017-02-25T06:51:55.905000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.02756e+13  | ns   | 2017-02-25T06:50:56.240000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.027224e+13 | ns   | 2017-02-25T06:49:56.408000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.026879e+13 | ns   | 2017-02-25T06:48:55.914000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.02653e+13  | ns   | 2017-02-25T06:47:55.896000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.026207e+13 | ns   | 2017-02-25T06:47:00.704000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.026182e+13 | ns   | 2017-02-25T06:46:55.891000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.025833e+13 | ns   | 2017-02-25T06:45:55.894000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | cpu  | cumulative | 2.025491e+13 | ns   | 2017-02-25T06:44:56.015000 |
```

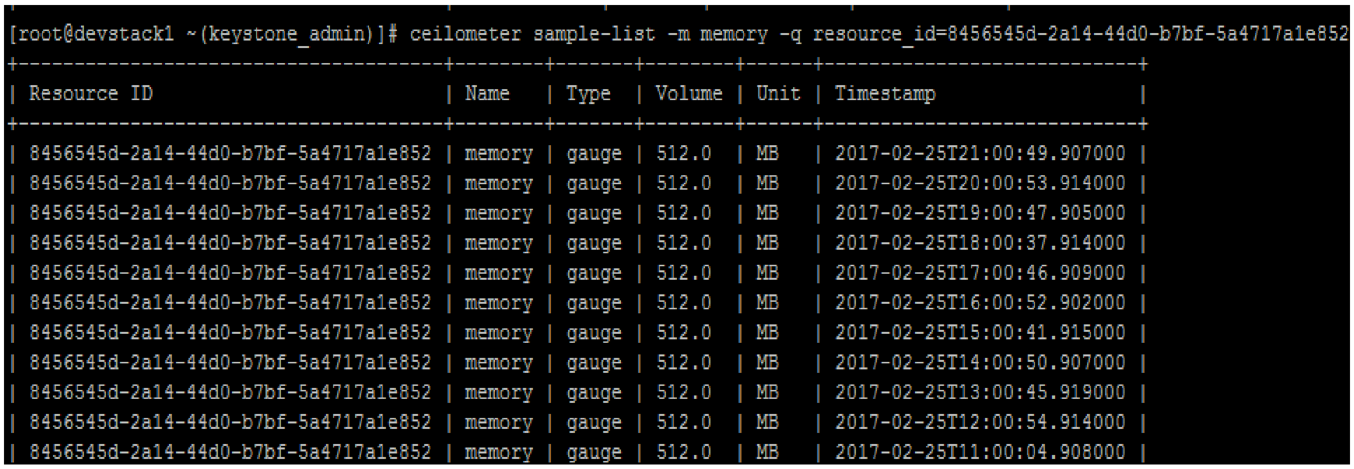
FIG.4.13 (CPU sample list for Test-2)



```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m cpu -q resource_id=9b02fbea-ca5b-4abf-bdbb-aae779f78c35
+--------------------------------------+------+------------+--------------+------+----------------------------+
| Resource ID                          | Name | Type       | Volume       | Unit | Timestamp                  |
+--------------------------------------+------+------------+--------------+------+----------------------------+
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.010548e+13 | ns   | 2017-02-25T06:47:55.896000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.010229e+13 | ns   | 2017-02-25T06:47:00.704000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.010203e+13 | ns   | 2017-02-25T06:46:55.891000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.009857e+13 | ns   | 2017-02-25T06:45:55.894000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.009508e+13 | ns   | 2017-02-25T06:44:56.015000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.009171e+13 | ns   | 2017-02-25T06:43:55.892000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.008822e+13 | ns   | 2017-02-25T06:42:55.907000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.008472e+13 | ns   | 2017-02-25T06:41:55.899000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.008129e+13 | ns   | 2017-02-25T06:40:55.886000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.007784e+13 | ns   | 2017-02-25T06:39:55.988000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.007446e+13 | ns   | 2017-02-25T06:38:55.881000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | cpu  | cumulative | 2.007103e+13 | ns   | 2017-02-25T06:37:55.895000 |
```

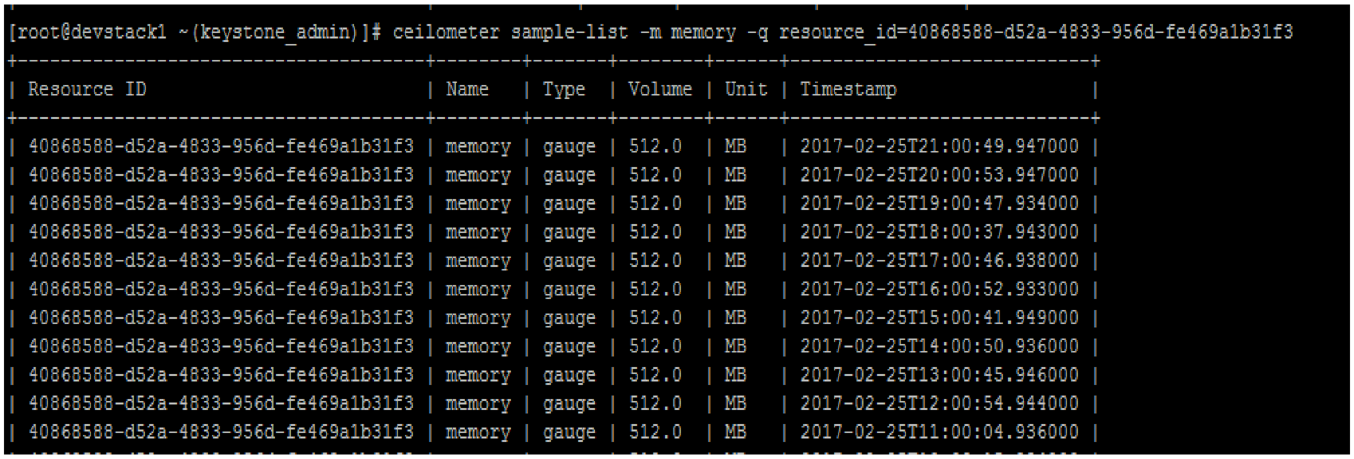
FIG.4.14 (CPU sample list for Test-3)

Further outputs are the sample list of the memory usage in the FIG 4.15, FIG 4.16, FIG 4.17 and FIG 4.18 of each instance monitored and metered by ceilometer to get the memory from every instance that is defined in Megabytes. This is the memory usage done by different instances. The metering type is gauge since it is discrete. As shown in the outputs is memory usage for all instances are the same as all the instances are using the same flavor that ends up with same memory usage.



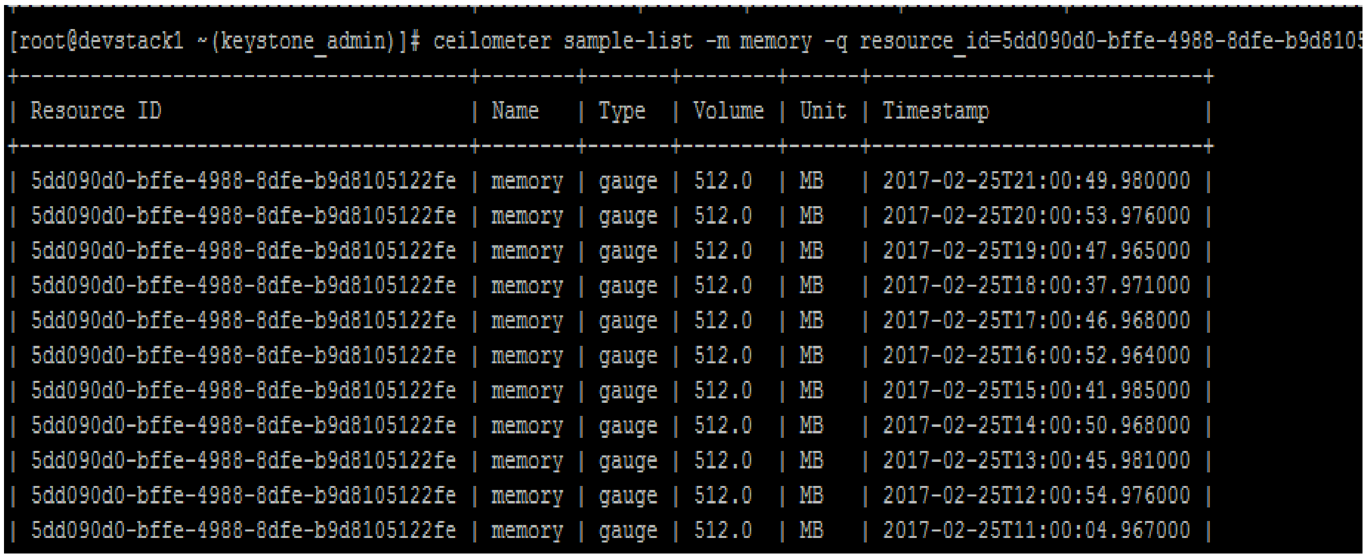
```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m memory -q resource_id=8456545d-2a14-44d0-b7bf-5a4717a1e852
+--------------------------------------+--------+-------+--------+------+----------------------------+
| Resource ID                          | Name   | Type  | Volume | Unit | Timestamp                  |
+--------------------------------------+--------+-------+--------+------+----------------------------+
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T21:00:49.907000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T20:00:53.914000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T19:00:47.905000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T18:00:37.914000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T17:00:46.909000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T16:00:52.902000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T15:00:41.915000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T14:00:50.907000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T13:00:45.919000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T12:00:54.914000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | memory | gauge | 512.0  | MB   | 2017-02-25T11:00:04.908000 |
```

FIG.4.15 (Memory sample list for Cirros Test)



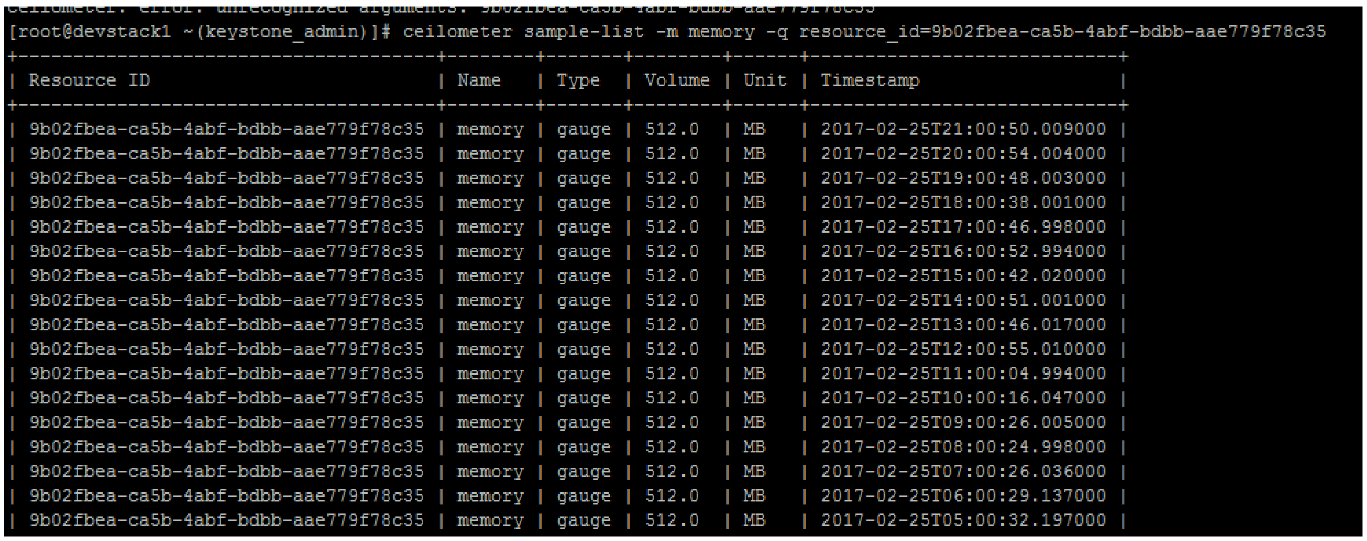
```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m memory -q resource_id=40868588-d52a-4833-956d-fe469a1b31f3
+--------------------------------------+--------+-------+--------+------+----------------------------+
| Resource ID                          | Name   | Type  | Volume | Unit | Timestamp                  |
+--------------------------------------+--------+-------+--------+------+----------------------------+
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T21:00:49.947000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T20:00:53.947000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T19:00:47.934000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T18:00:37.943000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T17:00:46.938000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T16:00:52.933000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T15:00:41.949000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T14:00:50.936000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T13:00:45.946000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T12:00:54.944000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | memory | gauge | 512.0  | MB   | 2017-02-25T11:00:04.936000 |
```

FIG.4.16 (Memory sample-list for Test-1)



