
KubeCon
CloudNativeCon
North America 2019

Introduction to OpenEBS

KubeCon | CloudNativeCon

North America 2019

@amitnist
@ivishnuvardhan

MayaData

Visit us at Booth SE23

Agenda

- Overview
- CAS
- Architecture & Design details
- Use cases
- Performance
- Future
- Conclusion

Overview

Overview

- OpenEBS was created in late 2016 and was initially sponsored by MayaData.
- **CNCF Sandbox project.**
- Users started running OpenEBS in production, fall of 2017.
- Recent uptick in usage - lots of feedback on community & top tech companies contributing and using.
- Growing usage ~30-40% month on month in 2019

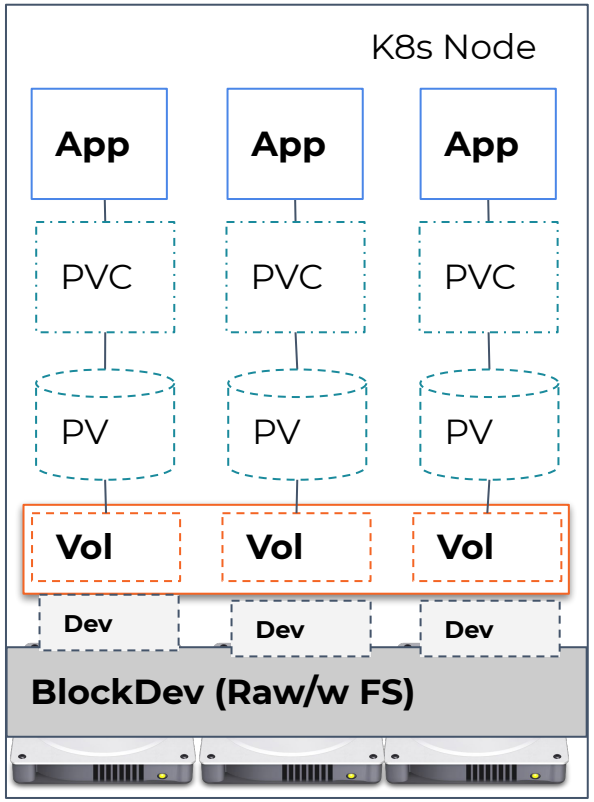
- Open Source from Start!
- Apache 2.0 Licensed
- 350+ contributors from different companies
- 1700+ Slack Members
- 600+ Forks
- 6000+ stars across main repositories
- MayaData is so far the biggest contributor to OpenEBS. It is a **data agility company**, that turns Kubernetes itself into your data plane.

OpenEBS Design Manifesto

- Easy to set up. Low entry barrier. *Developer and operator friendly. Offer both freedom and flexibility to control.*
- Optimize data operations for *running Stateful workloads seamlessly on *any* Kubernetes platform.*
- Built using containers and microservices architecture patterns. Orchestrated by Kubernetes and its ecosystem. *Containerized Storage for Containers!*
- *Stable, Secure and Scalable* - Fault tolerant, horizontally scalable and secure by default
- Seamless integration into any private and public cloud environments. *Vendor independent.*
- Non-disruptive software upgrades *all the way to storage.*

CAS

Container Attached Storage - CAS

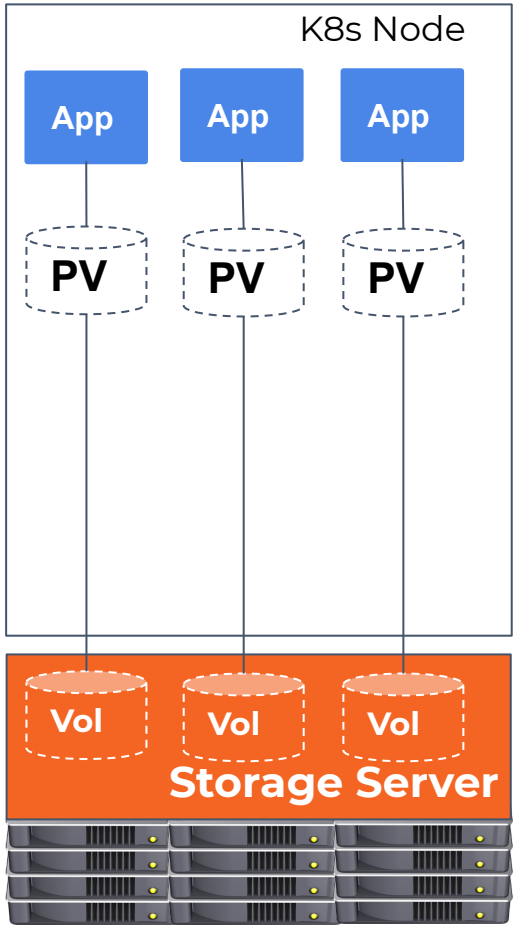


- Storage controllers run as microservices (containers).
- Avoids kernel dependencies.
- These storage containers are orchestrated by Kubernetes and its extensions (like any other workloads).
  - Installation and Upgrades
  - Scheduling
  - Monitoring, Debuggability
- Storage containers mainly deal with:
  - Disk/Storage Management
  - Data - High Availability and
  - Data - Protection
- CAS is Container Native

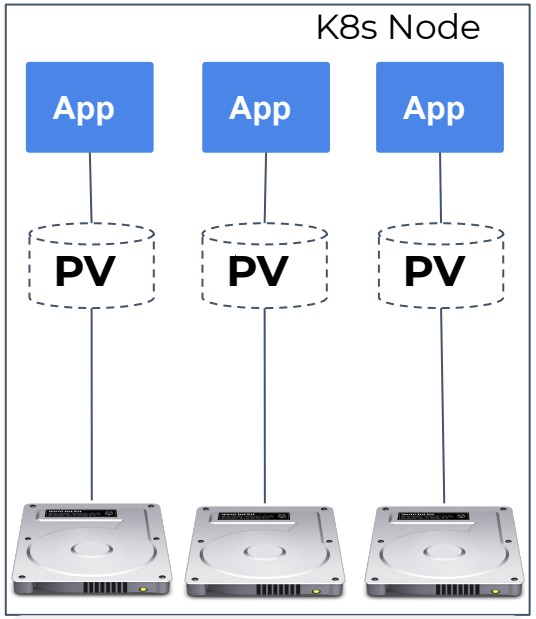
https://www.cncf.io/blog/2018/04/19/container-attached-storage-a-primer/

