OpenStack 기반의 Kubernetes

어디까지 해봤니?

그리고 어디로 갈 것인가

Jin Gi KONG

Bluemix Technical Evangelist, IBM

Outline

1. Overview
   Why Integrate OpenStack and Kubernetes
2. How it’s Done
   Integration Plug Points
3. Integrating Kubernetes with OpenStack’s IAM (Keystone)
4. Enabling Kubernetes to work with OpenStack networking (Neutron)
5. Utilizing Openstack storage services with Kubernetes (Cinder, Manila …)

Overview

Why integrating Kubernetes with OpenStack

Kubernetes is quickly becoming a popular approach for managing and scheduling containers across clusters

- Operating Kubernetes requires support from the underlying infrastructure
  - Provision manually or through the Cloud
  - Automation needed
  - Need access to storage and networking resources
- OpenStack is a great platform for deploying Kubernetes
  - Supports multi-tenancy
  - Has options for providing security, authentication/authorization, network and storage virtualization readily available
  - OpenStack provides excellent horizontal scaling support
  - Many companies have already deployed OpenStack to managing their infrastructure naturally making it a readily available option

How it’s done

The code: Kubernetes plugin for OpenStack

- Kubernetes plugin is how Kubernetes interfaces to IaaS layer
  - Supports plugging into many Cloud providers: OpenStack, GCE, AWS, Azure, ...
  - Easily configurable for each Cloud
  - Leveraged by key Kube components: kube-apiserver, kube-controllermanager, kubelet
- Kubernetes plugin for OpenStack supports:
  - OpenStack Identity
  - OpenStack Networking
  - OpenStack Storage
- Code:
  - gophercloud repo: https://github.com/rackspace/gophercloud
  - Kubernetes repo: https://github.com/kubernetes/kubernetes
    - pkg/cloudprovider/providers/openstack
    - pkg/volume/cinder
  - OpenStack repo: http://git.openstack.org/cgit/openstack/k8s-cloud-provider
- Magnum project delivers the integration of Kubernetes and OpenStack