```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m memory -q resource_id=5dd090d0-bffe-4988-8dfe-b9d8105
+--------------------------------------+--------+-------+--------+------+----------------------------+
| Resource ID                          | Name   | Type  | Volume | Unit | Timestamp                  |
+--------------------------------------+--------+-------+--------+------+----------------------------+
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T21:00:49.980000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T20:00:53.976000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T19:00:47.965000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T18:00:37.971000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T17:00:46.968000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T16:00:52.964000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T15:00:41.985000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T14:00:50.968000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T13:00:45.981000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T12:00:54.976000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | memory | gauge | 512.0  | MB   | 2017-02-25T11:00:04.967000 |
```

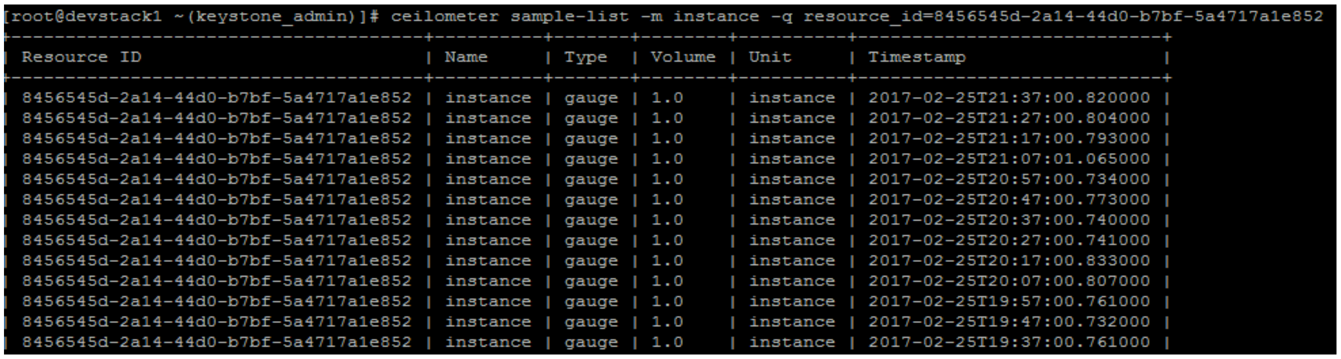
FIG.4.17 (Memory sample-list for Test-2)



