**COMP4442**

**Service and Cloud Computing**

**Full OpenStack Implementation**

Group 4

13103214D CHEN Yunkun

13103831D HAN Yunpeng

13104036D ZHENG Hongyi

**I. Motivation and Objectives**

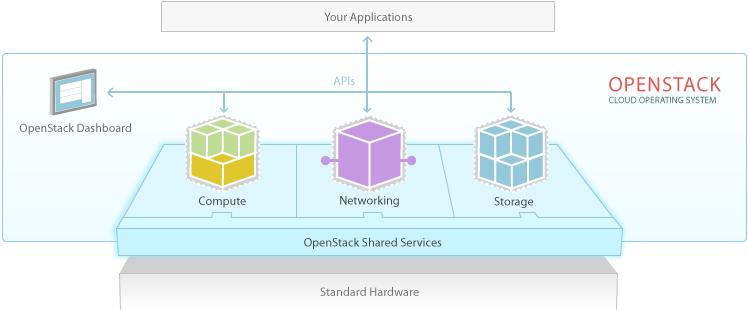
Motivation:

In the lecture, we learned the service-oriented architecture of cloud computing, including the evolution of computing, from functional, to object, to component and to service-oriented; benefits of Service abstraction; Software as a service (SaaS); Software-Oriented distribution; Web services fundamentals; Service composition.

OpenStack is a free and open-source software platform for cloud computing, mostly deployed as an infrastructure-as-a-service (IaaS). The software platform consists of interrelated components that control diverse, multi-vendor hardware pools of processing, storage, and networking resources throughout a data center.

In the lab, we implemented some basic setup and installation of OpenStack including basic components, like keystone, nova, neutron, etc. However, many work has been done by professor and we do not go into detail to understand the implementation.

We decide to do a full implementation of OpenStack including all service.



Objectives:

(1) Implement all core services of OpenStack as foundation for optional services.



(2) Implement optional services of OpenStack.

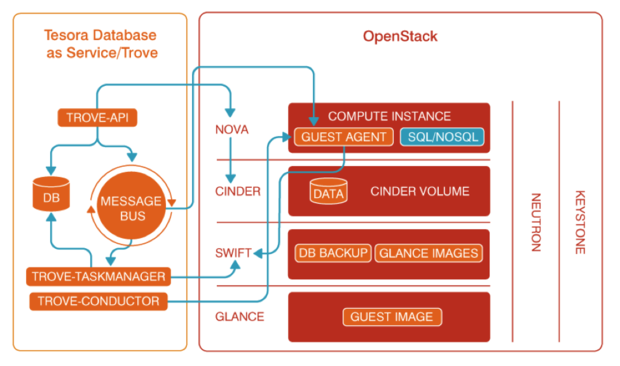


**II. Problem Statement**

The following are the services we plan to implement.  
1. Trove Database service overview

The Database service provides scalable and reliable cloud provisioning functionality for both relational and non-relational database engines. Users can quickly and easily use database features without the burden of handling complex administrative tasks. Cloud users and database administrators can provision and manage multiple database instances as needed.

The Database service provides resource isolation at high performance levels, and automates complex administrative tasks such as deployment, configuration, patching, backups, restores, and monitoring.

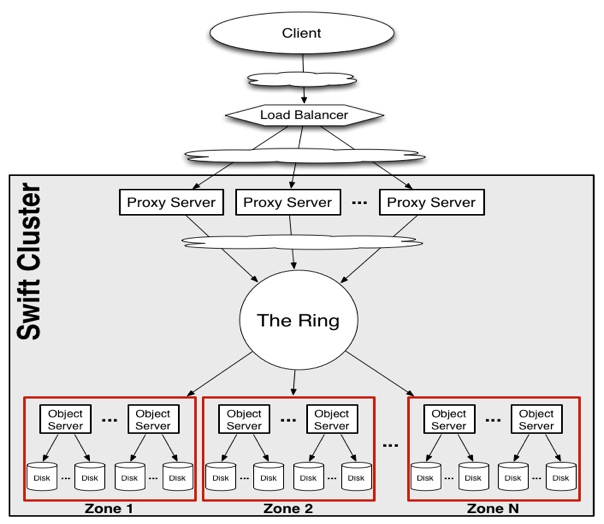


2. Swift Object Store

OpenStack Object Storage (Swift) is a scalable redundant storage system. Objects and files are written to multiple disk drives spread throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster. Storage clusters scale horizontally simply by adding new servers. Should a server or hard drive fail, OpenStack replicates its content from other active nodes to new locations in the cluster. Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used.

The Object Storage services (swift) work together to provide object storage and retrieval through a REST API.

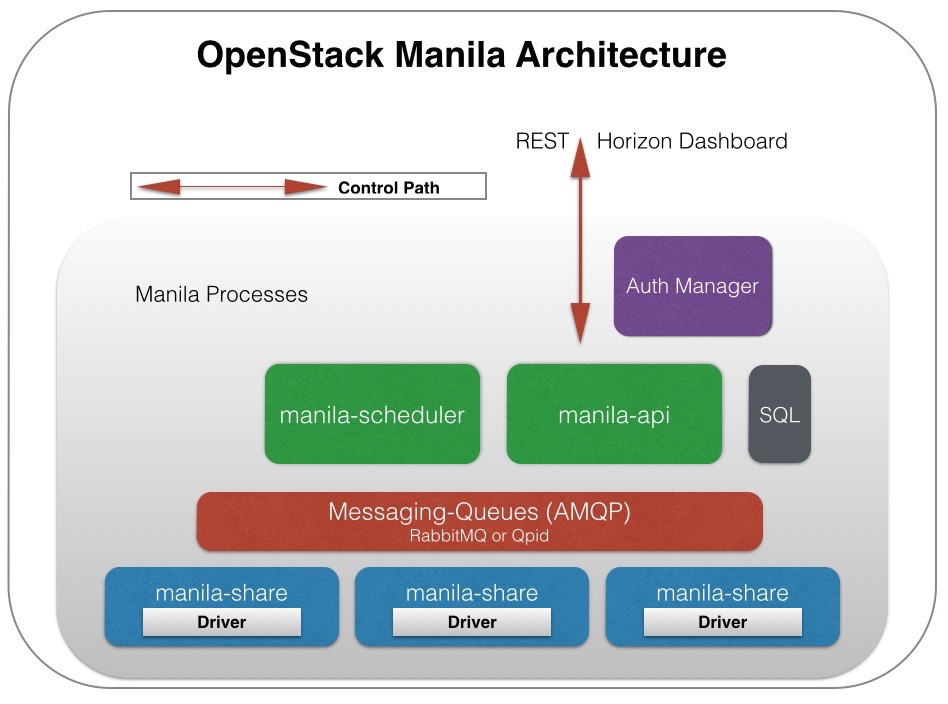
The OpenStack Object Storage is a multi-tenant object storage system. It is highly scalable and can manage large amounts of unstructured data at low cost through a RESTful HTTP API



3. Manila Shared File System

The Shared File Systems service (manila) provides coordinated access to shared or distributed file systems.

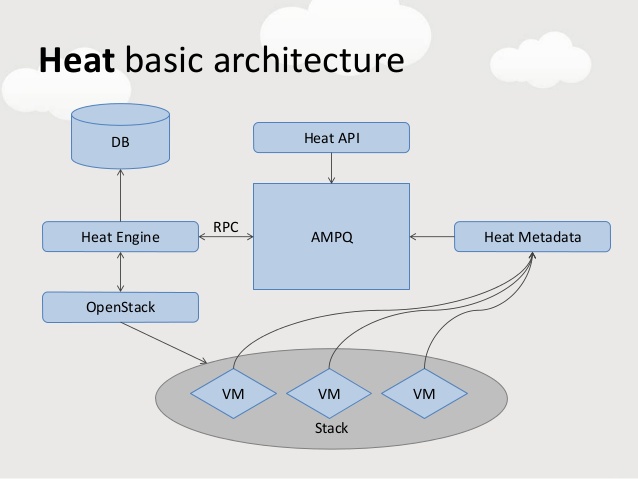
The OpenStack Shared File Systems service (manila) provides file storage to a virtual machine. The Shared File Systems service provides an abstraction for managing and provisioning of file shares. The service also enables management of share types as well as share snapshots if a driver supports them.



4. Heat Orchestration

The Orchestration service provides a template-based orchestration for describing a cloud application by running OpenStack API calls to generate running cloud applications. The software integrates other core components of OpenStack into a one-file template system. The templates allow you to create most OpenStack resource types such as instances, floating IPs, volumes, security groups, and users. It also provides advanced functionality such as instance high availability, instance auto-scaling, and nested stacks. This enables OpenStack core projects to receive a larger user base.

