Deploy and Manage OpenStack Pike on Ubuntu

Course Introduction

About the Course

This course is designed to help you understand the following services and how they fit together in an OpenStack Cluster:

- Identity
- Image
- Compute
- Networking
- Dashboard
- Block Storage
- Object Storage
- Orchestration

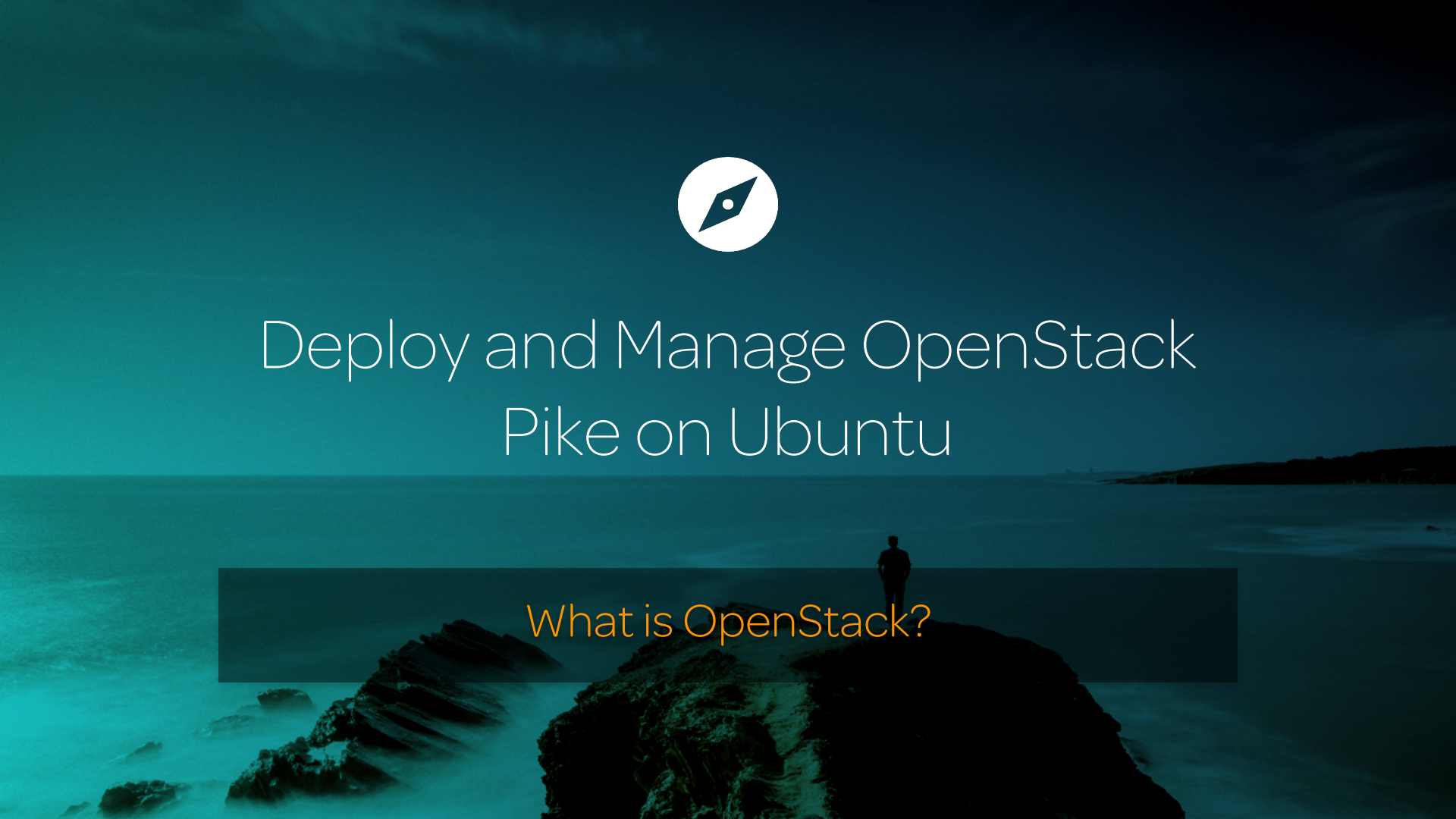
We will also:

- Walk through the basic architecture of a cluster and its networking
- Install a five-node OpenStack cluster on VirtualBox.
- Launch and SSH into an instance we create on our cluster

Course Pre-Requisites

Prior to taking this course, it is recommended you have an understanding of basic networking including VLANs, routing, and subnetting, and have taken the following courses:

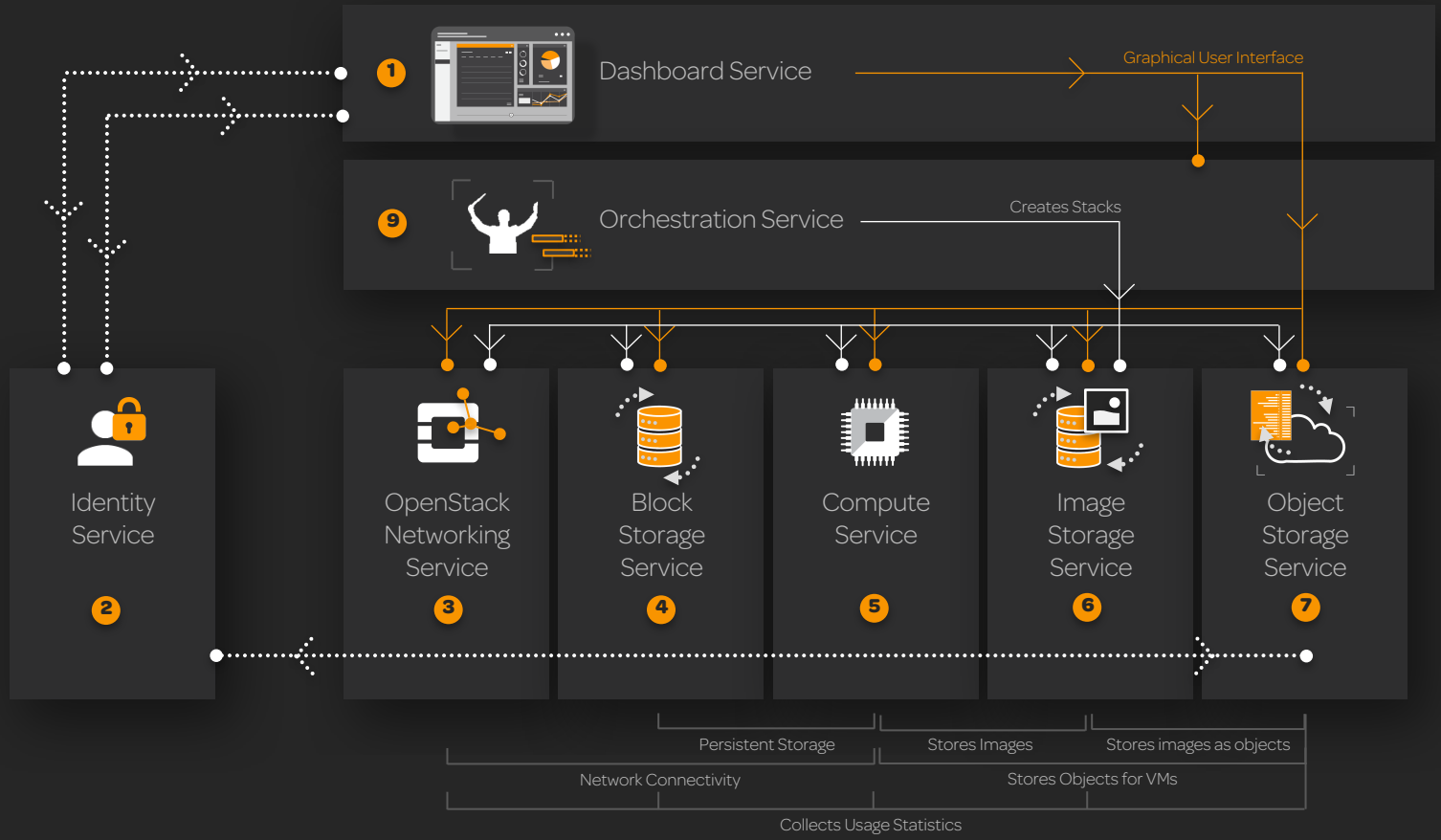
- Linux Essentials
- OpenStack Essentials



Deploy and Manage OpenStack Pike on Ubuntu

What is OpenStack?

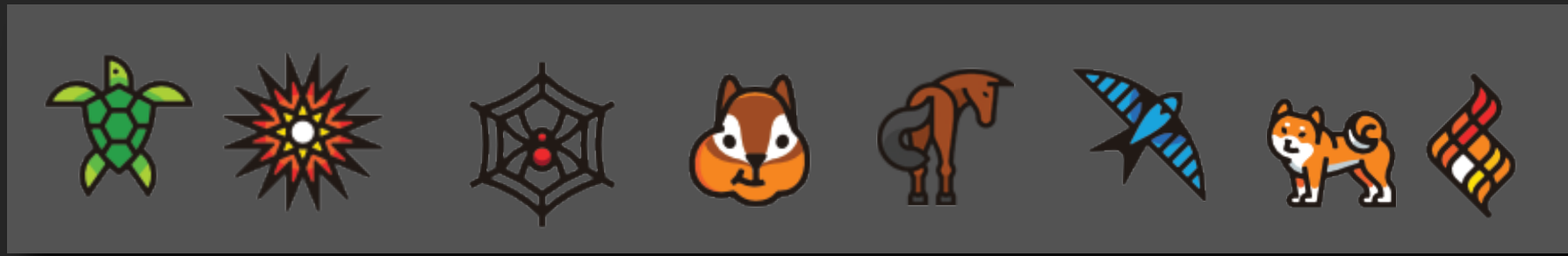
What is OpenStack?



Services Overview

The software platform consists of six core services as well as 13 optional services. There are also numerous projects which support OpenStack including several projects for deployment.

Each service offers an Application Programming Interface (API) that facilitates the integration of the various services which can be accessed via an integrated Command-Line Interface (CLI) as well.



