What’s the future of container and cloud native storage?

Cheryl Hung, Product Manager at StorageOS

Cheryl

Product manager

@oicheryl

STORAGEOS

Objectives

Why is container storage tricky?

How can I evaluate my storage options?

How should storage work in a microservices architecture?

Why do I need storage?

Why do I need storage?

Why do I need storage?

App binaries

App data

Config

Backup

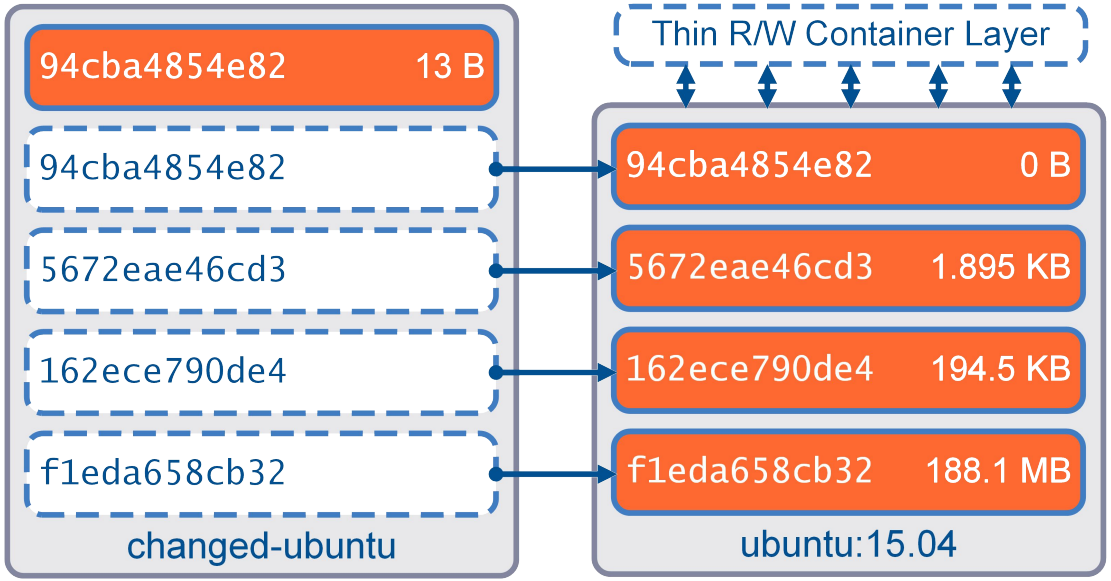
Why is this tricky with containers?

No pet storage

Data follows

Humans are fallible

Docker container layers

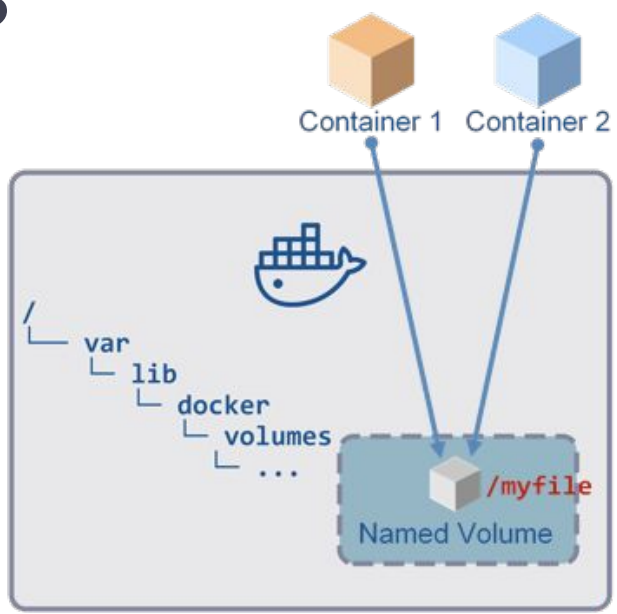


Docker local volumes

```
$ docker volume create --name mydata
```

```
$ docker run --rm -v mydata:/data:rw alpine ash -c \
  "echo hello world > /data/myfile"
```

```
$ sudo cat /var/lib/docker/volumes/mydata/_data/myfile
hello world
```



Eight principles of Cloud Native Storage

What is Cloud Native?

