KubeCon CloudNativeCon India 2026

Rook: Storage for Kubernetes

Deepika Upadhyay, Clyso
Madhu Rajanna, Rewant Soni, Malay Parida, Pratik Surve, IBM

Agenda

- Introduction to Rook and Ceph
- New Features
- Ceph-CSI Driver
- Erasure Coding with Rook: Evolution
- Application Disaster Recovery

Introduction to Rook

What is Rook?

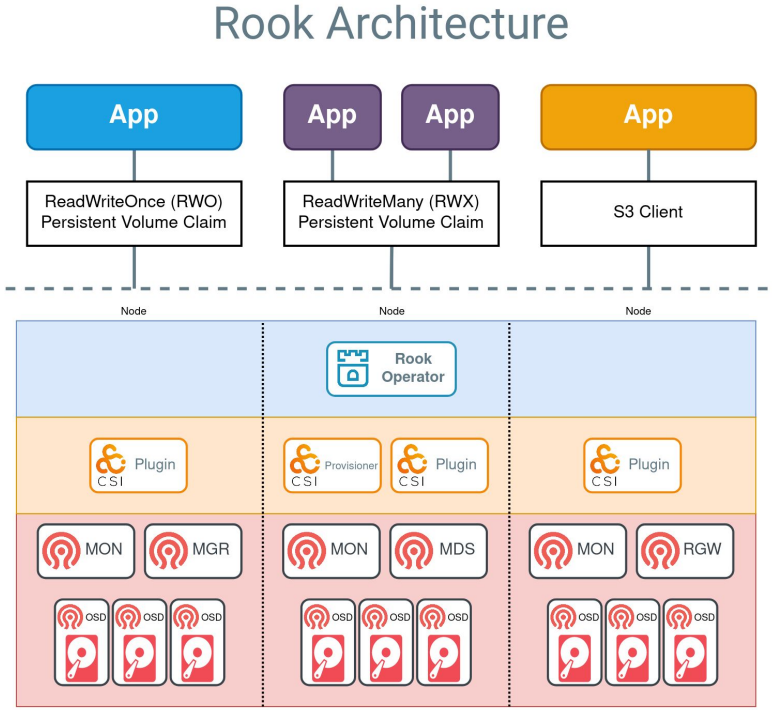
- Brings Ceph storage into your Kubernetes cluster
- Manages Ceph storage with an operator and CRDs
- Automated deployment, configuration, upgrades
- Allows apps to consume storage like any other K8s storage
  - Storage Classes, PVCs
- Open Source (Apache 2.0)
- Happy 10th Birthday!
  - Created November 2016

Ceph: Open-source, distributed Enterprise storage platform

- **Block** (RWO)
  Ceph RBD
- **File** (RWX)
  CephFS
- **Object** (S3)
  Ceph RGW

Architectural Layers

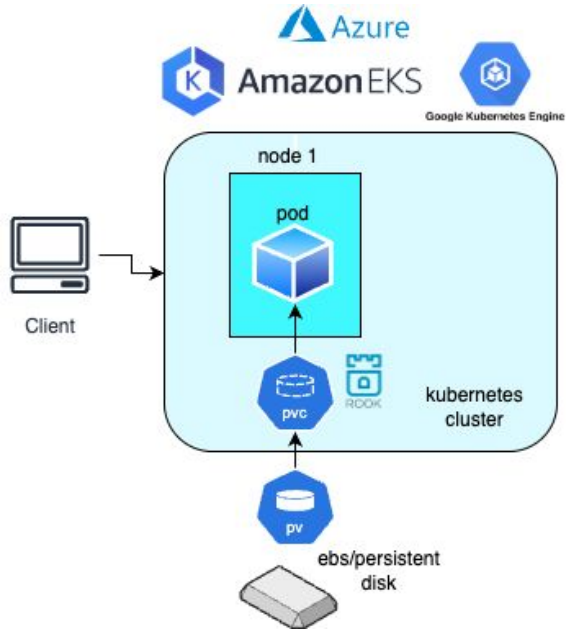
- Rook
  - Operator **deploys** and **manages** Ceph
- CSI
  - Ceph CSI driver dynamically **provisions** and **mounts** storage to user application pods
- Ceph
  - **Data** Layer



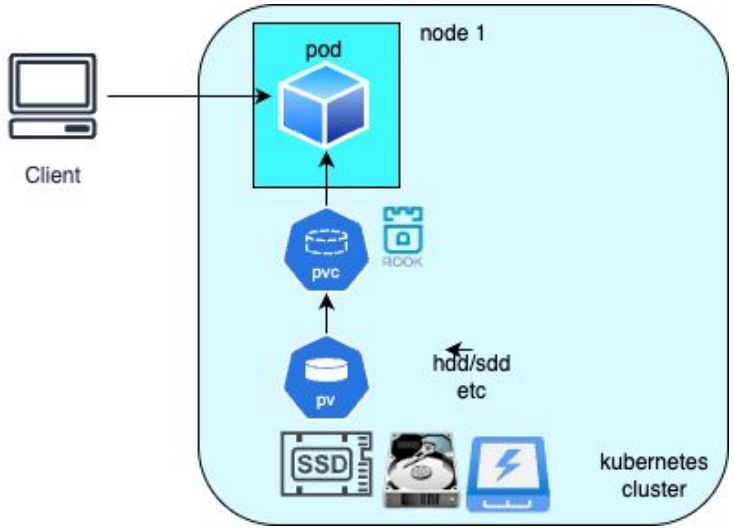
Installation Environments

Anywhere Kubernetes runs!

Installation Environments

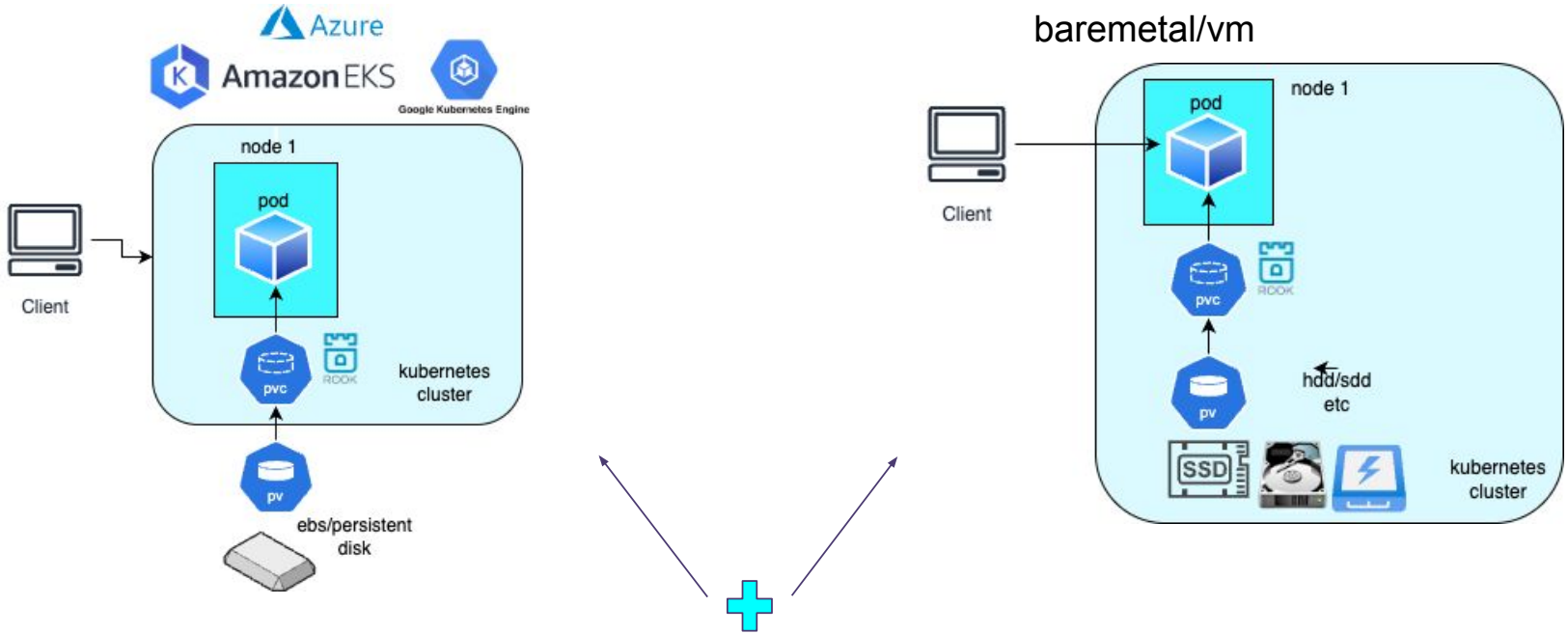


Running in the Cloud → (**EBS, Persistent Disks**, etc.) for flexibility



On-Premises/Bare Metal → Use local **SSDs/HDDs** for performance & control.

