
KubeCon
CloudNativeCon
Europe 2020
Virtual
HELM
KEEP CLOUD NATIVE
CONNECTED

OpenEBS 101

Hyperconverged Kubernetes Storage

Speakers

KubeCon CloudNativeCon Europe 2020 Virtual

Kiran Mova

Chief Architect
Co-Founder
MayaData Inc

@kiranmova

kiranmova

Vishnu Vardhan Itta

Director of Engineering
MayaData Inc

@ivishnuvardhan

Vishnu itta

About MayaData

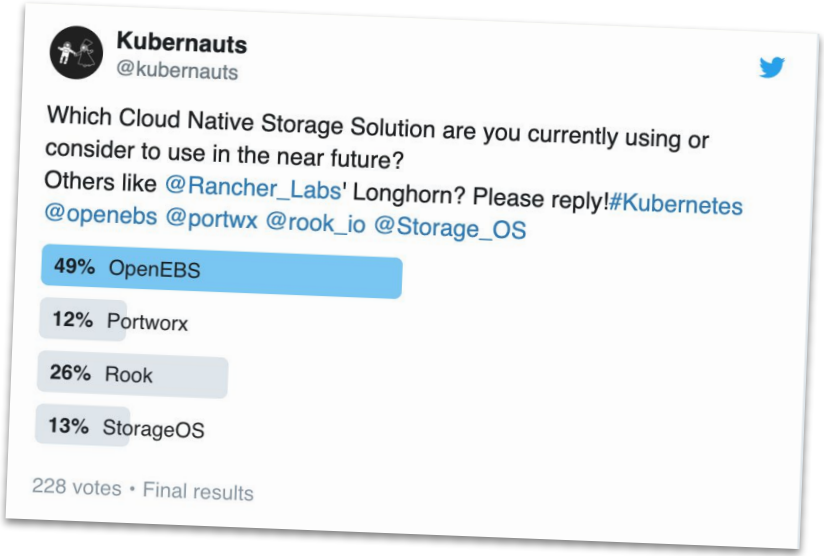
KubeCon CloudNativeCon Europe 2020 Virtual

Company	Rank for PRs to CNCF
Google	1st
Red Hat	2nd
VMware	3rd
Microsoft	4th
MayaData	5th

Source: All CNCF PRs authors

- Code *is* marketing
- Contributing in CNCF ecosystem
- 19 CKAs & growing

- 4X yr/yr growth in container pulls
- #1 CNS in trial per CNCF survey
- Rapidly becoming the defacto standard for stateful workloads on Kubernetes

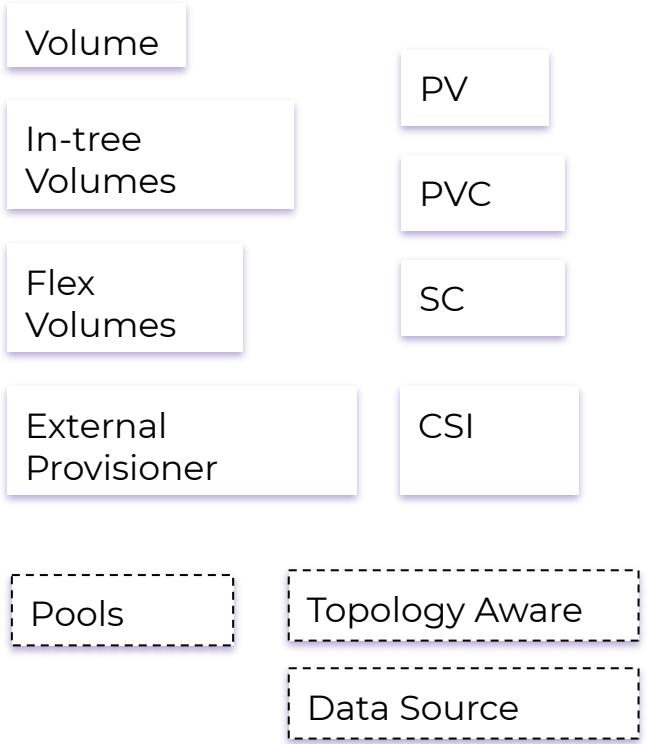


Agenda

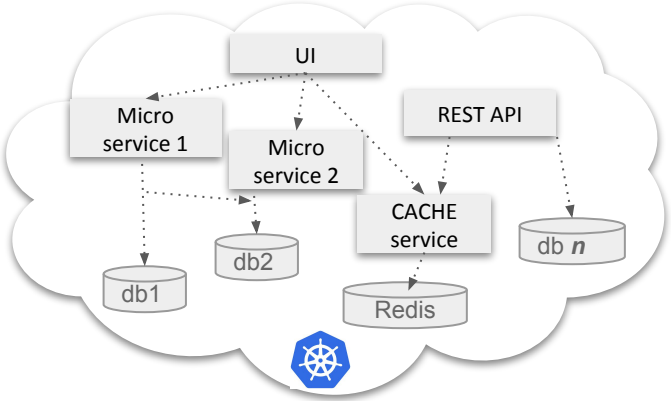
- K8s for Stateful
- Container Attached Storage (CAS)
- OpenEBS Storage Engines
- K8s as Data Layer - End User Stories

K8s for Stateful Workloads

- Native interfaces for connecting workloads (Pods) to Persistent Volumes (PVs).
- Dynamic provisioning of PV via Persistent Volume Claim (PVC) and Storage Class (SC).
- More abstraction through community efforts around Persistent Volumes (PV) and Persistent Volume Claims (PVC) and Container Storage Interface (CSI)
- CSI to handle vendor specific needs and avoid wildfire of “volume plugins” or “drivers” in K8s main repo



K8s for Stateful : Can't I Just?



CSI

A shared storage system is a complex monolithic distributed system built before **Kubernetes**

These systems have DBs for metadata
They have provisioning systems
They have retry & other logic

They take all the IO, mix it together, and do their best

Designed when storage media was slow and apps were NOT resilient

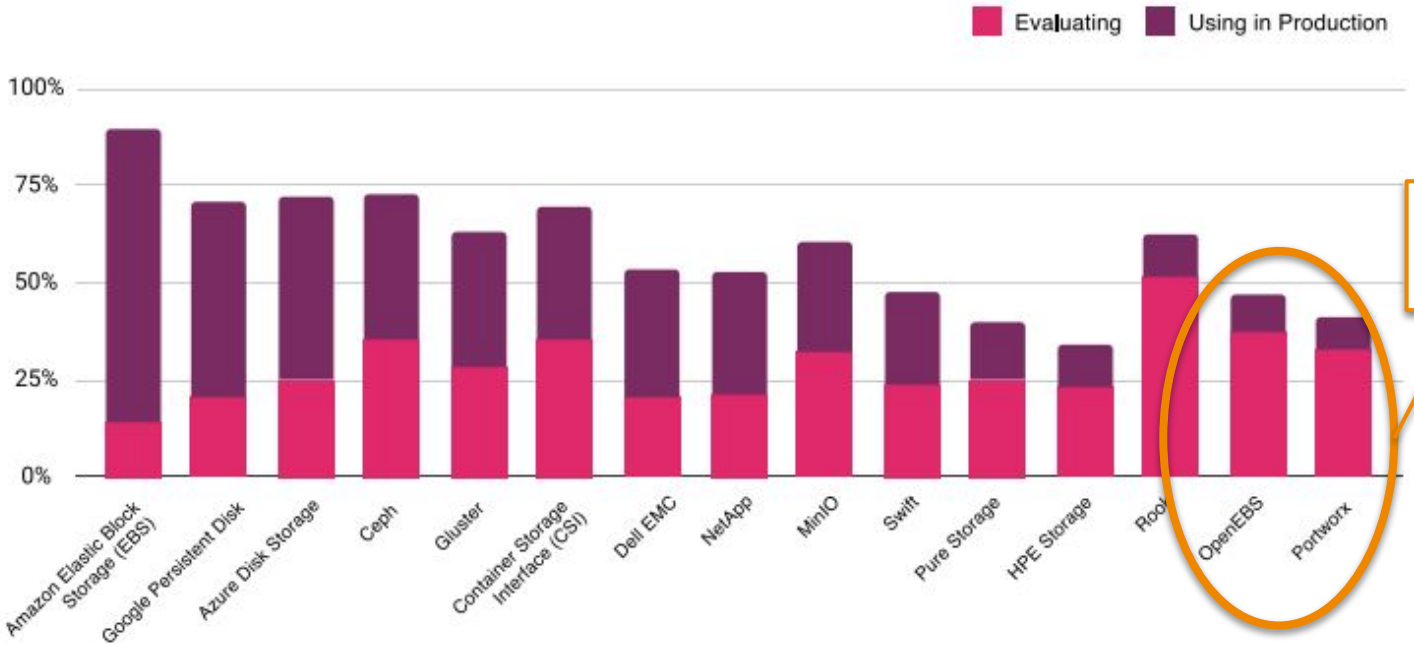
Of course you can. And you do. However you lose so many benefits of moving to Kubernetes.

Most workloads just use Direct Attached Storage instead.

