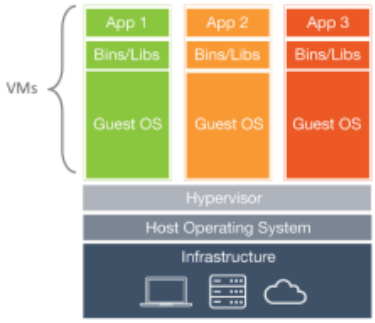
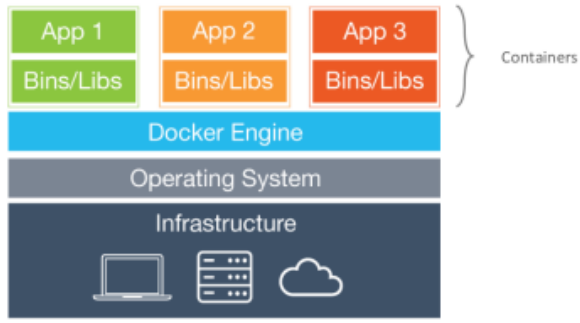
# Kubernetes and Openstack

**Boris Teabe**
boris.teabe@inp-toulouse.fr


In this class we will discuss about kubernetes and openstack

# Virtual Machines vs. Containers




- **Virtual machines**
  - Each virtual machine (VM) includes the app, the necessary binaries and libraries and an entire guest operating system

- **Containers**
  - Containers include the app & all of its dependencies, but share the kernel with other containers.
  - Run as an isolated process in userspaceon the hostOS
  - Not tied to any specific infrastructure–containers run on any computer, infrastructure and cloud.

Reminder of last class

# Ecosystem

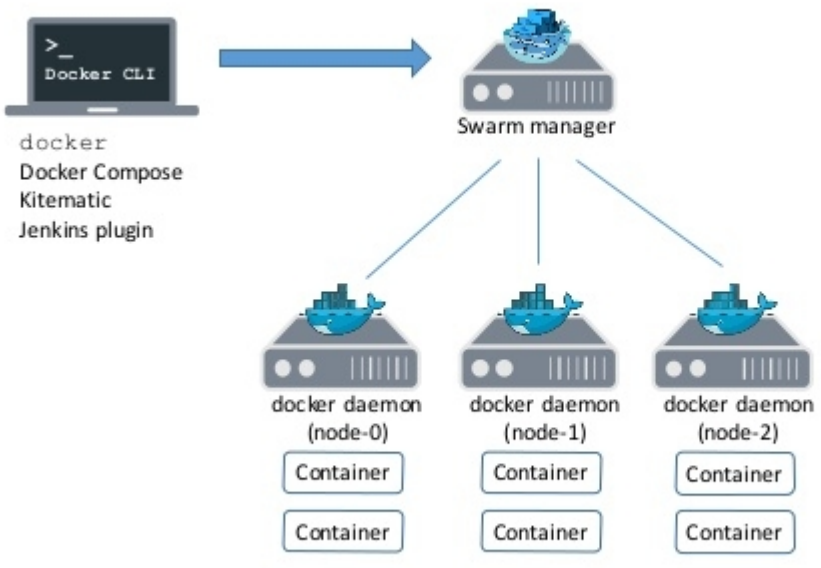
- Docker machine
  - Allow to easily install Docker hosts in a network
- Docker compose
  - Allow defining and running multi-container applications
- Kitematic
  - Graphical interface for the administration of a Docker host
- Docker swarm
  - Allow the management of a cluster of Docker hosts (container replication, load-balancer, elasticity, recovery …)
- Kubernetes

Reminder of last class

# Docker Swarm

- Native solution  of Docker for clustering
  - Turn  a cluster into a unique virtual host
  - Use the same API
- Allow to manage and schedule containers on a cluster

A Docker Swarm is a group of either physical or virtual machines that are running the Docker application and that have been configured to join together in a cluster.  achines that have joined the cluster are referred to as nodes.



# Docker Swarm deploy

- Run a Swarm image in a container
- Or install binary on your host
- Pull a "Docker Swarm" image
- Configure the Swarm Manager and the Workers (physical nodes able to host containers)
  - Open a TCP port on each node to communicate with the Swarm manager.
  - Install Docker on each node

Three way to deploy docker swarm master
- Either you run a docker image containing the master
- Or install the binary on you host.

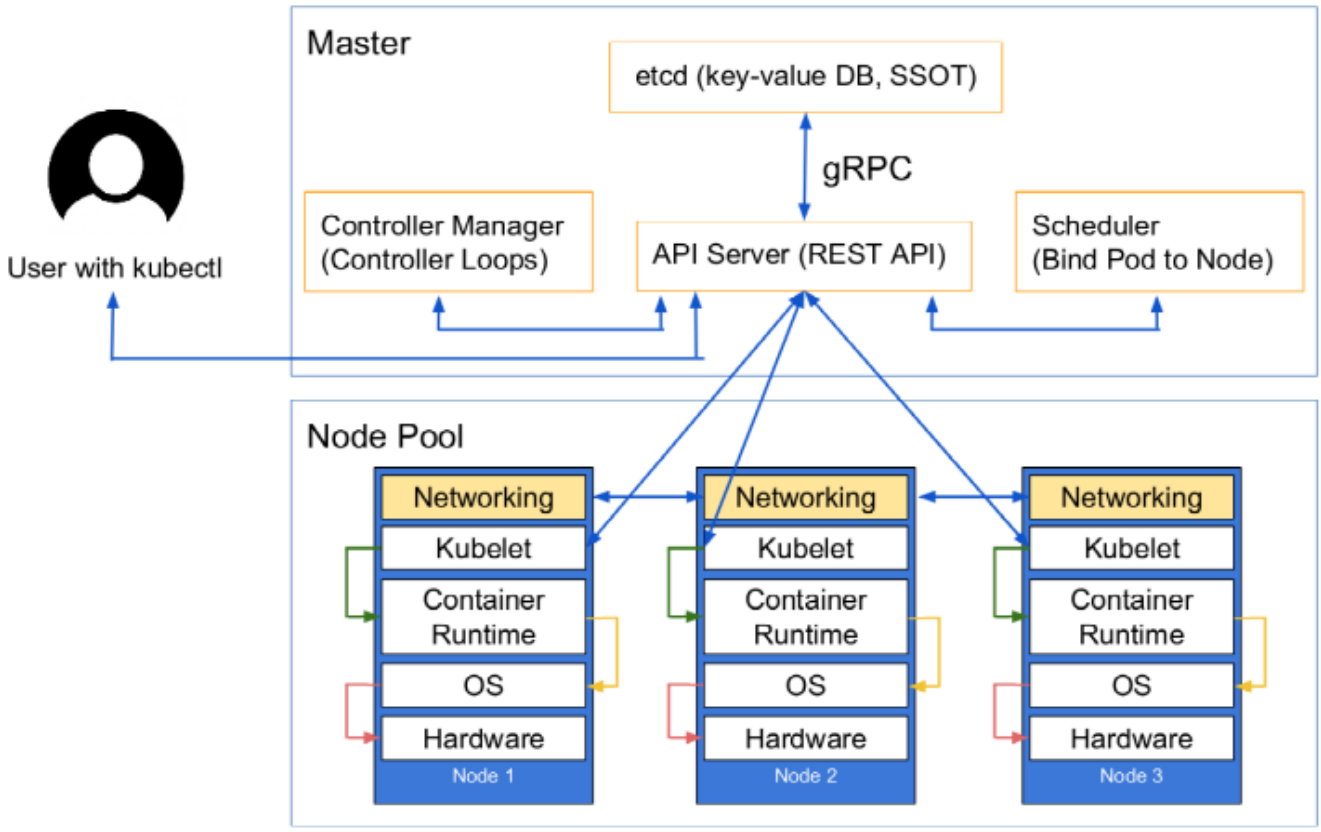
After that, you have to configure the swarm manager and add the nodes.

# What is Kubernetes ?

- A container orchestration system.
- Abstraction of the physical infrastructure thanks to the concept of "Node" Principle
- Kubernetes abstracts the thousands of nodes in a cluster and provides industry methods to manage applications. administrator describes and declares the "desired state", and Kubernetes converts the "current state" to "desired state".

# Users of Kubernetes ?

# Architecture of Kubernetes




Master is called the control plane and it has 3 further things, namely API Server, Scheduler, Controller Manager and the etcd.

API Server enables all the communication b/w API, we are going to talk to Kube API Server only. It takes the request and sends it to other services.

Etcd stores all the information in and other services also reads and store the information in it.

Kube Scheduler picks up the container and puts it on the right node based on different factors.

Manager Controller is responsible for checking the status of the node.

Kubelet is the agent that listens to the request of master and is going to do all the heavy lifting.

# Some concepts of Kubernetes

- **Pods** : is a group of one or more containers, with shared storage/ network, and a specification for how to run the containers. It represents an application in kubernetes

- **Deployment** : provides declarative updates for Pods and ReplicaSets. Describes a desired state, and the Deployment controller changes the actual state to the desired state at a controlled rate.

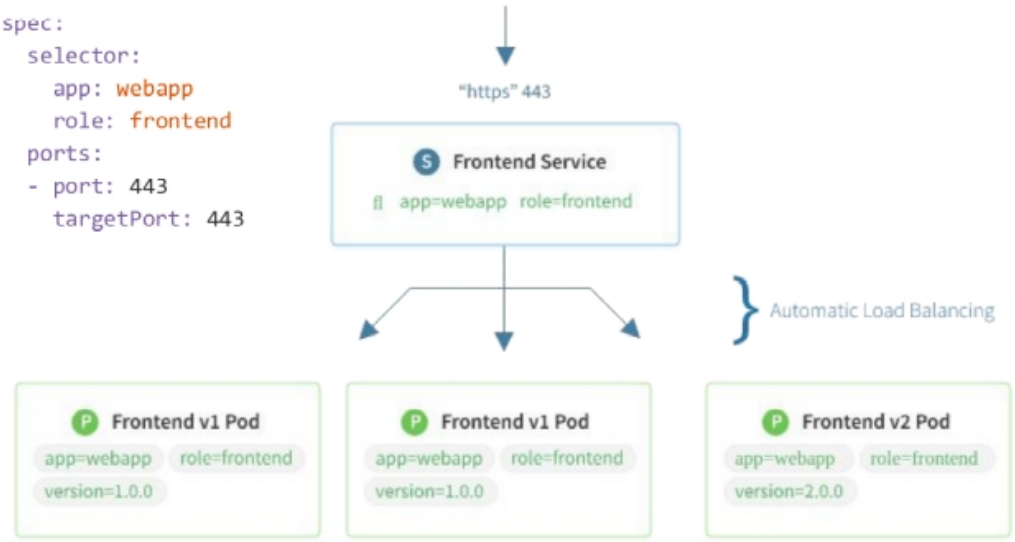
- **Services** : An abstract way to expose an application running on a set of Pods as a network service.

- **Namespace** : Kubernetes supports multiple virtual clusters backed by the same physical cluster. These virtual clusters are called namespaces.

# Kubernetes « manifest »

```yaml
apiVersion: apps/v1beta2
kind: Deployment
metadata:
  labels:
    app: webapp
    role: frontend
  name: web-frontend
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: webapp
        role: frontend
    spec:
      containers:
      - image: nginx:1.13.1
        name: nginx
        ports:
        - containerPort: 443
          name: https
```

```yaml
apiVersion: v1
kind: Service
metadata:
  name: web-frontend
spec:
  selector:
    app: webapp
    role: frontend
  ports:
  - port: 443
    targetPort: 443
```



# Some Kubernetes functionalities

- **Self-healing :**Kubernetes restarts containers that fail, replaces containers, kills containers that don't respond to your user-defined health check, and doesn't advertise them to clients until they are ready to serve.

