Application of OpenStack

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Vedant Nakrani (1914064) dept. of Information Technology  KJ Somaiya College of Engineering  Mumbai, India vedant.nakrani@somaiya.edu  Aditya Kore (1914067) dept. of Information Technology  KJ Somaiya College of Engineering  Mumbai, India aditya.kore@somaiya.edu | Viraj Patidar (1914065) dept. of Information Technology  KJ Somaiya College of Engineering  Mumbai, India viraj.patidar@somaiya.edu  Rajas Bondale (1914068) dept. of Information Technology  KJ Somaiya College of Engineering  Mumbai, India rajas.bondale@somaiya.edu | Aayush Kapoor (1914066) dept. of Information Technology  KJ Somaiya College of Engineering  Mumbai, India aayush.kapoor@somaiya.edu  Aayush Kapoor (1914066) dept. of Information Technology  KJ Somaiya College of Engineering  Mumbai, India aayush.kapoor@somaiya.edu |

Abstract— Cloud computing is a broad research area that uses many aspects of software and hardware solutions, including computing and storage resources, application runtimes or complex application functionalities. OpenStack is an opensource cloud computing platform having a modular architecture, based on the IaaS (Infrastructure as a Service) model. OpenStack is an open-source cloud computing infrastructure software project and is one of the three most active open-source projects in the world. The aim of this paper is to provide a detailed view of the OpenStack architecture, along with its applications and importance in today’s modern world.

Keywords—OpenStack, Applications of OpenStack, Cloud

Monitoring, Cloud, IaaS

# I. INTRODUCTION

OpenStack is a free, open standard cloud computing platform. It is mostly deployed as infrastructure-as-a-service (IaaS) in both public and private clouds where virtual servers and other resources are made available to users. The software platform consists of interrelated components that control diverse, multi-vendor hardware pools of processing, storage, and networking resources throughout a data center. Users manage it either through a web-based dashboard, through commandline tools, or through RESTful web services.

OpenStack is essentially a series of commands known as scripts. Those scripts are bundled into packages called projects that relay tasks that create cloud environments. In order to create those environments, OpenStack relies on 2 other types of software:

* Virtualization that creates a layer of virtual resources abstracted from hardware
* A base operating system (OS) that carries out commands given by OpenStack scripts

Think about it like this: OpenStack itself doesn't virtualize resources, but rather uses them to build clouds. OpenStack also doesn’t execute commands, but rather relays them to the base OS.

# II. GUIDELINES ON HOW TO CONTRIBUTE TO THE DEVELOPMENT OF OPENSTACK

1. Introduction

|  |
| --- |
| XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE |

Cloud Computing is a model, in which a computing infrastructure is seen as a cloud, from which an end user can access applications on demand from anywhere in the world. In Infrastructure as a Service (IaaS) clouds, computing resources are provided in the form of virtual machines. An IaaS service provider hides the details of the overall infrastructure like physical computing resources, location, security, backup, data partitioning, scaling, etc. from the subscriber of the service.

Some of the important IaaS providers are: Amazon EC2, Rackspace, Go Grid, OpenStack, Eucalyptus and CloudStack.

In this section, we will provide a step-by-step guide for setting up a local development environment for contributing to the OpenStack platform, and present the overall workflow of a development project for the same purpose.

1. The Openstack Project

OpenStack is a highly scalable open-source cloud computing platform for building public and private clouds of any size. It is largely deployed for operating IaaS private clouds. It was started in 2010 as a joint project of Rackspace Hosting and NASA. From 2016 onward, it is being managed by the OpenStack Foundation.

It has a variety of shared services that stand on these three main components: Compute, Networking and Object Storage.

* 1. Compute (Nova) –

The Compute component, also known as Nova, is the main management part of OpenStack, which enables ondemand access to computing resources by enabling the control of large networks of virtual machines. It supports wide range of virtualization technologies, bare metal and high-performance computing configurations, hypervisor technologies like KVM and XenServer and container technologies such as LXC on Linux and Hyper-V on Windows.

* 1. Networking (Neutron) –

Neutron offers network connectivity as a service across network interface devices shared by other OpenStack services. Using Neutron, users can easily define, create, implement and control their own network objects (like networks, subnets, ports, sockets etc.) and other OpenStack services can interact with them through API, and administrators can deploy complex set of networking technologies that support high levels of security, multi-tenancy and massive scalability.

* 1. Object Storage (Swift) –

Data stored on object storage devices can be accessed through APIs via an application or simple web services interfaces. Every object storage file has three important components: the data itself, metadata and a globally unique identifier. OpenStack Swift is a highly available and scalable object storage system. It keeps objects on multiple disk drives across object servers of a

data center. It guarantees data protection by keeping multiple instances (usually three) of the object across nodes in an OpenStack cluster.

1. Steps For Contributing Code To Openstack

1. Step 1: Setting up membership/user accounts

* 1. Setting up Launchpad account

Creating a login on Launchpad is the first important step. All OpenStack projects are hosted on Launchpad and Launchpad credentials are used for logging in on several OpenStack-related sites such as Wiki, Gerrit, Jenkins.

* 1. Setting up SSH Keys for Gerrit OpenStack uses Gerrit Code Review system to assure the code quality. To use Gerrit Code Review system contributor needs to login at the code review page with the previously created Launchpad account and upload his or her personal SSH public key on the ssh-keys upload page to so that he or she will be able to commit changes for review later.

* 1. Join OpenStack Foundation

A contributor needs to register as a Foundation Member. One can join OpenStack Foundation by going to the registration page of OpenStack foundation.

* 1. Signing Appropriate Contributors License Agreement
     + The first step before signing appropriate contributors license agreement is to login to the code review page using Launchpad ID of self.
     + Joining the OpenStack Contributors’ group and membership are required to submit code changes.

* + - Installing git.
    - Installing Eclipse which is an open-source IDE.
    - Installing PyDev plugin for Eclipse.
    - Installing EGit plugin for Eclipse.

* 1. Setting up OpenStack code base and test environment Using DevStack:

DevStack is ideal for OpenStack contributors who want to test against a local environment. It is an assertive script which quickly creates a complete OpenStack development environment on local system. To achieve this, one need to do the following: git cloning DevStack code from GitHub, creating a file named local rc under the just created devstack directory and running ./stack.sh script. Now the local environment is ready and all code can be found under /opt/stack.

* 1. Running unit tests

Unit testing is a software testing method in which individual units of source code of an application are one by one tested to determine their suitability for use. Unit testing is often automated but it can also be done manually. Tests in OpenStack are selfcontained in each project.

* 1. Running all the services by Devstack.

* 1. Running the services in Eclipse

* 1. Step 3: Configuring local development and test environment

OpenStack uses Gerrit as its code review system. So before beginning with the first contribution to OpenStack, one should make sure that he/she has GitHub Account created, git global configuration (same email address must be used across all your accounts) set up, git-review installed on the working system, and it must be configured to know about Gerrit. To configure the project to know about Gerrit, one just needs to open a terminal and go to the project directory, and run the command git reviews. It will ask to enter Gerrit username. By inputting the Launchpad id, the local development and test environment will be ready.

|  |  |  |
| --- | --- | --- |
| 2. Step 2: Setting up local development and test   |  |  | | --- | --- | | environment.    If one wants to get involved in OpenStack development process, devstack is a great way to run test environment. It has all the services up and running for testing out any changes made to the code using devstack.    i. Setting up the development environment   Installing Ubuntu 11.10 or higher with python. | If any issue is found in OpenStack, it should be registered as a bug. If one wants to add a new feature, it should be registered as a blueprint. The modifications that are going to be added should be in a branch other than the master. Multiple bug fixings or blueprint developments must not be mixed in one branch. | |

* 1. Starting a new Project

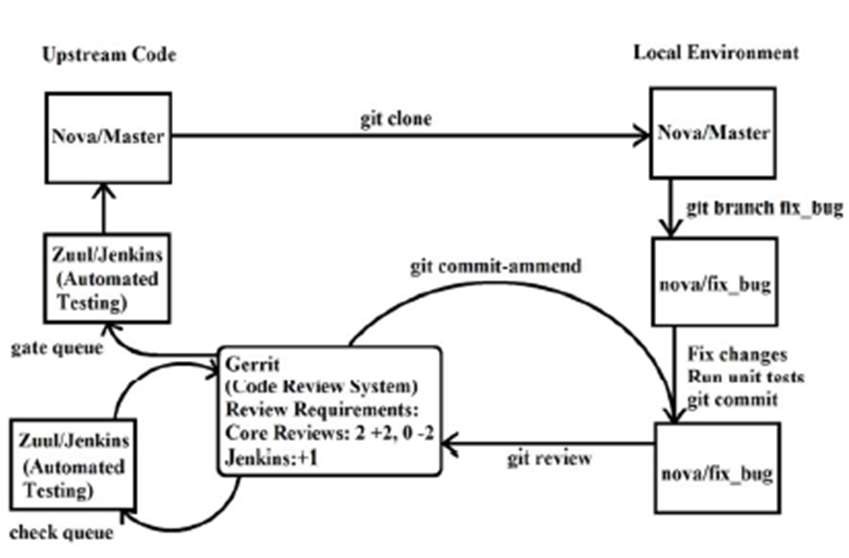


Fig 1. Overall Workflow

* 1. Selecting a module to contribute.
  2. Creating a local git branch based on the master. 3) Working to deal with the work got assigned to self to execute.

4) Creating a single commit based on local branch. 5) Making a patch using git review which is pushed to Gerrit (review.Openstack.org).

6) Waiting for the reviews to come. 7) Addressing the reviews as they appear (working as per the reviews).

8) Inspiring two core members give it a +2 vote and clicking ’Approved’.

5. Bugs and Bug Life Cycle

* + 1. Bugs.

Bugs are usually reported and tracked on Launchpad. Contributors may review these bug reports on a regular basis to get some works to complete.

Basically, there are four main tasks associated with bugs that anyone can do:

* + - * + Confirming new bugs
        + Solving inconsistencies
        + Reviewing incomplete bugs
        + Reviewing stale In Progress bugs

* + 1. Bug Status.

There are different fields available in the bugs reported on Launchpad. “New” is for the bugs that just have been reported. “Incomplete” is for when a bug awaits some more inputs from the bug reporter. “Confirmed” is when the is again generated and confirmed as a genuine one.

* + 1. Bug Attributes
       - * Importance
         * Assigned To
         * Milestone
         * Tags

6. Bug Lifecycle

Bugs travel multiple stages before final resolution.

* + - * + Reporting
        + Confirming & prioritizing
        + Debugging
        + Bug fixing
        + Review

1. Conclusion

This section introduced the overall architecture of OpenStack and provided all necessary information in one place for a beginner to quickly get involved in OpenStack development.

# III. BUILDING A PRIVATE CLOUD PLATFORM BASED ON OPEN-SOURCE SOFTWARE OPENSTACK

1. Introduction

OpenStack is an open source Infrastructure-as-a-Service (IaaS) cloud computing platform, which means that the service provided to consumers is a utilization of all facilities, including processing, storage, networking, and other essential computing resources, and that users are able to deploy and run arbitrary software, including operating systems and applications.