CNCF 2019 Survey

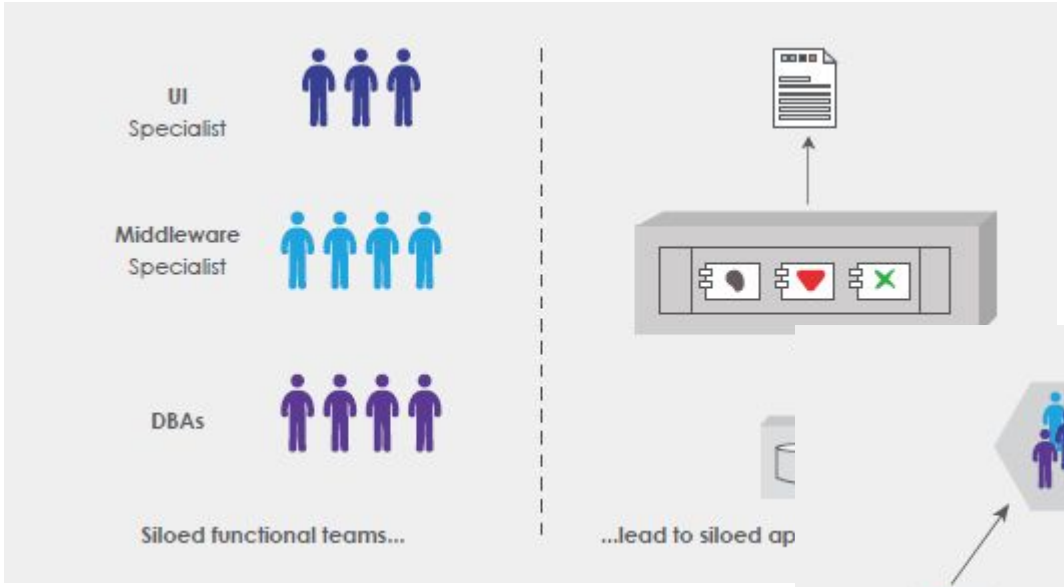
Cloud Native Storage

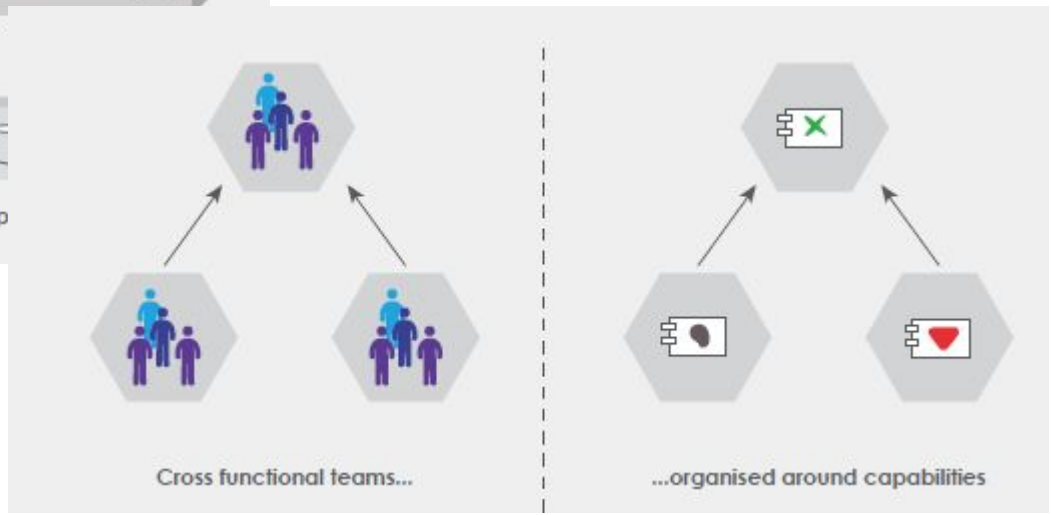
Given a considerable increase in the number of cloud native storage projects, we changed the storage question this year to include each storage project or product listed on the CNCF landscape. 14% of respondents are using storage projects in production, with another 27% evaluating storage projects. Only 5% of respondents indicated they were not planning on using or evaluating any storage projects.



Container Attached Storage most evaluated

Convway's Law - Culture Shift





Convway's Law - Culture Shift

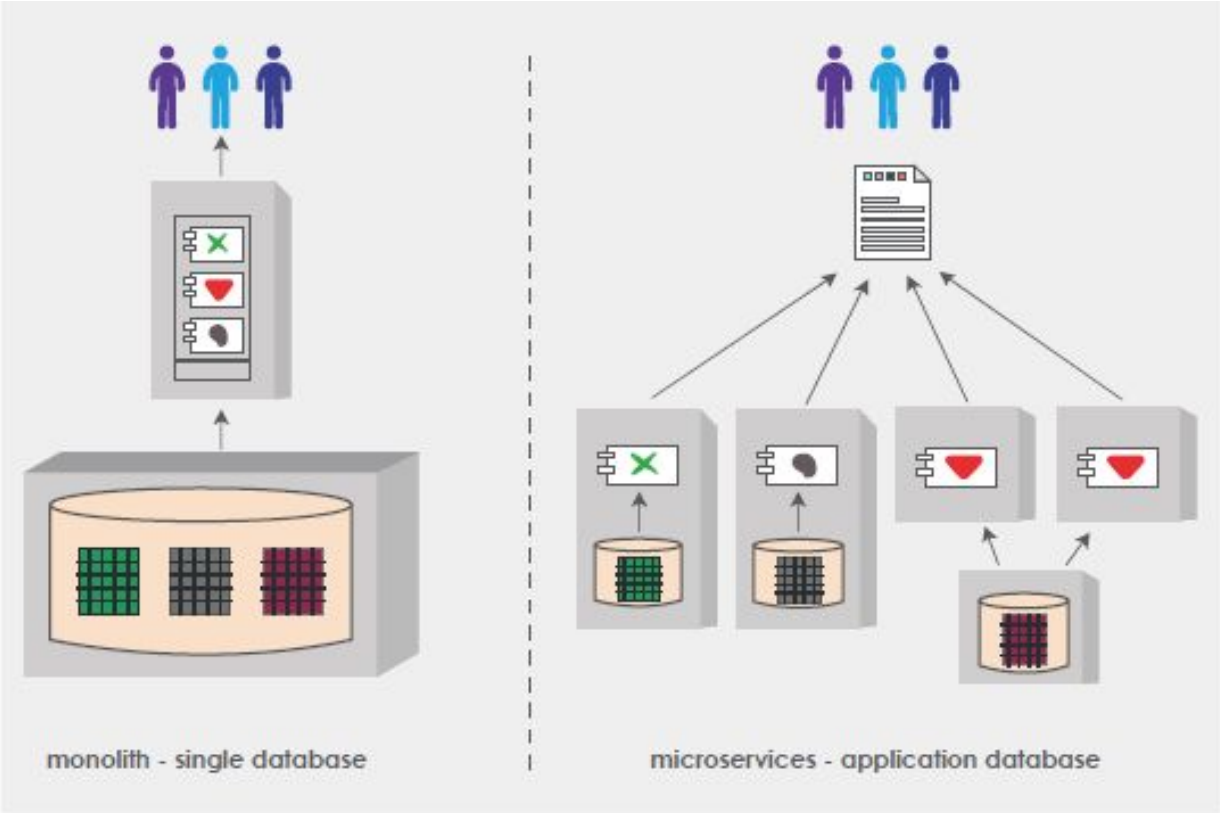
https://www.youtube.com/watch?v=0CEHN6ECaPs

https://www.youtube.com/watch?v=z_LbRfDKPvE

Steven Bower at Bloomberg

- Moved to Kubernetes in order to simplify and standardize their environments
- CNCF end user of the year 18/19
- Running dozens of different stateful workloads at scale
- Believes in open source
- Not about cost savings - about agility
- Everything loosely coupled
- Teams are autonomous and full stack
- Does not use shared storage
- Uses OpenEBS - different flavors

