openstack.

05.01.2019

# Shared Storage for Container Orchestrators with Manila

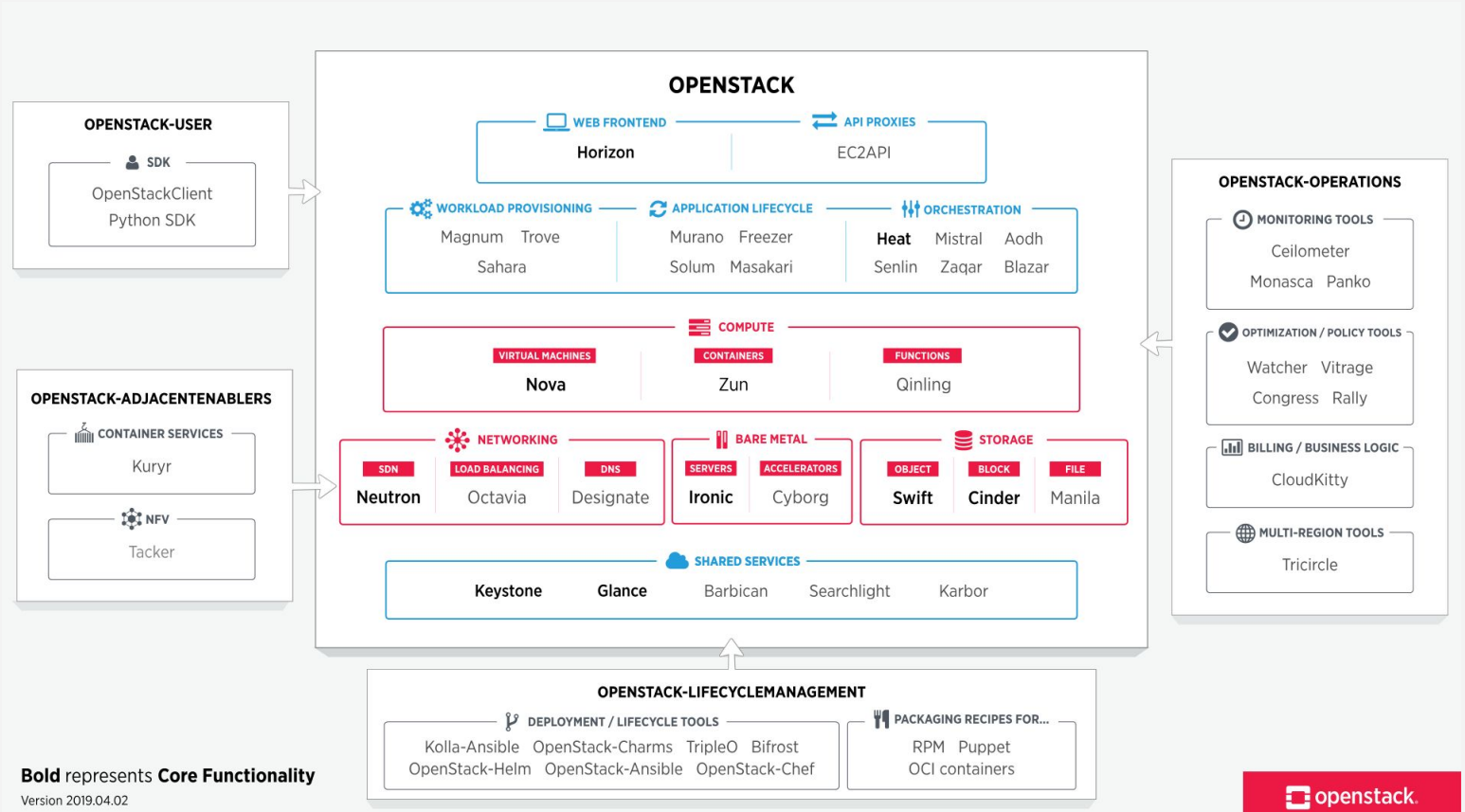
Open Infrastructure Summit 2019, Denver, CO

