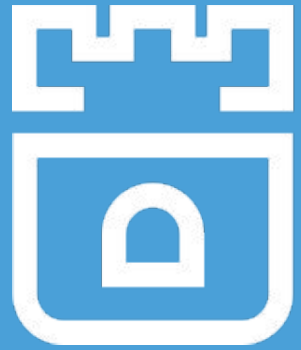


Distributed storage with Rook

Kim-Norman Sahm, Executive DevOps Architect, Cloudibility
Alexander Trost, Rook Core Contributor

Speakers

Kim-Norman Sahm

Executive DevOps Architect, Cloudibility

Alexander Trost (@galexrt)

Rook Core Contributor

Agenda

- What is Rook?
- Architecture
- Kubernetes native Integration
  - StorageClass
  - PersistentVolumeClaim
  - Custom Resource Definitions
- Demo

What is Rook?

- Cloud-Native Storage Orchestrator
- Extends Kubernetes with custom types and controllers
- Automates deployment, bootstrapping, configuration, provisioning, scaling, upgrading, migration, disaster recovery, monitoring, and resource management

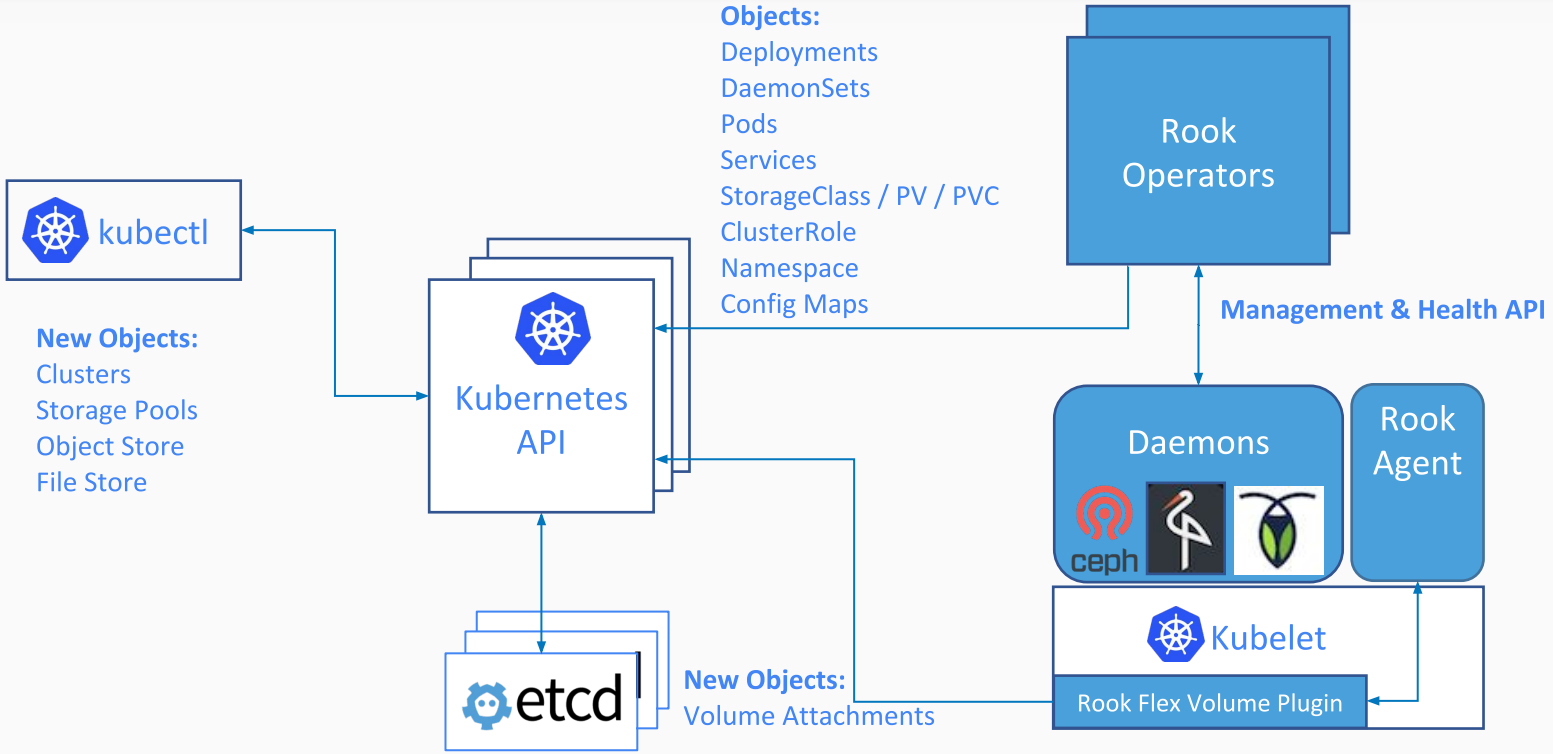
What is Rook?

