Eight principles for Cloud Native Storage

Cheryl Hung @oicheryl

Cheryl
@oicheryl

STORAGEOS

Why do I need storage?

Why do I need storage?

Why do I need storage?

App binaries

App data

Config

Backup

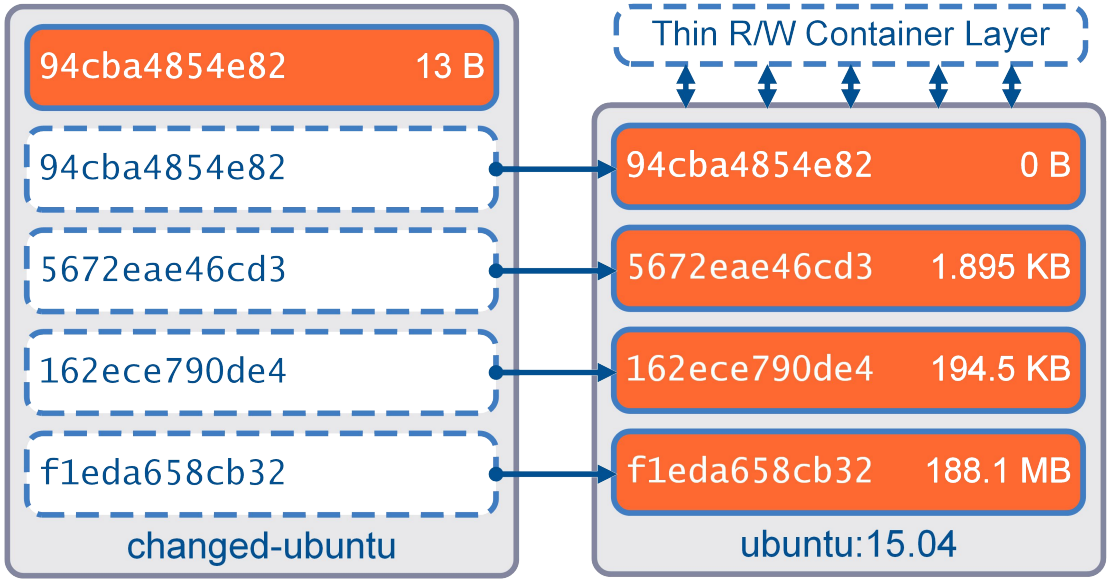
Why is this tricky with containers?