Tom Barron <tpb@dyncloud.net>
irc: tbarron

Goutham Pacha Ravi <gouthampravi@gmail.com>
irc: gouthamr

Victoria Martinez de la Cruz <victoria@redhat.com>
irc: vkmc

# Manila integrated into OpenStack

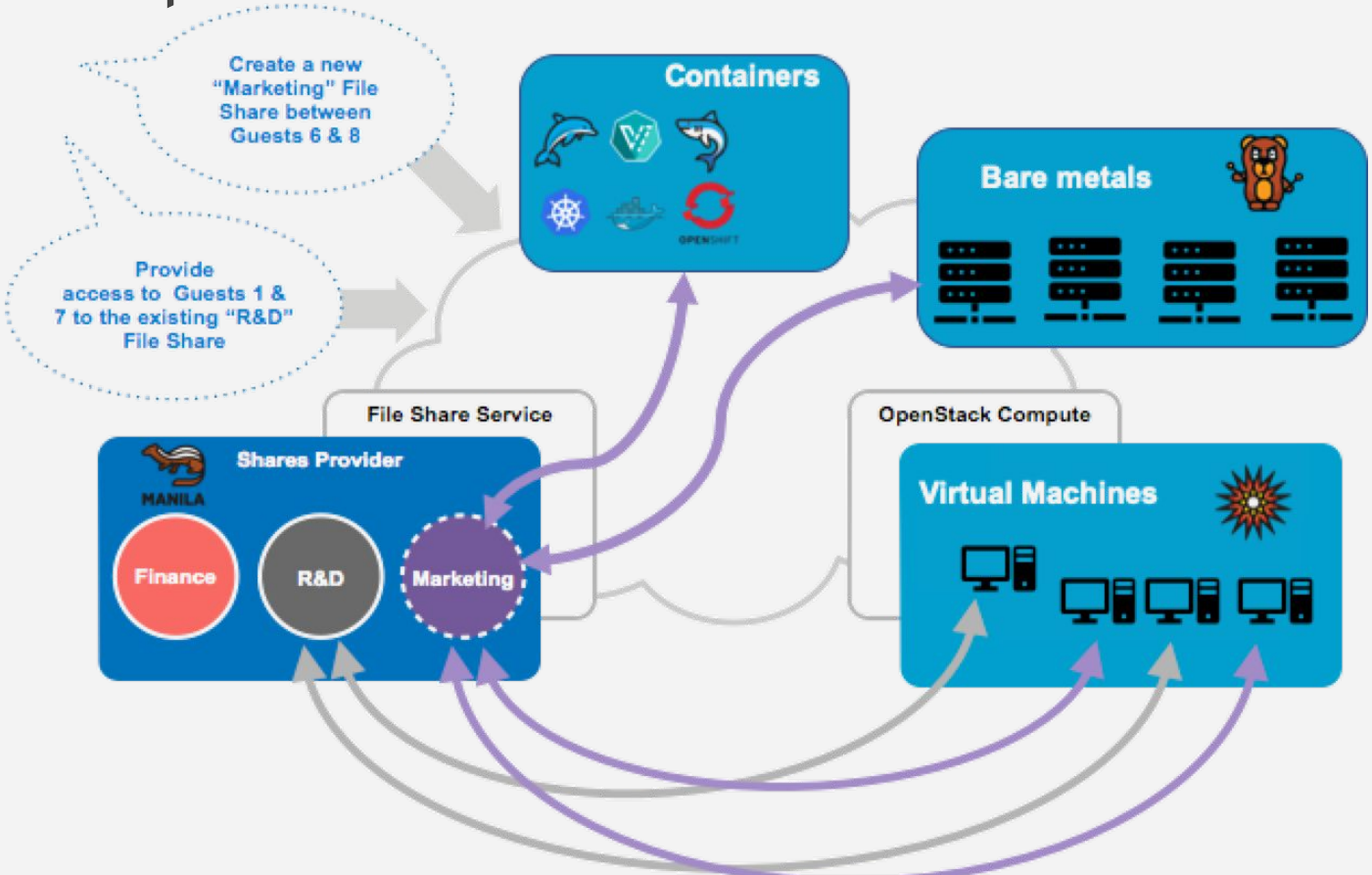



**Bold** represents **Core Functionality**

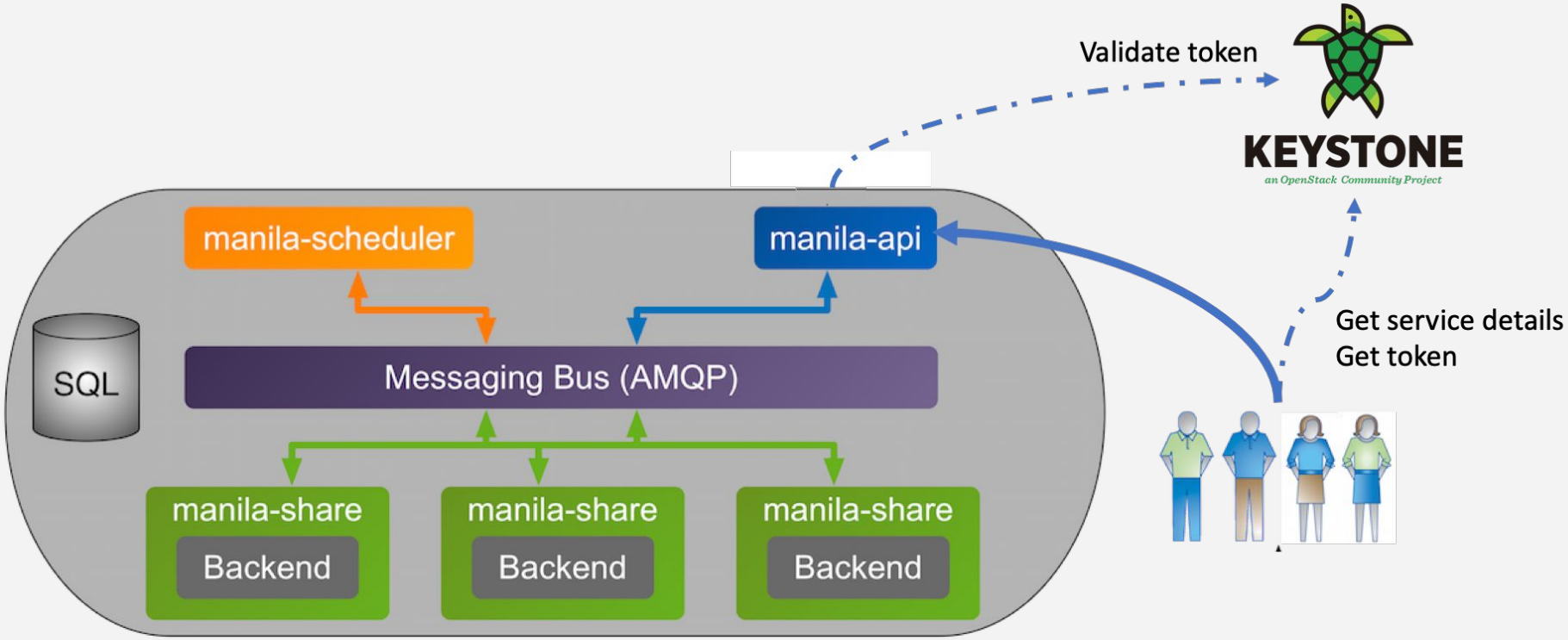
Version 2019.04.02

openstack.