Persistent Volume Categories

NAS/EBS

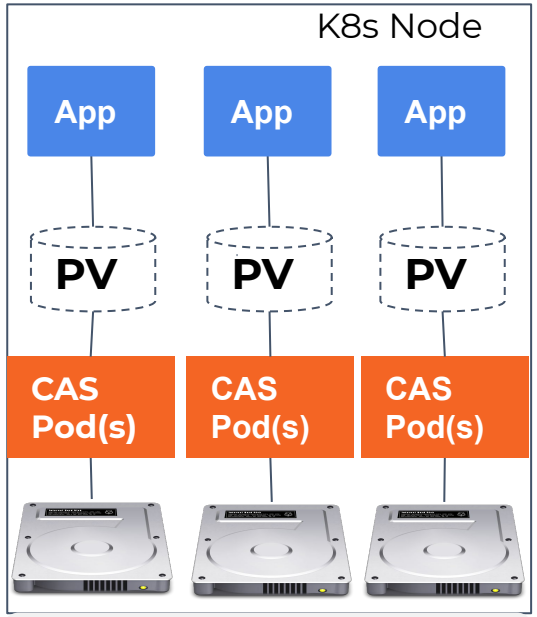


DAS/LocalPV



- No Snapshots
- No HA

CAS/OpenEBS

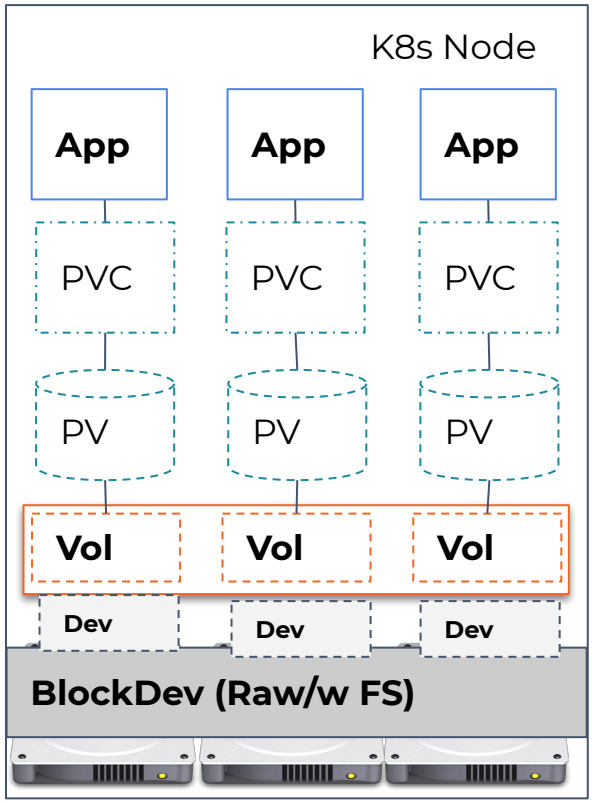


- Simple, familiar
- App granularity
- K8S native
- 100% user space

Represent stateful Pods like Databases, etc.

Indicates functionality like replication, snapshots, encryption, compression, etc.

CAS Examples



Examples of Open Source (CNCF) CAS Solutions

- OpenEBS Storage Engines (cStor, Jiva, MayaStor)
- Rancher Longhorn

Examples of CAS Helpers

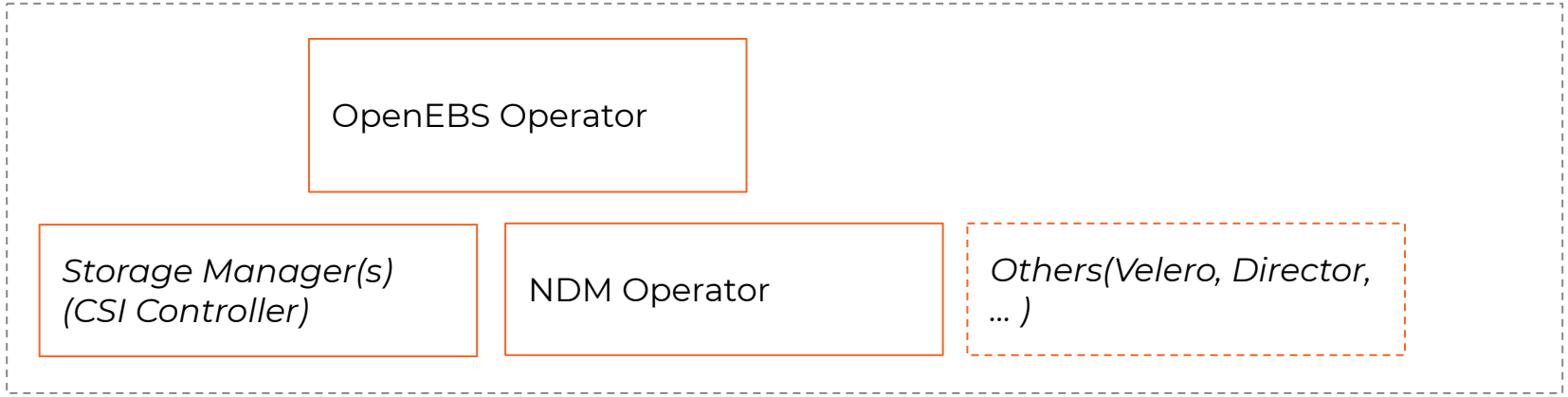
- Rook (Ceph or OpenEBS can be plugged in)

*"OpenEBS is a **CAS** solution, that provides storage as a service to stateful workloads. OpenEBS hooks-into and extends the capabilities of Kubernetes to orchestrate storage services (workloads)"*