Installation Environments



🌍 **Hybrid/Multi-Cloud** → Mix & match for resilience.

Why choose Rook

KubeCon CloudNativeCon India 2026

Challenge	How Rook Helps
🚫 **Fear of data loss:** No Storage Across AZs	✅ Replicates & distributes storage across AZs
🌍 Lack of Cross-Cloud Support	✅ Unified storage layer across multiple cloud providers
🔢 Limited PVs Per Node, Poor Performance of Small PVs	✅ unlimited scaling with dynamic and optimized data placement

Ceph at Scale

✅ **Add capacity without downtime**

- Ceph is designed to scale up and out

✅ **Thin provisioning = Efficient storage use**

- Adding storage capacity is independent from PVCs

Recent Features

Ceph-CSI Operator

- Enabled by default with Rook in v1.18+
- Provides configuration via CRDs for the Ceph-CSI drivers
  - RBD, CephFS, NFS, NVMe-oF
- Rook v1.20
  - All Ceph-CSI configuration is done via CRs instead of with Rook operator settings

RGW Accounts

- Configures RGW accounts for accessing the S3 API
- CephObjectStoreAccount CRD
- Experimental in Rook v1.20

NVMe-oF Gateway

- NVMe over Fabrics
  - Allows Ceph RBD(Block) volumes to be accessed via the NVMe/TCP protocol
- External clients outside the cluster can connect to Ceph block storage using **standard NVMe-oF initiators**
- Delivers low-latency, high-throughput block storage access over existing network infrastructure.
- Experimental
  - We need your feedback!

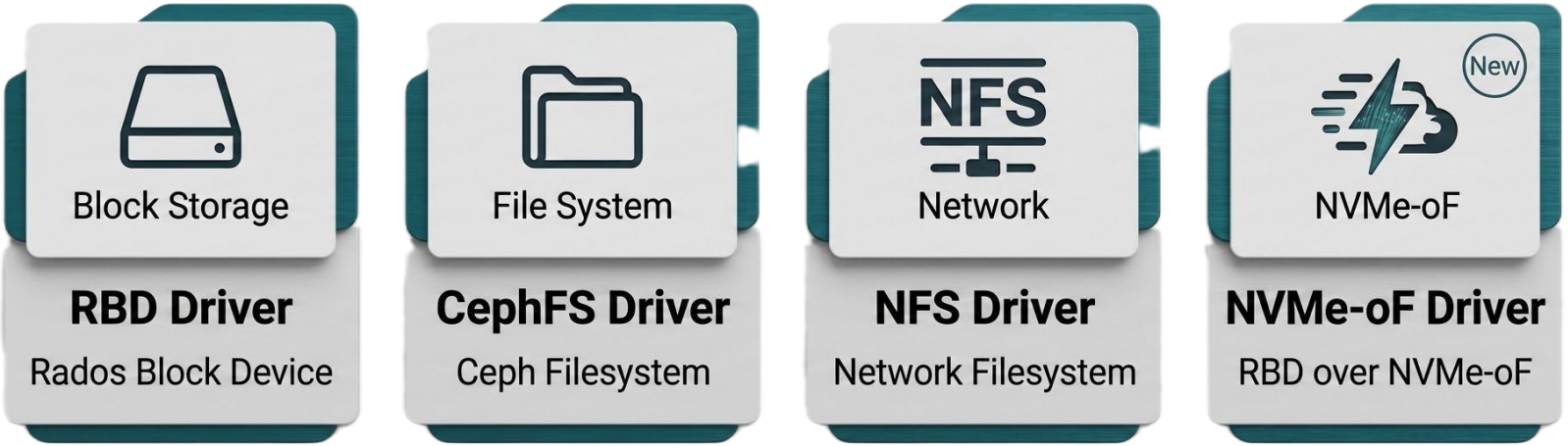
Two-Node Clusters

- Minimal clusters in edge scenarios
- Rook runs three mons where one of them “floats” between the two nodes/zones for resiliency
- Requires fencing solution with Pacemaker
- Attend the talk tomorrow at 14:30 dedicated to this topic!
  - “When the Edge Can’t Afford a Third Node: A Storage Solution for Two-Node Kubernetes Clusters” by Parth Arora

Ceph-CSI Driver

Ceph-CSI Driver

Ceph Container Storage Interface (**Ceph-CSI**) project hosts **CSI Drivers** for Ceph



Volume Mode recommendations

IN-CLUSTER WORKLOAD USAGE

RBD (Rados Block Device)

- RWX BlockMode: KubeVirt VMs, supports Live Migration
- RWO Filesystem (ext4): Databases & small file workloads

CephFS (Ceph Filesystem)

- RWX Filesystem mode: Simultaneous multi-node file access

EXTERNAL SYSTEMS / LIMITED KERNEL DRIVER

NFS (Network Filesystem)

- CephFS volumes exposed over over NFS

NVMe-oF (RBD over NVMe)

- RBD volumes exposed over NVMe-of

Ceph-CSI Internals

Single container-image that supports all different drivers

RADOS OMAP Usage for state and checkpointing

Ceph-CSI extensively makes use of **RADOS OMAP** to store state information of volumes and snapshots.

Go-Ceph: Go bindings for Ceph APIs

The **Go-Ceph** project is a collection of API bindings that support the use of native Ceph APIs, which are C language functions, in Go.

Standard Ceph-CSI Features

Kubernetes CSI features:

- PVC expansion
- Create and restore from Snapshots, clone volumes
- Topology aware
  - Read from the OSD nearest the client
- Ephemeral volumes

Non Kubernetes-native:

- PVC encryption
  - LUKS (RBD), fscrypt (ext4, CephFS)
  - Various Key Management Systems (KMS) such as Vault, Azure, IBM HPCS, KMIP etc is supported.

Recently added Ceph-CSI Features

VolumeGroup Snapshots
Label PVCs to create consistency groups.

RBD Block Usage Metrics
Size of the RBD-image and number of blocks allocated.

RBD Change Block Tracking (CBT)
Enables efficient differential backups by comparing snapshots.

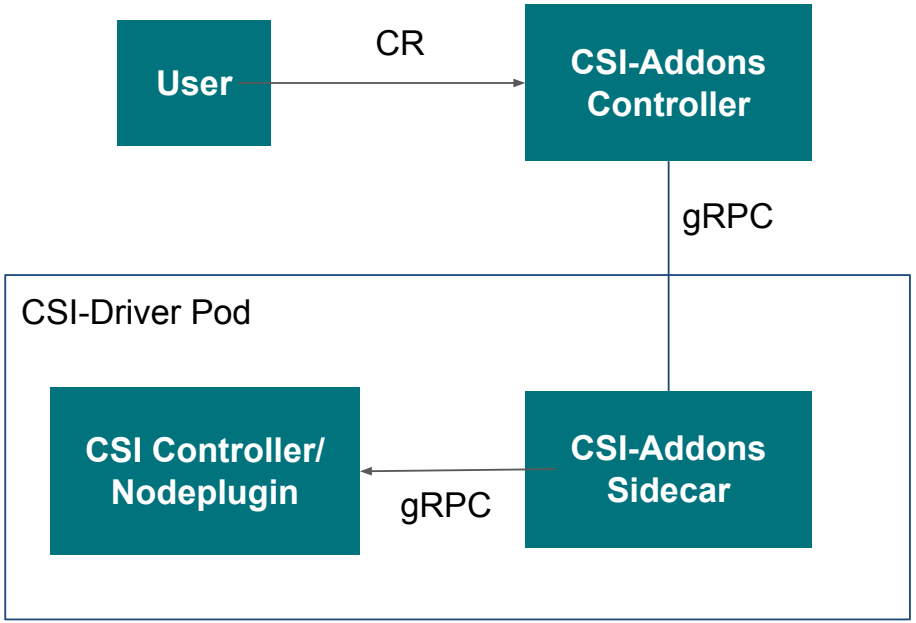
Auto Fencing Support
Prevents data corruption by blocking misbehaving clients.

