Container Storage Interface (CSI) for K8s

Storage types

  - DAS
  - SAN
  - NAS
- NFS
- GlusterFS
- Ceph
- CSI

Cloud Storages (Persistence)

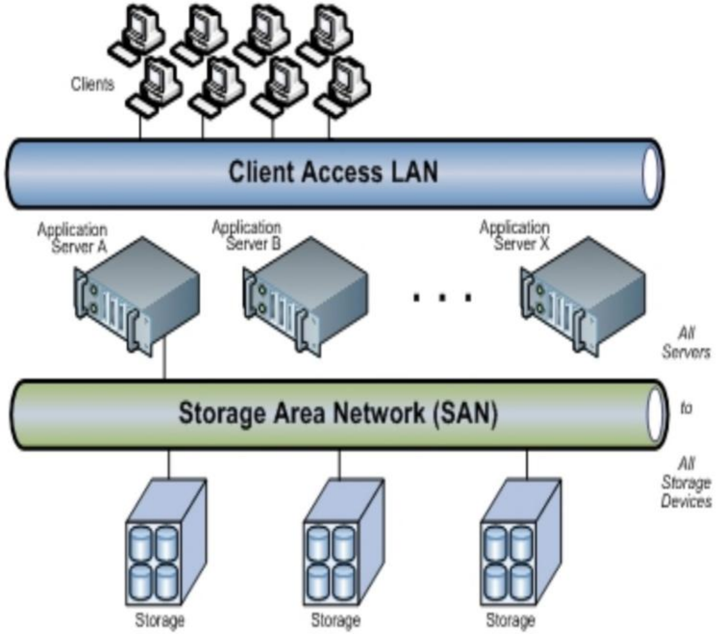
Direct attached storage (DAS)

- Directly attached (no network)
- Internal or External
- Disks - HDD, SSD
- RAID arrays
- Interface – IDE, SCSI, SATA, SAS

Storages

Storage Area Network (SAN)

- High speed network access
  - Fiber Chanel – FC – 128Gbps very expensive
  - iSCSI – slightly cheaper – not fast as FC
- composed of hosts, switches, storage elements
- Block level storage
- variety of technologies, topologies, and protocols
- Scalable
- Highly redundant
- Improve application availability
  - Replicate
  - Use RAID
- Enhance application performance



Storages

Network Attached Storage (NAS)

Principals

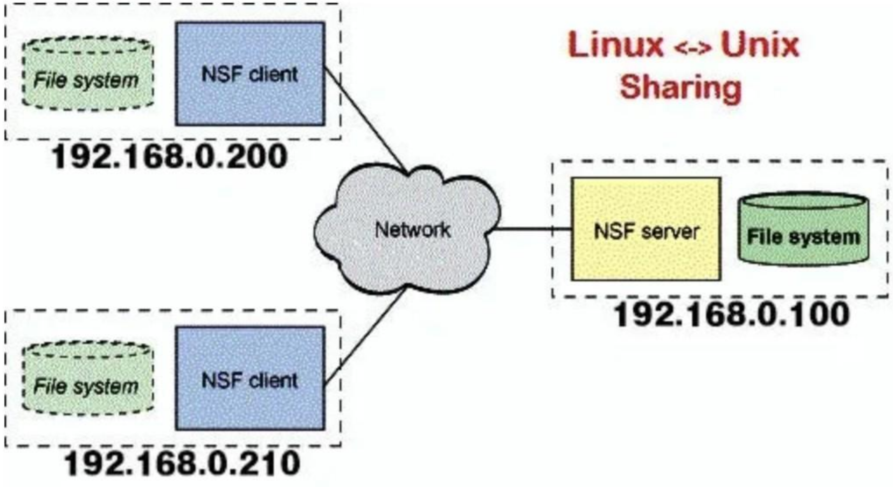
- File storage
  - Files, hierarchical directories/folders

- Block storage
  - Chucks or objects – break file into small chucks
  - Unique address for each chunks - logical block addressing (LBA)
  - direct access to individual data blocks
  - fast
- Object storage

  Discrete unit of data, without a hierarchy (unstructured data)
  Each object includes – data and metadata (descriptive info about data)

NFS

- Network File System (NFS)
  - Simplest shared storage
  - RAID level redundancy Over Ethernet
  - Shared directory
  - Single point of failure
  - Not expensive



