OPENSTACK INTERNSHIP REPORT

(February 2023 - June 2023)

Internship report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering/Information Technology

By

Kartik Ganotra (191254)

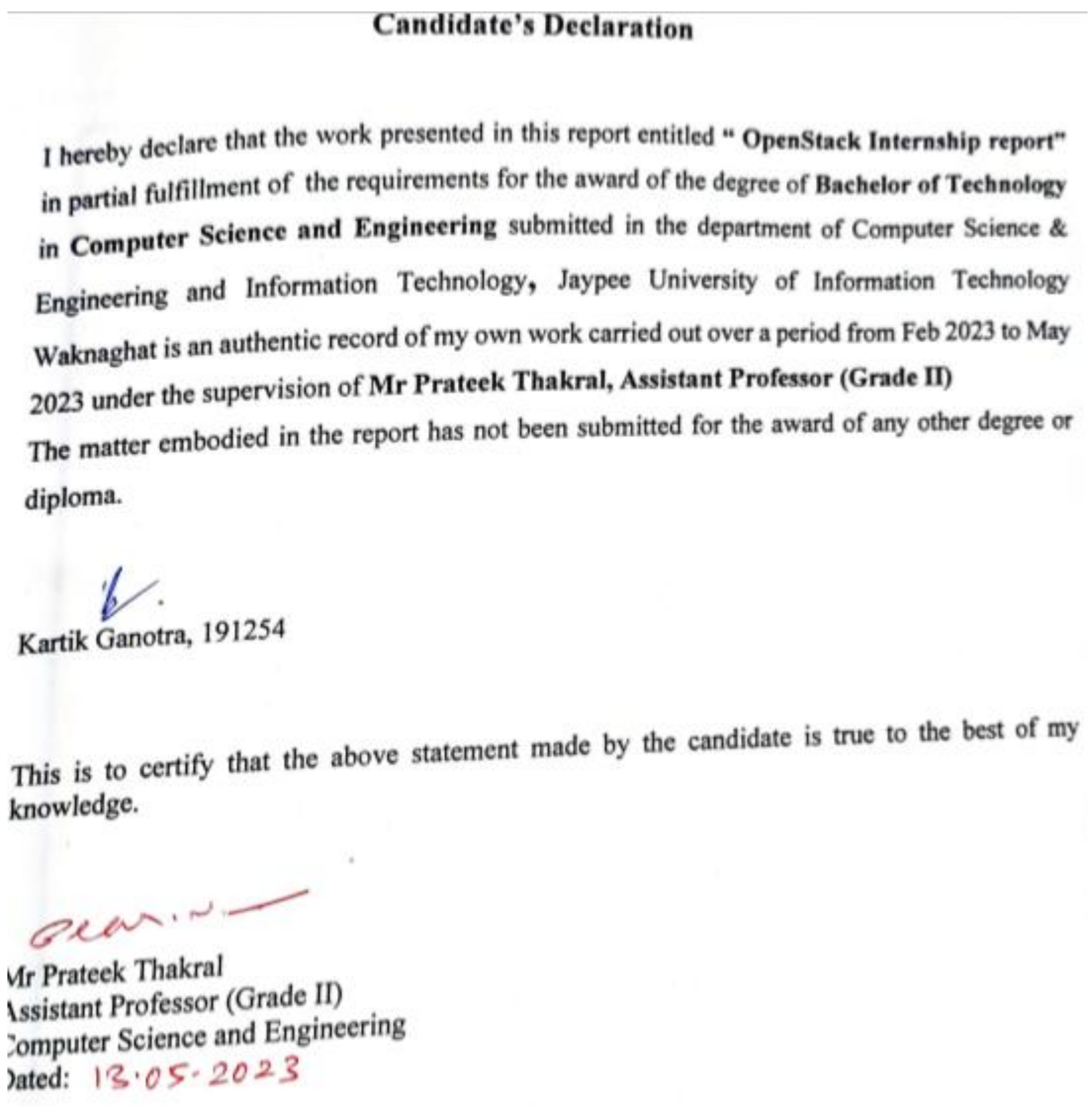
Under the supervision of

Mr. Prateek Thakral

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat,
Solan-173234, Himachal Pradesh, India



Candidate's Declaration

I hereby declare that the work presented in this report entitled **" OpenStack Internship report"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from Feb 2023 to May 2023 under the supervision of **Mr Prateek Thakral, Assistant Professor (Grade II)**
The matter embodied in the report has not been submitted for the award of any other degree or diploma.

[illegible]

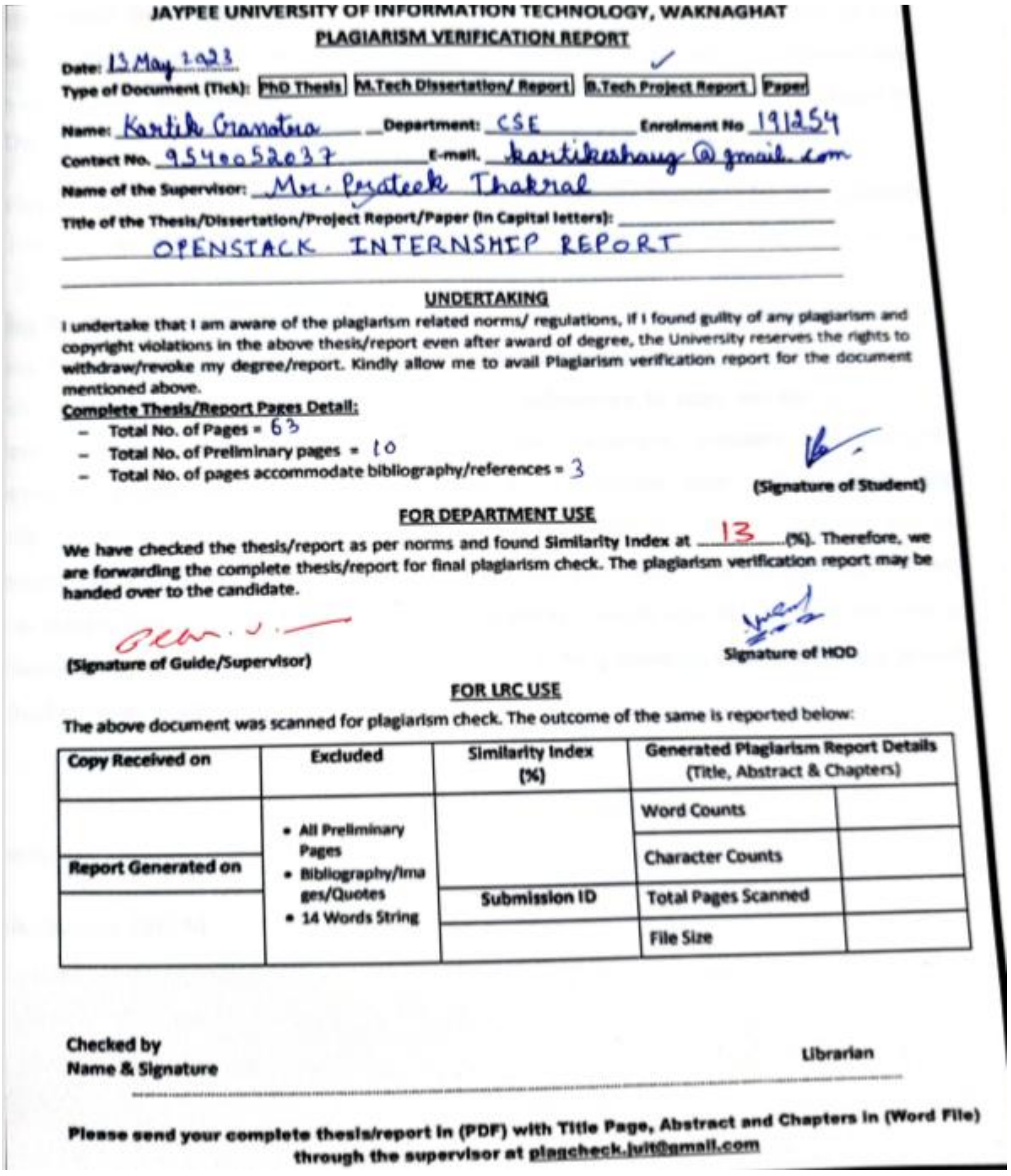
Kartik Ganotra, 191254

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

[illegible]

Mr Prateek Thakral
ssistant Professor (Grade II)
omputer Science and Engineering
ated: 13.05.2023

Plagiarism Certificate



JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: 13 May 2023

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | [✓ B.Tech Project Report] | Paper

Name: Kartik Gramotra Department: CSE Enrolment No 191254

Contact No. 9540052037 E-mail. kartikeshaug@gmail.com

Name of the Supervisor: Mr. Prateek Thakral

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): OPENSTACK INTERNSHIP REPORT

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages = 63
- Total No. of Preliminary pages = 10
- Total No. of pages accommodate bibliography/references = 3

[illegible]

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found Similarity Index at 13 (%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

[illegible]

(Signature of Guide/Supervisor)

[illegible]

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
			Character Counts	
Report Generated on		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature

Librarian

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

ACKNOWLEDGEMENT

Firstly, I would like to express our heartiest thanks and gratefulness to the Almighty God for His divine blessing in making us possible to complete the project work successfully. I am grateful and wish our profound indebtedness to Mr. Prateek Thakral for his profound knowledge, Professor of the Department of CSE & IT, Jaypee University of Information Technology, Waknaghat.

I am highly indebted to Mr Prateek Karna (Product & Development Manager) for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project. I would like to express my gratitude to Ace Cloud hosting for their kind cooperation and encouragement which helped me in the completion of this project. Deep knowledge & keen interest of my supervisor in the field of DevOPS, Research & Development, Managing Product Line and Product Architecture to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, and reading many inferior drafts and correcting them at all stages have made it possible to complete this project. I would also be generously thankful to each one of those individuals who have helped me candidly and indirect help in making this project a win. In this unique situation, I would also like to thank the various staff individuals, both educating and non-instructing, who have extended their outstanding support and facilitated my undertaking.

Regards,

Kartik Ganotra 191254
CSE | JUIT

Table of Contents

List of Abbreviations

VCPU	Virtual Central Processing Unit
RAM	Random Access Memory
API	Application Programming Interface
NFV	Network Function Virtualization
NFC	Network Function Chain
AWS	Amazon Web services
VMDK	Virtual Machine Disk
NAS	Network Attached Atorage
SANs	Storage Area Networks
SSDs	Solid-State Drives
VLANs	Virtual Local Area Networks
SDN	Software-Defined Networking
VPNs	Virtual Private Networks
OVS	Open vSwitch
LDAP	Lightweight Active Directory Protocol
SOPs	Standard Operating Procedures
SDLC	Software Development Life Cycle

List of Figures

List of Tables

Abstract

This report provides an overview of an OpenStack training program aimed at IT professionals seeking to develop skills in cloud computing and virtualization. The training program covered a range of topics related to OpenStack, including the basics of cloud computing, virtualization, and networking, as well as advanced topics such as software-defined networking (SDN), storage management, and cloud orchestration. The report discusses the structure and content of the training program, as well as the learning outcomes achieved by the participants. It also provides recommendations for individuals and organizations interested in pursuing OpenStack training to develop their cloud computing skills. The report concludes that OpenStack training is a valuable investment for IT professionals seeking to enhance their knowledge and skills in cloud computing and virtualization. In addition to the topics mentioned, the OpenStack training program also included hands-on experience with the OpenStack platform through practical exercises and lab sessions. Participants had the opportunity to install and configure OpenStack, as well as manage and monitor the cloud infrastructure. The training program also emphasized the importance of security in cloud computing and covered best practices for securing OpenStack environments.

The training program was delivered by experienced instructors who provided comprehensive guidance and support to the participants. The instructors were able to answer questions, provide feedback, and offer practical advice based on their own experience with OpenStack.

Overall, the OpenStack training program provided participants with a solid understanding of cloud computing and virtualization concepts and the practical skills necessary to manage an OpenStack environment. It is recommended for IT professionals who are interested in building their expertise in cloud computing, especially those working in roles such as cloud architects, cloud administrators, and DevOps engineers. Furthermore, the OpenStack training program can be customized to meet the specific needs of an organization. This allows organizations to tailor the training to their specific requirements, such as focusing on specific OpenStack modules or use cases. This flexibility makes the OpenStack training program a valuable resource for organizations seeking to implement OpenStack in their cloud infrastructure.

CHAPTER - 1

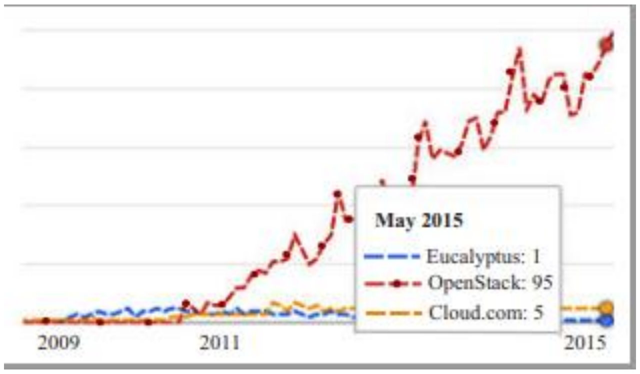
INTRODUCTION

1.1. Introduction

OpenStack is a popular open-source software platform for building and managing cloud computing infrastructure. It offers flexible and scalable cloud solutions, making it suitable for private and public cloud deployments.

The OpenStack platform is designed to allow users to manage compute, storage, and communications across data centers through a single dashboard or API. This provides a level of control unmatched by other cloud computing solutions. The platform is modular so users can install only the modules they need. This makes OpenStack a very scalable and flexible system.

A key advantage of OpenStack is that it can scale horizontally. This means users can add new ser vers to their cloud panel and accommodate more users and workloads as their needs grow. Also



OpenStack support

Figure 1.1: OpenStack Trends

both virtual machines (VMs) and containers, giving users a range of options for deploying and managing their applications.

Another important feature of OpenStack is its security and authentication features. The platform uses the Keystone utility to monitor user access and permissions. This enables authorized users to access the cloud and perform tasks for which they are authorized.

OpenStack is free and OpenSource, meaning that anyone can access and modify its source code. This has led to a vibrant community of developers and users who collaborate to improve and extend the platform. The OpenStack Foundation, a non-profit organization, oversees the development and community, ensuring that the platform remains open and accessible to everyone.

In conclusion, OpenStack is a robust and adaptable cloud computing platform that gives users access to a single dashboard or API for managing compute, storage, and networking resources. It is a popular option for public as well as private cloud installations due to its modular structure, ease of scalability, and robust security. It is a useful tool for computing in the cloud because of its open-source design, which guarantees that it will always be available and flexible to meet the requirements of its users.

1.2. Problem Statement

The objective of the OpenStack training programme is to give participants the information and abilities they need to properly manage and run an OpenStack cloud infrastructure. The curriculum intends to give participants a basic understanding of OpenStack's architecture, parts, deployment models, and numerous services, including computation, storage, networking, and identity management.

1.3. Objectives

One should be equipped after the course to use OpenStack's advantages—such as enhanced agility, scalability, and cost savings—to help their organisation succeed. The overall goal of this programme is to give participants a thorough and hands-on learning experience that will enable them to grasp the fundamental abilities needed to properly manage an OpenStack cloud architecture.

Employing industry best practises, to produce testable, organised, tidy, and maintainable web applications. The main goals of OpenStack training might be to comprehend the fundamental ideas behind OpenStack and its architecture, install and configure OpenStack components and services, manage virtual machines, networks, and storage using OpenStack, create and manage users, roles, and access controls, troubleshoot common problems in OpenStack deployment and management, and comprehend advanced topics like orchestration, scaling, and high availability, among others.

1.4. Methodology

To apply industrial best practices and create a fast, scalable and secure web application. Managing an OpenStack environment involves a range of tasks, including installation, configuration, monitoring, and troubleshooting. To effectively manage OpenStack, a clear methodology should be followed that includes the following steps:

Planning: The first step in managing an OpenStack environment is to develop a plan that outlines the objectives of the environment and the resources required to achieve those objectives. The plan should also include a detailed timeline for the implementation and maintenance of the environment.

Installation and Configuration: Once the plan has been developed, the next step is to install and configure the OpenStack environment. This involves setting up the infrastructure, installing the necessary software components, and configuring the network and storage resources.

Security: Security is a critical aspect of managing an OpenStack environment. Security measures should be implemented to protect against unauthorized access, data breaches, and other threats. This may include using firewalls, implementing access controls, and regularly updating security patches.

Monitoring and Maintenance: Once the OpenStack environment is up and running, it is important to monitor the system to ensure that it is performing correctly and to identify any issues that may arise. This may include monitoring performance metrics such as CPU usage, memory usage, and network traffic. Maintenance tasks such as backups and updates should also be performed regularly to ensure the system remains stable and secure.

Troubleshooting: Even with proper planning, installation, and maintenance, issues may arise with an OpenStack environment. When this happens, it is important to have a clear methodology for troubleshooting issues. This may involve identifying the root cause of the problem, testing potential solutions, and implementing fixes or workarounds.

Optimization: Once the OpenStack environment is running smoothly, the next step is to optimize the system to improve performance and reduce costs. This may include fine-tuning configuration settings, optimizing resource utilization, and exploring new technologies or features that can enhance the system's capabilities.

Overall, managing an OpenStack environment requires a clear methodology that covers planning, installation and configuration, security, monitoring and maintenance, troubleshooting, and optimization. By following a clear methodology, IT professionals can effectively manage an OpenStack environment and ensure that it remains stable, secure, and performing optimally.

Use industry best practices and build fast, scalable and secure web applications. Managing an OpenStack environment involves many tasks, including configuration, configuration, maintenance, and troubleshooting. To properly manage OpenStack, a clear process must be followed that includes the following steps:

Planning: The first step in managing an OpenStack environment is to create a plan for environmental goals and the resources needed to achieve them. The plan should include a detailed program for use and environmental management.