Deploy and Manage OpenStack Pike on Ubuntu

OpenStack Services

Identity Service - Keystone

- Provides an authentication and authorization service for other OpenStack services
- Provides a catalog of endpoints for all OpenStack services

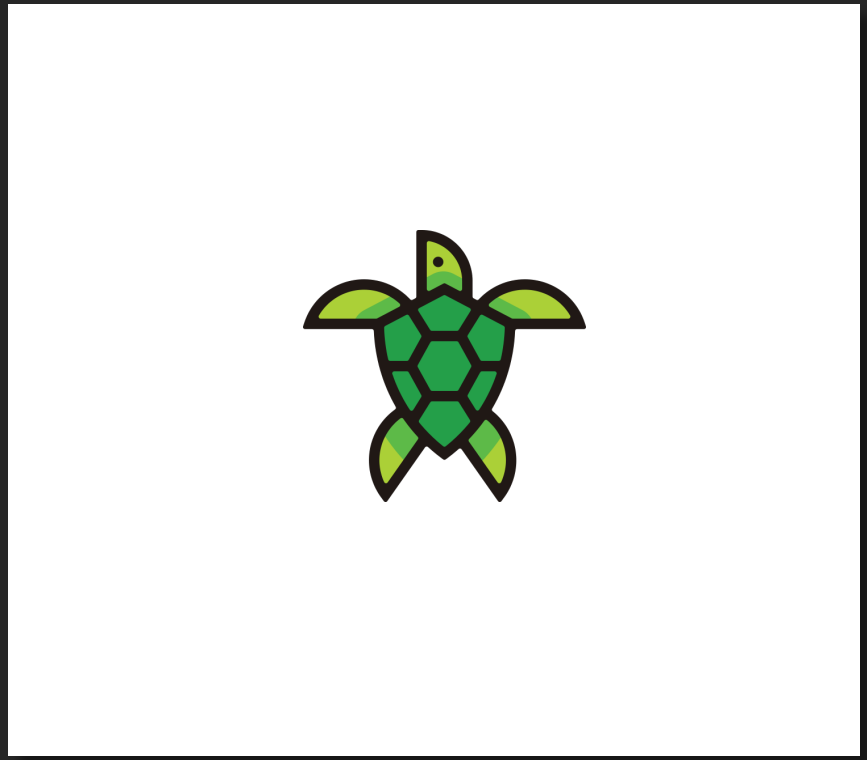
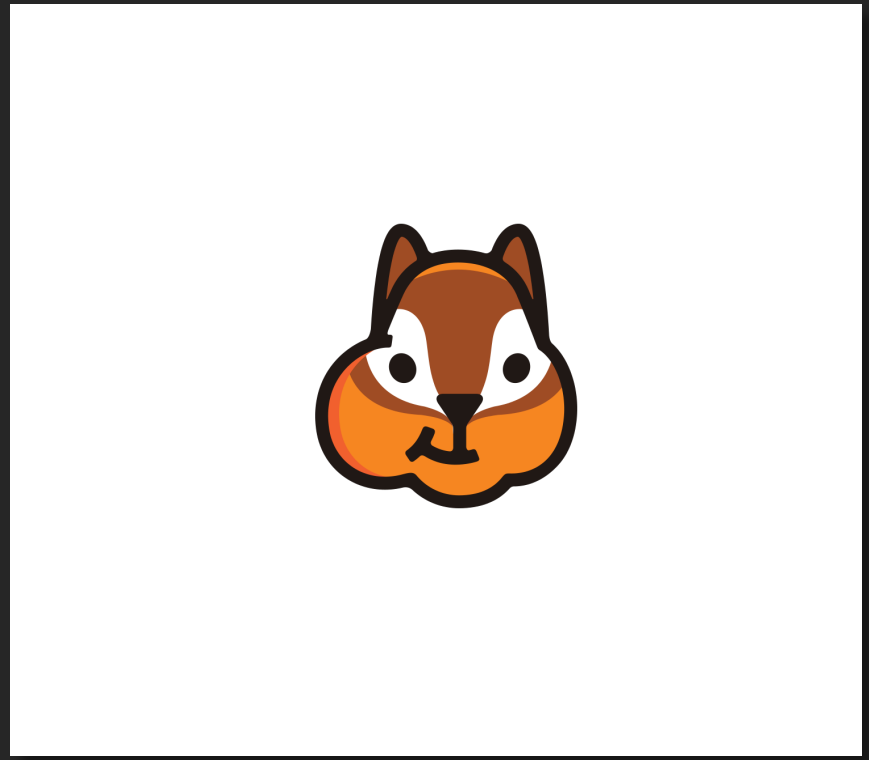


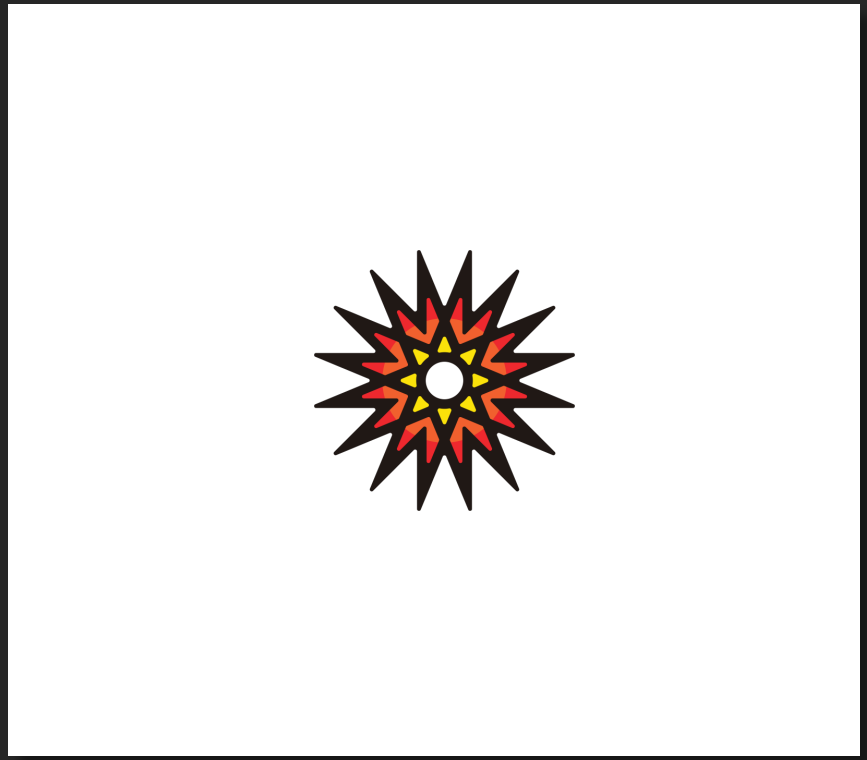
Image Service - Glance

- Stores and retrieves virtual machine disk images
- OpenStack Compute makes use of Glance during instance provisioning



Compute Service - Nova

- Manages the lifecycle of compute instances in an OpenStack environment
- Responsibilities include spawning, scheduling and decommissioning of virtual machines on demand



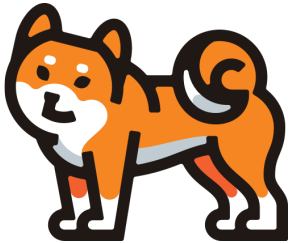
Networking Service - Neutron

- Enables Network-Connectivity-as-a-Service for other OpenStack services, such as OpenStack Compute
- Provides an API for users to define networks and the attachments into them
- Has a pluggable architecture that supports many popular networking vendors and technologies

Dashboard - Horizon

Provides a web-based self-service portal to interact with underlying OpenStack services, such as:

- Launching an instance
- Assigning IP addresses
- Configuring access controls

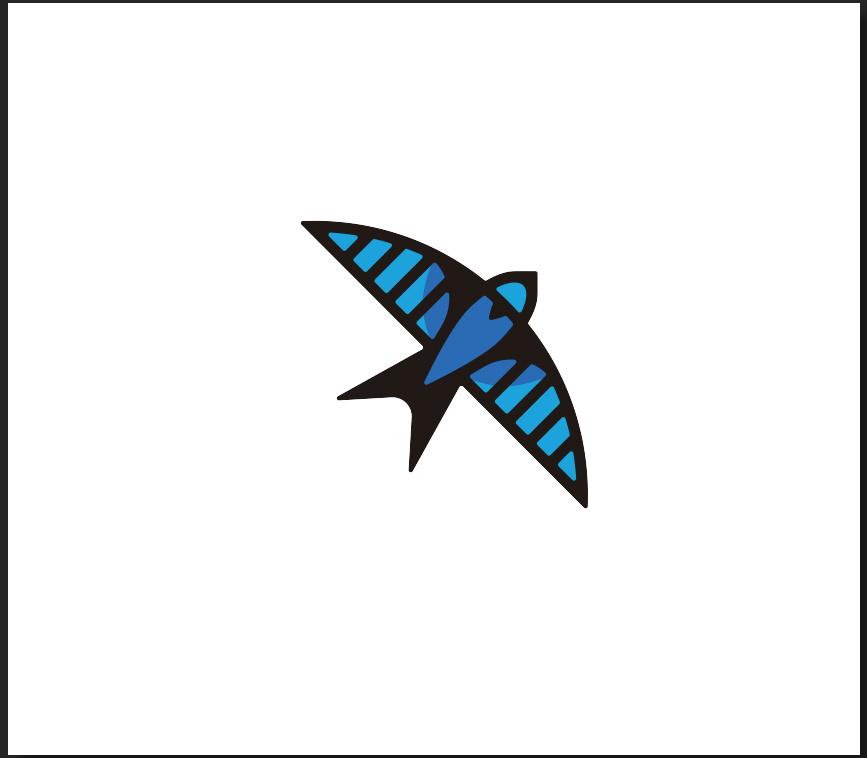


Block Storage Service - Cinder

- Provides persistent block storage to running instances
- Its pluggable driver architecture facilitates the creation and management of block storage devices