Convway's Law - Data



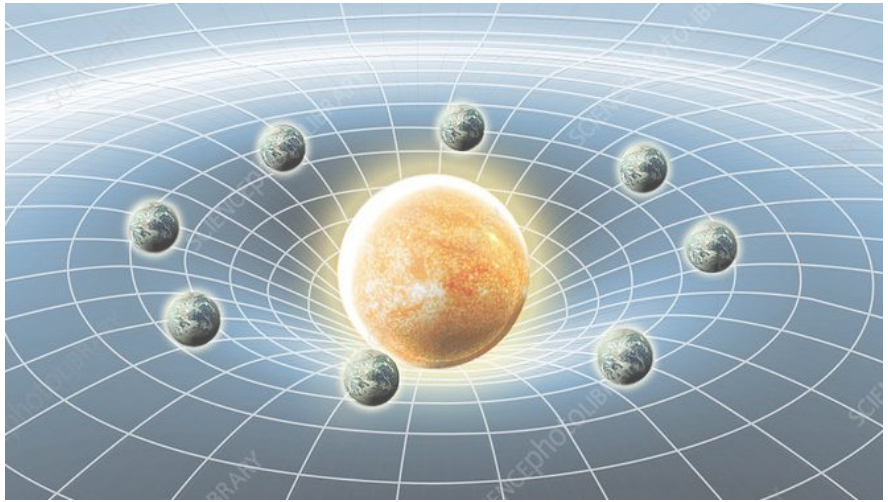
loosely coupled teams

loosely coupled applications

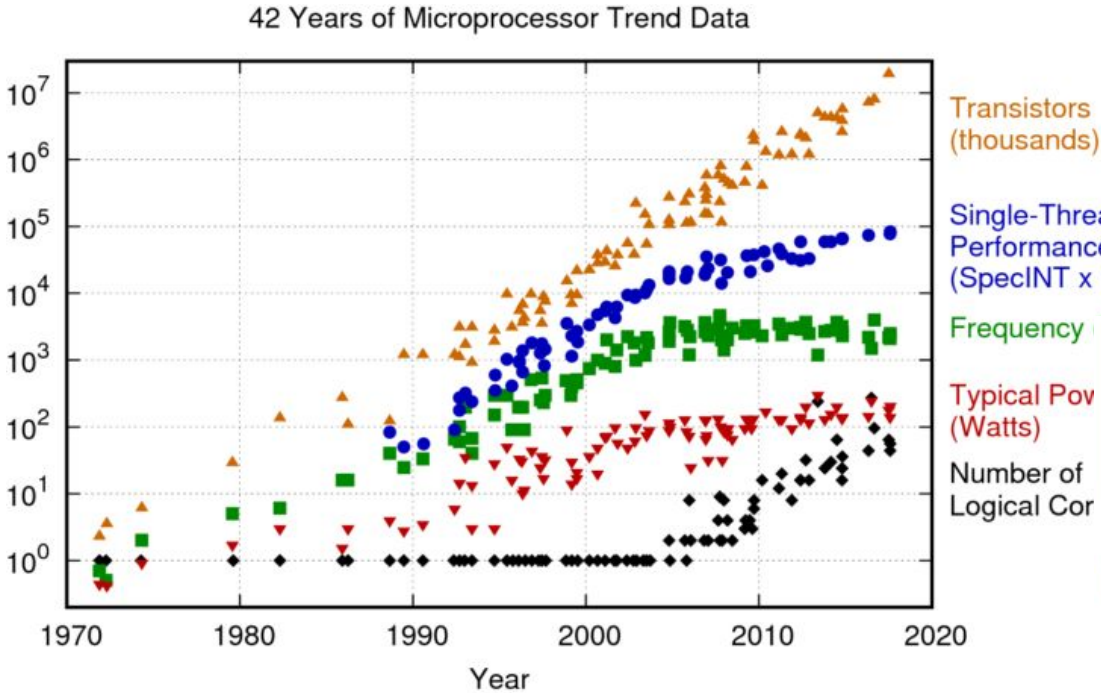
loosely coupled data

Data Gravity

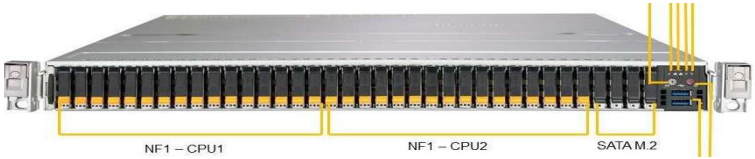
- As data grows — it has the tendency to pull applications towards it (gravity)
- Everything will evolve around the sun and it dominates the planets
  - Latency, throughput, IO blender
  - If the sun goes super nova — all your apps circling it will be gone instantly
- Some solutions involve replicating the sun towards some other location in the “space time continuum”
  - It works — but it exacerbates the problem

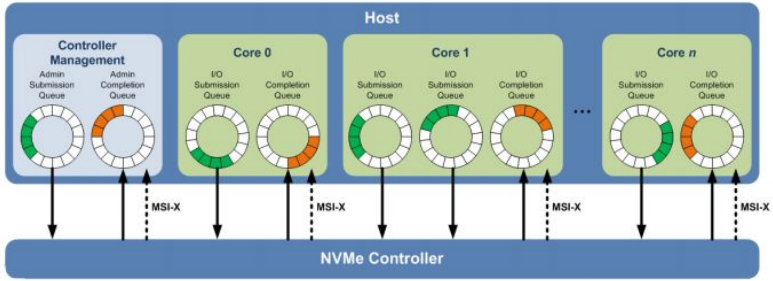


Evolving Hardware



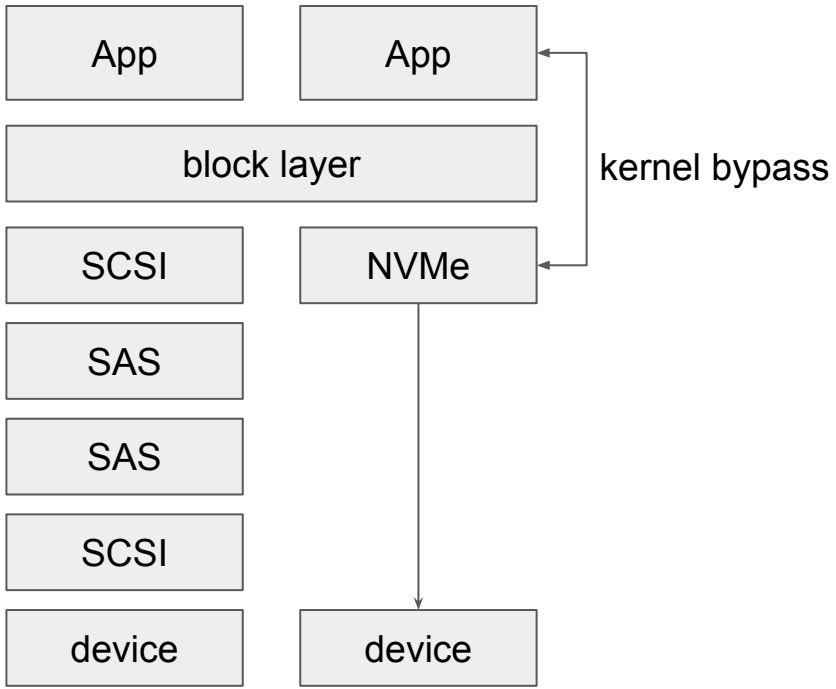
Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2017 by K. Rupp





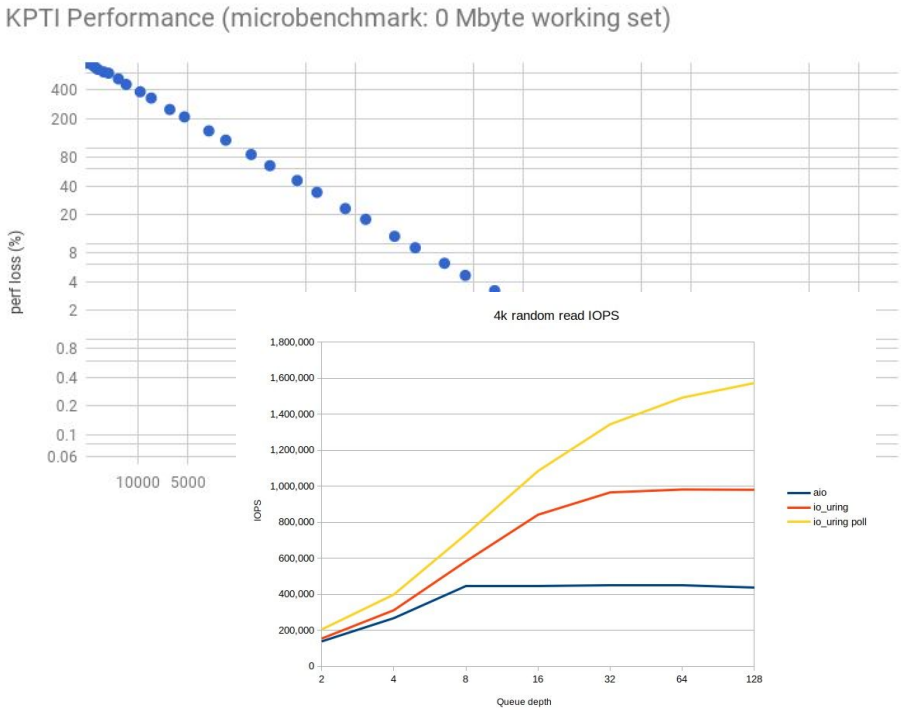
Evolving Hardware

- NVMe is a protocol that dictates how bits are moved between the CPU/device but also -- between devices
  - Its origin can be found with Infi Band used in HPC for many years (1999)
- NVMe over Fabrics extends the protocol over TCP, RDMA, FC, virtio
- A complete replacement of the SCSI protocol which goes back all the way to 1978



Software Paradigm Shift

- High number of system calls have a huge impact on performance
- Two solutions to mitigate this:
  - Making use of huge pages
  - Try to do as much as possible in **user space**
- Meta languages, Go, Rust,...
- **io_uring** a new interface added to the kernel to “catch up” with the high speed devices, poll mode FTW.



Rewrite! Resistance is Futile

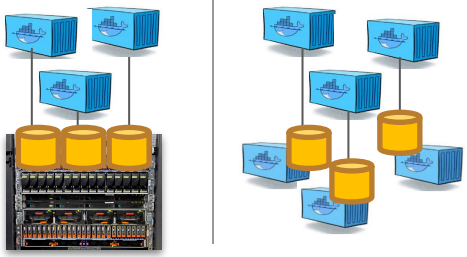
- Packets come in at a very high rate, single CPU 100% how to scale?
  - CPU has ~67ns per packet @3GHz
- Solution: spread across multiple cores which requires locking
  - **Locks are expensive** and locks are in memory which is 70-40ns away?
- Amdahl's law starts to dominate the performance envelope
- Context switches and system calls have gotten far more expensive post spectere meltdown
- What we seem to need are lockless queues that scale per core
  - **Poll mode drivers**
- Partial rewrites are inevitable, the rewards are high
  - *ScyllaDB, VPP, Open vSwitch,*

CAS : Motivation

KubeCon CloudNativeCon Europe 2020 Virtual

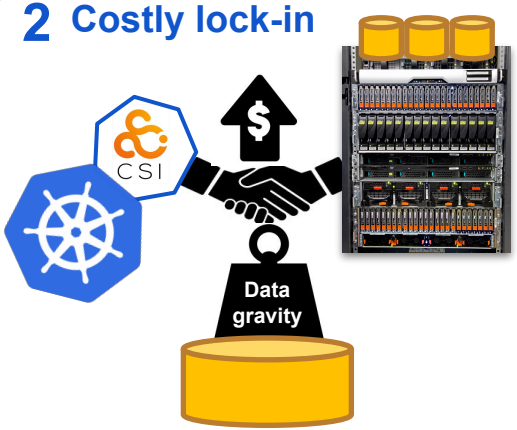
1 Conway's Law

Shared everything vs Per workload, per team



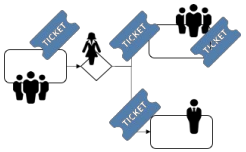
External storage — ***Container native***

