Cloud Native Application Development

Case-study

1. Introduction

This document describes the project undertaken by PalC Networks for developing **cloud native DNS System** application for our Malaysian client IncepXion.

1.1. Introduction on Cloud Native Application Development

For starters, we need to understand the difference between cloud based and cloud native application development.

Cloud-native development refers to application development that is container-based, dynamically orchestrated and leverages microservices architectures. Because cloud-native applications run in containers and are dynamically orchestrated, they exhibit many of the attributes of applications deployed in cloud-based infrastructures, such as elastic scalability and high availability.

Cloud-native applications are fundamentally container-native applications and require developers to achieve familiarity with containers and associated orchestration frameworks such as Kubernetes. The need to demonstrate proficiency with Kubernetes requires developers to obtain expertise with developer tools that provide insight into relationships between discrete containers. Furthermore, developers need proficiency with the design of microservices-based application architectures that are executed in Kubernetes.

Whereas cloud-based development refers to application development executed by means of a browser that points to a cloud-based infrastructure, cloud-native development refers more specifically to application development grounded in containers, microservices, and dynamic orchestration.

2. DNS System

The objective was to implement IncepXion patented Telephony based DNS System in Cloud native environment. The high level components of Telephony based DNS system are represented in below diagram.

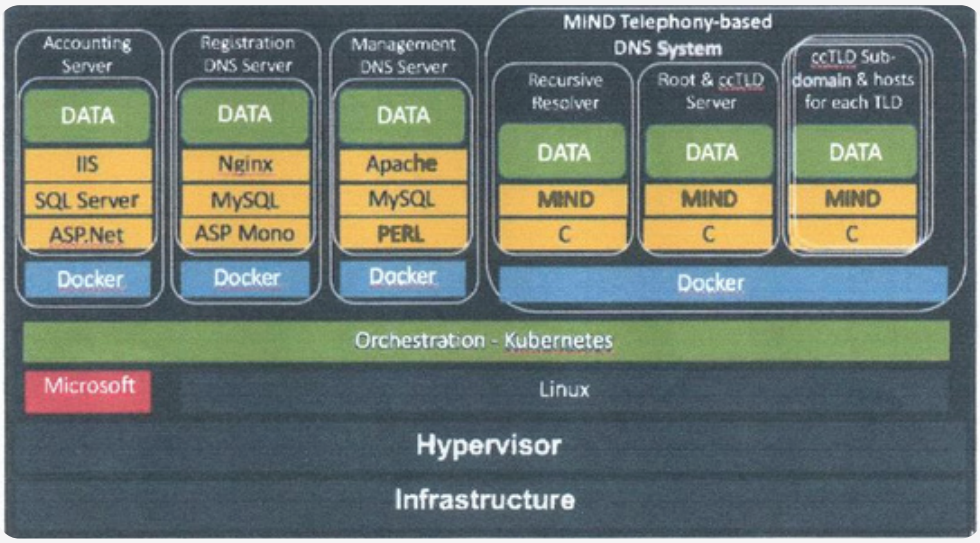

Figure1: Telephony based DNS System Components.

This requirement was achieved by using Openstack(Queens) as Iaas. After the VM orchestrations, Kubernetes was used for container orchestration.

2.1. VM Orchestration

We had used Openstack-Ansible(OA) flavour of Openstack for VM orchestration. It is an official Openstack project which aims to deploy production environments from source in a way that makes it scalable while also being simple to operate, upgrade and grow.

As the name suggests development framework of Openstack-Ansible is built using Ansible. The basis for all deployed Openstack software will be from source. This means that OpenStack services and their python dependencies will be built and installed from upstream source code as found within the OpenStack Git repositories. This project provides a system which will get patched and updated if needed from Openstack upstream code. Openstack services are run using LXC containers, this makes it more scalable.