Horizontally scalable

No single point of failure

Resilient and self healing

Minimal operator overhead

Decoupled from the underlying platform

Jane, DevOps engineer in a bank

How do I migrate the Postgres database to containers?

Eight principles of Cloud Native Storage

1. API driven

Eight principles of Cloud Native Storage

1. API driven
2. **Declarative and composable**

Eight principles of Cloud Native Storage

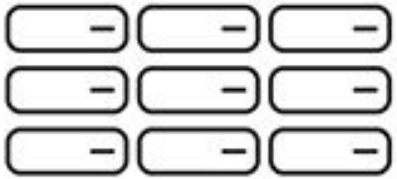
1. API driven
2. Declarative and composable
3. **Application centric**

Eight principles of Cloud Native Storage

1. API driven
2. Declarative and composable
3. Application centric
4. **Agile**

Eight principles of Cloud Native Storage

1. API driven
2. Declarative and composable
3. Application centric
4. Agile
5. **Performant**



Block storage

Data stored in fixed-size 'blocks' in a rigid arrangement—ideal for enterprise databases

File storage

Data stored as 'files' in hierarchically nested 'folders'—ideal for active documents

Object storage

Data stored as 'objects' in scalable 'buckets'—ideal for unstructured big data, analytics and archiving

Eight principles of Cloud Native Storage

1. API driven
2. Declarative and composable
3. Application centric
4. Agile
5. Performant
6. **Natively secure**

Eight principles of Cloud Native Storage

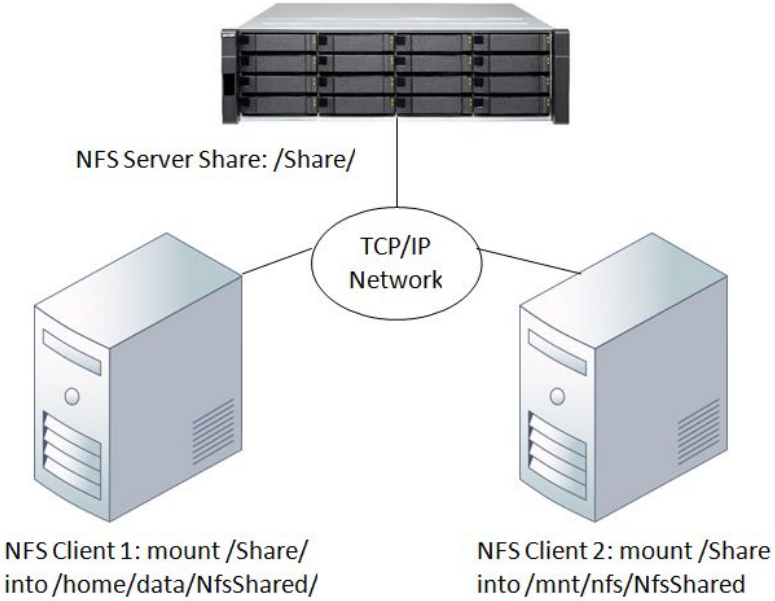
1. API driven
2. Declarative and composable
3. Application centric
4. Agile
5. Performant
6. Natively secure
7. **Consistently available**

Eight principles of Cloud Native Storage

1. API driven
2. Declarative and composable
3. Application centric
4. Agile
5. Performant
6. Natively secure
7. Consistently available
8. **Platform agnostic**