- **Automatic binpacking :** Kubernetes allows you to specify how much CPU and memory (RAM) each container needs. When containers have resource requests specified, Kubernetes can make better decisions to manage the resources for containers.

- **Automated rollouts and rollbacks** : You can describe the desired state for your deployed containers using Kubernetes, and it can change the actual state to the desired state at a controlled rate. For example, you can automate Kubernetes to create new containers for your deployment, remove existing containers and adopt all their resources to the new container.

# Some Kubernetes functionalities

- **Service Discovery and Load Balancing** : Kubernetes can expose a container using the DNS name or using their own IP address. If traffic to a container is high, Kubernetes is able to load balance and distribute the network traffic so that the deployment is stable.

- **Storage Orchestration**: Kubernetes allows you to automatically mount a storage system of your choice, such as local storages, public cloud providers, and more.

# Docker Swarm vs Kubernetes

| | Docker Swarm | Kubernetes |
|---|---|---|
| **Application definition** | Default Docker cli and docker compose | use pods |
| **Scalability** | Fast deployment of containers | Strong garantee on scalability |
| **High Availability** | Docker Swarm also offers high availability | High availability by tolerating the failure of application |
| **Container Setup** | Based on Docker cli, can use Docker compose | Kubernetes utilizes its own YAML |
| **Load Balancing** | Swarm mode consists of a DNS element that can be utilized for distributing incoming requests to a service name. | Pods are exposed via service, which can be utilized as a load balancer within the cluster |

**Kubernetes supports higher demands with more complexity while Docker Swarm offers a simple solution that is quick to get started with. Docker Swarm has been quite popular among developers who prefer fast deployments and simplicity. Simultaneously, Kubernetes is utilized in production environments by various high profile internet firms running popular services.**

# OpenStack Cloud

- What is OpenStack ?

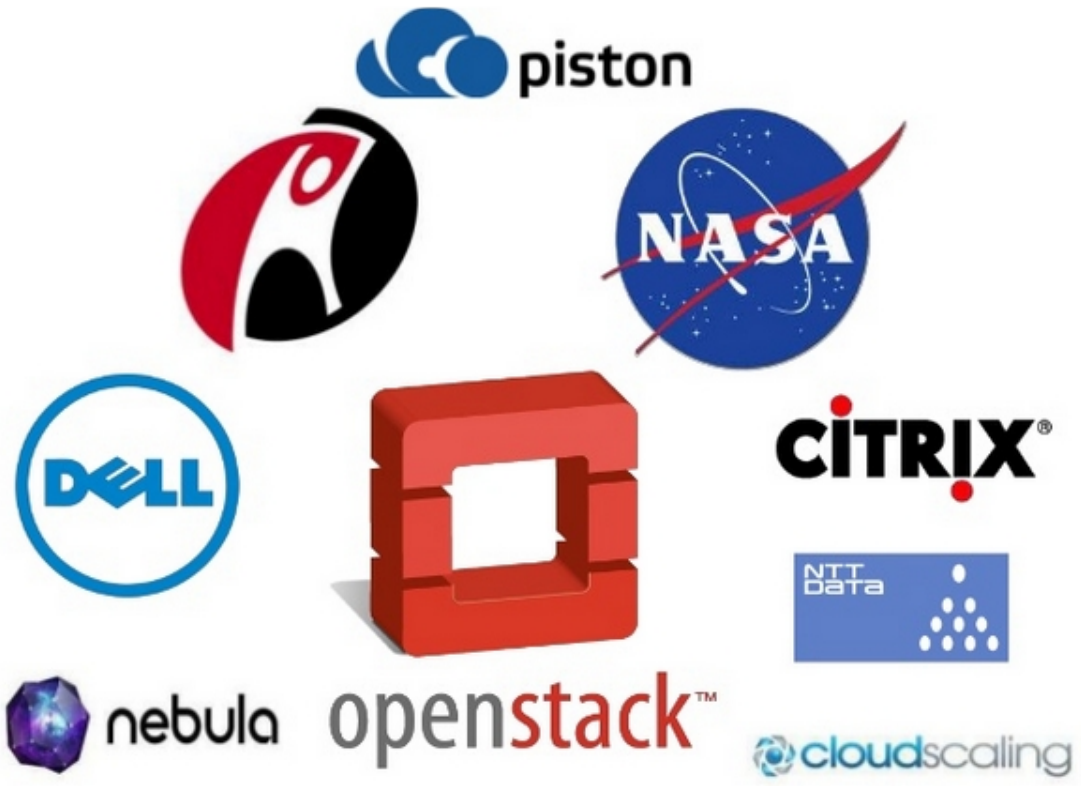

« OpenStack is an infrastructure as a Service which is know as a **Cloud Operating System,** that takes resources such as compute, storage, network, virtualization technologies and **controls those resources at a data center level** »

# OpenStack Cloud

- At the beginning


As of 2012, it was managed by the OpenStack Foundation, a non-profit corporate entity established in September 2012 to promote OpenStack software and its community.

# OpenStack Cloud




More organiization were interested by the project.

# OpenStack Cloud


The OpenStack Foundation was established in 2012 as a non-profit corporation to help promote the use of OpenStack software as well as to provide support to the OpenStack community. Since then, over 500 companies have joined OpenStack, including HP, Intel, Google, Red Hat, and Oracle. Each company has contributed to the project, some by creating their own distributions of OpenStack to implement in existing products. The OpenStack Foundation manages the OpenStack project to this day.