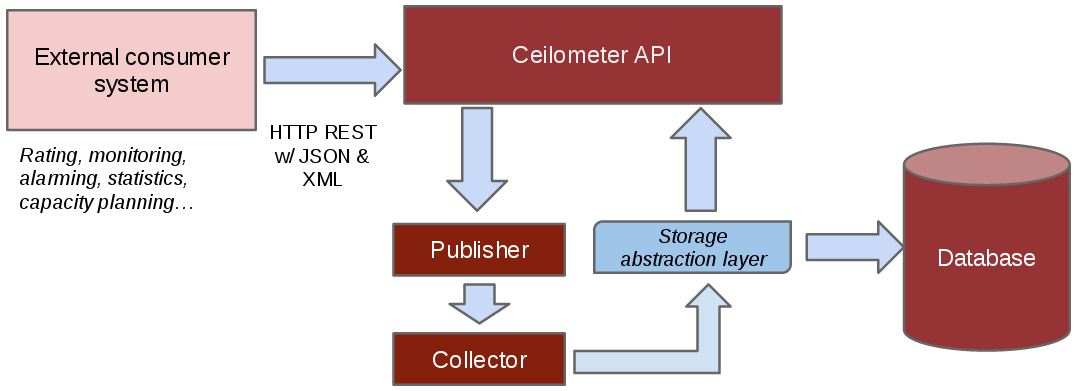
The service allows deployers to integrate with the Orchestration service directly or through custom plug-ins.



5. Ceilometer Telemetry

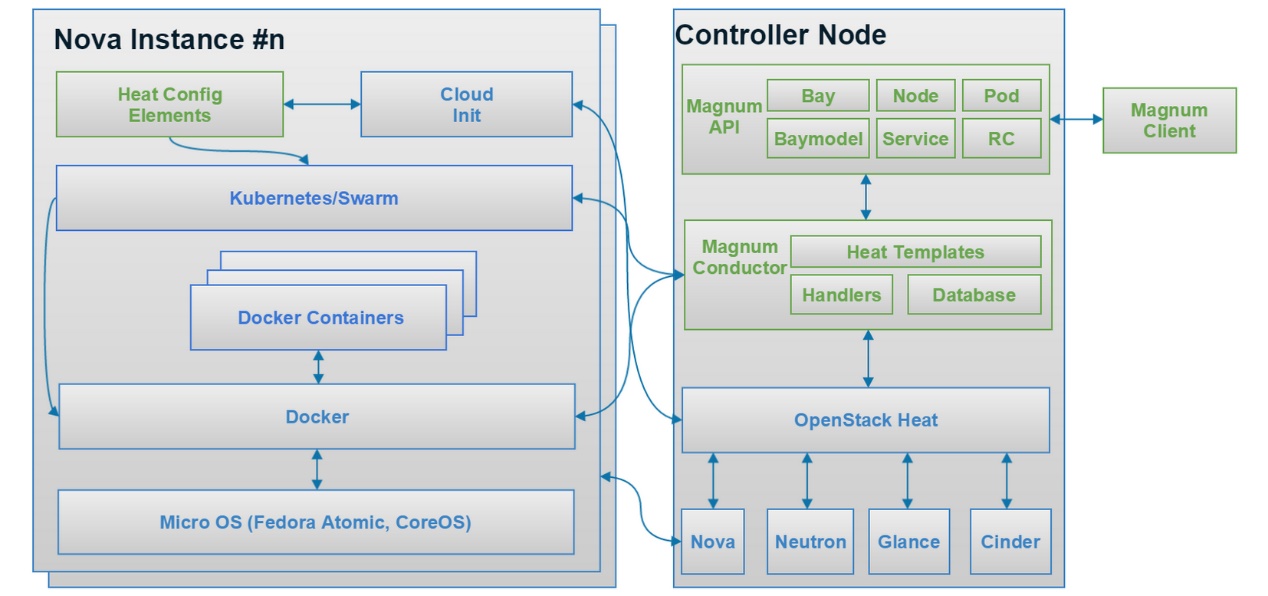
The Telemetry Data Collection services provide the following functions:

* Efficiently polls metering data related to OpenStack services.
* Collects event and metering data by monitoring notifications sent from services.
* Publishes collected data to various targets including data stores and message queues.



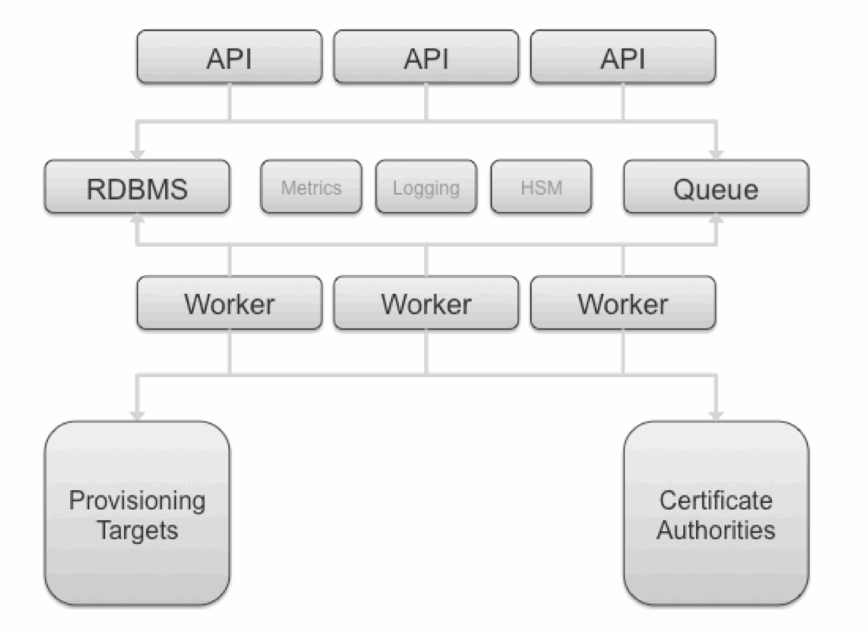
6. Magnum Container Service

The Container Infrastructure Management service codenamed (magnum) is an OpenStack API service developed by the OpenStack Containers Team making container orchestration engines (COE) such as Docker Swarm, Kubernetes and Mesos available as first class resources in OpenStack. Magnum uses Heat to orchestrate an OS image which contains Docker and Kubernetes and runs that image in either virtual machines or bare metal in a cluster configuration.



7. Barbican Key Manager Service

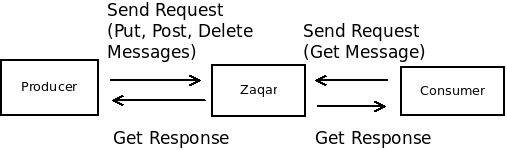
Barbican is a REST API designed for the secure storage, provisioning and management of secrets. It is aimed at being useful for all environments, including large ephemeral Clouds.



8. Zaqar Messaging Service

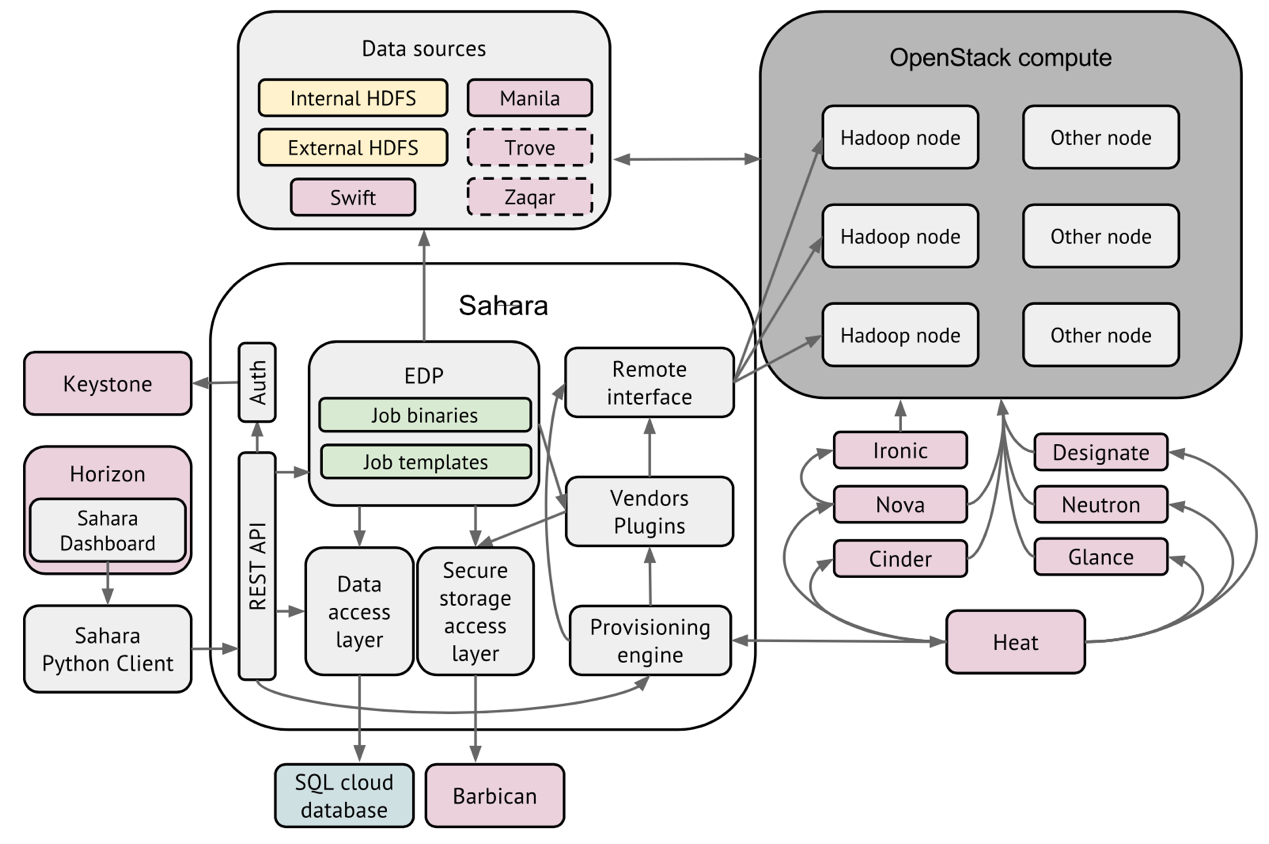
Zaqar is a multi-tenant cloud messaging service for Web developers. The Message service is multi-tenant, fast, reliable, and scalable. It allows developers to share data between distributed application components performing different tasks, without losing messages or requiring each component to be always available.

