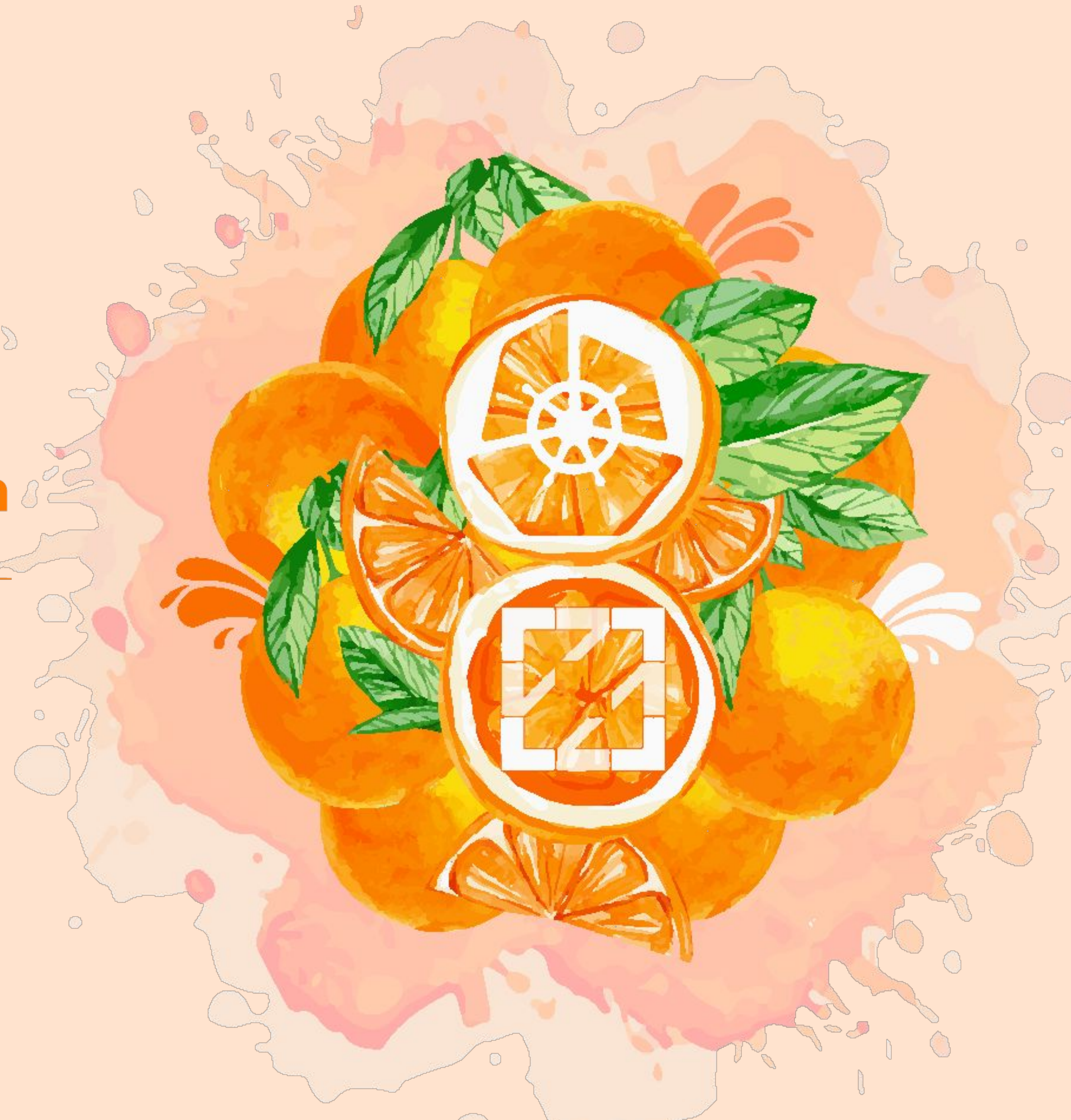
KubeCon | CloudNativeCon

Europe 2022

WELCOME TO VALENCIA



Rook Intro & Ceph Deep Dive

Rook maintainers
Blaine Gardner, Red Hat
Travis Nielsen, Red Hat
Satoru Takeuchi, Cybozu, Inc.
Sébastien Han, Red Hat

Agenda

- Kubernetes Storage Challenges
- What is Rook?
- What is Ceph?
- Rook key features
- Rook v1.9 new features
- Demo
- Q&A

Kubernetes Storage Challenges

- Kubernetes is a platform to manage distributed apps
  - Ideally stateless
- Reliance on external storage
  - Not portable
  - Deployment burden
  - Day 2 operations - who is managing the storage?
- Reliance on cloud provider managed services
  - Vendor lock-in

What is Rook?

What is Rook?