# OpenStack Cloud

> 110 Companies

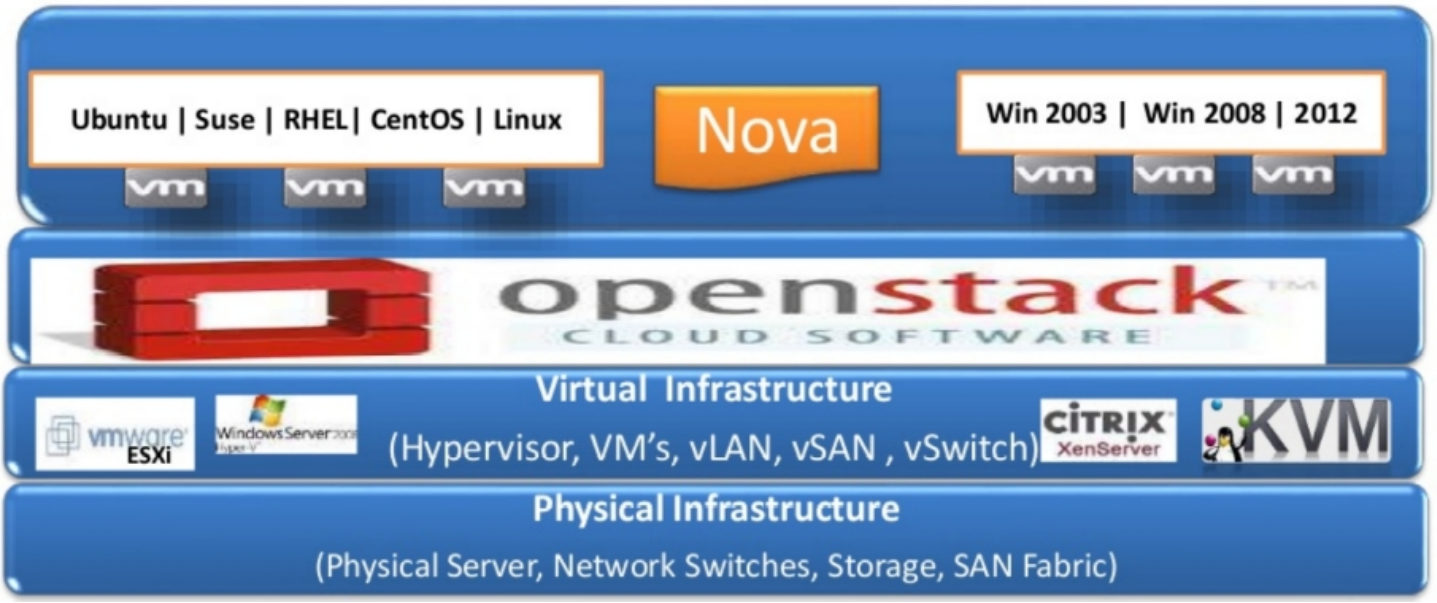
>1556 People

Hardware Vendors

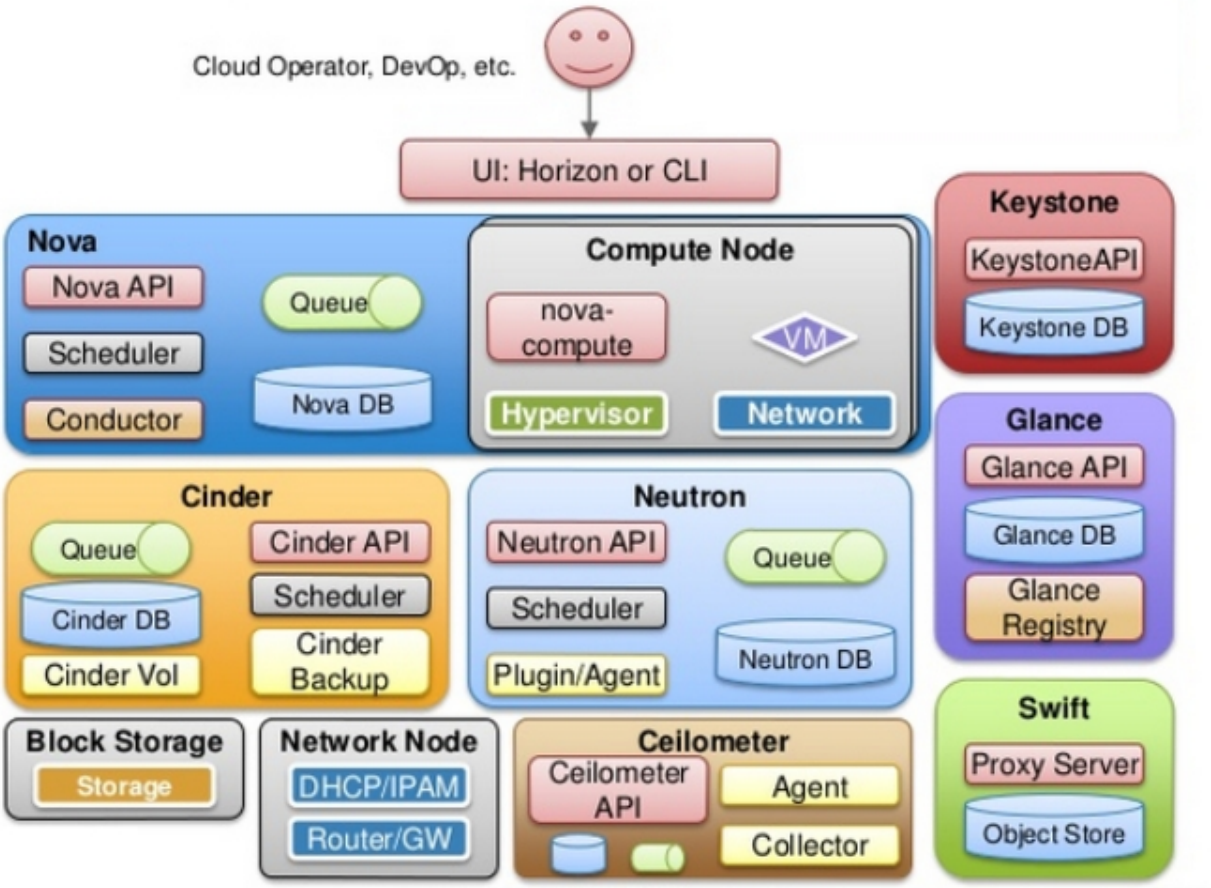
Service Providers

>50 M Venture Capital

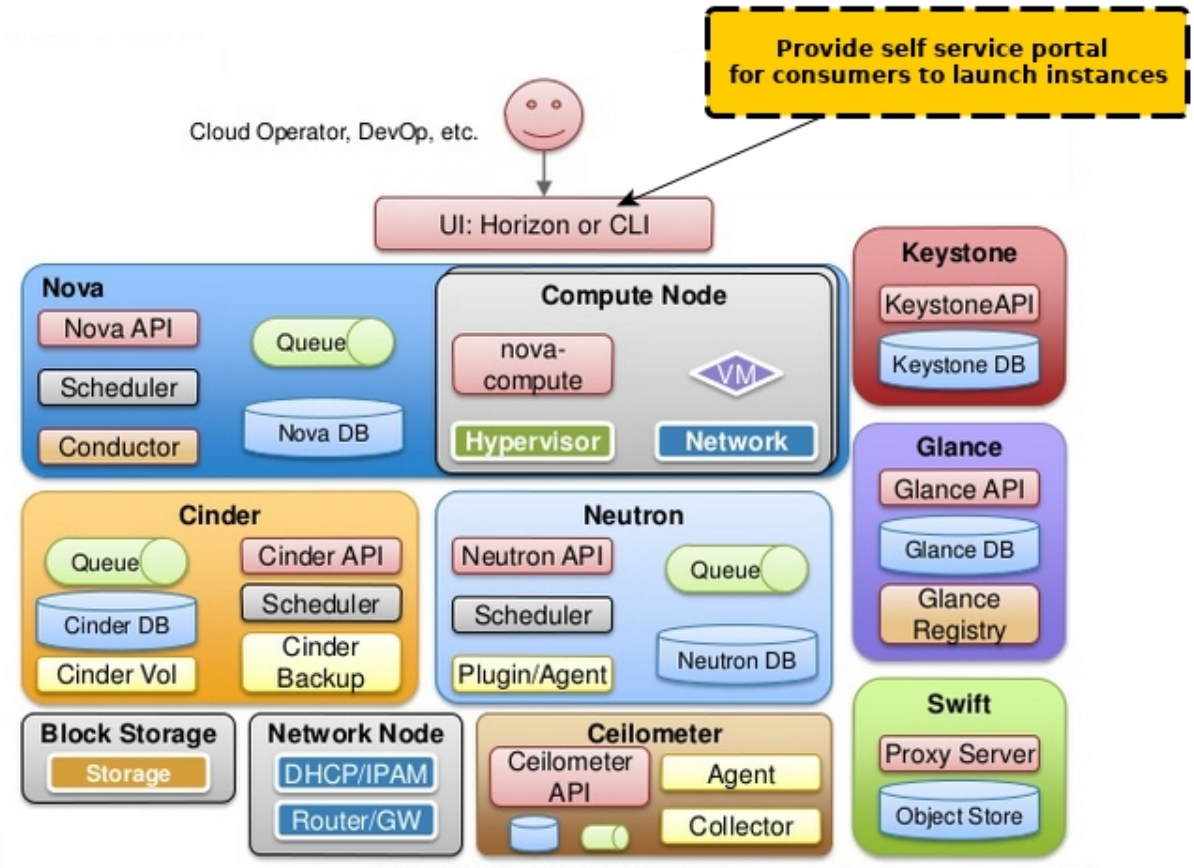
# OpenStack Cloud



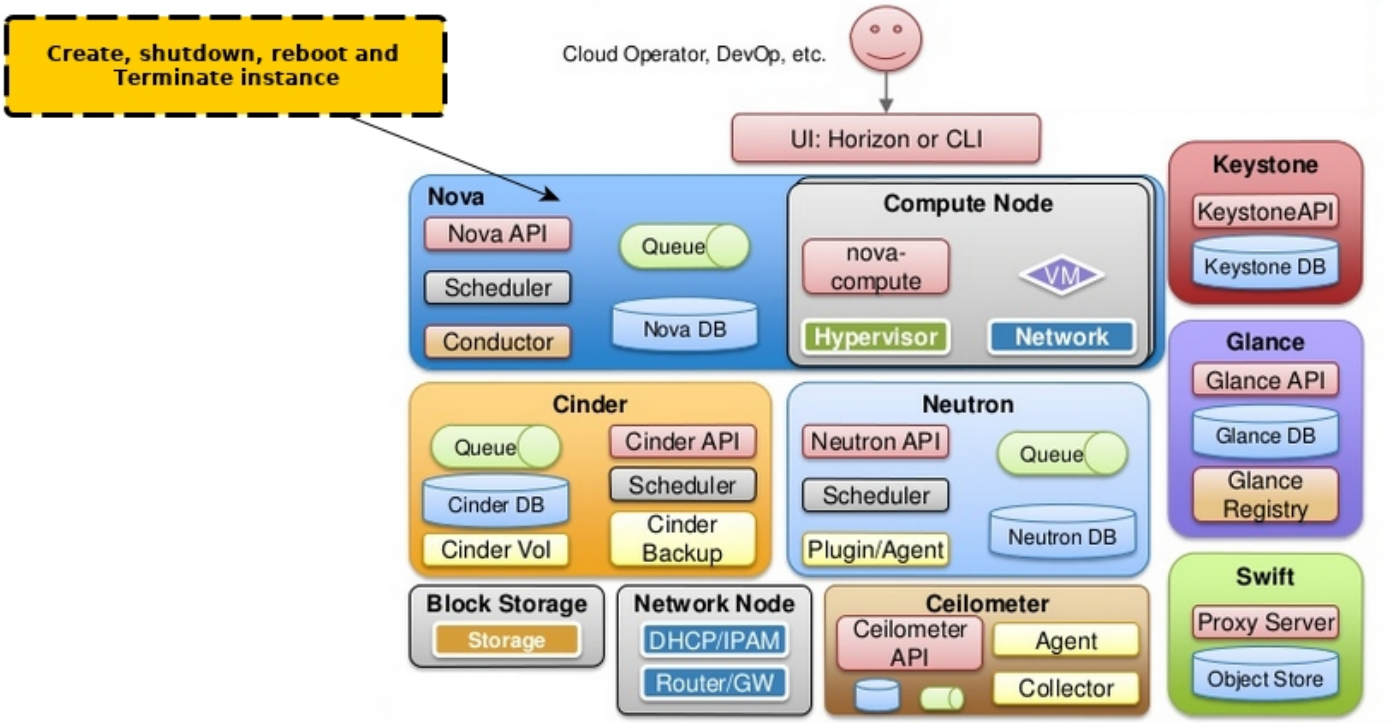
# OpenStack Cloud



# OpenStack Cloud

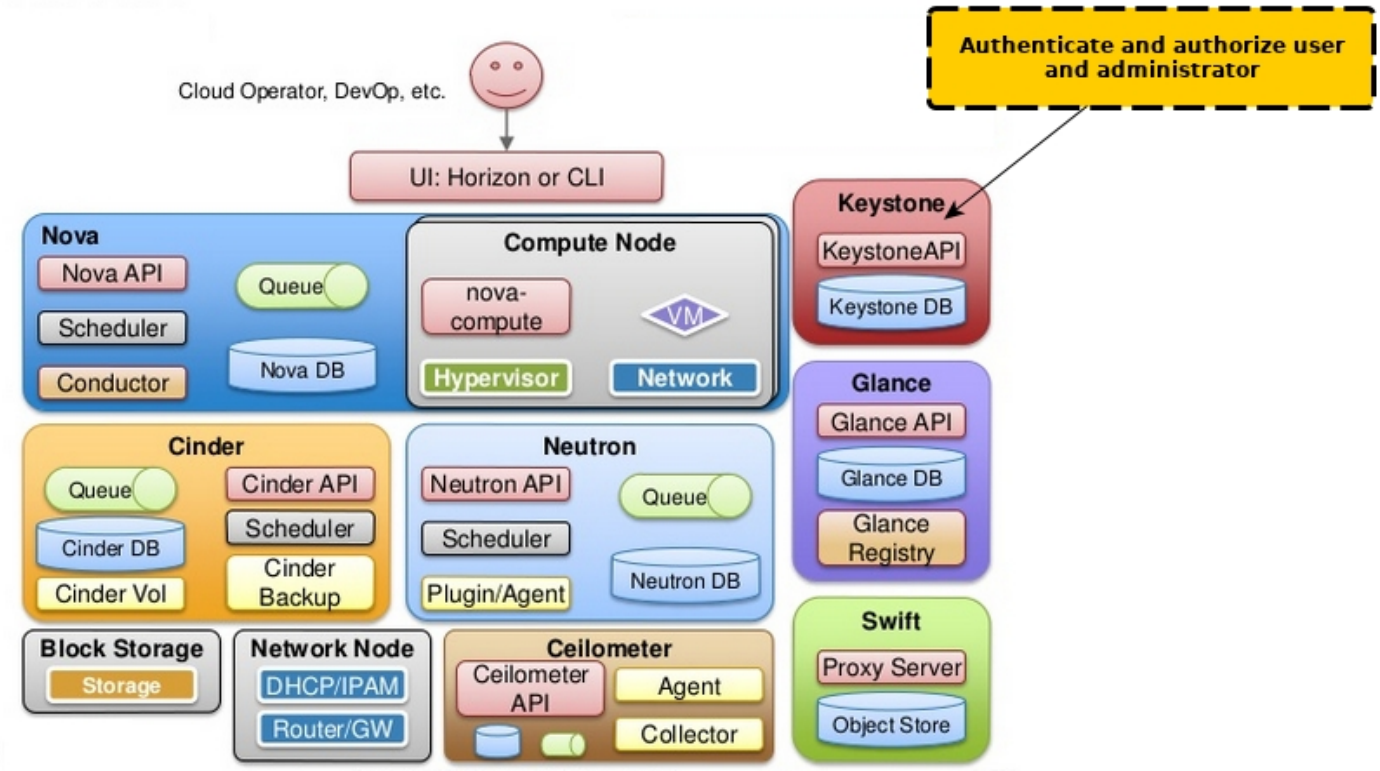


# OpenStack Cloud



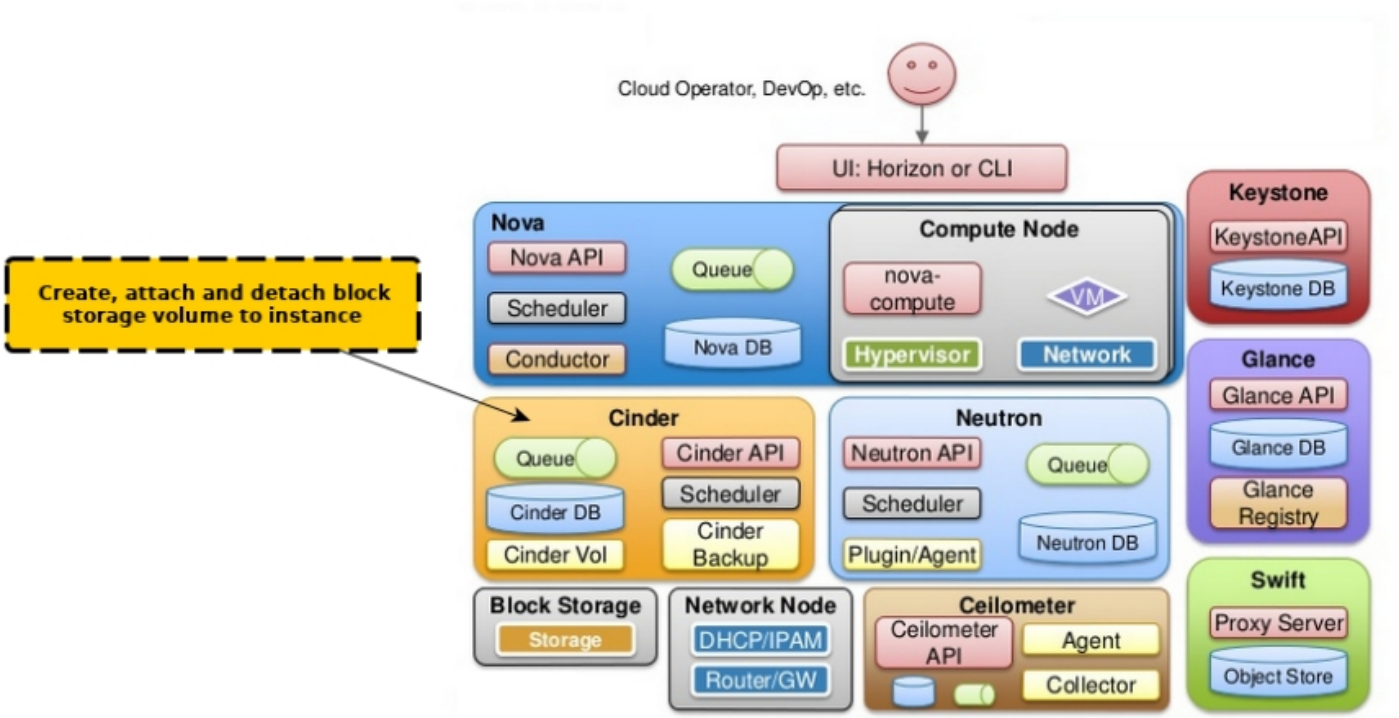

Nova is the OpenStack project that provides a way to provision compute instances (aka virtual servers). Nova supports creating virtual machines and has limited support for system containers. Nova runs as a set of daemons on top of existing Linux servers to provide that service.

# OpenStack Cloud



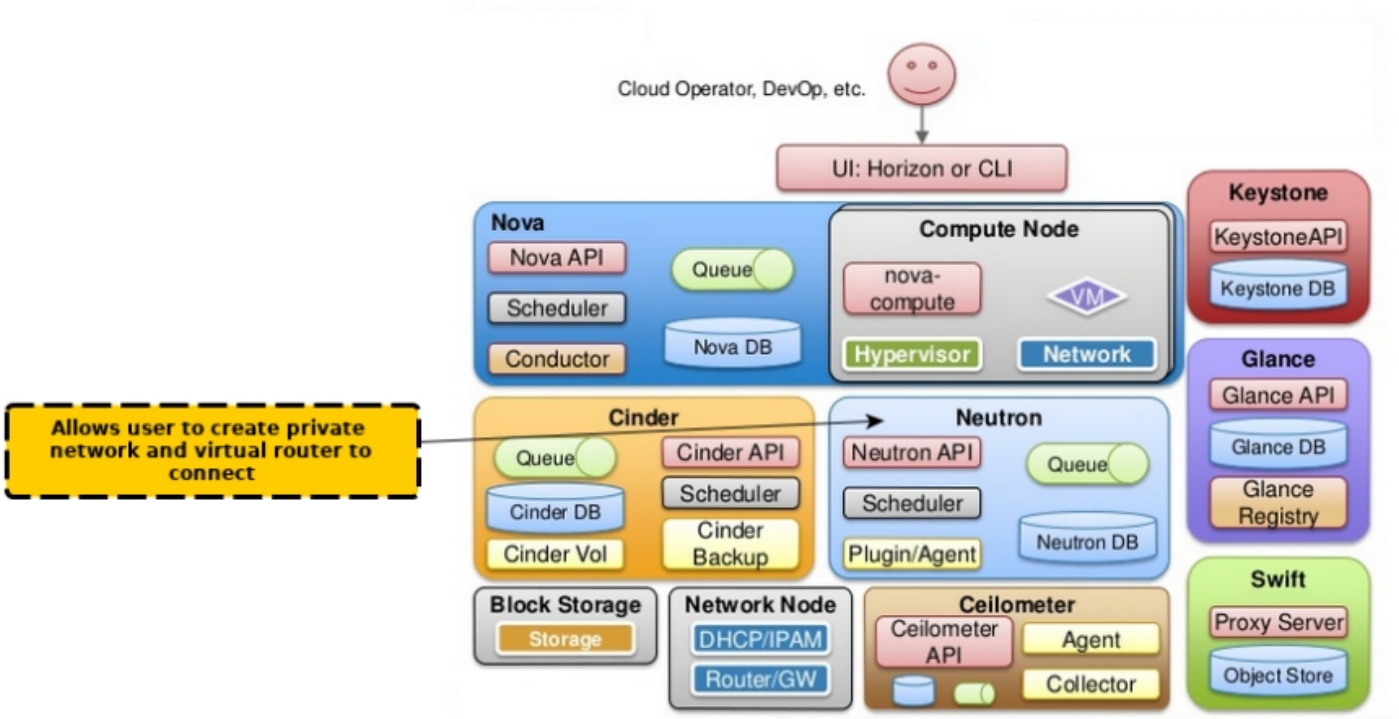