NFS

- NFS Server
  - *apt install nfs-kernel-server*
    - *mkdir /var/nfs/general -p*
    - *chown nobody:nogroup /var/nfs/general*
    - */etc/exports*
      - *directory_to_share   client(share_option1,…,share_optionN)*
        - */var/nfs/general   client_ip(rw,sync,no_subtree_check)*
    - *systemctl restart nfs-kernel-serve*

- NFS Client
  - *apt install nfs-common*
  - mkdir -p /nfs/general
  - /etc/fstab
    - host_ip:/var/nfs/general   /nfs/general   nfs auto,nofail,noatime,nolock,intr,
      tcp,actimeo=1800 0 0

Gluster FS

- What is GlusterFS
    - Open source software define storage
    - POSIX-Compliant Distributed File System *Portable Operating System Interface*
    - No metadata server
    - NAS
    - Heterogeneous commodity hardware
    - Segregated storage and memory
    - Simple and inexpensive
    - High redundancy
    - Flexible and agile scaling
    - Capacity – petabytes and beyond
    - Performance – thousands of clients

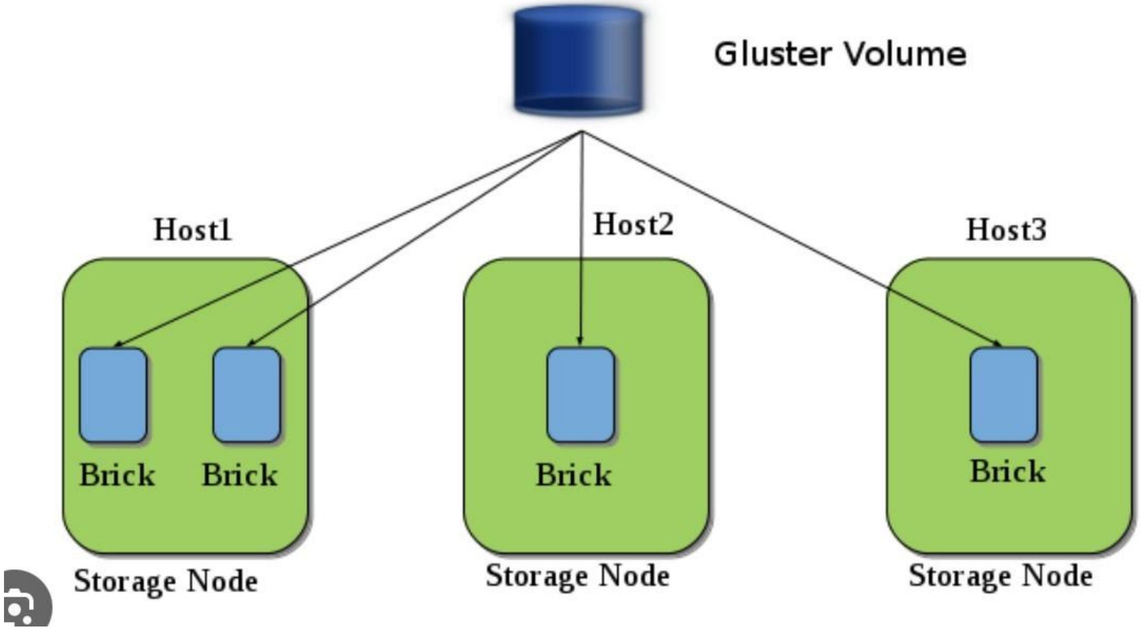
Gluster FS

- Technical requirements
  - Direct-attached storage (DAS)
  - Just a Buch of Disks (JBOD)
  - Hardware RAID
    - RAID 6 required
  - Logical Volume Management (LVM)
  - XFS, EXT3/4, BTRFS
  - Extended attributes support required
  - RHS: XFS required