CSI-Addons

CSI-Addons hosts the extensions to the CSI specification that provides advanced storage operations.

Various components involved:

- CSI-Addons Controller
- CSI-Addons Sidecar
- CSI-Driver



CSI-Addons features in Ceph-CSI

Reclaim Space Operation:

- **Reclaim Space** operation executes **`rbd sparsify`** on images and **`fstrim`** on filesystem mode volumes

Network Fence/Class Operation

- **Network Fence Class** operation to advertise the client IP on the node required for Network Fencing
- **Network Fence** operation provides an API for blocking a list of given CIDR IP ranges
- This plays a critical role in **Metro Disaster Recovery** and **Node-Loss** scenarios

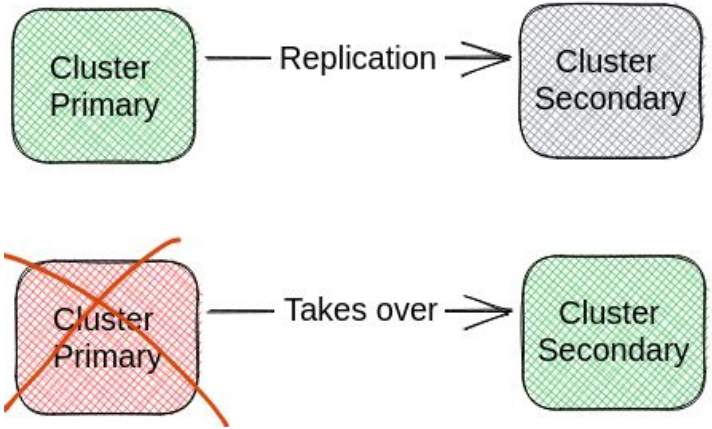
Encryption Key Rotation:

- Rotation of Key Encryption Keys(KEKs) for encrypted volumes.

CSI-Addons features in Ceph-CSI

Volume/VolumeGroup Replication Operation

- **Volume/VolumeGroup Replication** operation provides common and reusable APIs for storage disaster recovery
- It allows enabling/disabling mirroring and changing state(primary/secondary) of rbd mirrored group/images
- The volume/volume-group replication operation automates rbd-mirroring, allowing **promote**, **demote**, **resync** and get volume replication information operations on rbd group/images
- This plays a critical role in **Regional Disaster Recovery**.



Ceph CSI Operator

Ceph CSI Operator

KubeCon CloudNativeCon India 2026

Purpose
A native Kubernetes operator designed specifically to manage CephCSI plugins.

Automation & Management
It **automates** the deployment, configuration, and management of storage drivers using **Custom Resource Definitions (CRDs)**.

Lifecycle Decoupling
Decouples client-side CSI driver lifecycle from the core storage backend orchestrator.

Rook Integration
Enabled by default with Rook in **1.18+**. In **v1.20+**, configuration is handled **declaratively** via CRs.

Ceph CSI Operator

Operator Config

Manages operator-level configurations and offers a place to overwrite settings for CSI drivers.

Driver

Manages the installation, lifecycle management, and configuration for CSI drivers.

Ceph Connection

Stores connection and configuration details for a Ceph cluster.

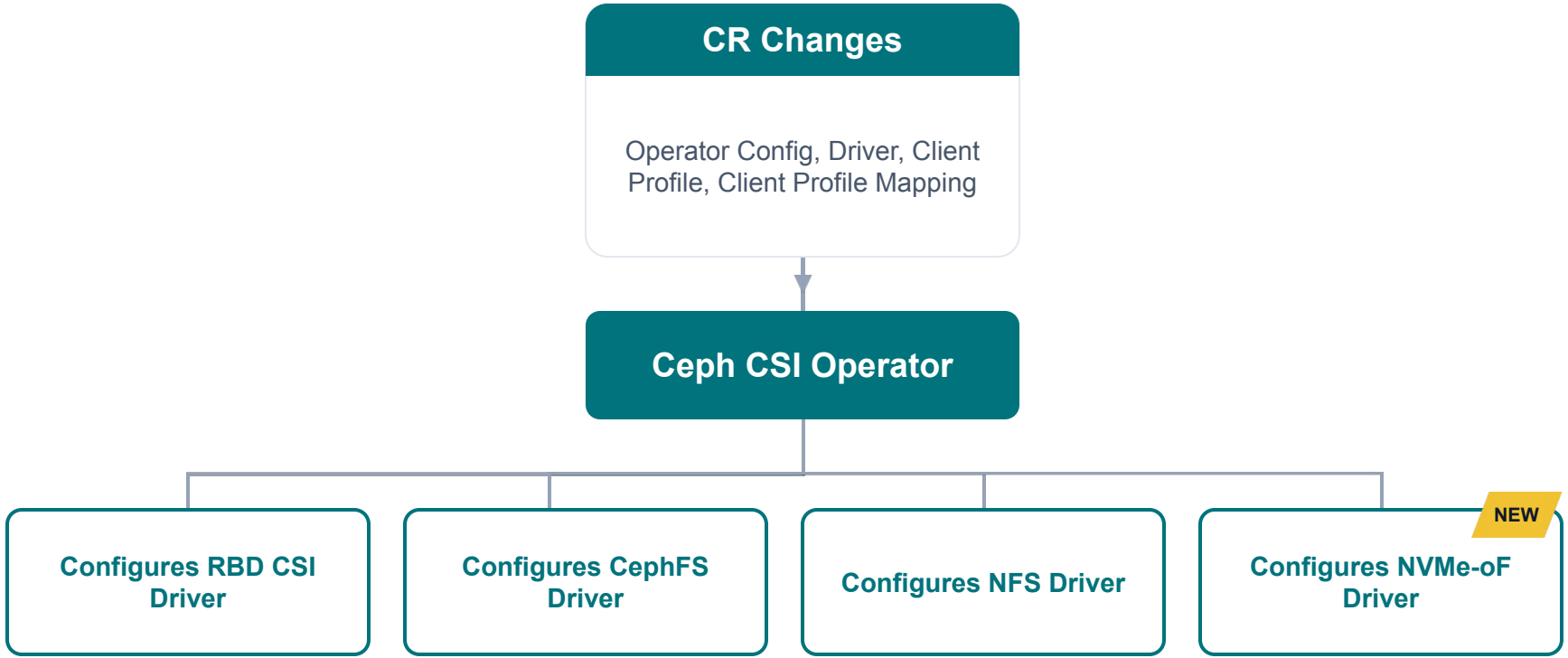
Client Profile

Contains details about CephFS, RBD, NFS, and NVMe-oF configuration used for Ceph communication.

Client Profile Mapping

CR contains mapping between pairs of Ceph CSI client profiles from different clusters for backup and DR scenarios.

Ceph CSI Operator



KubeCon

CloudNativeCon

India 2026

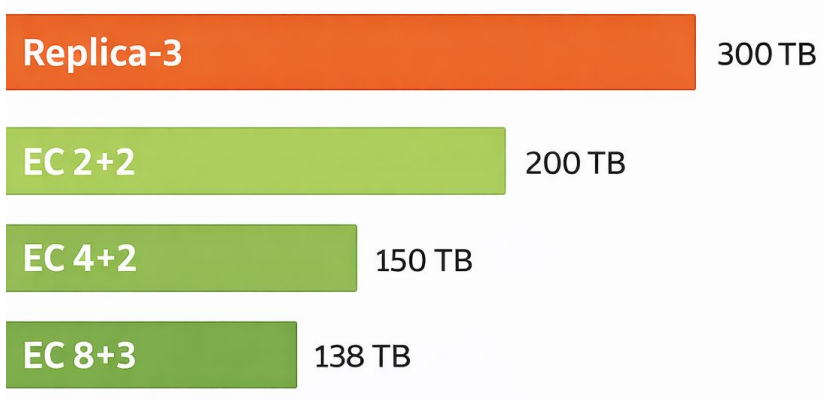
Erasure Coding with Rook: Evolving & Accelerating

Why Erasure Coding?

KubeCon CloudNativeCon India 2026

Why make 3 copies when maths can do the job?