Keystone is an OpenStack service that provides API client authentication, service discovery, and distributed multi-tenant authorization by implementing OpenStack's Identity API.

# OpenStack Cloud



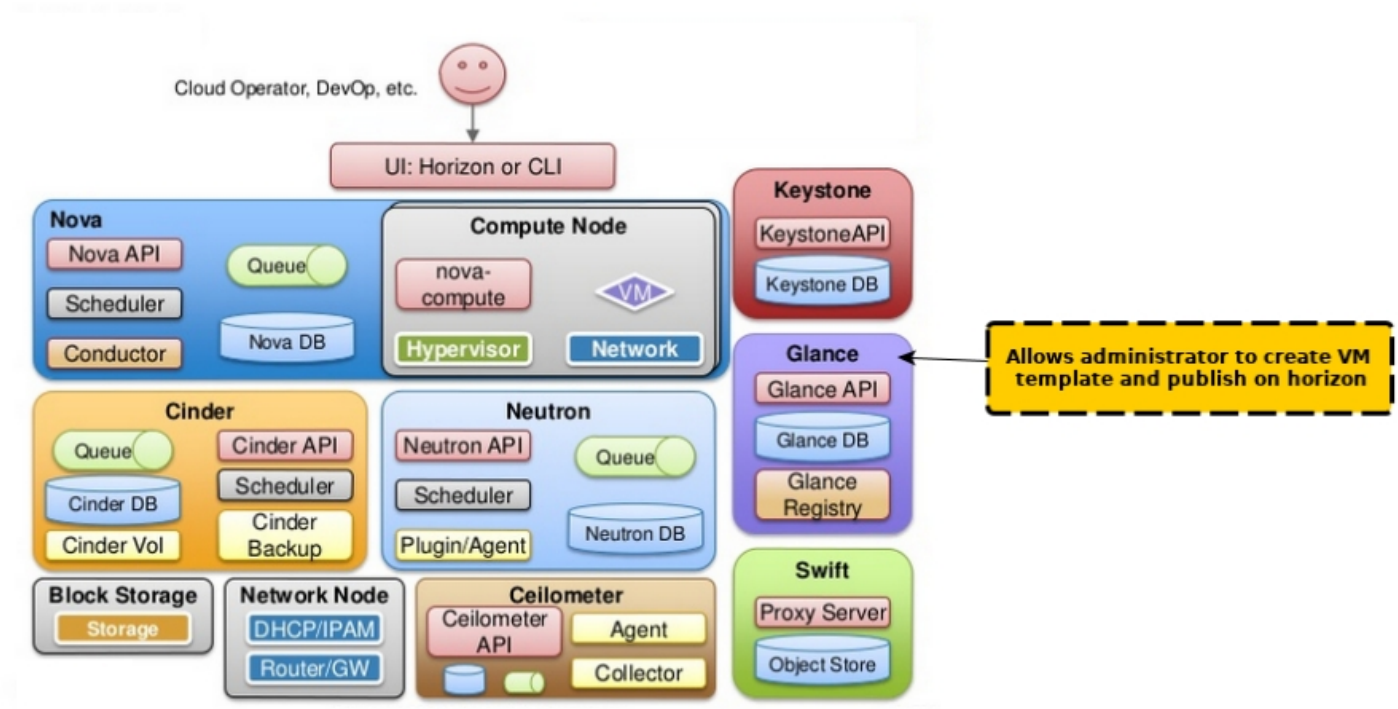

Cinder is a Block Storage service for OpenStack. It's designed to present storage resources to end users that can be consumed by the OpenStack Compute Project (Nova). The short description of Cinder is that it virtualizes the management of block storage devices and provides end users with a self service API to request and consume those resources without requiring any knowledge of where their storage is actually deployed or on what type of device.

# OpenStack Cloud



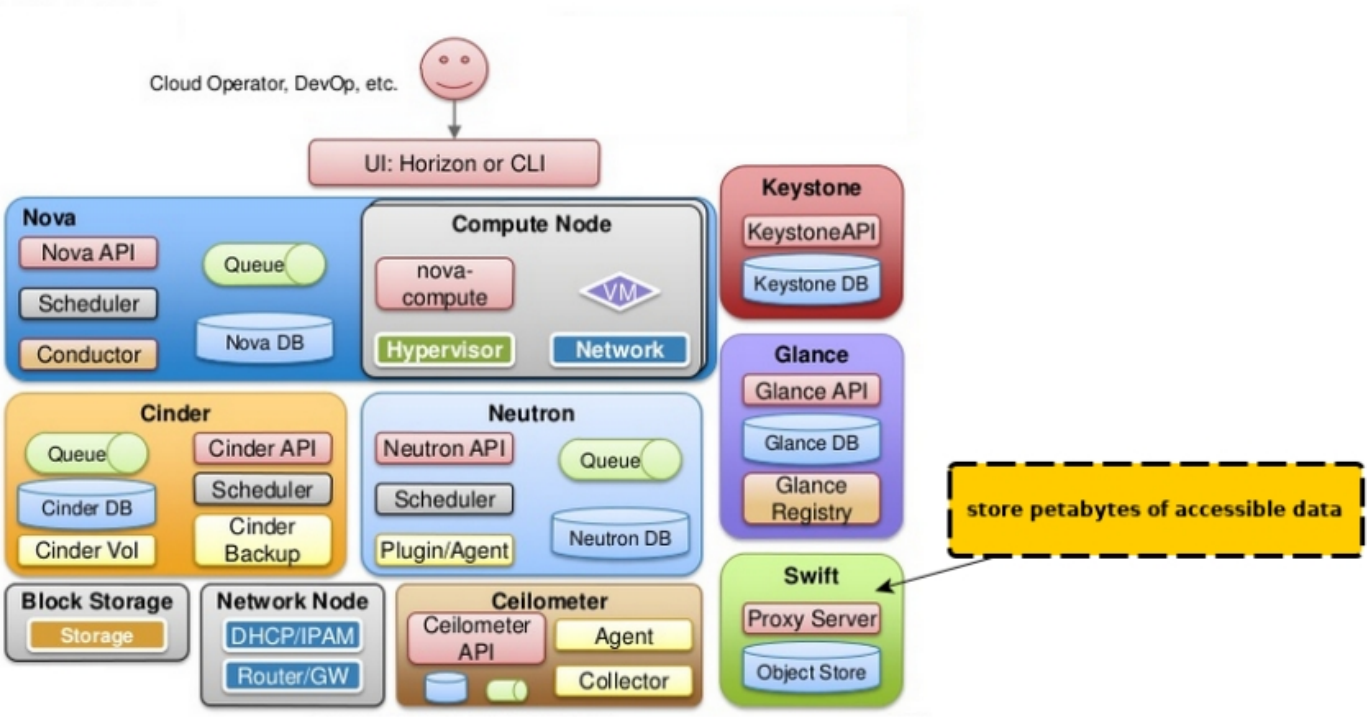

Neutron is an OpenStack project to provide "networking as a service" between interface devices (e.g., vNICs) managed by other Openstack services (e.g., nova).