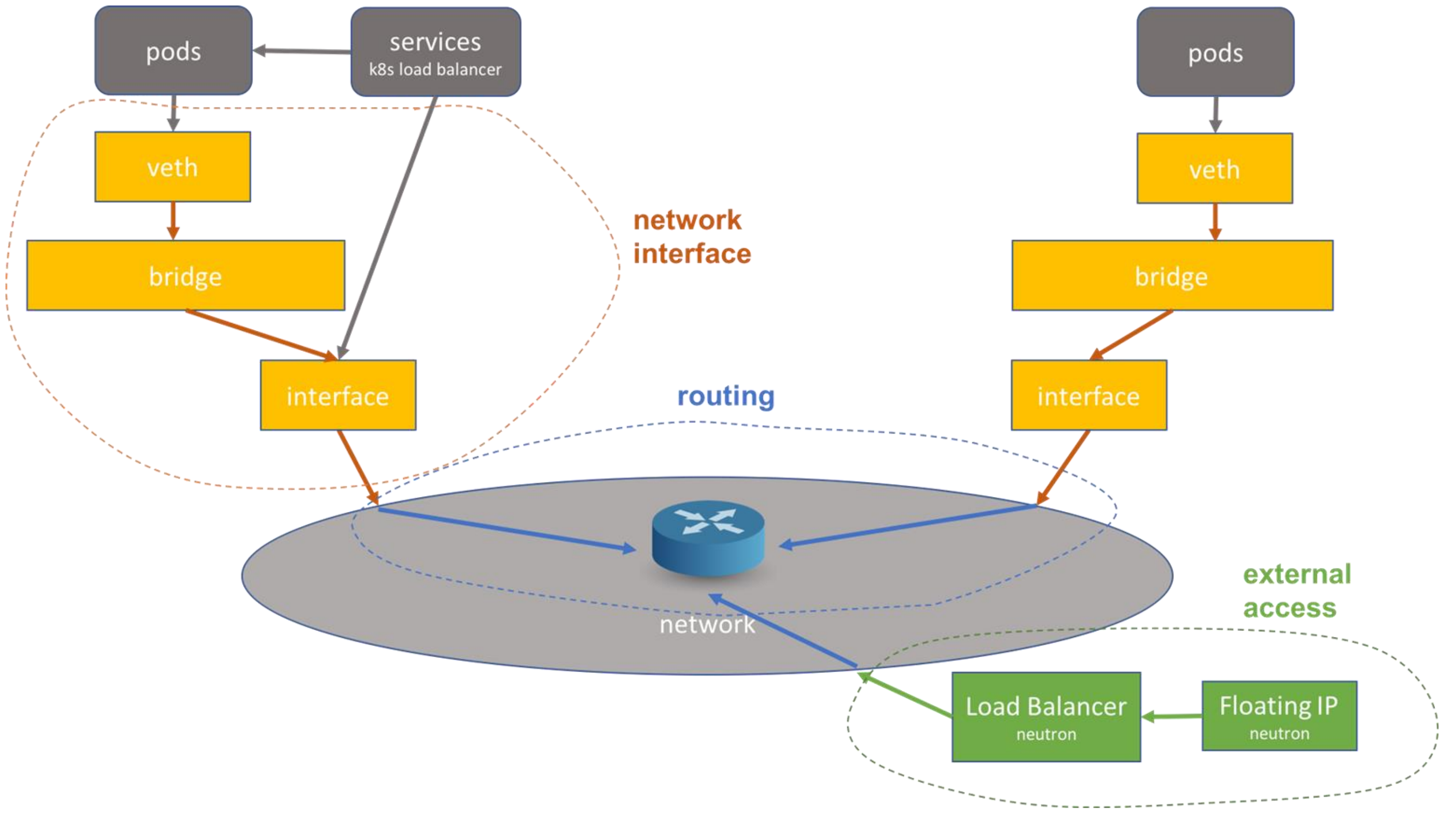
Integrating with Identity Management

Identity Management Integration

- Keystone:
  - Robust identity service already in place, fully populated by Cloud provider
  - Already provides multiple LDAPs and MS ADs integration and federated identity support
- How Kubernetes services access OpenStack services
  - Code: gophercloud package
  - Establish session to access neutron, cinder, ...
  - User credential: multi-tenancy in OpenStack
  - Keystone trust id for better security: automated by Magnum
- Future consideration
  - Multi-tenancy support within Kubernetes:
    - Proceed with caution: current container is not as secure and isolated as a Virtual Machine
  - Keystone for projects, users?

Integrating Networking

Networking Integration



Networking: create network interfaces

- What needs to be done:
  - Set up and tear down network interface, assigns IP
  - Done by kubelet when pod is created/deleted
- Current options in OpenStack
  - Network driver for kubelet: Kubenet, CNI
  - Kuryr CNI driver: neutron native support, plugin optimized for network provider
    - VM, pods on same network
- Future considerations
  - Kuryr: performance improvement
  - Development in alpha, many changes expected

Networking: routing

- What needs to be done:
  - Pods from any node can reach pods on any other nodes
  - Routing set up when the cluster is deployed and when a new node is added
- Current options in OpenStack:
  - Overlay network: Flannel
  - Possible to avoid encapsulation overhead: hostgw mode
  - Others: Calico, ...
- Future consideration
  - Leverage neutron routing: similar to native routing in GCE

Networking: external access

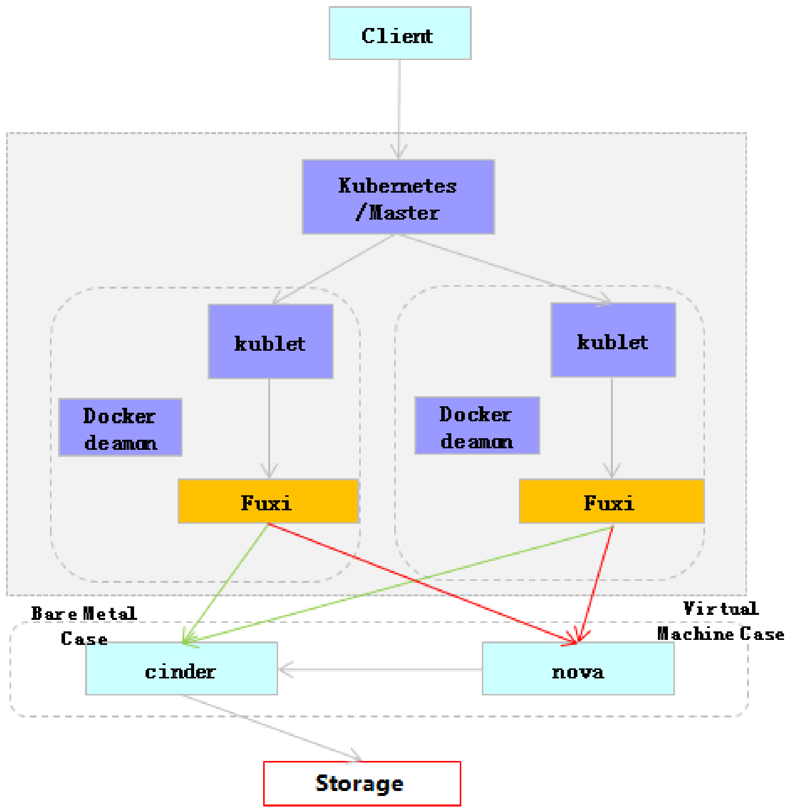
- What needs to be done:
  - Different from load balancer in Kubernetes service
  - Allow service to be accessible from external internet: get an external IP by neutron
- Current options in OpenStack:
  - Support for LBaaS v1 and v2
  - Automated by Magnum
- Future consideration
  - Add support for region: default to current region
  - Add floating IP automation