# Consumption



# Service architecture



# Manila is Open File System Infrastructure

Loosely coupled with Nova and other OpenStack components

Serves Storage over Network rather than through a hypervisor

Some have argued that this is a weakness in traditional Nova-centric OpenStack

But it is a strength in the new Open Infrastructure world order

# Manila is Open File System Infrastructure

Supports both proprietary and *production-quality* open-source back ends.


You can use Open Source software defined Ceph Storage for:

- Objects in Container Buckets
- Block devices
- File Systems


File Systems can be presented over NFS in addition to native CephFS.

File systems can be presented to VMs, bare metal, and containers running in VMs or on bare metal, inside or outside of OpenStack.

# Container Orchestrators

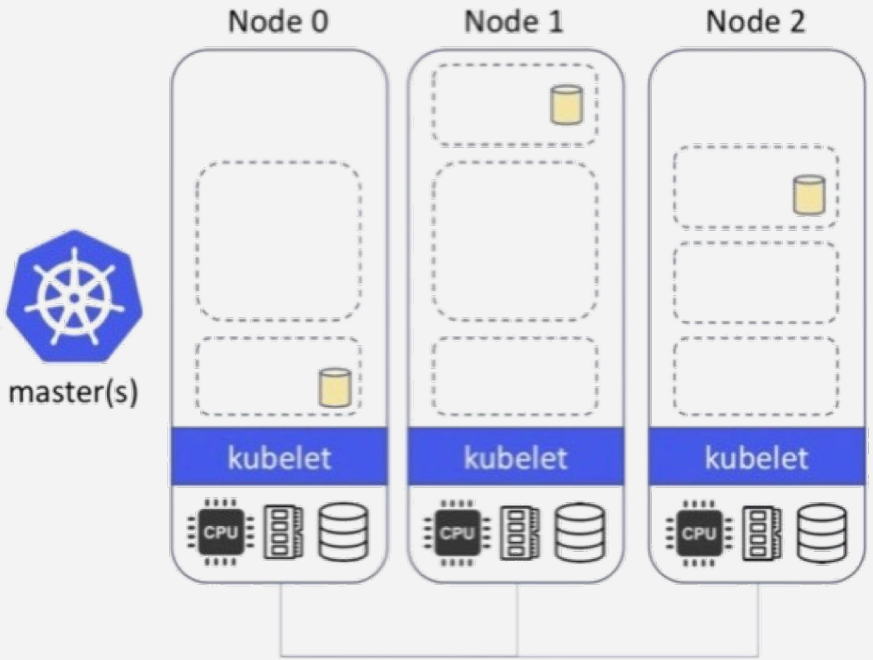



Source: https://blog.thecodeteam.com/2017/08/15/container-storage-interface-according-josh/

Container Orchestrators
need infrastructure to run on.

Either you rent it or you buy and
manage it.

# CO - Challenge one: Provisioning




?

MANILA

Microsoft Azure

amazon web services

Google Cloud Platform

CINDER

ceph

Source: https://www.slideshare.net/SeanCohen/storage-101-rook-and-ceph-open-infrastructure-denver-2019

# CO: Challenge 2 - storage consumption

ReadWriteOnce

ReadOnlyMany

ReadWriteMany

Single Node

Multiple Nodes

# Container Storage Interface

Unifying interface across Container Orchestrators

- Provides a scope for abstractions and simplifications
- Includes reasonable grounds for extensions and flexibility

A reference architecture on breaking down provisioning and allowing granular control of attachments.

An integration point for infrastructure provisioners such as Cinder, Manila, EBS, EFS, Azure Files, etc.