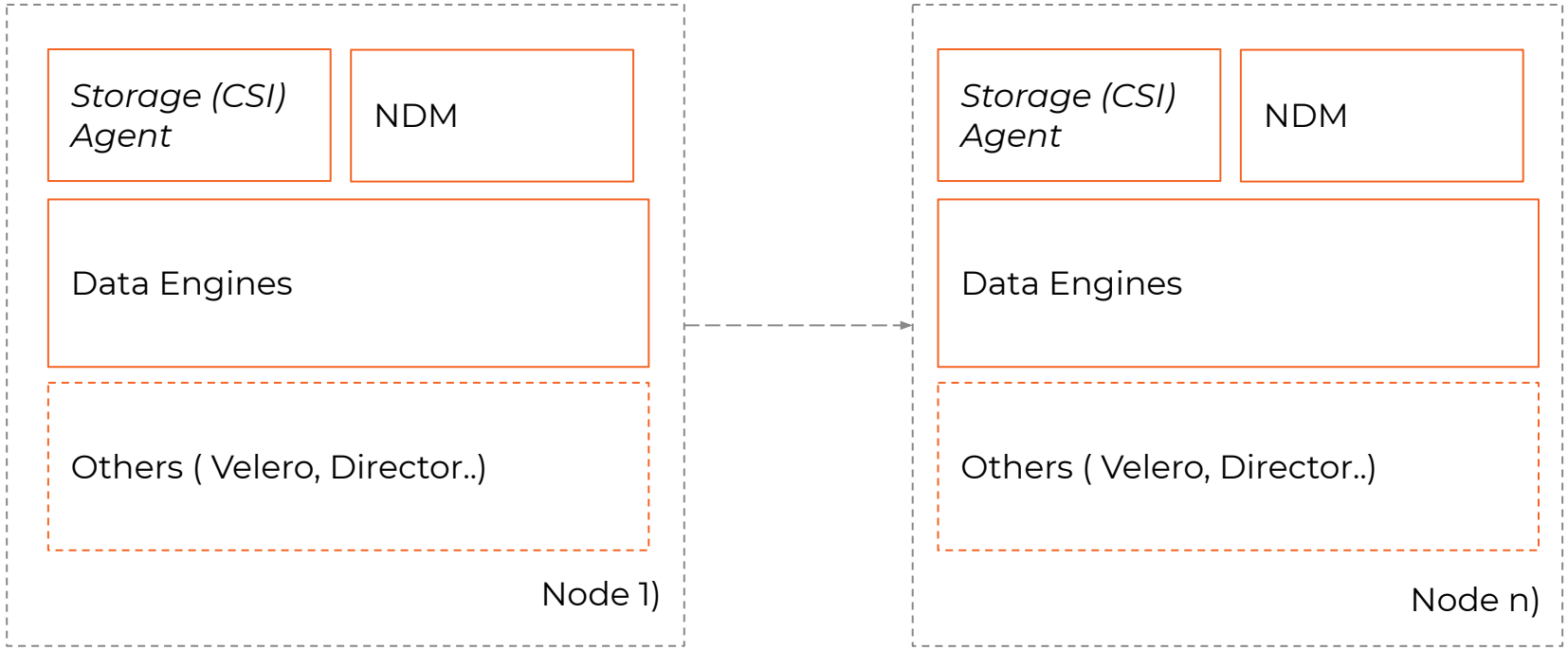
Architecture & Design details

Architecture Overview

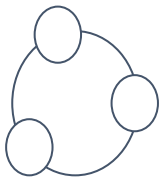
Cluster Components



Node Components

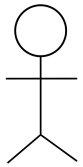


Node Device Manager



Operator

ndmctl



kubectl

Cluster Level Components

NDM Operator

Device Claim CR

Device CR

Disk CR (K8s)

Node (1 . . . n)

Metrics Exporter

Node Device Manager (DaemonSet)

Enclosure / Disk Management (SeaChest)

kube-apiserver
kube-scheduler . . .

etcd

Kubernetes Master Components

Monitoring, Alerting, Logging, Tracking, . . .

(Example: Prometheus Fluentd, Jaegar . . .)

Infra Management Components

NDM - Discovery


```
# lsblk
NAME          MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda             8:0    0   135G  0 disk
├─sda1          8:1    0   512M  0 part /boot
├─sda2          8:2    0    29G  0 part
│ ├─rhel-root 253:0    0    25G  0 lvm  /
│ ├─rhel-swap 253:1    0     4G  0 lvm
│ ├─rhel-home 253:2    0     5G  0 lvm  /home
│ ├─rhel-var  253:3    0    85G  0 lvm  /var
│ └─rhel-tmp  253:4    0     5G  0 lvm  /tmp
├─sda3          8:3    0     5G  0 part
│ └─app-opt   253:5    0     5G  0 lvm  /opt
├─sda4          8:4    0   512B  0 part
└─sda5          8:5    0 100.5G  0 part
  ├─rhel-root 253:0    0    25G  0 lvm  /
  └─rhel-var  253:3    0    85G  0 lvm  /var
sdb             8:16   0  1000G  0 disk
```

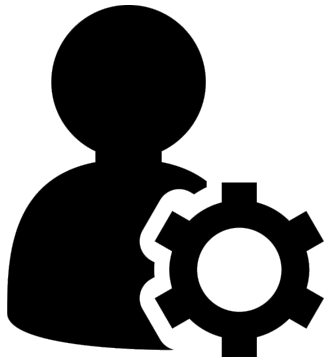
Block Device


```yaml
apiVersion: openebs.io/v1alpha1
kind: BlockDevice
metadata:
  labels:
    kubernetes.io/hostname: dmbu01lx03578b
    ndm.io/blockdevice-type: blockdevice
    ndm.io/managed: "true"
  name: blockdevice-ac032f45ad215a85582d64aa3c966c98
  namespace: multik8s-storage
spec:
  capacity:
    logicalSectorSize: 512
    physicalSectorSize: 0
    storage: 1073741824000
  claimRef:
    apiVersion: openebs.io/v1alpha1
    kind: BlockDeviceClaim
    name: bdc-pvc-87c8e37c-06b8-11ea-b474-005056b580b3
    namespace: multik8s-storage
    uid: 87c8e37c-06b8-11ea-b474-005056b580b3
  details:
    compliance: SPC-4
    model: Virtual_disk
    serial: 6000c294eba3eff75f6bb8823b00eba3
    vendor: VMware
  devlinks:
  - kind: by-id
    links:
```