Networking: demo

- Kubernetes 1.5.2
  - 1 master, 2 nodes
- Nova VM’s
- Private Neutron network
- Deployed by Magnum
- Code from Master

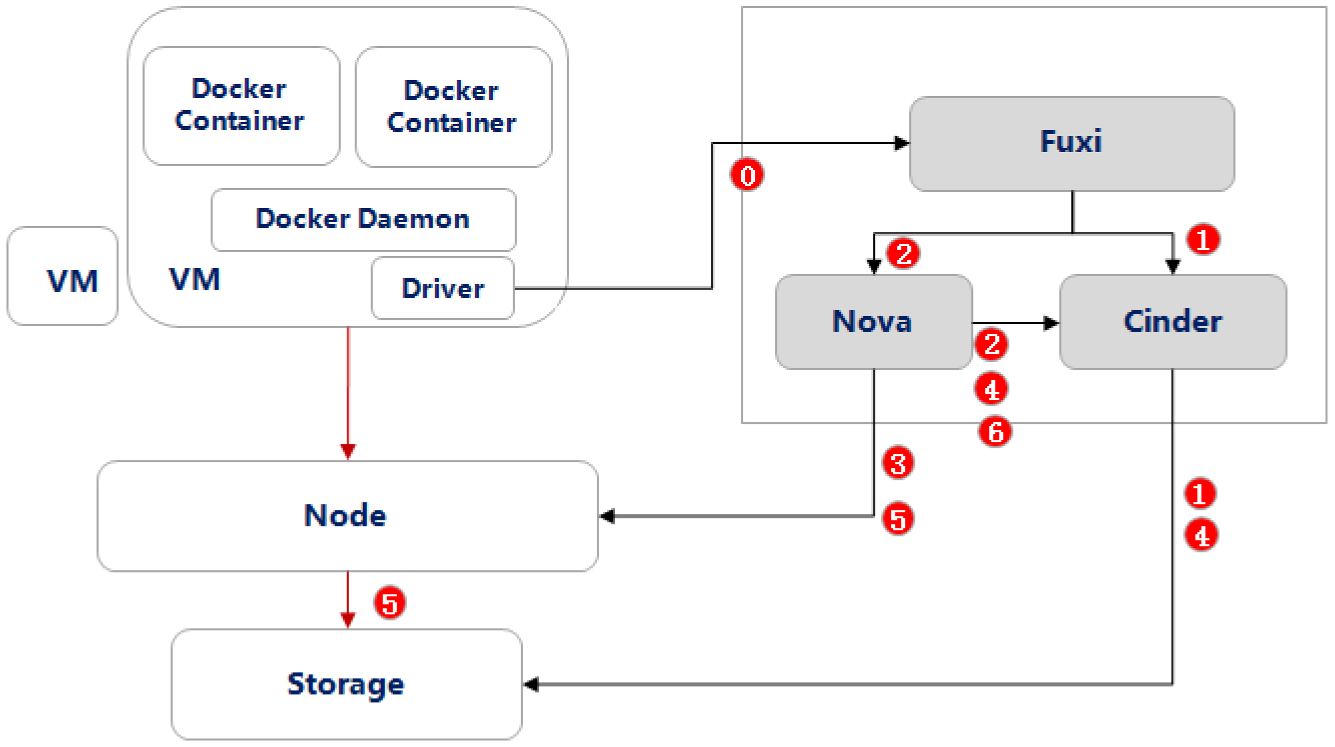
Integrating Storage

Storage: High Level



- Persistent Storage:
  - Cinder Volume: supports 50+ different storage drivers from numerous vendors
  - Connect to individual storage driver directly
- Raw Device
- Integrating with Cinder giving users convenience and wide range of storage options

Is Fuxi slow creating and attaching volume?