```
ceilometer: error: unrecognized arguments: 9b02fbea-ca5b-4abf-bdbb-aae779f78c35
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m memory -q resource_id=9b02fbea-ca5b-4abf-bdbb-aae779f78c35
+--------------------------------------+--------+-------+--------+------+----------------------------+
| Resource ID                          | Name   | Type  | Volume | Unit | Timestamp                  |
+--------------------------------------+--------+-------+--------+------+----------------------------+
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T21:00:50.009000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T20:00:54.004000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T19:00:48.003000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T18:00:38.001000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T17:00:46.998000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T16:00:52.994000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T15:00:42.020000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T14:00:51.001000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T13:00:46.017000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T12:00:55.010000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T11:00:04.994000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T10:00:16.047000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T09:00:26.005000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T08:00:24.998000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T07:00:26.036000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T06:00:29.137000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | memory | gauge | 512.0  | MB   | 2017-02-25T05:00:32.197000 |
```

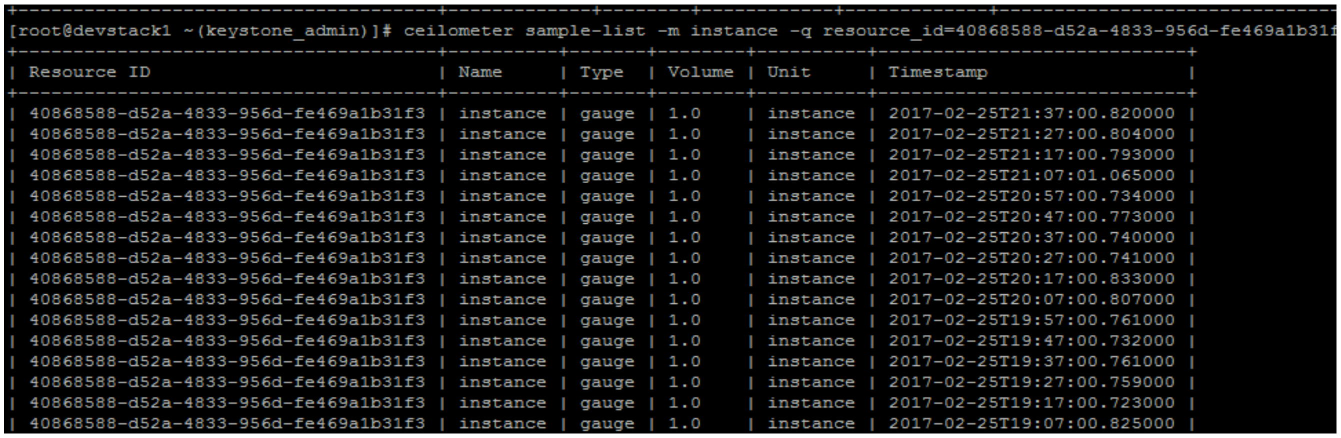
FIG.4.18 (Memory sample-list for Test-3)

Moreover, the next metering is for volume instances, the unit for this volume remains to be the instance since instance will remain to be the instance and has no particular unit. The flowing FIG 4.15, FIG 4.16, FIG 4.17 and FIG 4.18 are the outputs



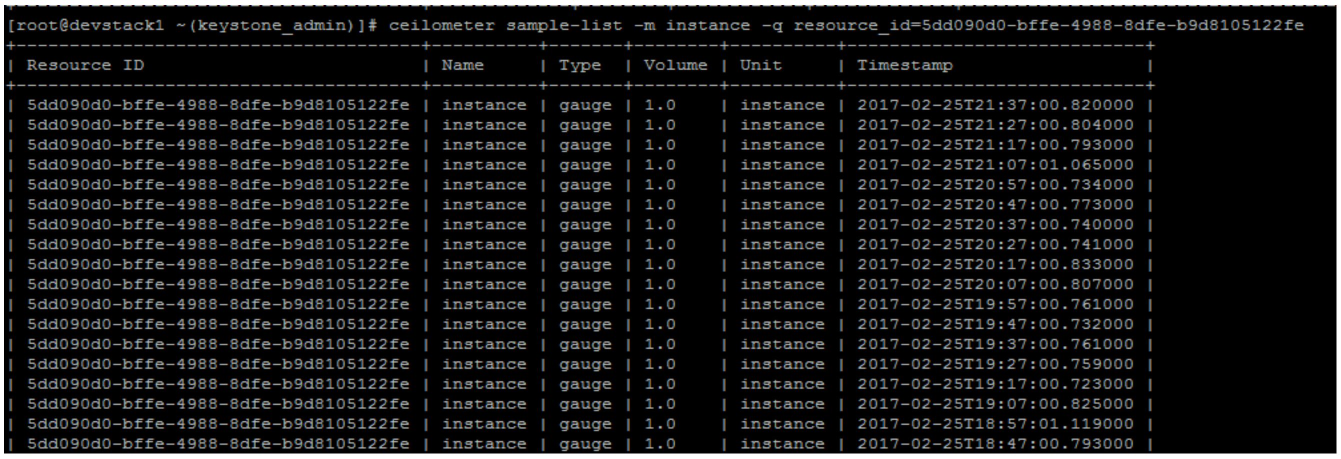
```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m instance -q resource_id=8456545d-2a14-44d0-b7bf-5a4717a1e852
+--------------------------------------+----------+-------+--------+----------+----------------------------+
| Resource ID                          | Name     | Type  | Volume | Unit     | Timestamp                  |
+--------------------------------------+----------+-------+--------+----------+----------------------------+
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T21:37:00.820000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T21:27:00.804000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T21:17:00.793000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T21:07:01.065000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T20:57:00.734000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T20:47:00.773000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T20:37:00.740000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T20:27:00.741000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T20:17:00.833000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T20:07:00.807000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T19:57:00.761000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T19:47:00.732000 |
| 8456545d-2a14-44d0-b7bf-5a4717a1e852 | instance | gauge | 1.0    | instance | 2017-02-25T19:37:00.761000 |
```

FIG.4.19 (Instance sample-list for Cirros Test)



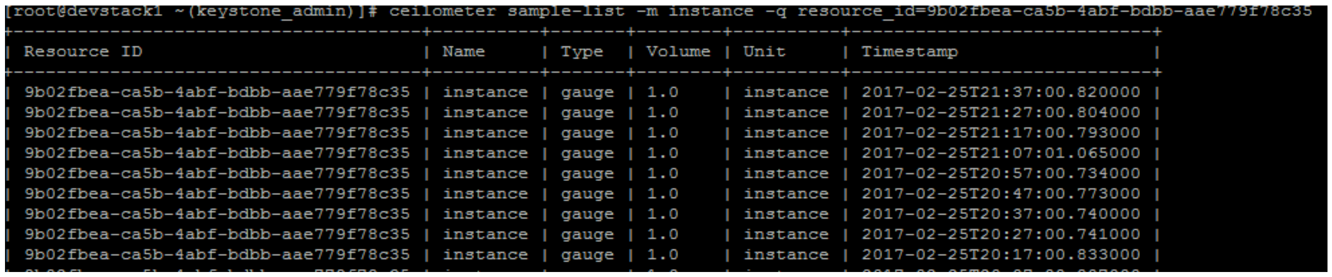
```
+--------------------------------------+----------+-------+--------+----------+----------------------------+
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m instance -q resource_id=40868588-d52a-4833-956d-fe469a1b31f
+--------------------------------------+----------+-------+--------+----------+----------------------------+
| Resource ID                          | Name     | Type  | Volume | Unit     | Timestamp                  |
+--------------------------------------+----------+-------+--------+----------+----------------------------+
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T21:37:00.820000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T21:27:00.804000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T21:17:00.793000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T21:07:01.065000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T20:57:00.734000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T20:47:00.773000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T20:37:00.740000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T20:27:00.741000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T20:17:00.833000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T20:07:00.807000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T19:57:00.761000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T19:47:00.732000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T19:37:00.761000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T19:27:00.759000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T19:17:00.723000 |
| 40868588-d52a-4833-956d-fe469a1b31f3 | instance | gauge | 1.0    | instance | 2017-02-25T19:07:00.825000 |
```

FIG.4.20 (Instance sample-list for Test-1)



```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m instance -q resource_id=5dd090d0-bffe-4988-8dfe-b9d8105122fe
+--------------------------------------+----------+-------+--------+----------+----------------------------+
| Resource ID                          | Name     | Type  | Volume | Unit     | Timestamp                  |
+--------------------------------------+----------+-------+--------+----------+----------------------------+
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T21:37:00.820000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T21:27:00.804000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T21:17:00.793000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T21:07:01.065000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T20:57:00.734000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T20:47:00.773000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T20:37:00.740000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T20:27:00.741000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T20:17:00.833000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T20:07:00.807000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T19:57:00.761000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T19:47:00.732000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T19:37:00.761000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T19:27:00.759000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T19:17:00.723000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T19:07:00.825000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T18:57:01.119000 |
| 5dd090d0-bffe-4988-8dfe-b9d8105122fe | instance | gauge | 1.0    | instance | 2017-02-25T18:47:00.793000 |
```

FIG.4.21 (Instance sample-list for Test-2)