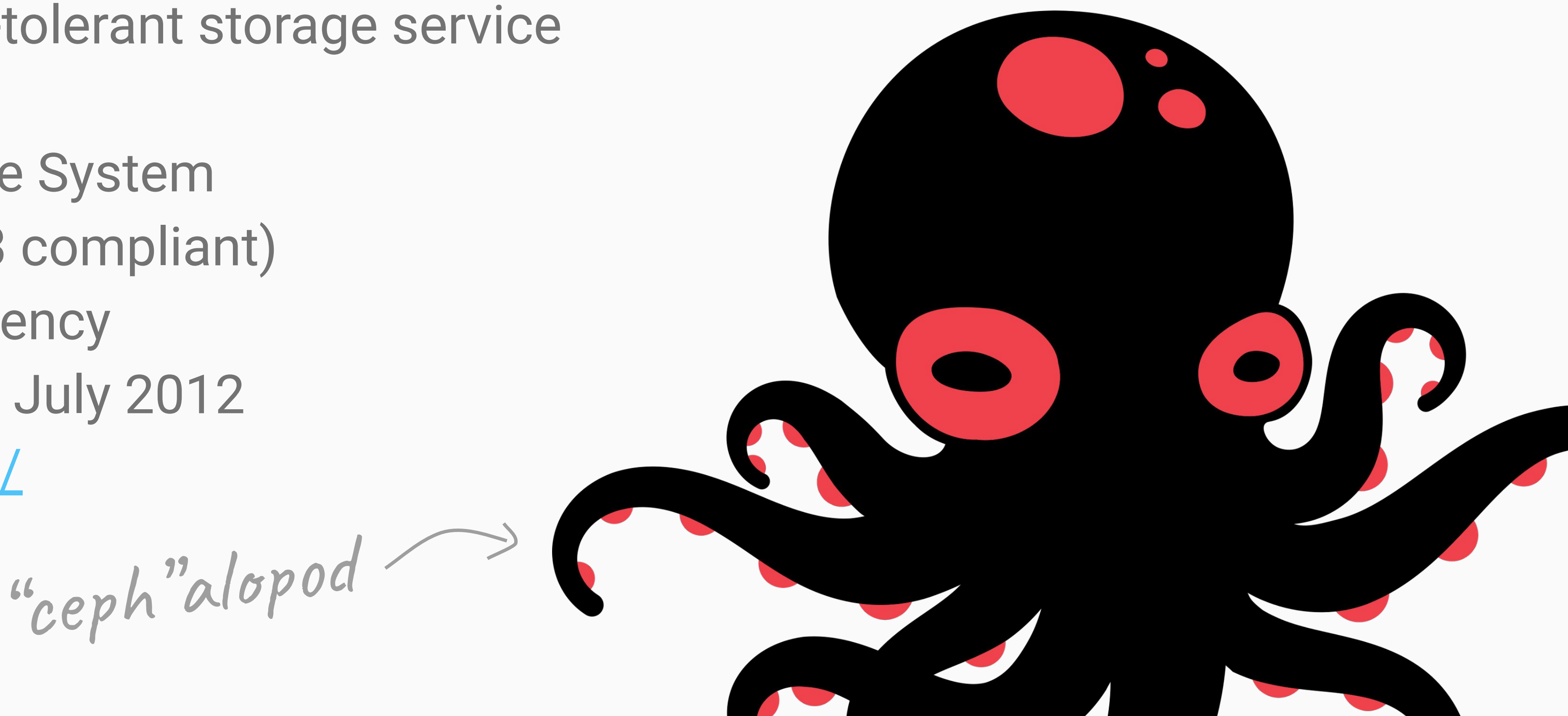
- Makes storage available inside your Kubernetes cluster
- Consume like any other K8s storage
  - Storage Classes, Persistent Volume Claims
- Kubernetes Operators and Custom Resource Definitions
- Automated management of Ceph
  - Deployment, configuration, upgrades
- Open Source (Apache 2.0)

Rook Resources

Website	https://rook.io/
Documentation	https://rook.io/docs/rook/v1.9/
Slack	https://rook-io.slack.com/
Contributions	https://github.com/rook/rook
Twitter	@rook_io
Community Meeting	https://github.com/rook/rook#community-meeting
Training Videos (new!)	https://kubebyexample.com/ → Learning Paths → Storage for Kubernetes with Rook

What is Ceph?

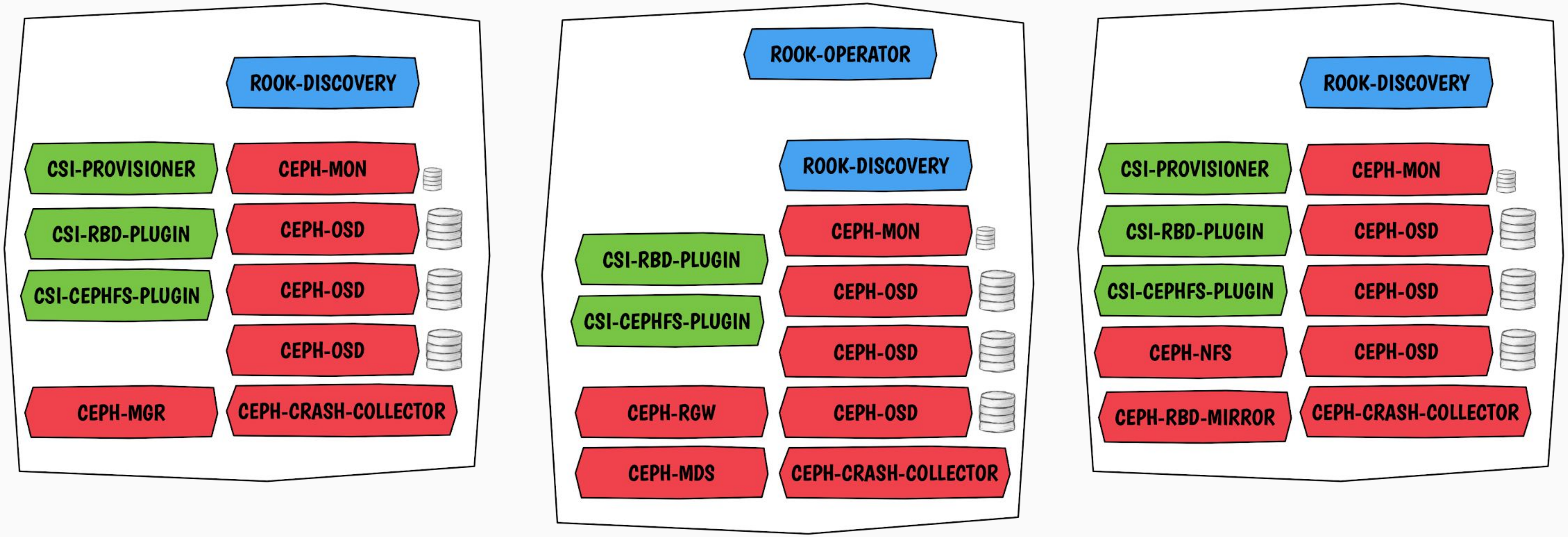
- Open Source
- Scalable, fault-tolerant storage service
    - Block
    - Shared File System
    - Object (S3 compliant)
- Favors consistency
- First release in July 2012
- https://ceph.io/