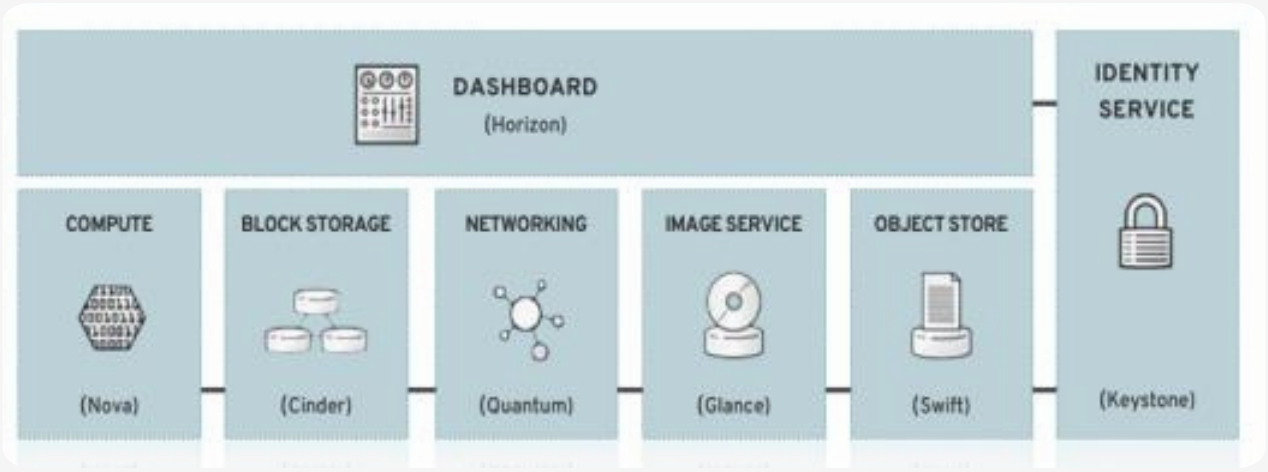

Figure 2: Core Services of Openstack

features supported in DNS system project:

1. Open vSwitch(OVS) for Layer 2 Services.
2. OpenDaylight(ODL) for Layer 3 Services.
3. Heat Templates for faster deployments.
4. Storage – Block Storage (Cinder), Object Storage (Swift) and CEPH Integration for storage servers.

Openstack-Ansible version Queens was used for this project.

2.2. Container Orchestration

Kubernetes was used for container orchestration.

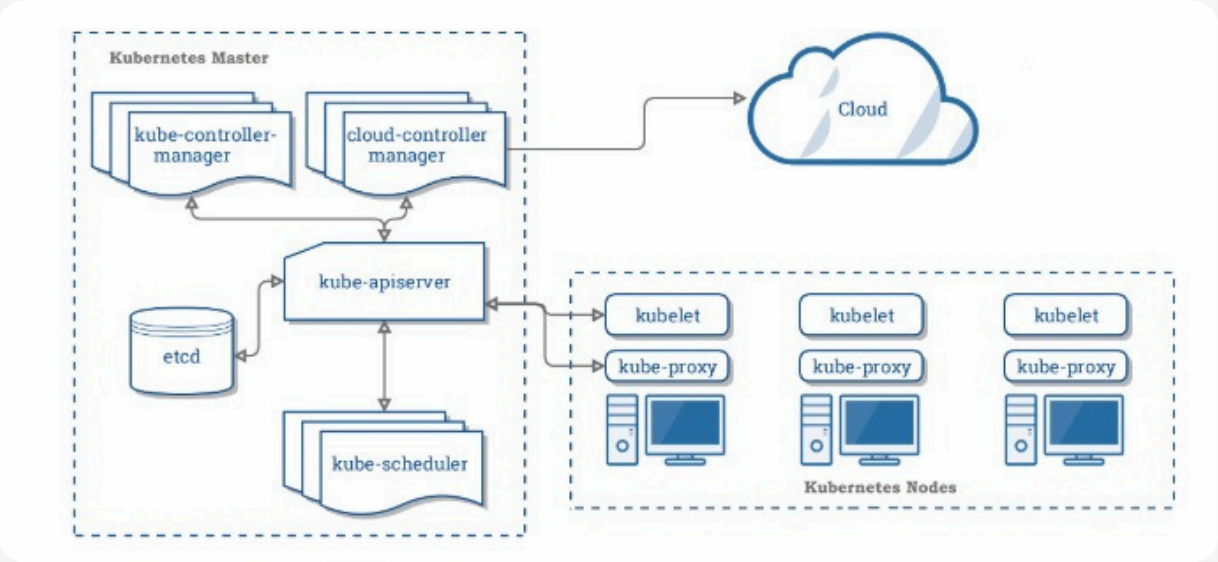

Figure 3: Kubernetes Cluster and its components.

Kubernetes is a powerful open-source system, initially developed by Google, for managing containerized applications in a clustered environment. It aims to provide better ways of managing related, distributed components and services across varied infrastructure. Kubernetes, at its basic level, is a system for running and coordinating containerized applications across a cluster of machines. It is a platform designed to completely manage the life cycle of containerized applications and services using methods that provide predictability, scalability, and high availability. Kubernetes Components are depicted in above diagram.

Highlights of the features supported in DNS system project:

1. Each component is dockerized.
2. Container to Container networking across VMs.
3. Clustering of some of the services.

3. GLOSSARY

DNS: Domain Name System
VM: Virtual Machine
OA: Open stack-Ansible
OVS: Open V Switch
ODL: Open Daylight