Gluster FS

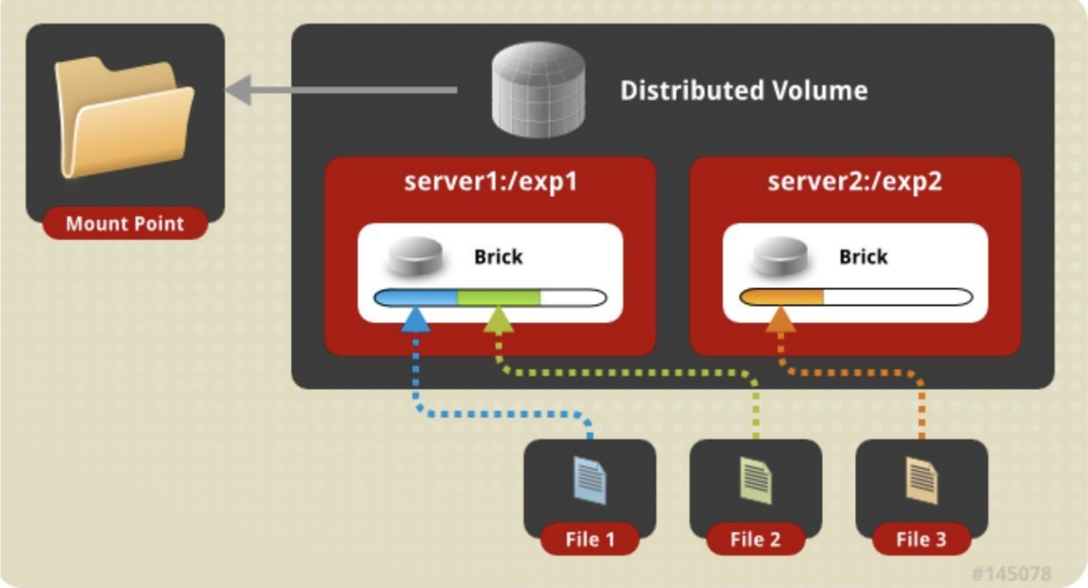
- Bricks
  - basic unit of storage
  - a mount point
  - export directory