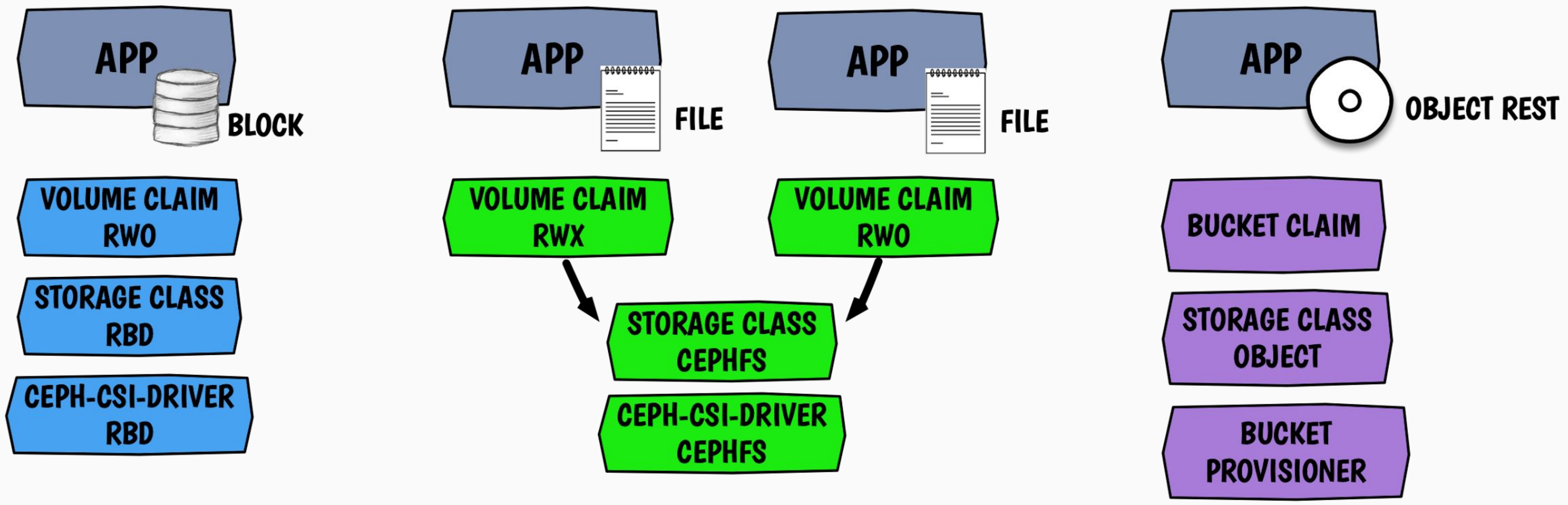
Architectural Layers

- Rook
  - Operator owns the **deployment** and **management** of Ceph and Ceph CSI (Container Storage Interface) driver
- Ceph-CSI
  - CSI driver dynamically **provisions** and **mounts** Ceph storage to user application Pods
- Ceph
  - **Data** layer