No storage pets

Data follows

Humans are fallible

Docker container layers

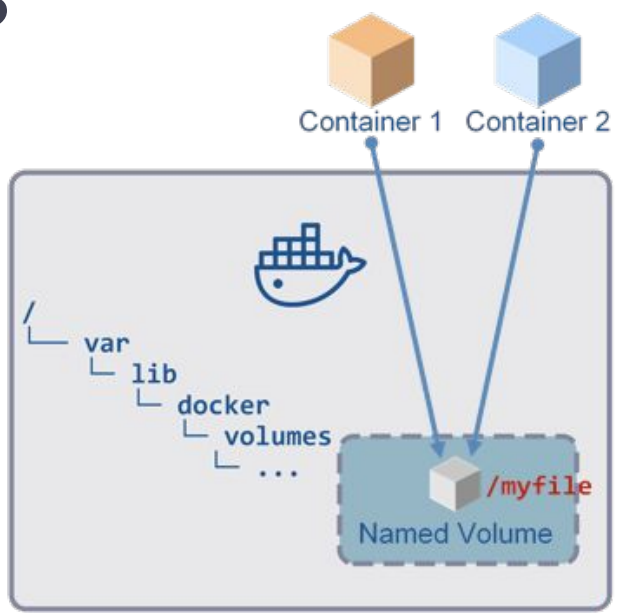


Docker local volumes

```
$ 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

Eight principles of Cloud Native Storage

1. **Platform agnostic**

Eight principles of Cloud Native Storage

1. Platform agnostic
2. **API driven**

Eight principles of Cloud Native Storage

1. Platform agnostic
2. API driven
3. **Declarative and composable**

Eight principles of Cloud Native Storage

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