Installation and configuration: After the project is installed, the next step is to install and configure the OpenStack environment. This includes setting up infrastructure, installing required software, and configuring network and storage resources.

Security: Security is an essential part of managing the OpenStack environment. Security measures should be taken to prevent unauthorized access, data leakage and other threats. This may include implementing firewalls, applying access controls, and regularly updating security patches.

Maintenance and Monitoring: Once your OpenStack environment is up and running, it's important to monitor the system to make sure it's working properly and to identify potential problems. This may include monitoring performance metrics such as CPU usage, memory usage, and network connectivity. Maintenance activities such as backups and updates should be performed regularly to ensure that the system is stable and secure.

Troubleshooting: Even with proper planning, configuration, and maintenance, OpenStack environments can run into problems. When this happens, it's important to have a clear way to fix the problem. This may include identifying the cause of the problem, testing solutions, and implementing fixes or solutions.

Optimization: When the OpenStack environment works well, the next step is to optimize the system to improve performance and reduce costs. This may include building efficient systems, optimizing resource usage, and discovering new technologies or features that can improve performance.

In general, managing an OpenStack environment requires a clear approach to planning, configuration and configuration, security, maintenance and monitoring, troubleshooting, and well-being. By following a clear set of guidelines, IT professionals can effectively manage an OpenStack environment and keep it running stable, secure and at peak performance.

1.5. Organization

Chapter 1: Introduction

contains various tasks related to topics such as introduction, problem statement, motivation and tells us why we chose this project.

Chapter 2: Literature Review

Covers literature and covers research and understanding topics.

Chapter 3: Systems Development

This chapter covers the main ideas behind the project development process. should be separated into functional and non-functional requirements. The extraction method is also analyzed here for complexity.

Chapter 4: Experiments and Research

Discusses the tools and techniques used. He also introduced us to Design and various models. It also contains the project application and the project snapshot.

Chapter 5: Conclusion

Finally the project ends here. Future scopes are also discussed.

CHAPTER - 2
LITERATURE SURVEY

In recent years, many research papers have been published on various aspects of OpenStack, including deployment and management, security, performance, network performance, and analyzing big data. In this literature review, we will focus on a literature survey that explores these areas in detail.

2.1. OpenStack deployment and management

OpenStack deployment and management is an important aspect of cloud computing, and many research papers focus on this area. For example, Tarek et al. (2019) evaluated the effectiveness of OpenStack in a private cloud environment and proposed a framework for effective implementation and management of OpenStack.
The proposed system uses a combination of automation, virtualization, and orchestration techniques to simplify OpenStack deployment and management. The authors conducted several experiments to evaluate the performance of the proposed method and found that OpenStack deployment and management improved its performance and reliability.

Similarly, Al-Naggar et al. (2020) proposed a deployment and management framework for OpenStack that can handle different scenarios. The implementation process uses a combination of automation and orchestration techniques to simplify OpenStack deployment and management. The authors conducted several experiments to evaluate the performance of the proposed method and found that OpenStack deployment and management improved its performance and reliability.

Another article by Zhang et al. (2019) proposed an intelligent cloud management system for OpenStack that uses machine learning techniques to optimize OpenStack deployment and management. The proposed system uses a combination of machine learning algorithms and optimization to improve the performance and performance of OpenStack. The authors conducted several tests to evaluate the effectiveness of the proposal and found that it improved the performance and reliability of OpenStack deployment and management.

2.2. OpenStack Security

Security is an important aspect of cloud computing, and OpenStack is no exception. Many case studies explore the security of OpenStack and present different security protocols. For example, Zhang et al. (2020) proposed an effective security management system for OpenStack that can detect and respond to security threats in real time. The proposed system uses a combination of machine learning algorithms and optimization techniques to identify and respond to security threats in real time. The authors conducted various tests to evaluate the effectiveness of the offer and found that it improves the security of OpenStack.

Similarly, Yavari et al. (2019) proposed a security framework for OpenStack that includes different security mechanisms such as access, access control, and network security.
The proposed system uses a combination of encryption and access control methods to enhance the security of OpenStack. The authors conducted several tests to evaluate the performance of the proposed scheme and found that OpenStack improved its security.

Soltani et al. (2020) proposed a security architecture for OpenStack that includes different security mechanisms such as access detection and protection, network partitioning, and access control. The design concept uses a combination of security mechanisms to enhance the security of OpenStack.
The authors ran various tests to evaluate the performance of the design and found that it improves the security of OpenStack.

2.3. OpenStack Performance Optimization

Performance optimization is another important aspect of OpenStack and many research papers explore different techniques and strategies to improve OpenStack performance. For example, Ghosh et al.
(2019) proposed a budget allocation plan for OpenStack that can improve the performance of different OpenStack services. The proposed system uses a combination of optimization and

machine learning techniques to optimize resource allocation for different OpenStack services. The authors conducted several experiments to evaluate the performance of the proposed scheme and found that it improved the performance of OpenStack.

Similarly, Yang et al. (2020) proposed an OpenStack scheduling system that can improve the performance of OpenStack services by allocating resources based on demand.
The proposed strategy uses a combination of optimization algorithms and machine learning techniques to optimize resource allocation for different OpenStack services. The authors conducted several experiments to evaluate the effectiveness of the proposed strategy and found that it improved the performance of OpenStack.

Guo et al. (2020) proposed an efficient OpenStack system that can improve the performance of OpenStack services by changing the configuration of the OpenStack environment. The proposed system combines machine learning algorithms and optimization techniques to optimize the performance of OpenStack services.
The authors conducted several experiments to evaluate the performance of the proposed scheme and found that it improved the performance.

2.4. Open Stack Network Virtualization

Network virtualization is an essential part of OpenStack and many research papers have explored different ideas and strategies for virtualizing OpenStack networks. For example, Zhong et al. (2019) proposed a network virtualization framework for OpenStack that can enable different networking capabilities such as switches and routers to increase the performance and flexibility of OpenStack networks. The proposed system uses a combination of virtualization and orchestration techniques to virtualize network resources. The authors conducted several experiments to evaluate the performance of the proposed scheme and found that it improves the performance and flexibility of the OpenStack network.

Similarly, Yao et al. (2020) proposed a Network Functions Virtualization (NFV) framework for OpenStack that can implement different network functions such as firewalls and load balancers to improve the performance and scalability of OpenStack networks. It is recommended that NFV projects use a combination of virtualization and orchestration technologies to virtualize network functions. The authors conducted various tests to evaluate the effectiveness of the NFV project and found that it improves the performance and efficiency of OpenStack networks.

Another article by Xu et al. (2020) proposed OpenStack's Network Function Chaining (NFC) framework, which can combine different network functions to increase the efficiency and flexibility of the OpenStack network. The proposed NFC protocol uses a combination of virtualization and orchestration techniques to document network transactions. The authors conducted several tests to evaluate the effectiveness of the NFC concept and found that it improves the performance and flexibility of the OpenStack network.

2.5. OpenStack and Big Data Analytics

Big data analytics is another critical area of cloud computing, and OpenStack can be used for big data analytics by providing a flexible and scalable platform for data processing and analysis. Several research papers have explored the use of OpenStack for big data analytics and proposed different strategies and techniques. For example, Zhou et al. (2020) proposed a big data analytics framework for OpenStack that can support different big data processing frameworks, such as Hadoop and Spark. The proposed framework uses a combination of virtualization and orchestration techniques to support the big data processing frameworks. The authors conducted several experiments to evaluate the performance of the proposed framework and found that it significantly improves the performance and scalability of the big data processing.

Similarly, Liu et al. (2020) proposed a big data storage framework for OpenStack that can store and manage large volumes of data for big data analytics. The proposed framework uses a combination of storage technologies, such as distributed file systems and object storage, to store and manage the data. The authors conducted several experiments to evaluate the performance of the proposed framework and found that it significantly improves the performance and scalability of the big data storage.

2.6. OpenStack and Multi-Cloud Environments

Multi-cloud environments are becoming increasingly popular in cloud computing, and OpenStack can be used in multi-cloud environments to provide a flexible and scalable platform for multi-cloud management. Several research papers have explored the use of OpenStack in multi-cloud environments and proposed different strategies and techniques. For example, He et al. (2020) proposed a multi-cloud management framework for OpenStack that can manage multiple clouds, such as AWS and Azure, from a single OpenStack dashboard. The proposed multi-cloud management framework uses a combination of cloud orchestration and API integration techniques to manage the multiple clouds. The authors conducted several experiments to evaluate the performance of the proposed multi-cloud management framework and found that it significantly improves the management efficiency of multi-cloud environments.

Similarly, Sun et al. (2020) proposed a multi-cloud data management framework for OpenStack that can manage the data in multi-cloud environments by providing a unified interface for data management. The proposed multi-cloud data management framework uses a combination of data virtualization and orchestration techniques to manage the data in multiple clouds. The authors conducted several experiments to evaluate the performance of the proposed multi-cloud data management framework and found that it significantly improves the data management efficiency of multi-cloud environments.

CHAPTER-3

SYSTEM DEVELOPMENT

The applications and services are developed as per the business need. The working architectural model of the OpenStack, which is in use, described below.

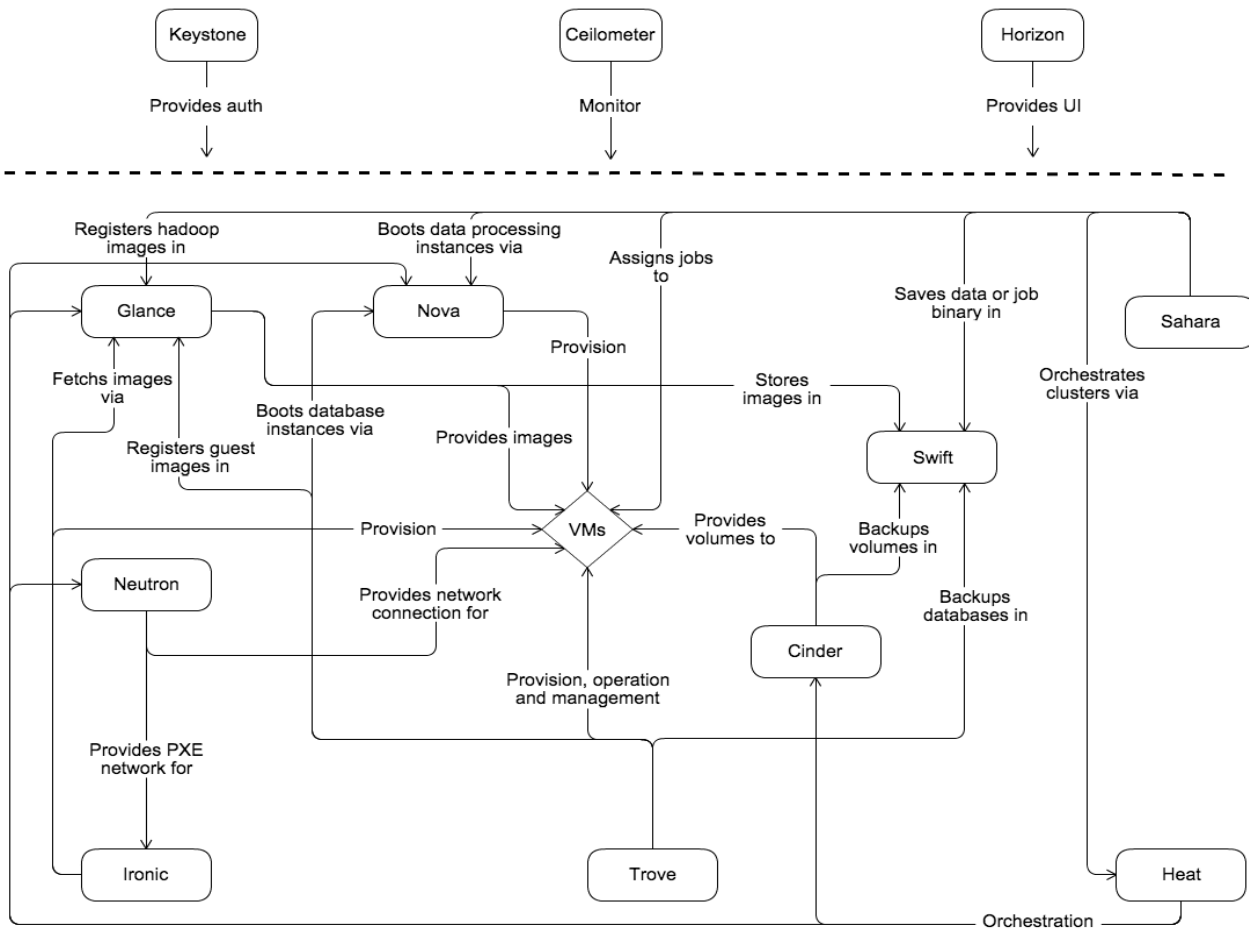


Figure 3.1: OpenStack Working Architecture

OpenStack architecture consists of several components that work together to provide a complete cloud computing platform.

The conceptual architectural diagram is depicted below, followed by the OpenStack Training SOP at the organization.

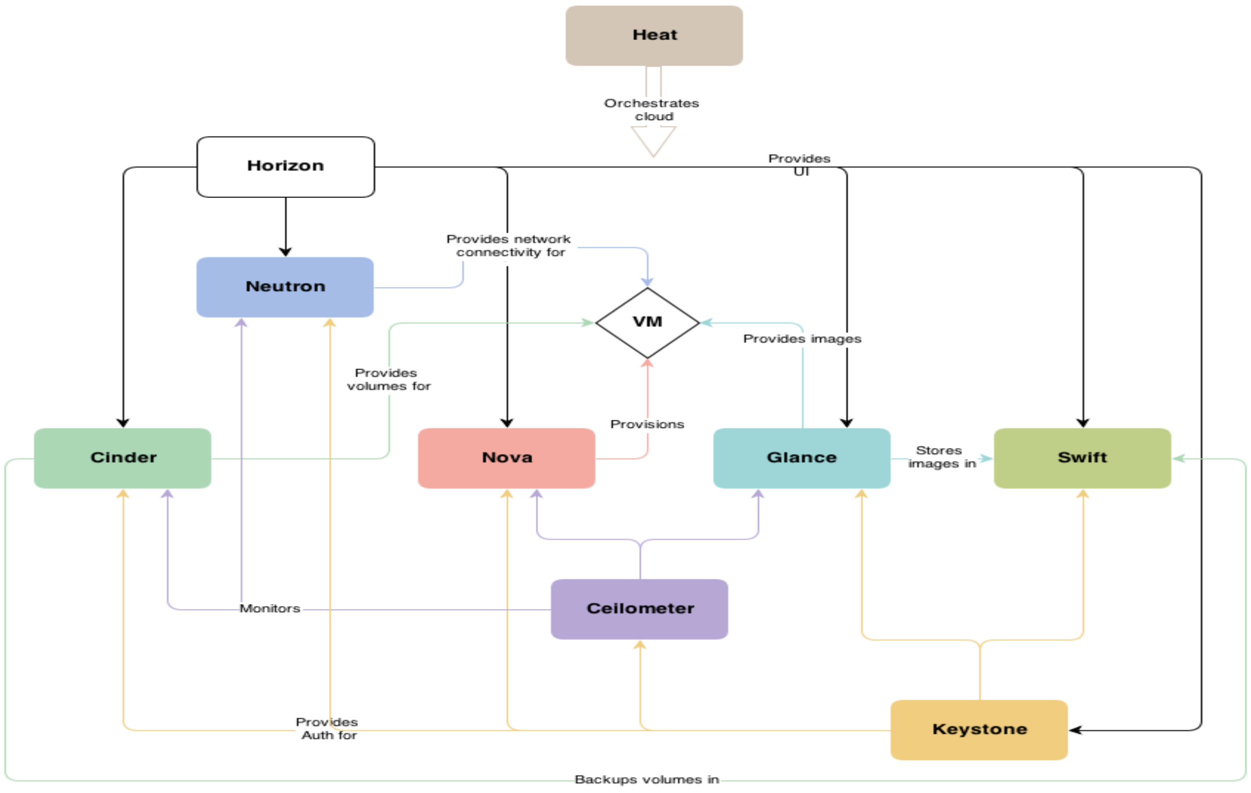
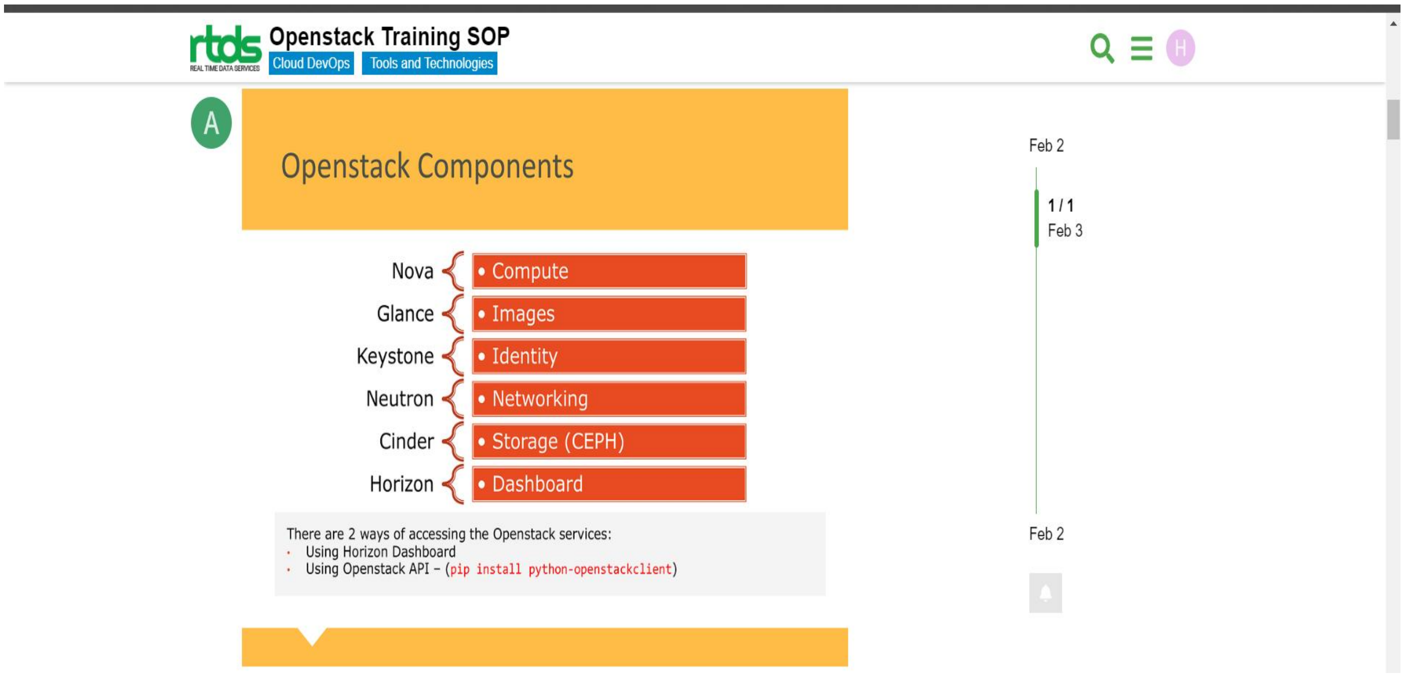


Figure 3.2: OpenStack Conceptual Diagram

At the core of OpenStack is the Compute module, also known as Nova, which provides the ability to create and manage virtual machines (VMs). It also manages networking, storage, and security for these VMs. Nova interacts with other OpenStack modules, such as Glance (image service), Cinder (block storage), and Neutron (networking service), to create a complete cloud infrastructure.

