CHAPTER 4

A MODEL OF SECURITY ARCHITECTURE ON PRIVATE CLOUD USING OPENSTACK

Cloud computing is becoming more widely utilized, it is important for enterprises to understand ways to maximize benefits and minimize risks of a move to the cloud also to carefully consider its operating expenses essential. Since most of the enterprise already has traditional IT setup and the major investment would be spent for having as a data center which has limitation and difficulty to change the amount of storage and workload.

The Private cloud is a model or architecture and often presented as being the solution for all computing issues in the enterprise. It is distinct and secure cloud based environment which can be accessed and managed by the enterprise. It is very closer to the more traditional model of individual local access networks (LANs) that is used in the past by enterprises but having the added advantages of virtualization. This can be also called either “Internal” or “corporate” or “enterprise” cloud and it will be protected by a firewall. The enterprise will have more control over its data and applications with this kind of setup. It also promises benefits such as energy savings, cost savings, rapid deployment and customer empowerment.

There could be additional security offered by ring fenced cloud model that could be ideal for any enterprises which needs to store and process private data or carry out some sensitive tasks. For instance, a private cloud service could be utilized by a financial company that is required by regulation to store sensitive data internally and who will still want to benefit from some of the advantages of cloud computing within their business infrastructure,
such as on demand resource allocation. On the other hand, technically the virtualization is not private cloud and private cloud is far beyond virtualization. Data Storage is the one of the important and primary resource enterprises wanted to keep with their control.

4.1 **Private Cloud – Strategy Planning**

Enterprises has to build a strategy or roadmap for the private cloud planning. The figure 4.1 shows the model for the planning and each will be described below. Since the model should be a reiterative process, it can have more repetition.

![Figure 4.1 Roadmap - Private cloud build strategy](image)

- **Application Classification** - need to classify the list of applications that can be moved to the private cloud.

- **Measure Application** – need to measure the impact or importance of the application and make it in the proper queue then this can be taken with the order.

- **Categorize Application and Platform** – most of the enterprises should be having a heterogeneous application and platform, need to categorize by platform.s
- **Migrate** – need to start migrate the application with the respective platform and also has to think about the support services for the platform that enterprise builds.

### 4.2 Private Cloud Challenges

There are key challenges that should be considered before the enterprises to build private cloud. So enterprises that

- Should think of bridging private cloud with the existing infrastructure or integrate with legacy systems and data.
- Should be able to have 24x7 support service for their end user when they experience any issue in private cloud.
- Have enough on-premises servers that can be virtualized and scaled for current and future demands.

There are some additional challenges that businesses need to consider on day-to-day business management.

- Managing the enterprise applications and responsible for updates and patches in the applications.
- Handling the security issues using like cryptography and authentication.
- Monitors and tests the system to ensure data and applications are properly backed up and readily retrievable.
- Managing the business critical deployment in the application and those activity tracking.
4.3 Architecture of OpenStack

OpenStack (OpenStack, 2012) is a free and open-source software cloud computing software platform. Users primarily deploy it as an infrastructure as a service (IaaS) solution. OpenStack works with popular enterprise and open source technologies making it ideal for heterogeneous infrastructure. It is highly configurable for many reasons. OpenStack community is one among the fastest growing open source communities in the world.

The figure 4.2 shows the series of OpenStack’s interrelated components that control pools of processing, storage, and networking resources throughout a data center in which the user manages these all, through a web-based dashboard and command-line tools. The aim of OpenStack is to produce open source cloud computing platform that will meet the needs of public and private clouds regardless of size by being simple to implement and massively scalable everywhere.

![Figure 4.2 An Architecture for OpenStack](image)
Each component is used to deal with specific resource in the overall OpenStack cloud platform which is explained below in detail.

- **OpenStack Identity Keystone** – Keystone (Keystone, 2012) is responsible for managing the restricted access services like authorization and authentication details for all other components in OpenStack platform which provides a central directory of users mapped to the component services.

- **OpenStack Networking (Neutron)** - This is pluggable and scalable API driven component to manage networks for all the virtual hosts (instances) and provide IP address both private and public. End user or enterprises can control and monitor the virtualization by dashboard called “Horizon”.