1.Create volume: Cinder api, RESTful

2.Fetch volume info, Cinder api, reserve the volume

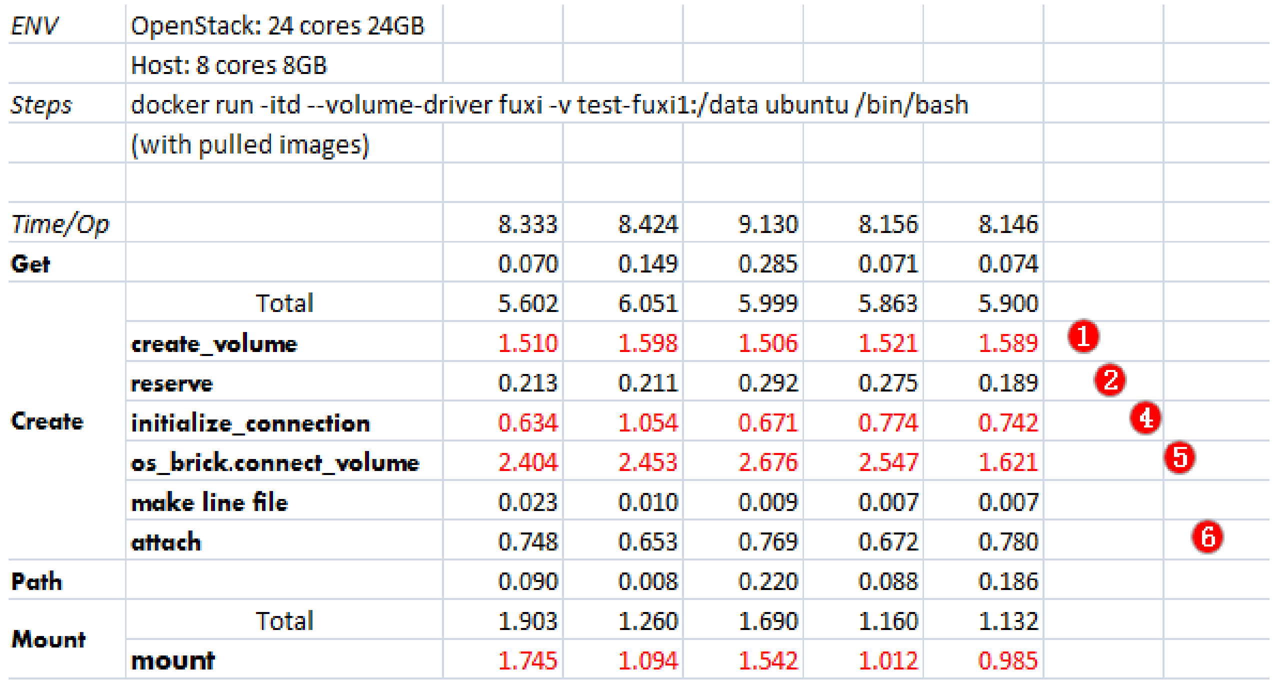
3.Nova gets host Connector info

4.Nova calls Cinder api, init connection, call storage RESTful api volume mapping

5.Nova uses LibvirtISCSVolumeDriver: “connect_volume”, to let host scan the disk and attach the volume to the host

6.Nova call Cinder API “attach” to set volume status

Is Fuxi slow creating and attaching volume?