SMBs prefer hybrid cloud deployments, with more than half of their infrastructure driven by private clouds, and OpenStack deployments to understand how cloud platforms work and how OpenStack can improve operational efficiency and accelerate innovation. The cloud computing industry has covered government, finance, transportation, enterprise, education, medical and other application fields, and is integrated with communication,

1. Cloud Computing and OpenStack
   1. Virtualization

Virtualization refers to the virtualization of a computer into multiple logical computers through virtualization technology. Multiple logical computers run simultaneously on a single computer, each logical computer can run a different operating system, and applications can all run in separate spaces without affecting each other, thus significantly increasing the efficiency of the computer. The management of virtual machines includes the management of the entire lifecycle of a virtual machine, as well as the management and control of the basic state of the virtual machine between clusters of virtual machines.

* 1. Cloud Computing

Based on the type of service, the cloud computing service system can be divided into three parts: infrastructure as a service (IaaS), platform as a service(PaaS), and software as a

service (SaaS).

* 1. OpenStack

OpenStack is a cloud computing platform launched by infrastructure service provider Rackspace and NASA, and has taken it completely open source. OpenStack is open because it is open, flexible because of its components, and expansive because it is inclusive. There are compute, network, object storage, block storage, identity, mirroring services, portal, measurement, deployment scheduling, database services, and other components, some of which can be installed as needed.

1. Build a cloud platform
   1. Establishment of a basic experimental environment

i. System environment

Control node controller

IP address: 192.168.109.10/24

Gateways: 192.168.109.2

DNS: 222.172.200.68,61.166.150.123

Compute node nova

IP address: 192.168.109.20/24

Gateways: 192.168.109.2

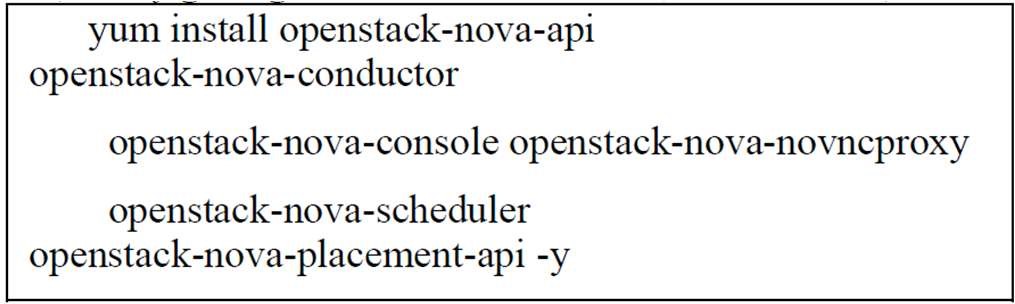
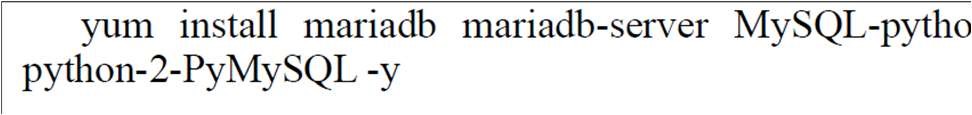
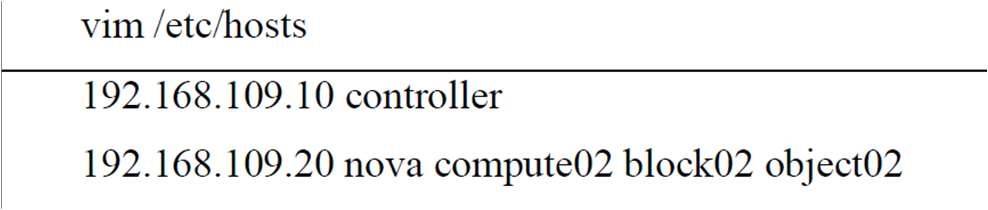
DNS: 222.172.200.68,61.166.150.123 ii. System configuration

iii.

Basic configuration of the control node

iii.

Initialized computing services (compute



Configuring authentication services (control nodes)

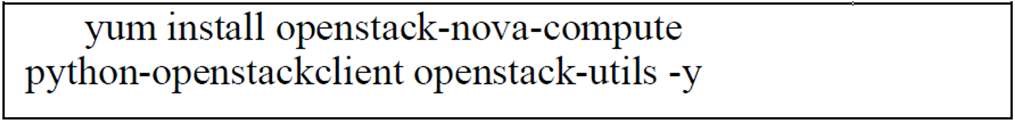
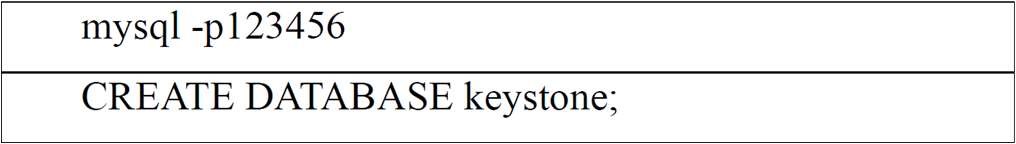
2.

i.

5.

Configuring network services (control nodes)

Host network configuration and testing



Creating keystone related databases

ii.

Configure authentication service-related

software packages

iii.

Initial

authentication

services

and

validation

ii.

Initializing

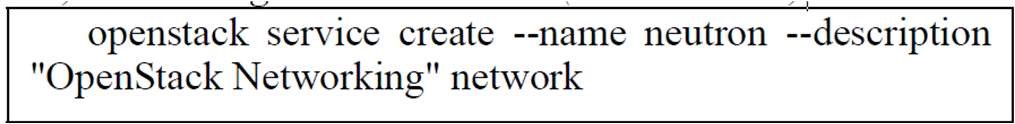
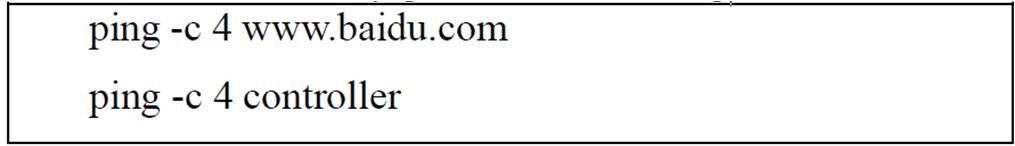
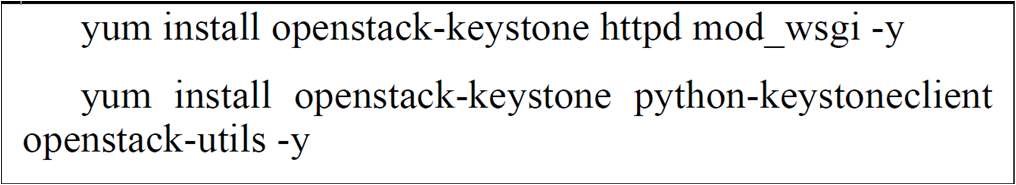
network

services

(

control

nodes)



3. Configuring mirror services (control nodes)

1. Initializing mirror services

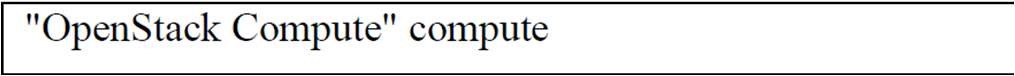
1. Starting the glance mirror service



4. Configuring computing services

1. Initial computing service (control node)





1. Configuring nova-related services (control nodes)

nodes)

i.

iv.

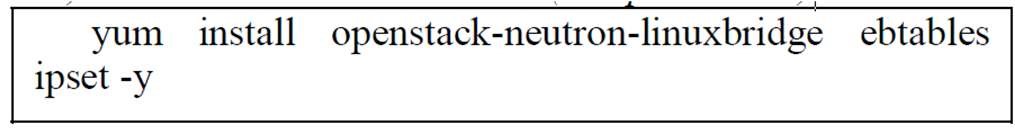
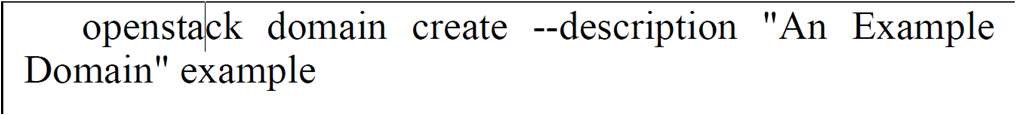
Creating OpenStack Client Environment

Scripts

iii.

Initialized network services (compute

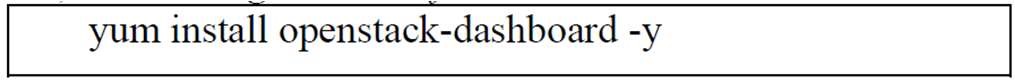
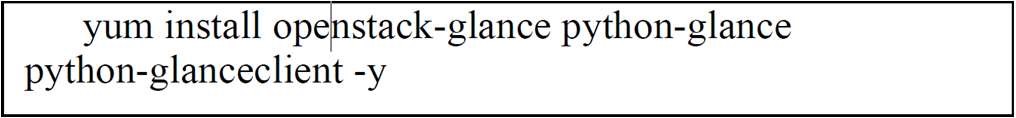
nodes)



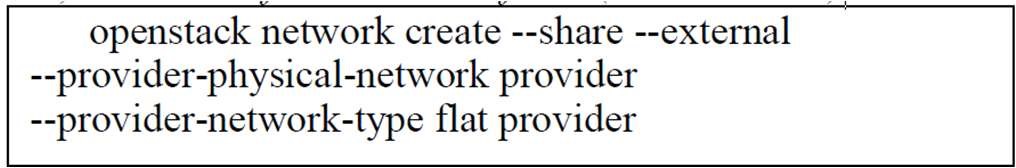
6. Configuring Web Interface Service Components

(Control Nodes)

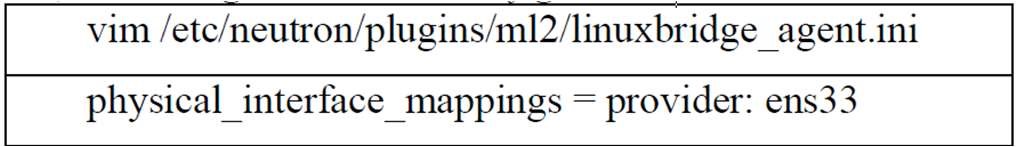
1. Initializing Web Interface Services



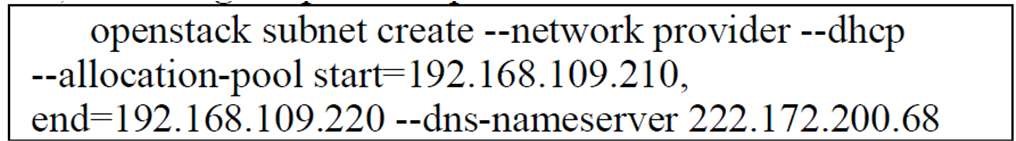
|  |
| --- |
| ii.    Configuring the firewall      iii.    Configuring a VPN      iv.    Configuring the Load Balancer        v.    Validating Web Services  Restart the web server and session storage service.      7.    Creating a public network (control node)  i.    Creation of network interfaces (control  nodes) |