Another important module of OpenStack is Horizon, which provides a web-based dashboard for managing OpenStack resources. Horizon allows administrators and users to easily manage their virtual machines, storage, and network resources from a single interface.

In addition to these core modules, OpenStack also includes other modules such as Keystone (identity service), Heat (orchestration service), and Swift (object storage). These modules work together to provide a complete cloud infrastructure that can be used to build private, public, and hybrid clouds.



Openstack Components

Nova	• Compute
Glance	• Images
Keystone	• Identity
Neutron	• Networking
Cinder	• Storage (CEPH)
Horizon	• Dashboard

There are 2 ways of accessing the Openstack services:
- Using Horizon Dashboard
- Using Openstack API – (`pip install python-openstackclient`)

Figure 3.3: OpenStack Components List

These components are:

Nova: Nova is the compute service that manages the creation, scheduling, and management of virtual machines. It also manages the networking and storage resources needed to support these virtual machines.

Glance: Glance is the image service that manages the virtual machine images used by Nova to create new instances.

Cinder: Cinder is the block storage service that provides persistent storage for virtual machines.

Neutron: Neutron is the networking service that manages the virtual networks used by virtual machines. It also provides the necessary connectivity between virtual machines and external networks.

Horizon: Horizon is the web-based dashboard that provides a graphical user interface for managing and monitoring the OpenStack environment.

Keystone: Keystone is the identity service that provides authentication and authorization services for all other OpenStack services.

Swift: Swift is the object storage service that provides a scalable and durable storage platform for unstructured data.

3.1. Main Components of OpenStack

3.1.1. Nova Compute

Nova is the compute service in OpenStack that manages the creation, scheduling, and management of virtual machines. It also manages the networking and storage resources needed to support these virtual machines.

Nova is designed to be highly scalable, which means that it can support a large number of virtual machines. It is also designed to be highly available, which means that it can continue to operate even if some of the underlying hardware fails.

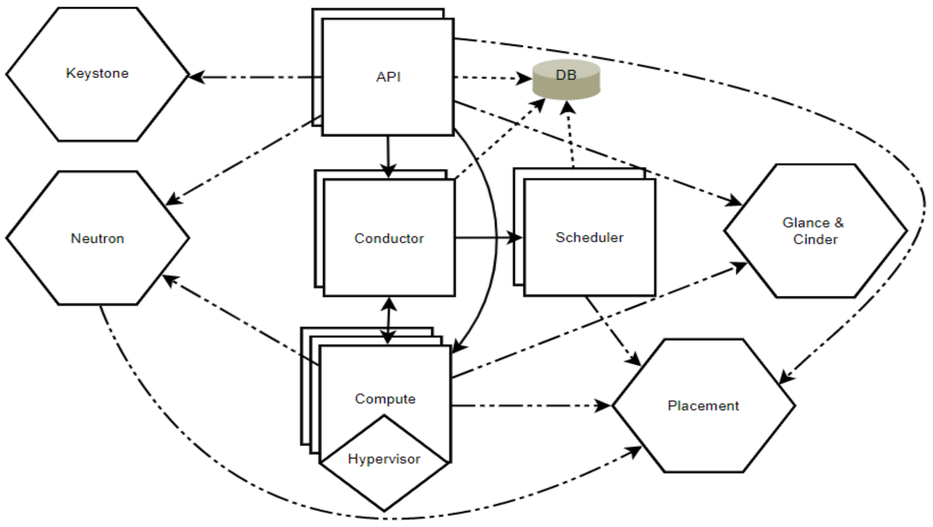


Figure 3.4: Nova Compute working Structure

Nova provides a range of features that make it easy to manage virtual machines. For example, it provides an API that allows administrators to create, start, stop, and delete virtual machines. It also provides a scheduler that can allocate virtual machines to the most appropriate physical server based on resource availability and workload.Nova also provides a range of networking and storage features. For example, it can manage virtual networks, which allow virtual machines to communicate with each other and with external networks. It can also manage block storage volumes, which provide persistent storage for virtual machines. -Nova is the OpenStack project that gives a method for provisioning register examples (otherwise known as virtual servers). Nova upholds making virtual machines, exposed metal servers (using amusing), and has restricted help for framework compartments.

3.1.2. Glance Image Service

Glance is the image service in OpenStack that manages the virtual machine images used by Nova to create new instances. It provides a central repository for virtual machine images, making it easy to manage and share these images across different OpenStack deployments.

Glance provides a range of features that make it easy to manage virtual machine images. For example, it provides an API that allows administrators to upload, download, and delete virtual machine images. It also provides a caching mechanism that can improve the performance of image retrieval.

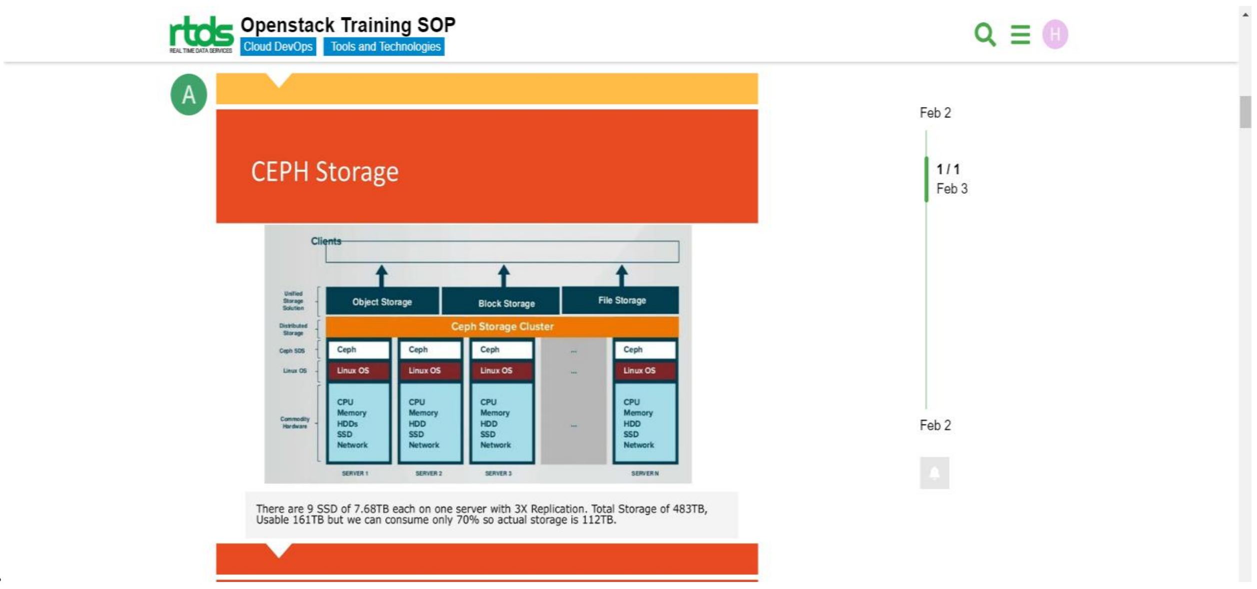
Glance supports a range of different image formats, including raw, qcow2, and VMDK. It also provides a metadata system that allows administrators to add custom metadata to images.

3.1.3. Cinder Block Storage

Cinder is the block storage service in OpenStack that provides persistent storage for virtual machines. It allows administrators to create and manage block storage volumes, which can be attached to virtual machines as additional storage.

Cinder provides a range of features that make it easy to manage block storage volumes. For example, it provides an API that allows administrators to create, delete, and resize volumes. It also provides support for snapshotting, which allows administrators to create point-in-time copies of volumes for backup and recovery purposes.

Cinder supports a range of different storage backends, including local disks, network-attached storage (NAS), and storage area networks (SANs). It also provides support for different types of storage, including magnetic disks, solid-state drives (SSDs), and hybrid storage.



There are 9 SSD of 7.68TB each on one server with 3X Replication. Total Storage of 483TB, Usable 161TB but we can consume only 70% so actual storage is 112TB.

Figure 3.5: Ceph Storage

Ceph capacity is a product characterized capacity arrangement that conveys information across bunches of capacity assets. It is a shortcoming open minded and scale-out capacity framework, where numerous Ceph stockpiling hubs (servers) participate to introduce a solitary stockpiling framework that can hold numerous petabytes (1PB = 1,000 TB = a million GB) of information.

3.1.4. Neutron Networking

Neutron is the networking service in OpenStack that manages the virtual networks used by virtual machines. It also provides the necessary connectivity between virtual machines and external networks.

Neutron provides a range of features that make it easy to manage virtual networks. For example, it provides an API that allows administrators to create, update, and delete networks, subnets, and routers. It also provides support for different types of networking, including virtual local area networks (VLANs), overlays, and software-defined networking (SDN). Neutron provides a range of different network services, including load balancing, firewalling, and virtual private networks (VPNs). It also provides support for different network plugins, including Open vSwitch (OVS), Linux Bridge, and Cisco Nexus.

3.1.5. Horizon Dashboard

Horizon is the web-based dashboard in OpenStack that provides a graphical user interface for managing and monitoring the OpenStack environment. It allows administrators to manage virtual machines, storage, and networking resources from a single web-based interface.

Horizon provides a range of features that make it easy to manage the OpenStack environment. For example, it provides a dashboard that displays the status of different OpenStack components, including Nova, Glance, Cinder, and Neutron. It also provides a range of widgets that allow administrators to manage virtual machines, storage volumes, and networks.

Horizon also provides support for different authentication and authorization methods, including LDAP, Active Directory, and OAuth.

OpenStack provides several tools for managing security groups, including the OpenStack dashboard and the command-line interface. These tools allow users to create, modify, and delete security groups, as well as create and edit the rules that define their behavior.

To create a new security group, users can access the Compute section of the OpenStack dashboard and select the "Security Groups" option. From there, they can create a new security group and add rules to control its behavior. The rules can be based on protocols, ports, IP addresses, or CIDR blocks, and can be either allow or deny rules.

Alternatively, users can create and manage security groups using the OpenStack command-line interface. The "OpenStack Security Group" command can be used to create, list, show, delete, and modify security groups and their rules.

Best Practices for Using Security Groups

When working with security groups in OpenStack, there are several best practices to follow to ensure the security of your VMs and networks. Some of these best practices include:

Keep security groups simple: The more complex a security group, the more difficult it can be to manage and troubleshoot. It's important to keep security groups as simple as possible and limit the number of rules to only those that are necessary.

Use the principle of least privilege: When creating rules for a security group, use the principle of least privilege and only allow the traffic that is required for the VM to function properly. This can help to prevent unnecessary exposure to potential security risks.

Regularly review and update security groups: Security risks and requirements can change over time, so it's important to regularly review and update security groups to ensure they are up-to-date and providing adequate protection.

Use security groups in combination with other security measures: While security groups are a fundamental aspect of network security in OpenStack, they should be used in combination with other security measures, such as encryption, authentication, and access controls, to provide comprehensive protection.

3.1.6. Keystone Identity Service

Keystone is the identity service in OpenStack that provides authentication and authorization services for all other OpenStack services. It provides a central repository for user and service information, making it easy to manage and share this information across different OpenStack deployments.

Keystone provides a range of features that make it easy to manage user and service information. For example, it provides an API that allows administrators to create, update, and delete users, groups, and roles. It also provides support for different authentication methods, including password-based authentication, token-based authentication, and multi-factor authentication. Keystone also provides support for different identity backends, including SQL databases, LDAP, and Active Directory.

3.1.7. Swift Object Storage

Swift is the object storage service in OpenStack that provides a scalable and durable storage platform for unstructured data. It allows administrators to store and retrieve large amounts of data, such as images, videos, and documents.

Swift provides a range of features that make it easy to manage object storage. For example, it provides an API that allows administrators to create, update, and delete containers and objects. It also provides support for different storage policies, including replication and erasure coding. Swift provides a highly scalable and durable storage platform. It is designed to store large amounts of data across a distributed set of nodes, making it highly fault-tolerant and resilient to hardware failures.

Swift is highly scalable and can handle a massive amount of data. It can store and retrieve objects in the order of petabytes, making it suitable for storing large datasets. It is also highly available and can tolerate failures of hardware or network components without losing data. Swift uses a distributed architecture, where objects are stored on multiple servers, and data is replicated across multiple storage nodes. This architecture ensures that data is available even in the event of hardware failures.

Swift also provides data durability. It uses erasure coding and replication to ensure that data is protected against data loss. Erasure coding is a technique that breaks data into small pieces and stores each piece on a different storage node. This technique ensures that data is available even if some storage nodes fail. Replication, on the other hand, makes multiple copies of the data and stores them on different storage nodes. This approach ensures that data is available even if some storage nodes fail.

3.2. OpenStack Architecture Layers

The OpenStack architecture is divided into four layers: the hardware layer, the infrastructure layer, the platform layer, and the application layer.

3.2.1. Hardware Layer

The hardware layer consists of the physical servers, storage devices, and networking equipment that make up the OpenStack environment. These components provide the underlying infrastructure on which the OpenStack services run.

3.2.2. Infrastructure Layer

The infrastructure layer consists of the OpenStack services that provide the core infrastructure services, including Nova, Glance, Cinder, Neutron, Keystone, and Swift. These services provide the necessary compute, storage, and networking resources to support the platform and application layers.

3.2.3. Platform Layer

The platform layer consists of the tools and frameworks that developers use to build and deploy applications on top of OpenStack. This layer includes tools like Heat and Trove, which provide infrastructure as code and database as a service capabilities, respectively.

3.2.4. Application Layer

The application layer consists of the applications and services that are deployed on top of OpenStack. This layer includes a wide range of applications, from simple web applications to complex data analytics and machine learning workloads.

3.2.5. Ecosystem Layer:

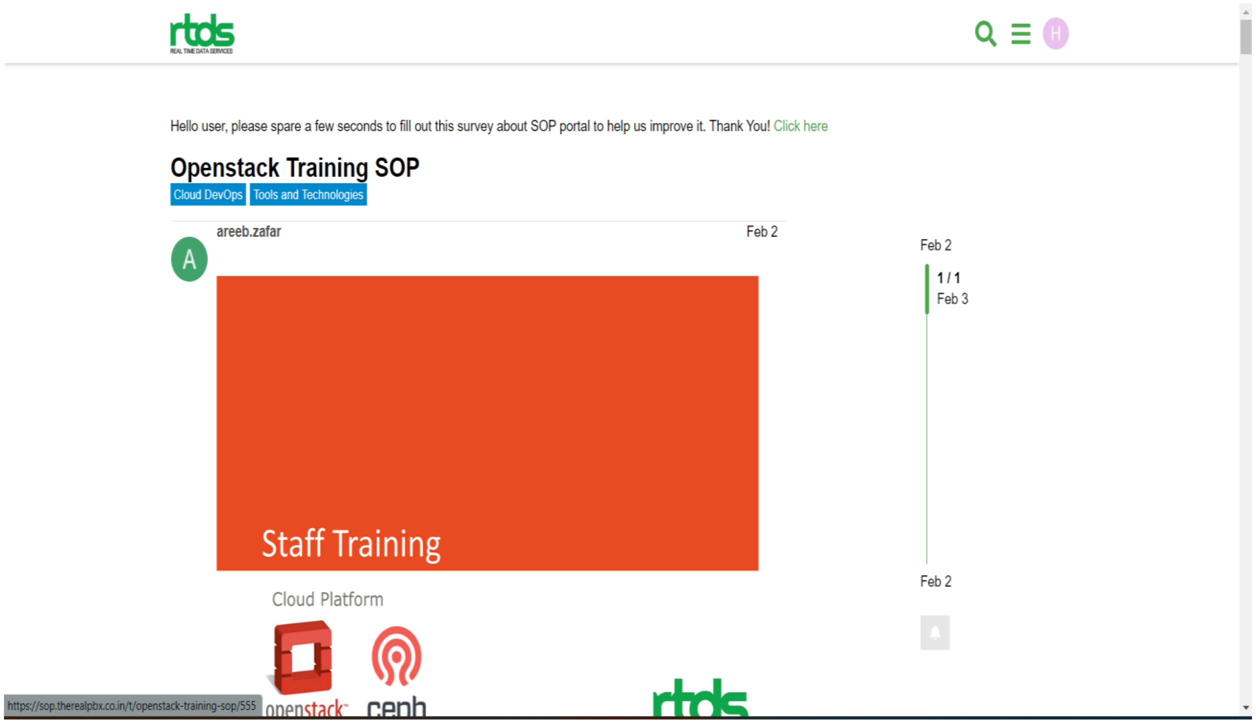
The ecosystem layer provides additional services and tools for integrating with other systems and extending the functionality of OpenStack. It includes services such as Telemetry (Ceilometer), Metering (Gnocchi), and Workflow (Mistral) which provide monitoring, metering, and workflow management respectively. These services enable the integration of OpenStack with other systems and provide additional functionality for managing and monitoring cloud services.