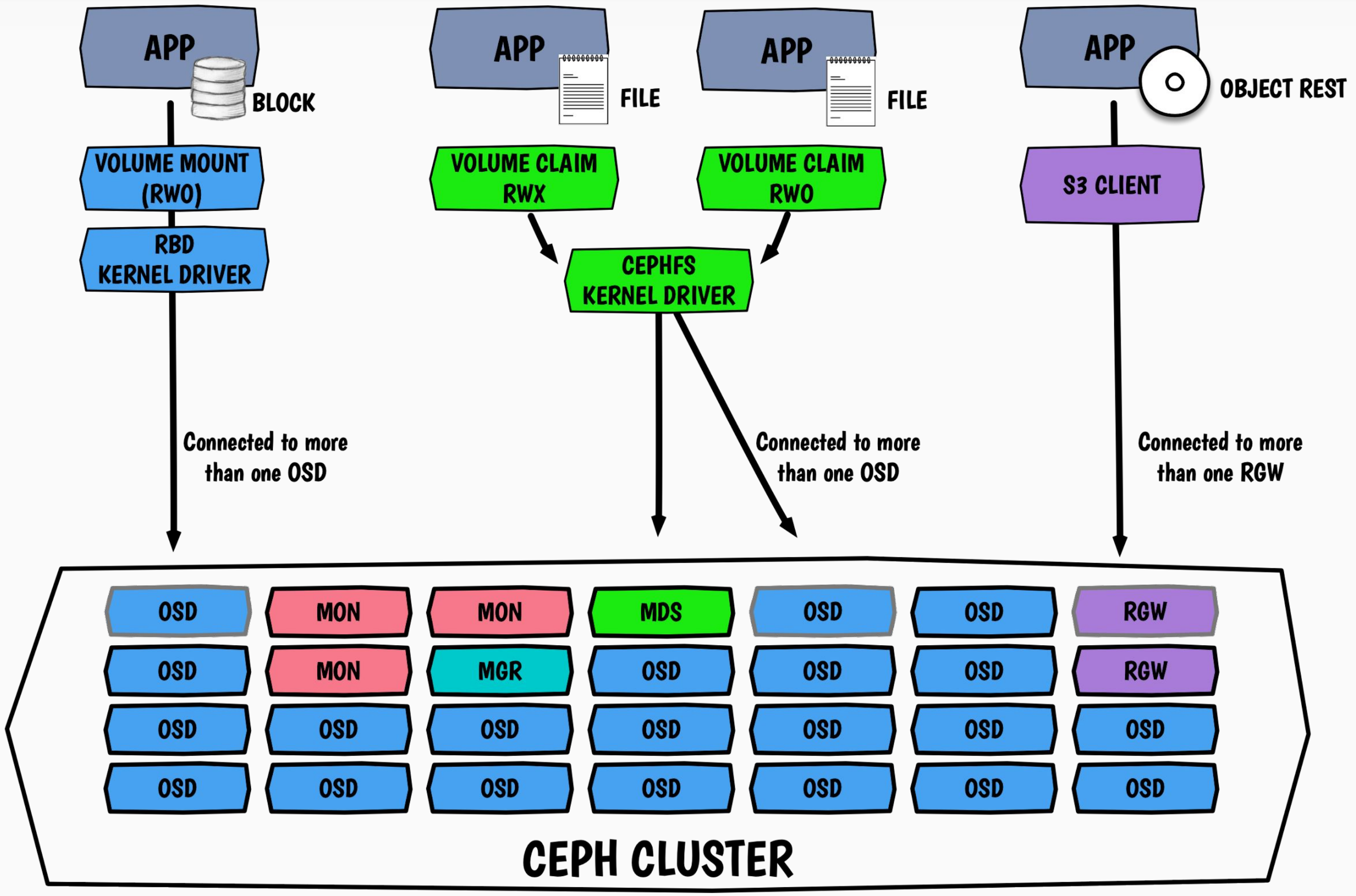
Rook: View of Pod Management



Provisioning storage with CSI



Ceph: Data path



Key Features

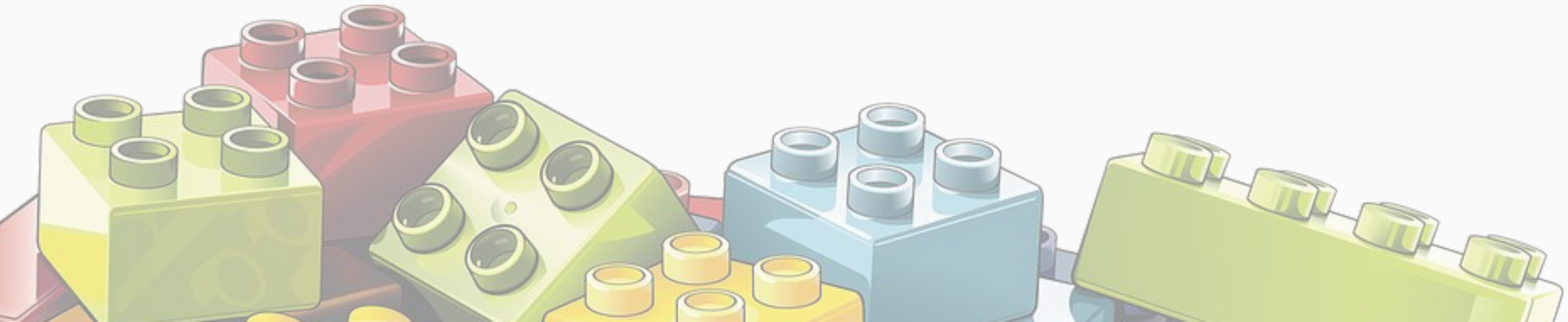
Installing Ceph is simple!

- Create Custom Resource Definitions
  - `kubectl create -f crds.yaml`
- Create authorization (RBAC)
  - `kubectl create -f common.yaml`
- Create the Rook-Ceph Operator
  - `kubectl create -f operator.yaml`
- Create the Ceph cluster resource
  - `kubectl create -f cluster.yaml`

```yaml
apiVersion: ceph.rook.io/v1
kind: CephCluster
metadata:
  name: rook-ceph
  namespace: rook-ceph
spec:
  cephVersion:
    image: quay.io/ceph/ceph:v16.2.5
  mon:
    count: 3
  storage:
    useAllNodes: true
    useAllDevices: true
```

Ceph CSI driver features

- Dynamic provisioning for Block and File storage
- Volume expansion
- Snapshots and Clones



Environments

Bare metal

- Bring your own hardware

Cloud providers

- Overcome cloud provider storage limitations

Rook in a Cloud Environment

- Overcome shortcomings of the cloud provider’s storage
  - Storage across availability zones (AZs)
  - Faster failover times (seconds instead of minutes)
  - Greater number of PVs per node (many more than ~30)
  - Use storage with better performance:cost ratio
- Consistent storage platform wherever K8s is deployed
- Ceph uses PVCs as underlying storage
  - No need for direct access to local devices

Configure for any cluster topology

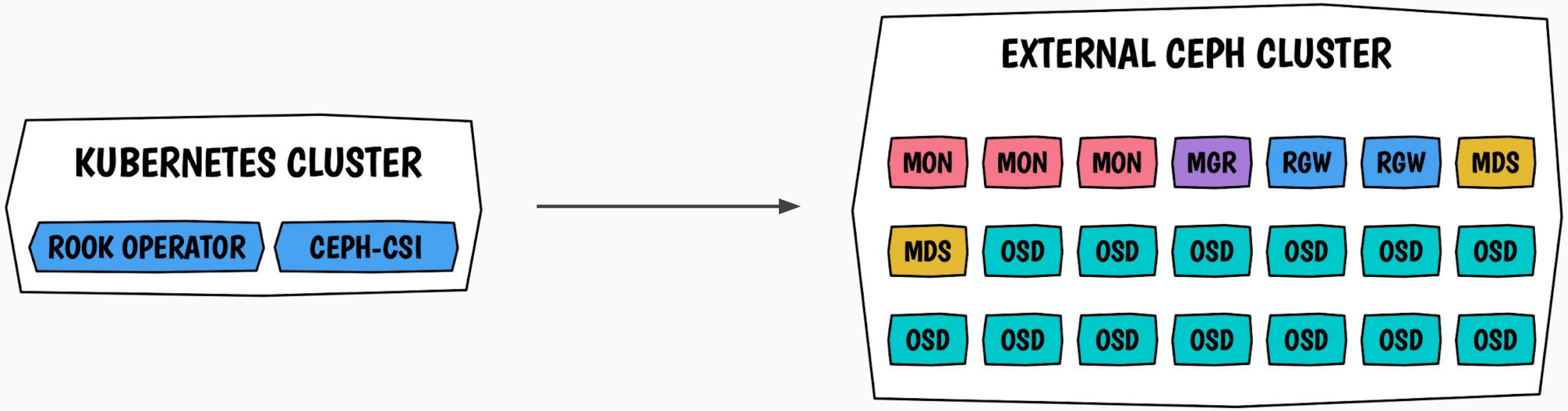
- Customizable across/within cluster topologies
- High availability and durability
  - Spread Ceph daemons and data across failure domains
- Deployable on specific nodes if desired
  - Node affinity, taints/tolerations, etc.