The service features a RESTful API and a Websocket API, which developers can use to send messages between various components of their SaaS and mobile applications, by using a variety of communication patterns. Underlying this API is an efficient messaging engine designed with scalability and security in mind. Other OpenStack components can integrate with Zaqar to surface events to end users and to communicate with guest agents that run in the "over-cloud" layer.



9. Sahara Elastic Map Reduce

Sahara aims to provide users with simple means to provision Hadoop clusters by specifying several parameters like Hadoop version, cluster topology, nodes hardware details and a few more. After a user fills all the parameters, Sahara deploys the cluster in a few minutes. Sahara also provides means to scale already provisioned cluster by adding and removing worker nodes on demand.

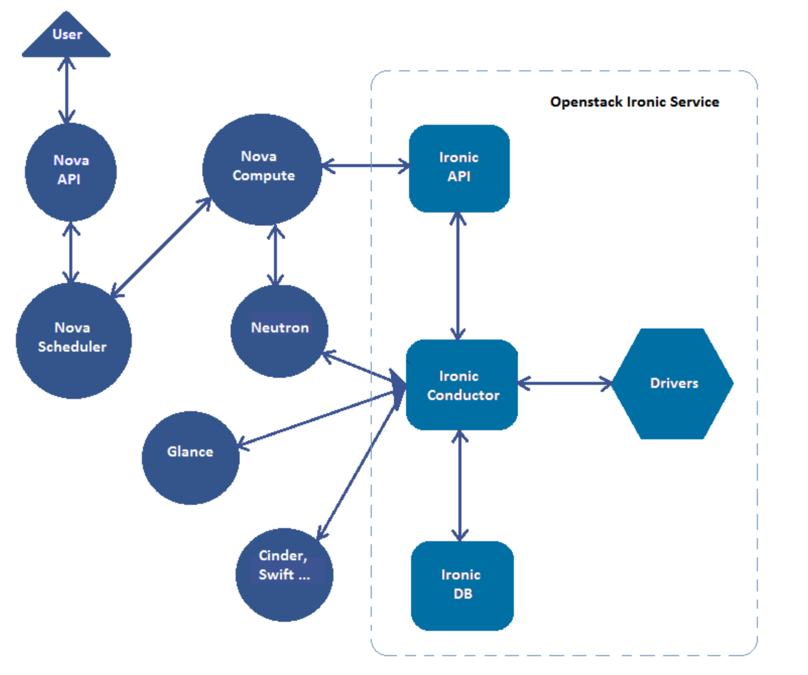


10. Ironic Bare-Metal Provisioning

The Bare Metal service is a collection of components that provides support to manage and provision physical machines.

Also known as the ironic project, the Bare Metal service may, depending upon configuration, interact with several other OpenStack services. This includes:

* the OpenStack Telemetry module (ceilometer) for consuming the IPMI metrics
* the OpenStack Identity service (keystone) for request authentication and to locate other OpenStack services
* the OpenStack Image service (glance) from which to retrieve images and image meta-data
* the OpenStack Networking service (neutron) for DHCP and network configuration
* the OpenStack Compute service (nova) works with the Bare Metal service and acts as a user-facing API for instance management, while the Bare Metal service provides the admin/operator API for hardware management. The OpenStack Compute service also provides scheduling facilities (matching flavors <-> images <-> hardware), tenant quotas, IP assignment, and other services which the Bare Metal service does not, in and of itself, provide.
* the OpenStack Object Storage (swift) provides temporary storage for the configdrive, user images, deployment logs and inspection data.

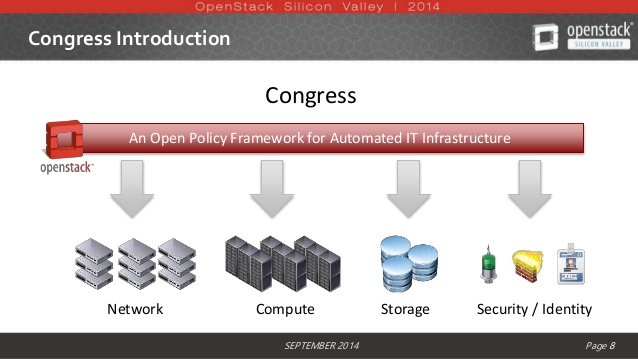


11. Congress Governance

Congress aims to provide an extensible open-source framework for governance and regulatory compliance across any cloud services (e.g. application, network, compute and storage) within a dynamic infrastructure. It is a cloud service whose sole responsibility is policy enforcement.

Congress aims to include the following functionality:

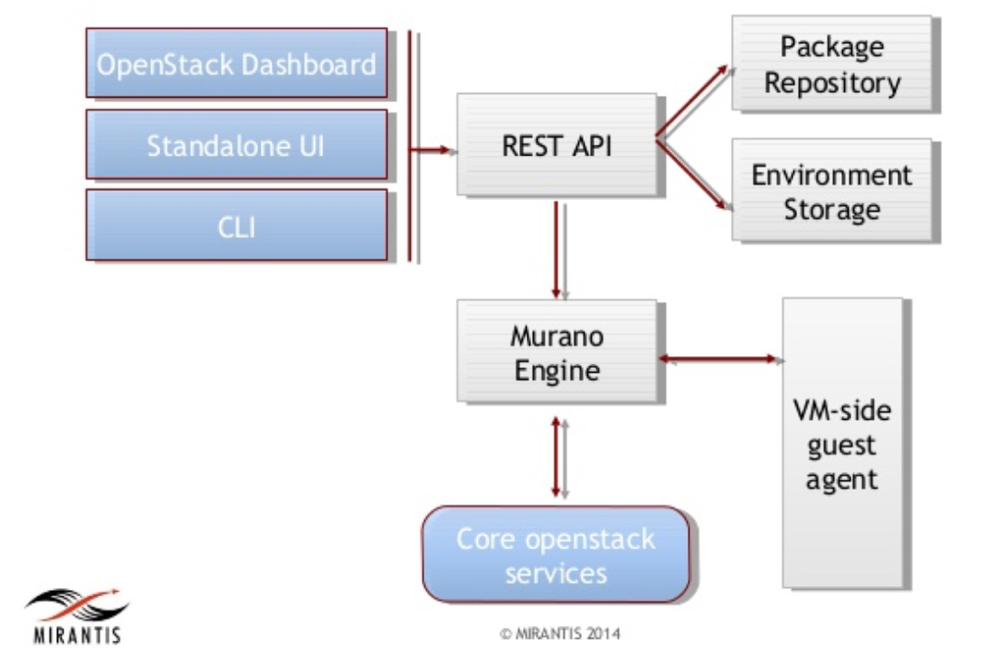
* Allow cloud administrators and tenants to use a high-level, general purpose, declarative language to describe business logic. The policy language does not include a fixed collection of policy types or built-in enforcement mechanisms; rather, a policy simply defines which states of the cloud are in compliance and which are not, where the state of the cloud is the collection of data provided by the cloud services available to Congress. Some examples:
  + Application A is only allowed to communicate with application B.
  + Virtual machine owned by tenant A should always have a public network connection if tenant A is part of the group B.
  + Virtual machine A should never be provisioned in a different geographic region than storage B.
* Offer a pluggable architecture that connects to any collection of cloud services
* Enforce policy
  + Proactively: preventing violations before they occur
  + Reactively: correcting violations after they occur
  + Interactively: give administrators insight into policy and its violations, e.g. identifying violations, explaining their causes, computing potential remediation, simulating a sequence of changes.