```
[root@devstack1 ~(keystone_admin)]# ceilometer sample-list -m instance -q resource_id=9b02fbea-ca5b-4abf-bdbb-aae779f78c35
+--------------------------------------+----------+-------+--------+----------+----------------------------+
| Resource ID                          | Name     | Type  | Volume | Unit     | Timestamp                  |
+--------------------------------------+----------+-------+--------+----------+----------------------------+
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | instance | gauge | 1.0    | instance | 2017-02-25T21:37:00.820000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | instance | gauge | 1.0    | instance | 2017-02-25T21:27:00.804000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | instance | gauge | 1.0    | instance | 2017-02-25T21:17:00.793000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | instance | gauge | 1.0    | instance | 2017-02-25T21:07:01.065000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | instance | gauge | 1.0    | instance | 2017-02-25T20:57:00.734000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | instance | gauge | 1.0    | instance | 2017-02-25T20:47:00.773000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | instance | gauge | 1.0    | instance | 2017-02-25T20:37:00.740000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | instance | gauge | 1.0    | instance | 2017-02-25T20:27:00.741000 |
| 9b02fbea-ca5b-4abf-bdbb-aae779f78c35 | instance | gauge | 1.0    | instance | 2017-02-25T20:17:00.833000 |
```

FIG.4.22 (Instance sample-list for Test-3)

Chapter 5 Heat Orchestration

5.1 Brief Details of Heat

Heat is the main project for orchestration part of OpenStack. Implementation of orchestration engine for multiple composite cloud application. It is the sequence of lines code in text file format. A native heat format can be evolving, but heat also endeavors to provide compatibility with AWS cloud information template format so that many existing cloud formation template can be launch on OpenStack. Heat provide both open stack rest API and cloud formation compatible query API. The orchestration is essentially for the software application. To manage configuration. Instead of manipulation of virtual infrastructure by hand or with the script

Heat focuses to work with the declarative model. Heat works out on the sequence of lines to perform and to bring reality in to model. The model takes the heat template and the resulting collective of infrastructure resources is known as the stack. Orchestration allows you to treat your infrastructure like code. Therefore you can store your templates version control system, such as GIT to track changes then you update the stack with the new template and heat do the rest of the actions. The main interface of heat is the OpenStack native rest API. Heat actually is between the user and the API of the core OpenStack services. In much the same way as the dashboard or the horizon does. Heat can be access through the horizon or the dashboard. Heat template describes the cloud application infrastructure in the code format that is changeable. The heat infrastructure resources include servers, floating IP, volume, security groups, and users.

5.2 Auto Scaling

Heat also provides auto scaling that integrates with ceilometer. Ceilometer adds scaling group as the resource within templates. Furthermore, the template, defines the relationship between two Auto scaling by heat integrated with ceilometer that leads to add scaling group in template. The templates defines the relationship between two resources. It also able heat to call OpenStack API in order to make everything systematic. Openstack also manage the whole lifecycle of the application. You need to do the modification in the code for existing stack and heat deals with the rest in order to change something.

Heat architecture components include:

Heat API It is used for processing API request to Heat engine via AMQP. It implements an Open stack-native RESTful API

HEAT-api-cfn it is used API compatibility with AWS cloud formation.

HEAT ENGINE is main orchestration functionality.

Heat uses back-end database for maintaining state information as other OpenStack services. Both communicate with heat engine via ANQ. The heat engine is the actual layer where actual integration is implemented. Furthermore, for high availability, Auto scaling abstraction is also done.

Auto Scaling Heat Templates

Heat and Ceilometer are the service that are used to scale CPU bound virtual machines. Heat stack is the environment itself. Heat stack manages processes and logics defined in templates. Auto Scale creation group and Ceilometer threshold is also defined in template. The environment template explains how to create the stack itself, what image or volume to use, network configuration, Software to install and everything an instance or instances need to properly function. All of these things can be in Heat Stack template

5.3 Deployment of Heat Orchestration

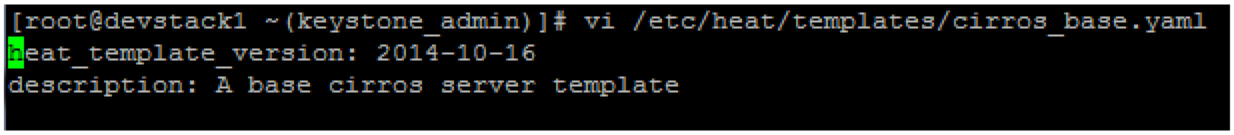
5.3.1 ENVIRONMENT TEMPLATE

Below we will create an environment template for a cirros image. As shown in FIG.5.3. The Cirros image will create the instance template, configure a cinder volume, add IP from the private network, add floating IP from the public network, add the security group, private ssh-key and generate 100% CPU load through user-data.

Hot is the new template format that to replace the Heat Cloud Formation-Compatible format as native format supported by heat. They are written in YAML format and JSON. Hot templates create

Stack in Heat. Structure for Hot consist of Heat Template version, description, parameter groups, parameters, resources, and outputs.

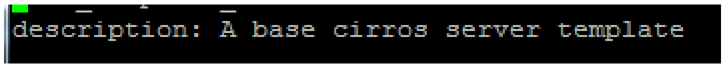
- Heat Template Version: Is just value with the key that indicates that the YAML document is a hot template of the specific version, if the date is 2013-05-23 or later date.Shown in FIG 5.1



```
[root@devstack1 ~(keystone_admin)]# vi /etc/heat/templates/cirros_base.yaml
heat_template_version: 2014-10-16
description: A base cirros server template
```

Fig.5.1 (Heat Template version)

- **Description:** It's an optional key allows for giving a description of the template.



```
description: A base cirros server template
```

Fig.5.2 (Description of heat template)

- **Parameters groups:** This section allows for specifying how the input parameters should be grouped and order to provide the parameter in. This option is also optional
- **Parameter**: This section allows for specifying input parameters that have to provide when instantiating the templates. This option is also optional as well
- **Outputs:** This part allows for specifying output parameters available to users once the template has been instantiated.
- **Resources**: It defines actual resources that are real stack from HOT template (instance for Compute, Network, and Storage Volume).Each resource is defined as a separate block in input parameters. As shown in FIG 5.3 there are five separate sections. Servers, port, volume, floating IP
  - **Resource ID**: must always be unique for every section
  - **Resource Types**: Must relate to the service that section of template define Such as the following

    Nova:: Server

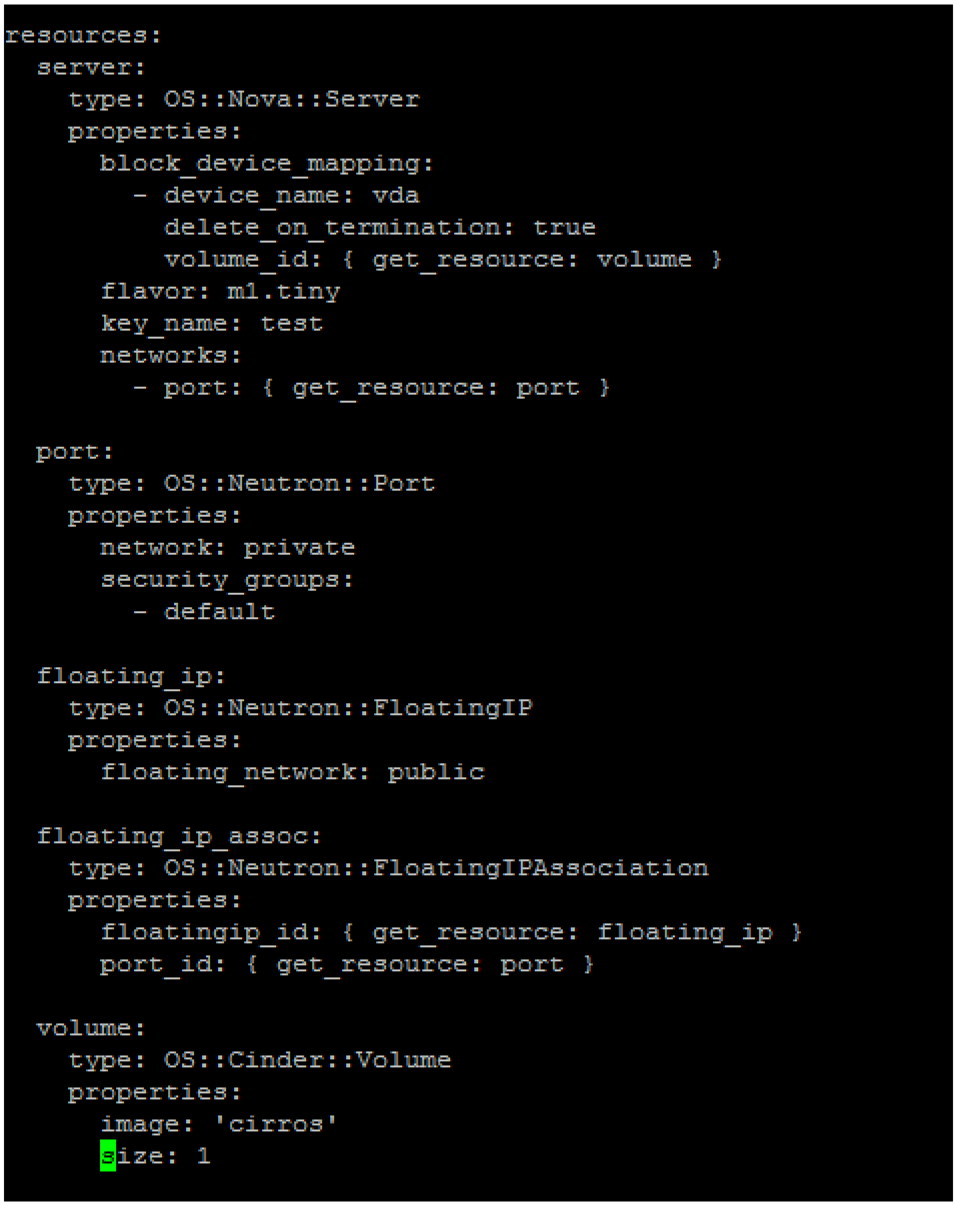
Neutron:: Port

Neutron:: FloatingIP

Neutron:: FloatingIPAssociation

Cinder:: Volume

- **Properties**: It is a list of resource specific property defines via the function.