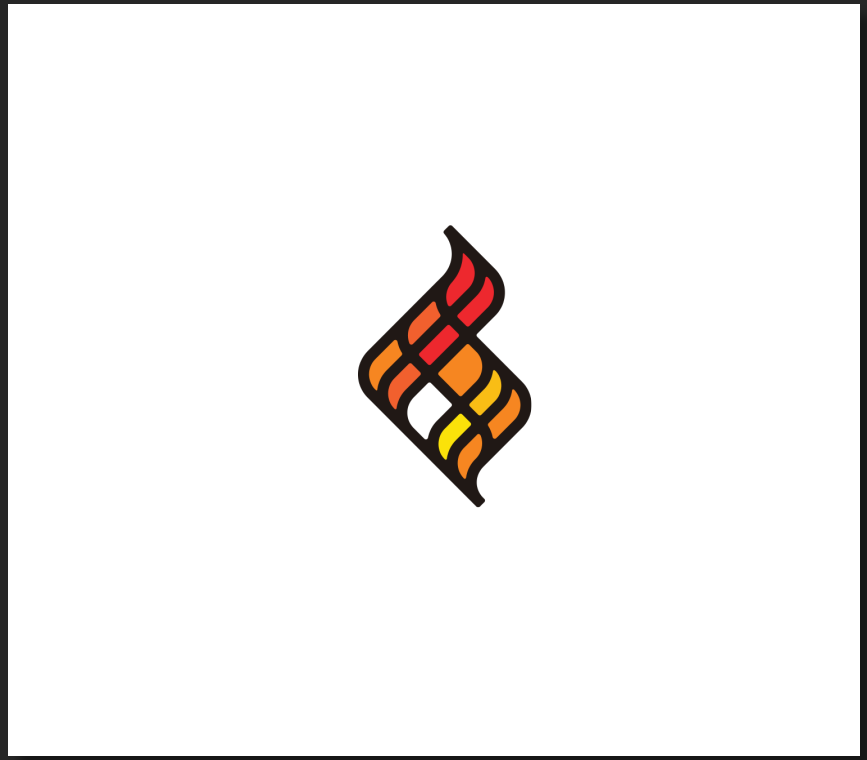
Object Storage Service - Swift

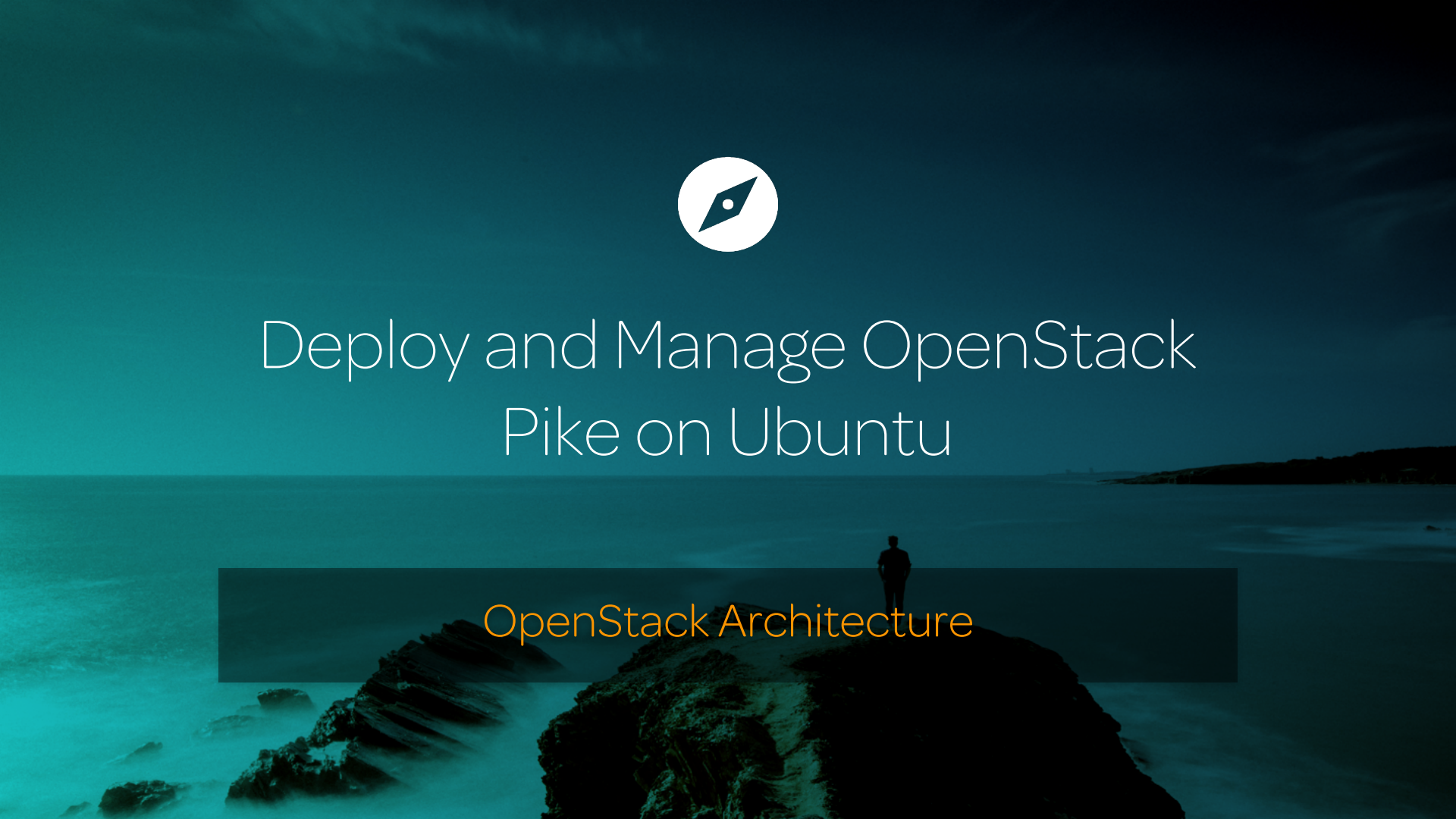
- Stores and retrieves arbitrary unstructured data objects via a RESTful, HTTP based API
- It is highly fault tolerant with its data replication and scale-out architecture
- Its implementation is not like a file server with mountable directories



Orchestration Service - Heat

Orchestrates multiple composite cloud applications by using either the native HOT template format or the AWS CloudFormation template format, through both an OpenStack-native REST API and a CloudFormation-compatible Query API.

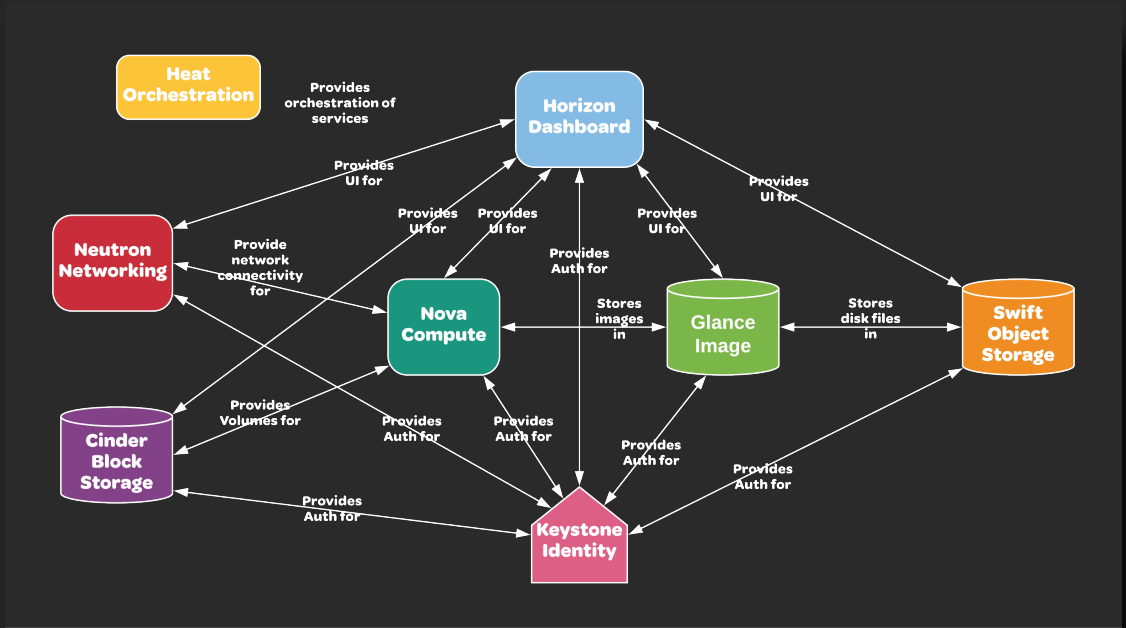




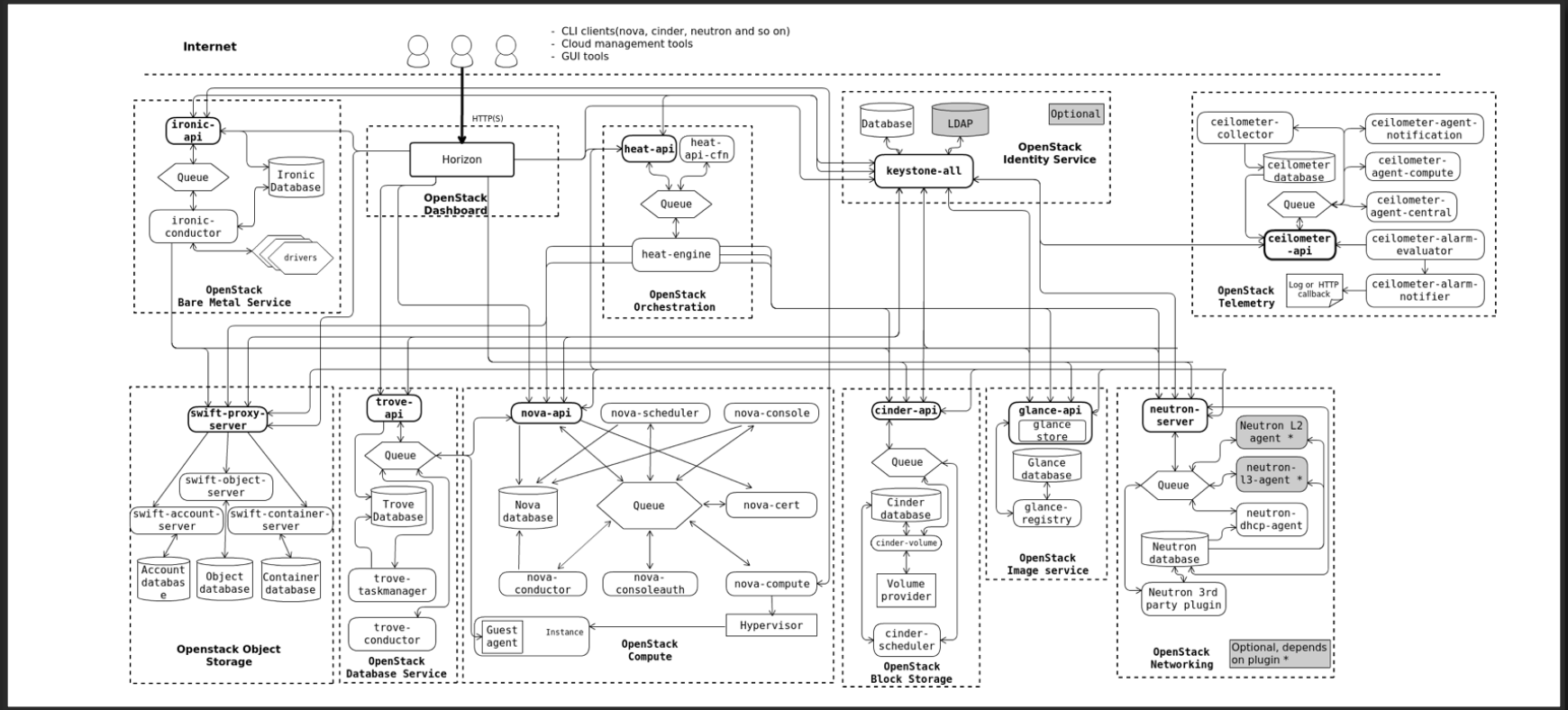
Deploy and Manage OpenStack Pike on Ubuntu

OpenStack Architecture

How the Services Interact



Logical Architecture



Deploy and Manage OpenStack Pike on Ubuntu

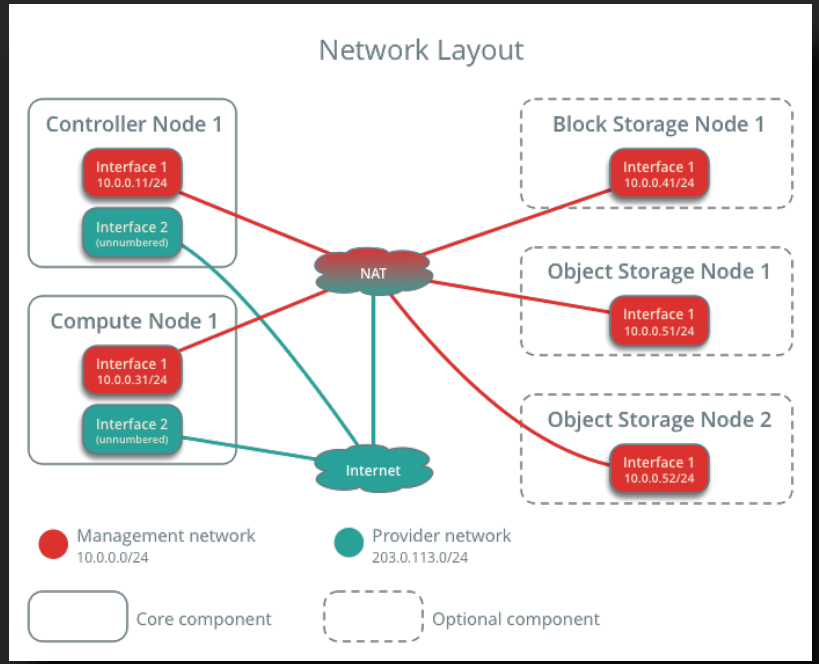
Virtual Box Setup

Software Downloads

- Virtual Box
  - Software - https://www.virtualbox.org/wiki/Downloads
  - Documentation - https://www.virtualbox.org/wiki/End-user_documentation

- Ubuntu 16.04.3 LTS (Xenial)
  - Server Software - https://www.ubuntu.com/download/server