Block Device contd...


```
spec:
  capacity:
    logicalSectorSize: 512
    physicalSectorSize: 0
    storage: 1073741824000
  claimRef:
    apiVersion: openebs.io/v1alpha1
    kind: BlockDeviceClaim
    name: bdc-pvc-87c8e37c-06b8-11ea-b474-005056b580b3
    namespace: multik8s-storage
    uid: 87c8e37c-06b8-11ea-b474-005056b580b3
  details:
    compliance: SPC-4
    model: Virtual_disk
    serial: 6000c294eba3eff75f6bb8823b00eba3
    vendor: VMware
  devlinks:
  - kind: by-id
    links:
    - /dev/disk/by-id/scsi
      -36000c294eba3eff75f6bb8823b00eba3
    - /dev/disk/by-id/wwn
      -0x6000c294eba3eff75f6bb8823b00eba3
  - kind: by-path
    links:
    - /dev/disk/by-path/fc---lun-0
    - /dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:1:0
  filesystem: {}
  nodeAttributes:
```

OpenEBS Local PV Provisioner



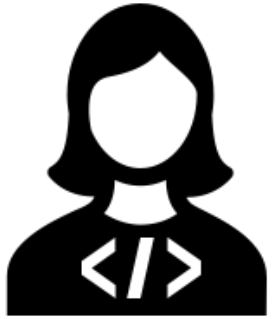
DevOps admin

Setup OpenEBS

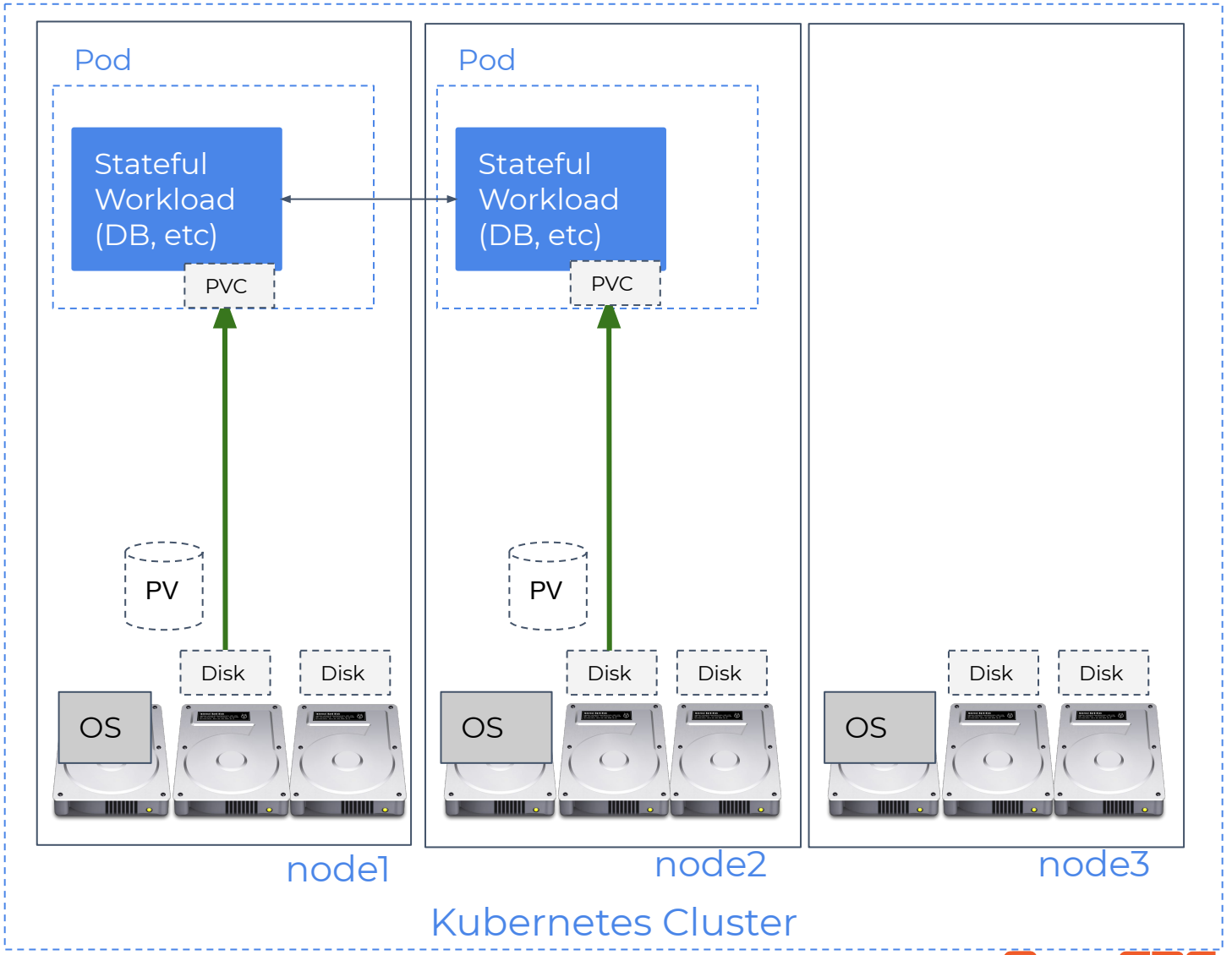
(1) node-disk-manager, provisioner,
(2) StorageClass

Using OpenEBS

(3) StatefulSet with PVC
(4) PV



Developer



OpenEBS

Dynamic Local Device


```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: openebs-device
  annotations:
    openebs.io/cas-type: local
    cas.openebs.io/config: |
      #device type will create a PV by
      # issuing a BDC and will extract the path
      # values from the associated BD.
      - name: StorageType
        value: "device"
provisioner: openebs.io/local
volumeBindingMode: WaitForFirstConsumer
reclaimPolicy: Delete
```