```
resources:
  server:
    type: OS::Nova::Server
    properties:
      block_device_mapping:
        - device_name: vda
          delete_on_termination: true
          volume_id: { get_resource: volume }
      flavor: m1.tiny
      key_name: test
      networks:
        - port: { get_resource: port }

  port:
    type: OS::Neutron::Port
    properties:
      network: private
      security_groups:
        - default

  floating_ip:
    type: OS::Neutron::FloatingIP
    properties:
      floating_network: public

  floating_ip_assoc:
    type: OS::Neutron::FloatingIPAssociation
    properties:
      floatingip_id: { get_resource: floating_ip }
      port_id: { get_resource: port }

  volume:
    type: OS::Cinder::Volume
    properties:
      image: 'cirros'
      size: 1
```

FIG.5.3 (Heat Template Resources)

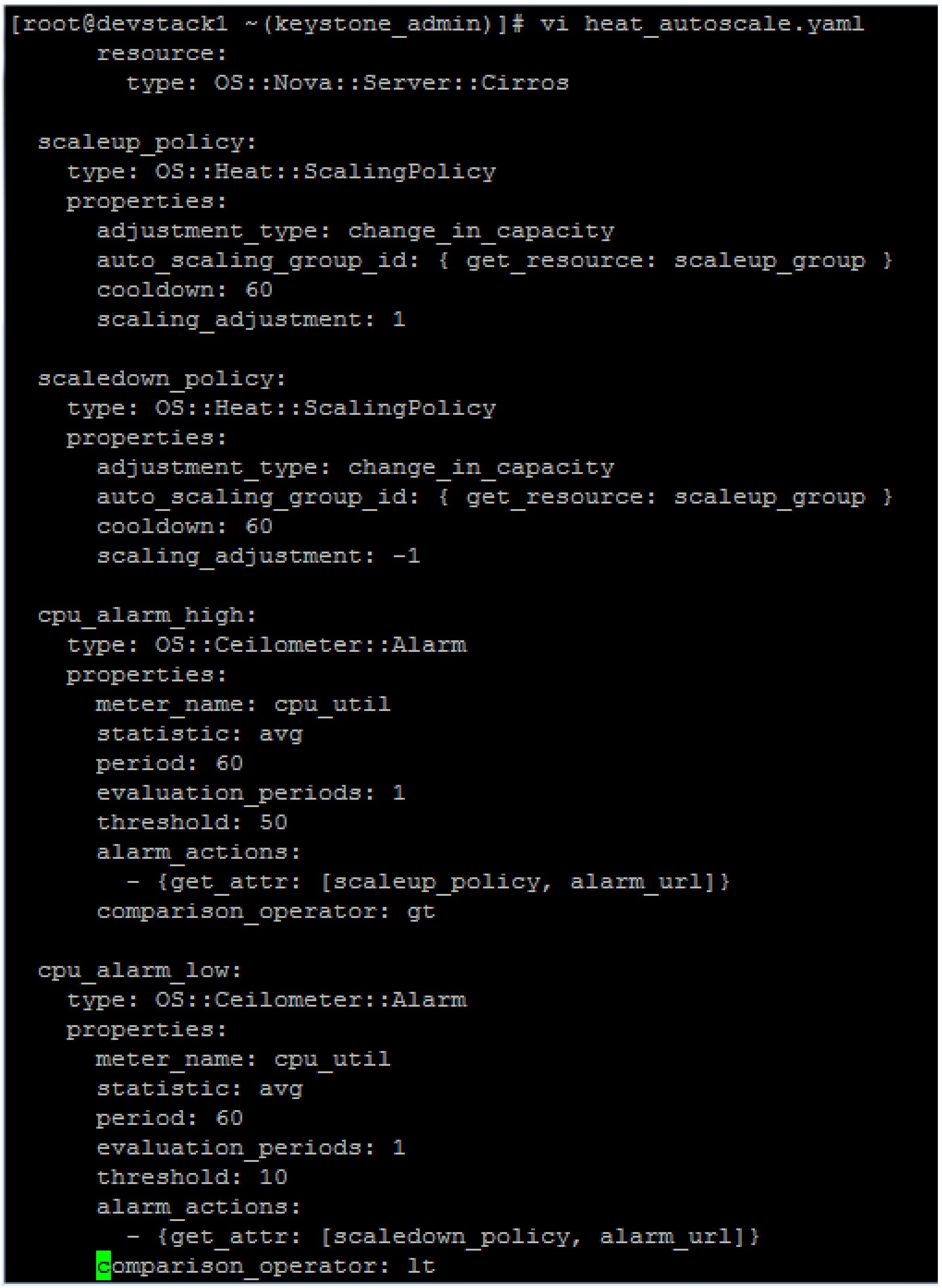
Now that we have an environment template, we need to create a Heat resource type and link it above file /etc/heat/templates/cirros_base.yaml.

resource_registry:

"OS::Nova::Server::Cirros": file:///etc/heat/templates/cirros_base.yaml

5.3.2 Heat Template:

The below template in FIG.5.4 defines the behavior of the stack e.g. when and under what conditions the stack will scale up and scale down. cpu_alarm_high and cpu_alarm_low are used in the template to scale up and scale down our environment.



```
[root@devstack1 ~(keystone_admin)]# vi heat_autoscale.yaml
      resource:
        type: OS::Nova::Server::Cirros

  scaleup_policy:
    type: OS::Heat::ScalingPolicy
    properties:
      adjustment_type: change_in_capacity
      auto_scaling_group_id: { get_resource: scaleup_group }
      cooldown: 60
      scaling_adjustment: 1

  scaledown_policy:
    type: OS::Heat::ScalingPolicy
    properties:
      adjustment_type: change_in_capacity
      auto_scaling_group_id: { get_resource: scaleup_group }
      cooldown: 60
      scaling_adjustment: -1

  cpu_alarm_high:
    type: OS::Ceilometer::Alarm
    properties:
      meter_name: cpu_util
      statistic: avg
      period: 60
      evaluation_periods: 1
      threshold: 50
      alarm_actions:
        - {get_attr: [scaleup_policy, alarm_url]}
      comparison_operator: gt

  cpu_alarm_low:
    type: OS::Ceilometer::Alarm
    properties:
      meter_name: cpu_util
      statistic: avg
      period: 60
      evaluation_periods: 1
      threshold: 10
      alarm_actions:
        - {get_attr: [scaledown_policy, alarm_url]}
      comparison_operator: lt
```

FIG.5.4 (Behavior Of Stack)

Update Ceilometer Collection Interval

By default, Ceilometer will collect CPU data from instances every 10 minutes. For this example, we want to change that to 60 seconds. Change the interval to 60 in the pipeline.YAML file and restart

OpenStack services.

Check the status of the stack in Horizon Dashboard:

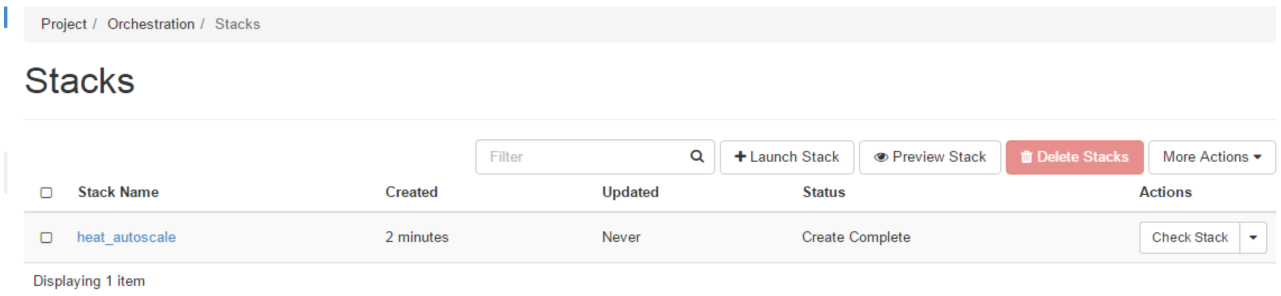
Heat will create one instance as per defined policy:

5.3.3 RUNNING THE STACK:

Run the following command to run the stack:

```
[root@devstack1 ~ (keystone_admin)]# heat stack-create heat_autoscale -f /root/heat_autoscale.YAML -e /root/environment.yaml
```

Check the status of the stack in Horizon Dashboard as in FIG 5.4:



Stack Name	Created	Updated	Status	Actions
heat_autoscale	2 minutes	Never	Create Complete	Check Stack

Displaying 1 item

FIG5.5 (Heat stack status)

In FIG 5.5 and FIG 5.6 shows the heat stack topology and resources, Events are also shown in FIG 5.6

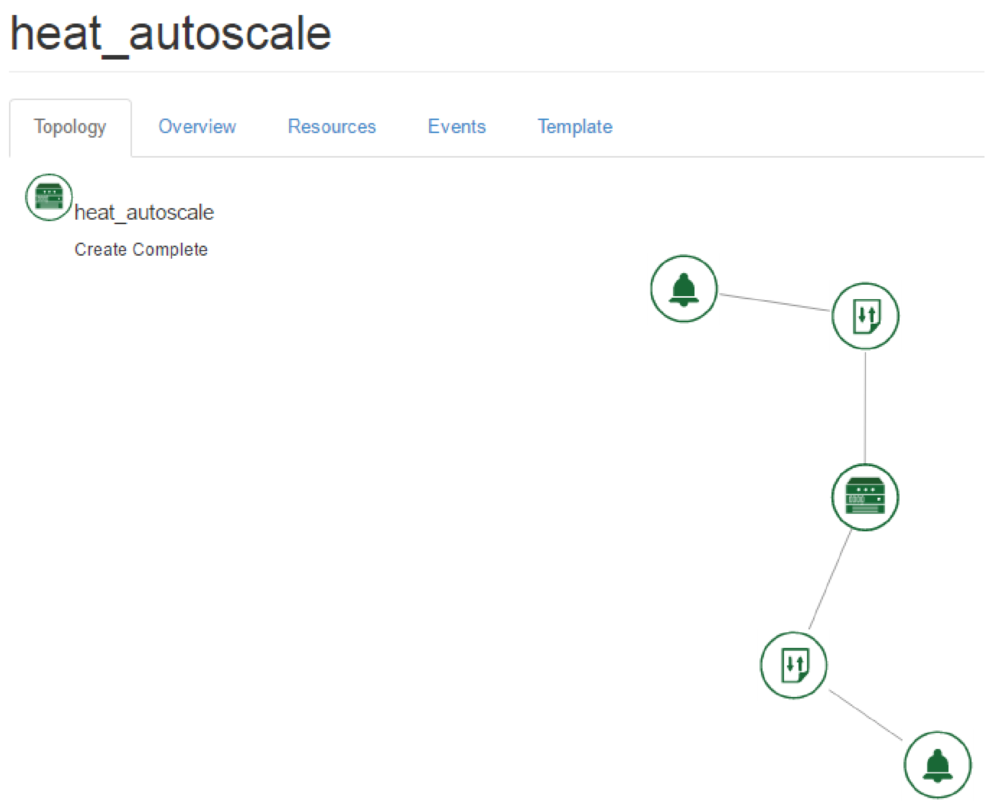
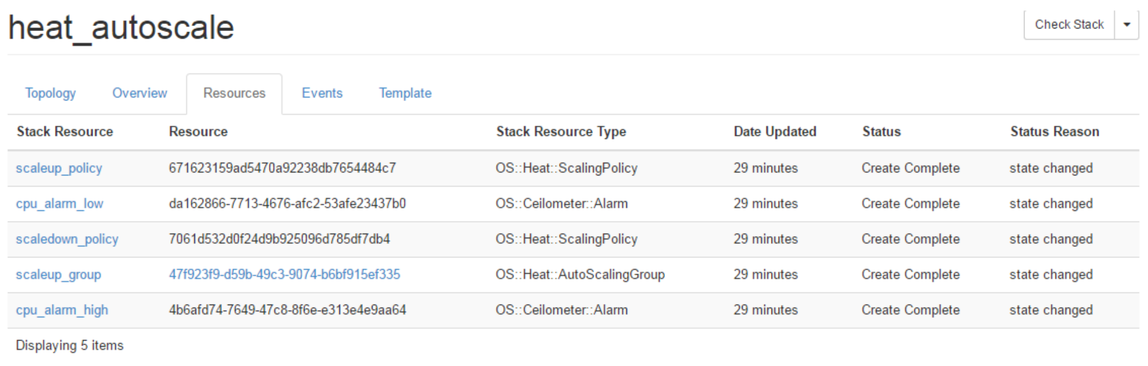


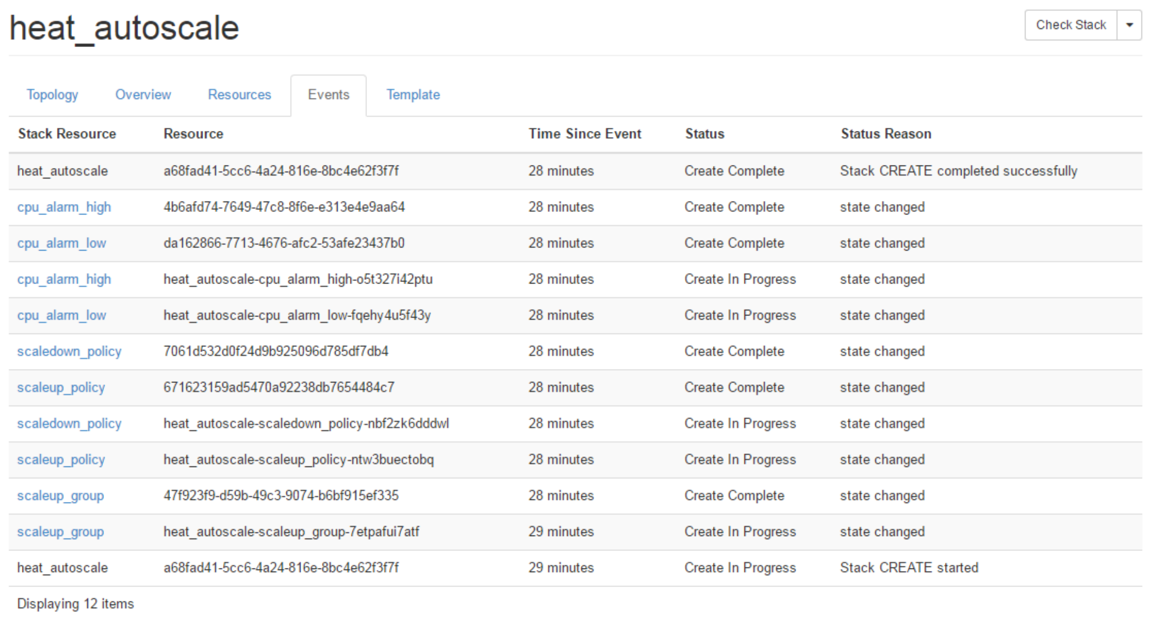
FIG 5.6 (Heat Stack Topology)


heat_autoscale

Stack Resource	Resource	Stack Resource Type	Date Updated	Status	Status Reason
scaleup_policy	671623159ad5470a92238db7654484c7	OS::Heat::ScalingPolicy	29 minutes	Create Complete	state changed
cpu_alarm_low	da162866-7713-4676-afc2-53afe23437b0	OS::Ceilometer::Alarm	29 minutes	Create Complete	state changed
scaledown_policy	7061d532d0f24d9b925096d785df7db4	OS::Heat::ScalingPolicy	29 minutes	Create Complete	state changed
scaleup_group	47f923f9-d59b-49c3-9074-b6bf915ef335	OS::Heat::AutoScalingGroup	29 minutes	Create Complete	state changed
cpu_alarm_high	4b6afd74-7649-47c8-8f6e-e313e4e9aa64	OS::Ceilometer::Alarm	29 minutes	Create Complete	state changed

Displaying 5 items

FIG 5.7 (Heat Stack Resources)


heat_autoscale

Stack Resource	Resource	Time Since Event	Status	Status Reason
heat_autoscale	a68fad41-5cc6-4a24-816e-8bc4e62f3f7f	28 minutes	Create Complete	Stack CREATE completed successfully
cpu_alarm_high	4b6afd74-7649-47c8-8f6e-e313e4e9aa64	28 minutes	Create Complete	state changed