# OpenStack Cloud




Glance image services include discovering, registering, and retrieving virtual machine (VM) images.

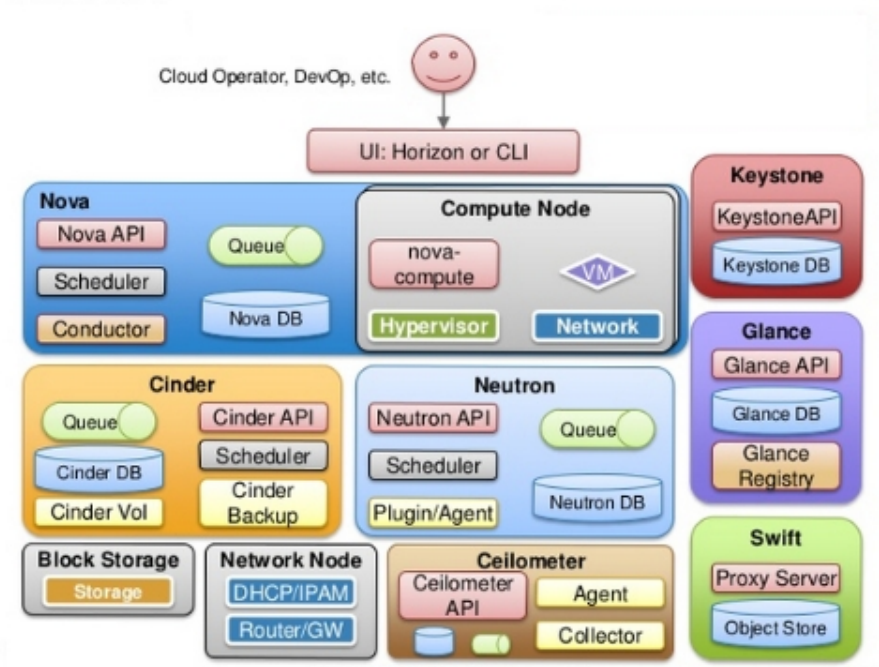
# OpenStack Cloud




Swift is a highly available, distributed, eventually consistent object/blob store. It can be used to store lots of data efficiently, safely, and cheaply.

# OpenStack Cloud

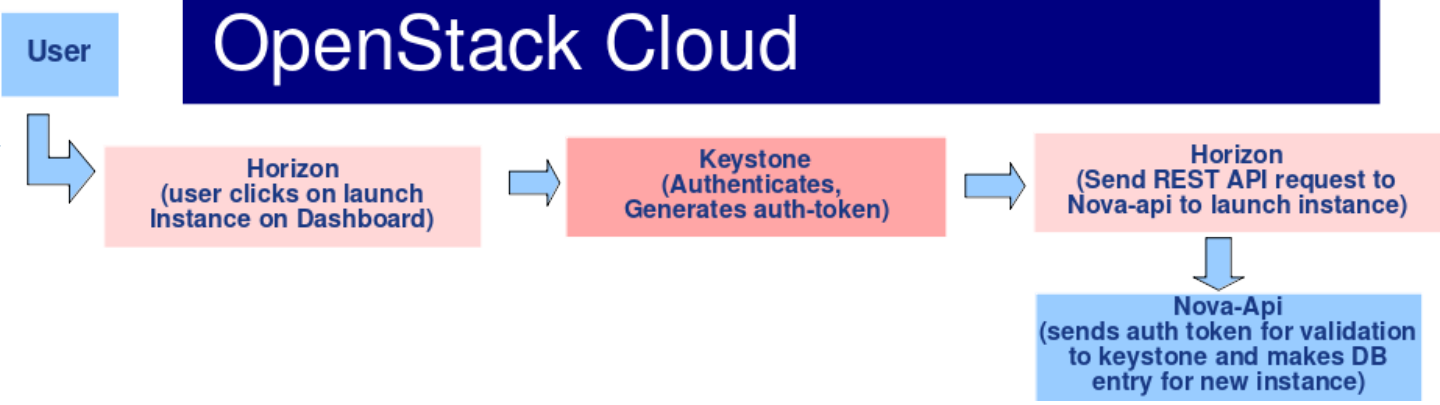
- VM provisionning
  - Is the most common and complex process in OpenStack
  - Involves interaction with most of OpenStack components

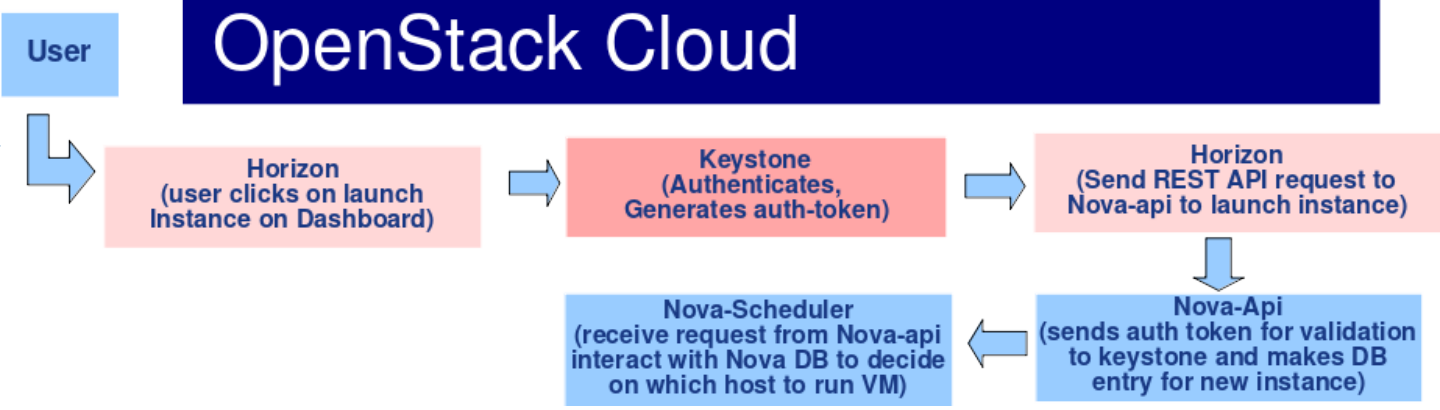


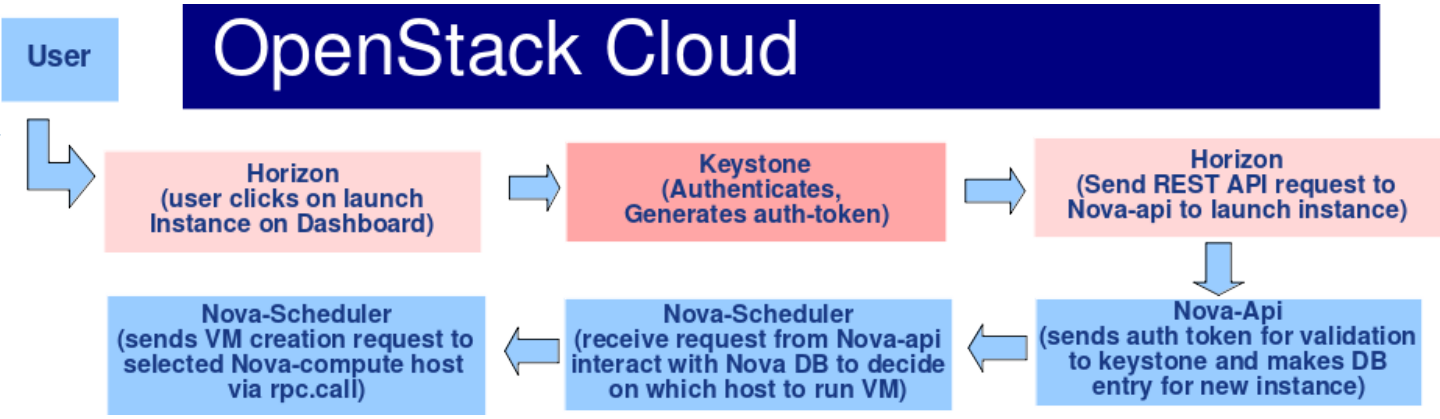

User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)

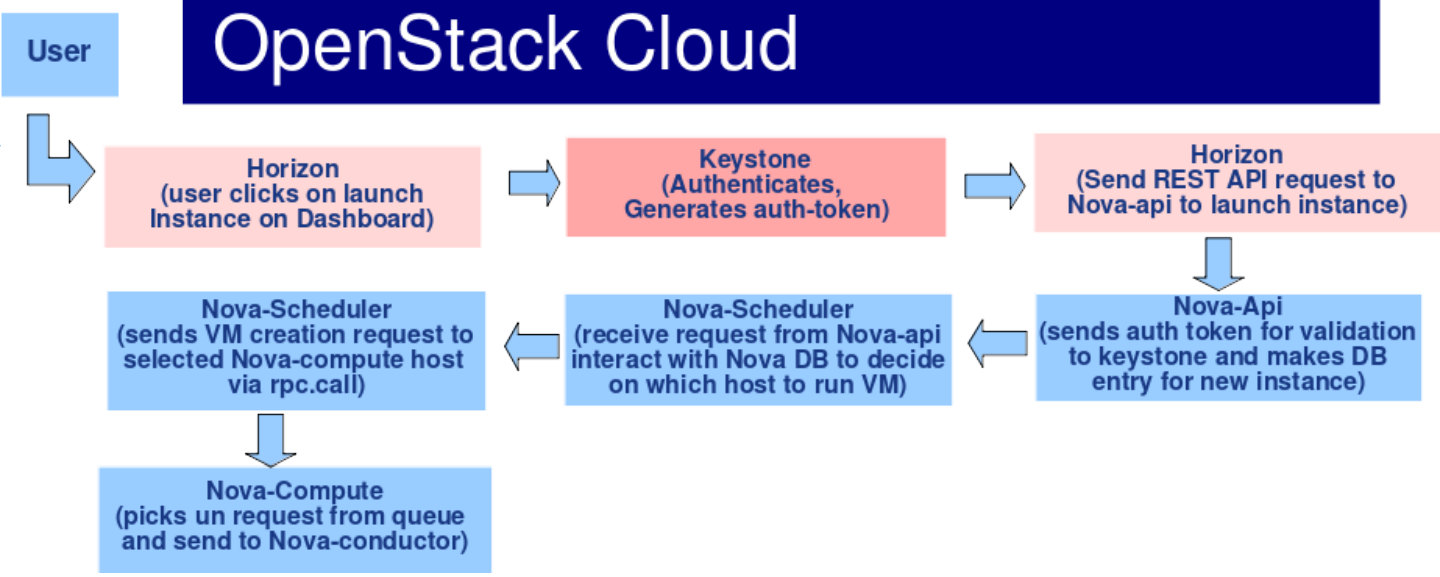

User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)
Keystone
(Authenticates,
Generates auth-token)