In summary, the OpenStack architecture is composed of four layers that work together to provide a complete cloud infrastructure. The infrastructure layer provides the physical and virtual resources necessary for building and managing cloud services, the platform services layer provides the core OpenStack services that enable the creation and management of cloud services, the application services layer provides services for running applications in the cloud, and the ecosystem layer provides additional services and tools for integrating with other systems and extending the functionality of OpenStack.

CHAPTER- 4

EXPERIMENTS & RESULTS ANALYSIS

4.1. Experimentation with different components


Hello user, please spare a few seconds to fill out this survey about SOP portal to help us improve it. Thank You! Click here

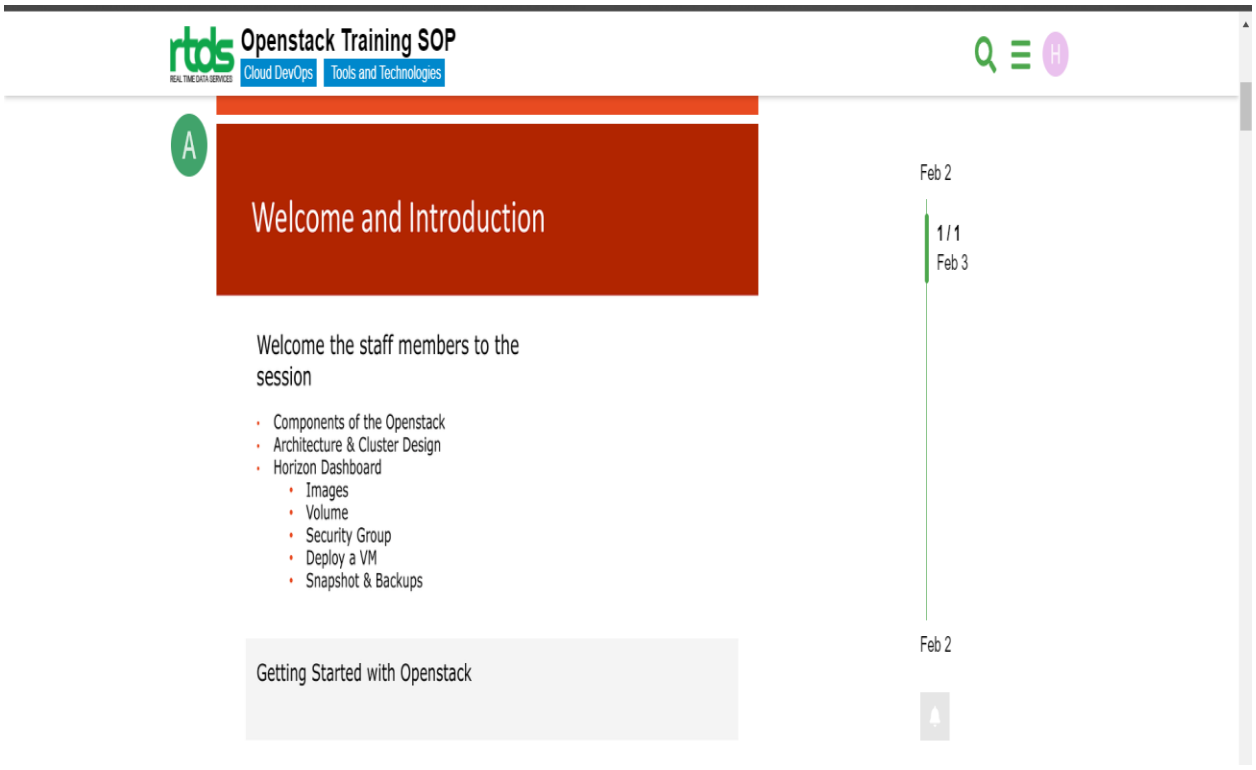
Openstack Training SOP

Cloud DevOps Tools and Technologies

areeb.zafar Feb 2

Figure 4.1: OpenStack Training SOP

Standard Operating Procedures (SOPs) for OpenStack training include guidelines for course content development, instructor selection and training, course delivery, certification process, and course evaluation. These procedures ensure that the training program is delivered consistently and effectively, with a focus on meeting the needs of learners and providing them with the necessary knowledge and skills to succeed in the field of Cloud computing.


Openstack Training SOP

Cloud DevOps | Tools and Technologies

Welcome and Introduction

Welcome the staff members to the session

- Components of the Openstack
- Architecture & Cluster Design
- Horizon Dashboard
  - Images
  - Volume
  - Security Group
  - Deploy a VM
  - Snapshot & Backups

Getting Started with Openstack

Feb 2

1 / 1

Feb 3

Feb 2

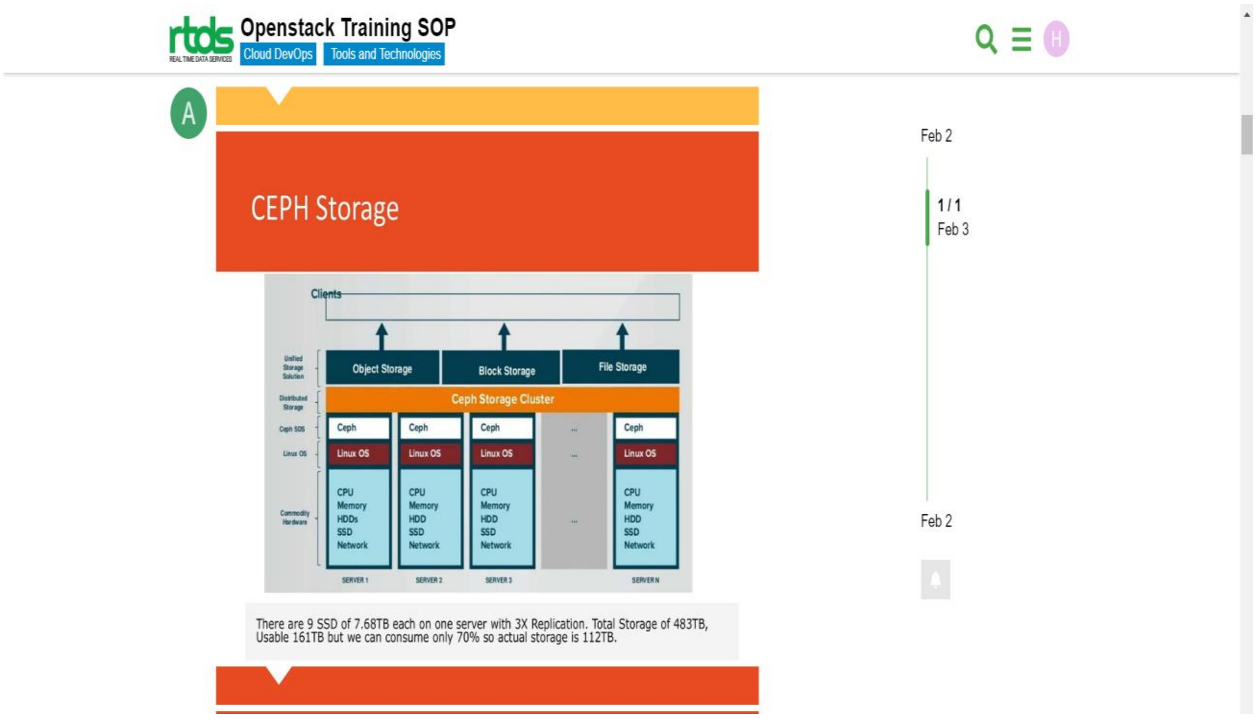
Figure 4.2: OpenStack Modules Content

Look is a picture administration that permits clients to find, recover, and register VM (virtual machine) pictures and compartment pictures, which can involve Swift or Ceph as its genuine stockpiling backend (for example not broadly useful article stockpiling).

Neutron is an OpenStack venture to give "organizing as a help" between interface gadgets (e.g., vNICs) oversaw by other OpenStack administrations (e.g., nova).

Soot is open-source programming intended to make and deal with a help that gives tireless information stockpiling to distributed computing applications. Soot is the code name for the OpenStack Block Storage project.

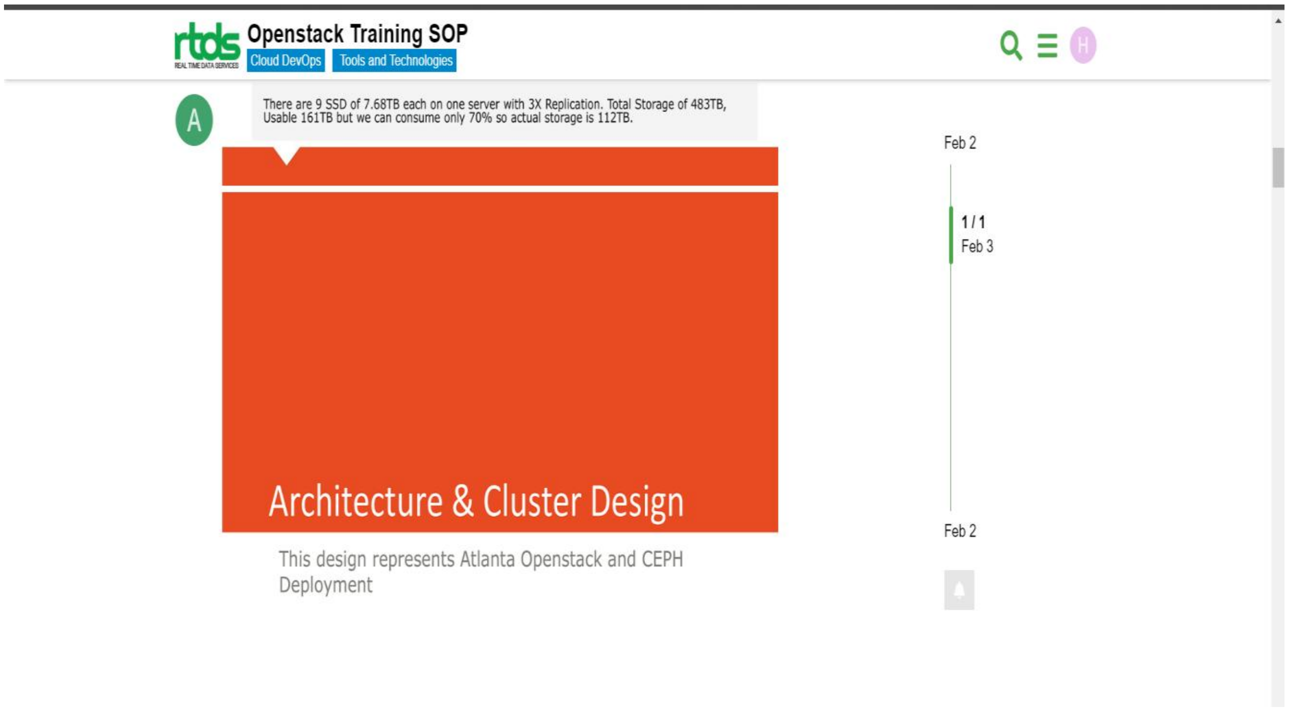
Horizon is the sanctioned execution of OpenStack's Dashboard, which gives an online UI to OpenStack administrations including Nova, Swift, Keystone, and so forth. Kindly see Introducing Horizon for an exhaustive glance at what Horizon is and what the points of the venture are.



There are 9 SSD of 7.68TB each on one server with 3X Replication. Total Storage of 483TB, Usable 161TB but we can consume only 70% so actual storage is 112TB.

Figure 4.3: Ceph Storage Structure Explanation

Ceph capacity is a product characterized capacity arrangement that conveys information across bunches of capacity assets. It is a shortcoming open minded and scale-out capacity framework, where numerous Ceph stockpiling hubs (servers) participate to introduce a solitary stockpiling framework that can hold numerous petabytes (1PB = 1,000 TB = a million GB) of information.



Openstack Training SOP

Cloud DevOps Tools and Technologies

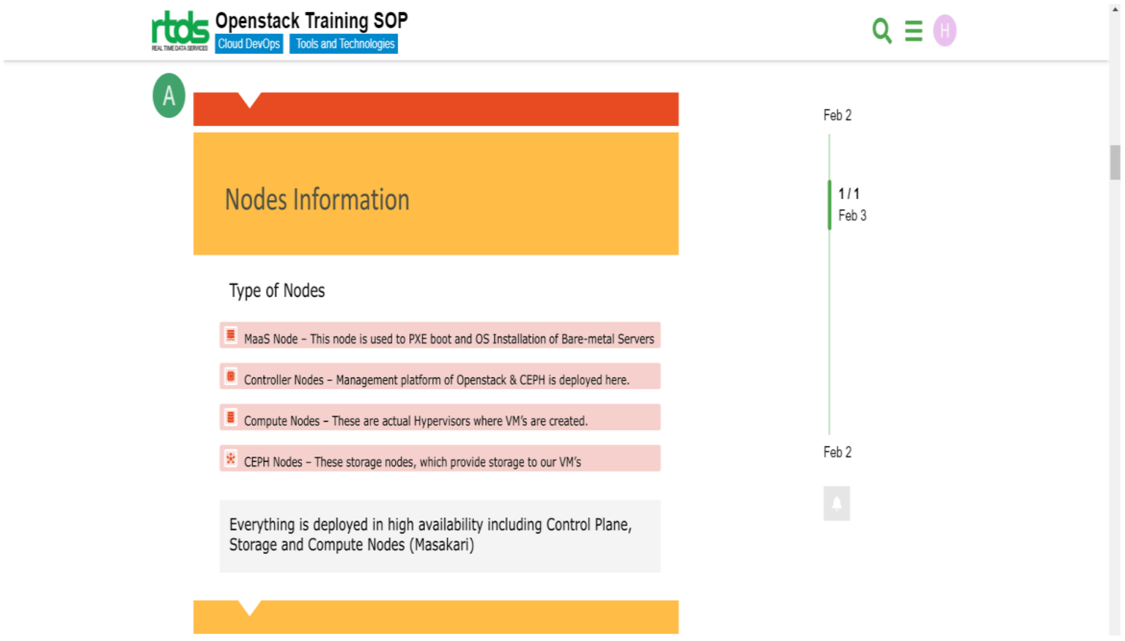
There are 9 SSD of 7.68TB each on one server with 3X Replication. Total Storage of 483TB, Usable 161TB but we can consume only 70% so actual storage is 112TB.

Architecture & Cluster Design

This design represents Atlanta Openstack and CEPH Deployment

Figure 4.4: Cluster Design Storage Specification

The Architecture Design Guide gives data on arranging and planning an OpenStack cloud. It makes sense of center ideas, cloud engineering plan prerequisites, and the plan models of key parts and administrations in an OpenStack cloud. The aide additionally depicts five normal cloud use cases.



Openstack Training SOP

Cloud DevOps | Tools and Technologies

Nodes Information

Type of Nodes

- MaaS Node – This node is used to PXE boot and OS Installation of Bare-metal Servers
- Controller Nodes – Management platform of Openstack & CEPH is deployed here.
- Compute Nodes – These are actual Hypervisors where VM's are created.
- CEPH Nodes – These storage nodes, which provide storage to our VM's

Everything is deployed in high availability including Control Plane, Storage and Compute Nodes (Masakari)

Figure 4.5: Nodes in Openstack

In OpenStack, a node is a physical or virtual machine that is used to provide compute, storage, or networking resources to an OpenStack cloud. Nodes are managed by OpenStack's compute service, which is called Nova.

There are several types of nodes in OpenStack:

Compute Nodes: These are the nodes that run the OpenStack compute service, Nova. They provide virtual machine instances to the OpenStack cloud. Compute nodes can be physical or virtual machines, and they are connected to the OpenStack cloud through a network.

Storage Nodes: These are the nodes that provide storage resources to the OpenStack cloud. They are used to store virtual machine images, block storage, and object storage data. Storage nodes can be physical or virtual machines, and they are connected to the OpenStack cloud through a network.

Network Nodes: These are the nodes that provide networking resources to the OpenStack cloud. They are responsible for routing traffic between virtual machine instances and between the

OpenStack cloud and external networks. Network nodes can be physical or virtual machines, and they are connected to the OpenStack cloud through a network.