cpu_alarm_low	da162866-7713-4676-afc2-53afe23437b0	28 minutes	Create Complete	state changed
cpu_alarm_high	heat_autoscale-cpu_alarm_high-o5t327i42ptu	28 minutes	Create In Progress	state changed
cpu_alarm_low	heat_autoscale-cpu_alarm_low-fqehy4u5f43y	28 minutes	Create In Progress	state changed
scaledown_policy	7061d532d0f24d9b925096d785df7db4	28 minutes	Create Complete	state changed
scaleup_policy	671623159ad5470a92238db7654484c7	28 minutes	Create Complete	state changed
scaledown_policy	heat_autoscale-scaledown_policy-nbf2zk6dddwl	28 minutes	Create In Progress	state changed
scaleup_policy	heat_autoscale-scaleup_policy-ntw3buectobq	28 minutes	Create In Progress	state changed
scaleup_group	47f923f9-d59b-49c3-9074-b6bf915ef335	28 minutes	Create Complete	state changed
scaleup_group	heat_autoscale-scaleup_group-7etpafui7atf	29 minutes	Create In Progress	state changed
heat_autoscale	a68fad41-5cc6-4a24-816e-8bc4e62f3f7f	29 minutes	Create In Progress	Stack CREATE started

Displaying 12 items

FIG 5.8 (Heat Stack Events)

Heat will create one instance as per defined policy in FIG 5.7:

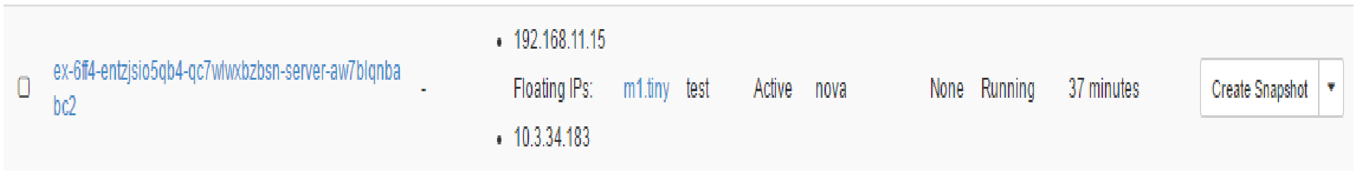
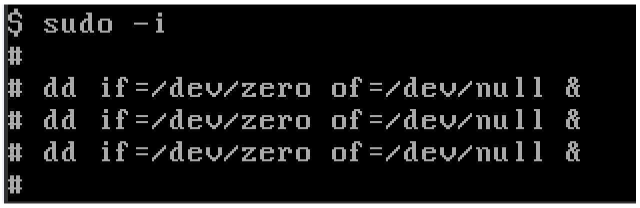

ex-6ff4-entzjsio5qb4-qc7wlwxbzbsn-server-aw7blqnba bc2 - • 192.168.11.15 Floating IPs: • 10.3.34.183 m1.tiny test Active nova None Running 37 minutes Create Snapshot

FIG 5.9 (Heat Stack Instance)

Automatic Scale UP:

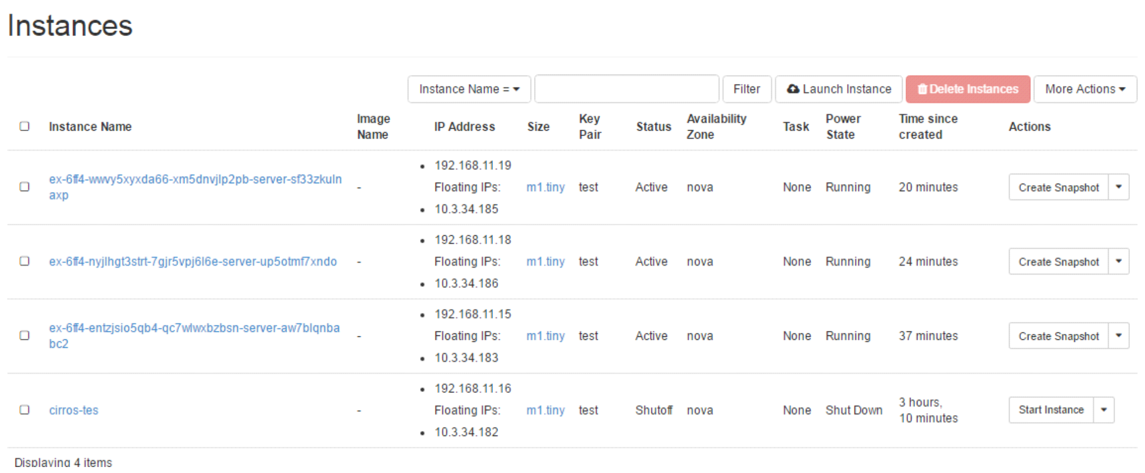
Now we will increase the cpu utilization on one of the instances and will verify if heat autoscales the environment or not. To do that run the following commands one the instance that heat created from the stack. As shown in FIG 5.8.