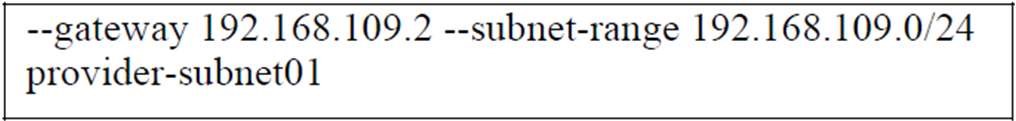


1. Checking the network configuration



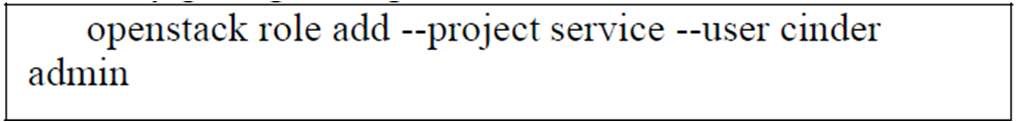
1. Creating the provider public network



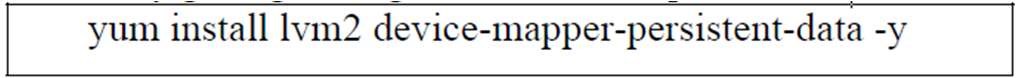


8. Configuring Storage Service Components

1. Configuring Storage Services (Control Node)



1. Configuring Storage Services (Compute Node)



D. SSH Connection Examples

# IV. VIRTUAL MEMORY MANAGEMENT OF PRIVATE CLOUD USING OPENSTACK API

1. About OpenStack

The OpenStack is a popular open-source cloud platform to construct a local cloud infrastructure which is developed by NASA and RackSpace. It is used for the deployment and management of a cloud in provision of infrastructure as a service layer. OpenStack supports both private and public cloud deployments. It can be freely installed on any Linux based system from a series of simple steps.

1. Virtual Memory Management

Step1: Installing OpenStack Cloud

The minimum requirement to install a private OpenStack cloud system is a computer with I3 processor and 4 GB RAM, ideally 8 GB RAM. The OpenStack can be installed on any Linux flavor like RedHat Linux, Mint Linux, Manjaro Linux. For the current study a 64 bit Ubuntu Linux 18.04 is installed on I5 computer with 8 GB RAM and 1 TB hard disk drive. The basic software required for the OpenStack is the latest version of Java and also the git client. The steps in installing OpenStack are as follows:

1. Create a user Stack

useradd –s/bin/bash –d/opt/stack –m stack

1. Provide sudo privileges to user Stack

apt-get install sudo –y || yum install –y sudo echo “stack ALL

=(ALL) NOPASSWD: ALL” >>/etc/sudoers

1. Downloading DevStack on local machine sudo apt-get install git –y || sudo yum install –y git git clone https://git.openstack.org/openstackdev/devstack cd devstack

1. Configuration of OpenStack Create local.conf in devstack directory as shown below to do the following: [[local|localrc]]

FLOATING\_RANGE=192.168.1.224/27

FIXED\_RANGE=10.11.12.0/24

FIXED\_NETWORK\_SIZE=256

FLAT\_INTERFACE=eth0

ADMIN\_PASSWORD=secret

DATABASE\_PASSWORD=secret

RABBIT\_PASSWORD=secret

SERVICE\_PASSWORD=secret

1. Run DevStack installation script

./stack.sh

At this point we should be able to access the dashboard from other computers on the local network. In this example that would be http://192.168.1.201/ for the dashboard (aka Horizon).

After launching virtual machines provided with floating IPs and security group access, the virtual machines become accessible from the computers on your network and also the internet. After installing OpenStack private cloud, two virtual machine images of Ubuntu and windows, were added through the dashboard and the instances were launched.

Step 2: Memory Management

Once the virtual machine images are added to OpenStack, their instances can be launched. Launching of multiple instances can be achieved by setting value greater than 1, the default value is 1. The instances can have different boot sources.

1. Boot from Image
2. Boot from Snapshot
3. Boot from Volume

This is all managed from the Horizon dashboard of the OpenStack. Each instance of the virtual machine is allocated with a processor, a memory size and data transfer. The memory management becomes a serious issue when there are multitude of instances running since it uses the memory of the cloud server. There are situations when some of the virtual machine instances run out of memory and may crash.

Memory ballooning feature is available in OpenStack whereby the allocated memory of some guest instance would be deflated if the instance does not need that much memory. It has also been observed that the Nova does not automatically reclaim memory. The memory can be reclaimed manually by “virsh libvirt” command issued to libvert. But this process of manually reclaiming memory becomes tedious for each and every running instances that claim more memory to run. RAM or memory over committing on the compute node is a great feature allowed by OpenStack.

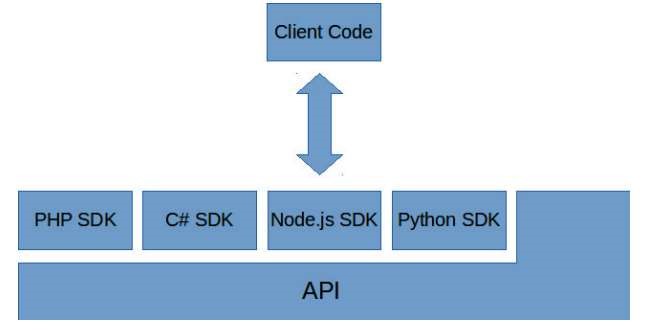
Large number of instances can be run on the cloud but that reduces the performance of the cloud. The default ratio provided by OpenStack for memory allocation is 1.5:1 which means that the scheduler allocates number of instances to a physical node till the sum of memory associated with the instances is less than 1.5 times the memory available on the node. Memory ballooning is a term used to describe a situation where a virtual machine demands more memory than the allocated to it and a balloon of virtual memory is inflated which maps to the physical memory of the node, to be provided to the demanding guest.

Steps to perform memory ballooning to provide additional memory to demanding virtual machine are as follows:

1. Monitor the running instances in memory to and check their virtual memory usage
2. Map the current memory usage with the allocated memory of the instance
3. Inflate the balloon, if the memory usage is close to the allocated virtual memory.
4. Deflate the balloon after the current memory usage is decreased.
5. OpenStack API And SDK

OpenStack provides an Application Program Interface. It is used by a programmer to write software through it can communicate with the cloud services. Applications can be written using OpenStack API to manage the cloud. API provided with the OpenStack are basic aspects for communicating with the cloud.

But for developing a complete application that makes frequent calls, an SDK or Software Development Kit can be used. The use of SDK makes job much easier for an application that runs behind and makes use of functions that combines the URLs and data that make the API calls to get the desired output back from the cloud. The OpenStack SDK are available for many of the popular programming languages like php, ruby, java and python. The implementation for the current project uses python SDK for developing application.



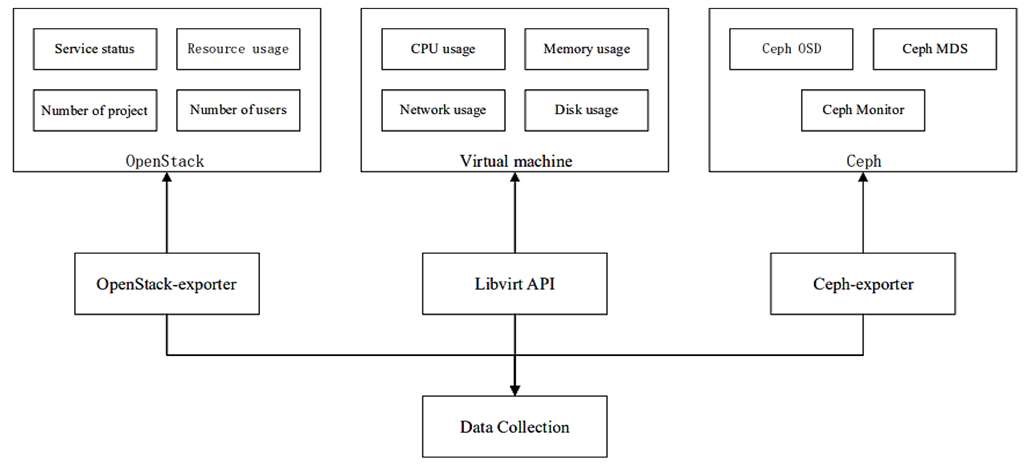
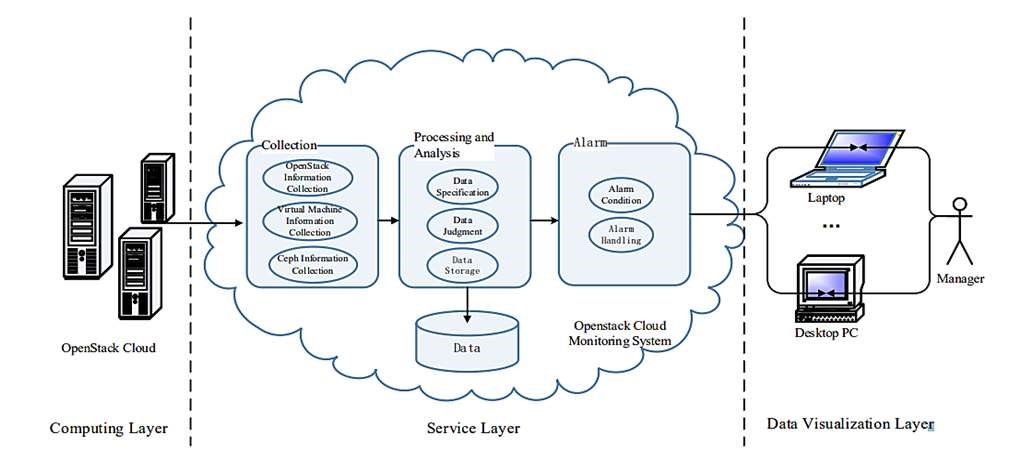
1. Result

Resource management, particularly the management of RAM to the guest virtual machine is one of the crucial tasks in a cloud infrastructure. Memory ballooning is one of the best methods by which a balloon of virtual memory gets inflated with the underlying physical memory allocation to meet the demands of guest. Once the memory of the demanding guest becomes free the balloon gets deflated and the memory is surrendered back to virtual machine.

# V. MICROSERVICES-BASED OPENSTACK MONITORING SYSTEM

Network monitoring is an important tool in cloud computing systems, which can improve the quality of the services provided and allow to scale resource utilization in an adaptive manner. It is widely used to detect key events and anomalies in distributed systems, and also helps to identify faults within the system and discover application patterns for users.