Updates are automated

- Ceph updates and even major upgrades are fully automated!
  - Rook handles *everything*
- Rook patch up*dates* are fully automated
- Rook minor up*grades*
  - Take advantage of latest features
  - Occasional K8s/Ceph/CSI/Rook feature deprecations
  - https://rook.io/docs/rook/latest/ceph-upgrade.html

Connect to an external Ceph cluster

- Connect to a Ceph cluster outside of the current K8s cluster
- Dynamically create Block/File/Object storage consumable by K8s applications



Provision object storage buckets

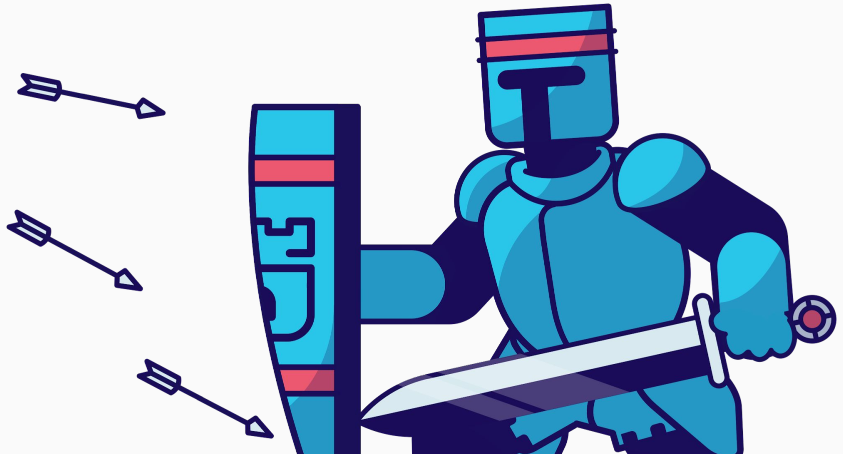
- Define a Storage Class for Ceph object storage
- Create an Object Bucket Claim (OBC)
    - Similar pattern to a Persistent Volume Claim (PVC)
    - Rook operator creates a bucket when requested
    - Give access via K8s Secret
- Container Object Storage Interface (COSI)
    - Kubernetes Enhancement Proposal
    - CSI but for object storage

Rook v1.9 features

April 2022

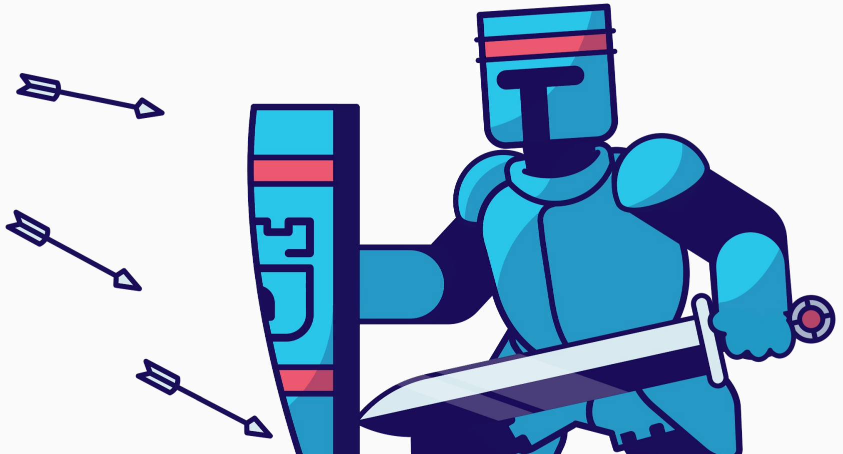
Ceph Quincy Support

- Ceph Quincy (v17) is now supported
- Also released April 2022



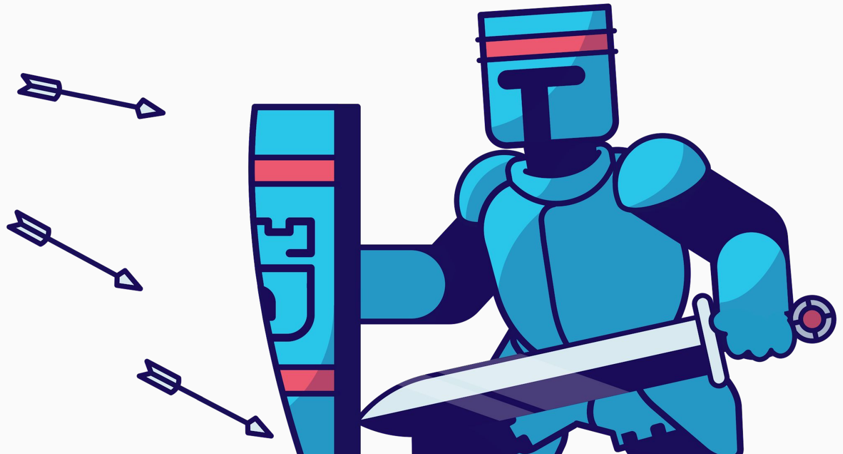
CSI Driver Updates

- Ceph-CSI 3.6 release
- Fuse mount recovery: Detection of corrupt Ceph fuse mounts will be detected and remounted automatically
- AWS KMS encryption: CSI can be configured to use Amazon STS



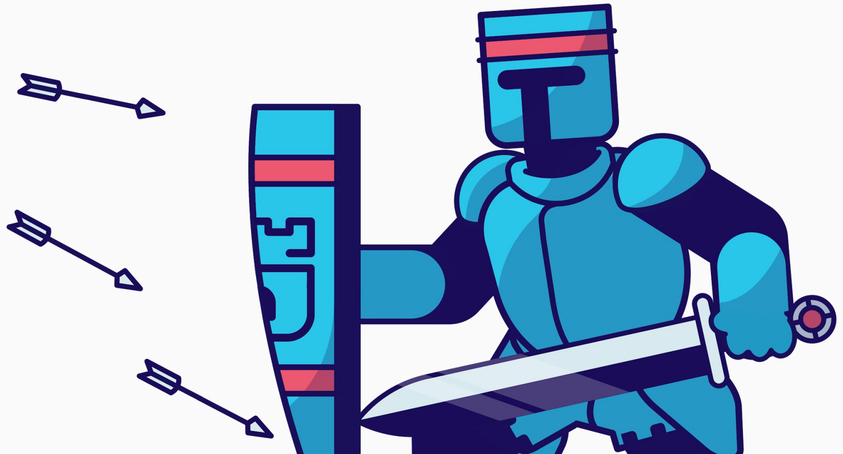
NFS Provisioning

- Create NFS exports via PVCs
- Ceph-CSI driver provisioning
- Mount the volumes with the K8s community NFS driver