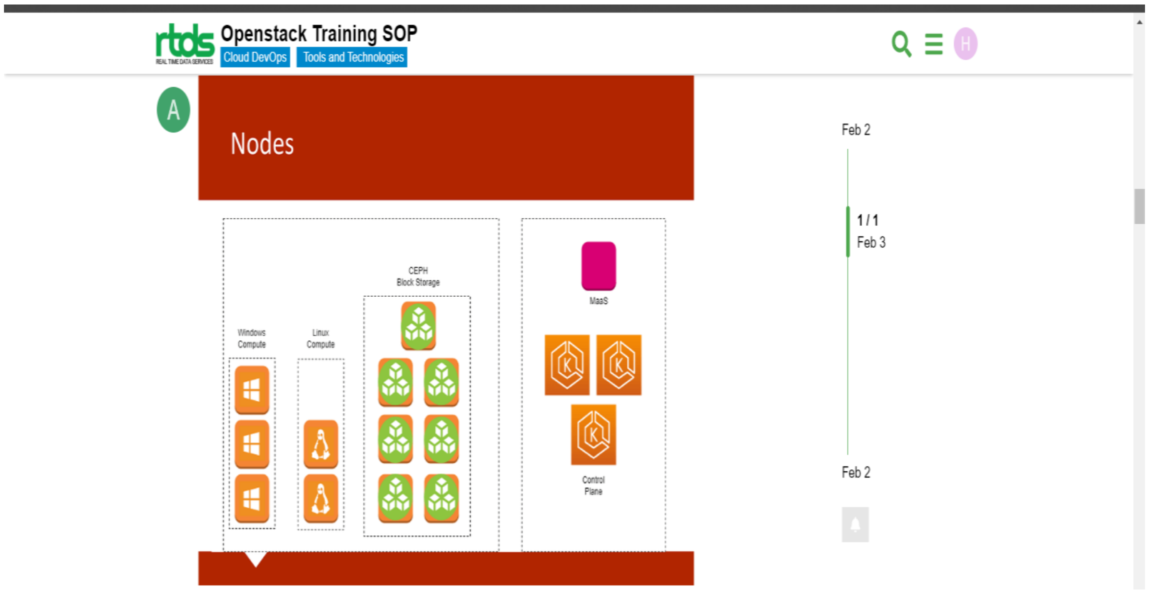


Figure 4.6: Nodes Structure

Each node in OpenStack has its own set of characteristics and properties that are used to manage it. Some of the key pieces of information that are associated with nodes in OpenStack include:

Hostname: This is the name of the node as it is known within the OpenStack cloud.

IP address: This is the IP address of the node on the network.

Status: This indicates whether the node is up or down.

Type: This indicates whether the node is a compute, storage, or network node.

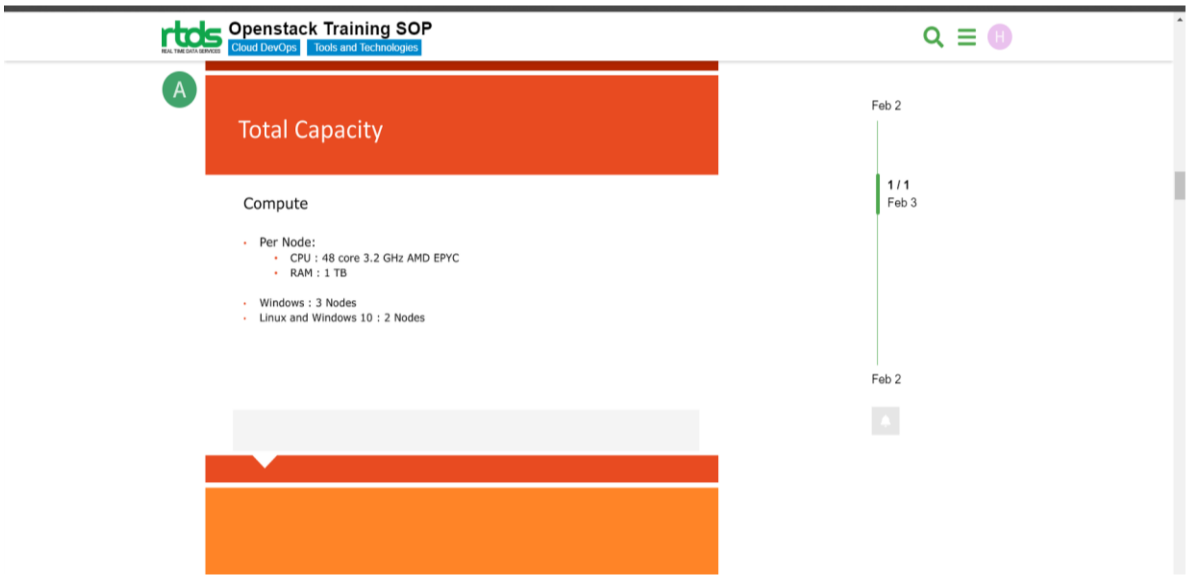
Capacity: This is the number of resources (CPU, memory, and disk) that are available on the node.

Hypervisor: This is the virtualization technology that is used on the node.

Projects: This is the set of OpenStack projects that are associated with the node.

By managing the information associated with nodes in OpenStack, administrators can effectively provision, monitor, and troubleshoot their OpenStack cloud.

A regulator hub is a framework running on Oracle Linux and is where the majority of the OpenStack administrations are introduced. The term regulator hub is utilized to examine hubs that don't run virtual machine examples. The regulator hubs might have all the non-figure administrations or just some of them. A regulator hub may likewise incorporate the Oracle OpenStack for Oracle Linux tool compartment, which is utilized to play out the sending of OpenStack administrations to different hubs.

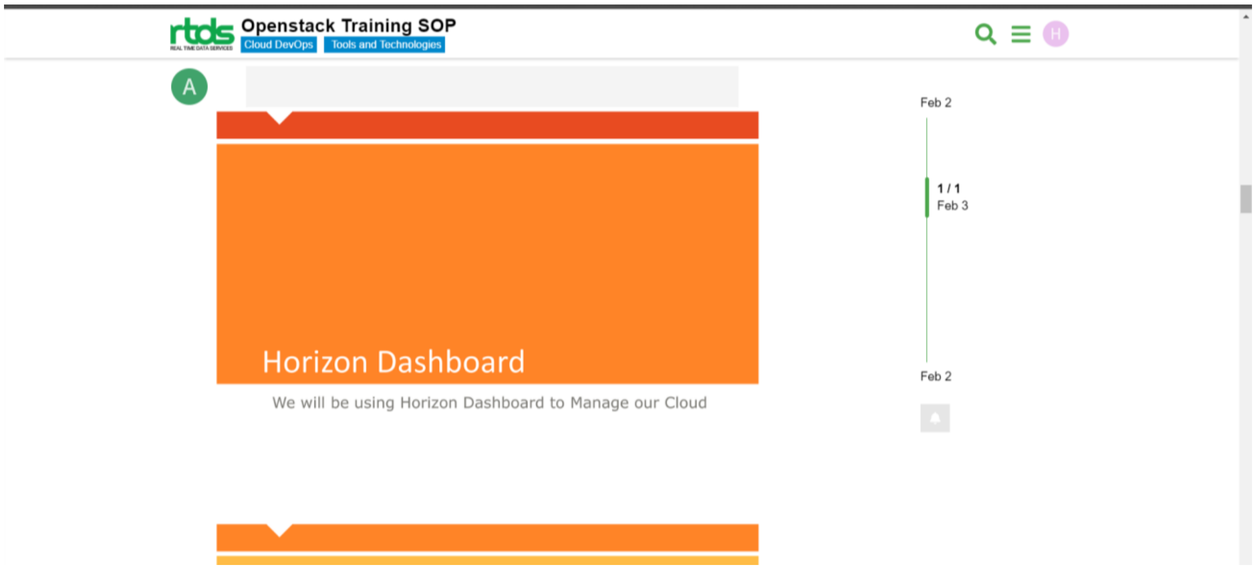


Total Capacity

Compute

- Per Node:
  - CPU : 48 core 3.2 GHz AMD EPYC
  - RAM : 1 TB
- Windows : 3 Nodes
- Linux and Windows 10 : 2 Nodes

Figure 4.7: Node Capacity

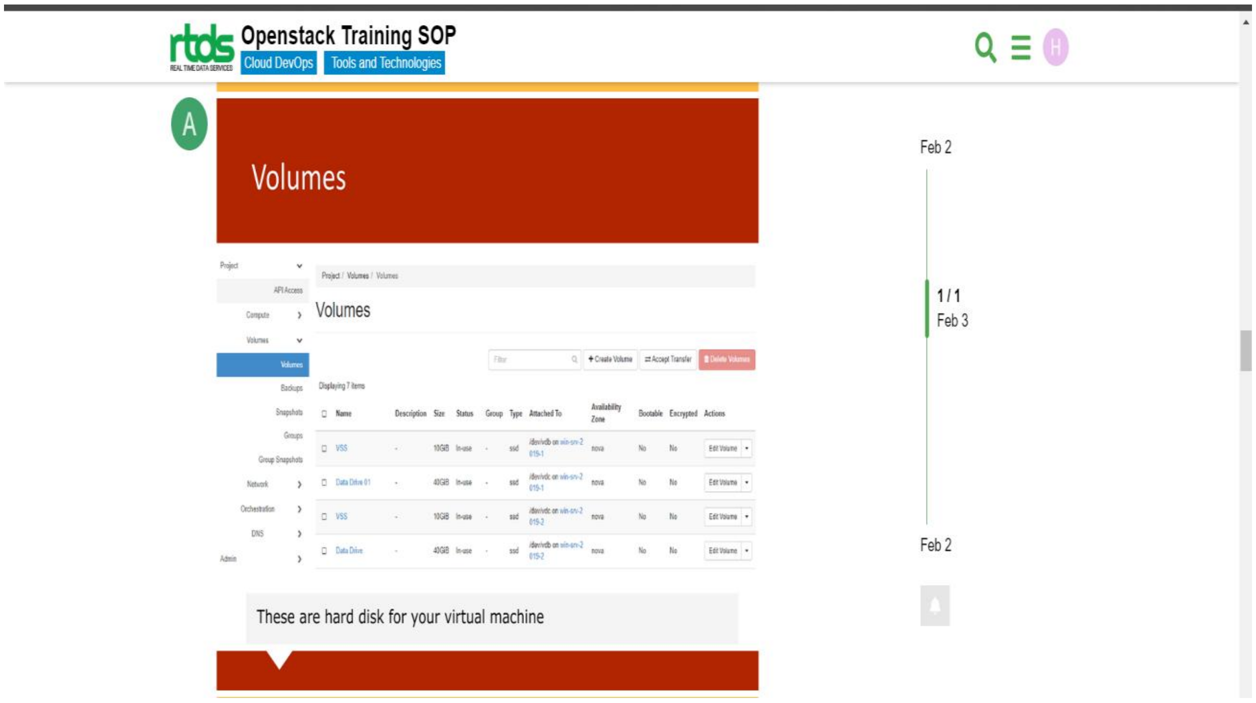


Horizon Dashboard

We will be using Horizon Dashboard to Manage our Cloud

Figure 4.8: Horizon Dashboard

Horizon is the standard execution of OpenStack's Dashboard, which gives an online UI to OpenStack administrations including Nova, Swift, Keystone, and so on.



Volumes

Project / Volumes / Volumes

Volumes

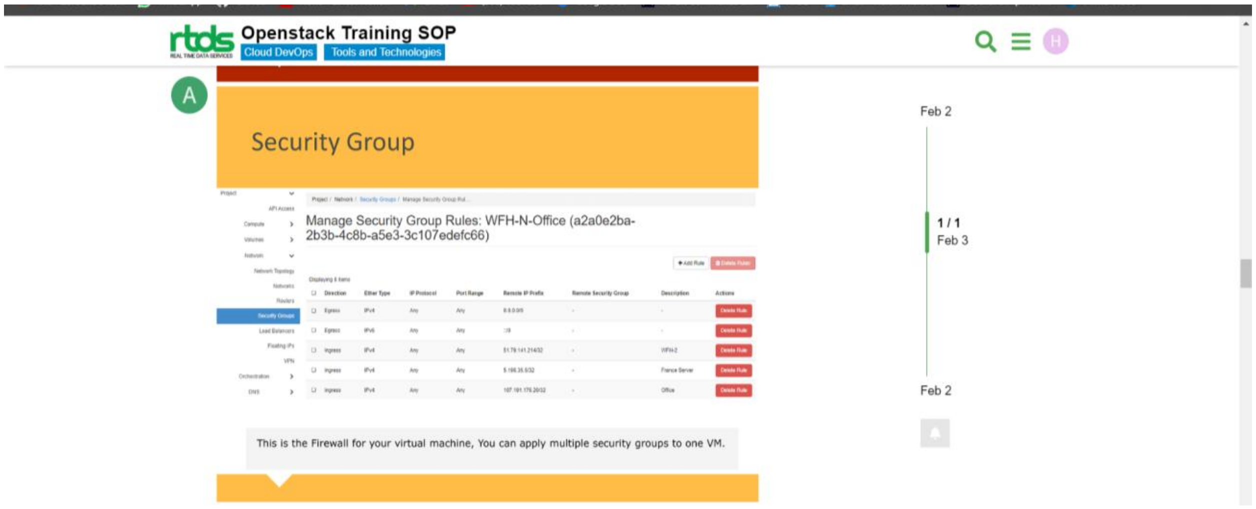
Displaying 7 items

☐	Name	Description	Size	Status	Group	Type	Attached To	Availability Zone	Bootable	Encrypted	Actions
☐	VSS	-	10GiB	In-use	-	ssd	/dev/vdb on win-srv-2019-1	nova	No	No	Edit Volume
☐	Data Drive 01	-	40GiB	In-use	-	ssd	/dev/vdc on win-srv-2019-1	nova	No	No	Edit Volume
☐	VSS	-	10GiB	In-use	-	ssd	/dev/vdc on win-srv-2019-2	nova	No	No	Edit Volume
☐	Data Drive	-	40GiB	In-use	-	ssd	/dev/vdb on win-srv-2019-2	nova	No	No	Edit Volume

These are hard disk for your virtual machine

Figure 4.9: Volumes in OpenStack

Utilize the OpenStack client orders to make and oversee volumes. This model makes a my-new volume in view of a picture. List pictures, and note the ID of your desired picture to use for your volume



Manage Security Group Rules: WFH-N-Office (a2a0e2ba-2b3b-4c8b-a5e3-3c107edefc66)

Direction	Ether Type	IP Protocol	Port Range	Remote IP Prefix	Remote Security Group	Description	Actions
Egress	IPv4	Any	Any	0.0.0.0/0	-	-	Delete Rule
Egress	IPv6	Any	Any	::/0	-	-	Delete Rule
Ingress	IPv4	Any	Any	51.79.141.214/32	-	WFH-2	Delete Rule
Ingress	IPv4	Any	Any	5.196.35.5/32	-	France Server	Delete Rule
Ingress	IPv4	Any	Any	107.191.175.20/32	-	Office	Delete Rule

This is the Firewall for your virtual machine, You can apply multiple security groups to one VM.

Figure 4.10: Security Group Configuration

OpenStack security bunches are IP channels allocated to virtual ports. A security bunch contains a named rundown of rules, which characterizes the traffic that is allowed to pass. It adjusts a default deny strategy, so traffic that don't match any of the principles is dropped.

Security groups operate at the network level, applying rules to traffic based on the source and destination IP address and port number. These rules define the types of traffic that are allowed to pass through the security group and those that are blocked. Each security group can contain one or more rules, and each rule can be either an inbound or outbound rule.

Inbound rules are used to control traffic that is directed towards the VM, while outbound rules are used to control traffic that is leaving the VM. For example, an inbound rule could be set to allow incoming SSH traffic to a VM from a specific IP address or range, while an outbound rule could be used to deny all outgoing traffic from the VM to a specific destination.

Security groups are applied at the instance level, meaning that each VM can be associated with one or more security groups. When a VM is launched, it is automatically associated with the default security group, which is preconfigured to allow incoming traffic from other instances that are members of the same security group. Additional security groups can be created and associated with the VM to provide additional network security.

OpenStack provides several tools for managing security groups, including the OpenStack dashboard and the command-line interface. These tools allow users to create, modify, and delete security groups, as well as create and edit the rules that define their behavior.

To create a new security group, users can access the Compute section of the OpenStack dashboard and select the "Security Groups" option. From there, they can create a new security group and add rules to control its behavior. The rules can be based on protocols, ports, IP addresses, or CIDR blocks, and can be either allow or deny rules.

Alternatively, users can create and manage security groups using the OpenStack command-line interface. The "OpenStack Security Group" command can be used to create, list, show, delete, and modify security groups and their rules.

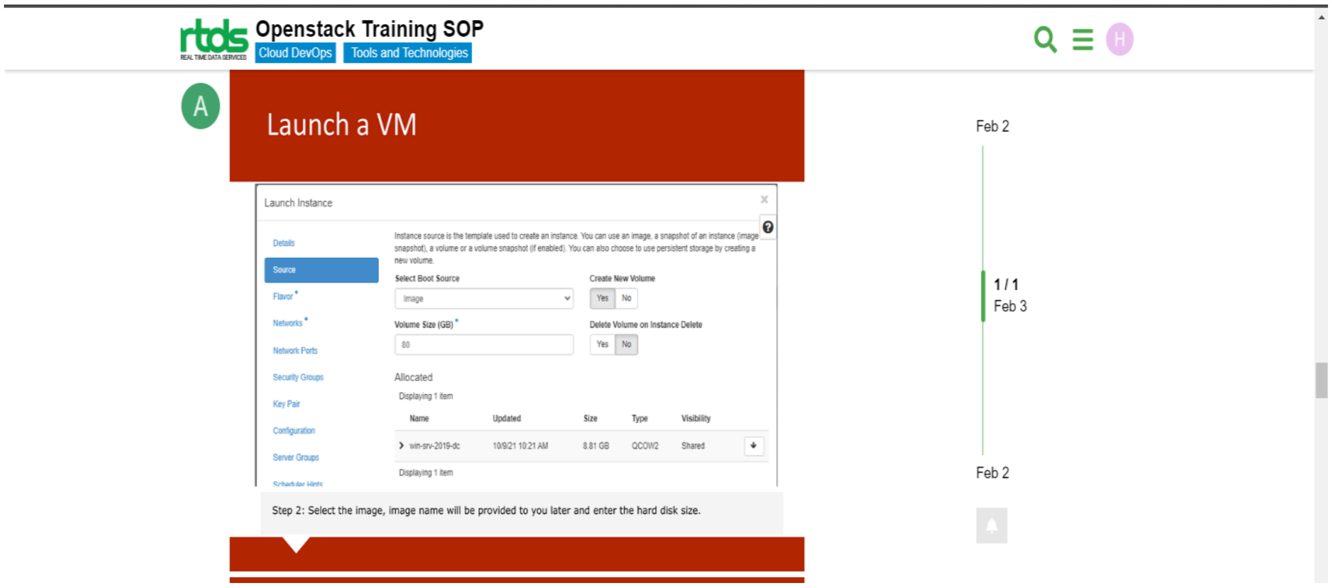
When working with security groups in OpenStack, there are several best practices to follow to ensure the security of your VMs and networks. Some of these best practices include:

The more complex a security group, the more difficult it can be to manage and troubleshoot. It's important to keep security groups as simple as possible and limit the number of rules to only those that are necessary.