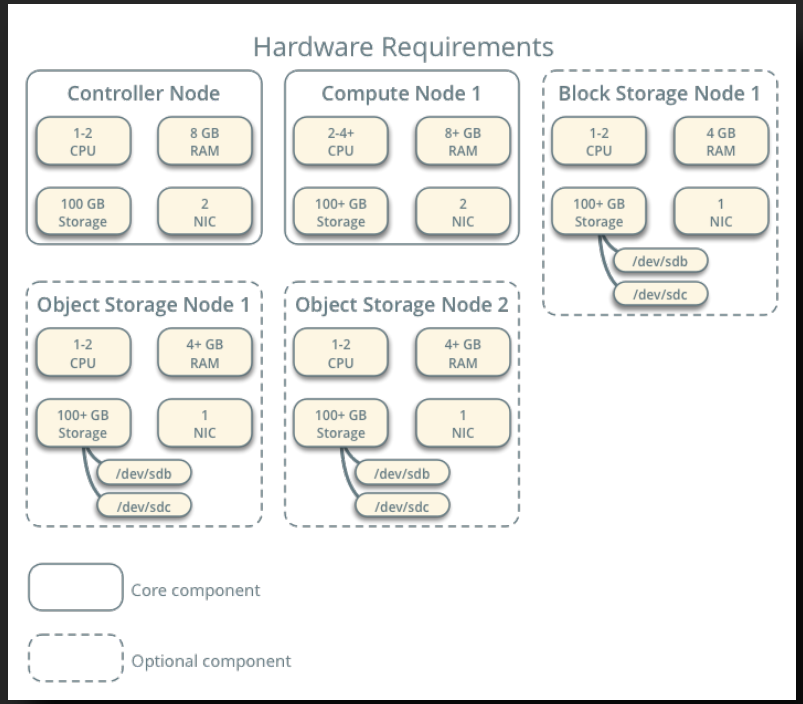
Network Layout



Virtual Box Network Configuration

- vboxnet0 – Management Network
  - Network CIDR – 10.0.0.0/24
- vboxnet1 - Provider Network
  - Network CIDR – 203.0.113.0/24

Hardware Requirements



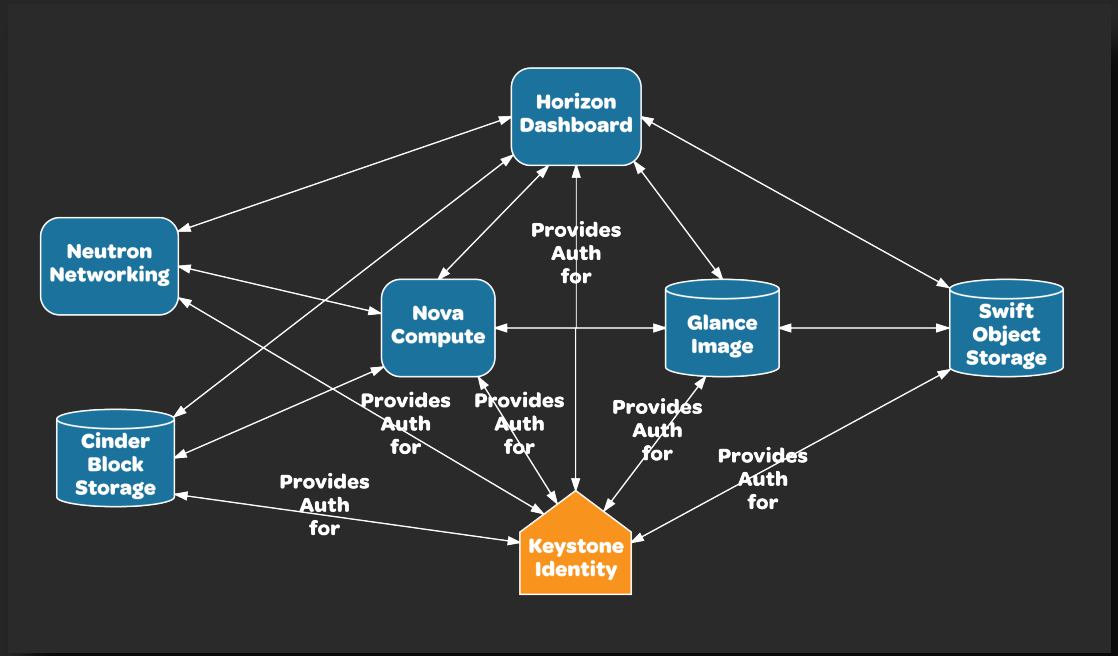
Virtual Box Node Configuration

- Controller Node:
  - 1 processor
  - 4 GB memory
  - 10 GB storage
- Compute Node:
  - 1 processor
  - 4 GB memory
  - 10 GB storage
- Block Storage Node:
  - 1 processor
  - 2 GB memory
  - 8 GB drive 1
  - 10 GB drive 2
- Object Storage Nodes (2 nodes):
  - 1 processor
  - 2 GB memory
  - 8 GB drive 1
  - 8 GB drive 2
  - 8 GB drive 3

Deploy and Manage OpenStack Pike on Ubuntu

Identity Service Overview

Identity Service - Keystone



Identity Terminology

Service

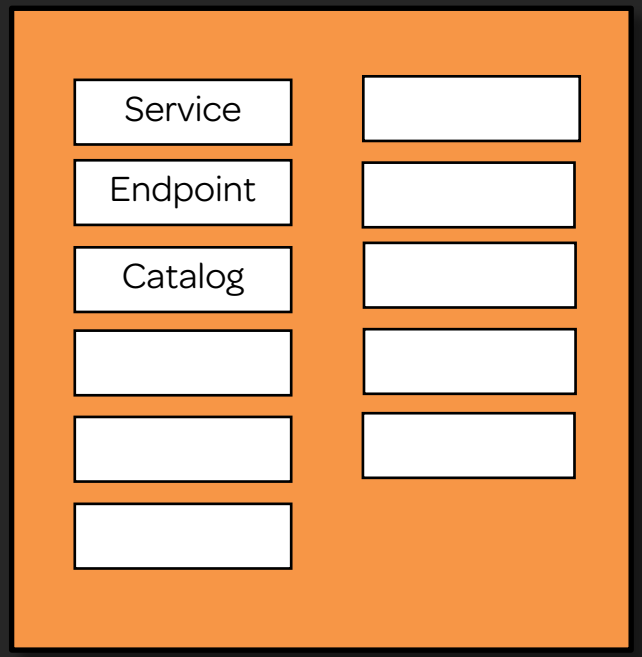
- Refers to a service running in OpenStack such as Compute (Nova), Object Storage (Swift), or Image Service (Glance).
- Provided by one or more endpoints in which users can access resources and perform operations.

Endpoint

- Network-accessible addresses where you can access a given service via a URL and port.
- Can be configured to service requests on three URLs: a public facing URL, an administration URL, and an internal URL.

Catalog

- A listing of the different endpoints that have been created for the OpenStack services.



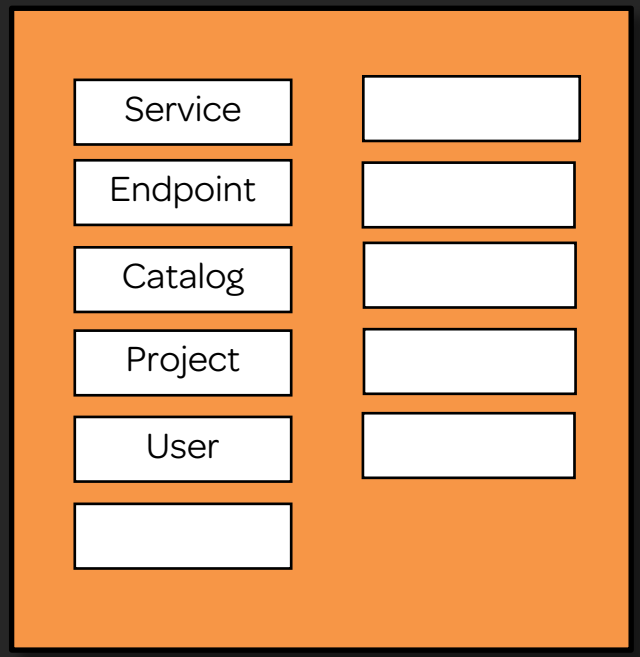
Identity Terminology

Project

- A set of any resources assigned to an isolated group of users.

User

- Used by services and administrators to manage the OpenStack cloud.
- A digital representation of a person, system, or service who uses OpenStack services.
- Keystone validates the incoming requests made by the user who claims to be making the call.
- Have a login and may be assigned a token to access resources.
- Must be assigned to a project and be assigned a role.



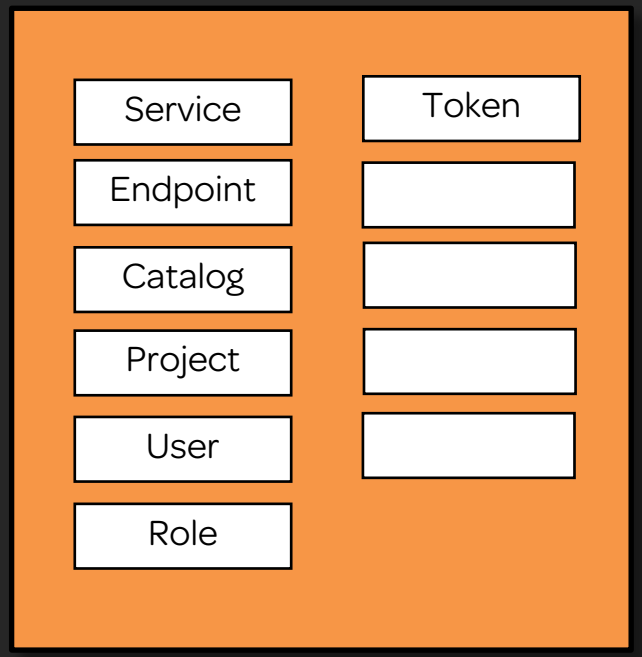
Identity Terminology

Role

- Roles are the permissions given to users within a project.
- A first-class piece of metadata associated with the user for the project. It is assigned directly to users or groups for projects or inherited from domains.

Token

- Identifying credential associated with a user, an arbitrary bit of text that is used to access resources.
- A token may be revoked at any time and is valid for a finite duration.
- While OpenStack Identity supports token-based authentication, the intention is to support additional protocols in the future.



Identity Terminology

Group

A collection of users.

Credential

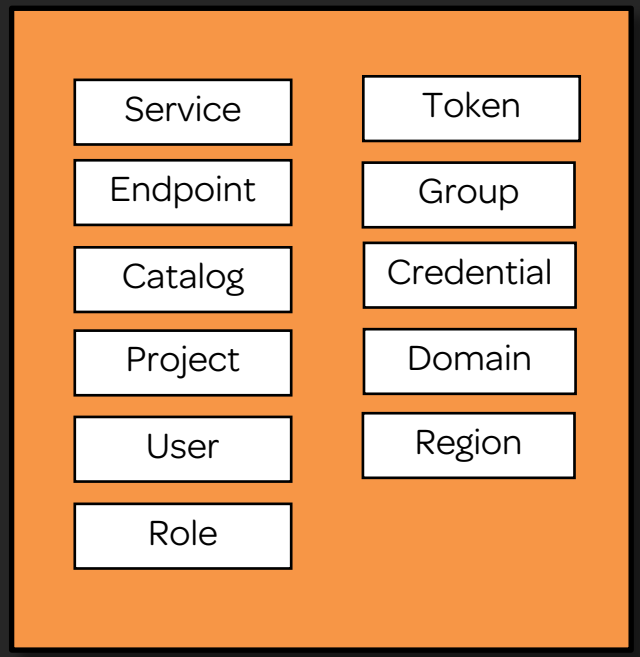
Data that is known only by the user that proves who they are, such as a username and password, a username and API key, or an authentication token.