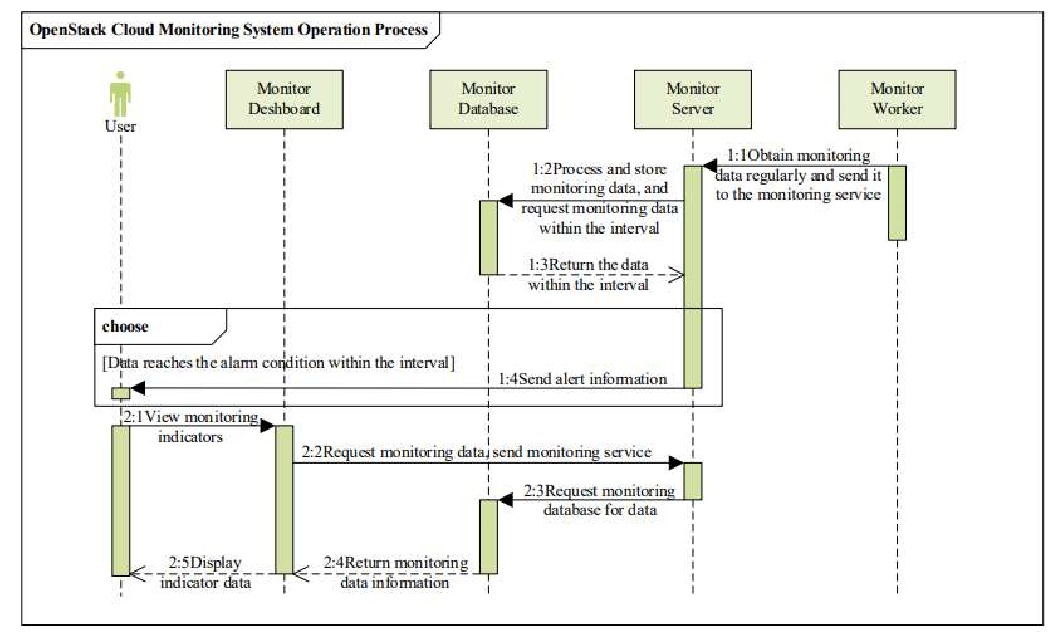
Managers must understand how the OpenStack cloud platform works and stores data. This role can be fulfilled through the monitoring system, which can increase the quality of cloud computing services while also assisting in the identification of system flaws. The paper presents a solution for cloud computing service monitoring that allows users and administrators to optimize computing resources in the cloud computing system based on changing business requirements. The OpenStack cloud monitoring system’s functions involve data collecting, data processing, analysis, presentation, and alert notification.



Openstack Monitoring System

The service layer of the OpenStack cloud platform monitoring system is mainly composed of four parts:

* Monitor Worker is mainly used to collect the performance information and status information of the OpenStack cloud platform.
* Moni-tor Database Server mainly stores the data collected by Monitor Worker.
* Monitor Server mainly collects and processes monitoring information sent by Monitor Worker, and then forwards it to Monitor Data Server for storage.
* Monitor Dashboard mainly displays the monitored indicator data information.



The OpenStack cloud platform monitoring system mainly includes three parts:

* collection of monitoring data information
* processing and analysis of monitoring data information
* storage and display of monitoring data information

1. Monitoring Data Acquisition

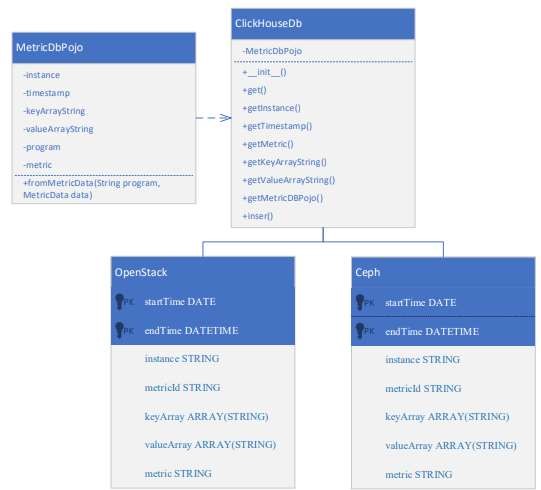
OpenStack monitoring system collects data by proxy mechanism. The monitoring information obtains data from the entire OpenStack cloud platform by calling OpenStackexporter, but OpenStack-exporter cannot obtain the data information of the virtual machine in the computing node and the storage data information of Ceph. For the monitoring of the virtual machine in the computing node, we use Libvirt to call the Hypervisor regularly through an agent module to obtain the data to be monitored by the virtual machine. Since the Hypervisor mainly used by the OpenStack cloud platform is KVM (Kernelbased Virtual Machine), and Libvirt supports KVM. For Ceph monitoring, we use Ceph-exporter to obtain relevant monitoring index data.

1. Monitoring Data Processing and Analysis
   1. Data format processing: The raw data obtained from Open-Stack-exporter, Libvirt, and Ceph-exporter is very large and in different formats for various metrics, which is not conducive to the storage of the system, therefore, it needs to be further processed and converted to a format that conforms to the Prometheus data model. Because most of the data follows the Prometheus style, we only need to modify a small amount of indicator data, reducing time consumption and improving the real-time performance of the monitoring system.

* 1. Data analysis and alerting: Analyse the data collected by the data collection module in the monitoring system of the OpenStack cloud platform to determine whether to send an alert information to the user. The module allows users to pre-set various resource monitoring thresholds. If the monitored value exceeds the predetermined threshold, indicating that there is a problem such as excessive load in the OpenStack cloud platform, the monitoring system will automatically send an alert email or message to the user.

1. Monitoring Data Storage

As the amount of monitored data will become larger and larger, it will be difficult to analyse some useful data. ClickHouse database was chosen for the monitoring data of the OpenStack cloud platform because it is a columnar storage database with the advantages of efficient data compression, multi-core parallel processing, real-time data update and suitable for online query.



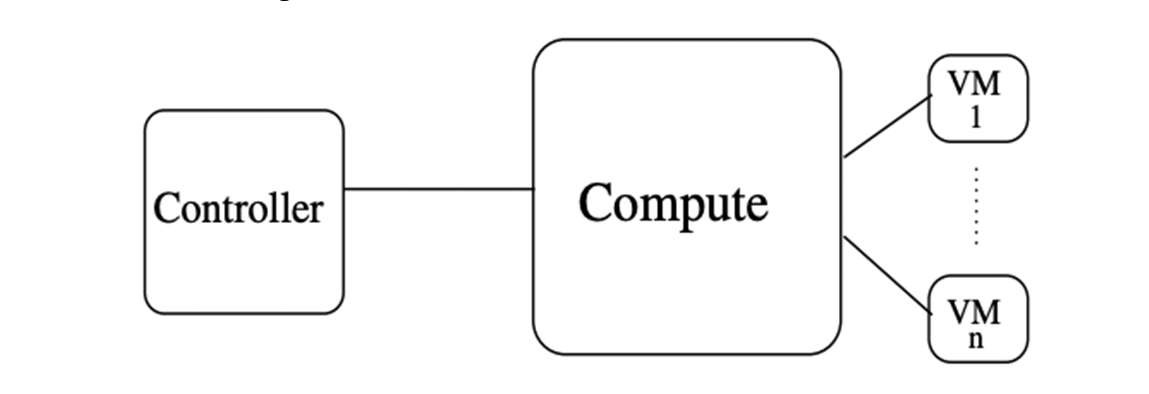
1. Result:

An OpenStack platform was built on a physical all-in-one machine. The platform consists of a control node and two computing nodes. The storage of the two hard disks were mounted on the two computing nodes respectively for Ceph services, which is Nova The virtual machine provides volumes. Six components were installed, including: keystone, glance, nova, neutron, cinder and horizon. The OpenStack collector was deployed in OpenStack, and the result of using the OpenStack collector to monitor our OpenStack cloud is shown below:



* 1. OpenStack-Server
  2. Heat-Server
  3. Tacker-Server
  4. NS-Server

One controller node and one computing node are the minimum requirements for an operational cloud system, as shown in the diagram below.



A. Server Creation Using OpenStack

Algorithm :-

* Fetch the rpm of the respective server
* Configure dynamic parameters specific to a site
* Fetch the static configurations
* Start the server

Now we describe the server creation methods which are available with OpenStack.

1. Server creation using openstack server creation command: This is the simplest way to create a server. First it will create a virtual machine, then it executes the script which is passed as user-data in the command line.
2. Server creation using OpenStack Heat: In this procedure of server creation, yaml file with Heat template is used with a provision to use a parameter file.

The cloud infrastructure is based on the OpenStack cloud platform, which provides several advantages as it provides maximum flexibility and allows cloud administrators to configure each component according to the requirements of the implementation system. Through the comprehensive information collection of the OpenStack cloud platform, users can have a more comprehensive understanding of the performance of the OpenStack cloud platform, and users can make some correct resource allocations based on the performance information provided.

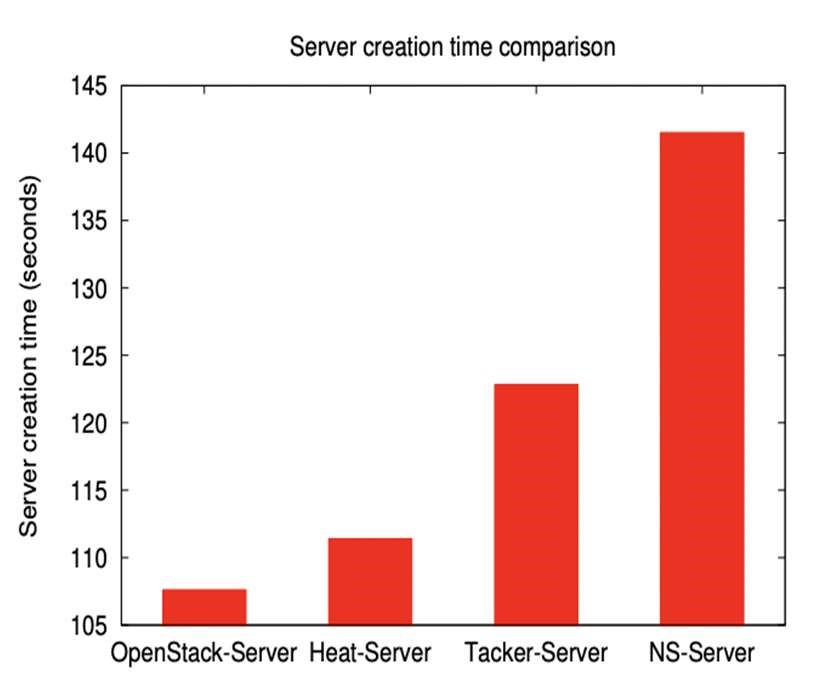
# VI. COMPARISON OF OPENSTACK ORCHESTRATION PROCEDURES

The OpenStack project is an open-source cloud computing platform. Through a series of interconnected services, OpenStack provides an Infrastructure-as-a-Service solution. Each service has an application programming interface (API) that makes integration easier. In OpenStack, there are numerous techniques for creating and deleting virtual server instances of various flavors (memory and CPU variants) and images. These methods are known as OpenStack Orchestration procedures.