Use the principle of least privilege: When creating rules for a security group, use the principle of least privilege and only allow the traffic that is required for the VM to function properly. This can help to prevent unnecessary exposure to potential security risks.

Regularly review and update security groups: Security risks and requirements can change over time, so it's important to regularly review and update security groups to ensure they are up-to-date and provide adequate protection.

Use security groups in combination with other security measures: While security groups are a fundamental aspect of network security in OpenStack, they should be used in combination with other security measures, such as encryption, authentication, and access controls, to provide comprehensive protection.



Instance source is the template used to create an instance. You can use an image, a snapshot of an instance (image snapshot), a volume or a volume snapshot (if enabled). You can also choose to use persistent storage by creating a new volume.

Select Boot Source: Image

Create New Volume: [✓ Yes] No

Volume Size (GB) *: 80

Delete Volume on Instance Delete: Yes [✓ No]

Allocated

Displaying 1 item

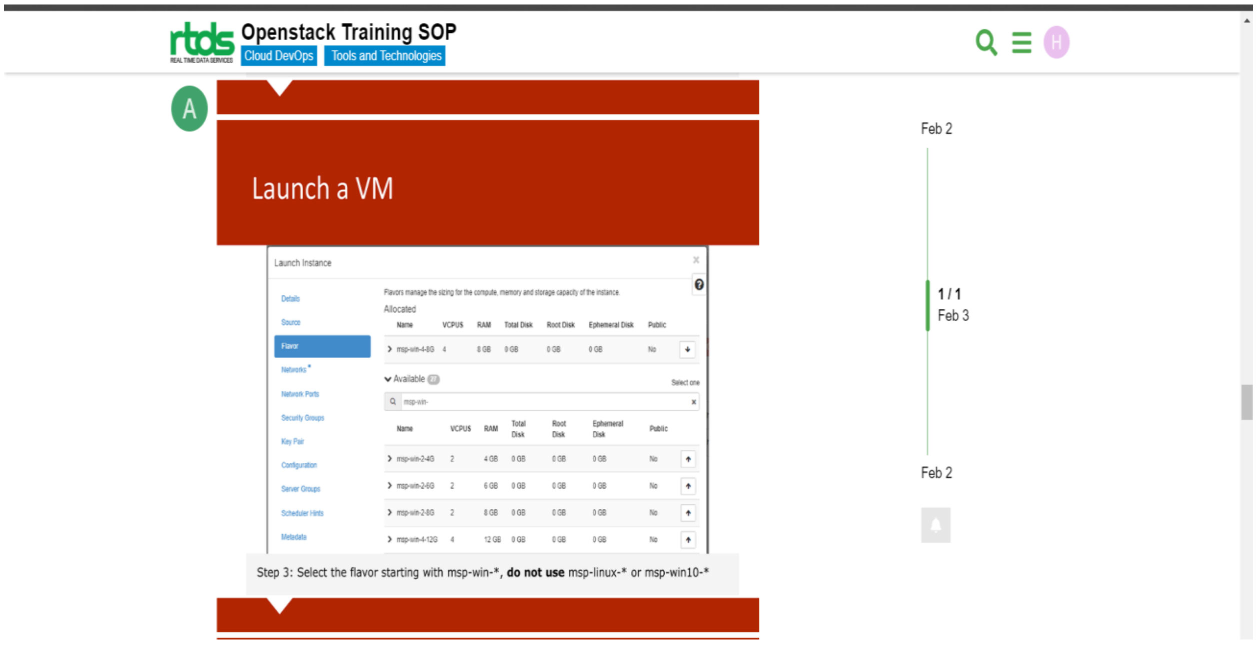
Name	Updated	Size	Type	Visibility
win-srv-2019-dc	10/9/21 10:21 AM	8.81 GB	QCOW2	Shared

Displaying 1 item

Step 2: Select the image, image name will be provided to you later and enter the hard disk size.

Figure 4.11: Configuring OS image

One of the "Four Opens" which outline the reason for an OpenStack project is Open Source. OpenStack projects don't deliver "open center" programming, they rather produce simply open-source programming. Furthermore, the product is delivered with a local area and patron acknowledged permit.



Launch a VM

Launch Instance

Flavors manage the sizing for the compute, memory and storage capacity of the instance.

Allocated

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
msp-win-4-8G	4	8 GB	0 GB	0 GB	0 GB	No

Available 27 — Select one

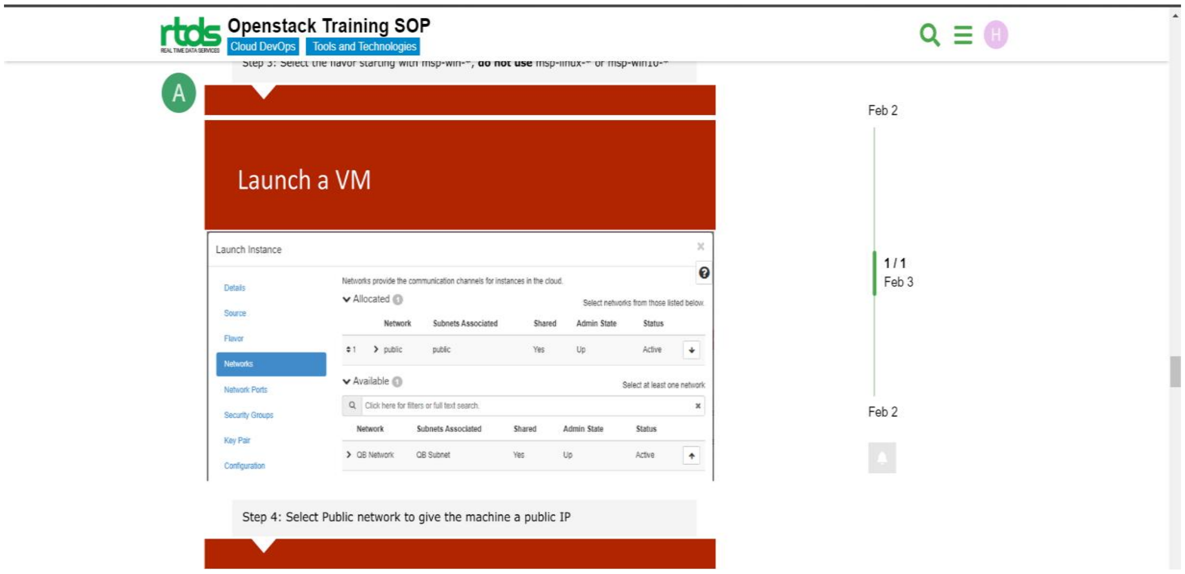
msp-win-

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public
msp-win-2-4G	2	4 GB	0 GB	0 GB	0 GB	No
msp-win-2-6G	2	6 GB	0 GB	0 GB	0 GB	No
msp-win-2-8G	2	8 GB	0 GB	0 GB	0 GB	No
msp-win-4-12G	4	12 GB	0 GB	0 GB	0 GB	No

Step 3: Select the flavor starting with msp-win-*, **do not use** msp-linux-* or msp-win10-*

Figure 4.12: Flavor Selection

In OpenStack, flavors characterize the register, memory, and capacity limit of nova figuring occasions. To lay it out plainly, a flavor is an accessible equipment setup for a server. It characterizes the size of a virtual server that can be sent off.


Openstack Training SOP

Launch a VM

Launch Instance

Networks provide the communication channels for instances in the cloud.

Allocated — Select networks from those listed below.

	Network	Subnets Associated	Shared	Admin State	Status
1	public	public	Yes	Up	Active

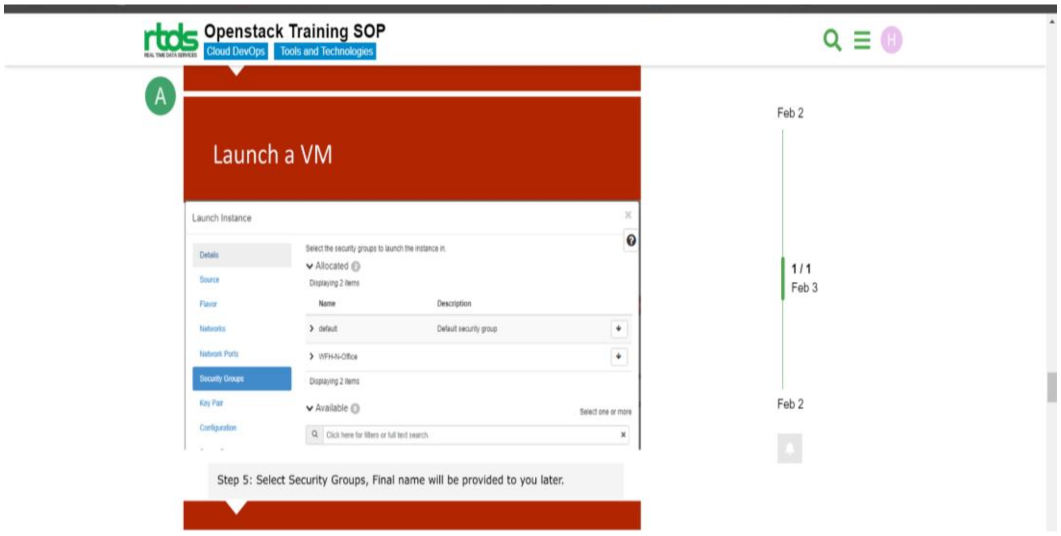
Available — Select at least one network

Network	Subnets Associated	Shared	Admin State	Status
QB Network	QB Subnet	Yes	Up	Active

Step 4: Select Public network to give the machine a public IP

Figure 4.13: Configuring Networks for VM launch.

To launch a Virtual Machine (VM) in OpenStack, users have the option to use either the OpenStack dashboard or the command-line interface. When using the dashboard, the user selects the project and navigates to the Compute section where they can click on the "Launch Instance" button to start the VM creation process. On the other hand, when using the command-line interface, the user can use the "OpenStack server create" command to create the VM.


Openstack Training SOP

Launch a VM

Launch Instance

Select the security groups to launch the instance in.

Allocated — Displaying 2 items

Name	Description
default	Default security group
WFH-N-Office	

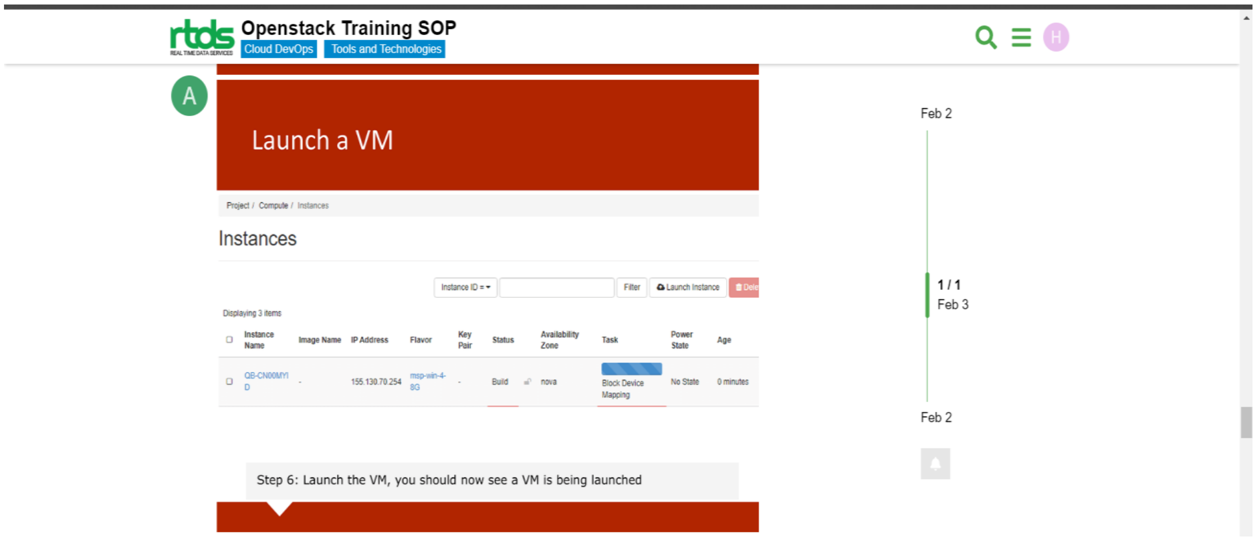
Displaying 2 items

Available — Select one or more

Step 5: Select Security Groups, Final name will be provided to you later.

Figure 4.14: Configuring Security Group

After selecting the desired method of launching the VM, the user needs to select the appropriate image for the VM. This image can either be a pre-configured image provided by OpenStack or a custom image created by the user. Additionally, the user needs to select the appropriate flavor, which determines the number of resources, such as CPU, RAM, and storage, allocated to the VM.



Instances

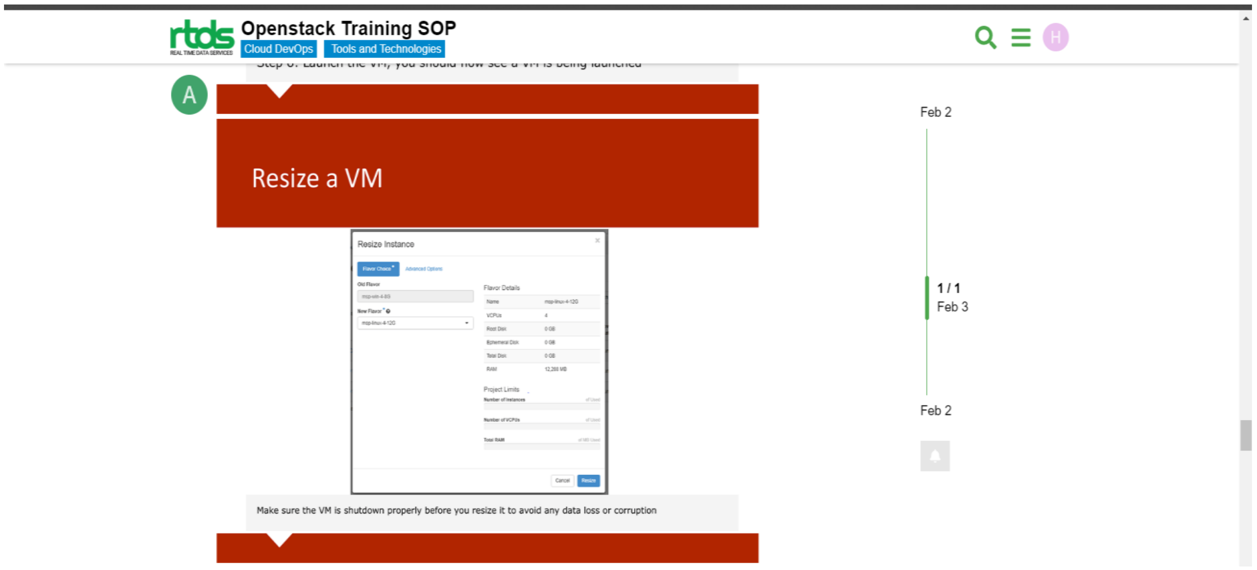
Displaying 3 items

Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Age
QB-CN00MYID	-	155.130.70.254	msp-win-4-8G	-	Build	nova	Block Device Mapping	No State	0 minutes

Step 6: Launch the VM, you should now see a VM is being launched

Figure 4.15: Launch the instance

Once the image and flavor are selected, the user can proceed with configuring other settings, such as networking, security, and user data. Finally, the user can launch the VM, which will spin up the VM and make it available for use.



Resize a VM

Resize Instance

Flavor Choice * | Advanced Options

Old Flavor: msp-win-4-8G

New Flavor *: msp-linux-4-12G

Flavor Details

Name	msp-linux-4-12G
VCPUs	4
Root Disk	0 GB
Ephemeral Disk	0 GB
Total Disk	0 GB
RAM	12,288 MB

Project Limits

Number of Instances — of Used

Number of VCPUs — of Used

Total RAM — of MB Used

Cancel | Resize

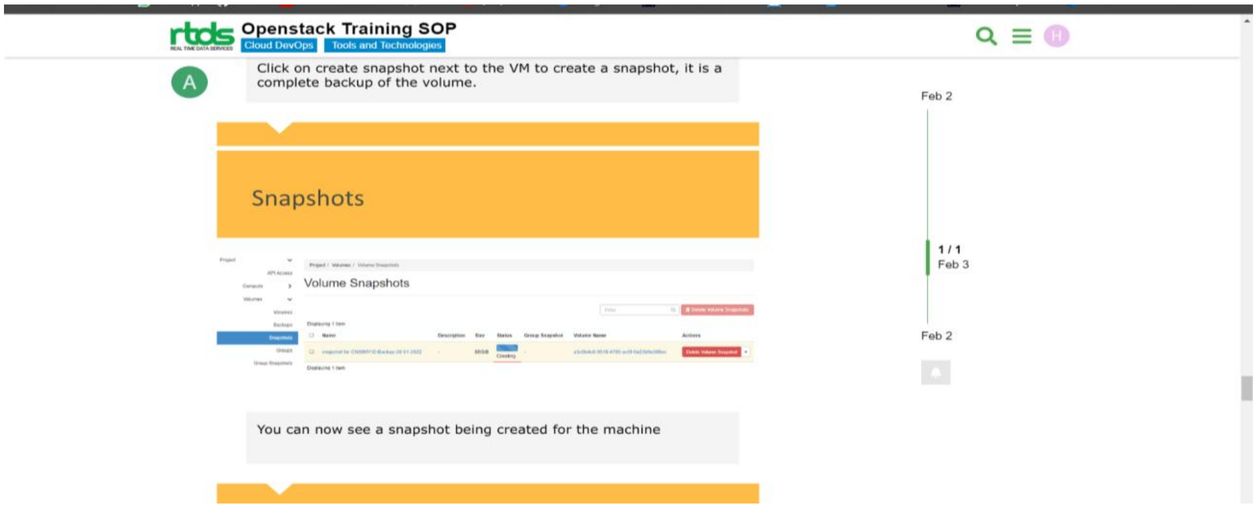
Make sure the VM is shutdown properly before you resize it to avoid any data loss or corruption

Figure 4.16: Resizing the VM

To resize a Virtual Machine (VM) in OpenStack, a user can use either the OpenStack dashboard or the command-line interface. To begin the process, the user would first select the project and navigate to the Compute section. From there, they would locate the VM that they want to resize and shut it down.