Domain

Collection of projects, groups, and users that define the administrative boundaries for managing OpenStack Identity entities.

Region

Separates the OpenStack environments that have dedicated API endpoints but utilize a common Keystone service.



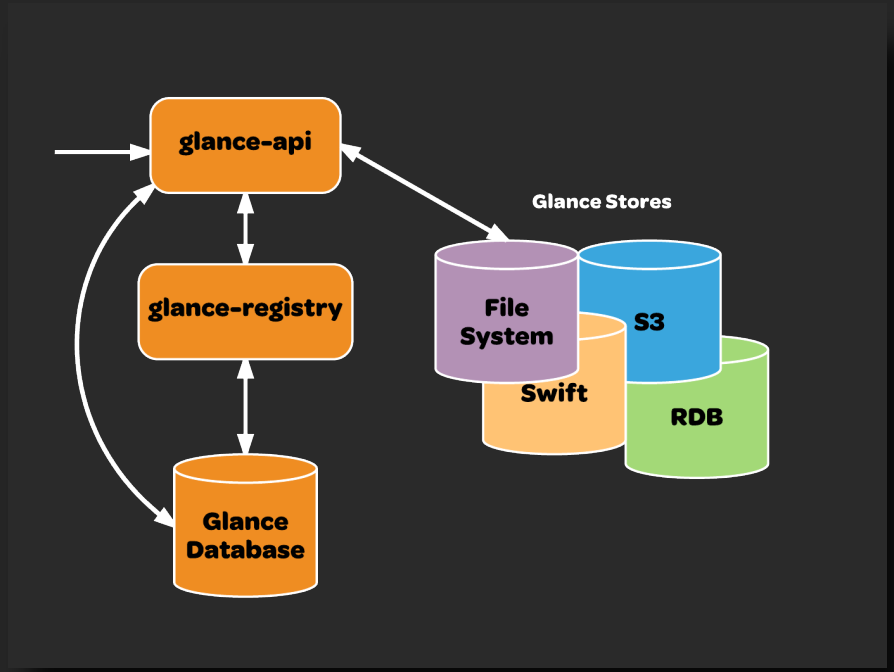
Deploy and Manage OpenStack Pike on Ubuntu

Image Service Overview

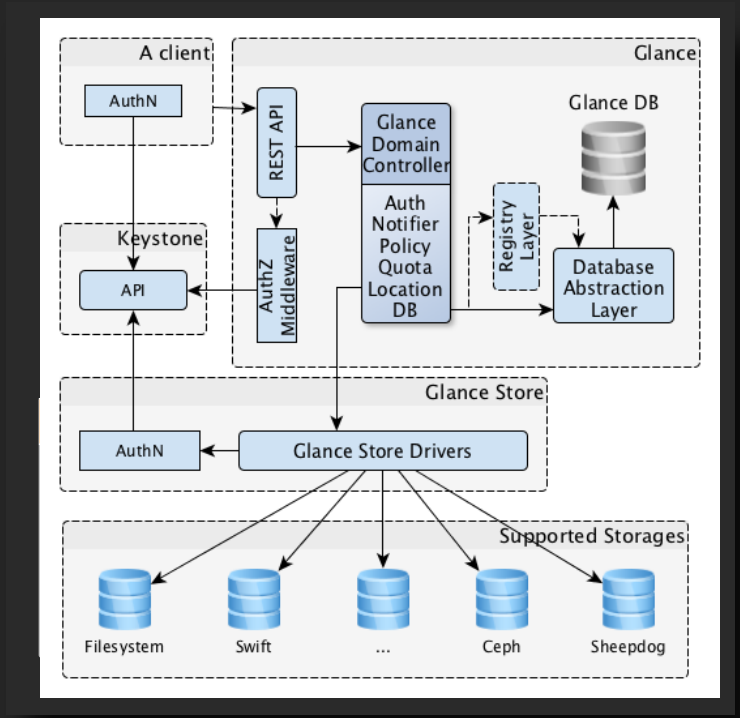
Image Management Service - Glance

- Glance is the component that discovers, registers, and retrieves virtual machine images
- Capabilities of the Image Service:
    - Administrators can create base templates from which their users can start new compute instances
    - Users can choose from available images or create their own from existing servers
    - Snapshots can be stored in the Image Service so that virtual machines can be backed up quickly

Glance Architecture



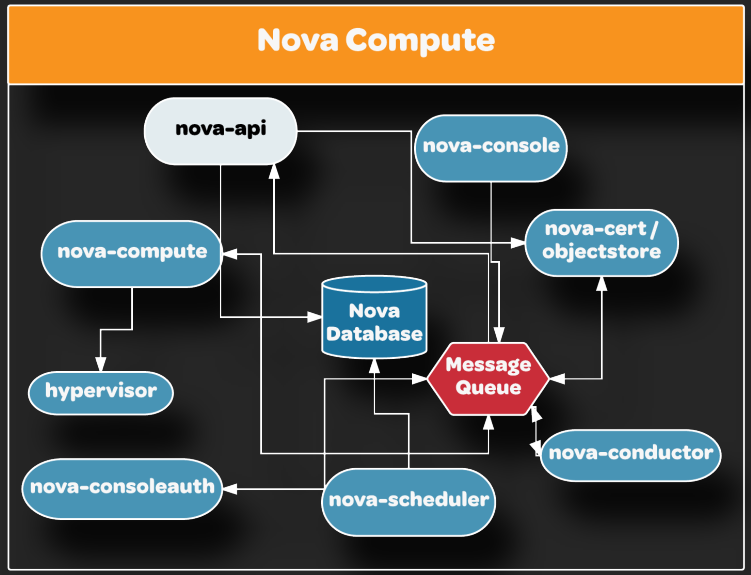
Glance Workflow



Deploy and Manage OpenStack Pike on Ubuntu

Compute Service Overview

How Does Nova Work?



Nova API

- The Compute API, run by the nova-api daemon, is the component of OpenStack Compute that receives and responds to user requests, whether they are direct API calls or via the CLI tools or dashboard
- The API enables users to specify an administrative password when they create or rebuild a server instance. If the user does not specify a password, a random password is generated and returned in the API response
- Nova API is stateless and can be used for HA deployments

Nova Database

- The Nova database stores the current state of all objects in the compute cluster
- The database is usually MySQL, but SQLite and PostgreSQL can also be used
- Nova API talks to the MySQL Nova DB via SQLAlchemy
- The nova-conductor service is the only service that writes to the database. The other Compute services access the database through the nova-conductor service
- If nova-conductor is not used, entries to the database are mostly written by the nova-scheduler service, although all services must be able to update entries in the database

Nova Scheduler

- Compute uses the nova-scheduler service to determine how to dispatch compute requests
- The nova-scheduler service determines which host will launch a VM
- In the context of filters used with the nova-scheduler, the term host means a physical node that has a nova-compute service running on it
- The daemon generally resides on the controller node

Nova's Message Queue

- Message Queue is a unified way for collaboration between sub-components
- RabbitMQ is the most commonly used queueing system
- Can use multiple queues within a single RabbitMQ instance:
    - Used by the OpenStack services to build machine state
    - Each compute node has a queue
- RabbitMQ nodes failover on the application and the infrastructure layers when used in HA

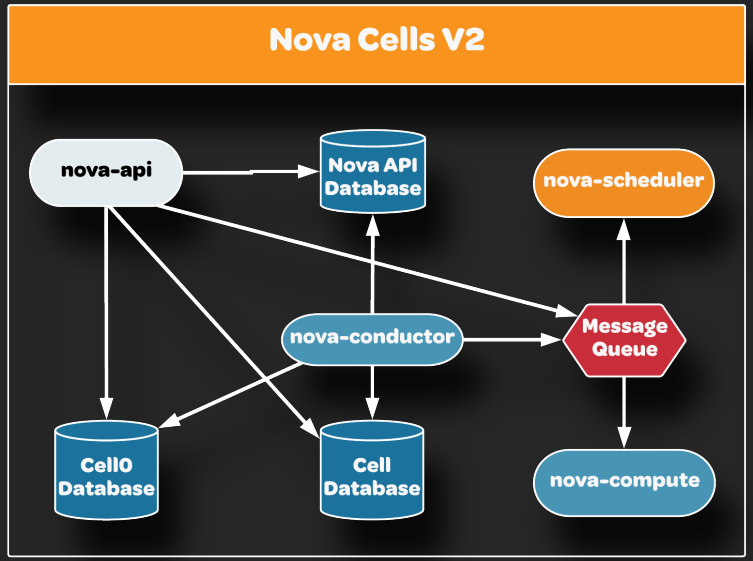
Nova Compute

- This daemon primarily creates and terminates VMs via Hypervisor API
- It allows multiple hypervisor types per cloud
- Libvirt/KVM and QEMU are the most commonly used in deployments

Nova Conductor

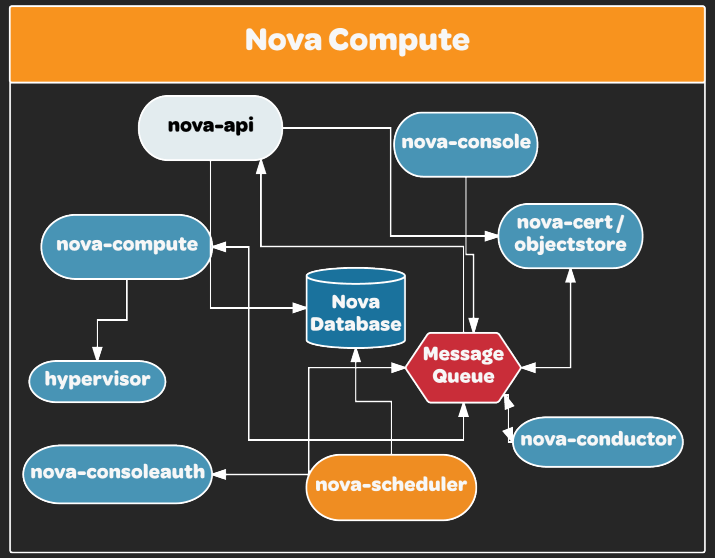
- The nova-conductor service enables OpenStack to function without compute nodes accessing the database
- Conceptually, it implements a new layer on top of nova-compute
- Just like other Nova services, such as nova-api or nova-scheduler, it can scale horizontally

Nova Cells V2



Virtual Machine Placement

- The nova-scheduler interacts with other components through the queue and central database
- All of the compute nodes periodically publish their status, resources available, and hardware capabilities to nova-scheduler through the queue
- nova-scheduler then collects the status data and uses it to make decisions when requests come in
- By default, the compute scheduler is configured as a filter scheduler
- In a default configuration, the scheduler considers hosts that meet the following criteria:
    - In the requested Availability Zone (AbilityZoneFilter)
    - Have sufficient RAM available (RamFilter)
    - Capable of servicing the request (ComputeFilter)