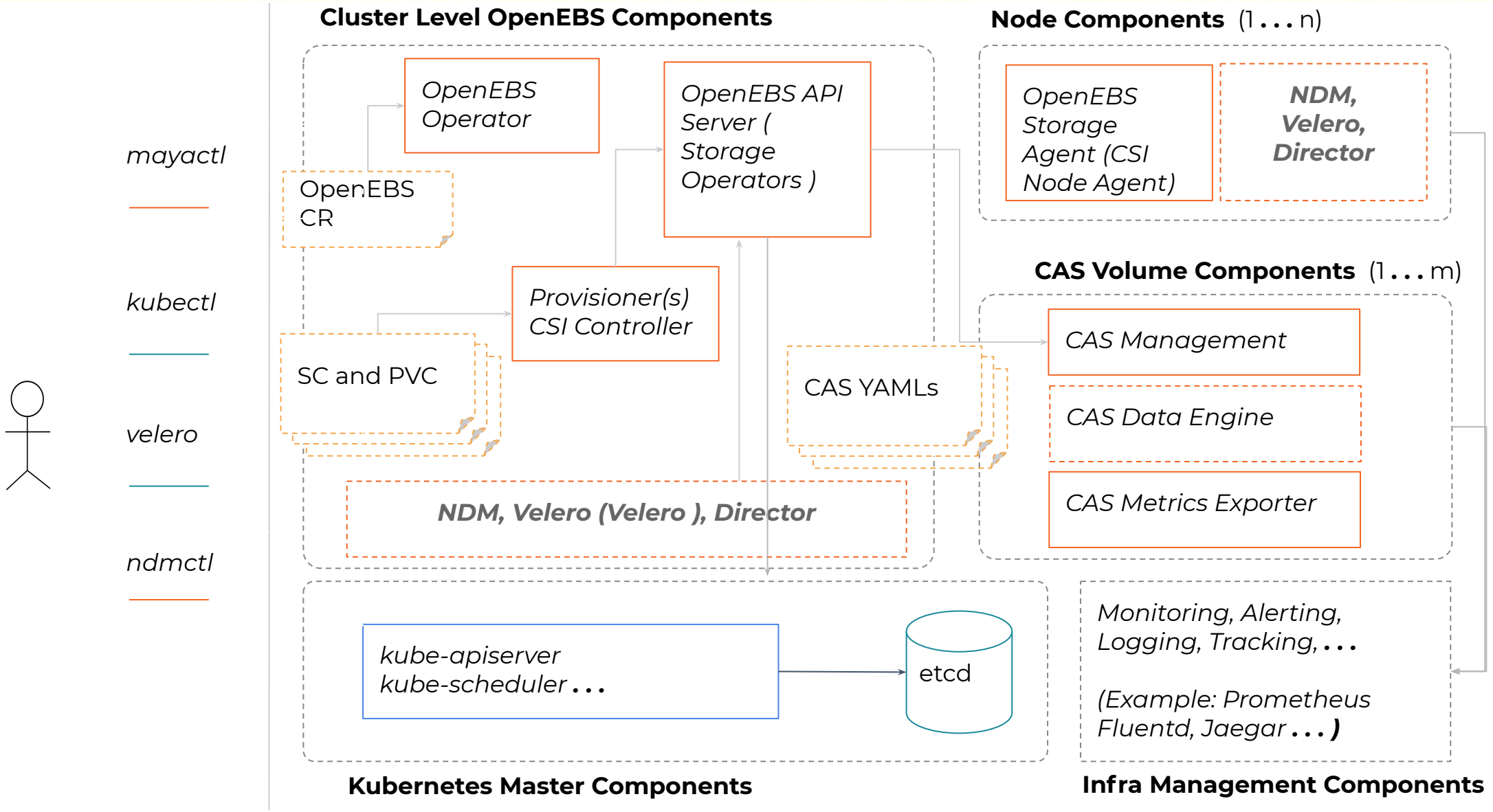
Block Device Claim


```
---
apiVersion: openebs.io/v1alpha1
kind: BlockDeviceClaim
metadata:
  name: bdc-pvc-87c8e37c-06b8-11ea-b474-005056b580b3
  namespace: multik8s-storage
spec:
  blockDeviceName: blockdevice
    -ac032f45ad215a85582d64aa3c966c98
  blockDeviceNodeAttributes:
    hostName: dmbu01lx03578b
  resources:
    requests:
      storage: 950Gi
```

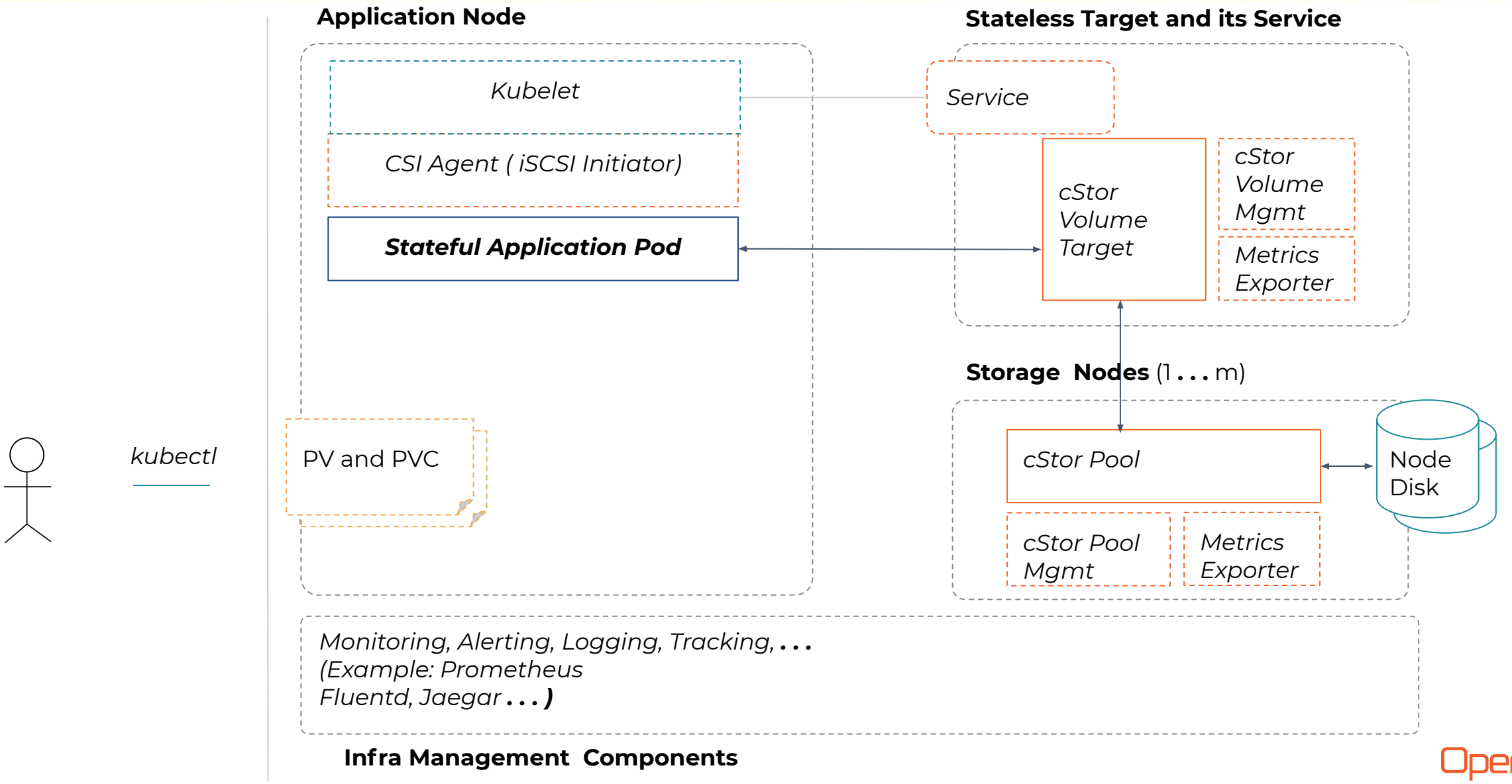
Dynamic HostPath


```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: openebs-hostpath
  annotations:
    openebs.io/cas-type: local
    cas.openebs.io/config: |
      # hostpath type will create a PV by creating a sub-directory under
      # the BASEPATH provided below.
      - name: StorageType
        value: "hostpath"
      # Specify the location (directory) where where PV(volume) data will
      # be saved. A sub-directory with pv-name will be created. When the
      # volume is deleted, the PV sub-directory will be deleted.
      # Default value is /var/openebs/local
      - name: BasePath
        value: "/var/openebs/local/"
provisioner: openebs.io/local
volumeBindingMode: WaitForFirstConsumer
reclaimPolicy: Delete
```