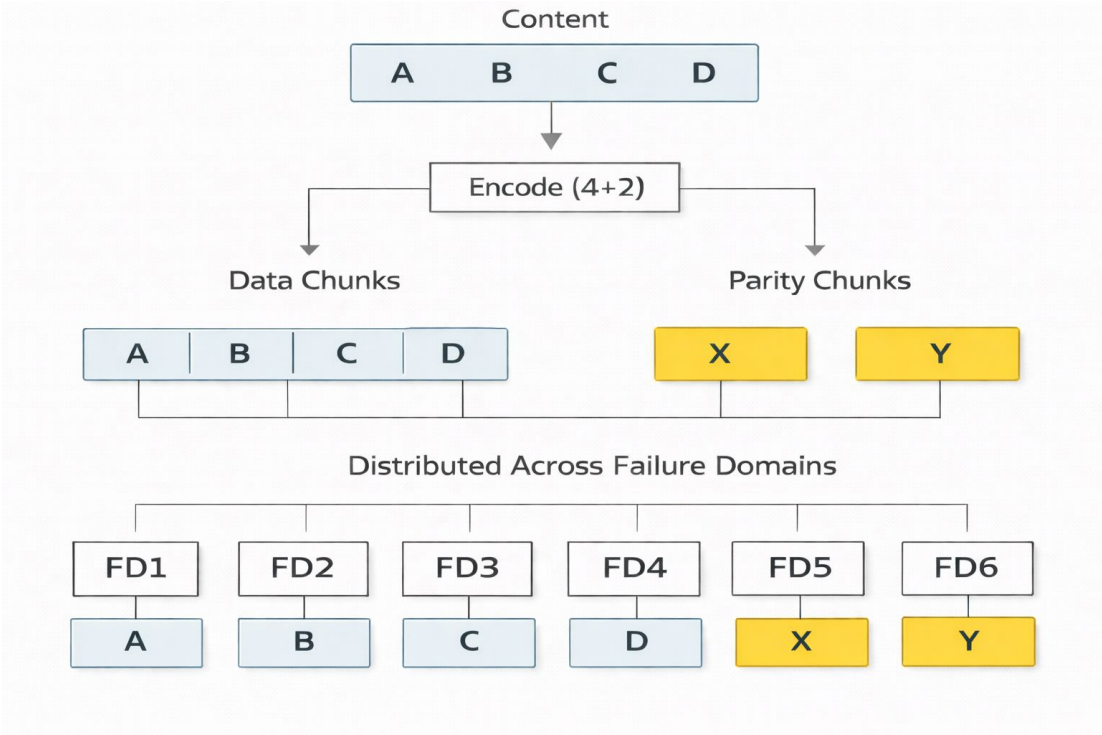
- Most Ceph clusters use **replica-3** for durability meaning **3x storage requirement**
- Erasure Coding provides **similar durability** with significantly lower storage overhead
- **Same durability, ~50% less storage (EC 4+2 vs Replica-3)**



How Ceph Erasure Coding works

KubeCon CloudNativeCon India 2026

Data is split into chunks and extra parity chunks are generated for recovery.



EC Profile: 4+2

k = 4 data chunks
m = 2 coding chunks

Minimum failure domains:

k + m = 6

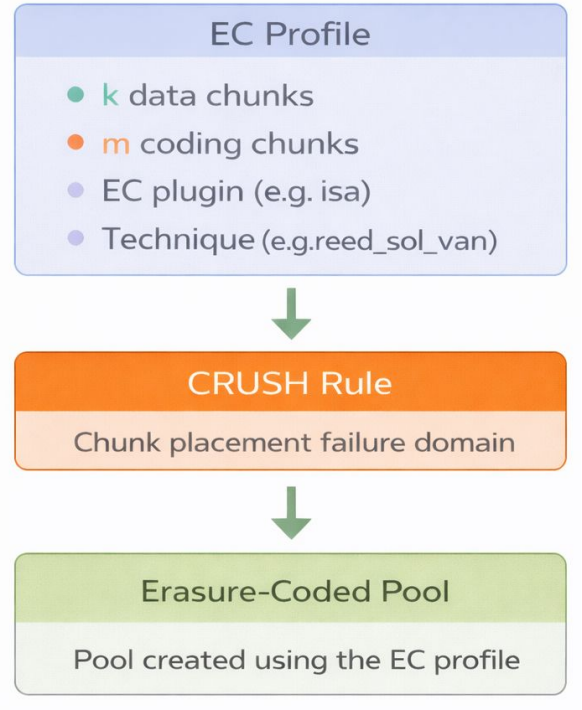
Failure tolerance:

m = 2

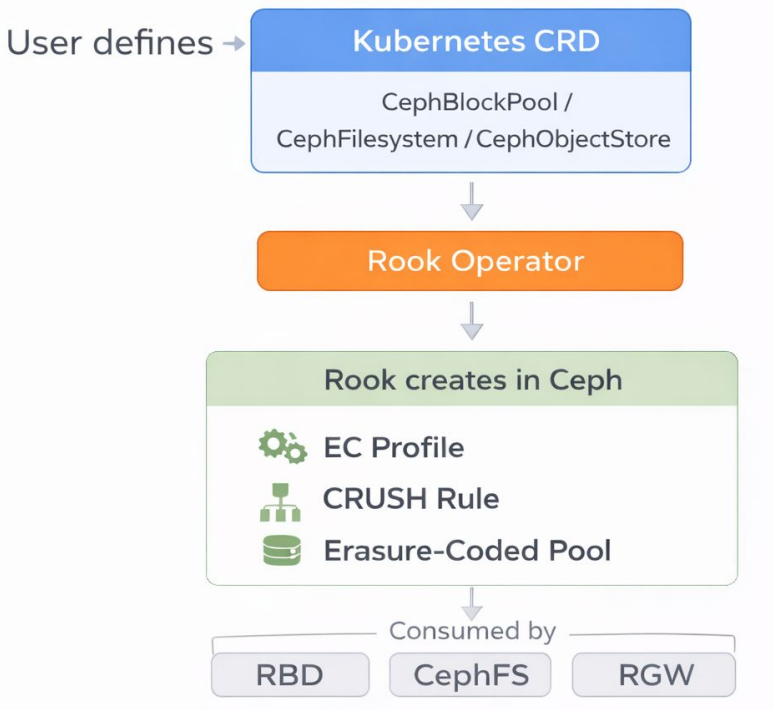
Setting up Erasure Coding with Rook

KubeCon CloudNativeCon India 2026

Ceph Erasure Coding Requires



Rook Simplifies this



Configuring Erasure Coding Example

Erasure-coded pools do not support OMAP as of now. Metadata stored by RBD, CephFS and Object requires a separate replicated metadata pool.

Metadata Pool

Metadata Pool

```yaml
apiVersion: ceph.rook.io/v1
kind: CephBlockPool
metadata:
  name: replicated-metadata-pool
  namespace: rook-ceph
spec:
  failureDomain: host
  replicated:
    size: 3
```

EC Data Pool

EC Data Pool

```yaml
apiVersion: ceph.rook.io/v1
kind: CephBlockPool
metadata:
  name: ec-data-pool
  namespace: rook-ceph
spec:
  failureDomain: host
  erasureCoded:
    dataChunks: 4
    codingChunks: 2
  parameters:
    allow_ec_optimizations: "true"
```

Architecture Flow

To Consider Before Using EC

KubeCon CloudNativeCon India 2026

Performance trade-off

- Higher **CPU usage** due to chunking & parity computation
- **Slower writes** than replication
- Not ideal for small or **latency-sensitive** workloads

Choose K+M carefully

- Determines capacity efficiency and failure tolerance
- Requires **K + M failure domains**; cannot be changed later
- Use **M ≥ 2** for production

Workload suitability

Ideal:

archives, backups, media, ML datasets, cold storage

Not Ideal:

databases, VM disks, transactional systems

“Fast EC”: Ceph Tentacle v20

KubeCon CloudNativeCon India 2026

Ceph v20 introduced major performance improvements for EC pools

Partial Reads

Reads only required shards instead of full stripe reconstruction, significantly improving small-read latency.

Partial Writes

Updates only affected chunks instead of rewriting the entire stripe, saving on drive operations.