Eight principles of Cloud Native Storage

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

Eight principles of Cloud Native Storage

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

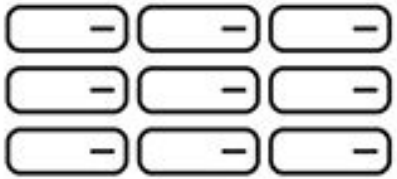
Eight principles of Cloud Native Storage

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

Eight principles of Cloud Native Storage

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

Storage landscape



Block storage

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

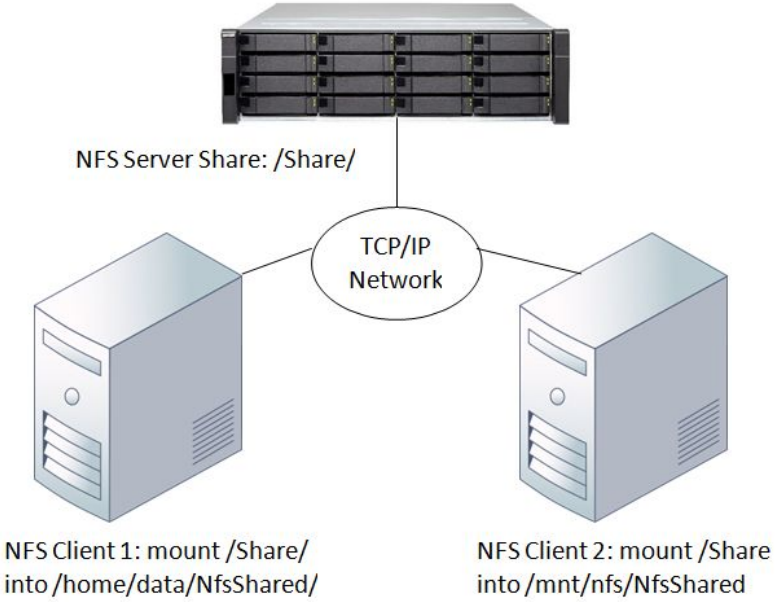
File storage