2 Costly lock-in



3 Process mismatch & 100x more dynamism

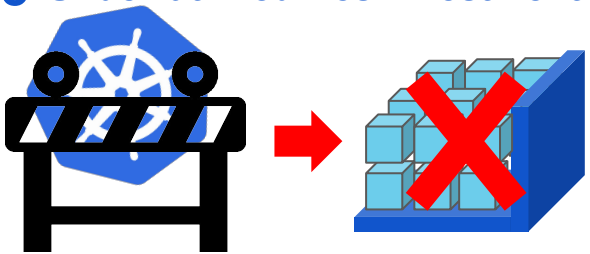
Traditional processes vs Automated Kube - Ops



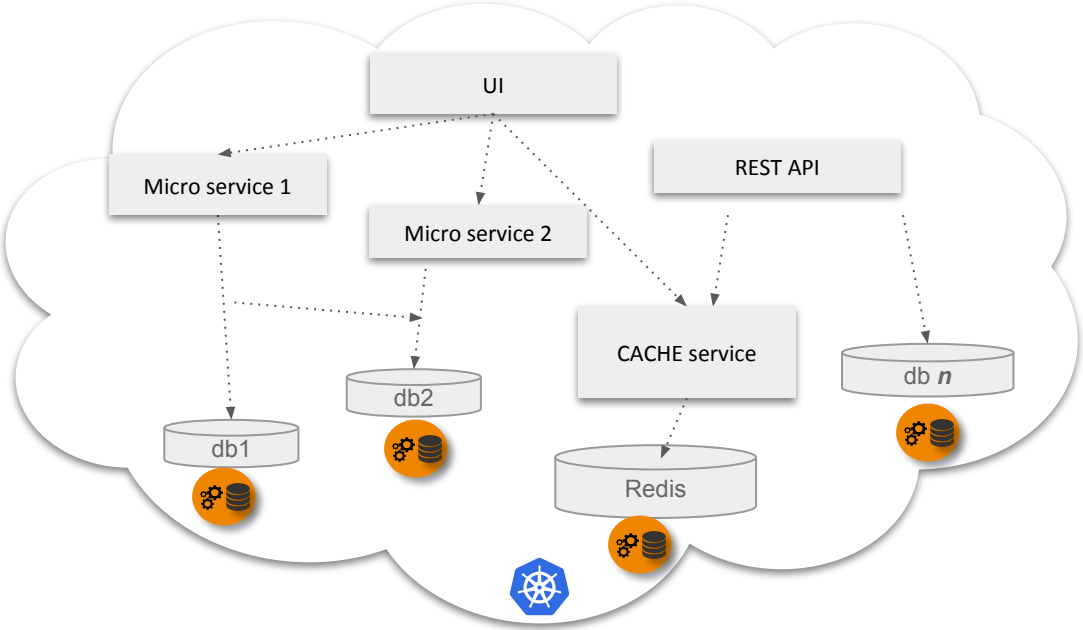
GitOps

4 External know-how required for traditional storage

5 Under-utilized K8s investment!



CAS : Conway's Law for storage



Every workload & team its own system

Different engines for different workloads

Built *on* Kubernetes *for* Kubernetes

Delivers the benefits of for data
- No lock-in
- Open source
- Runs consistently everywhere
  - Any underlying cloud or disk or SAN

AND the right architecture for NVMe

CAS : Early Adopters

OpenEBS Adopters

This is the list of organizations and users that publicly shared details of how they are using OpenEBS for running their Stateful workloads.

Organization	Stateful Workloads	Success Story
Agnes Intelligence	Apache Kafka, Apache Solr, NFS	English
Arista Networks	Gerrit (multiple flavors), NPM, Maven, Redis, NFS, Sonarqube, Internal tools	English
CLEW Medical	PostgreSQL, Keycloak, RabbitMQ	English
Clouds Sky GmbH	Confluent Kafka, Strimzi Kafka, Elasticsearch, Prometheus	English
CNCF, The Linux Foundation	PostgreSQL, MariaDB, ElasticSearch, Redis, DevStats	English
Code Wave	Bitwarden, Bookstack, Allegros Ralph, Limesurvey, Grafana, Hackmd/Codimd, Minio, Nextcloud, Percona XtraDB Cluster Operator, Nextcloud, Sonarqube, Sentry, Jupyterhub	English
Comcast	Prometheus, Alertmanager, Influxdb, Helm Chartmuseum	English
CORT	Magento, Elasticsearch, MariaDB	English
DISID	Minio, DataStax, Greenplum, Gridgain, mongoDB, Qlickhouse, PosgtreSQL	English

Hyperconverged and Local

- Postgres
- MySQL
- Kafka
- Redis
- ElasticSearch
- Prometheus
- Thanos

The vast majority of applications are able to better handle failover and replication than a block level device.

Instead of introducing another distributed system into an already complex environment, OpenEBS's Local PVs allow us to leverage fast local storage.

Additionally, by leveraging ZFS we are able to have encryption at rest for all of our workloads, compression, and the peace of mind of a COW based file system. OpenEBS has allowed us to **not introduce a complicated distributed system** into our platform.

The adoption has been smooth and completely transparent to our end users.

Kubernetes for DBaaS

Home / Blog / 2ndQuadrant / Local Persistent Volumes and PostgreSQL usage in Kubernetes

Local Persistent Volumes and PostgreSQL usage in Kubernetes

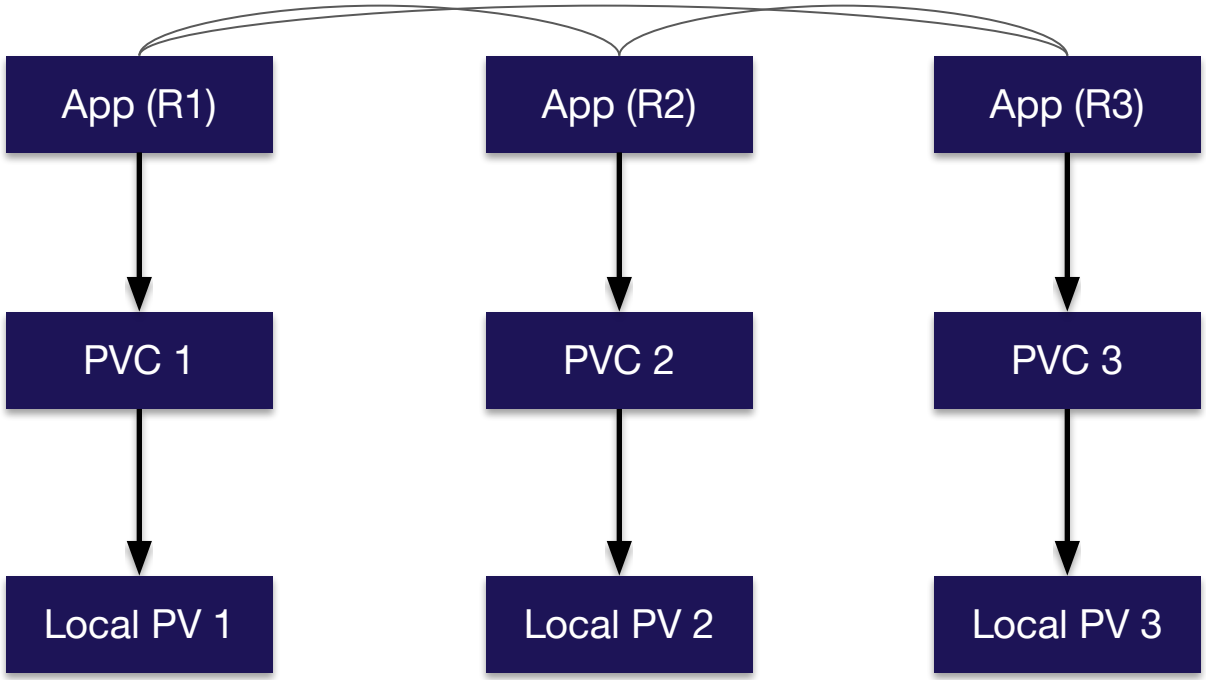
June 22, 2020 / in 2ndQuadrant, Cloud Native / by Gabriele Bartolini

Can I use PostgreSQL in Kubernetes and expect to achieve performance results of the storage that are comparable to traditional installations on bare metal or VMs? In this article I go through the benchmarks we did in our own Private Cloud based on Kubernetes 1.17 to test the performance of local persistent volumes using OpenEBS Local PV.