- Framework for many storage providers and solutions
- Open Source (Apache 2.0)
- Hosted by the Cloud-Native Computing Foundation (CNCF)

Rook Framework for Storage Solutions

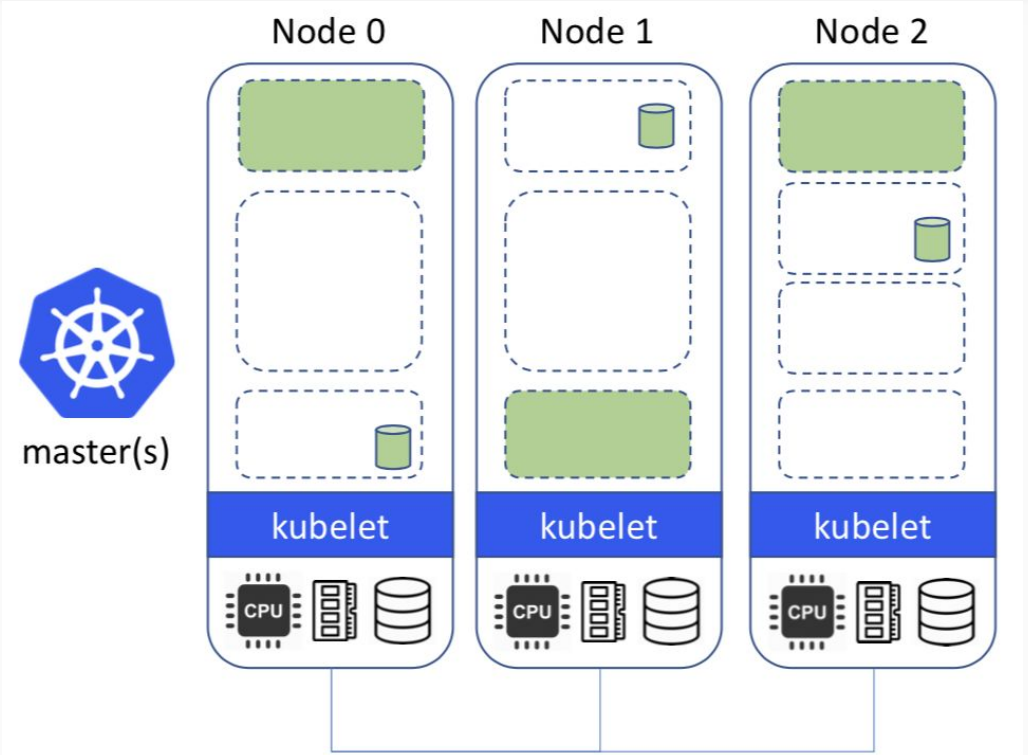
- Rook is more than just a collection of Operators and CRDs
- **Framework** for storage providers to integrate their solutions into cloud-native environments
  - Storage resource normalization
  - Operator patterns/plumbing
  - Common policies, specs, logic
  - Testing effort
- Ceph, CockroachDB, Minio, Nexenta, and more...