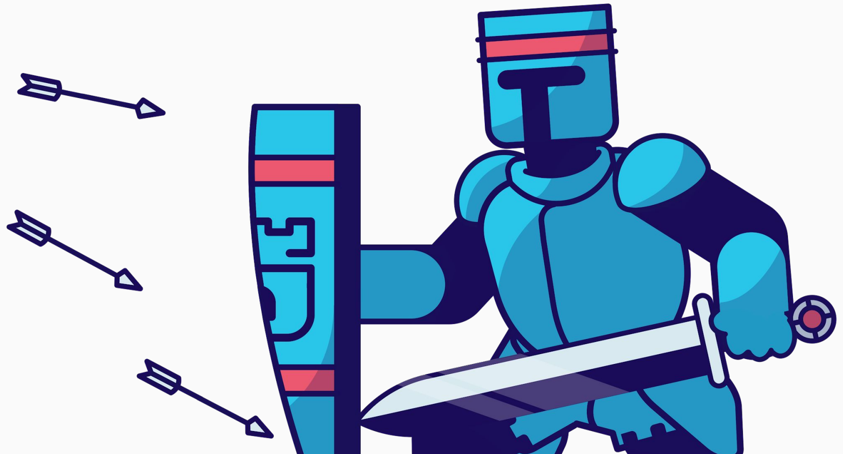
RADOS Namespace CRD

- Create RADOS namespaces in a pool
- Isolation/multi-tenancy without creating separate pools



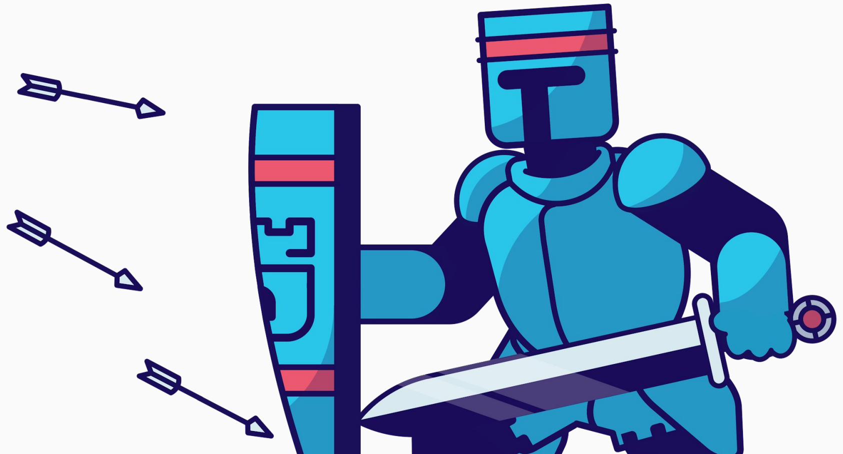
Network Features

- Network features exposed with a simple CephCluster CR setting:
  - Encryption on the wire
  - Compression on the wire
- Recent kernel (5.11) is required



And Much More...

- Admission controller enabled by default if cert manager is available
- Multus networking support
- Updated Prometheus alerts
- ...



Demo

Environment

- OpenShift v4.9.15 (Kubernetes v1.22.3)
- 3 control nodes, 3 worker nodes
- Amazon Web Services m5.8xlarge nodes
  - Run storage with about ~50% room left over for user applications
- Using gp2 for backing volumes
- Rook v1.9.0
- Ceph v17.1.0 (pre-release)

Two types of Rook/Ceph clusters

- Host-based cluster
  - Use disks attached to a node for backing storage
- PVC-based cluster
  - Use Persistent Volume Claims to get backing storage
  - Can be dynamic or local volumes

Host-based cluster

- Suitable for simple cluster

- Storage configuration gets complicated when...
  - Not all nodes/devices are used
  - Using heterogeneous nodes
  - Customizing device layout per-node

```
# ...
storage:
  useAllNodes: true
  useAllDevices: true
```

```
# ...
storage:
  nodes:
  - name: “foo”
    - devices:
      - name: “sdb”
      # ...
```

PVC-based cluster

- No need to describe hardware configuration

- Easy to expand
  - Increase the **`count`**
  - Increase the **`resources.storage`** size