12. Murano Application catalog

The Murano Project introduces an application catalog to OpenStack, enabling application developers and cloud administrators to publish various cloud-ready applications in a browsable categorized catalog. Cloud users -- including inexperienced ones -- can then use the catalog to compose reliable application environments with the push of a button.

The key goal is to provide UI and API which allows to compose and deploy composite environments on the Application abstraction level and then manage their lifecycle. The Service should be able to orchestrate complex circular dependent cases in order to setup complete environments with many dependent applications and services. However, the actual deployment itself will be done by the existing software orchestration tools (such as Heat), while the Murano project will become an integration point for various applications and services.



**III. Methodology and Implementation**

We follow the tutorial and guide from official website or other support web to implement the services step by step.

1. Trove Database service

Follow https://docs.openstack.org/project-install-guide/database/ocata/install-ubuntu.html

Main steps are:

(1) Create the database *trove*.

(2) Create the Database service API endpoints.

(3) Install the packages *python-trove, python-troveclient, python-glanceclient, trove-common, trove-api trove-taskmanager, trove-conductor.*

(4) In the */etc/trove* directory, edit the *trove.conf*, *trove-taskmanager.conf* and *trove-conductor.conf*, *trove-guestagent.conf.*

(5) In the */etc/init* directory, edit the *trove-taskmanager.conf* and *trove-conductor.conf.*

2. Swift Object Storage

Follow https://docs.openstack.org/project-install-guide/object-storage/ocata/controller-install-ubuntu.html

Main steps are:

(1) Configure network.

(2) Create the Identity service credentials.

(3) Create the Object Storage service API endpoints.

(4) Install the packages *swift, swift-proxy python-swiftclient, python-keystoneclient, python-keystonemiddleware, memcached*.

(5) Create the */etc/swift* directory and edit the */etc/swift/proxy-server.conf*.

(6) Configure storage node.

3. Manila Shared File System

Follow https://docs.openstack.org/project-install-guide/shared-file-systems/ocata/

Main steps are:

(1) Create the *manila* database.

(2) Create the service credentials.

(3) Create the Shared File Systems service API endpoints.

(4) Install the packages *manila-api, manila-scheduler, python-manilaclient*.

(5) Edit the */etc/manila/manila.conf* file.

(6) Configure share node.

4. Heat Orchestration

Follow https://docs.openstack.org/project-install-guide/orchestration/ocata/install-ubuntu.html

Main steps are:

(1) Create the *heat* database.

(2) Create the service credentials.

(3) Create the Orchestration service API endpoints.

(4) Install the packages *heat-api, heat-api-cfn, heat-engine*.

(5) Edit the */etc/heat/heat.conf*.

Ceilometer Telemetry Data Collection service

Follow https://docs.openstack.org/project-install-guide/telemetry/ocata/install-base-ubuntu.html

Main steps are:

(1) Create the service credentials.

(2) Register Gnocchi service in Keystone.

(3) Edit the */etc/ceilometer/ceilometer.conf*.

(4) Edit the */etc/gnocchi/gnocchi.conf*.

(5) Create Ceilometer resources in Gnocchi.

6. Magnum Container Service

Follow https://docs.openstack.org/project-install-guide/container-infrastructure-management/ocata/install-ubuntu.html

Main steps are:

(1) Create the magnum database.

(2) Create the service credentials.

(3) Create the Container Infrastructure Management service API endpoints.

(4) Install packages *magnum-api, magnum-conductor, python-magnumclient*.

(5) Edit the */etc/magnum/magnum.conf* file.

7. Barbican Key Manager Service

Follow https://docs.openstack.org/project-install-guide/key-manager/ocata/install-ubuntu.html

Main steps are:

(1) Create the barbican database.

(2) Create the service credentials.

(3) Create the Key Manager service API endpoints.

(4) Install the packages *barbican-api, barbican-keystone-listener, barbican-worker*.

(5) Edit the */etc/barbican/barbican.conf* and */etc/barbican/barbican-api-paste.ini*.

8. Zaqar Messaging Service

Follow https://docs.openstack.org/project-install-guide/messaging/ocata/install-ubuntu.html

Main steps are:

(1) Install and configure MongoDB replica-set on database servers.

(2) Create the service credentials.

(3) Create the Messaging service API endpoints.

(4) Install *memcached*, Messaging service and uWSGI.

(5) Create log file and edit configuration file.

(6) Create Messaging service’s configuration file */etc/zaqar.conf*.

9. Sahara Elastic Map Reduce

(1) Install follow: https://docs.openstack.org/developer/sahara/userdoc/installation.guide.html

(2) Configure follow:

https://docs.openstack.org/developer/sahara/userdoc/configuration.guide.html

10. Ironic Bare-Metal Provisioning

Follow https://docs.openstack.org/project-install-guide/baremetal/ocata/install-ubuntu.html

Main steps are:

(1) Create an ironic database.

(2) Install packages *ironic-api*, *ironic-conductor*, *python-ironicclient*.

(3) Edit */etc/ironic/ironic.conf* file.

(4) Configuring ironic-conductor service.

11. Congress Governance

Follow http://congress.readthedocs.io/en/latest/readme.html

Main steps are:

(1) Clone congress and install source code.

(2) Create congress database.

(3) Configure /etc/congress/congress.conf.

12. Murano Application catalog

Follow https://docs.openstack.org/developer/murano/administrator-guide/deploy\_murano/install\_manually.html

Main steps are:

(1) Clone murano and install source code.

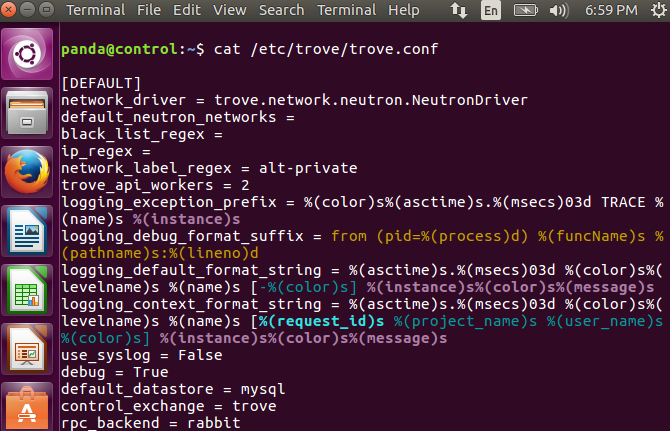
(2) Create murano database.