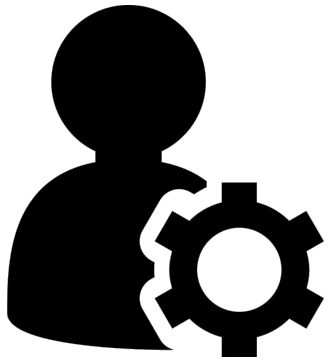
OpenEBS Control Plane (Maya)



cStor Data Engine



cStor Data Engine



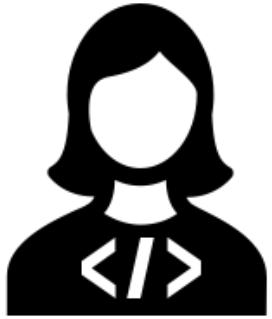
Cluster admin

Setup OpenEBS

(1) node-disk-manager, provisioner, cstor operator

(2) SPC=>StoragePool(s)

(3) StorageClass

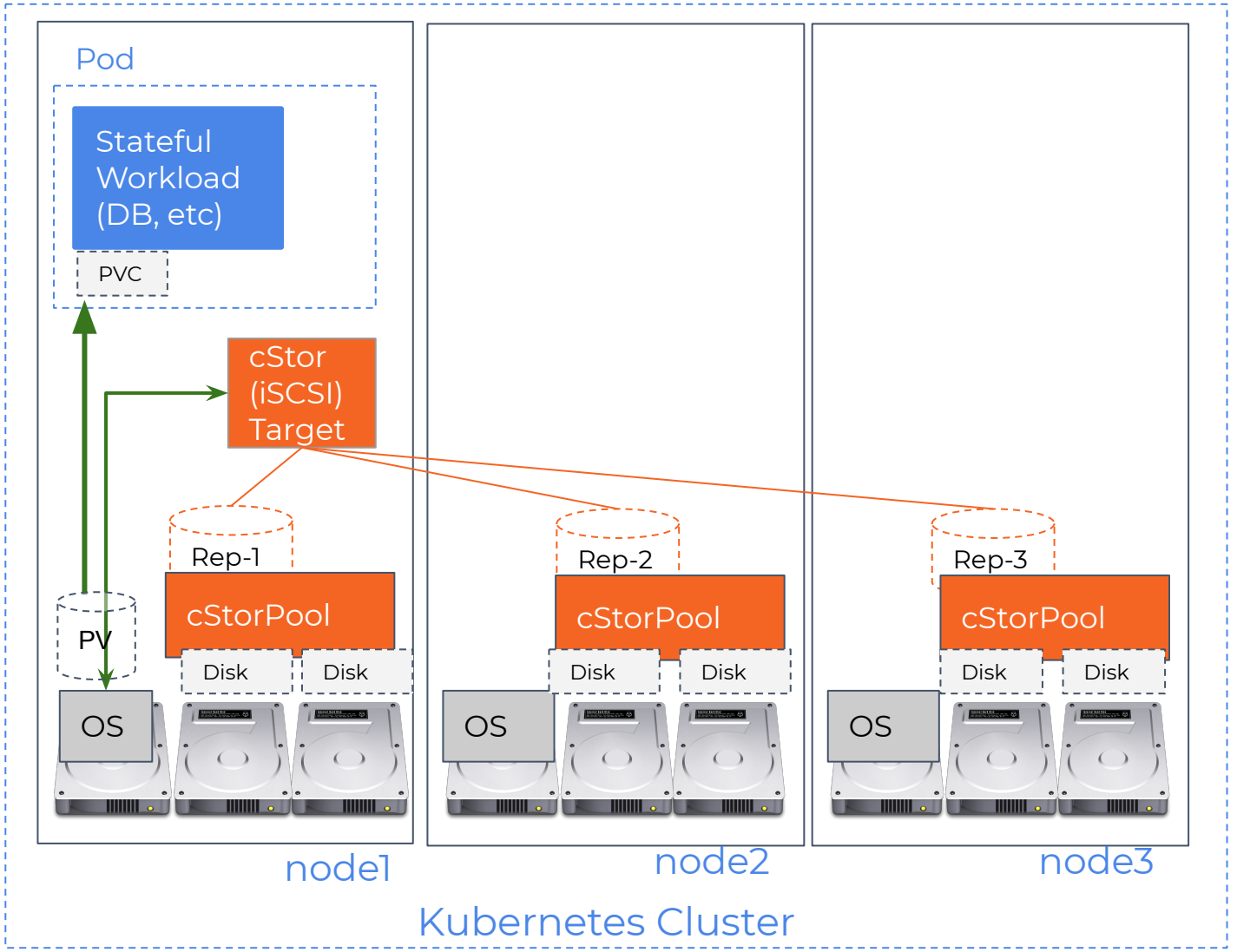


Developer

Using OpenEBS

(4) Pod with OpenEBS PVC

(5) PV



Use cases

Adopters include

https://github.com/openebs/openebs/blob/master/ADOPTERS.md

Workloads

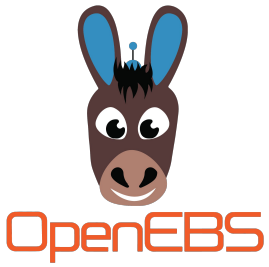



and many more

UC: CI/CD #devops

Problem Statement

- Pipelines spin up frequent and short lived workloads
- Too many workloads running with minimal IO needs like gitlab executors
  - overall cluster can meet workload resource requirements, but, not a single or few nodes
- Storage is available on few nodes
- Storage aware scheduling of workloads is problem



- Discovers storage devices
- Containerized iSCSI target per volume as per CAS architecture
- Scheduling issues due to too many workloads will not happen as workloads can run anywhere

UC: MLOps

Problem Statement:

- Workloads like data pipelines that need instant snapshots, clones for data sharing across teams
- Different steps of data pipelines running in diff cluster
- Replicating data pipelines to different cluster
- Data protection during OS / application upgrades

OpenEBS

- Copy-On-Write snapshots, clones
- velero-plugin to backup/restore data to another cluster

UC: Cloud Native Stateful Applications

Problem Statement:

- Applications take care of replication, high availability of data
- Workloads demanding low latency, high performance storage
- Sharing of underlying storage with multiple applications in K8s native way
- Hyperconverged K8s cluster with directly attached storage

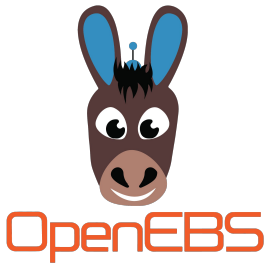
OpenEBS

- Discovers storage devices
- Provides dynamic provisioning of
  - local disks and their partitions
  - directories as local PV on another local PV
  - ZVOLs from underlying ZFS pools in the cluster nodes

UC: IoT/Edge computing, Monitoring apps

Problem Statement:

- Workloads run on Edge devices with
  - minimal hardware resources
  - minimal storage capacity
- Storage for containerized apps on Edge devices to store filtered offline data
- Monitoring, alerting and metrics gathering applications with smaller resource footprint
- Replicable setups to run stateful workloads at scale with ease



- ARM support
- OpenEBS related pods can be configured with resource limits
- These limits impacts IOPS and latencies
- Within configured limits, it provides storage to applications by giving enough room for workloads to perform

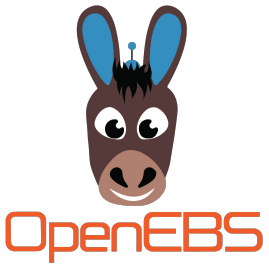
UC: Cost savings #noCloudLockin

Problem Statement:

- Flexible in selecting or changing cloud vendor
- In the cloud,
  - difficult to obtain new nodes, persistent disks
  - attach / detach of remote disks takes time
  - High cost ratio between ephemeral to remote disks, and, preemptible to regular nodes
  - Increase in a node cost many folds as HW specs increases

- Faulty domains with cloud or on-prem clusters, with respect to,