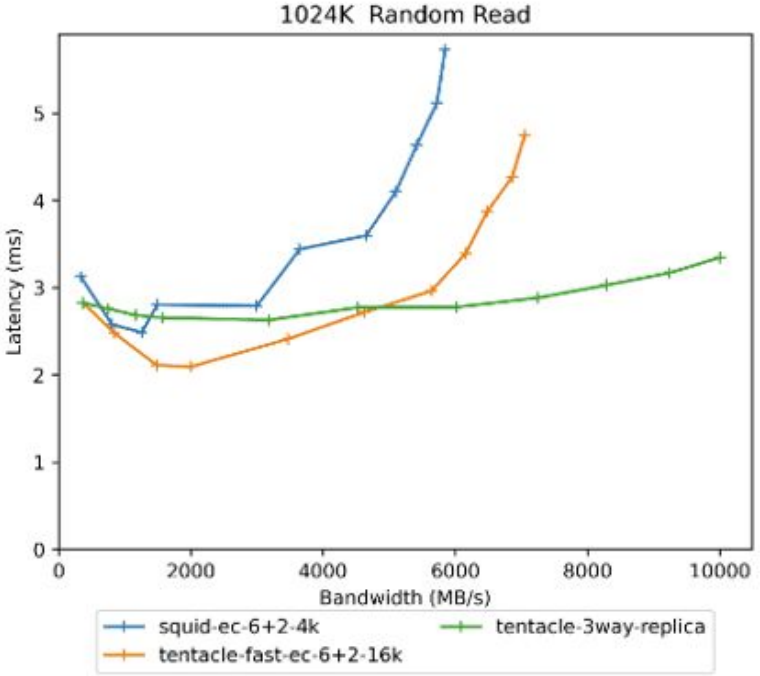
Parity Delta Writes

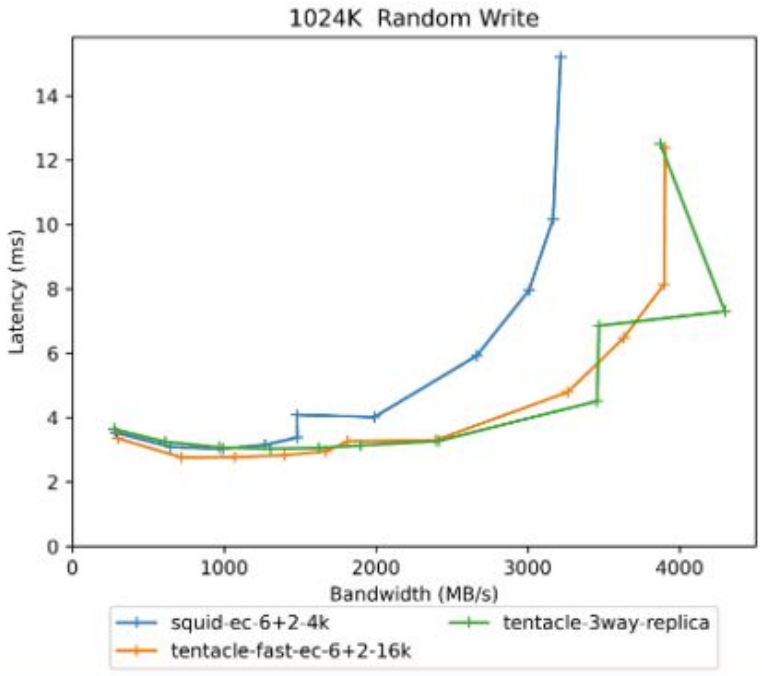
Updated using delta calculations rather than full recomputation, reducing CPU and network overhead.

Larger Striping

The default 16KB stripe_unit improves block and file performance significantly.

Perf Comparison: EC vs Replica-3





With Large Read & Write, EC performance is almost identical to Replica-3

EC Features Coming Soon

Coming in Ceph v21 (Umbrella) — late 2026

Direct Reads

Clients communicate directly with OSDs to retrieve data, matching the read performance of replicated pools.

OMap Support

Eliminates the requirement for a separate replicated pool for metadata, simplifying cluster architecture.

Pool Migration

Seamlessly convert pools between Replication and EC, or update EC parameters (K & M) with internal data migration.

Application Disaster Recovery

What is Disaster Recovery ?

KubeCon CloudNativeCon India 2026

Kubernetes keeps your apps running — but what happens when the entire cluster goes down?

✅ What Kubernetes Handles

Pod failures

Restarts or reschedules pods

Node failures

Moves workloads to healthy nodes

Resource issues

Self-heals within the cluster

❌ What K8s DOES NOT Protect Against

- Entire cluster failure
- Data corruption
- Human error
- Storage backend failure

Disaster Recovery ensures availability across clusters — covering what Kubernetes alone cannot.

Why Should You Care?

Business Impact of Downtime

Every minute of downtime = lost revenue, lost trust. Every hour of lost data = real business impact.

RPO

Recovery Point Objective

Guarantees **how much data you can lose**. Defines the maximum age of files that must be recovered for normal operations.

RTO

Recovery Time Objective

Guarantees **how fast you recover**. Defines the targeted duration within which a business process must be restored.

Key Components

OCM

Open Cluster Management

- Centralized control plane
- Manages lifecycle
- Application placement
- Integrates with Ramen

Ramen

DR Controller

- Open-source DR controller
- Hub orchestrator
- Manages DR policies
- Mirroring & VolSync

CSI-Addon

VR & VGR

- VR: per-PVC replication policy
- VGR: group PVCs into a consistency unit
- Provides volume replication CRDs
- Enables granular DR for stateful apps

Rook

Storage Operator

- Ceph storage on K8s
- RBD mirroring setup
- Orchestrates storage
- Resource management

How Rook Enables Replication for DR?

RBD Mirroring

CephRBDMirror (CRD)

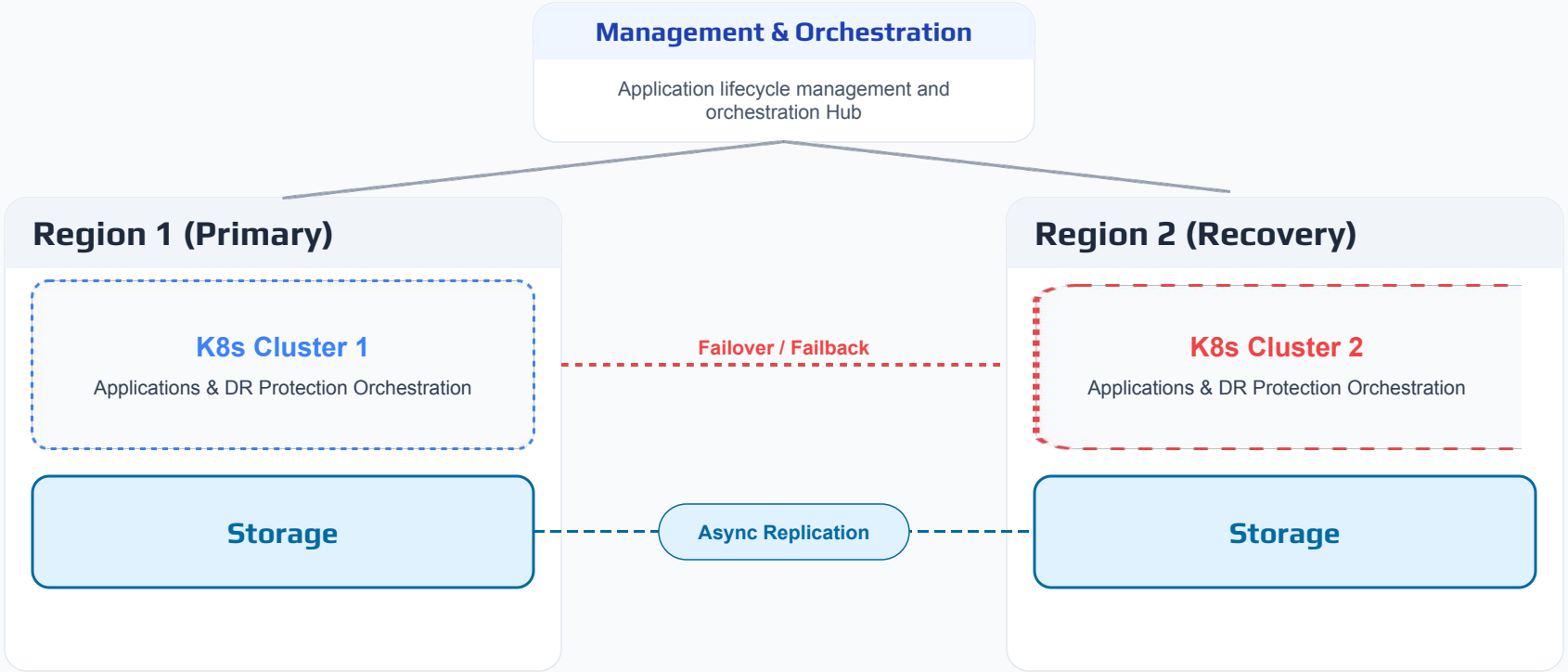
- Creates RBD pool & enables mirroring
- Bootstraps peers between clusters
- Deploys rbd-mirror daemon for async replication
- Snapshot-based sync — efficient & consistent
- Replication runs in both directions