OpenEBS 101

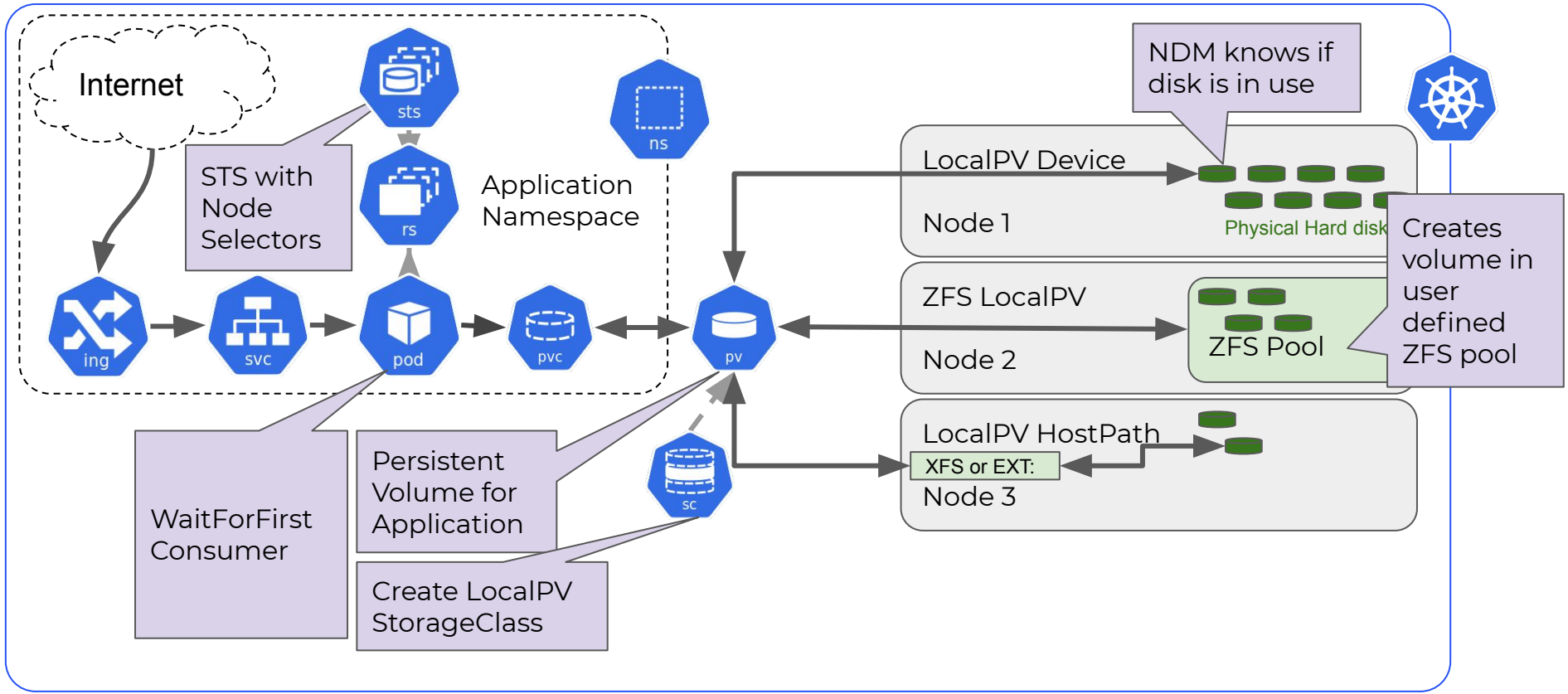
Storage Engines for every Workload

Distributed Workloads

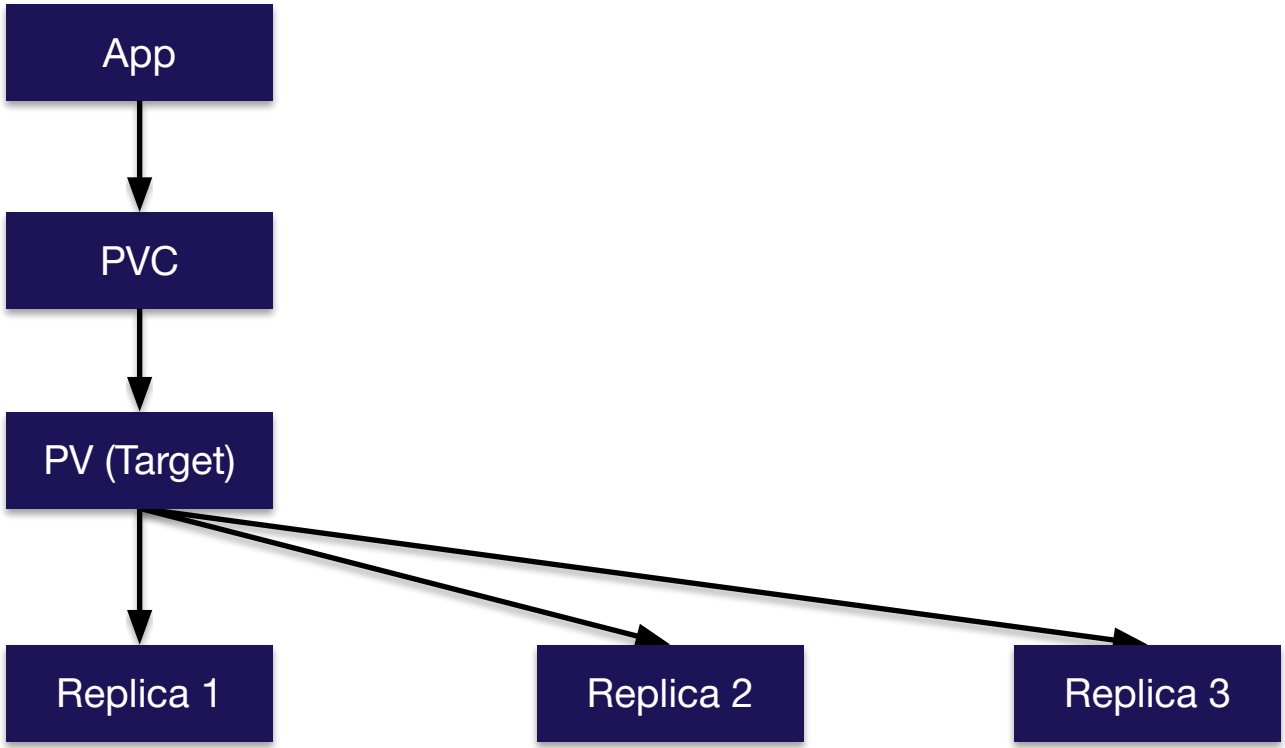


- Hostpath
- Block Device
- Local ZFS / LVM

OpenEBS Local PV Variants

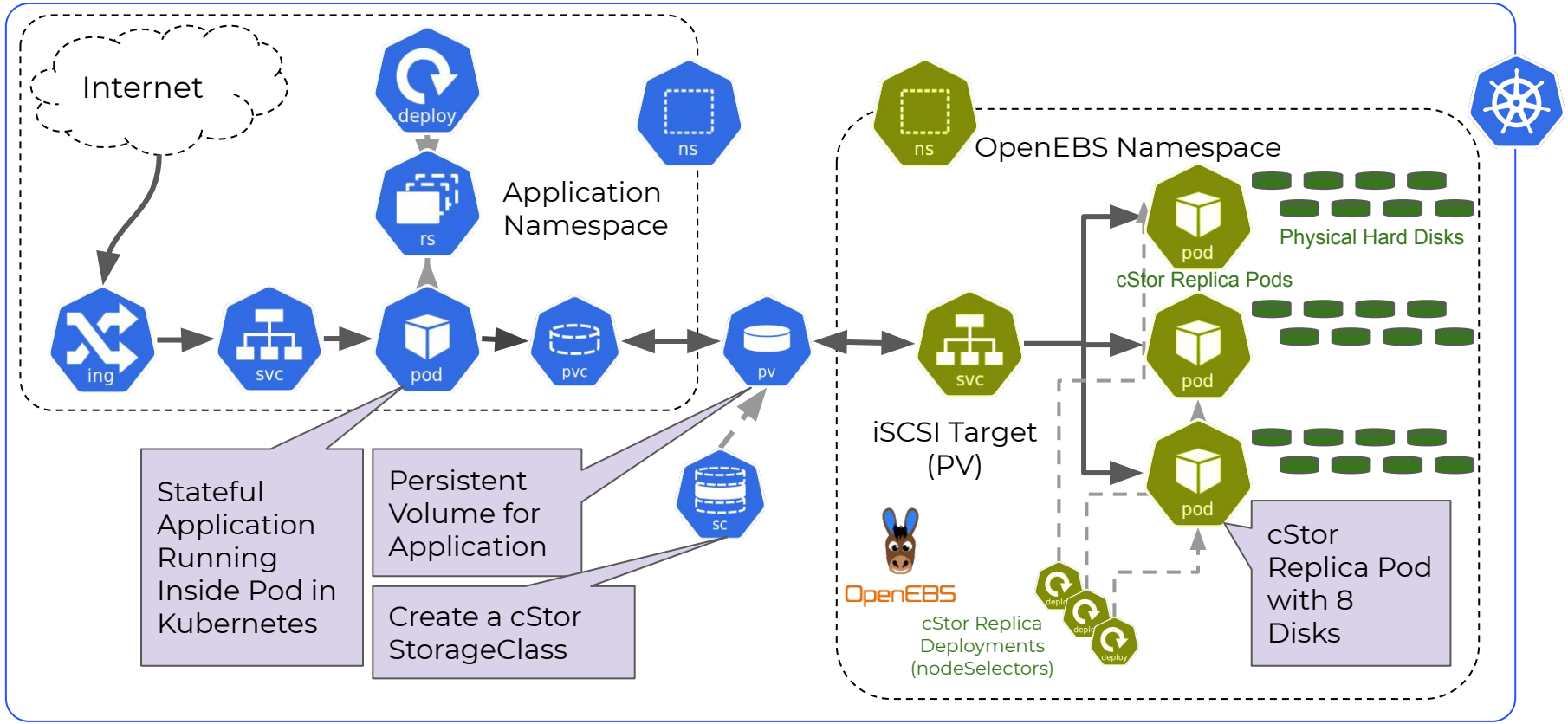


Persistent (non-distributed) Apps



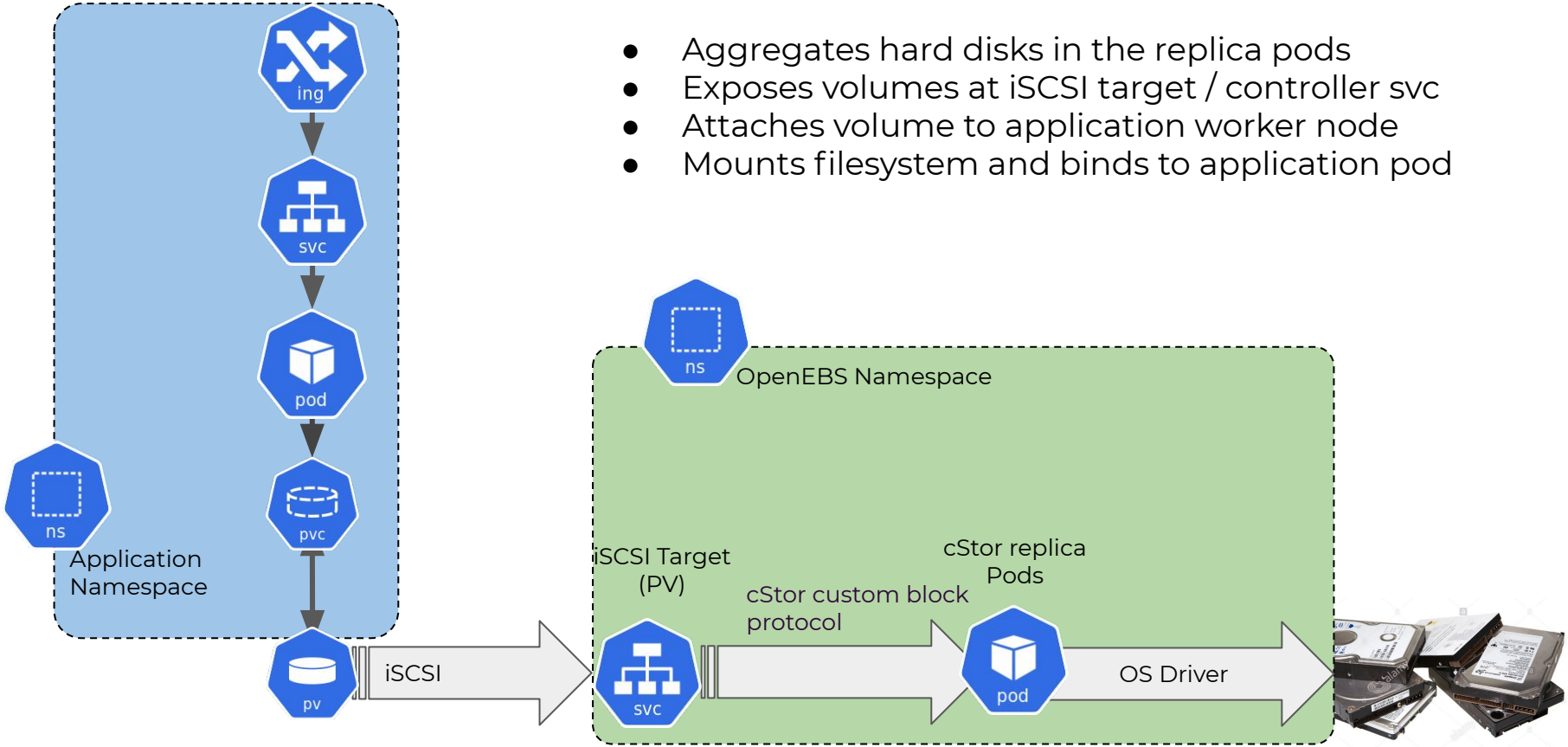
- Jiva/Longhorn
- cStor
- Mayastor

OpenEBS - Replicated Storage



OpenEBS - cStor