- Volume
  - a logical collection of bricks



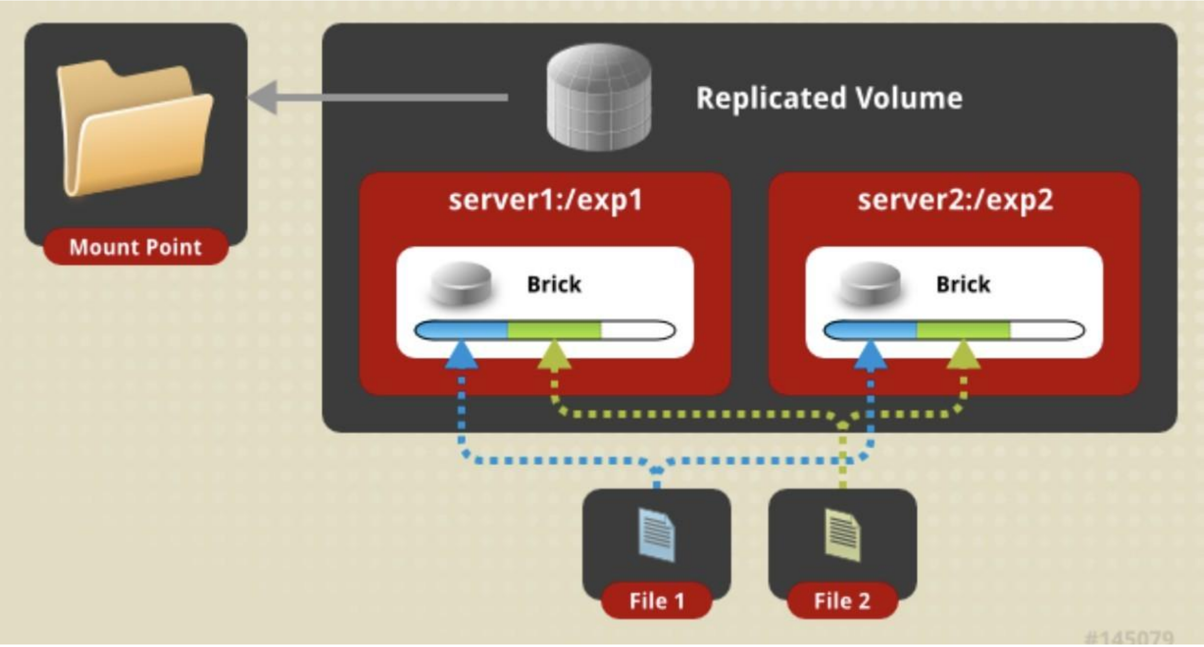
Gluster FS

- Volume types
  - Distributed volume



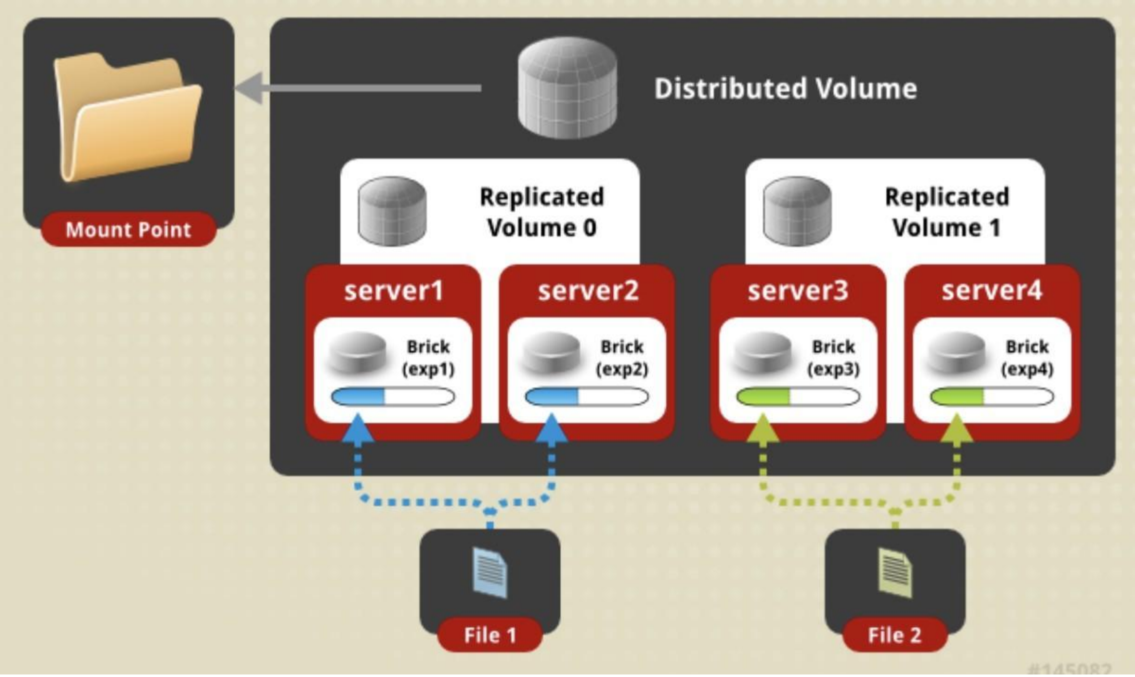
Gluster FS

- Volume types
  - Replicated volume