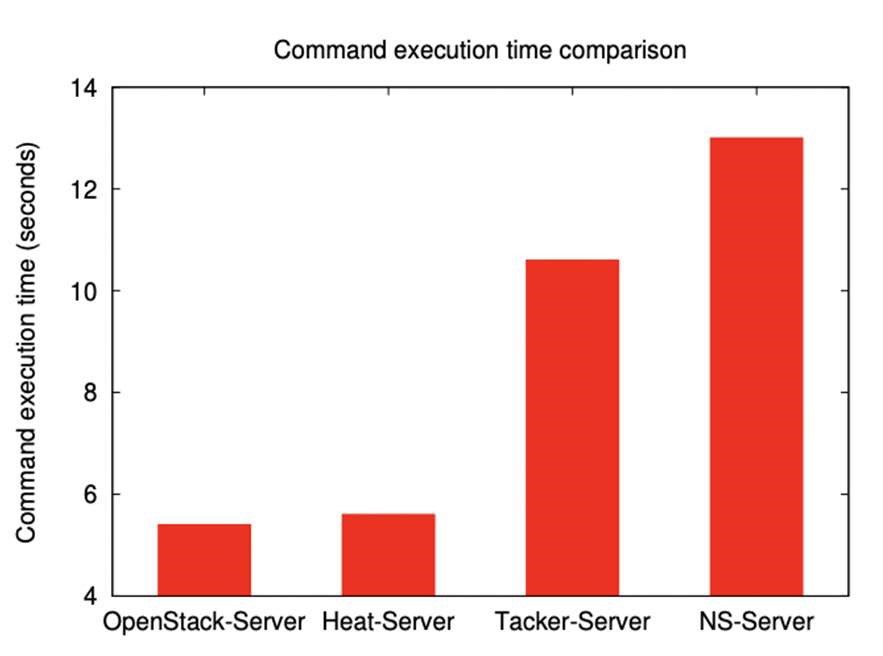
The well-known orchestration procedures are as follows:

1. Server creation using OpenStack Tacker: In this procedure of server creation, yaml file is used with a provision to use a parameter file. The format of the yaml file used in this case is TOSCA.
2. Server creation using OpenStack Mistral: In this procedure of server creation yaml file is used [1]. First the vnf descriptor is created using the yaml file explained at II-C, then the same vnf is called in the ns descriptor file.

B. Results I. Server Creation Time



II. Command Execution Time



In every scenario of comparison, we notice that utilizing OpenStack-Server to create a server takes less time than using any other OpenStack orchestration technique. However, Heat, Tacker, and Mistral all have some advantages.

If we only want to set up a single server, the OpenStackServer orchestration technique is a viable choice.

In terms of server construction time and command execution time, using Mistral, i.e., Network Service to build a single server is not a good solution.

# VII. A SCHEME AND IMPLEMENTATION OF AUTOMATIC DEPLOYMENT OF MULTILINGUAL INDUSTRIAL MECHANISM MODEL BASED ON OPENSTACK

The industrial mechanism model, which is the concentrated embodiment of the industrial Internet platform's technical capabilities, lies at the heart of the entire industrial Internet system. However, as more and more industrial mechanism models accumulate and precipitate, it is discovered that industrial mechanism models with varied languages and operation modes generate a lot of deployment, operation, and maintenance work for researchers.

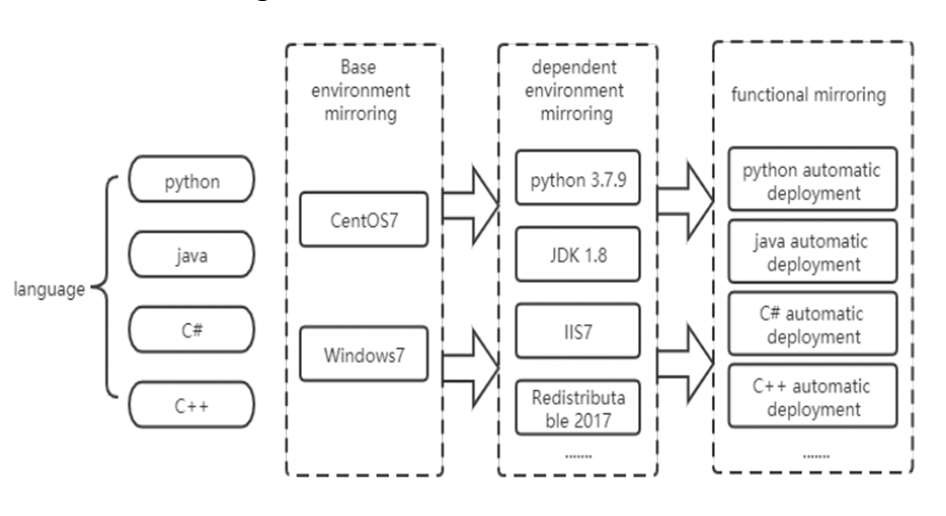
Through OpenStack, extensible and elastic cloud computing services can be provided for private enterprise clouds, and cloud computing management services can be provided with simple and extensible, rich and unified standards.

OpenStack has gotten progressively excellent after so many years of development and expansion of the open-source community. It has now encompassed every part of a network, including virtualization, operating systems, and servers. Several cloud computing platform projects are also in development, which are split into core projects, incubation projects, support projects, and associated projects based on their maturity and importance.

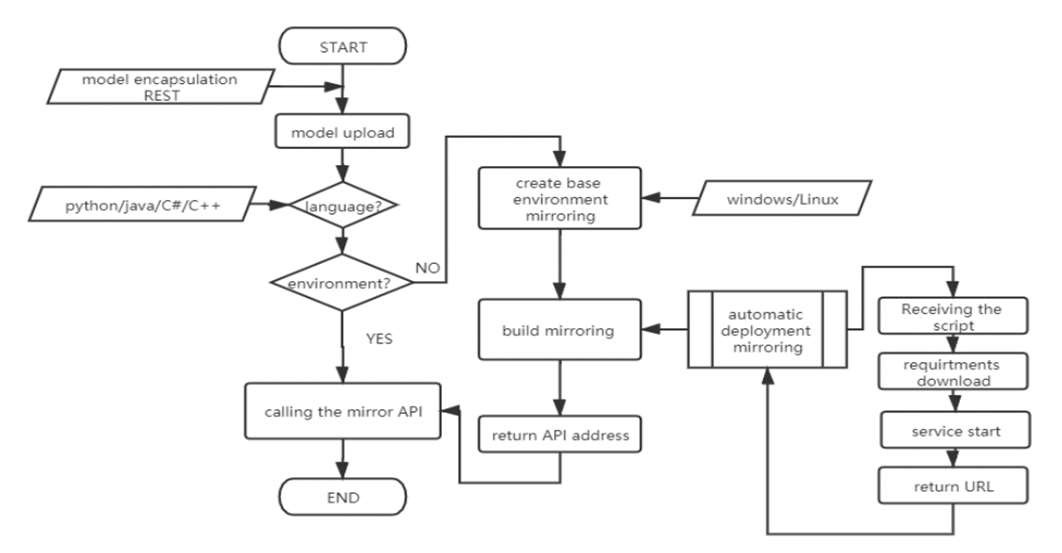
1. Introduction Of R&D Technology Scheme

The total automatic deployment scheme is divided into two parts: customization of the deployment environment and interaction with the deployment process.

* 1. Deployment environment custom scaffolding: The deployment environment consists of three layers of ready-made mirroring: base environment mirroring, dependent environment mirroring, and functional mirroring. The overall deployment is shown in the figure below



* 1. Deployment process interaction: Subsequent users can use the corresponding model deployment URL to perform model calculations and other functions.



This project seeks to automate the collecting and unified maintenance of industrial mechanism models around the industrial Internet platform, so that industrial APP developers can access them at any time, allowing for the precipitation, diffusion, reuse, and value generation of industrial information.

1. Automatic deployment is implemented in the following way :

* Get model details like the model name, model script zip package, and startup file, and compress the zip package.
* Create the model's required execution environment based on the model's requirements.
* Deployment port number setting and management:

Because there is no data source in this project's automatic deployment function image, the used port number is logged in a text file that is incremented sequentially.

* Start-up model scripts
* Simply save the model script and startup information to an XML file for further tracing.
* Use Taskkill to terminate the process by looking for the process with the corresponding port number, and then rerun the boot model script step.

C. Results

This paper proposes an automatic deployment scheme of mechanism model suitable for multiple languages, realizes the automatic construction of deployment environment and automatic deployment start script during the model uploading process, and achieves unified management and monitoring of mechanism model based on the problem of mechanism model precipitation in the industrial Internet platform and the rich API of OpenStack. It makes it easier for industrial APP developers to get in touch, encourages the collection, dissemination, reuse, and generation of value from industrial information, and gives guidance for the implementation and administration of multilingual models.

# VIII. DESIGN AND APPLICATION OF CLOUD PLATFORM BASED ON OPENSTACK IN REMOTE ONLINE COLLECTION AND MONITORING SYSTEM OF INTELLIGENT WORKSHOP

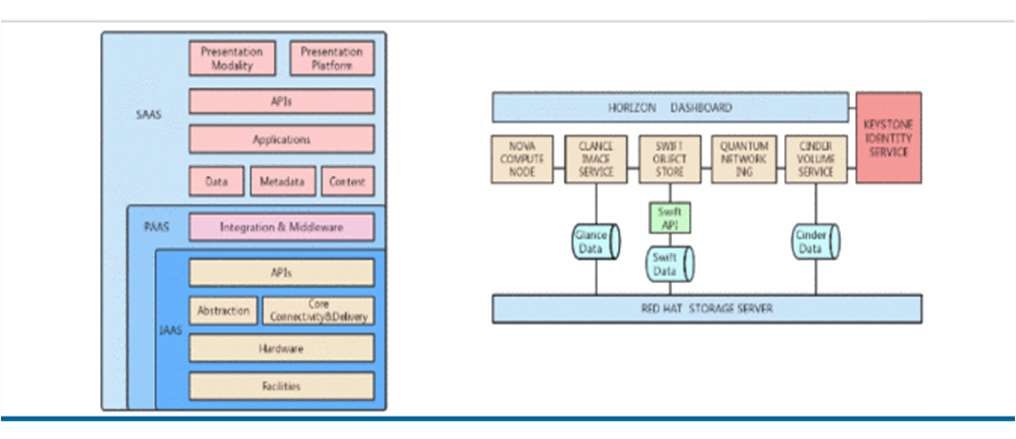
Intelligent workshop remote on-line acquisition and monitoring system is a set of integrated systems using modern sensor technology, network communication technology, Internet of Things technology, computer application technology, and cloud computing technology.

1. Architecture Design of Remote Online Collection and Monitoring System for Intelligent Workshop Based on

Openstack Cloud Computing

OpenStack is an integrated cloud computing management platform that supports deployment of public, private and hybrid clouds. OpenStack provides the basis for building networks and running universal virtual machines. It can realize resource abstraction, resource allocation, load scheduling, application lifecycle management, system operation and human-computer interaction.