ENV	OpenStack: 24 cores 24GB						
	Host: 8 cores 8GB						
Steps	docker run -itd --volume-driver fuxi -v test-fuxi1:/data ubuntu /bin/bash						
	(with pulled images)						
Time/Op		8.333	8.424	9.130	8.156	8.146	
Get		0.070	0.149	0.285	0.071	0.074	
	Total	5.602	6.051	5.999	5.863	5.900	
	create_volume	1.510	1.598	1.506	1.521	1.589	1
	reserve	0.213	0.211	0.292	0.275	0.189	2
Create	**initialize_connection**	0.634	1.054	0.671	0.774	0.742	4
	os_brick.connect_volume	2.404	2.453	2.676	2.547	1.621	5
	make line file	0.023	0.010	0.009	0.007	0.007	
	attach	0.748	0.653	0.769	0.672	0.780	6
Path		0.090	0.008	0.220	0.088	0.186	
Mount	Total	1.903	1.260	1.690	1.160	1.132	
	mount	1.745	1.094	1.542	1.012	0.985	

Time spent:

Longest:

(5) The host attaches the volume - related to volume size

2nd:

(1) create volume

3rd:

(4) volume mapping

Is Fuxi slow?

- PV/PVC
  - What is solved and what is not
    - Create volume separately - save ¾ of the time
    - Still need about 2 seconds to mount the volume to the container
    - Don’t need to talk to Nova in bare metal case
  - Manila:
    - File Level Storage can help
    - Don’t need to attach the volume to the host

Storage: demo

- Mount Cinder volume to k8s pod
- Create dynamic PVC/PV from Cinder

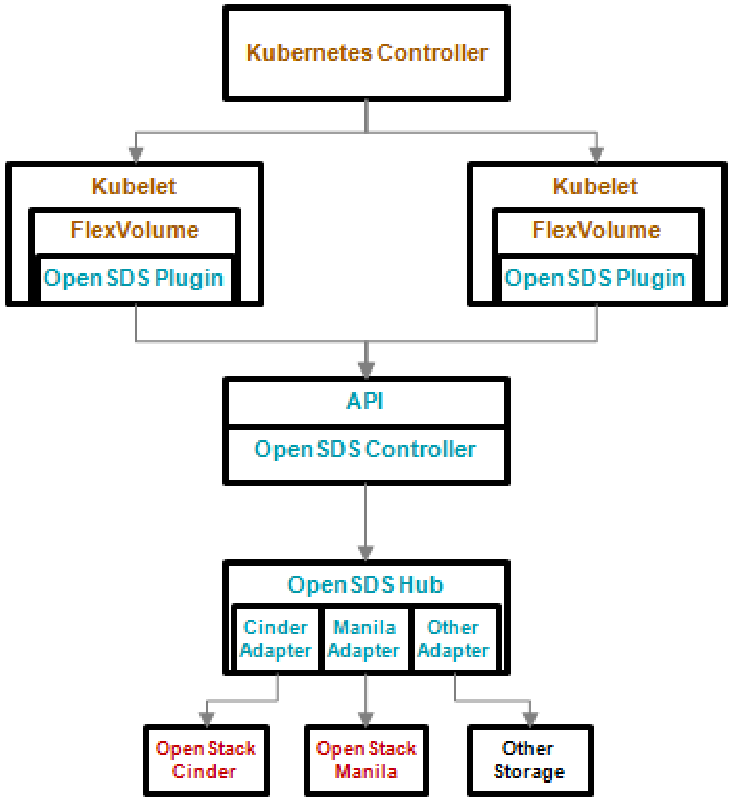
Storage: stateful container and data protection

- Integration with Karbor
  - What is Karbor:
    - Protect the Data and Meta-Data that comprises an OpenStack-deployed Application.
    - Provide a standard framework of APIs and services that enables vendors to introduce various data protection services into a coherent and unified flow for the user.
  - Snapshot:
    - For rollback
    - During upgrade:
  - Replication:
    - Protecting Data and Meta-Data
    - Working with application level data replication and protection

Storage: working with OpenSDS

- OpenSDS helps OpenStack and K8S with:
    1. Storage discovery
    2. Storage lifecycle management
    3. data protection

Repo: https://github.com/opensds
Slack: opensds.slack.com
Mailing list: https://groups.google.com/forum/?hl=en#!forum/opensds-dev



Conclusion

Conclusion

- Presented how OpenStack is an excellent IaaS option for Kubernetes and enables more efficient multi-tenant utilization of resources
- Provides out of the box options for security, authentication/authorization, network and storage virtualization
- Combination is a great example of leveraging the best technology from multiple open source communities
- Provided an overview of the benefits of integrating OpenStack and Kubernetes
- Described the integration: code is available, active development
- Many improvements are possible in the near future

Thank You

Research based on:

Brad Topol
btopol@us.ibm.com
@bradtopol

Ton Ngo
ton@us.ibm.com
@tango245

Yin Ding
Yin.Ding@huawei.com