The emphasis is not only “provisioning” storage, but also support advanced storage orchestration

# Manila Container Storage Interface

Why:

- Flexibility: multi-vendor, multi-protocol
- Security: multi-tenancy
- Maturity: day 2 operations

Why not:

- Homogenous storage
- Single tenant deployments

# Manila Container Storage Interface

Common scenario:

- In-house OpenStack serves multiple COs run by sub-organizations
- One sub-org has bought dedicated vendor storage with special-sauce features that they like
- Others just want whatever storage is available
- Some of the sub-organizations are trusted tenants so that it makes sense to give them CephFS native
- Some of the sub-organizations are not trusted in this sense so CephFS storage should be mediated by an NFS gateway
- Sub organization using storage X wants to archive their data, or make it available to other applications that don’t mind using storage Y

Manila CSI can handle deployments of this kind

- Storage Classes and Manila Share Types
- Manila data motion APIs

# Manila CSI: How we got here

- Manila+K8s dynamic storage provisioner

- CERN presented their work with hybrid external service provider (on master) and CephFS native CSI driver (on worker nodes)
- Dynamic Storage Provisioning of Manila/CephFS Shares on Kubernetes
  - https://www.openstack.org/summit/berlin-2018/summit-schedule/events/21997/dynamic-storage-provisioning-of-manilacephfs-shares-on-kubernetes
  - slides
  - https://github.com/kubernetes/cloud-provider-openstack (master)
  - https://github.com/ceph/ceph-csi (worker)
- Good performance and scale results with k8s 1.12 using CSISkipAttach

# Manila CSI: How we got here

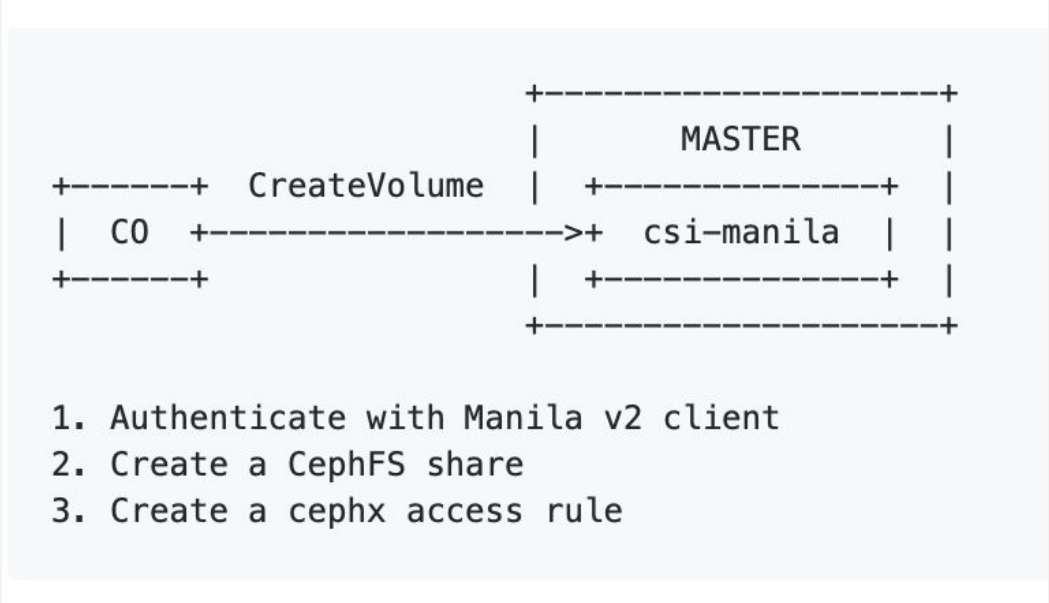
https://www.openstack.org/summit/berlin-2018/summit-schedule/events/22830/setting-the-compass-for-manila-rwx-cloud-storage

https://www.openstack.org/summit/berlin-2018/summit-schedule/events/22752/sig-k8s-working-session