  - Disks going bad or unreachable
  - Nodes can be down due to (other than HW failures)
    - vMotion kind of cases in vSphere
    - node upgrades
  - Zones becoming unreachable

UC: Cost savings, Easy operations (contd..)



- KubeMove, Velero plugin support for data migration
- Thin provisioning of storage (add disks on demand)
- Software RAID is available for data protection against disks turning bad
- Synchronous replication across nodes and zones to guarantee data high availability to workloads even in case of node/zone unreachability
  - Allows workloads to run anywhere leading to efficient utilization of HW resources
- Works with ephemeral disks that takes care of reconstructing entire data into new disks automatically
- Works with preemptible nodes as well on attaching remote disks to new preemptible nodes
- No kernel dependencies as storage engines runs in user space
- Same storage experience on different IaaS like openstack, vsphere and K8s deployments with bosh

Performance

- Multiple provisioners can run to scale volume provisioning requests using leader-election
- Workloads can achieve near disk performance using local PV dynamic provisioner of DAS architecture
- CAS architecture provisions volumes and allows to scale workload count that require lesser IOPS
- Jiva/cStor - being a replicated block storage, the performance is as good as Ceph and has the benefits of being more resilient to multiple fault domains, easy-to-setup/maintain.
- MayaStor - low latency, high throughput engine based on NVMe-oF technology

Future

- High performing storage engine with synchronous replication, snapshots
- Cluster Autoscale aware storage
- Application consistent snapshots
- Workload migration from one cluster to another along with data
- Disk unique identification and unique access
- More events, alerts, metrics

Conclusion

- Flexibility
- Easy-to-use
- Persona oriented
- Cloud native storage in K8s way
- Storage for workloads in Hyper Converged and on-prem clusters
- High availability of data
- Cost aware storage provisioning layer
- Storage engine as per application storage demands
- Resources limits for storage pods
- Synchronous replication and rebuilding
- Snapshots / Clones
- Backup / Restore

References

- https://github.com/openebs/openebs/blob/master/ADOPTERS.md
- CNCF Landscape storage whitepaper
- PC for devOps image
- PC for MLOps image


KubeCon
CloudNativeCon
North America 2019
Thank You
MayaData
Visit us at Booth SE23

NDM Config - Sample


```
probeconfigs:
  - key: udev-probe
    name: udev probe
    state: true
  - key: seachest-probe
    name: seachest probe
    state: false
  - key: smart-probe
    name: smart probe
    state: true
filterconfigs:
  - key: os-disk-exclude-filter
    name: os disk exclude filter
    state: true
    exclude: "/,/etc/hosts,/boot"
  - key: vendor-filter
    name: vendor filter
    state: true
    include: ""
    exclude: "CLOUDBYT,OpenEBS"
  - key: path-filter
    name: path filter
    state: true
    include: "/dev/nvme0n1,/dev/nvme1n1"
    exclude: "loop,fd0,sr0,/dev/ram,/dev/dm-,/dev/md,"
```