Once the VM is shut down, the user can then choose the new flavor that they want to resize the VM to. The new flavor can have more or less resources, such as CPU, RAM, and storage, compared to the original flavor.

Next, the user would select the "Resize" option for the VM and select the new flavor. OpenStack will then attempt to resize the VM to the new flavor. If the resize is successful, the user can start the VM and use it with the new resources.It's important to note that the resize process can take some time and may require some downtime for the VM. Additionally, not all VMs can be resized, and some images may have restrictions on resizing. It's recommended to verify the image's documentation to ensure that resizing is possible before attempting the process.



Click on create snapshot next to the VM to create a snapshot, it is a complete backup of the volume.

Project / Volumes / Volume Snapshots

Volume Snapshots

Displaying 1 item

Name	Description	Size	Status	Group Snapshot	Volume Name	Actions
snapshot for CNSMY-D-Backup-28-01-2022	-	80GiB	Creating	-	[illegible]	Delete Volume Snapshot

Displaying 1 item

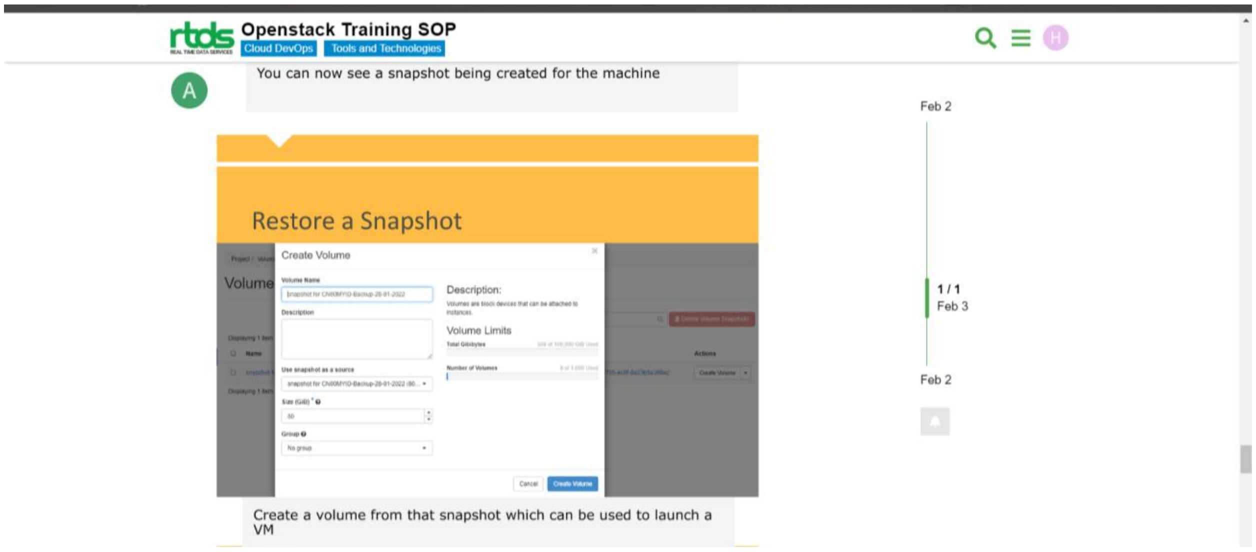
You can now see a snapshot being created for the machine

Figure 4.17: Snapshots of instances

In OpenStack, users can take a snapshot of their Virtual Machine (VM) to save its current state. A snapshot is essentially a point-in-time copy of the VM's disk image, including all of its contents, settings, and configurations. This can be useful for creating backups, rolling back changes, or replicating VMs across multiple instances.

To take a snapshot of a VM in OpenStack, users can access the OpenStack dashboard or use the command-line interface. Once in the Compute section, the user would select the VM that they want to snapshot and choose the "Create Snapshot" option. They would then enter a name and description for the snapshot and select the appropriate options for creating the snapshot, such as whether to include memory and whether to perform the operation while the VM is running or powered off.

After the snapshot is created, the user can use it to create a new VM or restore the original VM to its state at the time the snapshot was taken. Snapshots can also be used for cloning VMs or migrating them between different OpenStack instances. It's important to note that snapshots can take up a significant amount of storage space, and users should monitor their storage usage to avoid exceeding their quota. Additionally, snapshots are not a substitute for regular backups, and it's recommended to create backups of important VMs to ensure their data is properly protected.



You can now see a snapshot being created for the machine

Restore a Snapshot

Create Volume

Volume Name: snapshot for CN00MYID-Backup-28-01-2022

Description:

Use snapshot as a source: snapshot for CN00MYID-Backup-28-01-2022 (80...

Size (GiB) *: 80

Group: No group

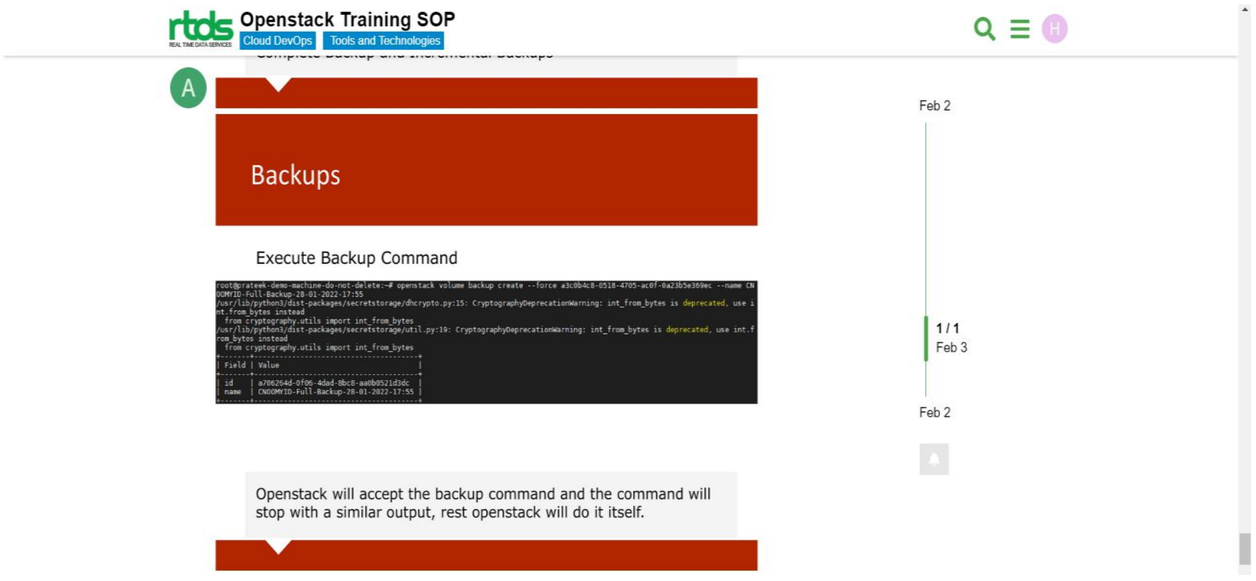
Description:
Volumes are block devices that can be attached to instances.

Volume Limits

Create a volume from that snapshot which can be used to launch a VM

Figure 4.18: Resizing the VM

It's important to note that restoring a VM from a snapshot is not a guaranteed process and may not always result in the exact same state as when the snapshot was taken. Factors such as changes to the underlying infrastructure or software on the VM can affect the restoration process. Therefore, it's recommended to test and validate the restored instance before using it in a production environment.



Backups

Execute Backup Command

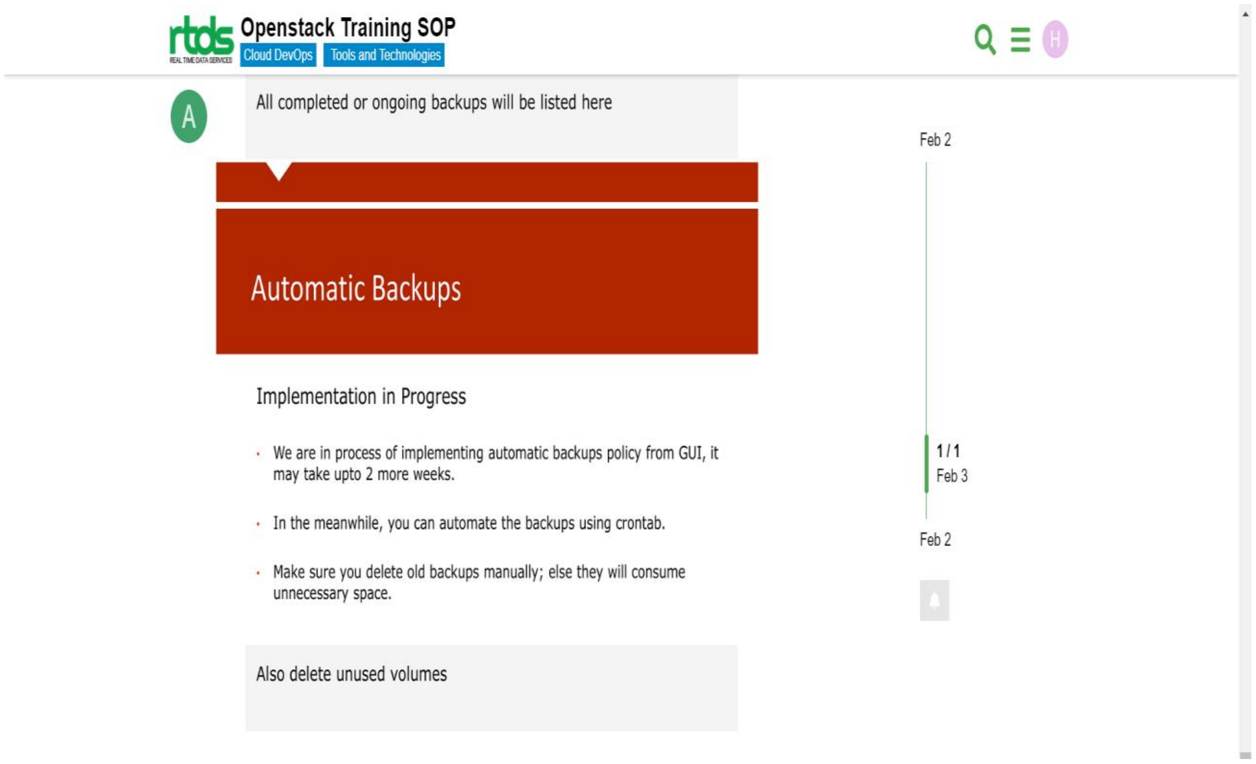
```
root@prateek-demo-machine-do-not-delete:~# openstack volume backup create --force a3c0b4c8-0518-4705-ac0f-0a23b5e369ec --name CN
OOMYID-Full-Backup-28-01-2022-17:55
/usr/lib/python3/dist-packages/secretstorage/dhcrypto.py:15: CryptographyDeprecationWarning: int_from_bytes is deprecated, use i
nt.from_bytes instead
  from cryptography.utils import int_from_bytes
/usr/lib/python3/dist-packages/secretstorage/util.py:19: CryptographyDeprecationWarning: int_from_bytes is deprecated, use int.f
rom_bytes instead
  from cryptography.utils import int_from_bytes
+-------+-------------------------------------+
| Field | Value                               |
+-------+-------------------------------------+
| id    | a706264d-0f06-4dad-8bc8-aa0b0521d3dc |
| name  | CNOOMYID-Full-Backup-28-01-2022-17:55 |
+-------+-------------------------------------+
```

Openstack will accept the backup command and the command will stop with a similar output, rest openstack will do it itself.

Figure 4.19: Backups in OpenStack

Once a backup is created, you can use it to restore the VM in case of a disaster or data loss. To restore a backup, you can create a new instance from the backup or use the backup to restore specific files or directories.

It's important to regularly test and validate backups to ensure that they can be used in case of a disaster. Additionally, it's recommended to store backups in a separate location or offsite to ensure that they are not affected by any issues that may affect the primary data center.



All completed or ongoing backups will be listed here

Automatic Backups

Implementation in Progress

- We are in process of implementing automatic backups policy from GUI, it may take upto 2 more weeks.
- In the meanwhile, you can automate the backups using crontab.
- Make sure you delete old backups manually; else they will consume unnecessary space.

Also delete unused volumes

Figure 4.20: Automatic Backup

The figure part of OpenStack, nova, has different choices to make pictures (previews) of an occurrence. One is with the nova picture make order. This makes a moment preview and transfers that to Glance, the picture stockpiling part.

The subsequent choice is nova reinforcement. This is practically equivalent to picture make, with the expansion of revolution of the reinforcement depictions. There are an additional two boundaries, backup type and revolution. The documentation is a piece inadequate on this, however the programming interface and the source code give more detail. The reinforcements are typical look pictures. In view of the backup type a set number of pictures is saved. When there are more pictures of that sort then determined in pivot, the most established is erased. On the off chance that we make a reinforcement consistently with backup type set to everyday and turn set to 7, on the 8'th day, the most established picture will be consequently eliminated (on production of the new reinforcement)

4.2. Results and Analysis

The outcomes of experiments are provided in this section. The study includes:
I. A thorough analysis of the CPU and memory resources' utility and consumption (both under load and without),

II. A performance measurement of task execution based on a chosen OpenStack set of virtualized resources, and

III. An assessment of the efficacy of the used technologies.

Five groups of online resources called flavors were taken into consideration for this study. These were the sets that were displayed in Table 4 as tiny, small, medium, large, and very large. The default environment setup permits the creation of up to 10 instances and assigns 20 VCPUs, 50 gigabytes of RAM, and 1 TB of storage, among additional resources.

Table 4.1: Flavor Specifications

Flavors	VCPUs	Disks [GB]	RAM[MB]
m1.tiny	1	1	512
m1.small	1	20	2048
m1.medium	2	40	4096
m1.large	4	80	8192
m1.xlarge	8	160	16384

OpenStack by default offers two virtualized operating systems images [16]:
Focal Fossa (244.4 MB) is an Ubuntu 20.04 server edition, while CirrOS (9.3 MB) is a tiny Linux system made to be used as a test image on clouds.

The following resources are required to comply with OpenStack's minimum specifications for a proof-of-concept setup that runs several CirrOS instances [16]:

• One CPU, two gigabytes of memory, and five gigabytes of storage;

• One CPU, 512 MB of memory, and 5 GB of storage make up a network node.

• Compute node: 1 CPU, 2 GB of memory, and 10 GB of storage.

The research's initial phase used the CirrOS Linux distribution.

The experimental server (whose characteristics are shown in Table 4.2) authorised the creation of two virtual nodes for computation with a maximum of three instances each [1]. First, the effect of having running instances in the environment was evaluated. CPU and RAM the consumption were tested variables. The three non-load instances had been launching in unison each time for this purpose. Table 4.3 presents the findings.

The CPU use escalated as there were additional instances running at once. About 2% of processing power is unrecoverably obtained and used by each launched instance (or between 5-7% for every three running instances). Operating usage of memory stayed constant [1].

Table 4.2: Resources consumption caused by running instances.

Instances	CPU usage [%]	RAM usage [%]
0	5	96.29
3	12	97.12
6	18	96.99

The use of resources in the full load circumstances was subsequently investigated. Matrix multiplication was carried out as the CPU-intensive functioning in this study. One of the most well-liked benchmark jobs is this one [21]. The used matrices were 1024 by 1024 in size. Table 4.3 displays the measures' conclusions. It should be observed that the first compute node automatically received examples 1, 3, and 5. As can be seen, the potential of the resources was not entirely realised. This is because the server hardware specifications have a cap on the number of virtual computing machines. But the results show that the OpenStack platform enables full utilisation of the computing resources that are available [1].

Table 4.3: Resources consumption per instance

Instances	CPU Usage [%]	RAM Usage[%]
0	5	97.2
1	18	97.3
2	31	97.4
3	40	97.4
4	52	97.5
5	58	97.5
6	55	97.6

The research's enhanced component used the Ubuntu 20.04 Focal Fossa Linux version. The tests were run on servers being a member to an external cloud service provider and powered by AMD Opteron 6274 @ 2.2 GHz CPUs. The purpose of this research was to examine how performance on the computer varied depending on the OpenStack flavours (set of accessible virtual resources) that was chosen. Similar to the previous experiment, matrices measuring 2048 by 2048 have been multiplied for this purpose. Using the static scheduling clause of the OpenMP API, multiplication was parallelized. Table 4.4 lists the outcomes of the measurements.

It ought to have noted that just the task's parallelizable component multiplication was assessed. The top efficiency for a sequential circumstance (one thread) was attained with a decent flavour, which sounds remarkable given the findings. In this situation, the environment's properties, including its dispersion and virtualization, should be considered when taking consideration. The experiment illustrates how a rise in virtual resource availability may result in decreased performance. In addition to the observable switching among physical assets (threads and fragmented memory), it may also be brought on by communication between remote resources.