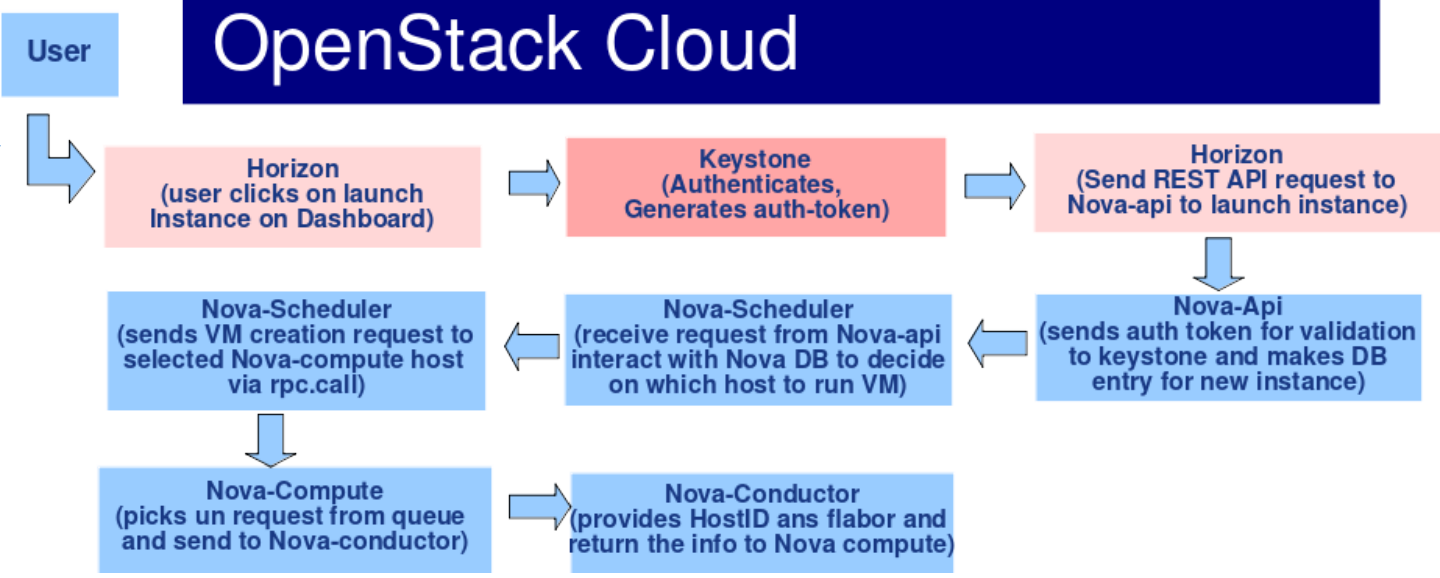

User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)
Keystone
(Authenticates,
Generates auth-token)
Horizon
(Send REST API request to
Nova-api to launch instance)

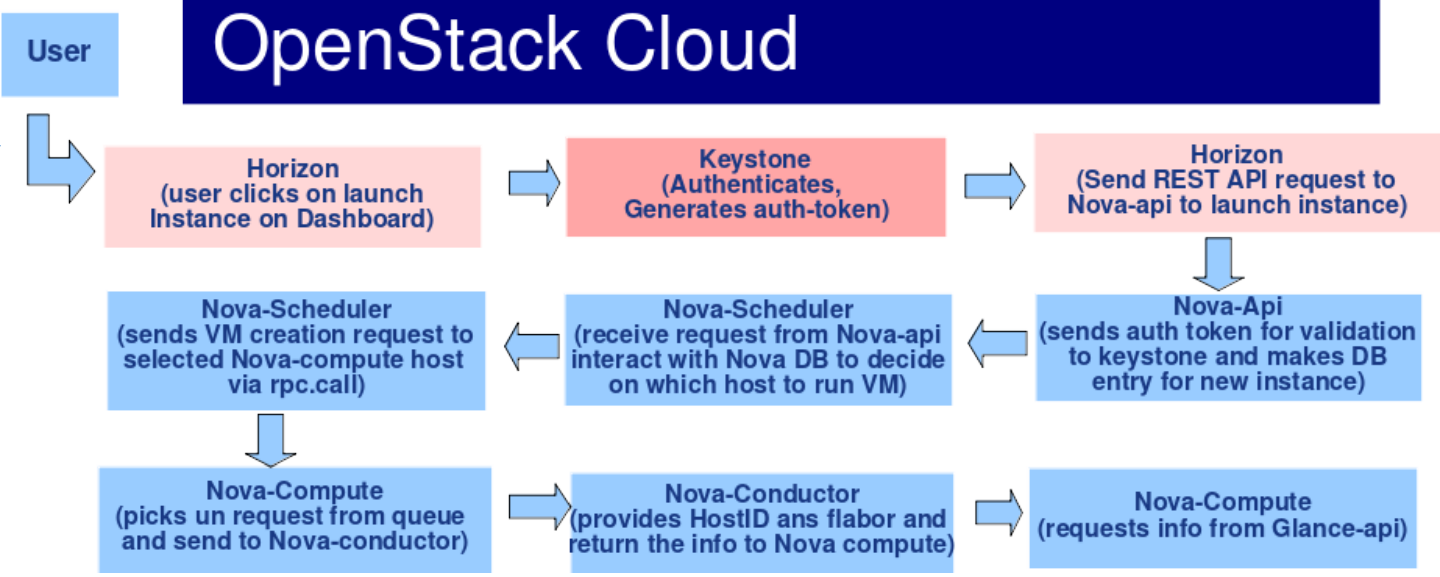

User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)
Keystone
(Authenticates,
Generates auth-token)
Horizon
(Send REST API request to
Nova-api to launch instance)
Nova-Api
(sends auth token for validation
to keystone and makes DB
entry for new instance)

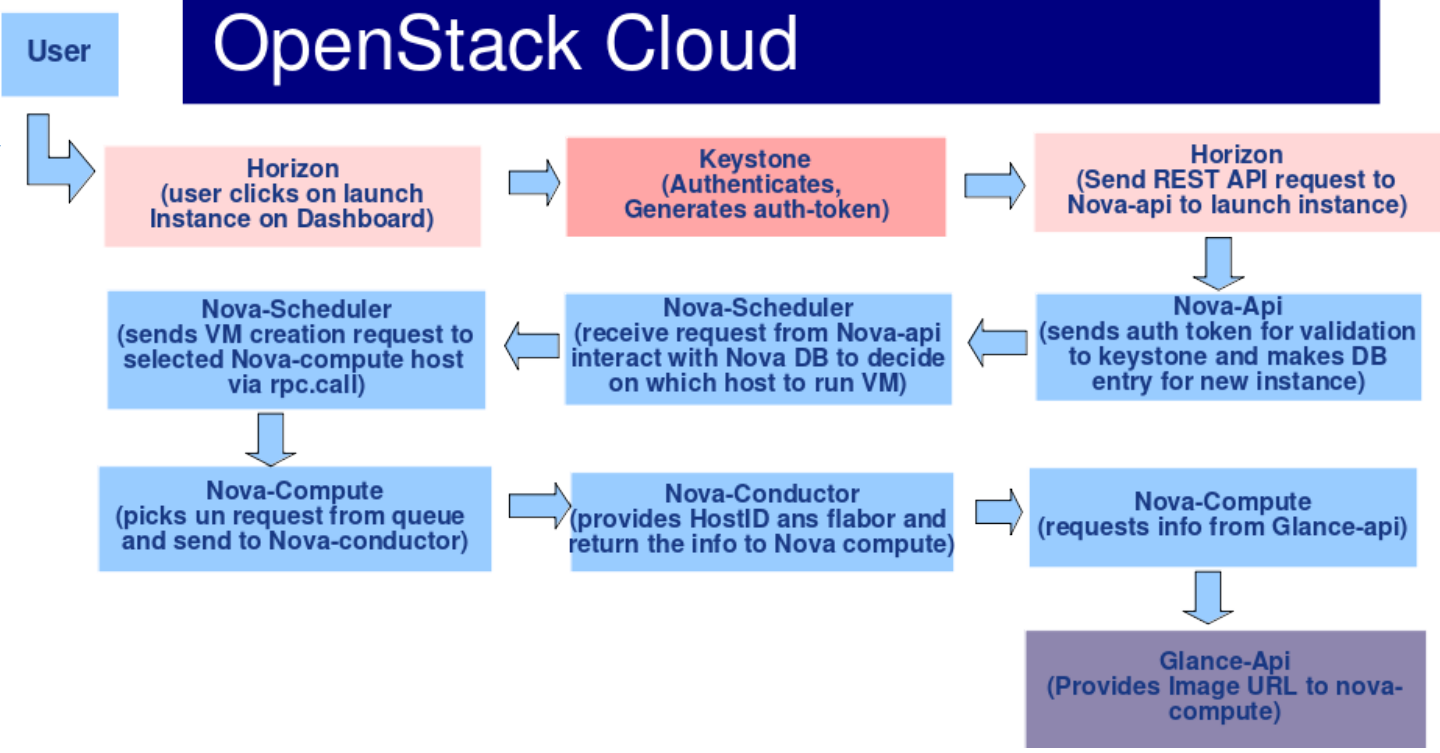

User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)
Keystone
(Authenticates,
Generates auth-token)
Horizon
(Send REST API request to
Nova-api to launch instance)
Nova-Api
(sends auth token for validation
to keystone and makes DB
entry for new instance)
Nova-Scheduler
(receive request from Nova-api
interact with Nova DB to decide
on which host to run VM)