1. Concept

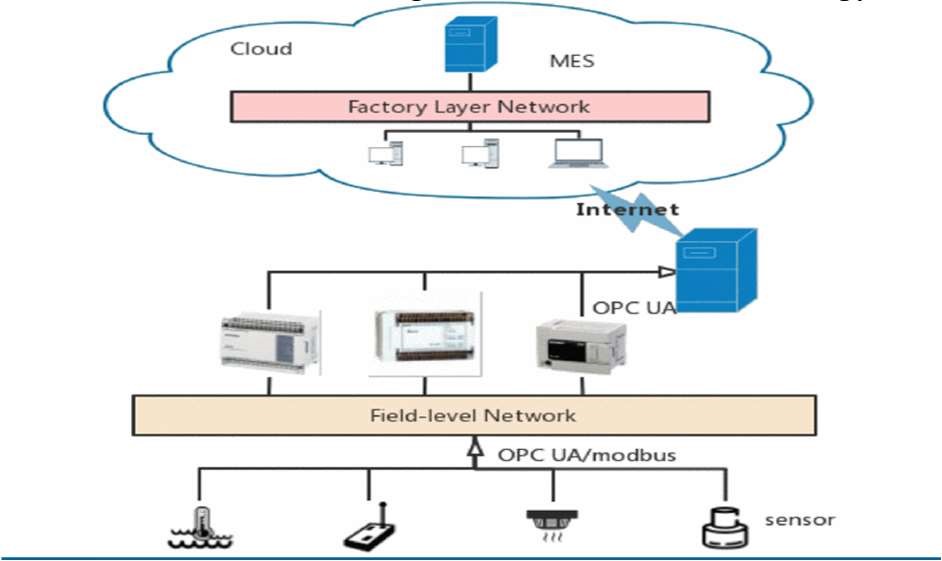


Keystone components can provide authentication services, manage users, tenants, roles, services and service endpoints. It is also the core of authentication among components. It supports SQL, PAM and LDAP as backend.

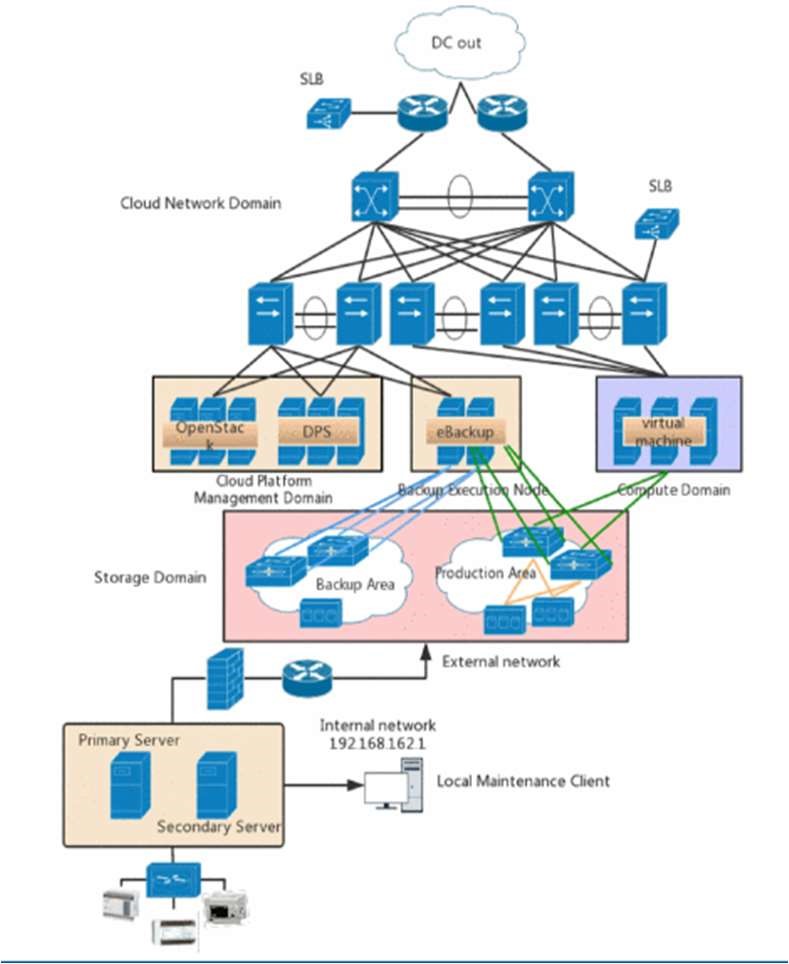
* NOVA component provides computing function.
* GLANCE provides mirroring services.
* SWIFT provides object storage services and easy-to-use services for accessing data.
* NETWORK provides network functions.
* CINDER provides block storage services.
* HORIZON provides unified interface management.
* CEILOMETER provides monitoring and measurement service.

1. The Structure of Cloud Collection and Monitoring System:

The data of all kinds of sensors are collected by OPC UA and Modbus protocol, and then transmitted to PLC through a field network. After the data is transferred to the field server gateway by PLC, the field service gateway interacts with the cloud platform through the Internet, which completes the functions of data storage, processing, analysis, prediction and display.DCI (Downlink Control Information) is used to allocate downlink and downlink resources, HARQ information, power control and so on. Installing and running OpenStack on hardware devices enables cloud server clusters to be built through virtualization technology.



Topology of Cloud Collection and Monitoring System:



1. Implementation of Remote Online Collection and

Monitoring System Architecture for Intelligent

Workshop Based on Openstack Cloud -

TCP/ IP protocol is used for remote data transmission. Data of different types of sensors are packaged and sent to cloud servers remotely at device level, thus realizing the interconnection of heterogeneous networks. A. Application of OPC UA Protocol:

After processing the request, the server returns a response message. And the information exchange adopts the asynchronous mode.

## B. Application of Modbus Rtu Protocol

Modbus is the first communication protocol used in the industrial field. The Modbus protocol is divided into three modes: RTU, ASCII and TCP.

## C. Deployment and Use of Devstack

Devstack is a rapid automated deployment Bash script for Openstack developers, providing a source

environment for assisting development and debugging

Security, Disaster Recovery and Backup of Remote Online Collection and Monitoring System Architecture for

Intelligent Workshop Based on Openstack Cloud -

A. Security of Cloud Collection and Monitoring System Cloud computing infrastructure generally has hardware firewalls, DOS-proof devices, etc. By authorizing different users and managing the privileges, security administrators fulfill the security requirements of network layer and application layer. At the same time, resource isolation and resource sharing functions are completed according to user authorization.

## B. Disaster Recovery and Backup of Cloud Collection and Monitoring System Before the Disaster

• With asynchronous remote replication, the master IPSAN generates snapshots according to the backup cycle and replicates the snapshots to the standby IPSAN.

## After the Disaster

* When the main data center fails, the administrator should stop the replication relationship stored;
* The main data center needs to repair the main site storage or use alternative storage, and restore the MN with the backup data of MN, and restore the management nodes;
* Restoring primary site storage through disaster preparedness site storage; • Finally, open the business of the main site.

E. Result

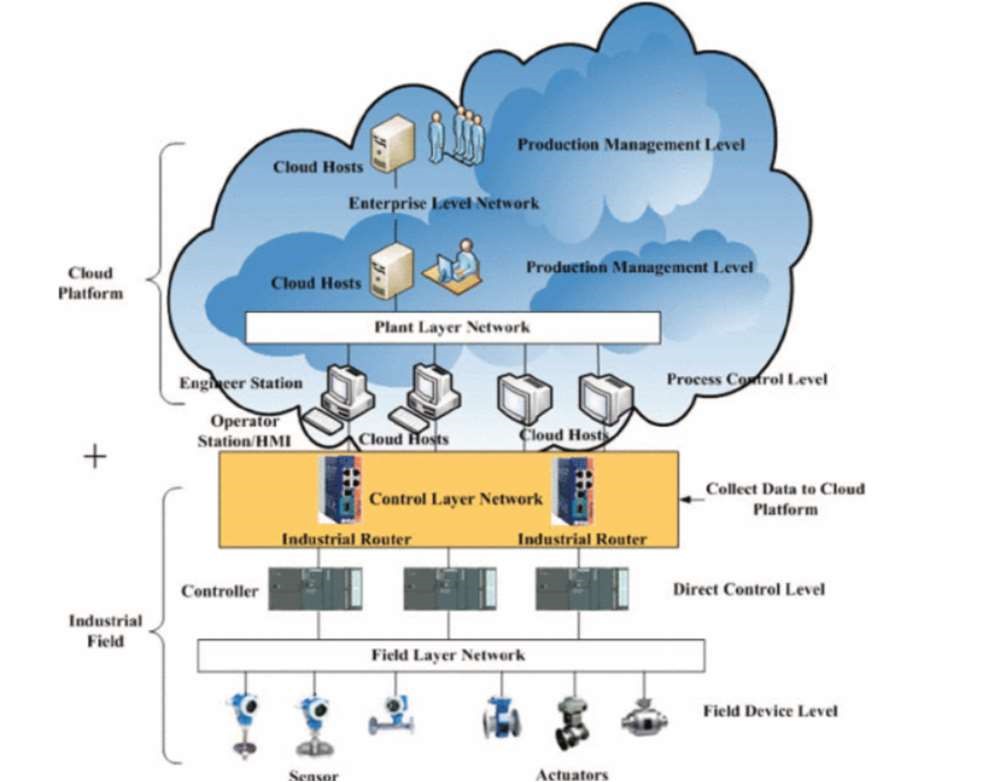
Cloud platforms use cloud computing and virtualization technology to separate physical hardware from operating systems. The server cluster is pooled so that the hardware resources of physical servers are shared by multiple virtual servers. From the local area network where the underlying field devices are located, the collected data need to be transmitted to the cloud server remotely across the network segment, and this process can realize network communication through the Internet. Moreover, this paper realizes the decoupling of hardware infrastructure and software service systems. Based on infrastructure and network connection, information sharing, data storage and remote control from point to cloud are realized. Really achieve virtualization of software services, hardware infrastructure is ready-to-use, stable and reliable.

# IX. RESEARCH ON THE ARCHITECTURE AND APPLICATION OF INDUSTRIAL CLOUD EXPERIMENTAL PLATFORM BASED ON OPENSTACK

With the rapid development of internet technology and cloud computing, the original plant HMI (human machine interface) monitoring and control system has been unable to meet the needs of the current automation system. Every level of the automation system is not independent any more, and information interaction between different levels is becoming more and more frequent. So, a highly integrated system is urgently needed to be built. Accordingly, this paper presents an industrial cloud experimental platform architecture based on OpenStack.

A. Industrial Cloud Experimental Platform Architecture

Based on Openstack



In this architecture,

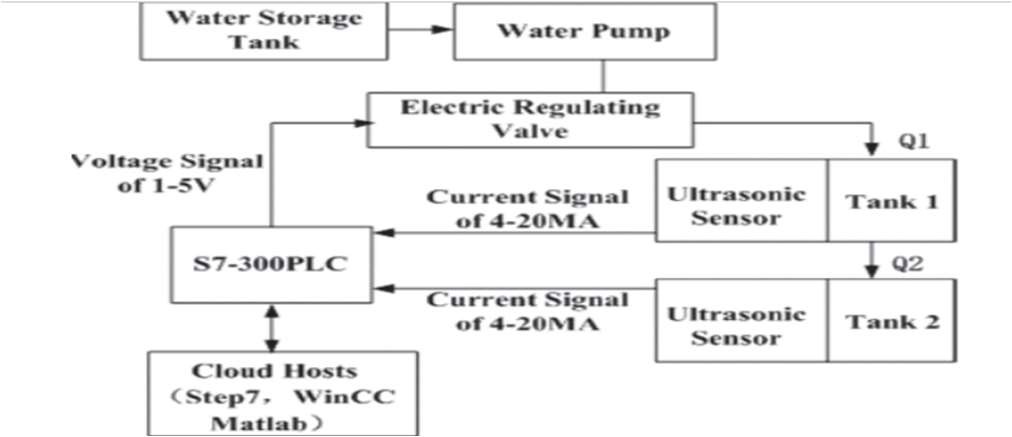
1. The field device level consists of sensors and actuators;
2. The direct control level takes PLC (Programmable Logic Controller) as controller;
3. Communication of the control layer network is the key problem. It is mainly realized by establishing communication connection between the cloud hosts(virtual machines) and PLC using the VPN(virtual private network) technology.
4. Unlike the traditional automation systems, all the applications and the intelligent software of the automatic system are deployed on cloud hosts instead of physical machines.