Block Device

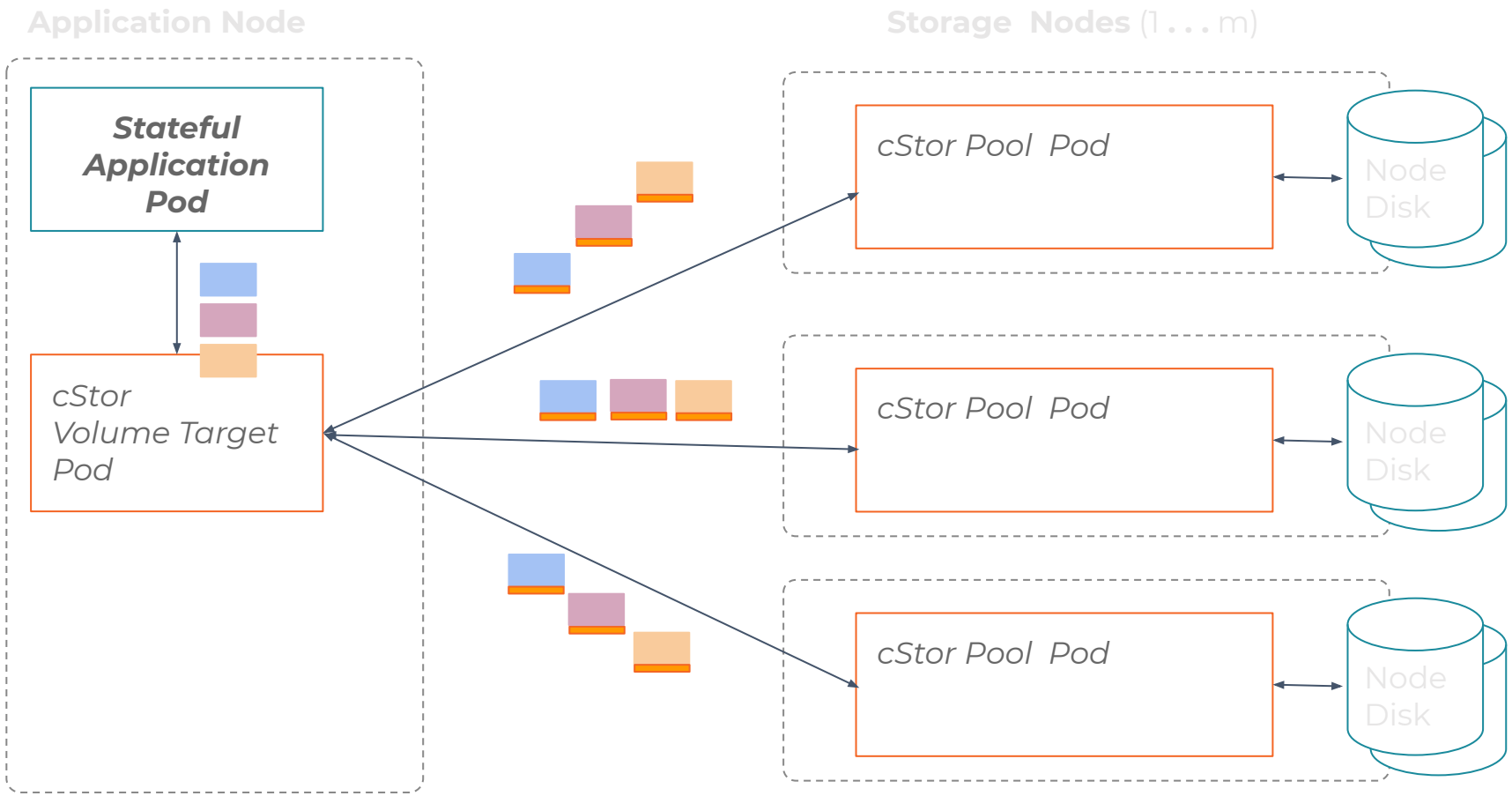

```yaml
apiVersion: openebs.io/v1alpha1
kind: BlockDevice
metadata:
  labels:
    kubernetes.io/hostname: dmbu01lx03578b
    ndm.io/blockdevice-type: blockdevice
    ndm.io/managed: "true"
  name: blockdevice-ac032f45ad215a85582d64aa3c966c98
  namespace: multik8s-storage
spec:
  capacity:
    logicalSectorSize: 512
    physicalSectorSize: 0
    storage: 1073741824000
  claimRef:
    apiVersion: openebs.io/v1alpha1
    kind: BlockDeviceClaim
    name: bdc-pvc-87c8e37c-06b8-11ea-b474-005056b580b3
    namespace: multik8s-storage
    uid: 87c8e37c-06b8-11ea-b474-005056b580b3
  details:
    compliance: SPC-4
    model: Virtual_disk
    serial: 6000c294eba3eff75f6bb8823b00eba3
    vendor: VMware
  devlinks:
  - kind: by-id
    links:
```

Block Device contd...


```
spec:
  capacity:
    logicalSectorSize: 512
    physicalSectorSize: 0
    storage: 1073741824000
  claimRef:
    apiVersion: openebs.io/v1alpha1
    kind: BlockDeviceClaim
    name: bdc-pvc-87c8e37c-06b8-11ea-b474-005056b580b3
    namespace: multik8s-storage
    uid: 87c8e37c-06b8-11ea-b474-005056b580b3
  details:
    compliance: SPC-4
    model: Virtual_disk
    serial: 6000c294eba3eff75f6bb8823b00eba3
    vendor: VMware
  devlinks:
  - kind: by-id
    links:
    - /dev/disk/by-id/scsi
      -36000c294eba3eff75f6bb8823b00eba3
    - /dev/disk/by-id/wwn
      -0x6000c294eba3eff75f6bb8823b00eba3
  - kind: by-path
    links:
    - /dev/disk/by-path/fc---lun-0
    - /dev/disk/by-path/pci-0000:03:00.0-scsi-0:0:1:0
  filesystem: {}
  nodeAttributes:
```

cStor Data Engine - High Availability

cStor Volume Target does Synchronous Replication, i.e writes copies of the data to each of the available Replica Pools.



cStor Volume Target attaches an unique sequence number to each of the block - before sending the copies to Replica Pools.