Volume Replication CRDs

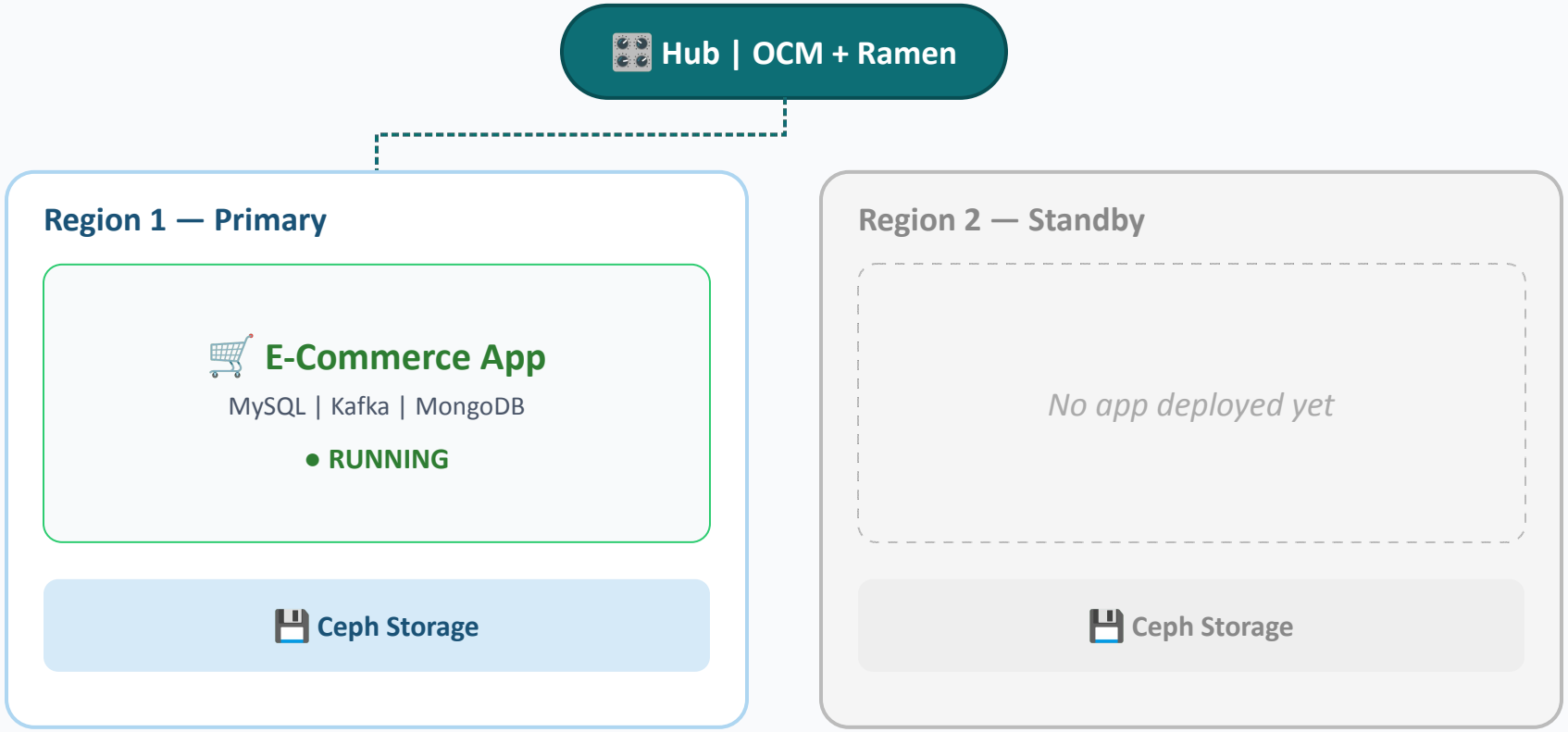
CSI Addons — VR & VGR

- **VolumeReplicationClass** — driver config & parameters
- **VolumeReplication CR** — replicates a single PVC
- **VolumeGroupReplication** — replicates multiple PVCs as a consistency group
- Manages failover & failback via promote/demote
- Integrates with Ramen & OCM