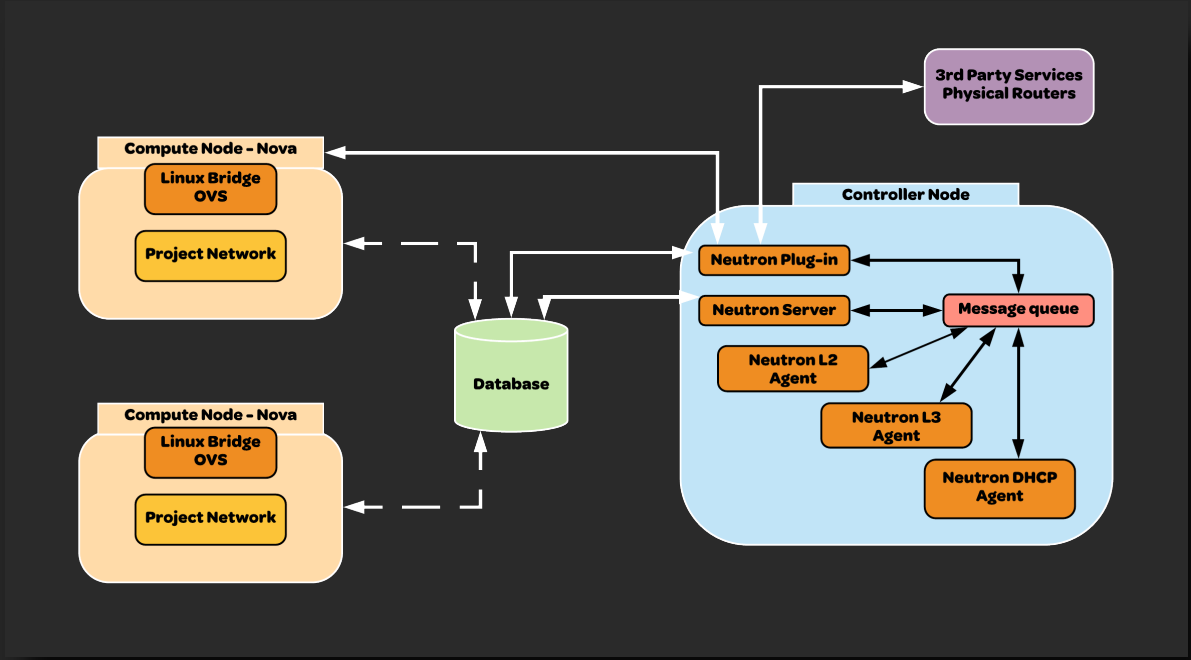
Placement API

- Is a separate REST API stack and data model used to track resource provider inventories and usages, along with different classes of resources
- Example resource providers are:
    - A compute node
    - A shared storage pool
    - An IP allocation pool
- The types of resources consumed are tracked as classes
- Example resource classes are:
    - DISK_GB
    - MEMORY_MB
    - VCPU

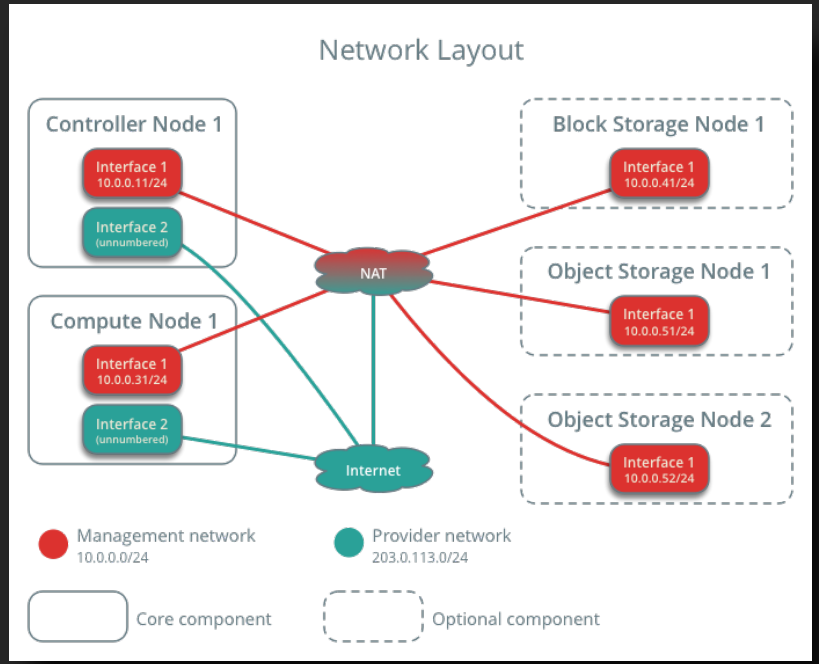
Deploy and Manage OpenStack Pike on Ubuntu