(3) Configure /etc/murano/murano.conf

**IV. Results and Conclusion**

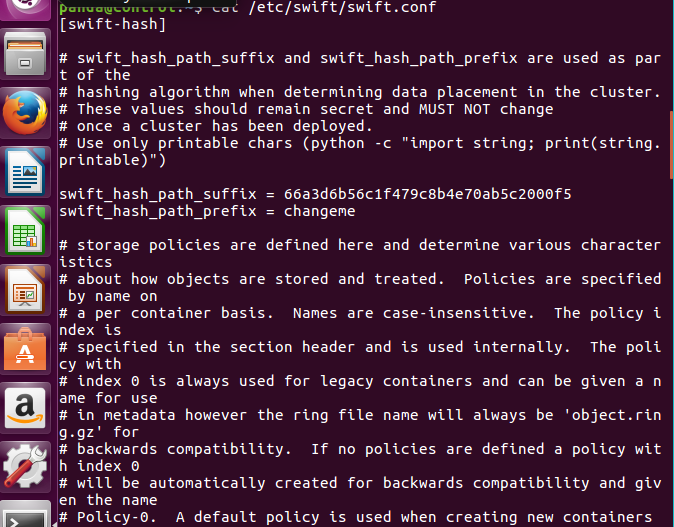
1. Trove Database service





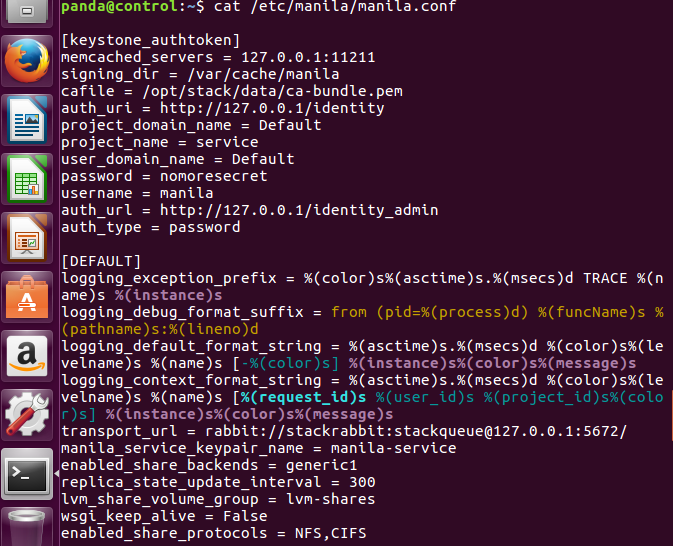
2. Swift Object Storage





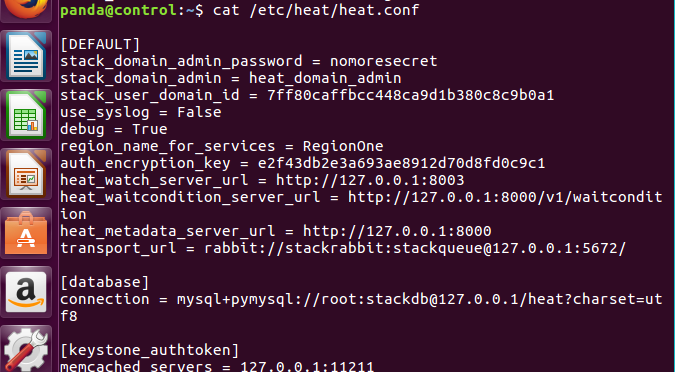
3. Manila Shared File System



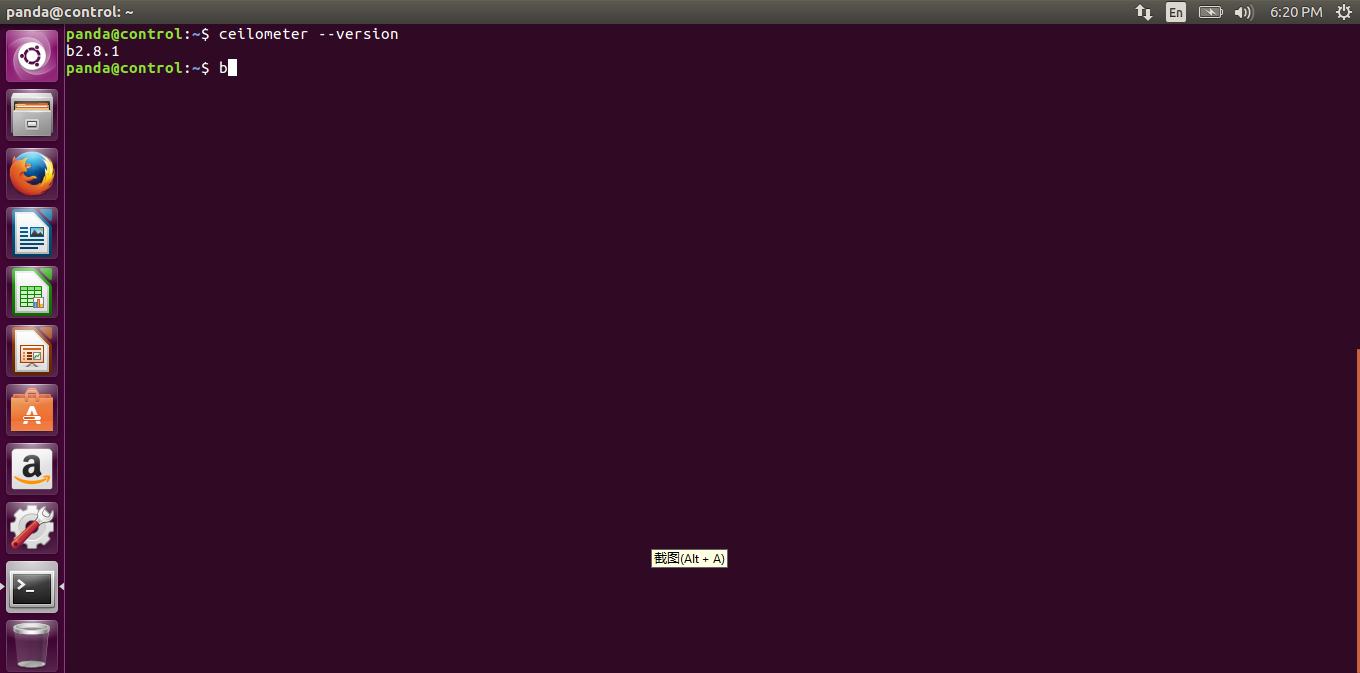


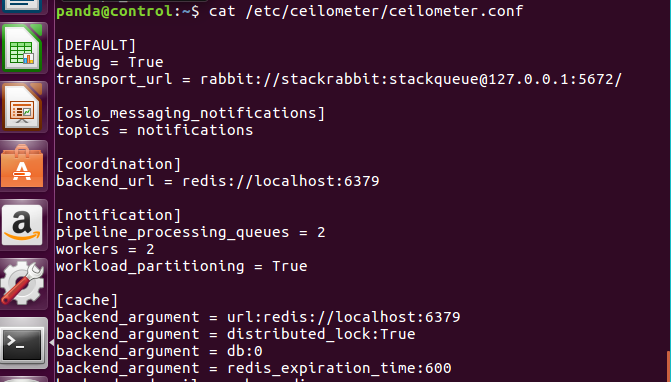
4. Heat Orchestration



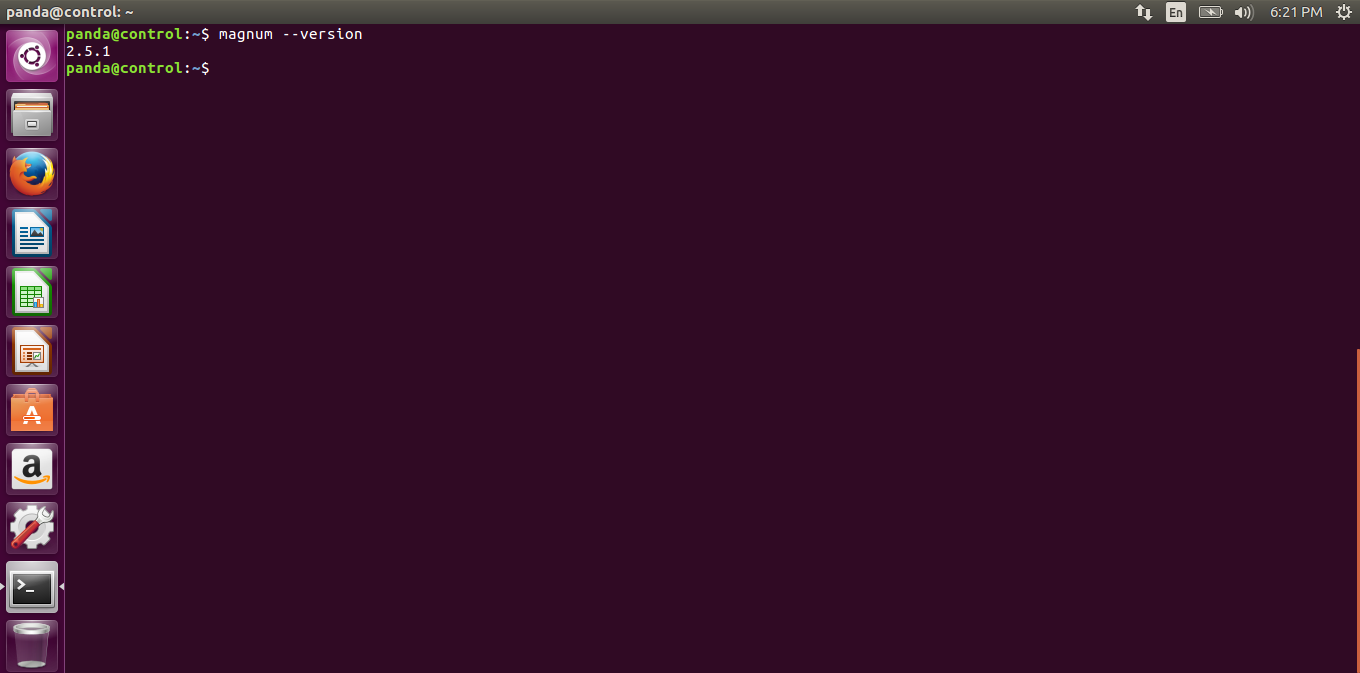


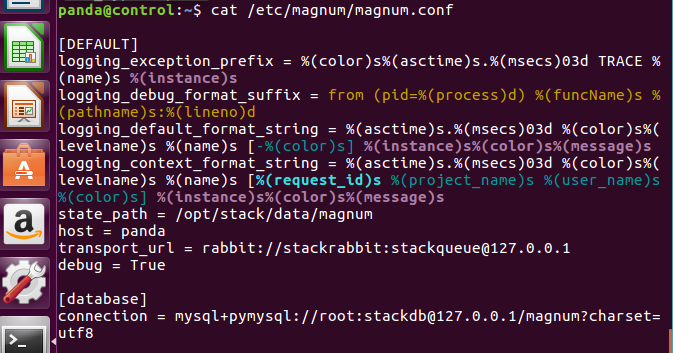
5. Ceilometer Telemetry



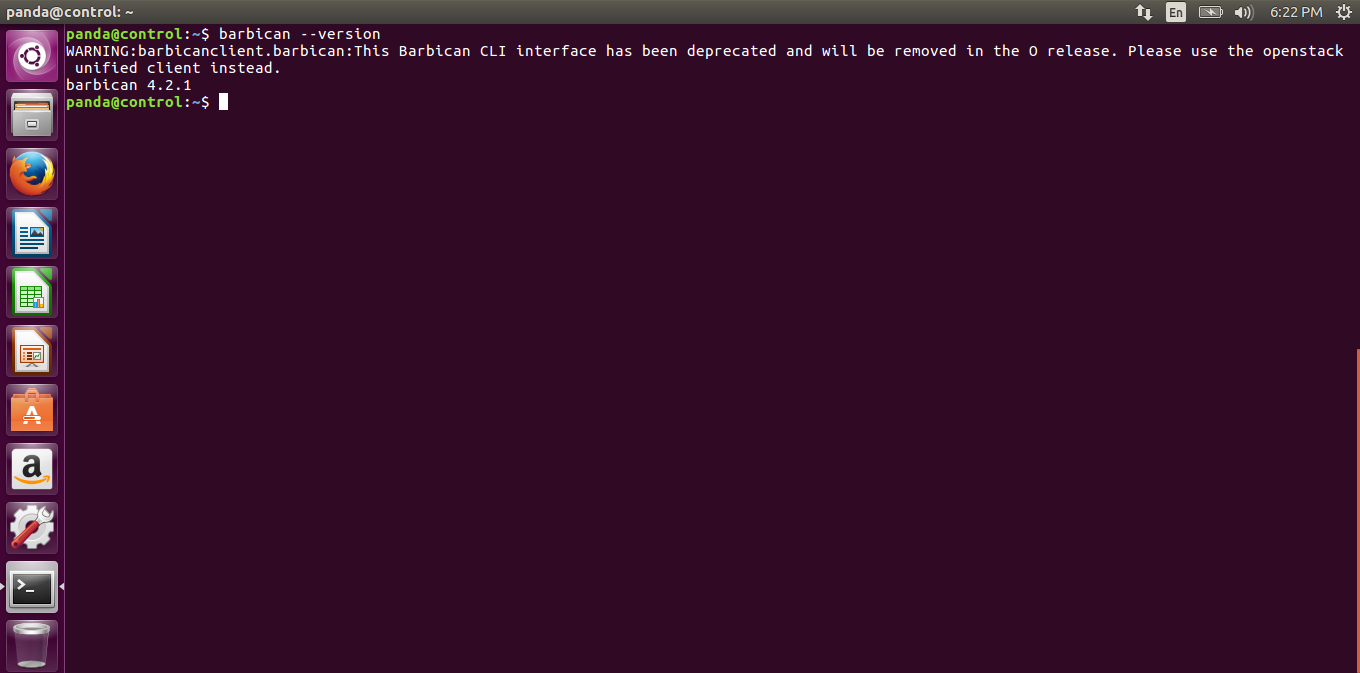


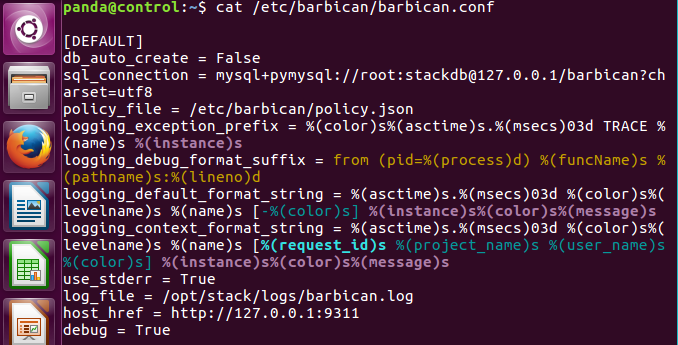
6. Magnum Container Service



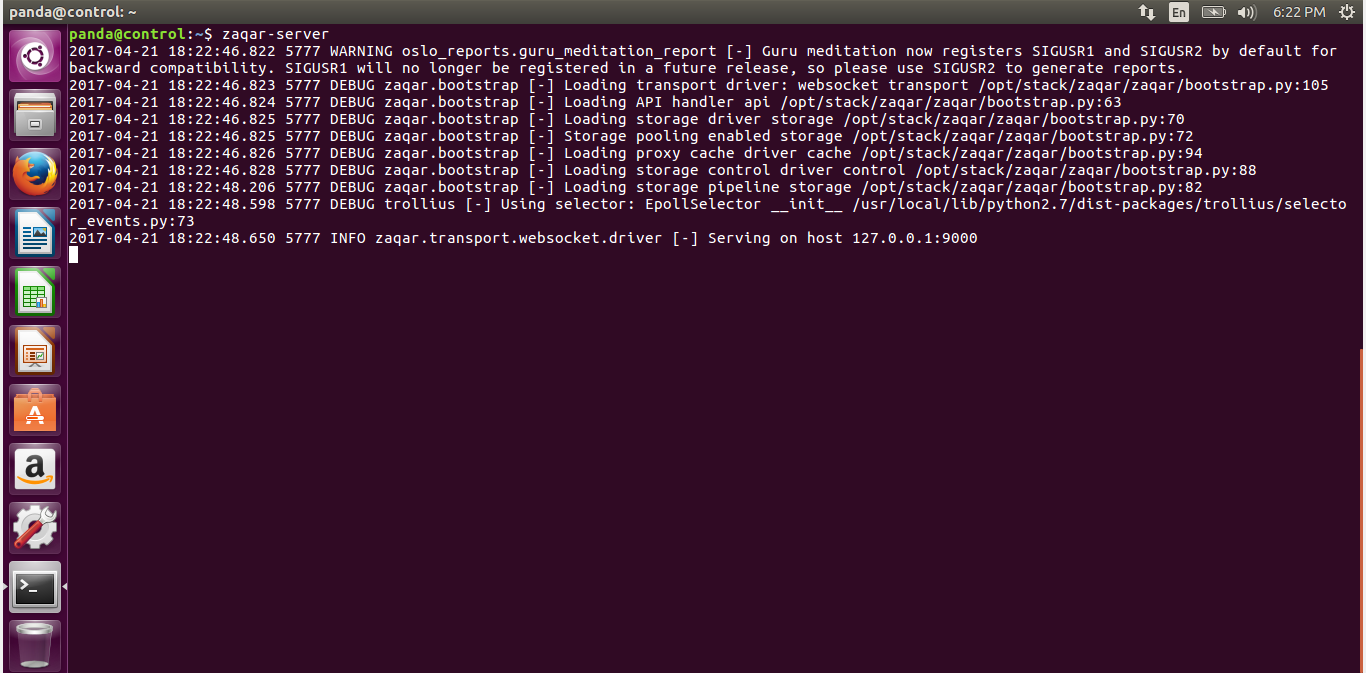


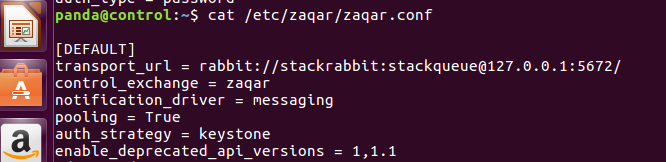
7. Barbican Key Manager Service





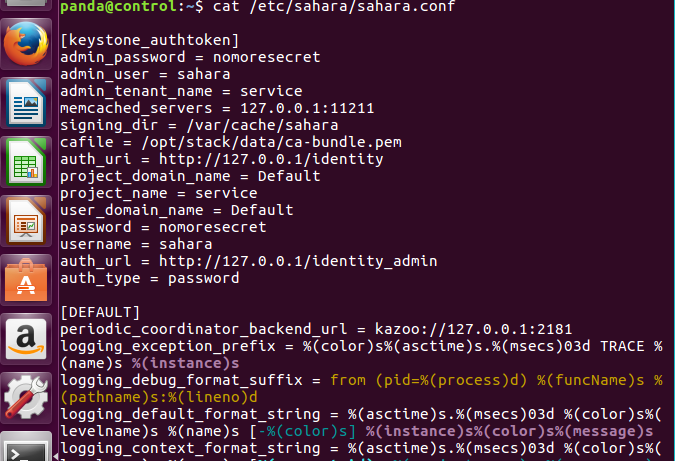
8. Zaqar Messaging Service