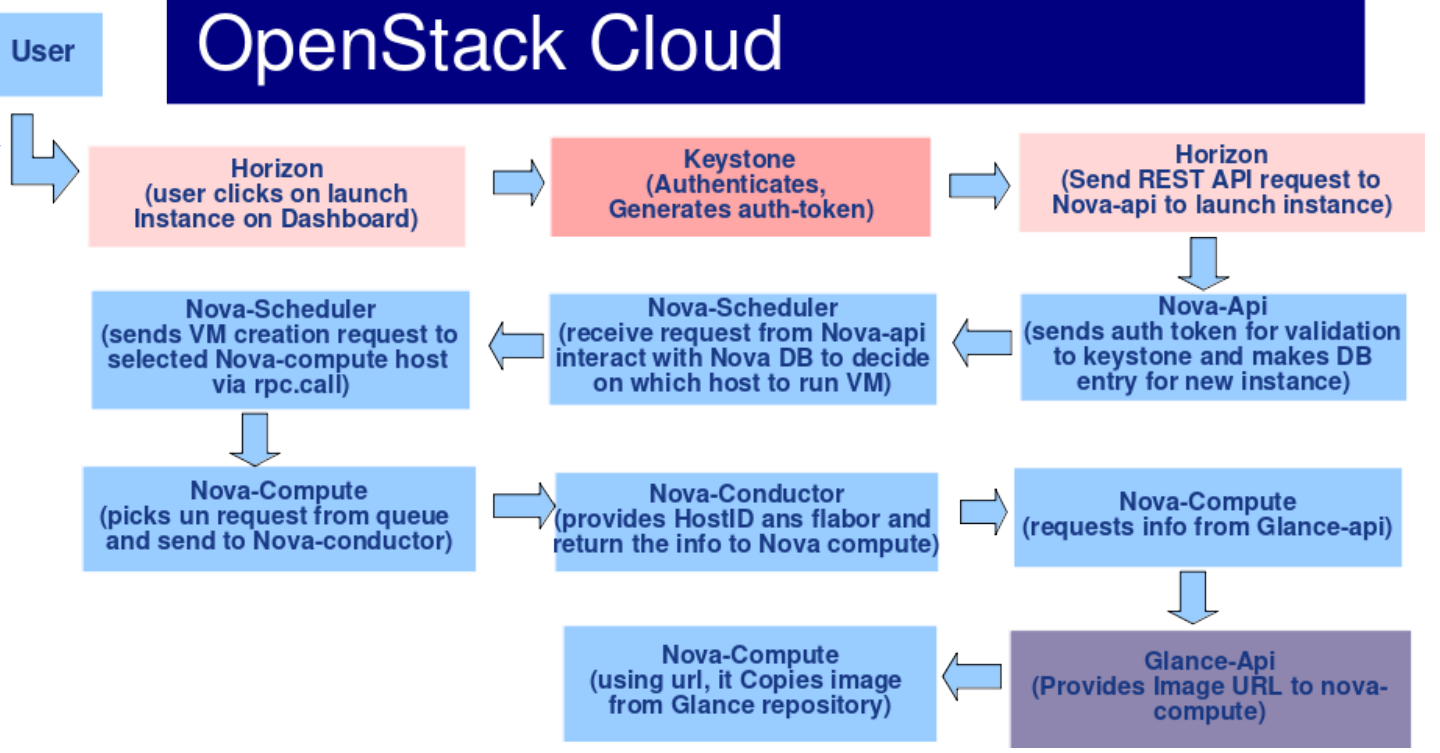

User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)
Keystone
(Authenticates,
Generates auth-token)
Horizon
(Send REST API request to
Nova-api to launch instance)
Nova-Api
(sends auth token for validation
to keystone and makes DB
entry for new instance)
Nova-Scheduler
(receive request from Nova-api
interact with Nova DB to decide
on which host to run VM)
Nova-Scheduler
(sends VM creation request to
selected Nova-compute host
via rpc.call)

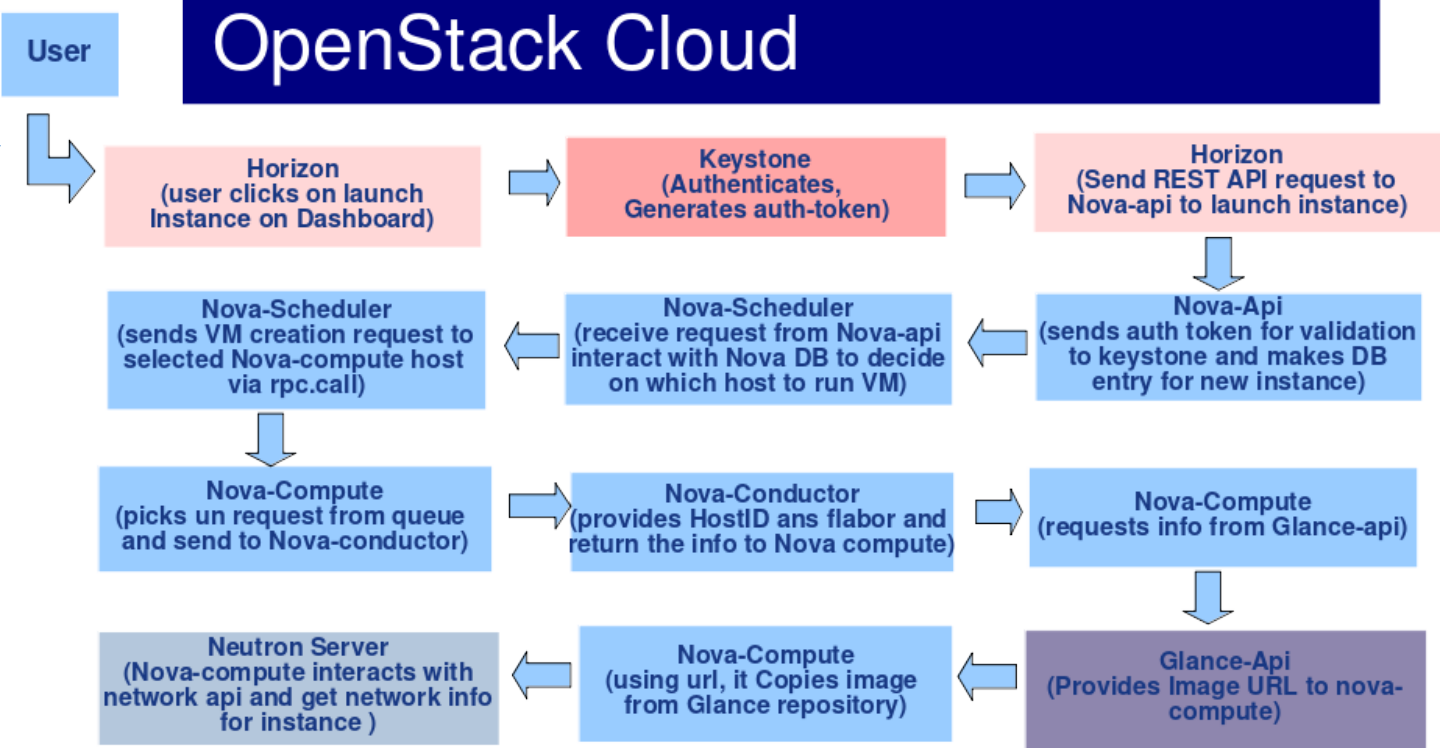

User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)
Keystone
(Authenticates,
Generates auth-token)
Horizon
(Send REST API request to
Nova-api to launch instance)
Nova-Api
(sends auth token for validation
to keystone and makes DB
entry for new instance)
Nova-Scheduler
(receive request from Nova-api
interact with Nova DB to decide
on which host to run VM)
Nova-Scheduler
(sends VM creation request to
selected Nova-compute host
via rpc.call)
Nova-Compute
(picks un request from queue
and send to Nova-conductor)


User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)
Keystone
(Authenticates,
Generates auth-token)
Horizon
(Send REST API request to
Nova-api to launch instance)
Nova-Api
(sends auth token for validation
to keystone and makes DB
entry for new instance)
Nova-Scheduler
(receive request from Nova-api
interact with Nova DB to decide
on which host to run VM)
Nova-Scheduler
(sends VM creation request to
selected Nova-compute host
via rpc.call)
Nova-Conductor
(provides HostID ans flabor and
return the info to Nova compute)
Nova-Compute
(picks un request from queue
and send to Nova-conductor)

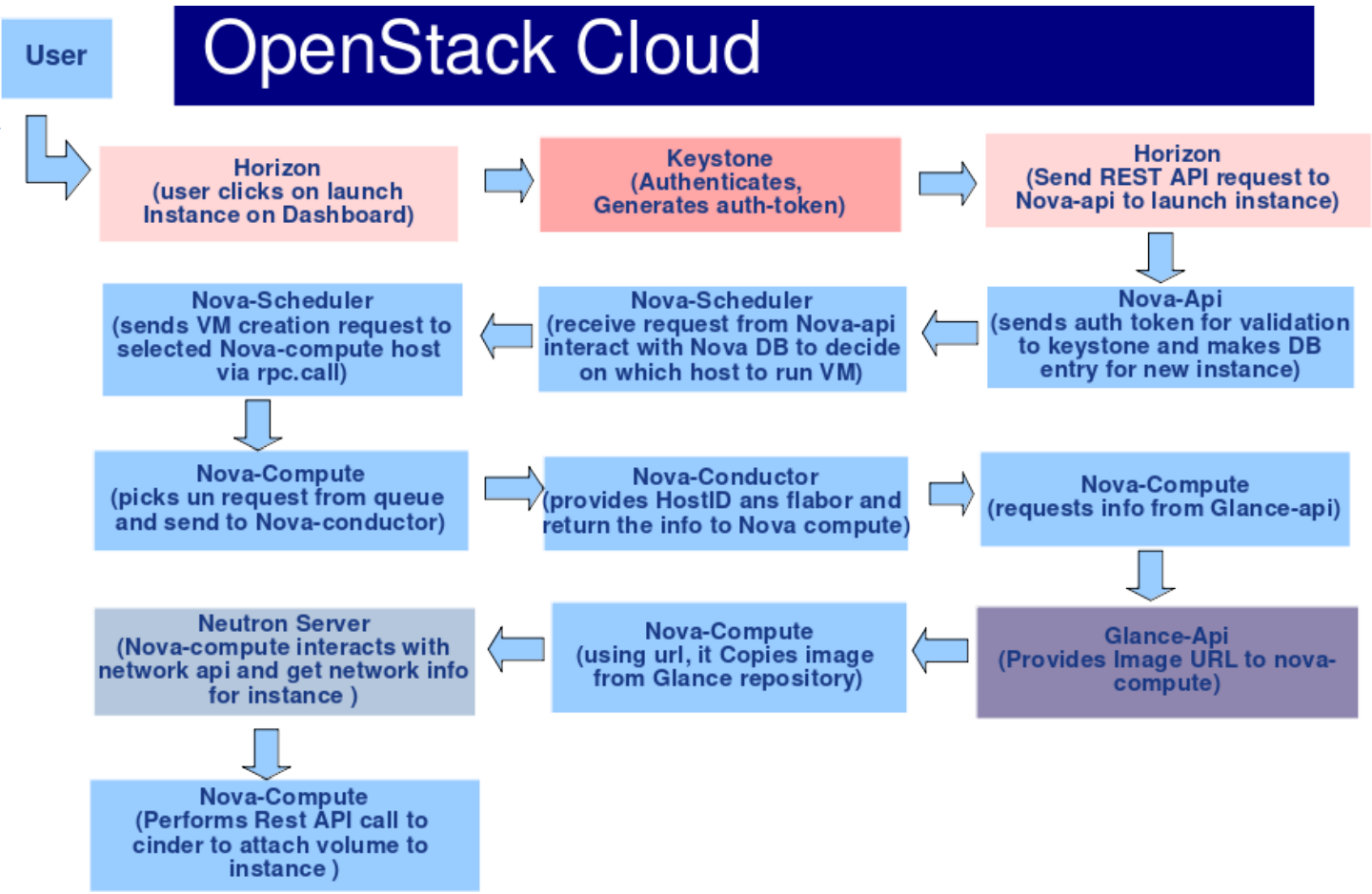

User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)
Keystone
(Authenticates,
Generates auth-token)
Horizon
(Send REST API request to
Nova-api to launch instance)
Nova-Api
(sends auth token for validation
to keystone and makes DB
entry for new instance)
Nova-Scheduler
(receive request from Nova-api
interact with Nova DB to decide
on which host to run VM)
Nova-Scheduler
(sends VM creation request to
selected Nova-compute host
via rpc.call)
Nova-Compute
(requests info from Glance-api)
Nova-Conductor
(provides HostID ans flabor and
return the info to Nova compute)
Nova-Compute
(picks un request from queue
and send to Nova-conductor)