Architecture

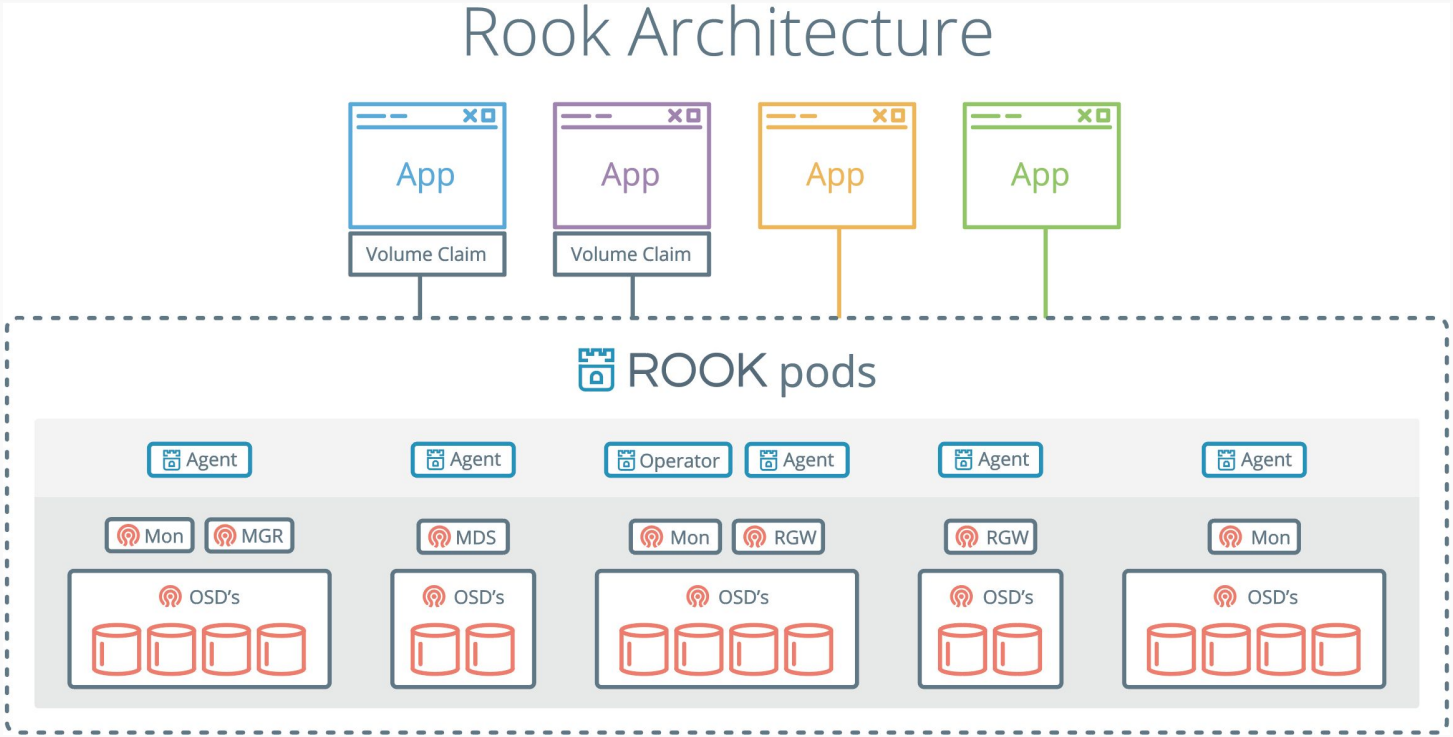


Storage ON Kubernetes

- Kubernetes can manage our storage solution
- Highly portable applications (including storage dependencies)
- Dedicated K8s storage cluster also possible