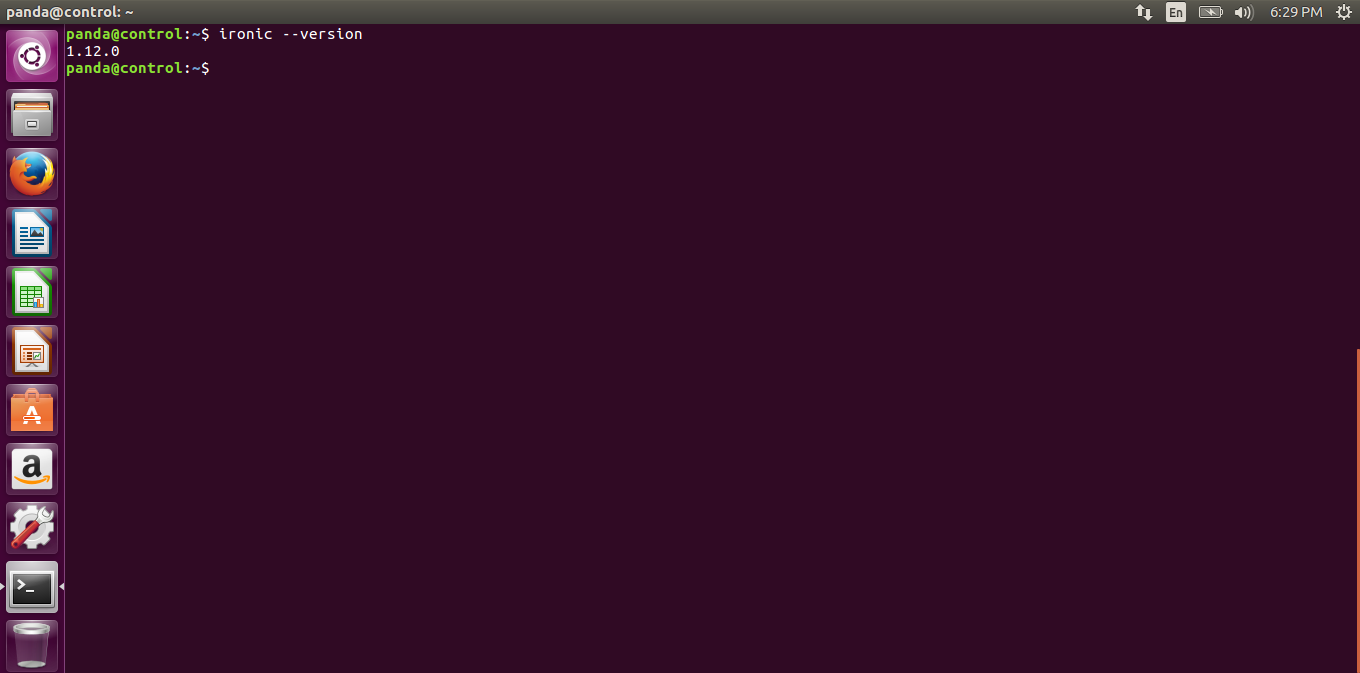


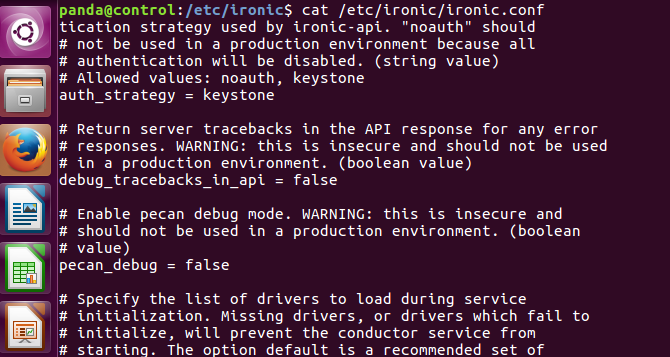
9. Sahara Elastic Map Reduce



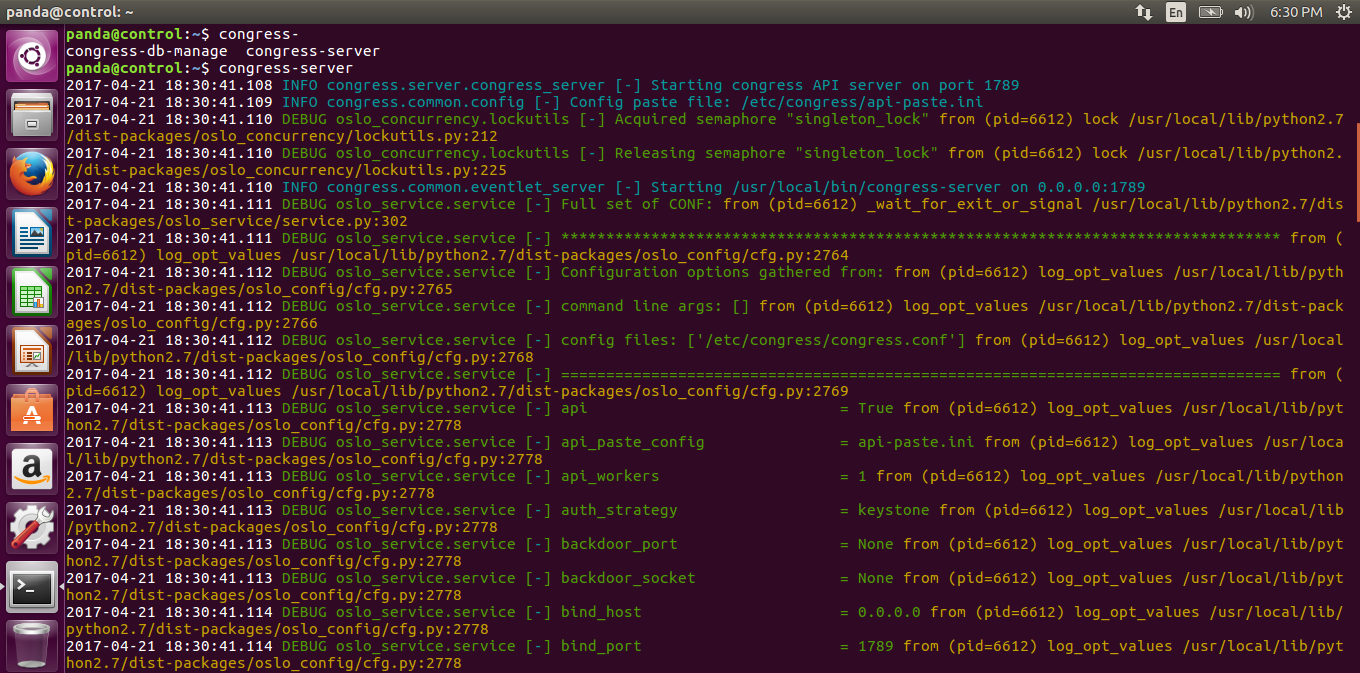


10. Ironic Bare-Metal Provisioning



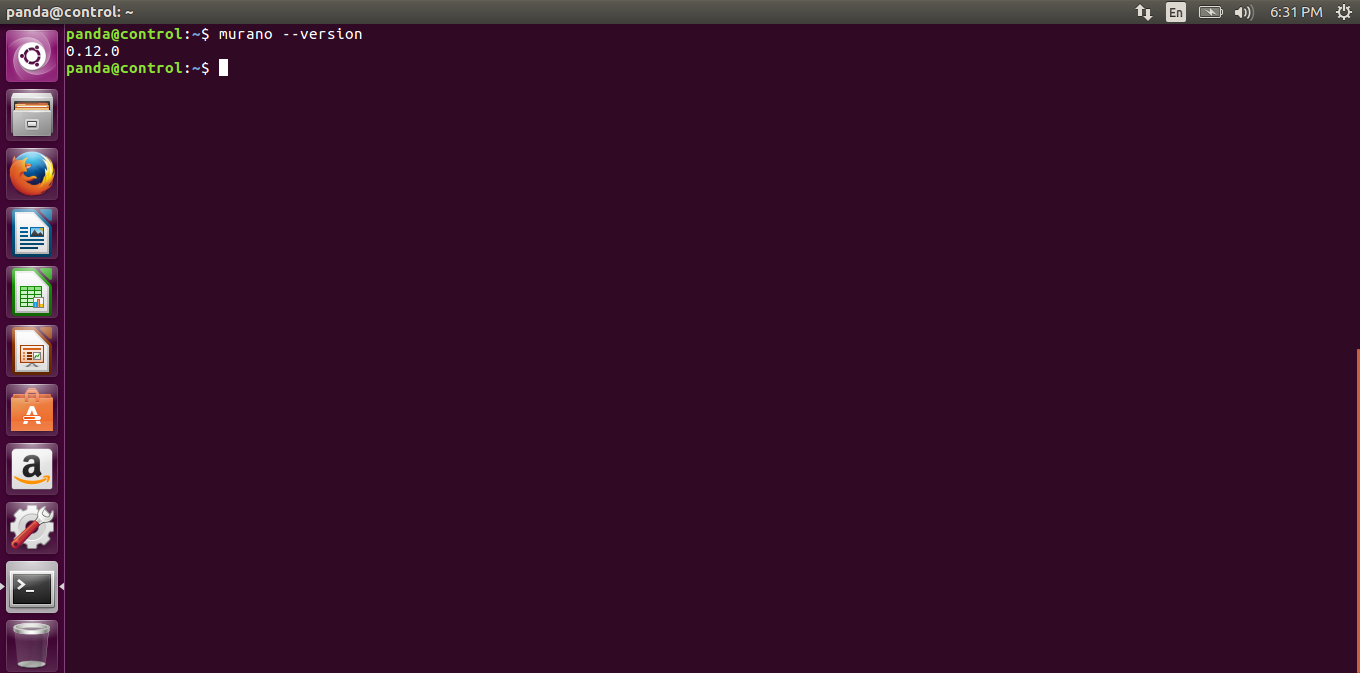


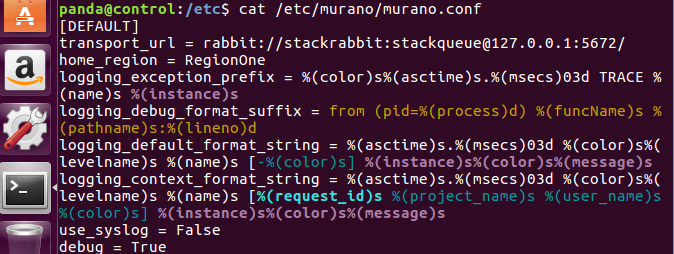
11. Congress Governance



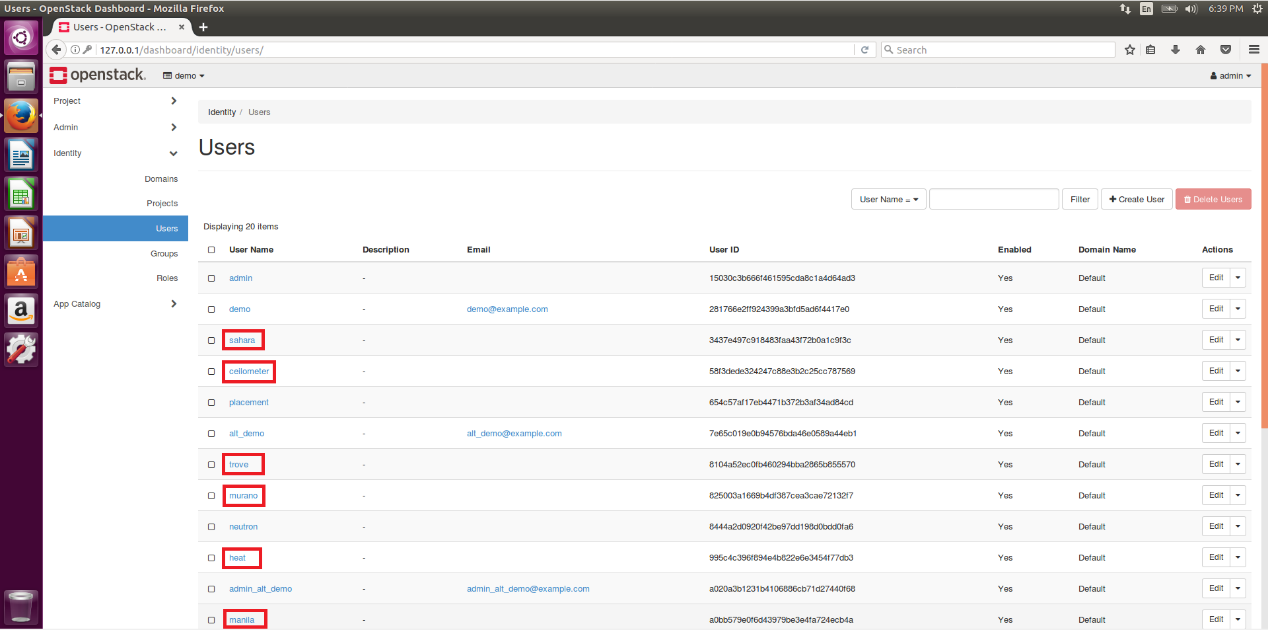
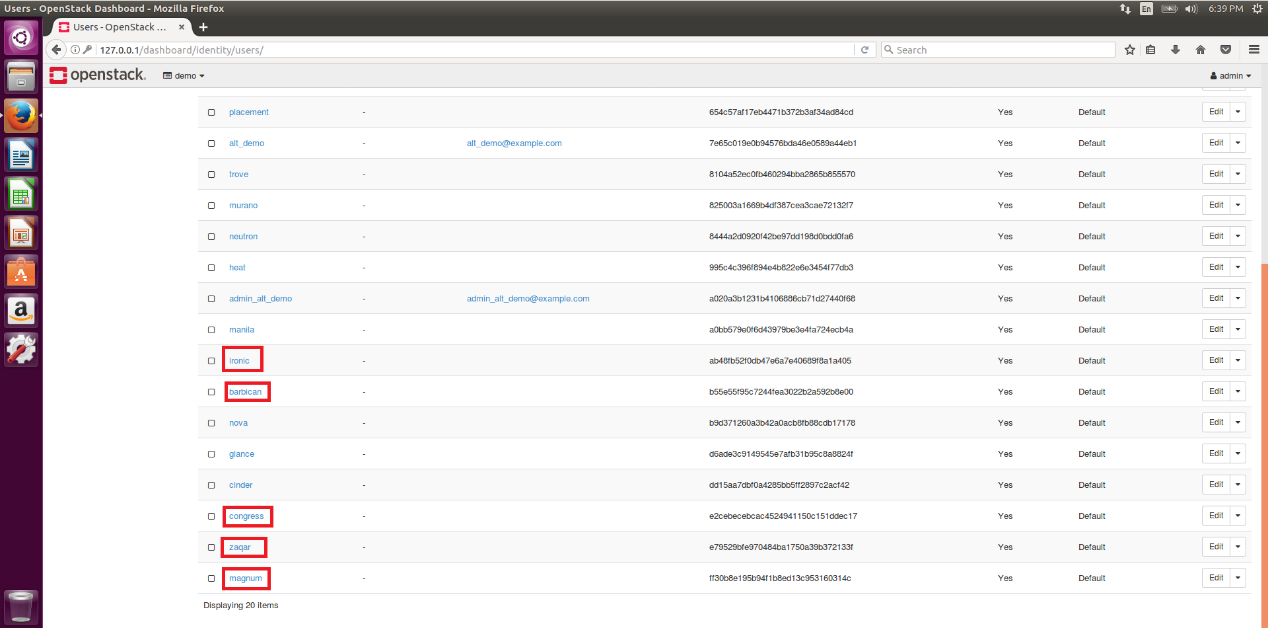


12. Murano Application catalog





13. Services shown in Horizon



Conclusion:

We have implemented above services of OpenStack on the node of Virtual Machine on Ubuntu 16.04.

During the implementation, we strengthen the knowledge and understanding of cloud computing and the OpenStack system.

**V. Background Literature and References**

1. 5 ways OpenStack Trove will change how you manage databases http://www.infoworld.com/article/2881933/database/5-ways-openstack-trove-will-change-how-you-manage-databases.html