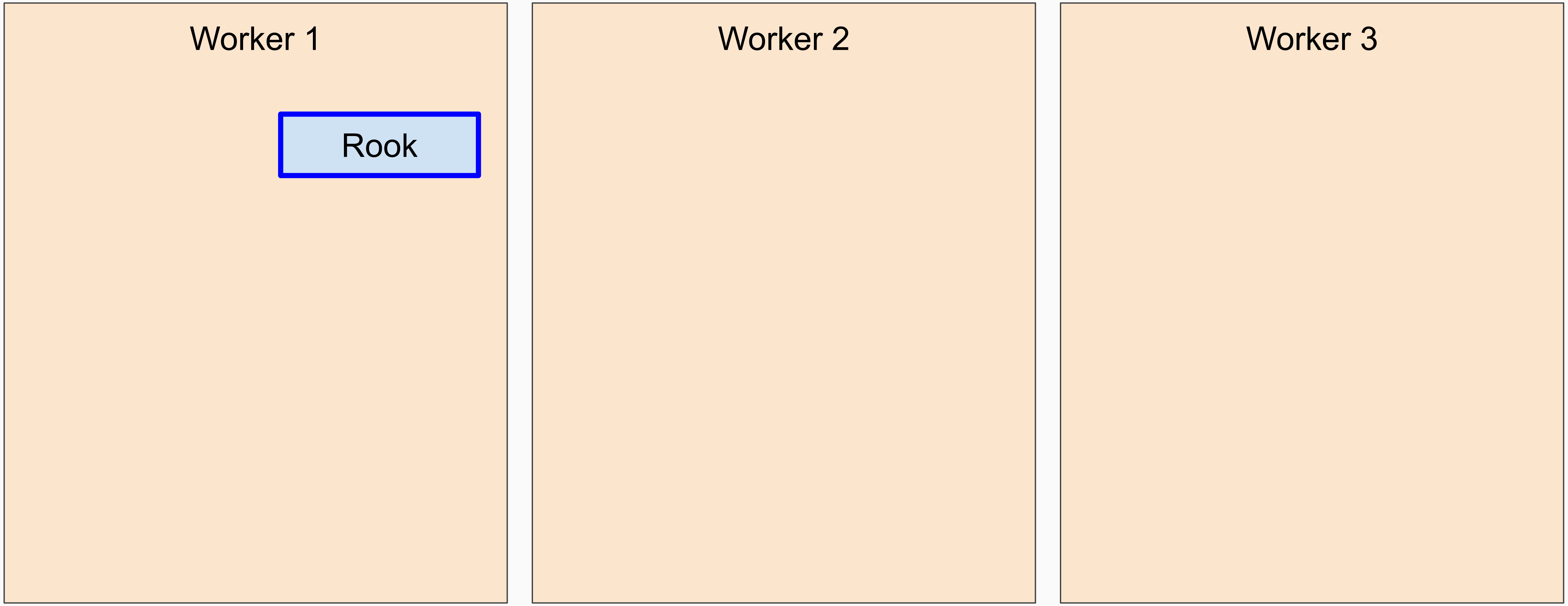
```
# ...
storage:
 storageClassDeviceSets:
  - name: set1
    count: 1
    volumeClaimTemplates:
    - spec:
        resources:
          requests:
            storage: 10Gi
        storageClassName: gp2
        # ...
```

Create a PVC-based cluster

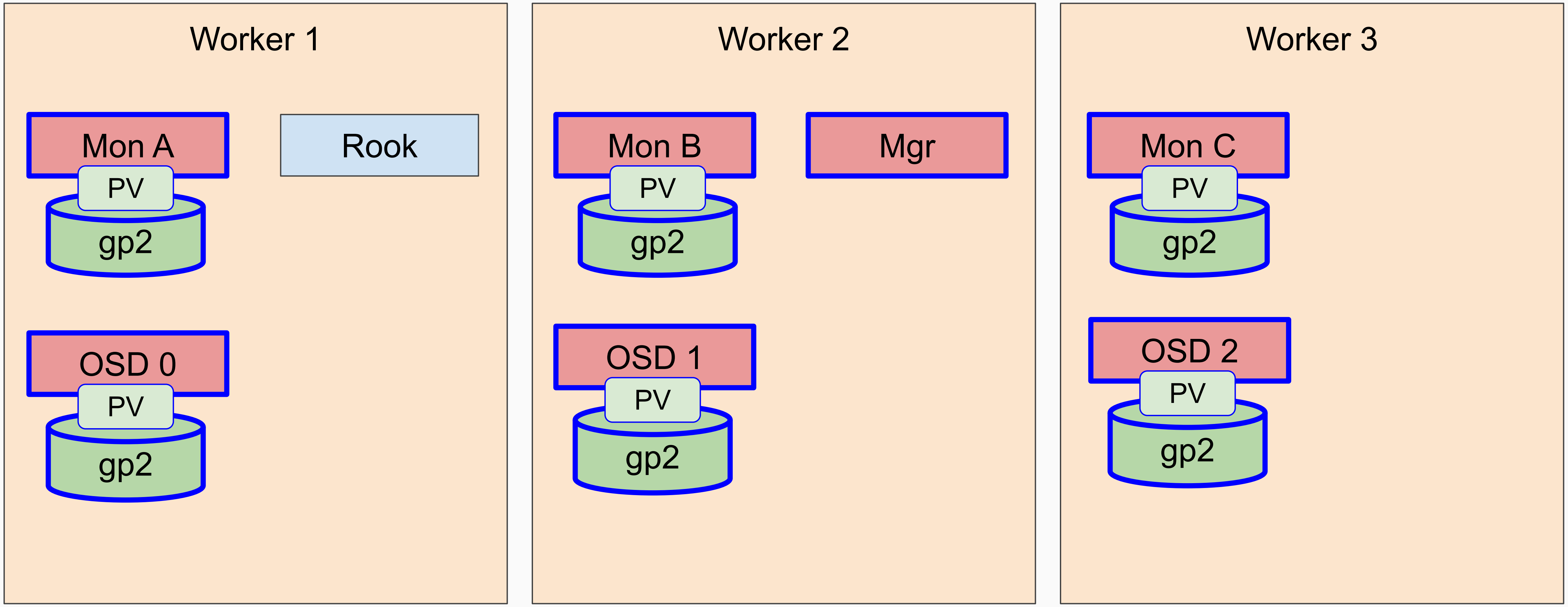
- Steps
  1. Create the Rook operator
  2. Create a Rook-Ceph cluster
  3. Use rook-ceph Krew plugin to see cluster details
  4. Expand the Ceph cluster’s OSD size
  5. Expand the Ceph cluster’s OSD count