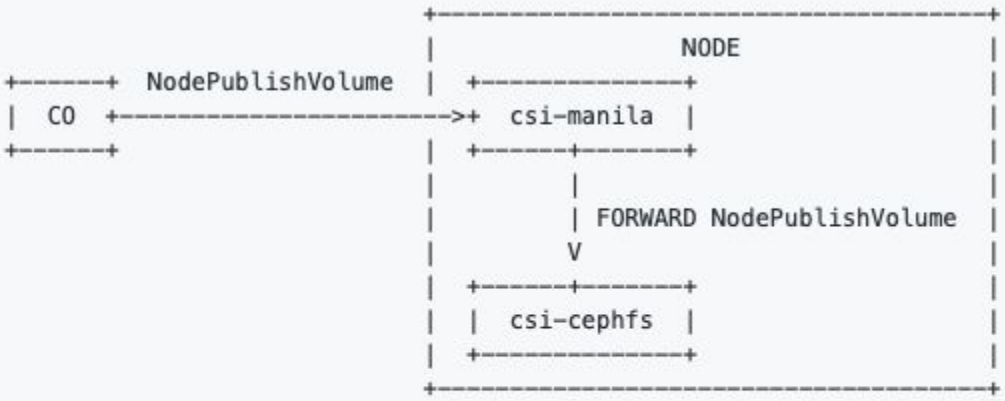
The plan: develop a true multi-protocol Manila CSI driver, integrated into cloud-provider-openstack:

http://lists.openstack.org/pipermail/openstack-dev/2018-November/136557.html

# Manila Container Storage Interface



# Manila Container Storage Interface




1. Authenticate with Manila v2 client
2. Retrieve the CephFS share
3. Retrieve the cephx access rule
4. Connect to csi-cephfs socket
5. Call csi-cephfs's NodePublishVolume, return its response

# The way forwards: manilakube integration lab

- K8s cluster -- currently master node and three workers
- Also deploys OpenStack devstack
  - Default devstack is minimal:
    - Manila, keystone, mysql, rabbitmq -- nothing else
    - Manila has native CephFS and CephFS via NFS back ends
- Golang environment, crictl, etc. installed in the environment
- Kubectl all set up both within the cluster and from the staging platform
- Automated install of Ceph CSI and Cloud Provider OpenStack with Manila CSI.
- Sufficient to do end-to-end tests of Manila CSI with native CephFS and NFS
- All implemented via ansible-playbooks that provision the k8s cluster on an OpenStack cloud