- **OpenStack Horizon (Dashboard)** – This component has web based portal to interact with all other OpenStack components to administrate the resources. It has core support to ship with three different dashboards like User, System and Settings dashboard. For the dashboard application development purpose, the Horizon application comes with a set of abstraction APIs for the core OpenStack components in order to provide a consistent, stable set of reusable methods for developers.

- **OpenStack Volume (Cinder)** – It can be detachable block storage to any virtual instance similar to an external hard drive. It provides the APIs to create and manage a service that provision the storage in the form of block storage called as “Cinder volumes”. It also features basic storage capabilities like snapshot or volume clones and management which are often enhanced through vendor-specific drivers.
- **OpenStack Compute (Nova)** - It allows to create and manage virtual servers using the machine images through the dashboard or API for Infrastructure as a Service (IaaS). It is the core service component in the OpenStack platform to build a cloud which provisions and manages even large networks of virtual machines.

- **OpenStack Image (Glance)** - It is a imaging service access by Restful API which helps to discover, register and retrieve virtual machine images. It follows the client-server architecture to make VM images available and it can store in a various location from simple file systems to object-storage systems like the OpenStack Swift project.

- **OpenStack Object Store (Swift)** - It is used for Object storage that can be highly scalable redundant unstructured data store designed to store large amounts of data with lower cost. Since it supports highly scalable architecture which means that it can scale to thousands of machines with tens of thousands of hard drives. It is designed to support horizontally scalable so here is no single point of failure. It can be more appropriate for large-scale deployments as the cluster grows larger. It can store static objects like multimedia files, backups, store images, archives etc.

### 4.3.1 Key Benefits of OpenStack Private Cloud

- Open Architecture – any component that can be adopted and used.
- Robust – there should be common or centralized administration to obtain best performance benchmarking.

- Scalability – automatic infrastructure creation when the server gets high load or downtime.

- Hybrid Cloud – bridge private and public cloud.

- No Downtime – environment should be always up and running.

- Client Support – there should be defined SLA and 24/7 support for the users.

4.4 A Model of Security Architecture for Private Cloud using OpenStack

The basic requirement to setup private cloud environment using OpenStack is network and right hardware. Since OpenStack allows us to setup everything in same flat network, it will not be the best approach for security reason. At least two different network (support VLAN) environments (Nurmi et al. 2009) are strongly suggested for the best model to build a secure private cloud. In this, one can be used to manage the network traffic and for virtual machines or cloud instances to communicate with to each other. This means each compute nodes need to have two network cards and the network manager. Each network should be configured and run on different IP ranges.

The next step is to identify and determine what and where to deploy all the OpenStack components. The minimal requirement and mandatory components to build private cloud in OpenStack platform is that networking,
compute, keystone and storage. In figure 4.3, shows the model architecture to build secure private cloud. A classic deployment should have one controller and series of compute nodes. The controller manages the message server, database and other components to orchestrate the private cloud when the compute nodes run the instances.

![Diagram of Security Architecture for Private Cloud using OpenStack]

**Figure 4.3 A Model of Security Architecture for Private Cloud using OpenStack**

There can be also another approach to break out the individual pieces of the controller for the better performance such as placing the database server (e.g., MySQL database) on a different physical box. To ensure the security measure, each of these pieces is installed on a secure host and that will be attached into the network which is required to keep the private cloud running.
As part of best practice for enterprises private cloud to end user that web management and administrative console and network manager can be exposed. For the security practice, by default the console runs with “http” protocol and it should be configured with Apache and SSL to make sure it runs with “https” secure protocol. Additionally, all these servers need to be hardened (Chapter 3) and back-end traffic isolated from user connections.

4.4.1 OpenStack Components

4.4.1.1 Security Architecture using Keystone Identity Service

Keystone provides a single point of integration for OpenStack identity, token, catalog and Policy services for projects. Keystone ensures the listed below things to help us to model the secure architecture to build private cloud.

- **User** - incoming requests are from valid or approved from person, service or system

- **Role** - group of user assigned to set of privileges and perform specific operations.

- **Credentials** - specific user provides username and password or an authentication token.

- **Authentication** - identity service issues authentication token that user is allowed to make subsequent requests.

- **Token** - each token has a scope describing accessible resources. A token may be revoked at any time and is valid for a finite duration.
- **Endpoint** - network accessible address which usually described by URL for authenticated users.