Storage landscape

Centralised file system: NFS



Centralised file system: NFS

0

Single point of failure
Hard to scale horizontally
No native integration

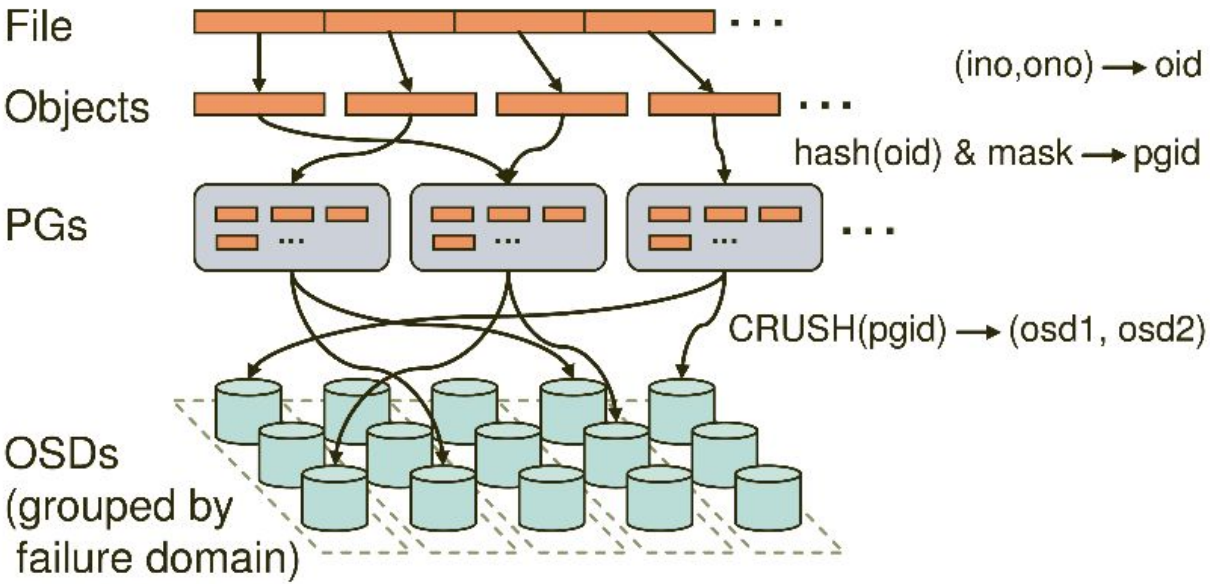
Storage array: Dell EMC

Storage array: Dell EMC

2

Deterministic performance
Vendor lock in
No thin provisioning
Hard to scale horizontally
Expensive and long lead times

Distributed: Ceph

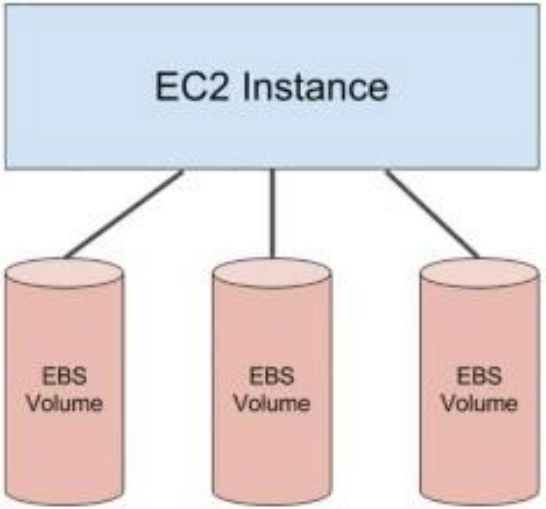


Distributed: Ceph

4

Horizontally scalable
Hardware agnostic
Complicated to set up (see: Rook)
Writes fan out 13-40 times
Failures are expensive

Public cloud: AWS EBS

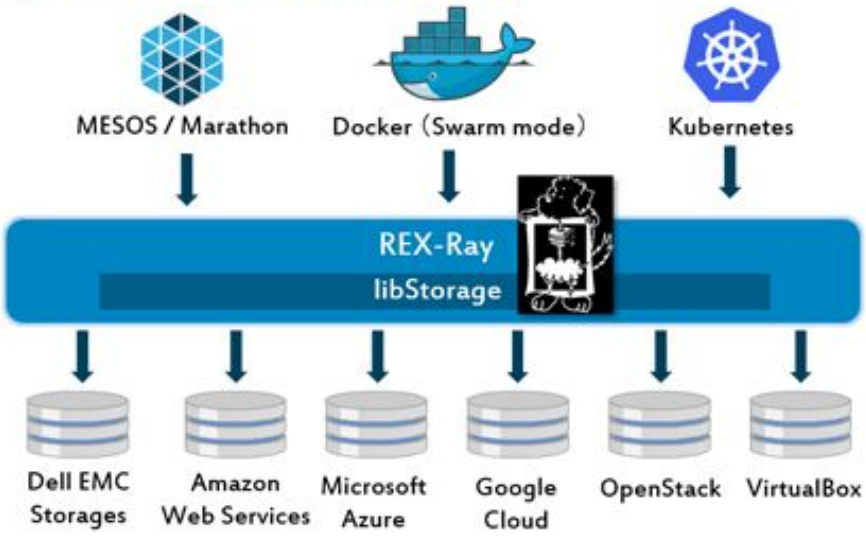


Public cloud: AWS EBS

6

Horizontally scalable
Consistent and performant
40 EBS volumes per EC2 instance
Mounting physical block devices is slow
Expensive, vendor lock in, compliance