- Using some recommended configs for production

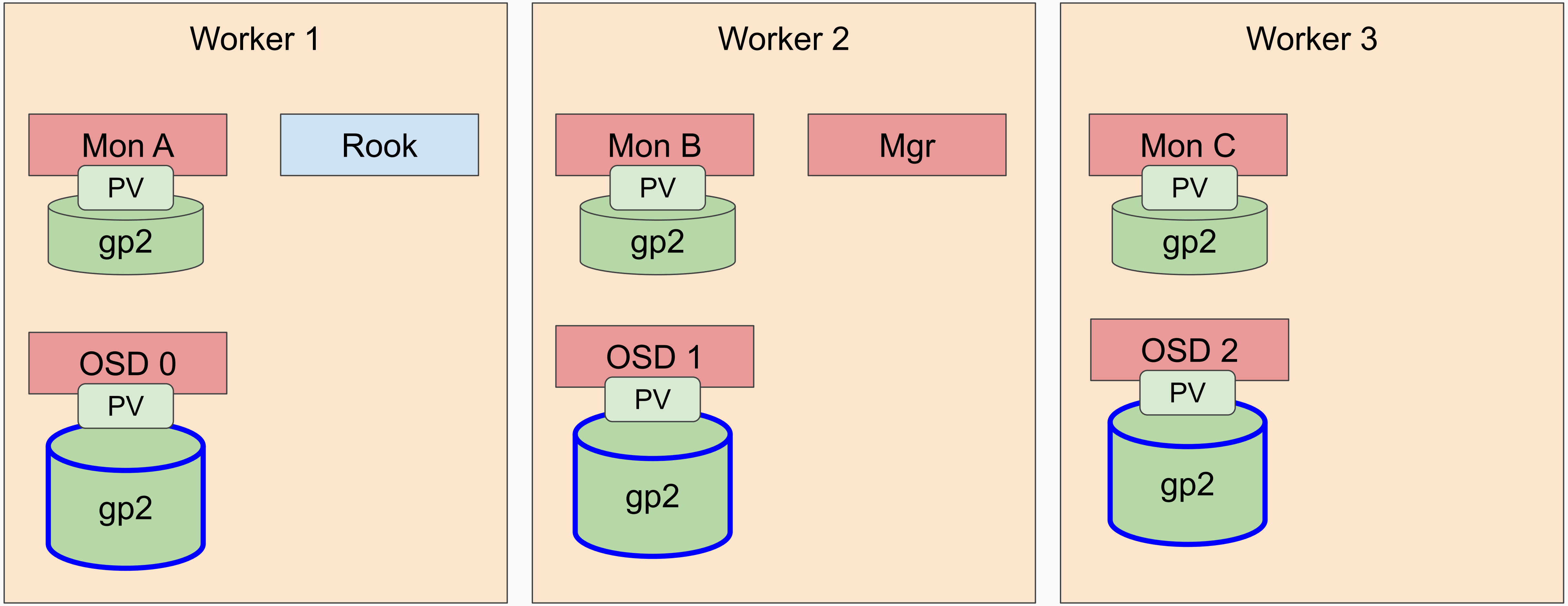
Create the Rook operator



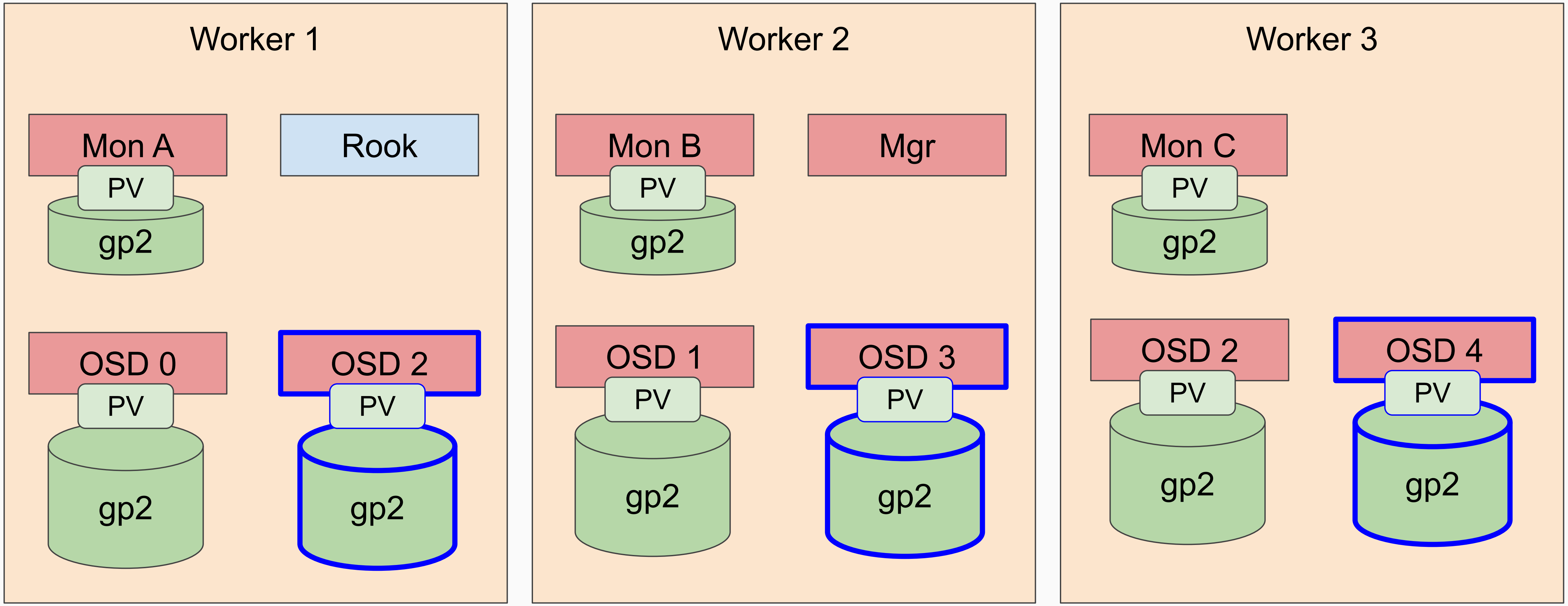
Create a Rook-Ceph cluster



Expand the Ceph cluster's OSD size



Expand the Ceph cluster's OSD count



Questions?

KubeCon | CloudNativeCon
Europe 2022

Website	https://rook.io/
Documentation	https://rook.io/docs/rook/v1.9/
Slack	https://rook-io.slack.com/
Contributions	https://github.com/rook/rook
Twitter	@rook_io
Community Meeting	https://github.com/rook/rook#community-meeting
Training Videos (new!)	https://kubebyexample.com/ → Learning Paths → Storage for Kubernetes with Rook