B. Experimental Verification

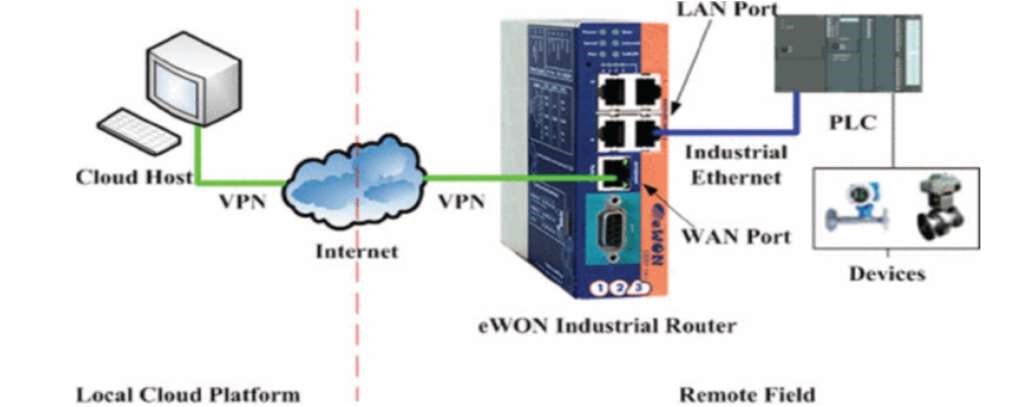
The experiment is divided into three parts:

1. devices and controller,
2. communication, 3) cloud monitoring.

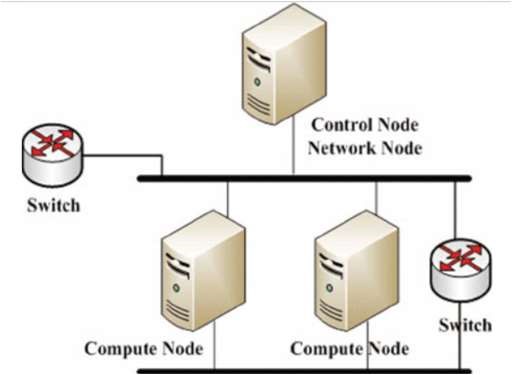
1. Device and Controller: Device and controller includes the double-tank system and the controller S7-300 PLC. The input of the controller is the current signal of 4–20MA which is collected by the ultrasonic sensors, it is converted into 0– 30cm water tank level value by the controller S7-300 PLC. The output of the controller is voltage signal of 1–5V, which corresponds to the opening value range from 0 to 100 of the electric regulating valve.



1. Communication: The VPN technology is used to solve the communication connection problem between the cloud hosts and S7-300 PLC.

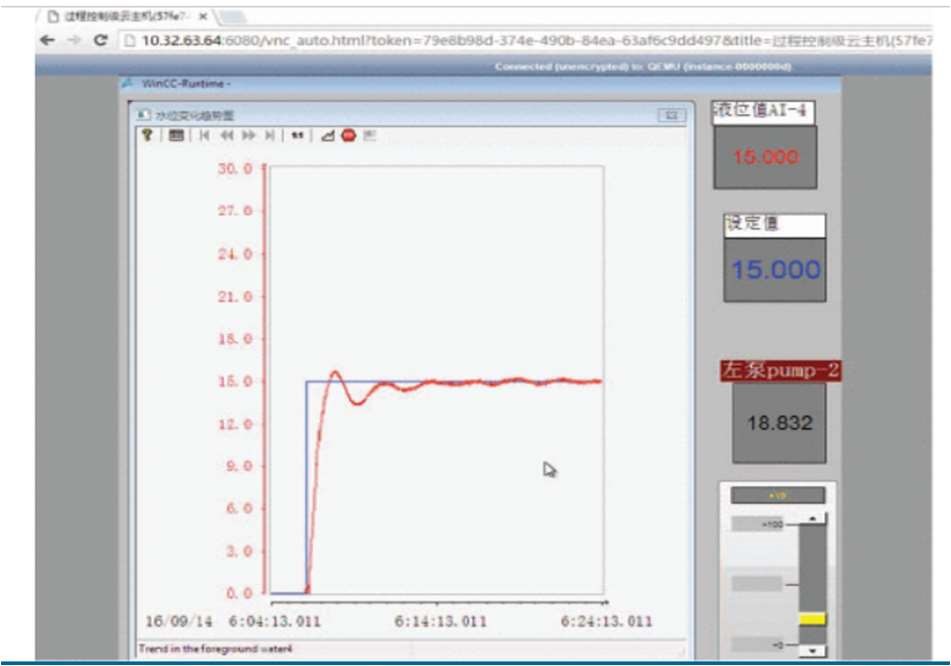


1. Cloud Monitoring: The OpenStack cloud platform consists of control nodes, computing nodes and network nodes. Compared with traditional architecture, OpenStack integrates hardware resources into a large resource pool, users get resources from the resource pool and create specific virtual machines according to performance requirements and cost requirements.



C. Results:

The experimental results show that the double-water tank system is well controlled by the OpenStack cloud platform and prove that the industrial cloud experimental platform based on OpenStack is feasible.



In this paper, an industrial cloud experimental platform architecture based on OpenStack is proposed. This paper introduces the whole architecture of the cloud platform, especially the communication technology of the control layer network and the key technology of data acquisition and HMI monitoring. The proposed architecture and technology are experimentally verified with a double-tank system. Experimental results show that the industrial cloud experimental platform is convenient to deploy, and has a strong practical value.

Compared with the traditional industrial automation system, the proposed architecture has the following advantages: 1) higher integration, different levels of automation systems are deployed on the OpenStack cloud platform;

2) better accessibility, users can operate the controlled objects and the cloud platform through a web browser, which is convenient and time-saving.

# X. OPENSTACK AND DOCKER: BUILDING A HIGHPERFORMANCE IAAS PLATFORM FOR INTERACTIVE SOCIAL MEDIA APPLICATIONS

The current development and broad adoption of cloud technologies brought in new alternative technologies that are being used to make the launch of applications in the cloud faster and easier. Interactive social media applications are faced with the challenge of efficiently provisioning new resources in order to meet the demands of the growing number of application users. This paper describes how Docker can run as a hypervisor, and how we managed to enable the fast provisioning of computing resources inside of an OpenStack IaaS using the novadocker plugin that we developed.

1. Introduction

The paper presents a free open source technology for IaaS platforms for social media applications using Docker as a hypervisor using OpenStack. The researchers used nova docker driver, along with some patches to reduce boot time, to allow deployment and handling of multiple Docker instances on OpenStack. The service model presented in this paper is an IaaS model consisting of OpenStack running both KVM instances and Docker images to make application deployment faster and efficient. This is outlined as the capability provided to the consumer to access and manage processing power, storage space, network speed, and other fundamental computing resources, where the consumer is able to install and manage any software type, which can include both operating systems and personal applications.This paper presents the advantages of running containers on top of bare metal compute nodes inside IaaS platforms through the implementation of Docker as a hypervisor inside IaaS platforms based on OpenStack.

1. Nubomedia

The researchers demonstrated the use of the proposed architecture by deploying Nubomedia, an elastic PaaS. NUBOMEDIA is somehow unique because of its cloud infrastructure that acts as a single virtual supercomputer incorporating all resources available on one basic server. Because of this, NUBOMEDIA applications can adaptively scale to meet the required load, maintaining Quality of Service, Quality of Experiment and Service Level Agreement guarantees.

1. Docker v/s VM:

Docker is a free open platform for developers and system administrators which allows them to create, deploy, and run microservice based applications. With the use of Docker Engine, which is a lightweight and portable packaging tool, and Docker Hub, as a service that enables the distribution of the Docker containers, Docker allows applications to be rapidly packed from different components and deployed. KVM (Kernel-based Virtual Machine) is a full software virtualization hypervisor that can be found on all Linux distribution kernels. The main hardware able to run KVM are Intel or AMD processors.

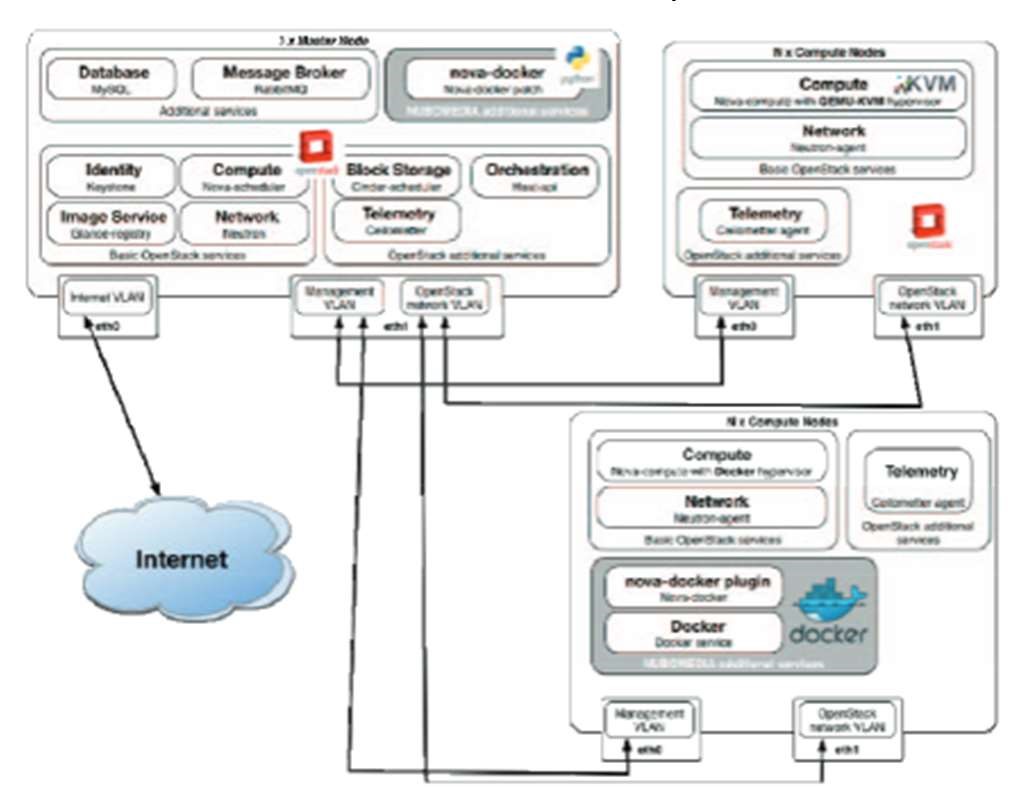
1. Standard architecture

A standard architecture of OpenStack is by default composed of the micro-service with nova-compute having qemu-kvm as a hypervisor. OpenStack works with many hypervisors, with Docker gaining popularity over recent years.