Plugin framework: REX-Ray



Volume plugin: StorageOS

A software-defined, scale-out storage platform for running enterprise containerized applications in production

Volume plugin: StorageOS

~~*Software defined*~~	Platform agnostic
scale-out storage	
enterprise	
containerized applications	
in production	

Volume plugin: StorageOS

~~*Software defined*~~	Platform agnostic
~~*scale-out storage*~~	Horizontally scalable
enterprise	
containerized applications	
in production	

Volume plugin: StorageOS

~~*Software defined*~~	Platform agnostic
~~*scale-out storage*~~	Horizontally scalable
~~*enterprise*~~	Optimized for databases ie. block storage
containerized applications	
in production	

Volume plugin: StorageOS

~~*Software defined*~~	Platform agnostic
~~*scale-out storage*~~	Horizontally scalable
~~*enterprise*~~	Optimized for databases ie. block storage
~~*containerized applications*~~	Docker and K8s integration
in production	

Volume plugin: StorageOS

~~*Software defined*~~	Platform agnostic
~~*scale-out storage*~~	Horizontally scalable
~~*enterprise*~~	Optimized for databases ie. block storage
~~*containerized applications*~~	Docker and K8s integration
~~*in production*~~	Highly available

Volume plugin: StorageOS

Provisioning storage

Step 1 of 5

Storage pool

For this tutorial we will use the StorageOS CLI to explore the cluster. This is already installed; run `storageos --help ↵` to see available options.

Open terminals on the second and third hosts, and check that the StorageOS container is running:

```
ssh root@host02 ↵ ssh root@host03 ↵
docker ps ↵
```

On start-up, StorageOS nodes communicate with each other to discover the overall status of the cluster and establish consensus. List the nodes in this cluster:

```
storageos node ls --format "table
{{.Name}}\t{{.Address}}\t{{.Capacity}}\t{{.C
↵
```

Once the cluster is established, StorageOS creates a default storage pool from all the nodes in the cluster:

```
storageos pool ls --format "table
{{.Name}}\t{{.Nodes}}\t{{.Total}}\t{{.Capaci
↵
```

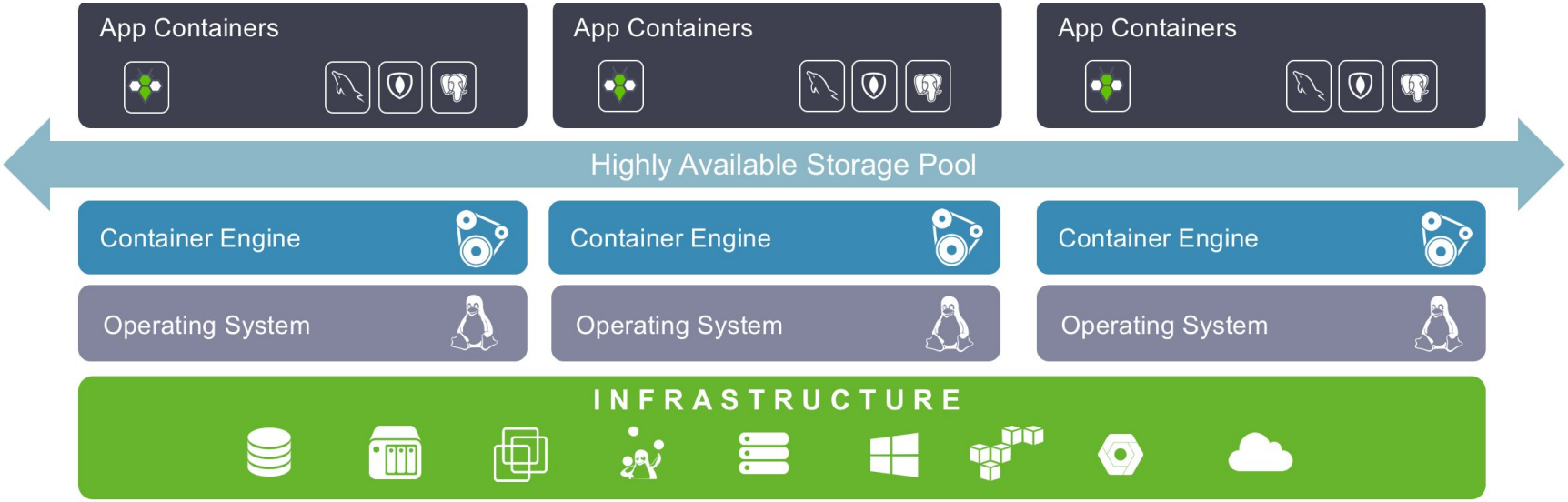
The pool's capacity should be about the three nodes' aggregated capacity.