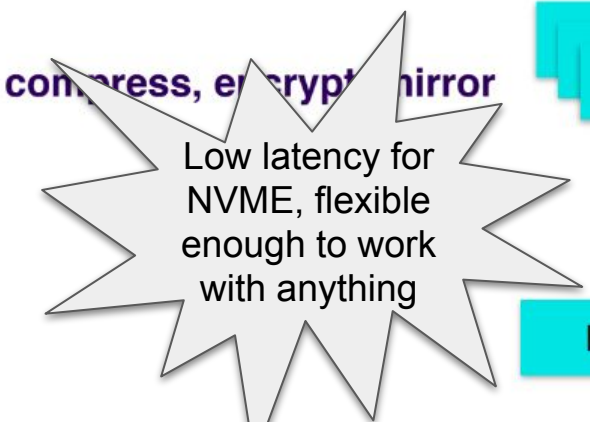
- Aggregates hard disks in the replica pods
- Exposes volumes at iSCSI target / controller svc
- Attaches volume to application worker node
- Mounts filesystem and binds to application pod



OpenEBS - Mayastor

New
Storage Engine for
Performance with Features.

Same
Declarative,
Composable Data Plane with
Developer Friendly Mgt and
API-Driven Orchestration.



PV

Ingress

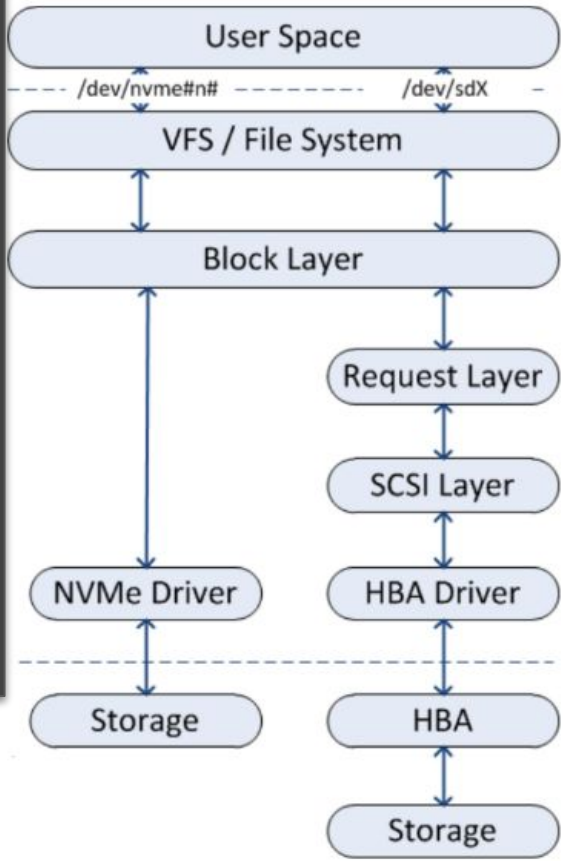
T(x)