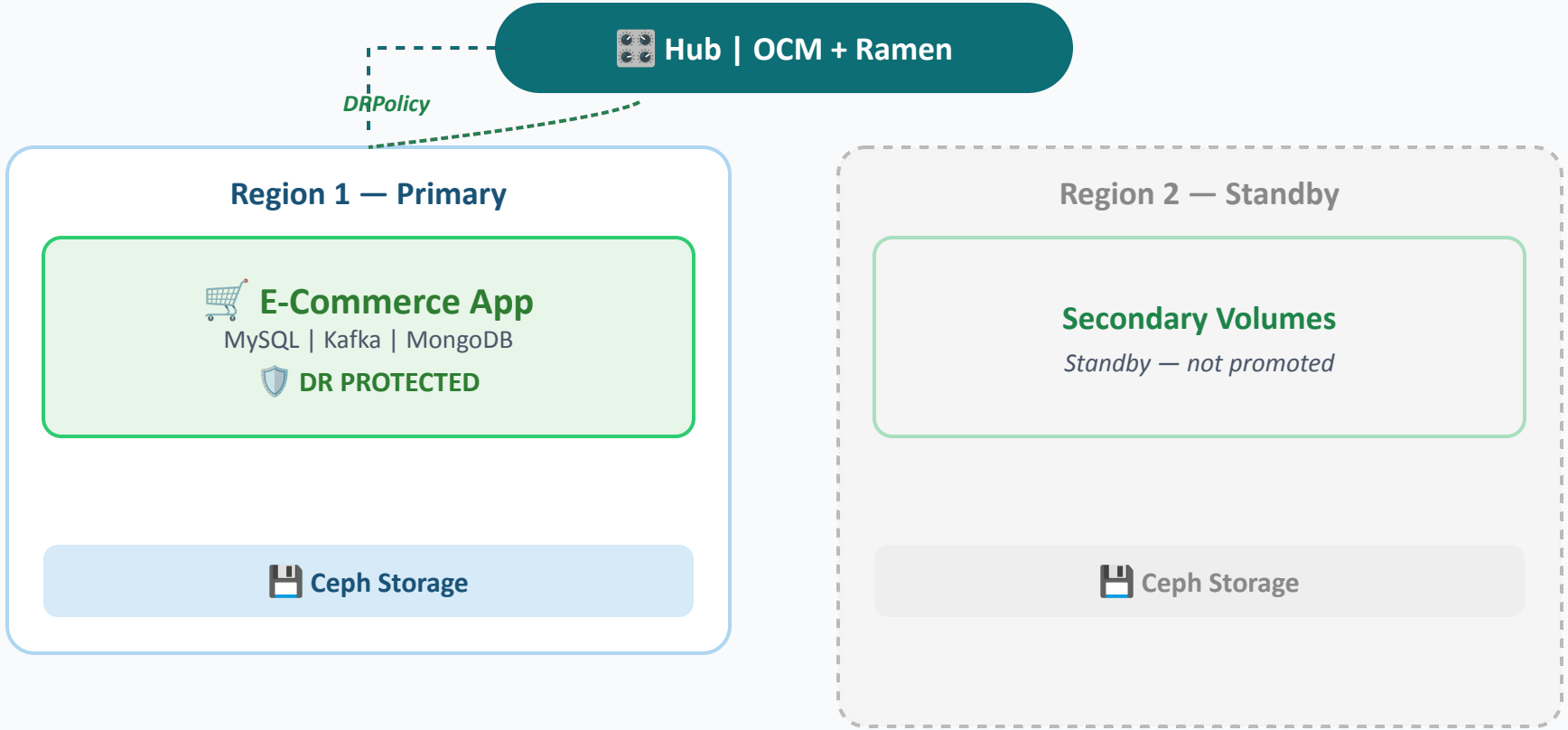
Architecture: Building Blocks



Step 1: Deploy App running on Region 1



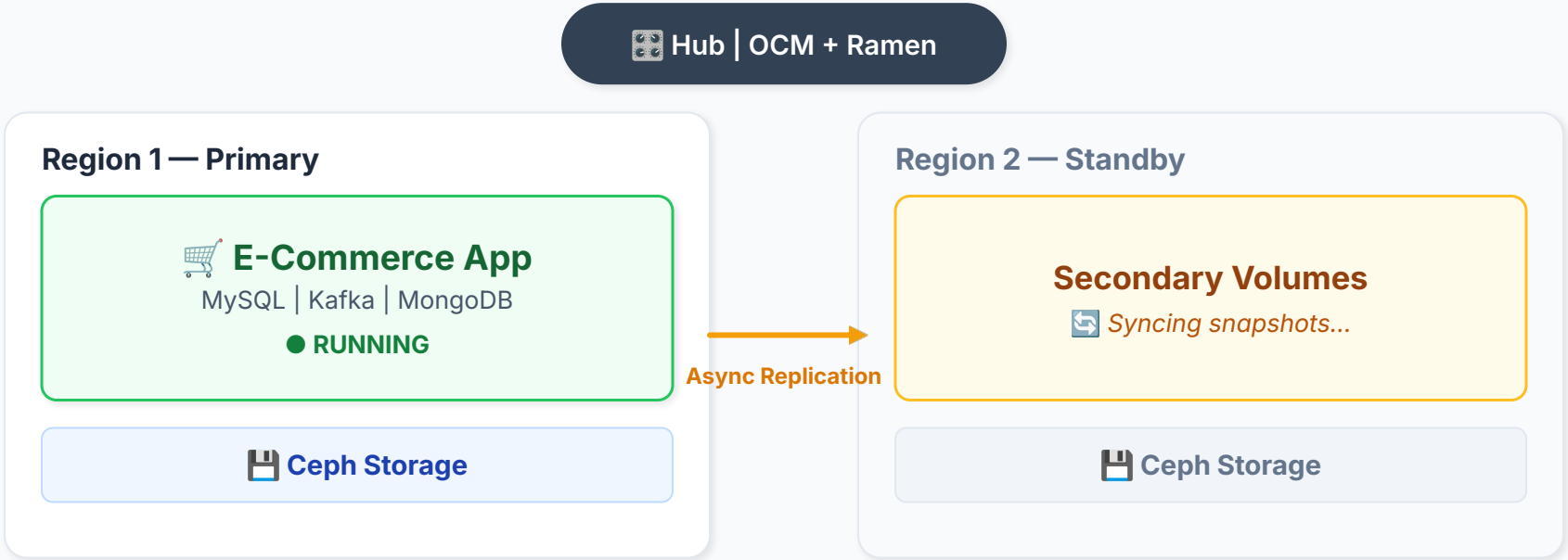
Step 2: Apply DR Policy from Hub



Step 3: Replicate

Async data from Region 1 → Region 2

KubeCon CloudNativeCon India 2026



Step 4: Failover

Disaster hits! Region 1 down

KubeCon CloudNativeCon India 2026

🎛️ Hub | OCM + Ramen

Region 1 — PRIMARY

💥 **E-Commerce App**

⛔ **CLUSTER DOWN**

Storage corrupted

💾 **Storage — FAILED**

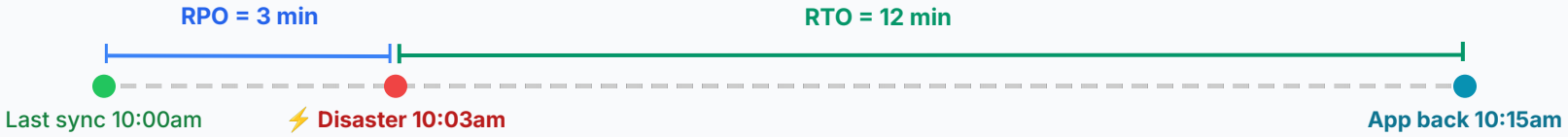
Region 2 — FAILOVER ✅

🛒 **E-Commerce App**

Volumes PROMOTED

● **RUNNING**

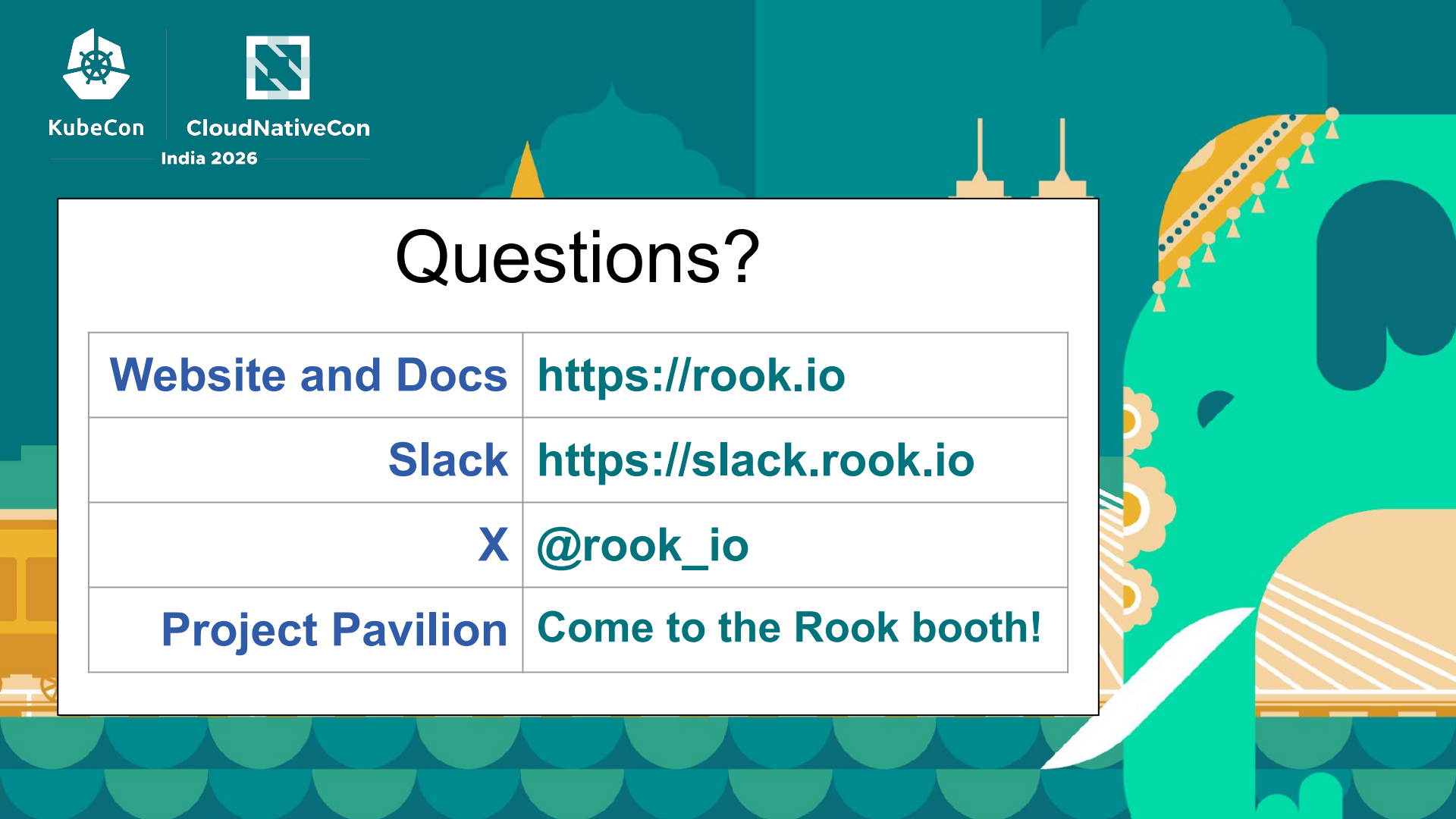
💾 **Storage — PRIMARY ✅**



How It Works: The DR Workflow

KubeCon CloudNativeCon India 2026

1	2	3	4	5
Deploy	**Protect**	**Replicate**	**Failover**	**Failback**
App deployed on primary cluster (Region 1)	Enable DR protection from the Hub	Data continuously replicates to Region 2	Disaster hits → Hub initiates failover	Relocate back to Region 1 — RPO = zero, no data loss



Questions?