There are two primary functions which can be achieved by using keystone.

- **User Management** - tracks of user and their security scope in which they are permitted to do.

- **Service Catalog** - provides a catalog of what services can be available for the user.

4.4.1.2 **Networking Management using OpenStack Networking (Neutron)**

OpenStack Network is called as “Neutro” and it provides “networking as a service” between interface devices which is managed by other OpenStack services components. It allows to do list of below options:

- To build complex and rich “Network” topologies by applying advanced network policies.

- To create “Subnet” to represent IPv4 or IPv6 address blocks from which IPs to be assigned to VMs on a given network are selected.

- Allows us to create “Port” that represents virtual or logical switch ports in the network. Virtual instances attach their interfaces into ports and the logical port defines the MAC address and the IP address which is assigned to the interfaces plugged into them.
4.4.1.3 **Infrastructure Management using OpenStack Compute**

The main strength of OpenStack Compute is to support different virtualization technologies (King et al. 2006) in the infrastructure level.

- Manage virtualized commodity server resources like CPU, memory, disk, and network interfaces
- Programmatically allocate IPs and VLANs to manage networks
- Designed for automation and security to make it easy for end users to manage compute resources and prevent users from impacting each other with excessive API utilization
- Supports the distributed and asynchronous architecture.
- Virtual Machine (VM) image management like run, reboot, suspend, resize and terminate instances
- Security Groups and Role Based Access Control (RBAC)

4.4.1.4 **Storage and Backup Management using OpenStack Storage**

OpenStack Storage supports Object and Block storage with different and unique architecture for deployment and many enterprises have variety of storage needs with respect to performance and price requirements.

- Reliability and data redundancy to protect from failures
- Scale vertically and horizontally-distributed storage
- Account/Container/Object structure and Built-in replication
- RAID not required
- Fully integrated with Compute to attach block volumes and exposed the usage report.
4.4.2 Monitoring Dashboard (Horizon)

OpenStack Horizon is a web based GUI application and it provides a custom management interface to create a typical base dashboard structure for the OpenStack underlying resources. It brings all OpenStack projects together in a single window. It is also exposed as an API so that end user can create their own custom dashboard using the API and framework. It is built on Django with Python. Django helps to create the complex database-driven websites and it emphasizes reusability and plug ability.

4.4.3 Applications (Corporate)

Applications are business specific and owned by enterprises or corporate. It can be mobile or web based application. In this dissertation, there are two main applications (Chapter 6) are written, which are called “File Management Application (FMA)” and “Efficient File Transfer Management (EFTM)”. FMA is a web based GUI application and EFTM is a service based offline application, but both are deployed in the tomcat web server.

4.5 Design and Implement Private Cloud Infrastructure

To maximize the benefits of a private cloud to be sure the enterprises to design and implement by choosing best and flexible cloud automation tools and processes which can avoid unnecessary overhead to hire more system administration. These are key points to be implemented:

- **Automation** - Infrastructure automation provisioning which can be achieved by automation tools. These are called as “DevOps” tools (DevOps for Developers, 2012).

- **Resource Management Dashboard** - The cloud management dashboard is essential and the infrastructure automation provisions are carried out behind the scenes by scripts. It
presents an easy-to-use interface that allows users to provision and de-provision resources, track resource use, modify the access controls to resources and view services available in the private cloud.

- **Workload and Service Monitoring** - Cloud automation monitoring will be useful for real-time monitoring (hardware failures or disrupted services) and long-term business support planning (aggregate information like use of cloud resources, demand for particular types of resources and costs).

- **Virtualization** – It allows the resource sharing with more than one environment. The resource will be like operating system, storage, server or network components.

### 4.6 Post Private Cloud Migration

Once the private cloud is up and running, then the enterprises has to provide the right tools to help users to maintain and get maximum benefit out of cloud services. There are few important points highlighted below.

- Implement automation for the actual private cloud benefits. This can be done by using some famous DevOps tools like “Puppet” or “Chef”.

- Meter the private cloud usage on “Bill for the usage or infrastructure” basis.

- Security tools are essential for the private cloud and it can be done by using self-service model.

- Maintaining a private cloud which means that server failover and downtime must not lead any problem.

- Determines that it is cost-effective with “Return on Investment (ROI)” software or tools.