Egress

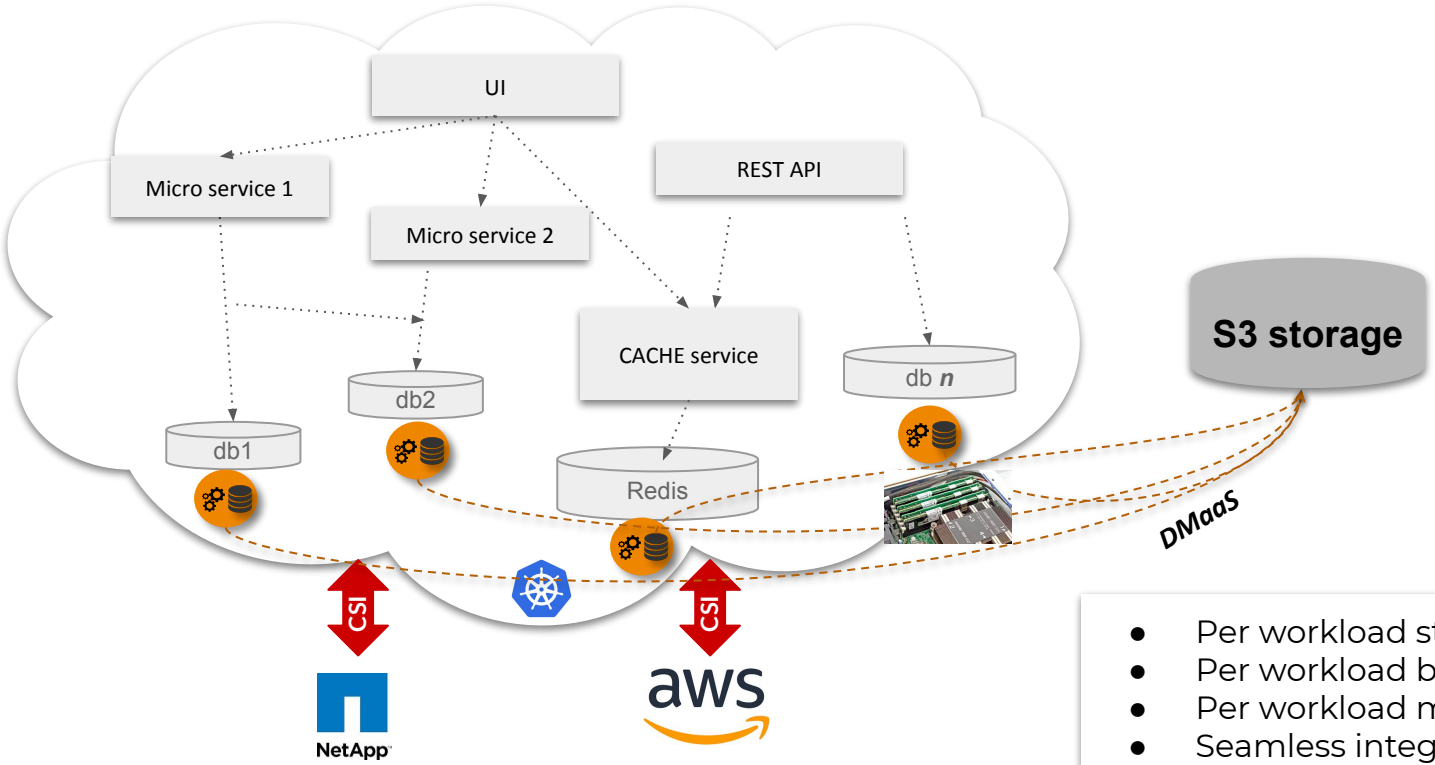
local

remote

```
sequential read 2-way mirror:
(groupid=0, jobs=1): err= 0:
pid=32573: Thu Feb 13 21:23:41
2020
  read: IOPS=79.2k,
BW=309MiB/s
(324MB/s)(9280MiB/30002msec)


sequential write 2-way mirror:
(groupid=1, jobs=1): err= 0:
pid=32651: Thu Feb 13 21:23:41
2020
  read: IOPS=77.8k,
BW=304MiB/s
(319MB/s)(9115MiB/30002msec)


random write 2-way mirror:
(groupid=2, jobs=1): err= 0:
pid=32718: Thu Feb 13 21:23:41
2020
 write: IOPS=47.6k,
BW=186MiB/s
(195MB/s)(5582MiB/30003msec)


random read 2-way mirror:
(groupid=0, jobs=1): err= 0:
pid=1020: Thu Feb 13 21:23:41
2020
  read: IOPS=76.9k,
BW=301MiB/s
(315MB/s)(9017MiB/30002msec)
```



Data Protection

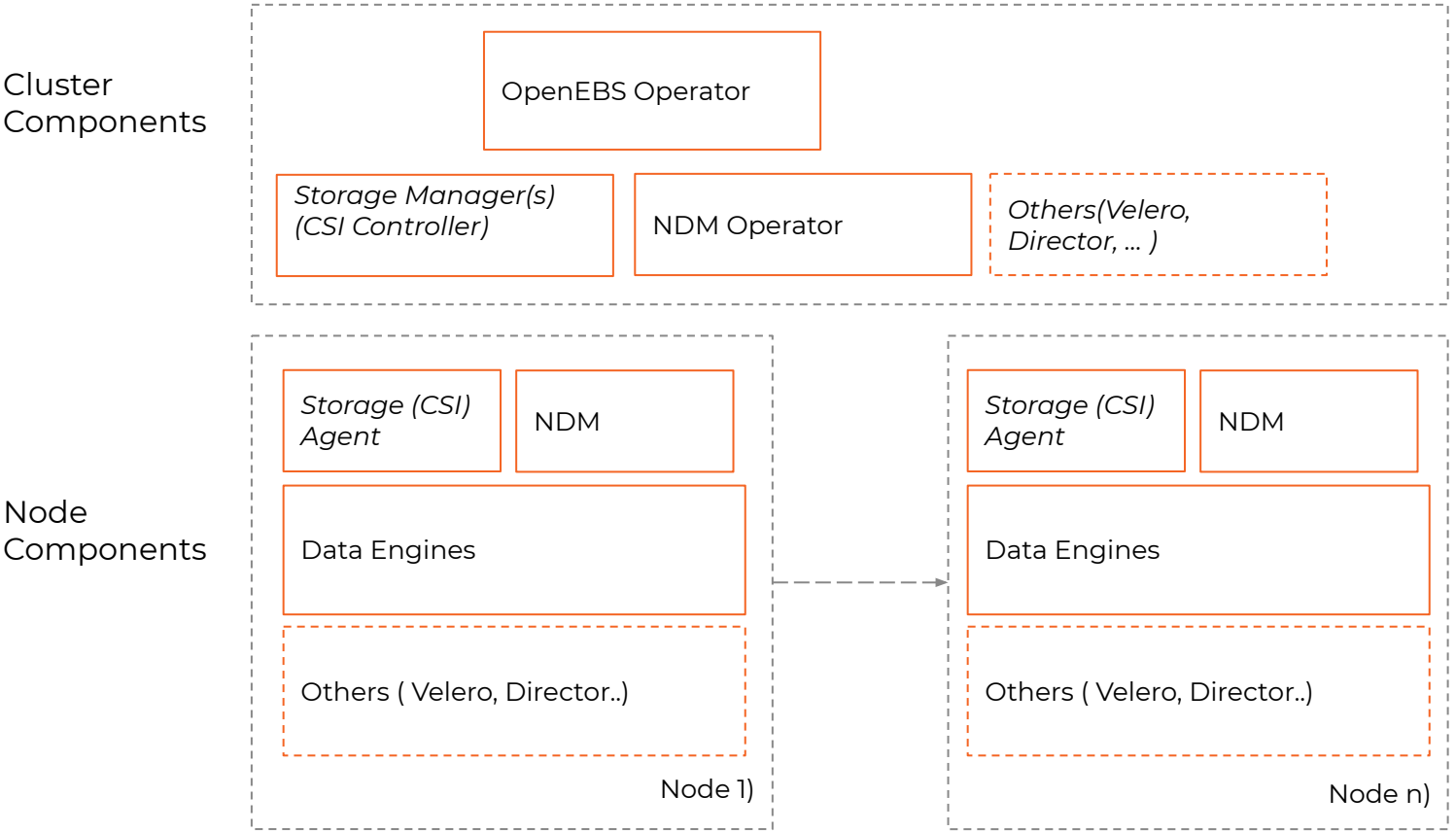


- Per workload storage
- Per workload backup
- Per workload management
- Seamless integration w Optane & bare metal & CSI provisioned clouds and legacy storage

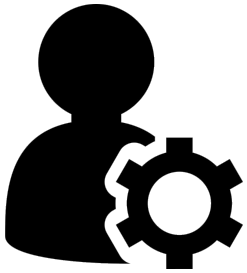
Q/A

- https://openebs.io
- https://docs.openebs.io
- https://github.com/openebs/openebs
- Kubernetes Slack #openebs
- Kubernetes Slack #openebs-dev
- https://github.com/openebs/openebs/blob/master/ADOPTERS.md
- https://www.meetup.com/Data-on-Kubernetes-community/

Architecture



Architecture



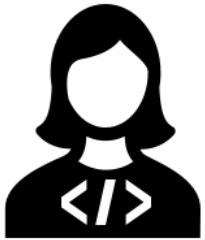
Cluster admin

Setup OpenEBS

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

(2) SPC=>StoragePool(s)