Networking Service Overview

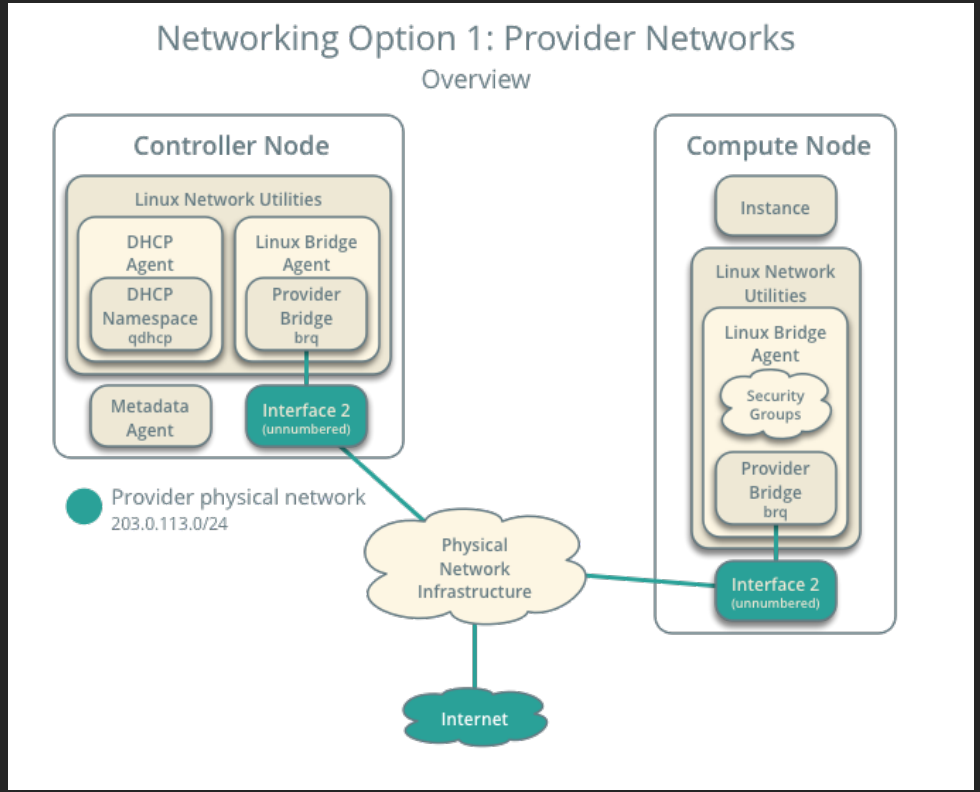
Neutron Architecture



Network Layout



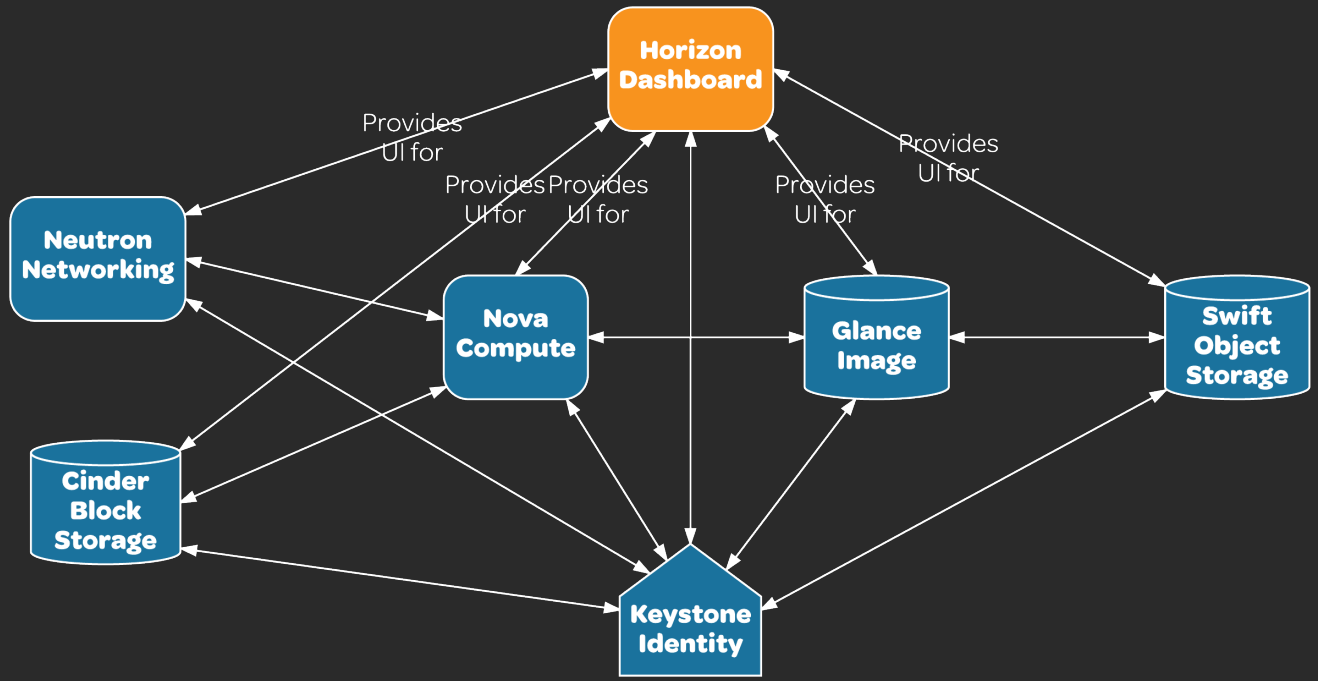
Network Layout

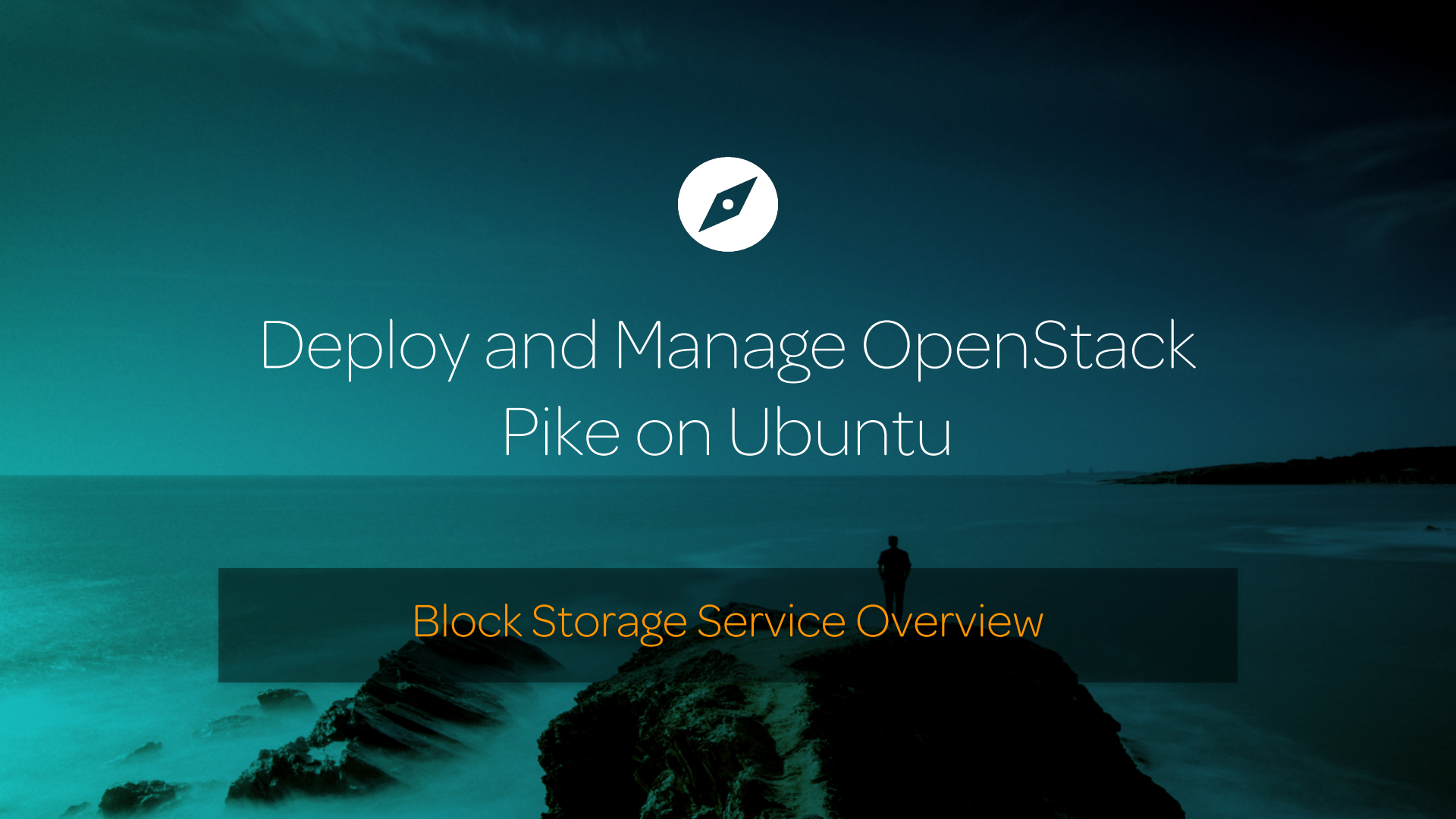


Deploy and Manage OpenStack Pike on Ubuntu

Dashboard Overview

Horizon

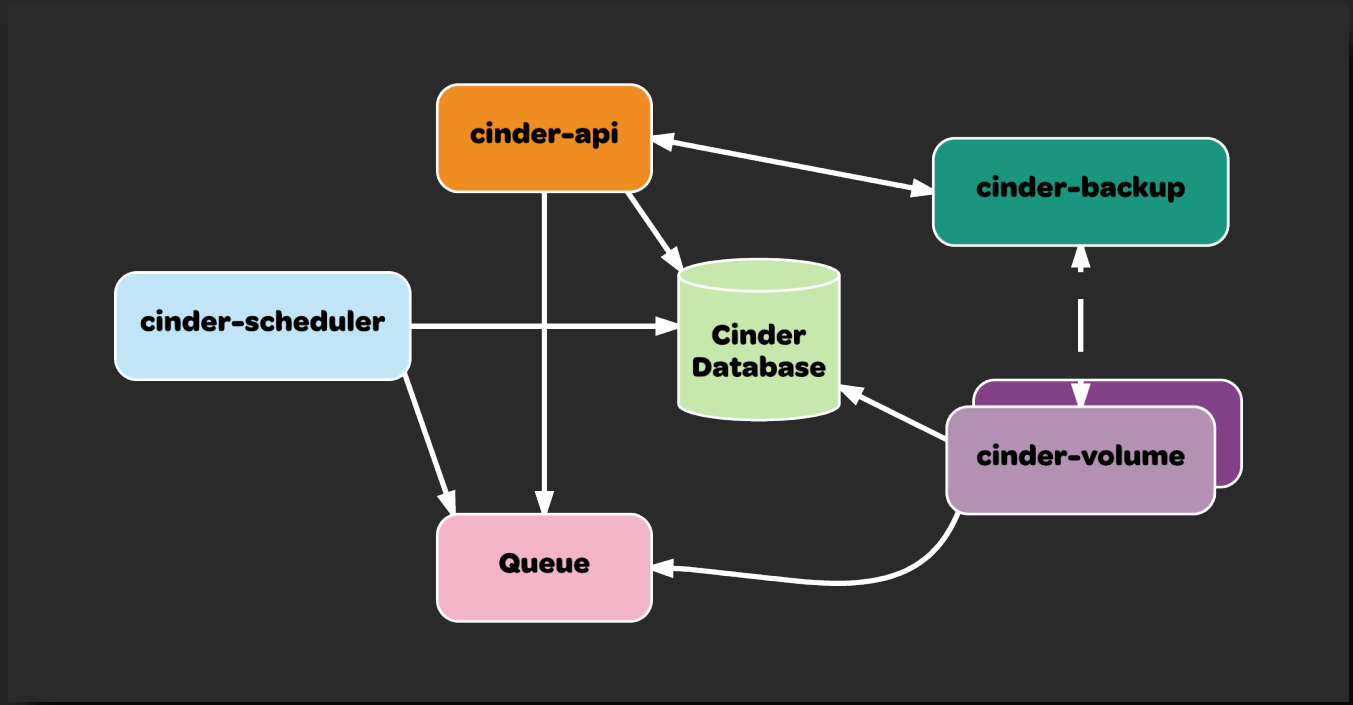

Horizon Dashboard
Provides UI for
Provides UI for
Provides UI for
Provides UI for
Provides UI for
Neutron Networking
Nova Compute
Glance Image
Swift Object Storage
Cinder Block Storage
Keystone Identity



Deploy and Manage OpenStack Pike on Ubuntu

Block Storage Service Overview

Cinder Architecture



Deploy and Manage OpenStack Pike on Ubuntu

Object Storage Overview

What is Object Storage?

Object Storage (Swift) is a robust, highly-scalable, and fault-tolerant storage platform for unstructured data such as objects.

It is commonly used to archive and back up data, with use cases in virtual machine image, photo, video, and music storage.

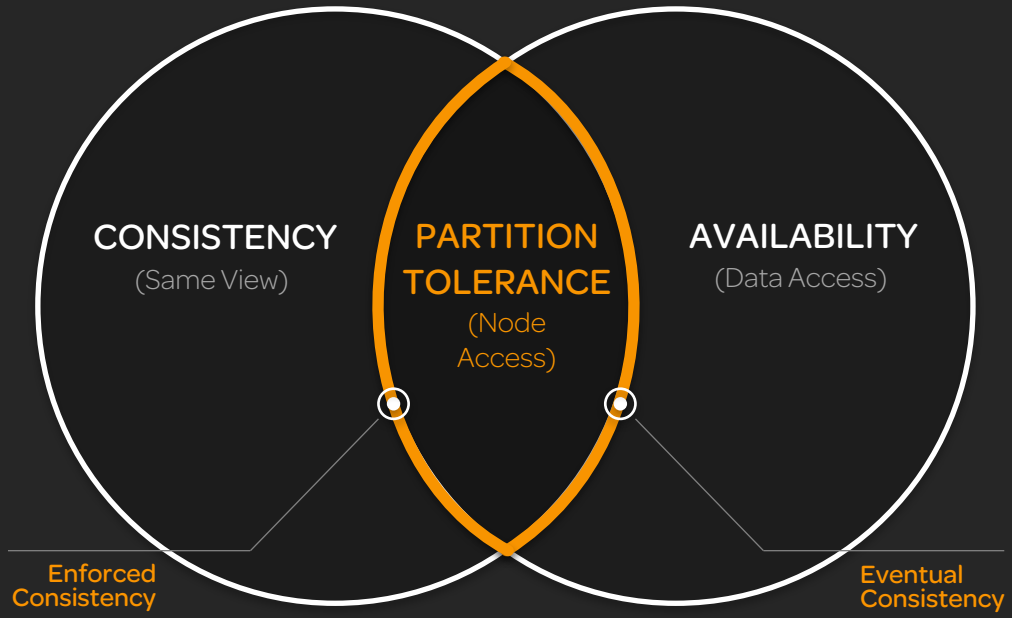
Provides a high degree of availability, throughput, and performance with its scale-out architecture.

CAP THEOREM

Distributed systems can not simultaneously guarantee:

- Consistency
- Availability
- Partition tolerance

Swift chooses Availability and Partition Tolerance over Consistency



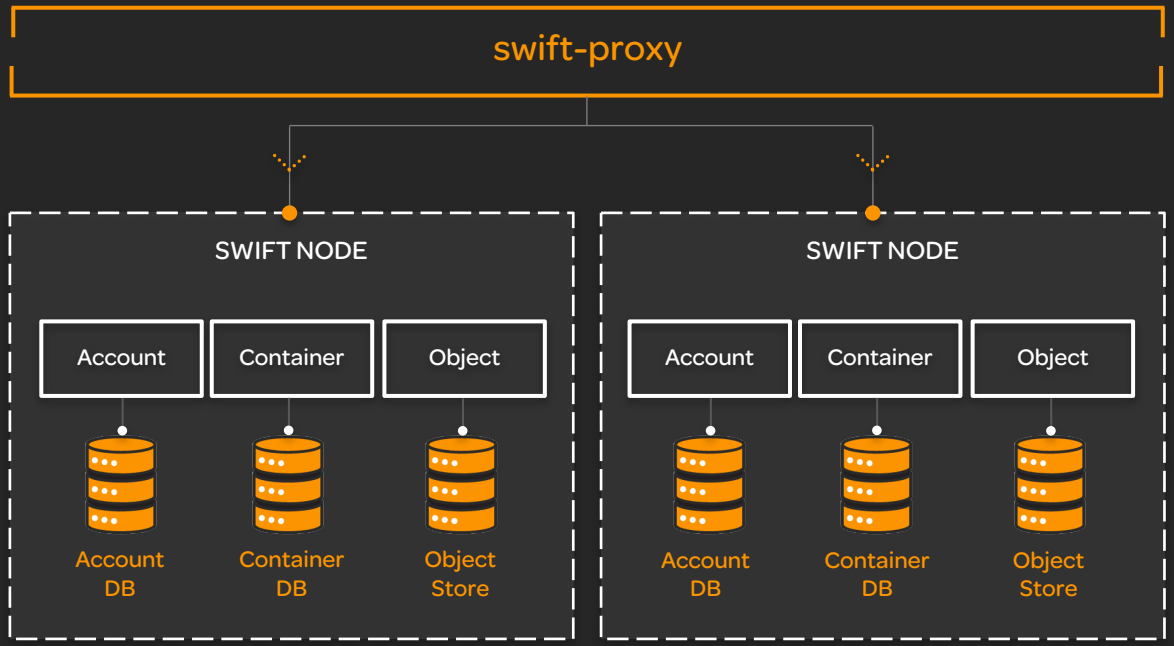
FEATURES AND BENEFITS

- Leverages commodity hardware
- Self-healing, reliable, data redundancy that protects from failures
- Unlimited storage
- Scale-out architecture
- Built-in replication
- Can easily add capacity without RAID resizing as it is not required
- Drive auditing
- Ability to set expiration on objects