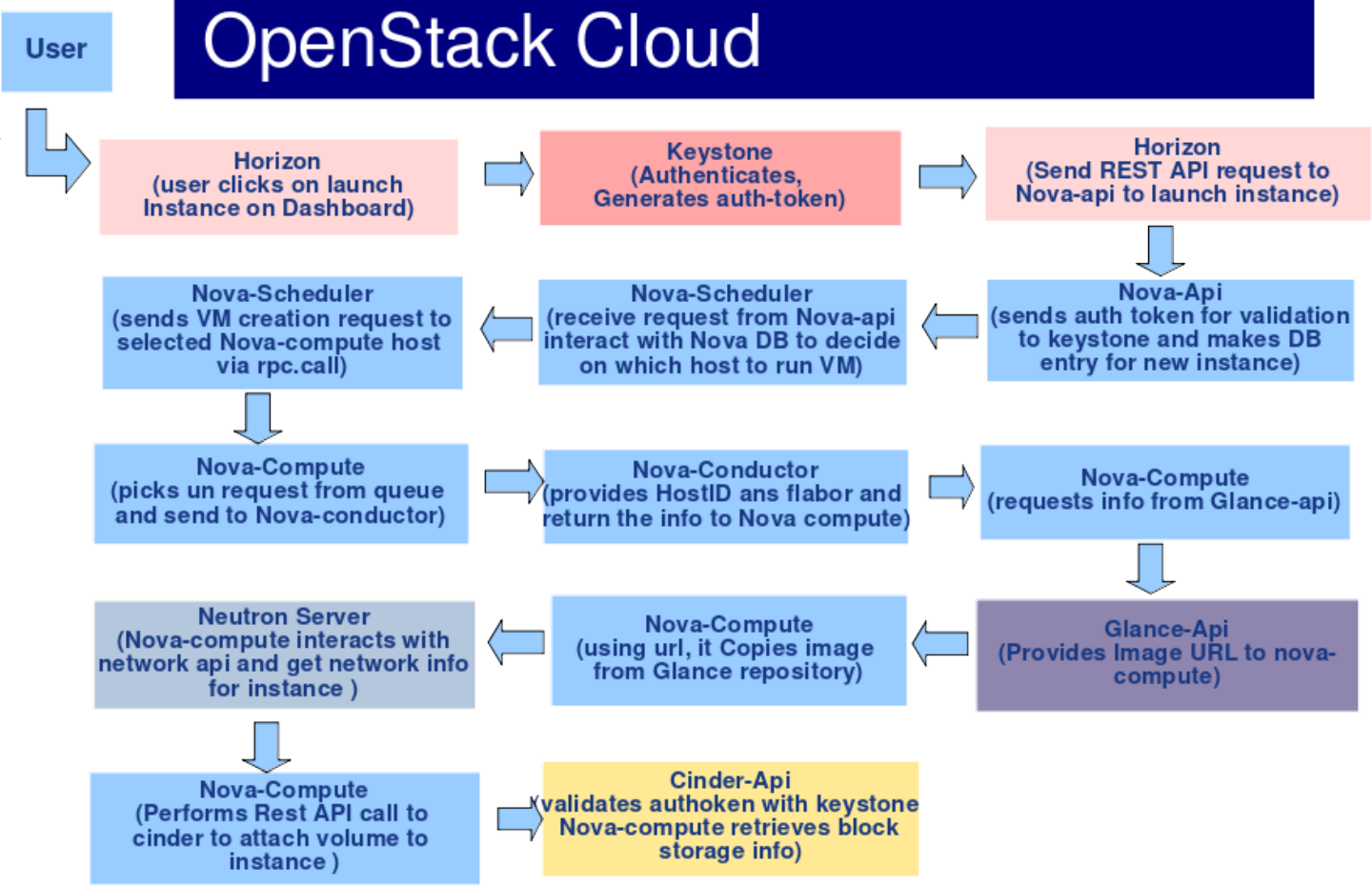

User
OpenStack Cloud
Horizon
(user clicks on launch
Instance on Dashboard)
Keystone
(Authenticates,
Generates auth-token)
Horizon
(Send REST API request to
Nova-api to launch instance)
Nova-Api
(sends auth token for validation
to keystone and makes DB
entry for new instance)
Nova-Scheduler
(receive request from Nova-api
interact with Nova DB to decide
on which host to run VM)
Nova-Scheduler
(sends VM creation request to
selected Nova-compute host
via rpc.call)
Nova-Compute
(picks un request from queue
and send to Nova-conductor)
Nova-Conductor
(provides HostID ans flabor and
return the info to Nova compute)
Nova-Compute
(requests info from Glance-api)
Glance-Api
(Provides Image URL to nova-
compute)

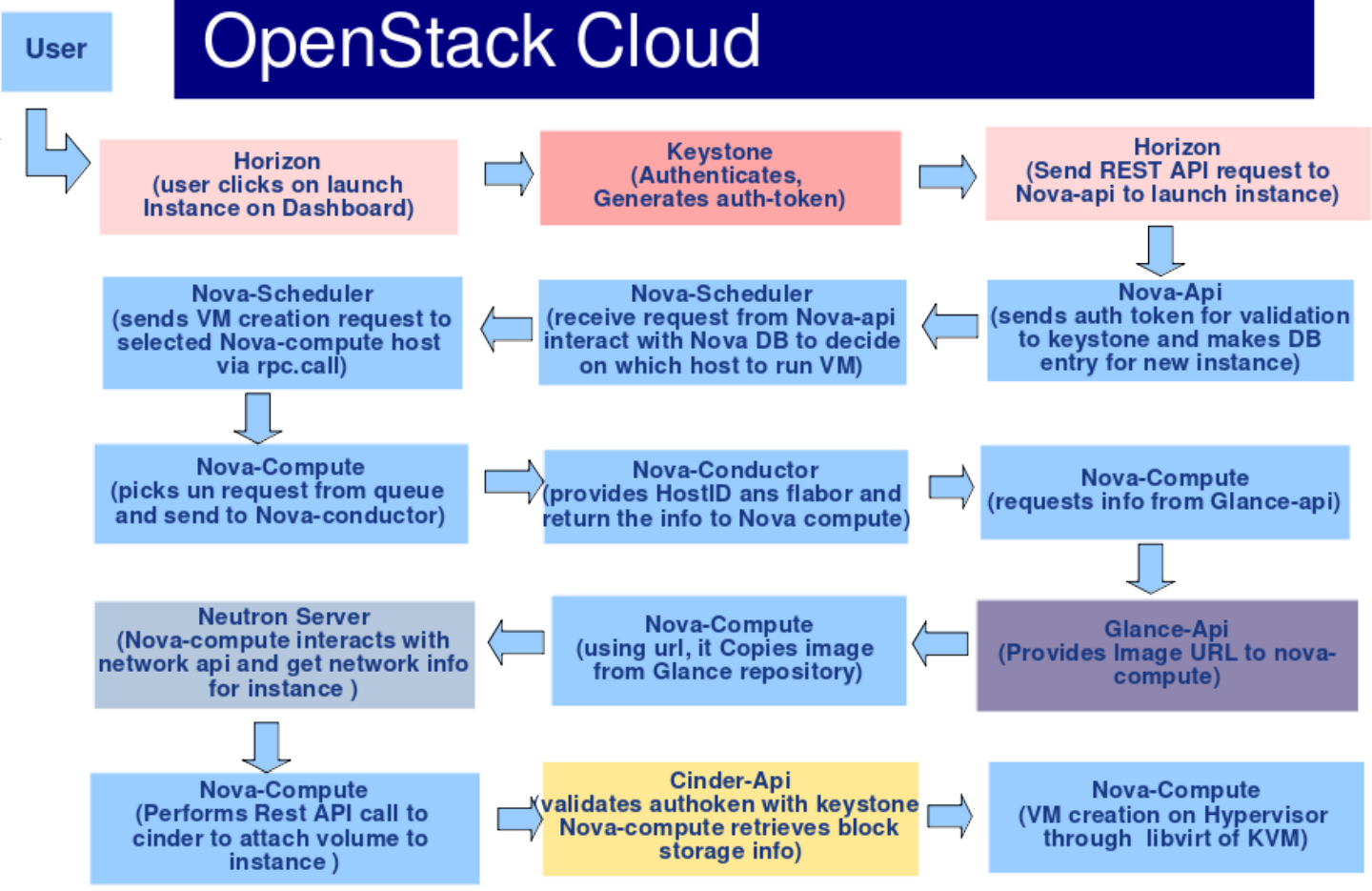

User
OpenStack Cloud
Horizon
(user clicks on launch Instance on Dashboard)
Keystone
(Authenticates, Generates auth-token)
Horizon
(Send REST API request to Nova-api to launch instance)
Nova-Api
(sends auth token for validation to keystone and makes DB entry for new instance)
Nova-Scheduler
(receive request from Nova-api interact with Nova DB to decide on which host to run VM)
Nova-Scheduler
(sends VM creation request to selected Nova-compute host via rpc.call)
Nova-Compute
(picks un request from queue and send to Nova-conductor)
Nova-Conductor
(provides HostID ans flabor and return the info to Nova compute)
Nova-Compute
(requests info from Glance-api)
Glance-Api
(Provides Image URL to nova-compute)
Nova-Compute
(using url, it Copies image from Glance repository)


User
OpenStack Cloud
Horizon
(user clicks on launch Instance on Dashboard)
Keystone
(Authenticates, Generates auth-token)
Horizon
(Send REST API request to Nova-api to launch instance)
Nova-Api
(sends auth token for validation to keystone and makes DB entry for new instance)
Nova-Scheduler
(receive request from Nova-api interact with Nova DB to decide on which host to run VM)
Nova-Scheduler
(sends VM creation request to selected Nova-compute host via rpc.call)
Nova-Compute
(picks un request from queue and send to Nova-conductor)
Nova-Conductor
(provides HostID ans flabor and return the info to Nova compute)
Nova-Compute
(requests info from Glance-api)
Glance-Api
(Provides Image URL to nova-compute)
Nova-Compute
(using url, it Copies image from Glance repository)
Neutron Server
(Nova-compute interacts with network api and get network info for instance )

# OpenStack Cloud




User
OpenStack Cloud
Horizon
(user clicks on launch Instance on Dashboard)
Keystone
(Authenticates, Generates auth-token)
Horizon
(Send REST API request to Nova-api to launch instance)
Nova-Api
(sends auth token for validation to keystone and makes DB entry for new instance)
Nova-Scheduler
(receive request from Nova-api interact with Nova DB to decide on which host to run VM)
Nova-Scheduler
(sends VM creation request to selected Nova-compute host via rpc.call)
Nova-Compute
(picks un request from queue and send to Nova-conductor)
Nova-Conductor
(provides HostID ans flabor and return the info to Nova compute)
Nova-Compute
(requests info from Glance-api)
Glance-Api
(Provides Image URL to nova-compute)
Nova-Compute
(using url, it Copies image from Glance repository)
Neutron Server
(Nova-compute interacts with network api and get network info for instance )
Nova-Compute
(Performs Rest API call to cinder to attach volume to instance )
Cinder-Api
(validates authoken with keystone Nova-compute retrieves block storage info)


User
OpenStack Cloud
Horizon
(user clicks on launch Instance on Dashboard)
Keystone
(Authenticates, Generates auth-token)
Horizon
(Send REST API request to Nova-api to launch instance)
Nova-Api
(sends auth token for validation to keystone and makes DB entry for new instance)
Nova-Scheduler
(receive request from Nova-api interact with Nova DB to decide on which host to run VM)
Nova-Scheduler
(sends VM creation request to selected Nova-compute host via rpc.call)
Nova-Compute
(picks un request from queue and send to Nova-conductor)
Nova-Conductor
(provides HostID ans flabor and return the info to Nova compute)
Nova-Compute
(requests info from Glance-api)
Glance-Api
(Provides Image URL to nova-compute)
Nova-Compute
(using url, it Copies image from Glance repository)
Neutron Server
(Nova-compute interacts with network api and get network info for instance )
Nova-Compute
(Performs Rest API call to cinder to attach volume to instance )
Cinder-Api
(validates authoken with keystone Nova-compute retrieves block storage info)
Nova-Compute
(VM creation on Hypervisor through libvirt of KVM)