Table 4.4: Flavor Thread No.-Time Trade-off

Flavor	Time [s]				
m1.tiny	231.90	232.80	233.90	234.56	234.99
m1.small	54.74	55.08	55.89	56.30	57.43
m1.medium	48.91	35.74	37.01	37.83	38.92
m1.large	51.93	37.07	19.95	22.04	23.43
Number of threads	1	2	4	8	16

Table 4.4 also presents scenarios where the software takes full advantage of available resources to those achieved the best time results.

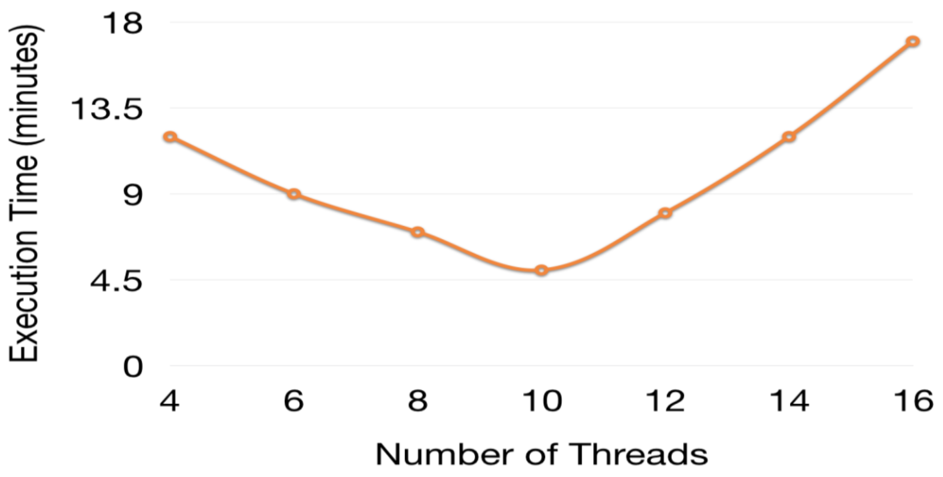


Figure 4.21: Execution-Time and Thread Tradeoff

This study demonstrates how crucial it is to choose the right virtual resources for the job at hand as well as how much spread and virtualization resources affect performance.

Table 4.5: Flavor Thread No.-Speedup trade off

Flavors	Speedups [S]				
m1.tiny	1	0.997	0.990	0.987	0.985
m1.small	1	0.997	0.979	0.972	0.951
m1.medium	1	1.369	1.322	1.293	1.255
m1.large	1	1.411	2.603	2.356	2.214
Ideal Speedup	1	2	4	8	16
Number of Threads	1	2	4	8	16

The speedup which occurred in relation to the virtual resource set can be observed in Table 4.5. The Amdahl's law, a model for anticipated speedup and the connection between parallelized applications, has been employed to establish the results [22]. The most significant speedups were obtained when the number of VCPUs and the number of threads were equal. In contrast to Table 4.4's findings, the speedup frequently does not lead to quicker task execution.

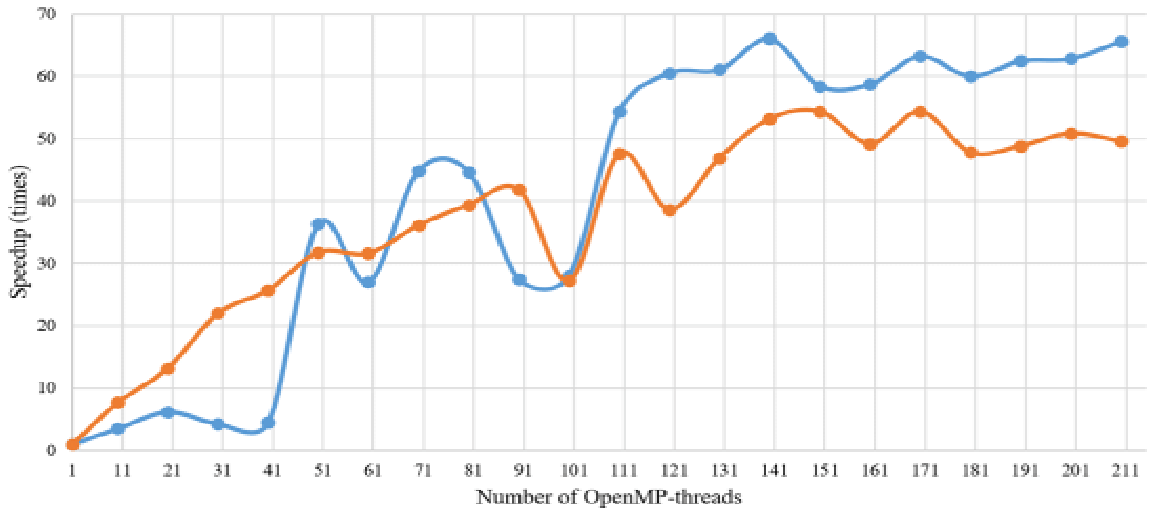


Figure 4.22: Speedup vs Number of Thread

CHAPTER-5

5.1. Conclusion

Finally, I want to say how fantastic and meaningful this internship has been. Sincerely, I can say that my time spent with RTDS proved to be beneficial to me. It goes without saying that the technical portions of my work can benefit from some improvement. Despite having minimal prior experience with cloud & devops, I think the time I spent in discovering and comprehending was valuable because it helped create a fully functional app service. One of the most significant lessons I've learned is the importance of time management and self-motivation.

Software engineering is the practise that employs engineering principles in the design, development, and technical management of software. To deal with the issues that were brought on by efforts to provide low-quality software, software engineering was developed. Problems arise when software exceeds time constraints, budgets, and quality standards. It guarantees that the computer programme is produced consistently, accurately, on time, within budget, and in compliance with the specifications. To keep up with the constantly changing user demands and the environment in which the programme is to run, software engineering has become critically important.

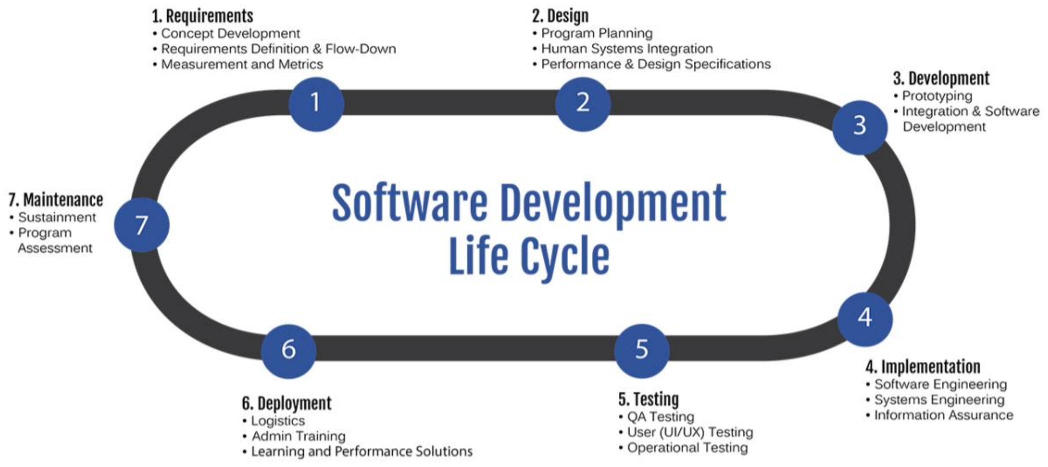


Figure 5.1: SDLC Flow Diagram

Helping with the creation and development of software is the responsibility of the software engineer trainee. They cooperate with the rest of the team to deliver dependable and safe software solutions. Software development (code, programming), code debugging, testing fresh software applications, diagnosing, and dealing with a wide range of technical challenges, working with top executives to identify issues, and offering solutions are among the duties and responsibilities of this profession. I was able to understand the task and the specifications at this time, and I will be performing this as a Cloud DevOPS engineer.

5.2. Future Scope

Consequently, whatever features and technologies I make in the future will have a major impact on small businesses by elevating them technologically and assisting them to develop smart stores through touch. As a consequence, additional work will be done later on to improve the self-checkout experience for customers and to produce a point of sale application which is more precise and current.OpenStack as an a free and open-source cloud computing platform, is increasing in popularity and there is an urgent requirement for experts with the necessary knowledge. Internships and training programmes in OpenStack provide an outstanding chance for anyone to gain the knowledge and expertise needed to satisfy this demand and succeed in the industry.

There are several reasons why OpenStack training is valuable for future employment opportunities. Firstly, as more companies shift to cloud-based solutions, the demand for OpenStack professionals is increasing rapidly. Additionally, OpenStack is being adopted by a broad range of organizations, from small and medium-sized businesses to large enterprises, leading to more job opportunities for professionals who are skilled in the platform. OpenStack is also at the forefront of cloud technology innovation, making it essential for professionals to keep up-to-date with the latest advancements through continuous learning.

OpenStack internships, on the other hand, provide a unique opportunity for individuals to gain hands-on experience working with the platform, which is highly valuable for future job opportunities. Internships also provide industry exposure, networking opportunities, and a chance to build a strong resume. Building relationships with professionals in the field can be highly beneficial for finding job opportunities and advancing in the industry.

In conclusion, OpenStack training and internships are a promising path to a successful career in cloud computing. The increasing demand for OpenStack professionals, growing adoption of OpenStack, advancements in cloud technology, high earning potential, hands-on experience, and networking opportunities make OpenStack training and internships highly valuable for individuals seeking employment in this exciting and rapidly evolving field.

REFERENCES

[1] D. Grzonka, M. Szczygieł, A. Bernasiewicz, A. Wilczyński, and M. Liszka, “Short analysis of implementation and resource utilization for the OpenStack cloud computing platform”, in Proc. 29th Eur. Conf. Modell. Simul. ECMS 2015, Albena (Varna), Bulgaria, 2015, pp. 608–614.

[2] E. W. Dijkstra, “The humble programmer”, Commun. ACM, vol. 15, no. 10, 1972. [3] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility”, Future Gener. Comp. Syst., vol. 25, no. 6, pp. 599–616, 2009.

[4] M. Armbrust et al., “A view of cloud computing”, Commun. ACM, vol. 53, no. 4, pp. 50–58, 2010.

[5] C. D. Graziano, “A performance analysis of Xen and KVM hypervisors for hosting the Xen Worlds Project”, Graduate Theses and Dissertations, Iowa State University, paper 12215, 2011.

[6] K. Scarfone, M. Souppaya, and P. Hoffman, “Guide to Security for Full Virtualization Technologies”, Recommendations of the National Institute of Standards and Technology, Special Publication 800-125, USA, 2011.

[7] M. Serafin, Wirtualizacja w praktyce (Virtualization in practice). Gliwice, Poland: Wydawnictwo Helion, 2012 (in Polish).

[8] G. J. Popek and R. P. Goldberg, “Formal requirements for virtualizable third generation architectures”, Commun. ACM, vol. 17, no. 7, pp. 412–421, 1974.

[9] M. Tajvidi, R. Ranjan, J. Kołodziej, and L. Wang, “Fuzzy cloud service selection framework”, in Proc. IEEE 3rd Int. Conf. Cloud Networking CloudNet 2014, Luxembourg, 2014, pp. 443–448.

[10] J. Sahoo, S. Mohapatra, and R. Lath, “Virtualization: A survey on concepts, taxonomy and associated security issues”, in Proc. 2nd Int. Conf. Comp. Netw. Technol. ICCNT 2010, Bangkok, Thailand, 2010, pp. 222–226.

[11] P. Mell and T. Grance, “The NIST Definition of Cloud Computing”, Recommendations of the National Institute of Standards and Technology, Special Publication 800-145, USA, 2011.

[12] Y. Mhedheb, F. Jrad, J. Tao, J. Zhao, J. Kołodziej, and A. Streit, "Load and Thermal-Aware VM Scheduling on the Cloud", in Algorithms and Architectures for Parallel Processing, LNCS, vol. 8285, pp. 101–114. Springer, 2013.

[13] M. Marks and E. Niewiadomska-Szynkiewicz, "Hybrid CPU/GPU platform for high performance computing", in Proc. 28th Eur. Conf. Model. Simul. ECMS 2014, Brescia, Italy, 2014, pp. 508–514.

[14] K. Smelcerz, "Recent developments in mobile cloud scheduling: State-of-the-art, challenges and perspectives", J. Telecommun. Inform. Technol., no. 4, pp. 51–57, 2013.

[15] F. Gens, "Defining Cloud Services and 'Cloud Computing", IDE, 2008 [Online]. Available: http://blogs.idc.com/ie/?p=190

[16] OpenStack Website and Documentation [Online]. Available: http://www.openstack.org/

[17] Google Trends [Online]. Available: http://www.google.com/trends/

[18] O. Sefraoui, M. Aissaoui and M. Eleuldj, "OpenStack: Toward an Open-Source Solution for Cloud Computing", Int. J. Comp. Appl., vol. 55, no. 3, pp. 38–42, 2012.

[19] K. Pepple, Deploying Openstack, 2nd ed. O'Reilly Media, USA, 2011.

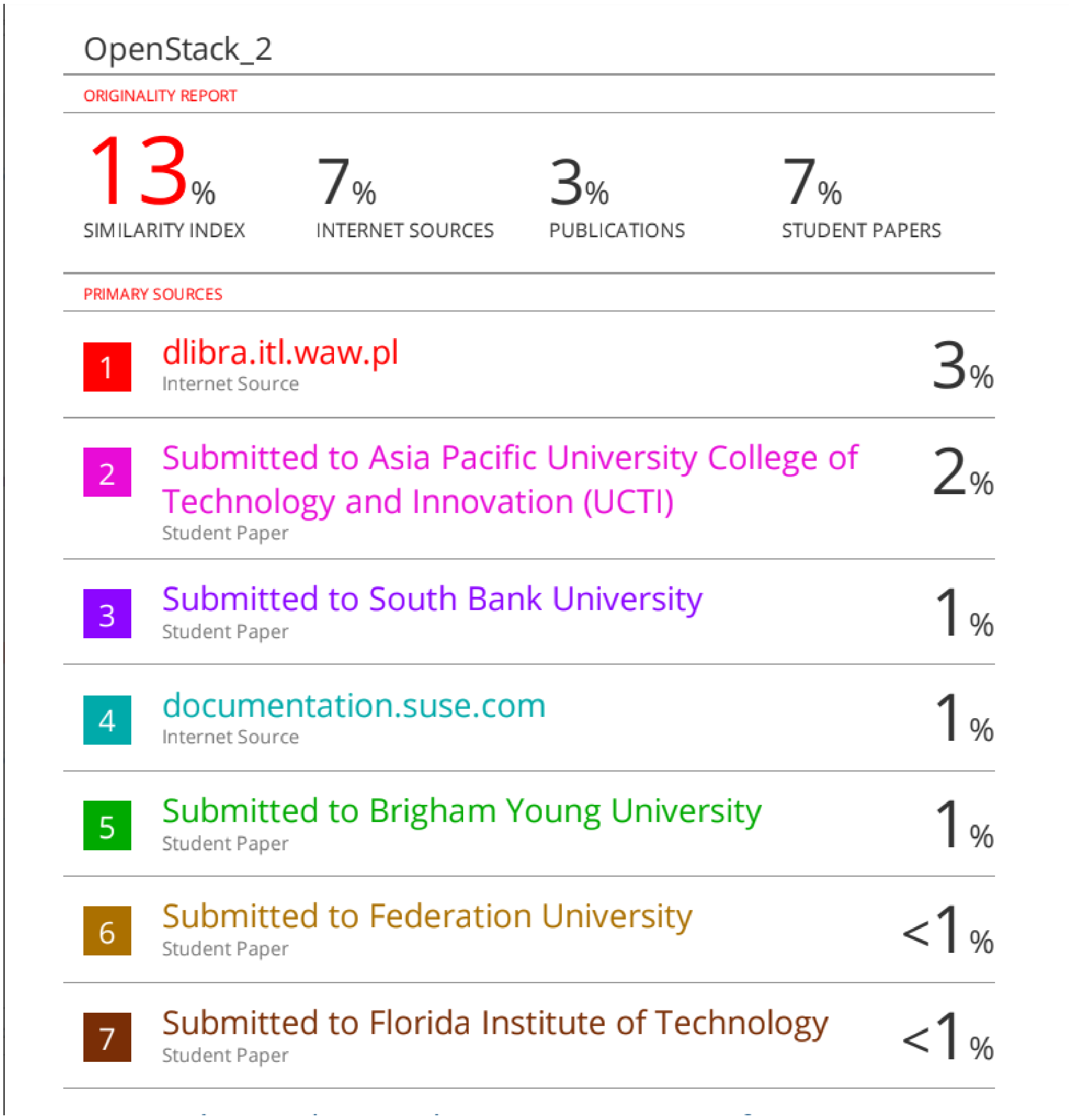
[20] K. Jackson and C. Bunch, OpenStack Cloud Computing Cookbook, 2nd ed. Packt

Publishing, UK, 2013.

[21] J. M. Cecilia, J. M. Garc´ıa, and M. Ujaldón, "The GPU on the Matrix-Matrix Multiply: Performance Study and Contributions", in Parallel Computing: From Multicores and GPU's to Petascale, B. Chapman et al., Eds. Advances in Parallel Computing, vol. 19, pp. 331–340, 2010.

[22] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities", in Proc. Spring Joint Comp. Conf. AFIPS '67 (Spring), Atlantic City, New Jersey, USA, 1967, pp. 483–485.

Plagiarism Report



OpenStack_2

ORIGINALITY REPORT

13%	7%	3%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dlibra.itl.waw.pl Internet Source	3%
2	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	2%
3	Submitted to South Bank University Student Paper	1%
4	documentation.suse.com Internet Source	1%
5	Submitted to Brigham Young University Student Paper	1%
6	Submitted to Federation University Student Paper	<1%
7	Submitted to Florida Institute of Technology Student Paper	<1%