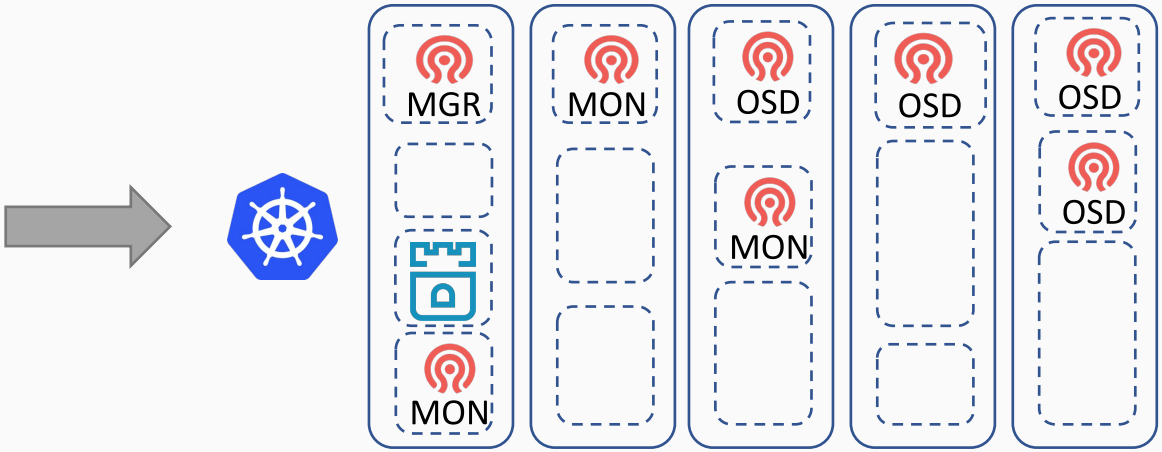


Ceph on Kubernetes with Rook



Ceph on Kubernetes with Rook

```yaml
apiVersion: ceph.rook.io/v1alpha1
kind: Cluster
metadata:
  name: rook-ceph
  namespace: rook-ceph
spec:
  dataDirHostPath: "/var/lib/rook"
  network:
    hostNetwork: false
  storage:
    useAllNodes: true
    useAllDevices: false
    deviceFilter:
    location:
    config:
      storeType: "bluestore"
```



Kubernetes Native Integration

StorageClass

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
   name: rook-block
   namespace: rook
provisioner: rook.io/block
parameters:
  pool: replicapool
  clusterName: rook
```

PersistentVolumeClaim

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: wp-pv-claim
  labels:
    app: wordpress
spec:
  storageClassName: rook-block
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 20Gi
```

Results in..

PersistentVolume
20 Gigabyte

Custom Resource Definitions (CRDs)

- Teaches Kubernetes about new first-class objects
- Custom Resource Definition (CRDs) are arbitrary types that extend the Kubernetes API
  - Look like any other built-in object (e.g. Pod)
  - Native `kubectl` experience enabled
- A means for user to describe their desired (custom) state

Rook Cluster CRD

```
apiVersion: ceph.rook.io/v1alpha1
kind: Cluster
metadata:
  name: rook-ceph
  namespace: rook-ceph
spec:
  dataDirHostPath: "/var/lib/rook"
  network:
    hostNetwork: false
  storage:
    useAllNodes: true
    useAllDevices: false
    deviceFilter:
    location:
    config:
      storeType: "bluestore"
```

Demo

Demo

Deploying a Ceph Cluster with a Stateful Application

How to get involved?

- Contribute to Rook
  - https://github.com/rook/rook
  - https://rook.io/
- Slack - https://rook-io.slack.com/
  - #conferences now for ContainerDays
- Twitter - @rook_io
- Forums - https://groups.google.com/forum/#!forum/rook-dev
- Community Meetings

Questions?

https://github.com/rook/rook

https://rook.io/

Thank you!

https://github.com/rook/rook

https://rook.io/