1. Openstack with Docker

The scope of the paper is to enable users to create and manage Docker

containers in a consistent way with how they are used to start virtual machines using the Nova service. Docker can be used to manage multiple containers on a single physical machine. However, when it is used in Nova, it becomes much more powerful since it becomes possible to manage several hosts, which in turn manage hundreds of containers. The paper proposes an architecture where Docker images and VM instances run side by side to allow users to deploy applications of varying size quickly. For smaller applications, it’s wasteful to initialize an entire VM instance. Instead a simple Docker image can be used as it’s very lightweight and communicates with the hardware directly.



OpenStack Architecture with both Docker and KVM as hypervisors

1. Result

Deploying Iaas architecture requires choosing between virtual machines and Docker containers. Containers have a huge advantage over VMs because of performance improvements and reduced startup time. The paper solved the issue of a long boot time by adding a patch to nova docker driver that significantly reduced boot time making it feasible to use docker containers over OpenStack.

# XI. ENABLING SECURE RESTFUL WEB SERVICES IN IOT USING OPENSTACK

Thanks to the impact of the advancement in the hardware field, the network size and usage scope of the Internet are continuously growing. Indeed, new smart devices, e.gSensors, actuators, home appliances are becoming strong enough to communicate and exchange data over the Internet. To conceive new IoT services, enabling the smart devices to join the Internet and expose their capabilities/data through the Web is fundamental. For this purpose, Web Application Programming Interfaces (APIs) or what we refer to also as RESTful Web Services is a paradigm that can enhance the IoT application scope by making smart things part of the Web.

1. Introduction

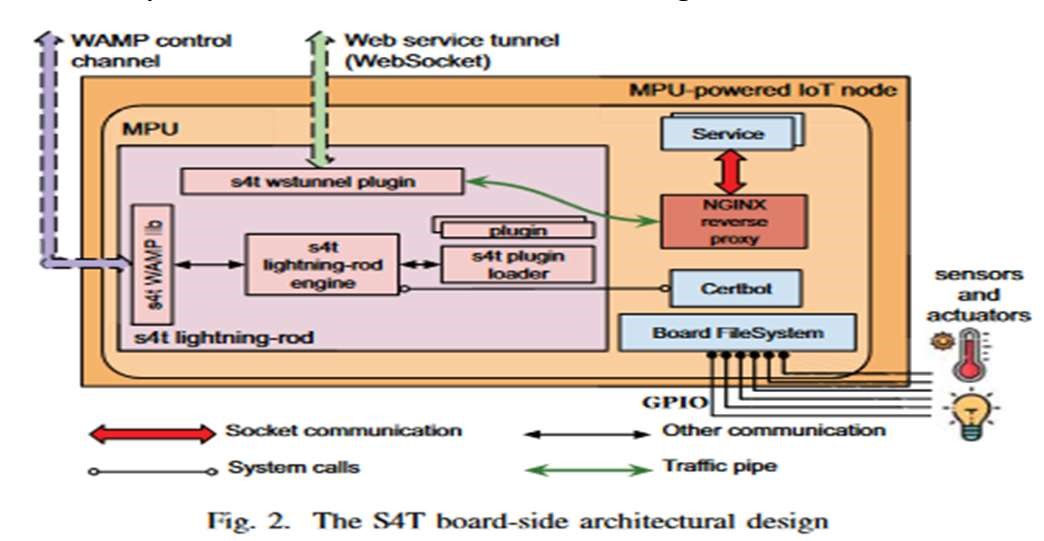
Nowadays, with the advances in the hardware design and production fields, embedded systems’ prices have decreased significantly thereby, making their adoption faster. Accordingly, in the past few years, a new trend that aims at embedding computational capability within everyday objects (e.g., light bulbs, HVAC systems, doors) has shown up under the umbrella of a paradigm known as pervasive/ubiquitous computing. In the vision of IoT, the Internet is spreading rapidly beyond its classical core composed by powerful servers and other fringe machines/devices (e.g., computers, smartphones) to billions of constrained embedded devices (e.g., sensors and actuators). With its capacity to connect objects/assets to the Internet and blending digital artifacts with the physical world, the IoT will mark a revolution of how people communicate and interact with their surrounding environments.

1. Stack4Things:

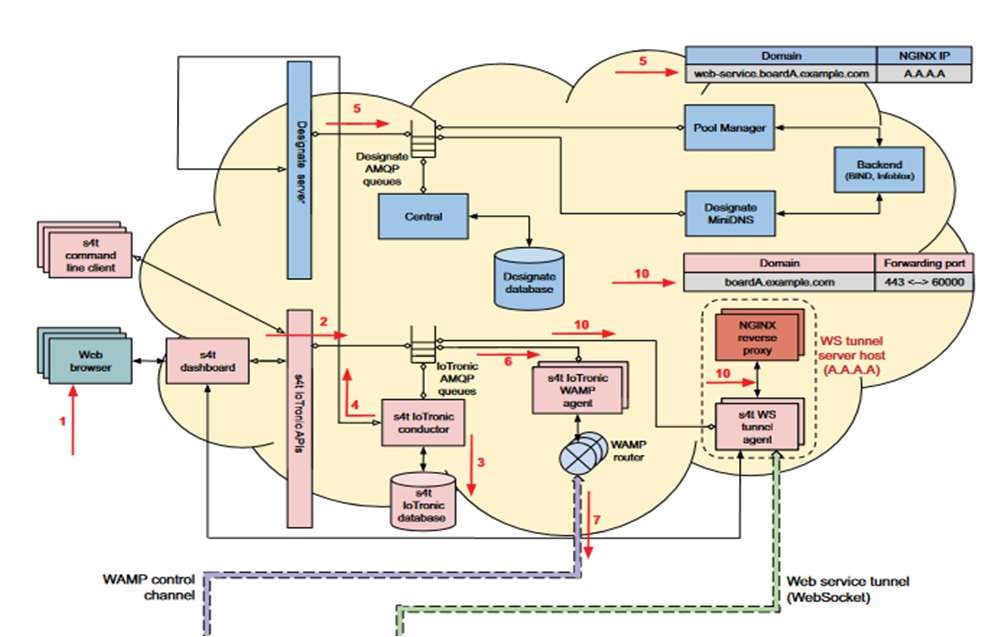
Stack4Things is an OpenStack-based Internet of Things framework developed by the Mobile and Distributed Systems Lab (MDSLab) at the University of Messina, Italy. Stack4Things is an open source project that helps you in managing IoT device fleets without caring about their physical location, their network configuration, or their underlying technology. It is a Cloud-oriented horizontal solution providing IoT object virtualization, customization, and orchestration.

1. Implementation:

To deploy our approach, the researchers extended the OpenStack Horizon dashboard with specific panels for IoT. Also, they developed a set of APIs to make the system able to interact with other systems. They used a Raspberry Pi as an IoT device to implement our approach and prove its feasibility. The board runs the S4T LR agent as well as an



Apache Web server (listening on port 5000) to serve a Web page showing the data of two sensors hosted on that board. In this deployment, a user agent can interact with the resources hosted on the board via HTTPS requests, just like any other resource on the Web.



1. Result:

In this paper, we introduced our system for managing and deploying Web services on remote IoT devices. RESTful Web services for IoT are an appealing paradigm that will open the door for ecosystems’ interoperability and better integration for IoT with the Web. Therefore, IoT service will become easier to build and interact with other systems.

# XII. CONCLUSION

OpenStack has multiple advantages. It has a vibrant ecosystem, and it is open source and free. Nowadays, more companies are beginning to adopt OpenStack as a part of their cloud tool kit. It is being used by a number of companies of varying sizes including giants like Yahoo, Tencent, Target and smaller, lesser known companies and organizations.

OpenStack has its fair share of advantages and challenges like any other software. OpenStack is open source allowing anyone to check its code and suggest modifications, fix bugs and release patches. It is easily scalable, easy to automate, has a large and welcoming community and allows fast development.

On the other hand, it’s installation isn’t straightforward and easy, has a steep learning curve, many documentations as it is open source and doesn’t have long term support as it is constantly upgrading.

OpenStack is one of the best cloud computing environments in the market. The databases of many top-notch companies have integrated OpenStack in their back-end. The ease of linear scalability and open-source nature has attracted many customers and technology enthusiasts to come forward and contribute to the development. This has made OpenStack, only better with the years. Seeing all the advantages and endless scalable features, OpenStack can be considered for cloud computation as it proves to be an affordable solution for the longer run.

REFERENCES

1. B. Mishra, "Guidelines on how to contribute to the development of OpenStack," 2017 International Conference on Computing Networking and Informatics (ICCNI), 2017, pp. 1-6, doi: 10.1109/ICCNI.2017.8123813.
2. Z. Kai, L. Youyu, L. Qi, S. C. Hao and Z. Liping, "Building a private cloud platform based on open source software OpenStack," 2020 International Conference on Big Data and Social Sciences (ICBDSS), 2020, pp. 84-87, doi: 10.1109/ICBDSS51270.2020.00027.
3. P. P. Deshmukh and S. Y. Amdani, "Virtual Memory Management of Private Cloud using OpenStack API," 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), 2019, pp. 1-4, doi: 10.1109/I2CT45611.2019.9033916.
4. H. Wang, X. Zhang, Z. Ma, L. Li and J. Gao, "An Microservices-Based OpenStack Monitoring System," 2022 11th International Conference on Educational and Information Technology (ICEIT), 2022, pp. 232236, doi: 10.1109/ICEIT54416.2022.9690713.
5. P. Mandal and R. Jain, "Comparison of OpenStack Orchestration Procedures," 2019 International Conference on Smart Applications, Communications and Networking (SmartNets), 2019, pp. 1-4, doi: 10.1109/SmartNets48225.2019.9069772.
6. C. Zhang, E. Sun, X. Cheng, D. Zang and Z. Chen, "A Scheme and Implementation of Automatic Deployment of Multilingual Industrial Mechanism Model Based on OpenStack," 2021 IEEE International Conference on Advances in Electrical Engineering and Computer

Applications (AEECA), 2021, pp. 76-79, doi:

10.1109/AEECA52519.2021.9574124.

1. F. Han, Y. Hu, D. Yu, N. Cui and Q. Fu, "Design and Application of Cloud Platform Based on OpenStack in Remote Online Collection and Monitoring System of Intelligent Workshop," 2019 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), 2019, pp. 1-5, doi: 10.1109/ICSPCC46631.2019.8960725.
2. L. Zhen-wu, Z. De-min, S. Yun-tao and S. De-hui, "Research on the architecture and application of industrial cloud experimental platform based on OpenStack," 2017 29th Chinese Control And Decision

Conference (CCDC), 2017, pp. 7401-7406, doi:

10.1109/CCDC.2017.7978523.

1. Z. Benomar, F. Longo, G. Merlino and A. Puliafito, "Enabling Secure RESTful Web Services in IoT using OpenStack," 2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), 2020, pp. 410-417, doi: 10.1109/MASS50613.2020.00057.
2. Calinciuc, C. C. Spoiala, C. O. Turcu and C. Filote, "OpenStack and Docker: Building a high-performance IaaS platform for interactive social media applications," 2016 International Conference on Development and Application Systems (DAS), 2016, pp. 287-290, doi:

10.1109/DAAS.2016.7492588.