CHARACTERISTICS

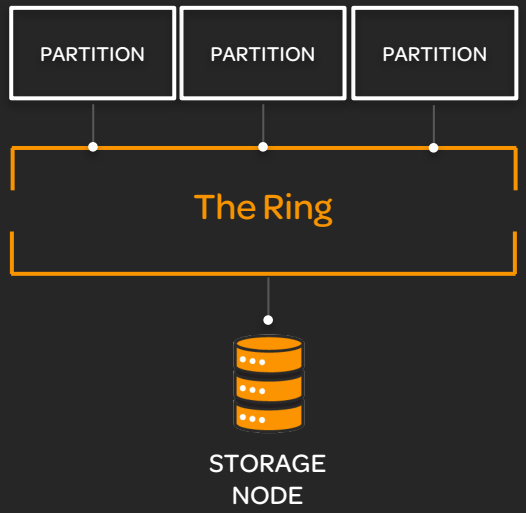
- All objects stored in Object Storage have a URL
- All objects stored are replicated 3X in as-unique-as-possible zones, which can be defined as a group of drives, a node, a rack, and so on
- All objects have their own metadata
- Developers interact with the object storage system through a RESTful HTTP API
- Object data can be located anywhere in the cluster
- The cluster scales by adding additional nodes without sacrificing performance, which allows a more cost-effective linear storage expansion than fork-lift upgrades
- Data doesn't have to be migrated to an entirely new storage system
- New nodes can be added to the cluster without downtime
- Failed nodes and disks can be swapped out without downtime
- It runs on industry-standard hardware, such as DELL, HP, and Supermicro

SWIFT OVERVIEW



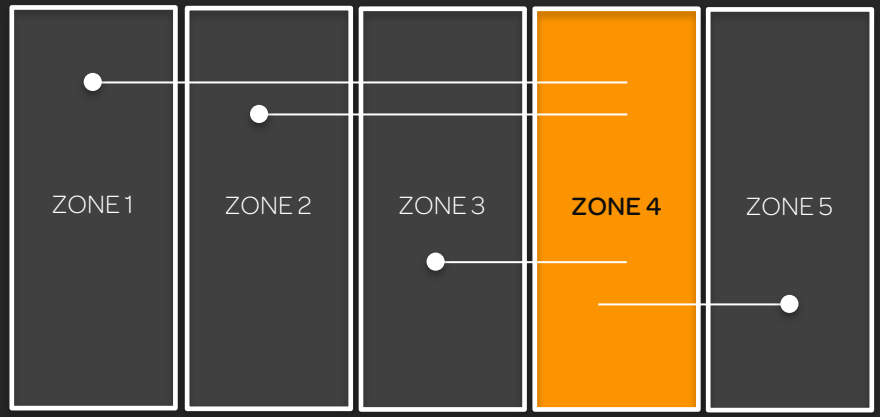
RINGS

- A ring represents a mapping between the names of entities stored on a disk and their physical locations
- The ring maintains this mapping using zones, devices, partitions, and replicas:
  - Each partition in the ring is replicated, by default, three times across the cluster, and partition locations are stored in the mapping maintained by the ring
  - The ring is also responsible for determining which devices are used for handoffs in failure scenarios



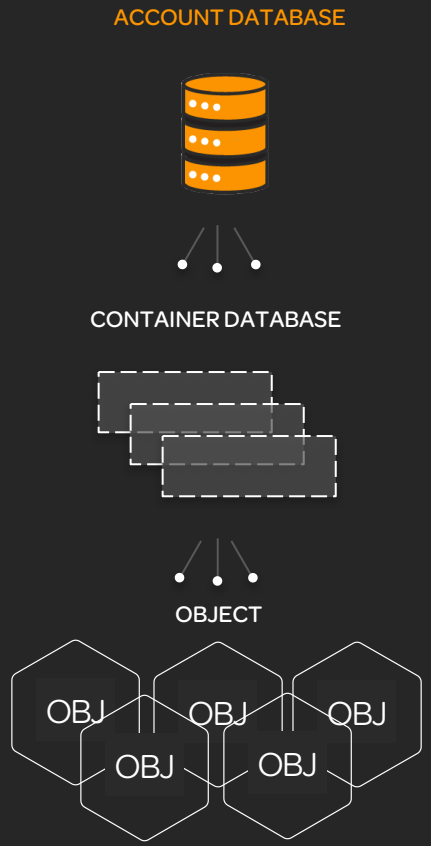
ZONES

- Object Storage allows configuring zones in order to isolate failure boundaries:
  - Each data replica resides in a separate zone, if possible
  - At the smallest level, a zone can be a single drive or a grouping of a few drives
  - If there were five object storage servers, then each server would represent its own zone
- The goal of zones is to allow the cluster to tolerate significant outages of storage servers without losing all replicas of the data
- Everything in Object Storage is stored, by default, three times



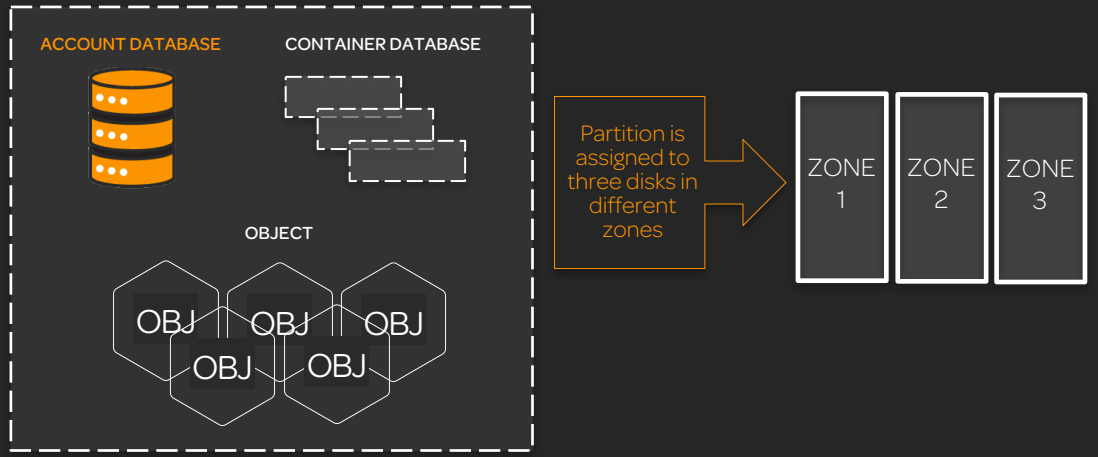
ACCOUNTS AND CONTAINERS

- Each account and container is an individual SQLite database that is distributed across the cluster
- An account database contains the list of containers in that account
- A container database contains the list of objects in that container



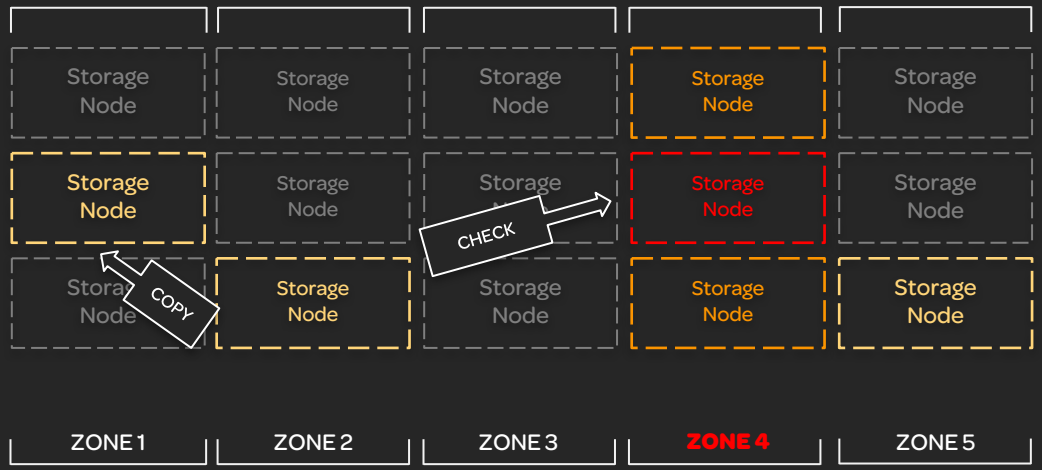
PARTITIONS

- A partition is a collection of stored data, including account databases, and objects:
  - Partitions are core to the replications system
- System replicators and object uploads/downloads operate on partitions:
  - As the system scales up, its behavior continues to be predictable because of the number of partitions is a fixed number
  - Implementing a partition is conceptually simple; a partition is just a directory sitting on a disk with a corresponding hash table of what it contains



REPLICATORS

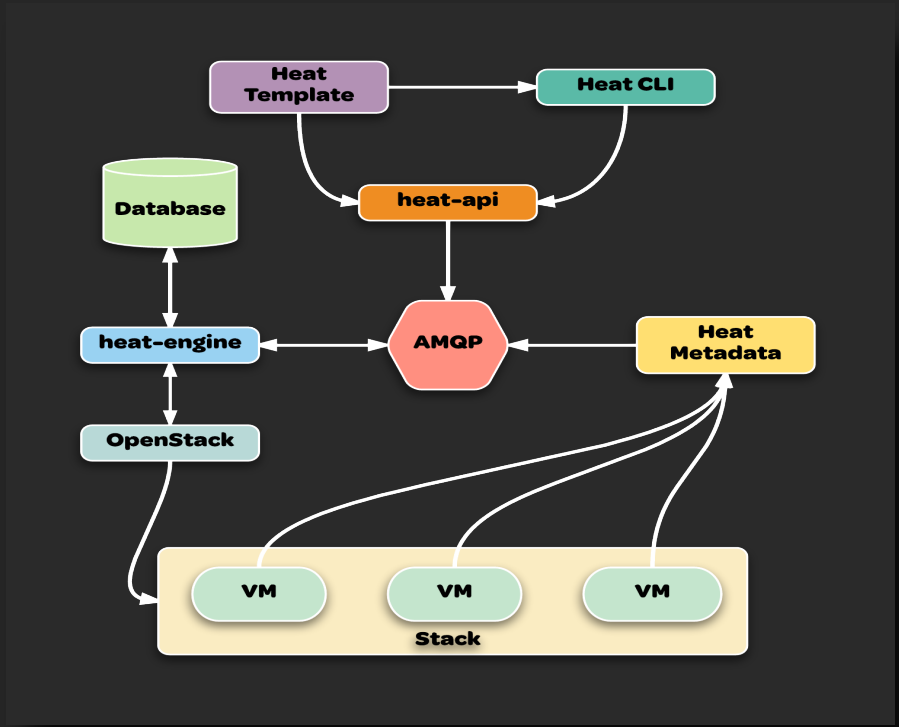
- In order to ensure that there are three copies of the data everywhere, replicators continuously examine each partition
- The replicator knows if the replication needs to take place by examining hashes
- This is where partitions come in handy
- The cluster eventually has a consistent behavior where the newest data has priority



Deploy and Manage OpenStack Pike on Ubuntu

Orchestration Overview

Heat Architecture



How Heat Works

A Heat template describes the infrastructure for a cloud application in a text file that is readable and writable by humans and can be checked into version control and used with, for example, Git

Infrastructure resources that can be described include:

Servers
Floating IPs
Volumes
Security groups
Users

How Heat Works

Heat also provides an auto scaling service that integrates with Ceilometer, so you can include a scaling group as a resource within a template.

Templates can also specify the relationships between resources (e.g. this volume is connected to this server). This enables Heat to call out to the OpenStack APIs to create all of your infrastructure in the correct order to completely launch your application.

Heat manages the whole lifecycle of the application

When you need to change your infrastructure, simply modify the template and use is to update your existing stack.
Heat knows how to make the necessary changes. It will delete all of the resources when you are finished with the application, too.
Heat primarily manages infrastructure, but the templates integrate well with software configuration management tools such as Puppet and Chef.

Deploy and Manage OpenStack Pike on Ubuntu

What's Next?

What's Next

Certification:

OpenStack Foundation Certified OpenStack Administrator
OpenStack MCA100 – Associates Certification
Linux Academy Red Hat OpenStack Administrator Certification Prep Course

What's Next

- Other Areas to Explore:
  - AWS Essentials
  - Linux Academy Red Hat Certified Systems Administrator Prep Course
  - DevOps Essentials
  - Docker Quick Start