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

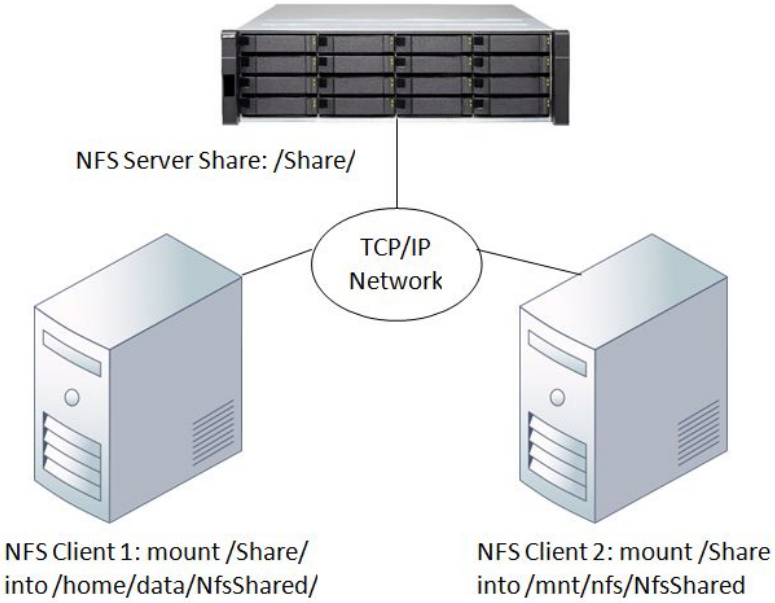
Object storage

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

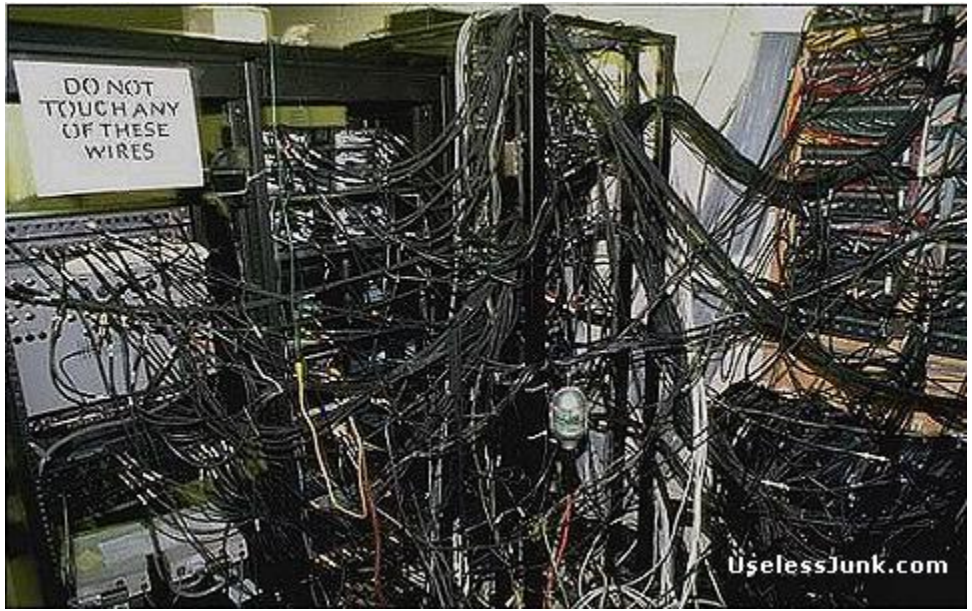
Centralised file system: NFS



Centralised file system: NFS



Storage array: Dell EMC



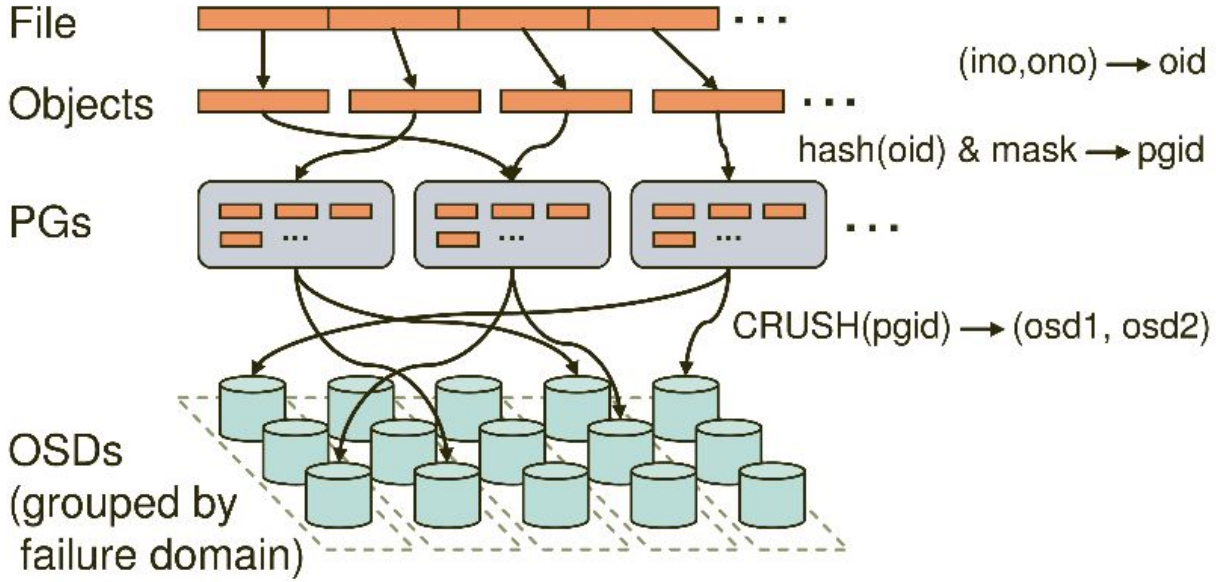
Storage array: Dell EMC

2

Deterministic performance

Vendor lock in

Distributed: Ceph

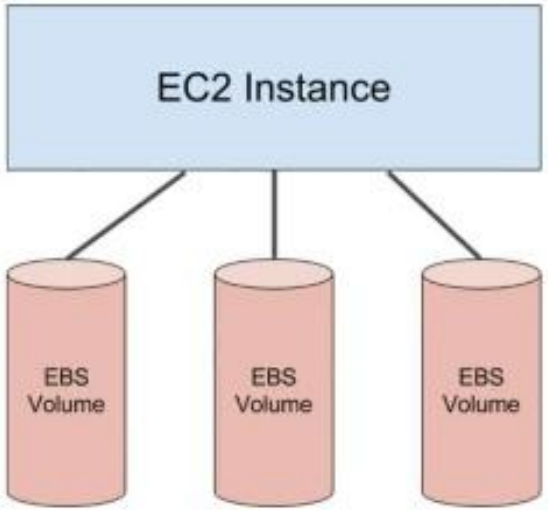


Distributed: Ceph

4

Horizontally scalable
Hardware agnostic
Complicated to set up (see Rook)
Failures are expensive