(3) StorageClass

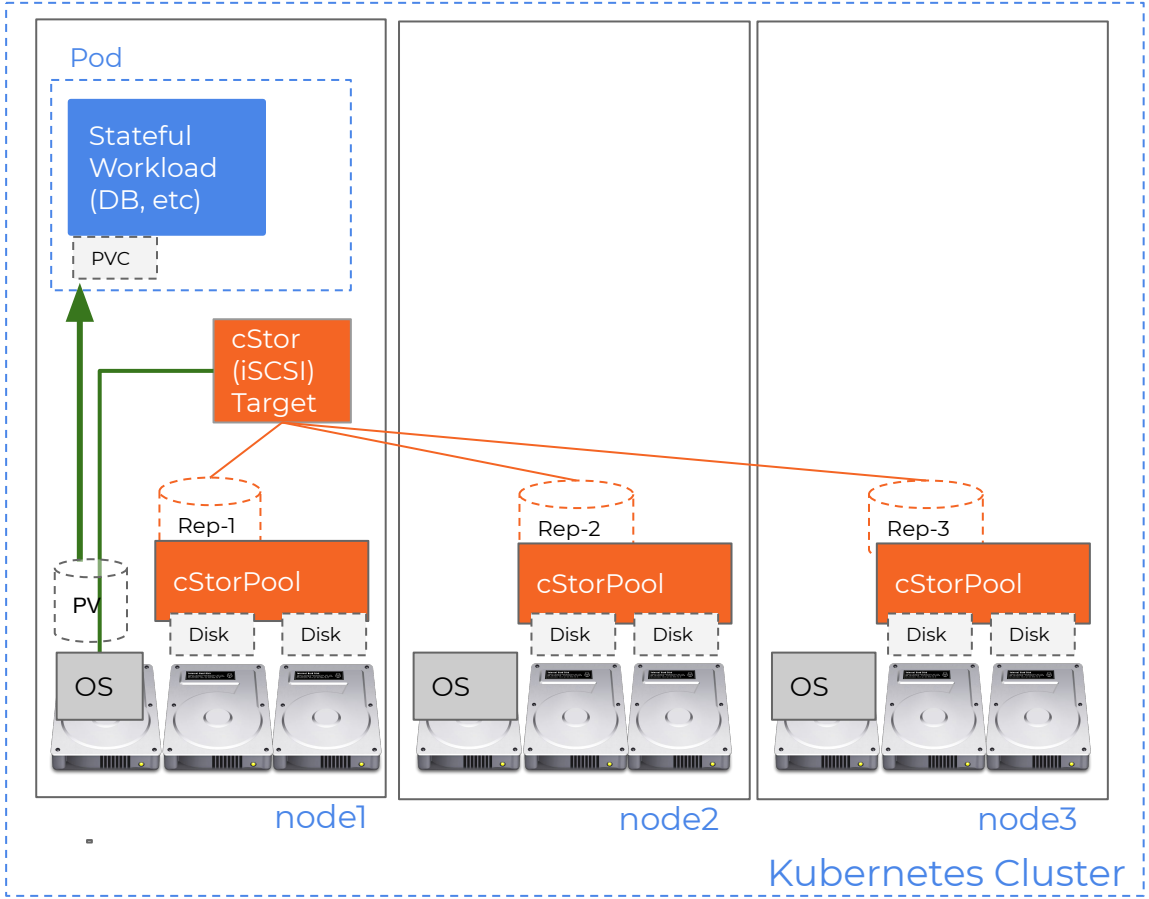


Developer

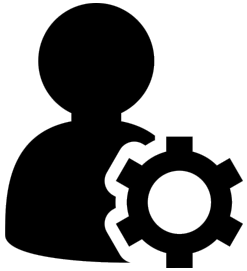
Using OpenEBS

(4) Pod with OpenEBS PVC

(5) PV



Architecture



DevOps admin

Setup OpenEBS

(1) node-disk-manager, provisioner,

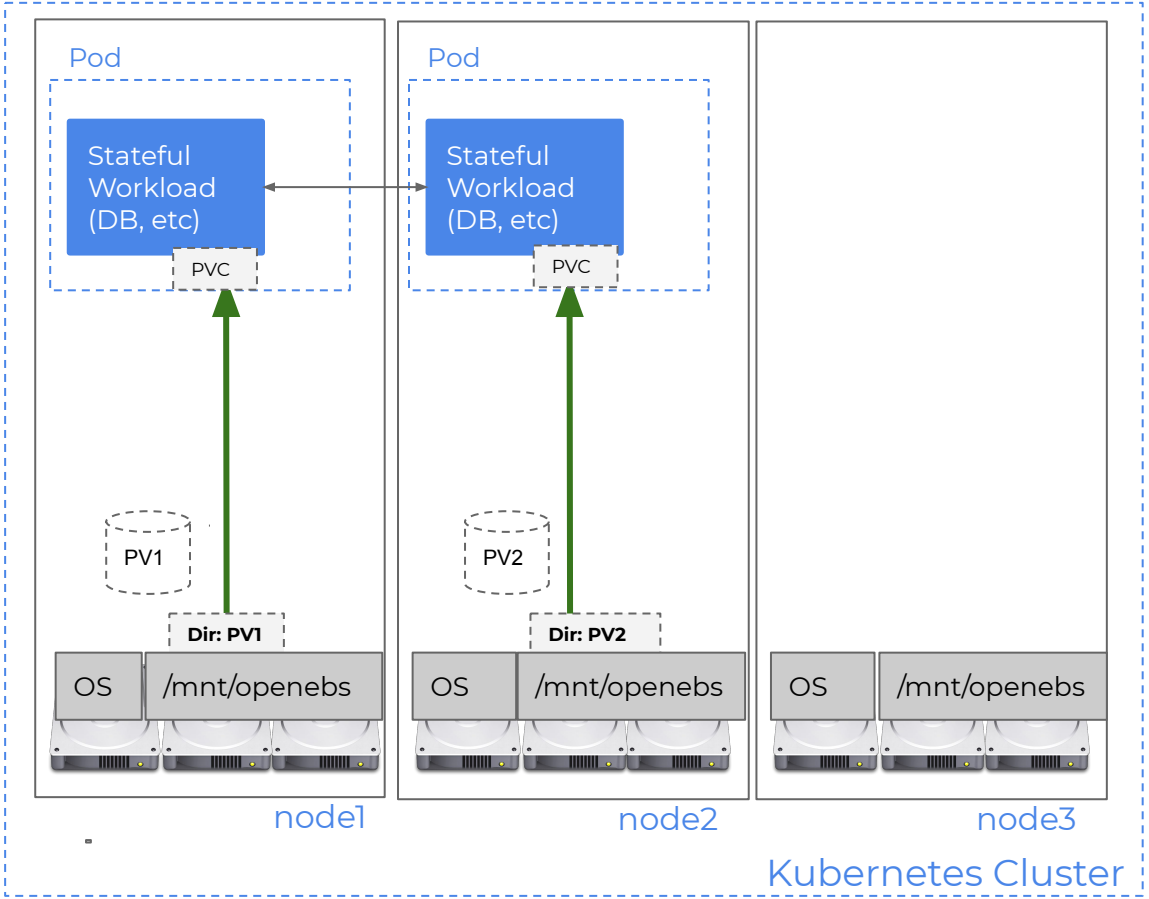
(2) StorageClass

Using OpenEBS

(3) StatefulSet with PVC

(4) PV

Developer



Performance Benchmarking

Kubernetes Cluster Details

- m5ad.2xlarge on AWS (8 cores, 32GiB RAM, 300GiB NVMe SSD)
- Kubernetes 1.16.8
- Amazon Linux 2

FIO and pgbench

- Fio profiles for Postgres were generated with pgbench and blktrace.
- Fio command to replay the profile.

https://github.com/openebs/performance-benchmark/tree/master/benchmark-tool

Performance Benchmarking

- Tuning the nodes for performance - as part of your Terraform / Ansible
- Test for scale - number of workloads
- Day 2 Operations in Progress
- Noisy neighbour / Load
- Chaos

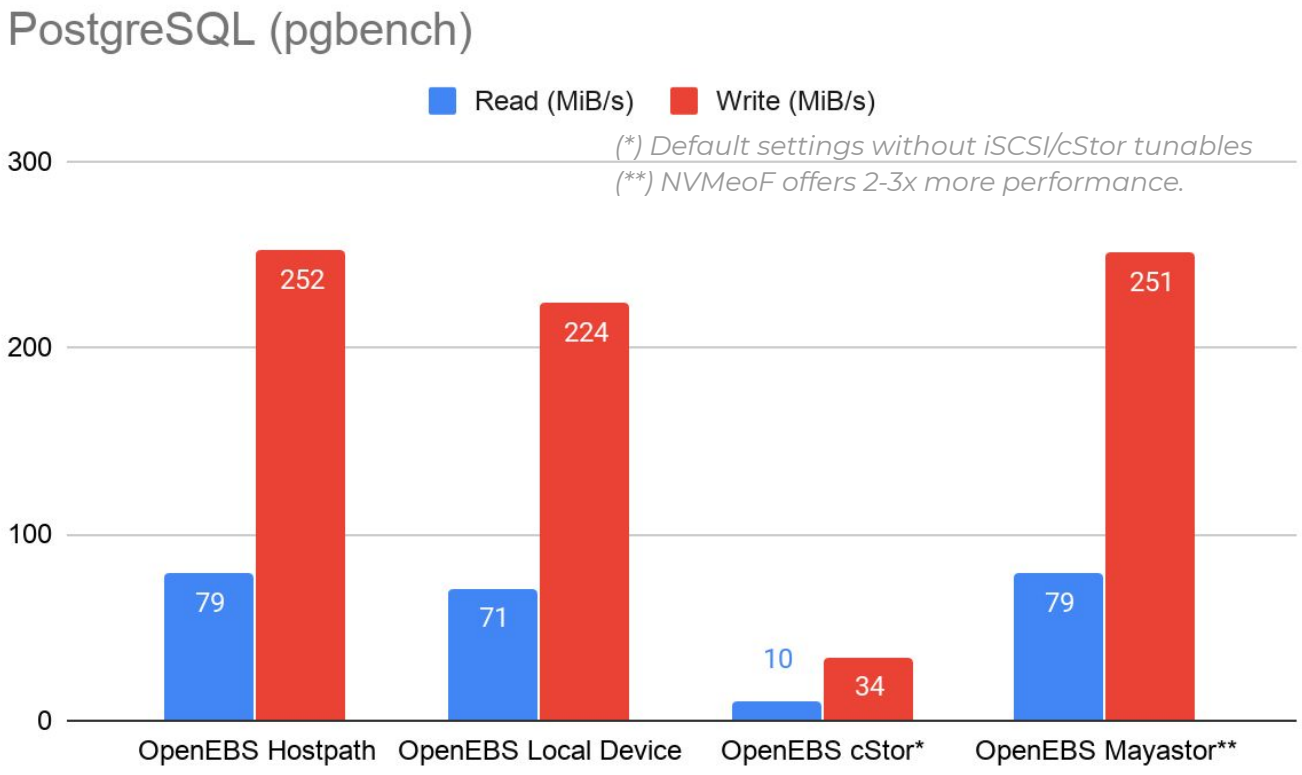
in /etc/iscsi/iscsid.conf and change:

```
node.session.cmds_max = 4096
node.session.queue_depth = 128
```

etc/sysctl.conf:

```
net.ipv4.tcp_timestamps = 1
net.ipv4.tcp_sack = 0
net.ipv4.tcp_rmem = 10000000 10000000 1000000
net.ipv4.tcp_wmem = 10000000 10000000 1000000
net.ipv4.tcp_mem = 10000000 10000000 10000000
net.core.rmem_default = 524287
net.core.wmem_default = 524287
net.core.rmem_max = 524287
net.core.wmem_max = 524287
net.core.optmem_max = 524287
net.core.netdev_max_backlog = 300000
```

Performance Benchmarking



Performance Benchmarking

	IOPS		Throughput (MiB/s)	
	Read	Write	Read	Write
OpenEBS Hostpath	10200	16300	79.9	252
OpenEBS Device	9042	14500	71	224
OpenEBS cStor*	1383	2214	10.9	34.3
OpenEBS Mayastor**	10200	16200	79.5	251

(*) Default settings without iSCSI/cStor tunables
(**) NVMeoF offers 2-3x more performance.