Website and Docs	https://rook.io
Slack	https://slack.rook.io
X	@rook_io
Project Pavilion	Come to the Rook booth!

KubeCon CloudNativeCon India 2026

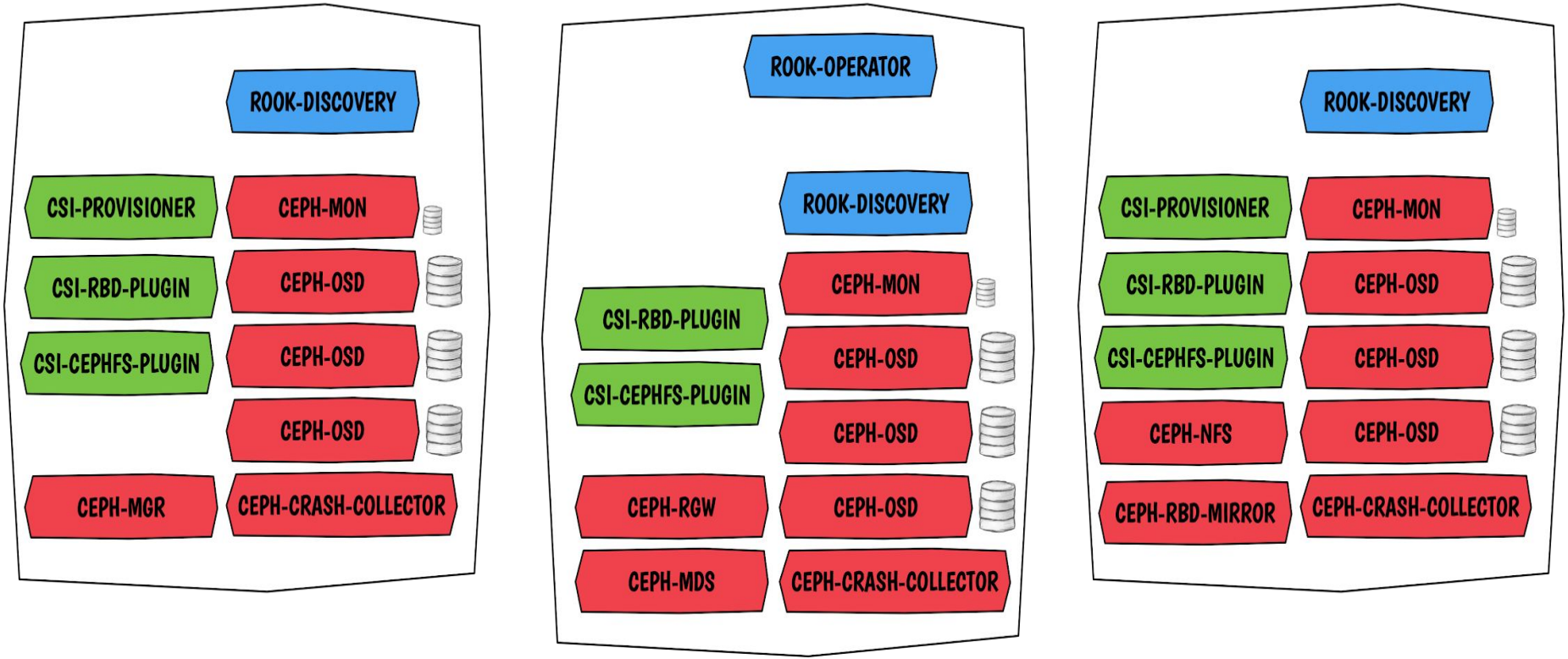
Appendix

RPO and RTO Explained

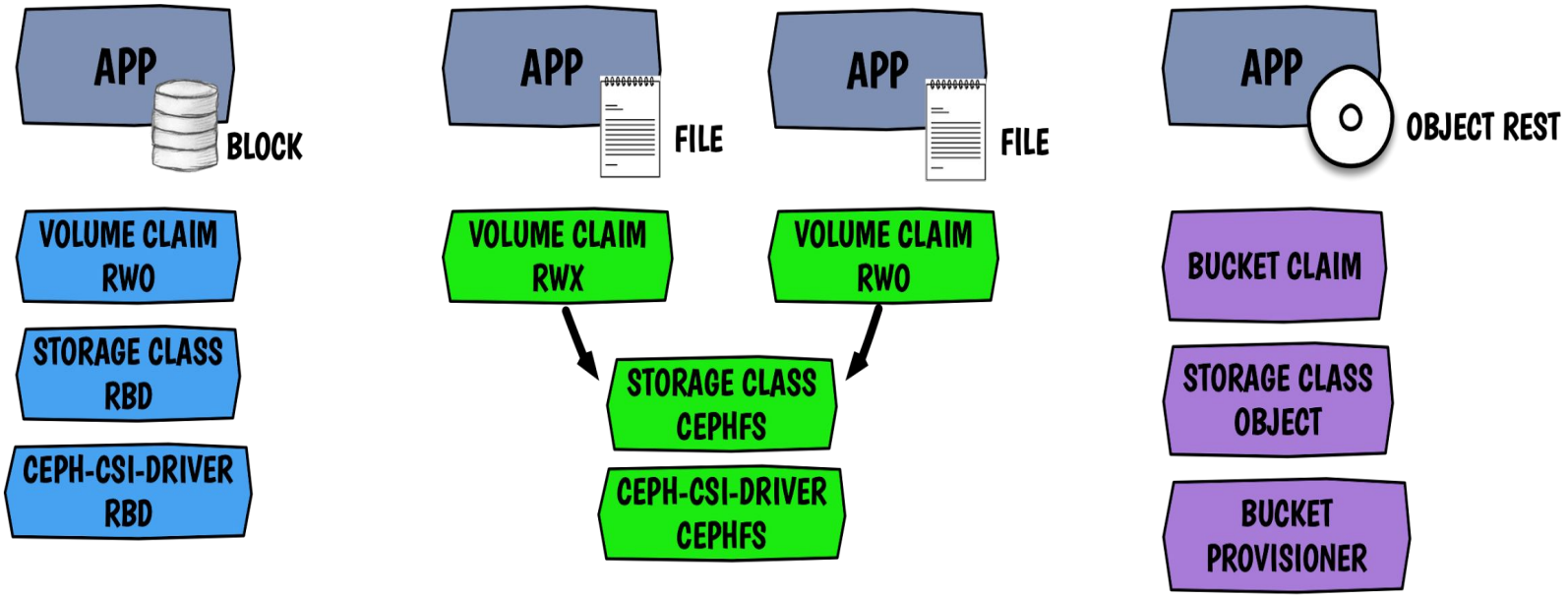
KubeCon CloudNativeCon India 2026

Your DR strategy is designed around these two numbers. **The lower, the better** and the more it costs.

Rook Pods


ROOK-DISCOVERY
CSI-PROVISIONER
CEPH-MON
CSI-RBD-PLUGIN
CEPH-OSD
CSI-CEPHFS-PLUGIN
CEPH-OSD
CEPH-OSD
CEPH-MGR
CEPH-CRASH-COLLECTOR
ROOK-OPERATOR
ROOK-DISCOVERY
CSI-RBD-PLUGIN
CEPH-MON
CSI-CEPHFS-PLUGIN
CEPH-OSD
CEPH-OSD
CEPH-RGW
CEPH-OSD
CEPH-MDS
CEPH-CRASH-COLLECTOR
ROOK-DISCOVERY
CSI-PROVISIONER
CEPH-MON
CSI-RBD-PLUGIN
CEPH-OSD
CSI-CEPHFS-PLUGIN
CEPH-OSD
CEPH-NFS
CEPH-OSD
CEPH-RBD-MIRROR
CEPH-CRASH-COLLECTOR

CSI Provisioning



Ceph Data Path