```
$ sudo -i
#
# dd if=/dev/zero of=/dev/null &
# dd if=/dev/zero of=/dev/null &
# dd if=/dev/zero of=/dev/null &
#
```

FIG 5.10 (Heat Autoscaling)

The heat created two more instances based defined policy in the orchestration template. This is because the maximum scale up policy is 3 instances. As shown in FIG 5.8.



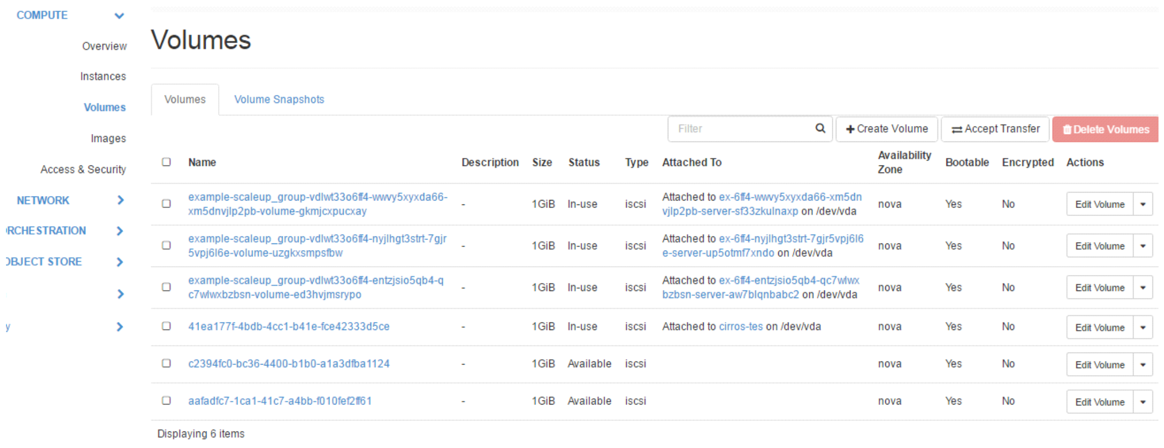
Instances

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
ex-6ff4-wwvy5xyxda66-xm5dnvjlp2pb-server-sf33zkulnaxp	-	• 192.168.11.19 Floating IPs: • 10.3.34.185	m1.tiny	test	Active	nova	None	Running	20 minutes	Create Snapshot
ex-6ff4-nyjlhgt3strt-7gjr5vpj6l6e-server-up5otmf7xndo	-	• 192.168.11.18 Floating IPs: • 10.3.34.186	m1.tiny	test	Active	nova	None	Running	24 minutes	Create Snapshot
ex-6ff4-entzjsio5qb4-qc7wlwxbzbsn-server-aw7blqnbabc2	-	• 192.168.11.15 Floating IPs: • 10.3.34.183	m1.tiny	test	Active	nova	None	Running	37 minutes	Create Snapshot
cirros-tes	-	• 192.168.11.16 Floating IPs: • 10.3.34.182	m1.tiny	test	Shutoff	nova	None	Shut Down	3 hours, 10 minutes	Start Instance

Displaying 4 items

FIG 5.11 (Two Instance base on policy)

List of volumes that heat created based on defined policy threshold as shown in FIG 5.9:



Volumes

	Name	Description	Size	Status	Type	Attached To	Availability Zone	Bootable	Encrypted	Actions
☐	example-scaleup_group-vdlwt33o6ff4-wwvy5xyxda66-xm5dnvjlp2pb-volume-gkmjcxpucxay	-	1GiB	In-use	iscsi	Attached to ex-6ff4-wwvy5xyxda66-xm5dnvjlp2pb-server-sf33zkulnaxp on /dev/vda	nova	Yes	No	Edit Volume
☐	example-scaleup_group-vdlwt33o6ff4-nyjlhgt3strt-7gjr5vpj6l6e-volume-uzgkxsmpsfbw	-	1GiB	In-use	iscsi	Attached to ex-6ff4-nyjlhgt3strt-7gjr5vpj6l6e-server-up5otmf7xndo on /dev/vda	nova	Yes	No	Edit Volume
☐	example-scaleup_group-vdlwt33o6ff4-entzjsio5qb4-qc7wlwxbzbsn-volume-ed3hvjmsrypo	-	1GiB	In-use	iscsi	Attached to ex-6ff4-entzjsio5qb4-qc7wlwxbzbsn-server-aw7blqnbabc2 on /dev/vda	nova	Yes	No	Edit Volume
☐	41ea177f-4bdb-4cc1-b41e-fce42333d5ce	-	1GiB	In-use	iscsi	Attached to cirros-tes on /dev/vda	nova	Yes	No	Edit Volume
☐	c2394fc0-bc36-4400-b1b0-a1a3dfba1124	-	1GiB	Available	iscsi		nova	Yes	No	Edit Volume
☐	aafadfc7-1ca1-41c7-a4bb-f010fef2ff61	-	1GiB	Available	iscsi		nova	Yes	No	Edit Volume

Displaying 6 items

FIG.5.12 (Volumes that heat created based on defined policy)

New Network Topology after adding instances to the private network in FIG 5.10

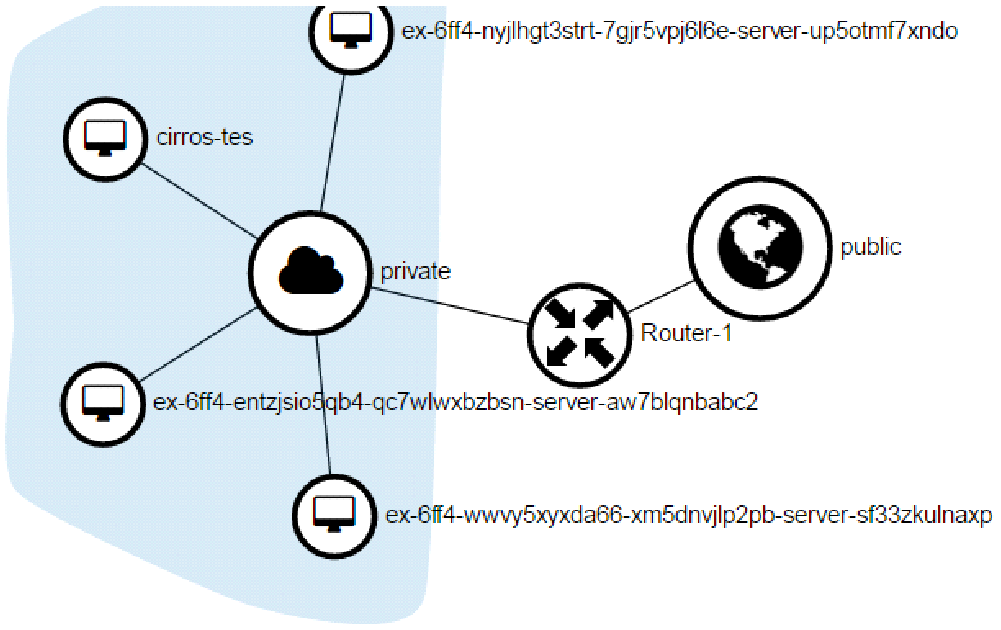

ex-6ff4-nyjlhgt3strt-7gjr5vpj6l6e-server-up5otmf7xndo
cirros-tes
private
public
Router-1
ex-6ff4-entzjsio5qb4-qc7wlwxbzbsn-server-aw7blqnbabc2
ex-6ff4-wwvy5xyxda66-xm5dnvjlp2pb-server-sf33zkulnaxp

FIG 5.13 (Heat Topology after 2 instances)

5.3.4 SCALE DOWN:

Scale down is the process in heat. Heat automatically scales down once the CPU utilization goes down on the instances. As the load goes back to normal and CPU cools down. The extra instances that were appeared to overcome the load will go back to one instance and all instances will be used efficiently through this way. In our scenario instance “aw7blqnbabc2” is the original instance and the rest instances are to overcome the load.

Chapter 6 Conclusion

Cloud Computing is one of the most powerful technology today in the IT industry. It is one of the greatest evolution in the industry. OpenStack is playing dominant role in cloud computing. The growing interest of different vendors in OpenStack not only making it more attractive, but also taking it to further stage of advancement. Openstack Elastic Cloud Service is also one of the most spectacular services for IT industry, where it is possible to scale up the resources under the required circumstances through heat services. Furthermore, it automatically scale down when CPU cools down and load comes back to normal, without any extra effort to be done. On the other side of the page ceilometer is helpful in metering and monitoring purpose. These both services with openstack itself a quality makes itself visible in many of the selections in IT industry

Chapter 7 Reference

1. https://en.wikipedia.org/wiki/OpenStack
2. linuxacademy.com/cp/modules/view/id/13(Module:Introduction to the Linux Academy)
3. https://linuxacademy.com/cp/modules/view/id/18(Module:OpenStack Essentials)
4. https://linuxacademy.com/cp/modules/view/id/61 (Modules:OpenStack Foundation Certified OpenStack Administrator)
5. https://linuxacademy.com/cp/modules/view/id/28(Module:Linux Academy Red Hat OpenStack Administrator Certification Prep Course)
6. https://docs.openstack.org/ops-guide/architecture.html
7. http://www.stratoscale.com/resources/campaigns/simplify-openstack/?network=g&device=c&placement=&keyword=%2Bopen%20%2Bstack&utm_source=adwords&utm_medium=cpc&utm_campaign=sorezki&utm_source=adwords&utm_medium=cpc&utm_campaign=mktg_envy_690915556&utm_term=%2Bopen%20%2Bstack&utm_content=165527634601&gclid=CKKAve-Gw9ICFYa2wAodl_4DvQ
8. https://www.youtube.com/watch?v=J4BpqwMuIc8
9. docs.openstack.org/mitaka/install-guide-ubuntu/heat-install.html
10. https://wiki.openstack.org/wiki/Nova
11. https://wiki.openstack.org/wiki/Glance
12. https://docs.openstack.org/developer/glance/
13. Trainer.edu.mirantis.com/os100L2.0.0/ceilometer.html
14. youtube.com/watch?v=eOlIB323c8s
15. https://wiki.openstack.org/wiki/Heat
16. https://docs.openstack.org/developer/heat/template_guide/hot_guide.html
17. https://www.wired.com/insights/2012/04/openstack-histor
18. https://www.openstack.org/videos/vancouver-2015/public-or-private-cloud-amazon-web-services-or-openstack-what-and-039s-the-difference-and-can-i-use-both
19. http://redhatstackblog.redhat.com/2015/05/13/public-vs-private-amazon-compared-to-openstack
20. http://www.networkworld.com/article/3055195/cloud-computing/google-cozies-up-with-openstack.html
21. https://keithtenzer.com/2015/09/02/auto-scaling-instances-with-openstack/
22. http://docs.openstack.org/developer/heat/template_guide/hot_spec.html
23. https://es.scribd.com/document/312104113/Heat-Introduction