Terminal

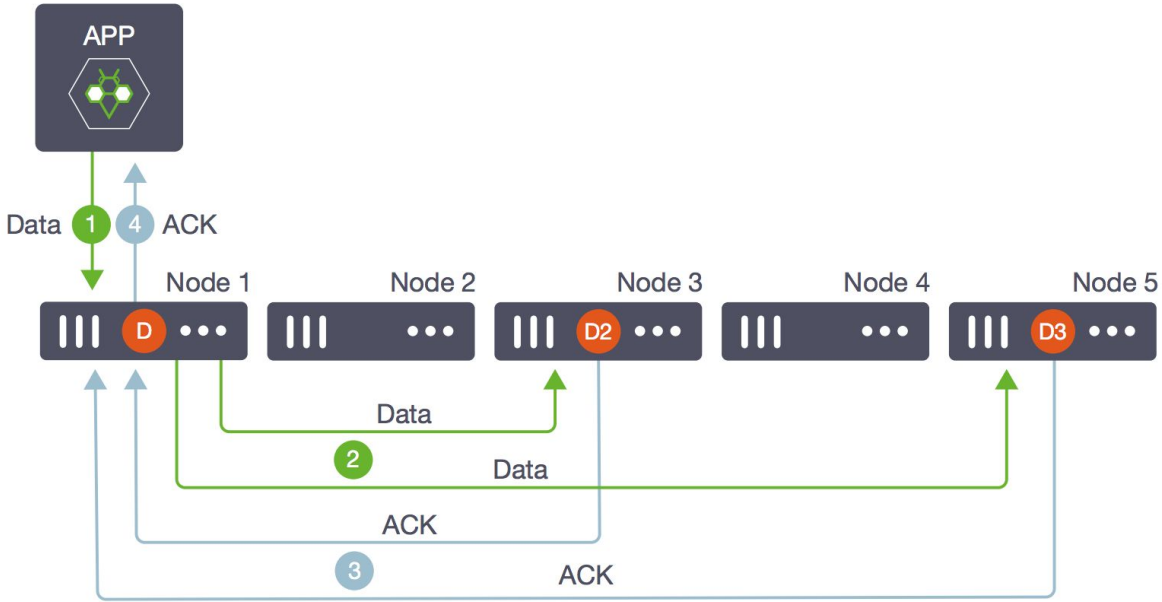
```
Your Interactive Bash Terminal. A safe place to learn and execute commands.

for i in {1..20}; do /opt/verify-healthy.sh && break || sleep 1; done

for i in {1..20}; do /opt/verify-healthy.sh && break || sleep 1; done
$
$ for i in {1..20}; do /opt/verify-healthy.sh && break || sleep 1; done
Waiting for StorageOS to initialise... this will take a minute.
.
```

Powered by Katacoda

Volume plugin: StorageOS



High availability with StorageOS



Volume plugin: StorageOS

8

Horizontally scalable
Consistent and performant
Platform agnostic
Synchronous replication
Volume is limited to the size of one node

Conclusion

Recap

Container storage is tricky... but critical!

Eight Principles for Cloud Native Storage

How StorageOS does persistent storage and high availability

K8S Storage SIG / CNCF Storage WG

Objective is to define an industry standard “Container Storage Interface” (CSI) that will enable storage vendors to develop a plugin once and have it work across a number of container orchestration systems.

Try StorageOS

my.storageos.com/main/tutorials

Quickstart

storageos.com/install

Meetup on Wednesday - Justin Cormack from Docker

meetup.com/Cloud-Native-London

STORAGEOS

Thanks

Slides at oicheryl.com