Public cloud: AWS EBS

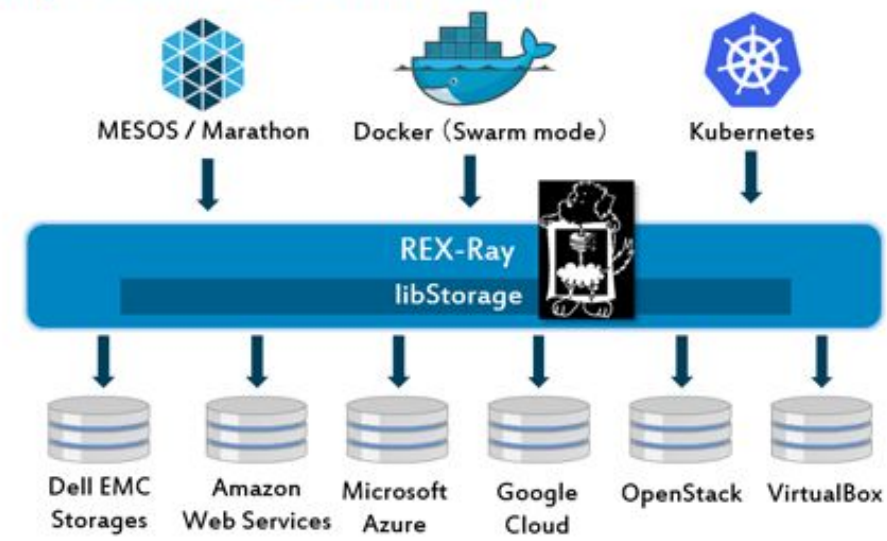


Public cloud: AWS EBS

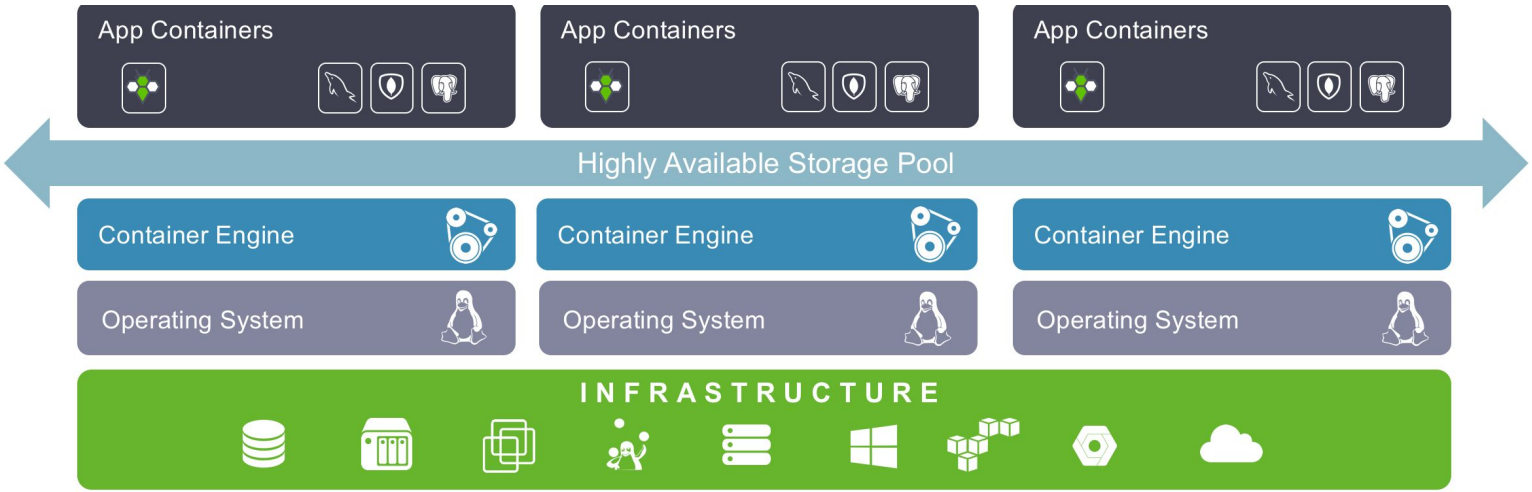
6

Horizontally scalable
Consistent and performant
Vendor lock in
Mounting physical block devices
Expensive and privacy issues

Plugin framework: REX-Ray

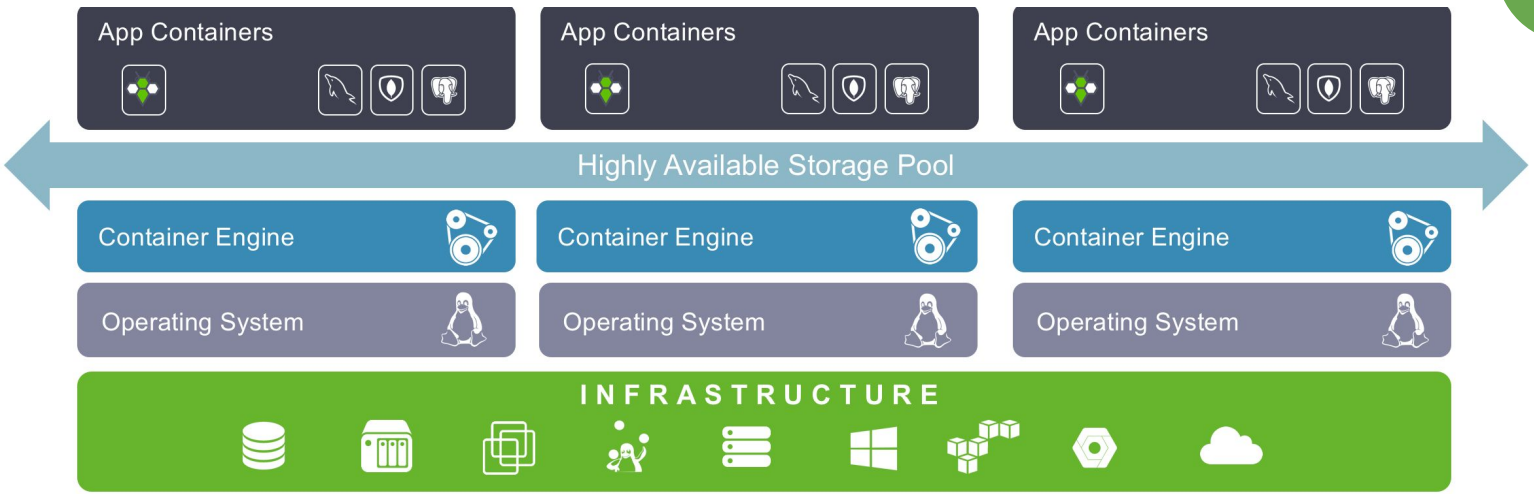


Volume plugin: StorageOS



Volume plugin: StorageOS

8



Volume plugin: StorageOS

Conclusion

K8S Storage SIG & CNCF Storage WG: https://github.com/cncf/wg-storage

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

Thanks

Slides at oicheryl.com