2. Architecture for Barbican. https://docs.openstack.org/developer/barbican/contribute/architecture.html
3. Architecture for Sahara. https://docs.openstack.org/developer/sahara/architecture.html
4. Container Infrastructure Management servicehttps://docs.openstack.org/project-install-guide/container-infrastructure-management/newton/
5. Bare Metal service overview. https://docs.openstack.org/project-install-guide/baremetal/ocata/get\_started.html
6. Configure OpenStack nova with remote Bind Server. http://stackoverflow.com/questions/15779082/configure-openstack-nova-with-remote-bind-server
7. Database service overview https://docs.openstack.org/project-install-guide/database/newton/get\_started.html
8. Introduction to Ironic. https://docs.openstack.org/developer/ironic/deploy/user-guide.html
9. Install and configure MongoDB replica-set on database servers:Wikipedia. OpenStack. https://en.wikipedia.org/wiki/OpenStack
10. Openstack and Docker – Part 2 https://sreeninet.wordpress.com/2015/06/14/openstack-and-docker-part-2/
11. OpenStack Series: Part 11 Ceilometer – Monitoring and Metering Service http://cloudn1nblogspot.hk/2014/11/openstack-series-part-11-ceilometer.html
12. OUTREACHY WITH OPENSTACK ZAQAR. http://exploreshaifali.github.io/talks/6-8-2015-outreachy-with-openstack-zaqar.html#/cover-page
13. Policy in OpenStack - Martin Casado, CTO, VMware - OpenStackSV 2014

https://www.slideshare.net/mirantis/openstack-policy-martin-casado-vmware-keynote-openstacksv

1. Shared File Systems Installation Tutorial. https://docs.openstack.org/project-install-guide/shared-file-systems/newton/