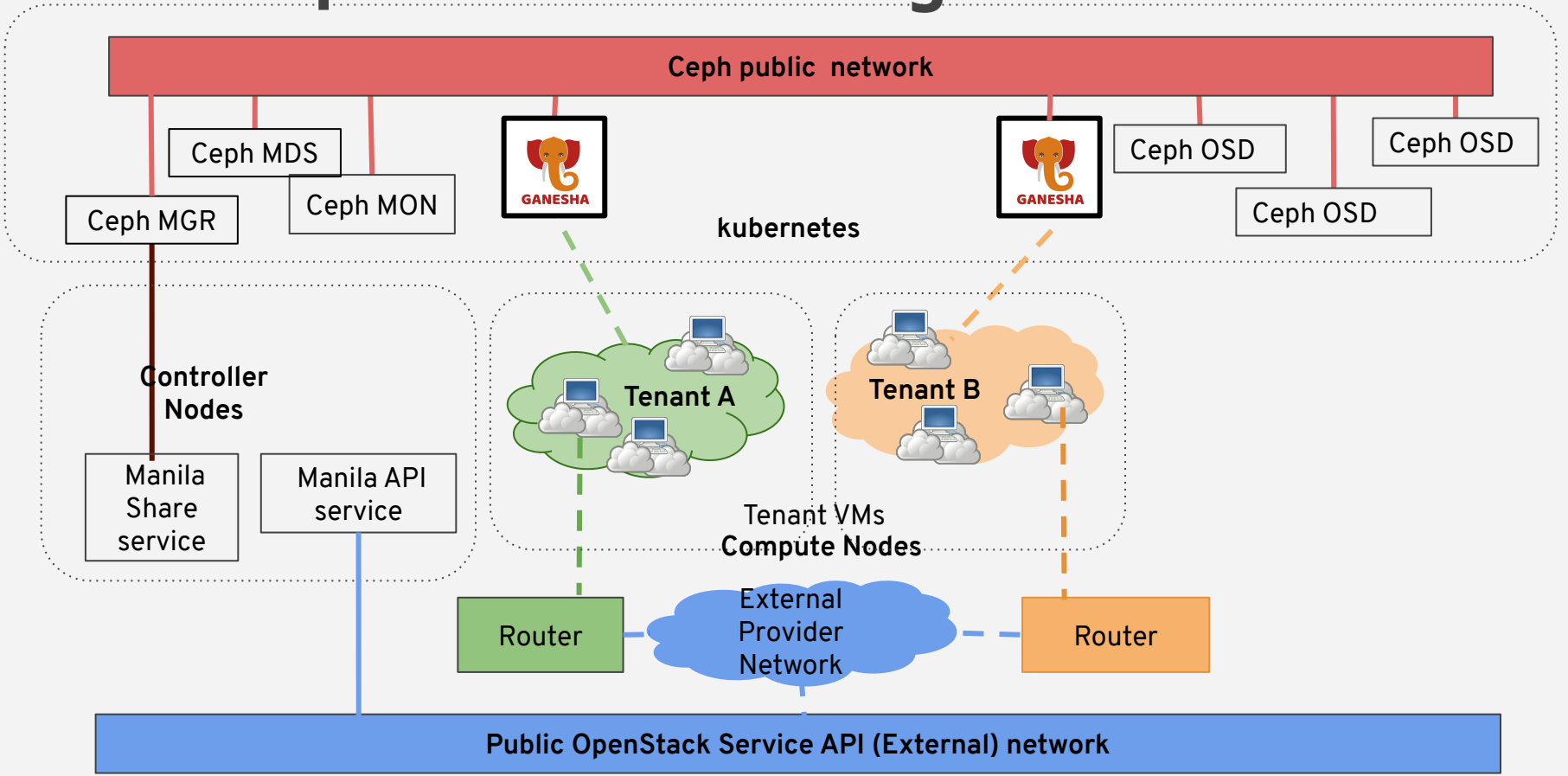
# The way forwards: manilakube and rook

- Instead of having devstack deploy CephFS and ganesha, use rook
  - Jeff Layton shows how to do this with minikube here.
- This sets up an external, scalable Ceph Cluster independent of OpenStack and Manila so that manila can use it as an external storage appliance just as it would use a proprietary NAS appliance.
- HA for ganesha is achieved via Kubernetes stateful set rather than by e.g. running a single instance of ganesha under control of pacemaker-corosync as we do today downstream
- Not having only a single ganesha instance under pacemaker control enables us to scale out NFS service

# The way forwards: manilakube, rook and kuryr

- Add kuryr to the manilakube mix
- Enhance the manila CephFS driver to run with full DHSS=True multitenancy support
- Scale out Ganesha servers per-tenant

# Ganesha per Tenant running under k8s control



# Summary

- We are working full steam ahead to integrate Manila CSI for K8s from OpenStack
- Next: bring in more CSI features - Snapshots, Volume Extension, Topology
- Exploring running Ceph and Ganesha daemons under k8s control
    - Scale out ganesha services per-tenant
    - Per-tenant networking via kuryr
- Actively investigating: scale-down hyperconverged deployments using minimal manila w/o the rest of OpenStack
    - Maybe drop keystone (run manila in *no-auth*mode)
    - manila services plus rabbitmq and mysql running under k8s
- Investigating  if k8s stateful sets are sufficient for our HA/availability requirements
    - Ganesha
    - manila-share

openstack.

# THANKS.

Questions?