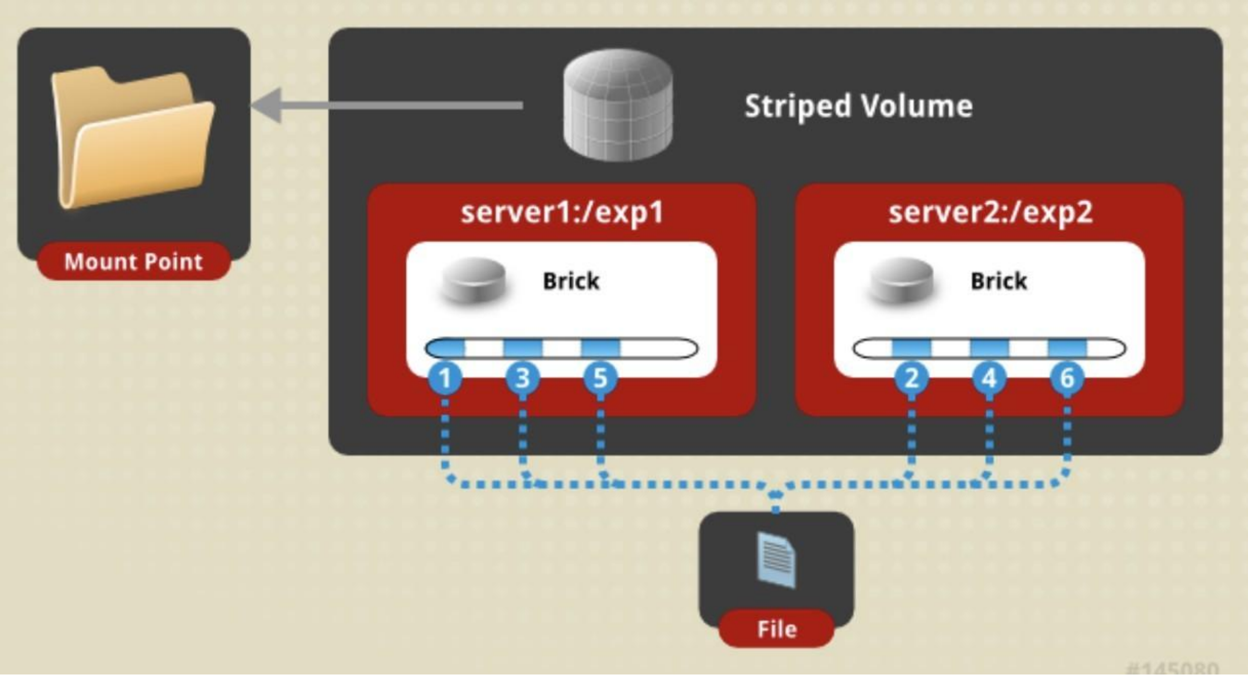
Gluster FS

- Volume types
  - Distributed replicated volume



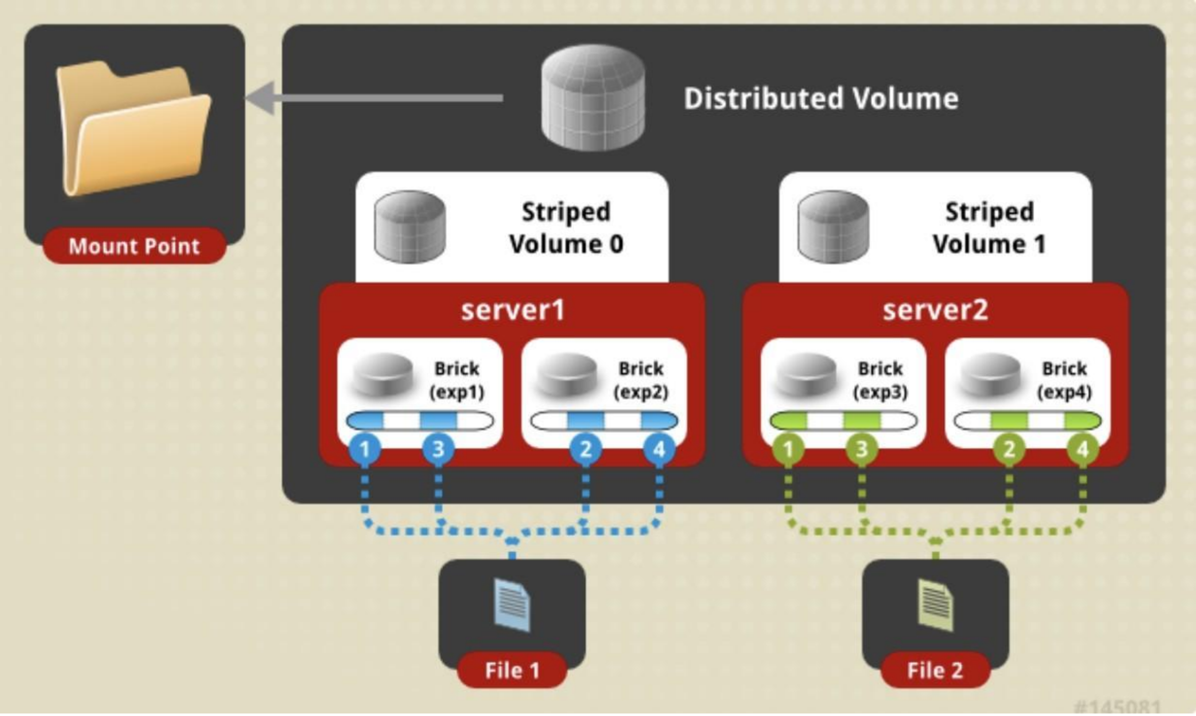
Gluster FS

- Volume types
  - Striped volume



Gluster FS

- Volume types
  - Distributed Striped volume



CEPH

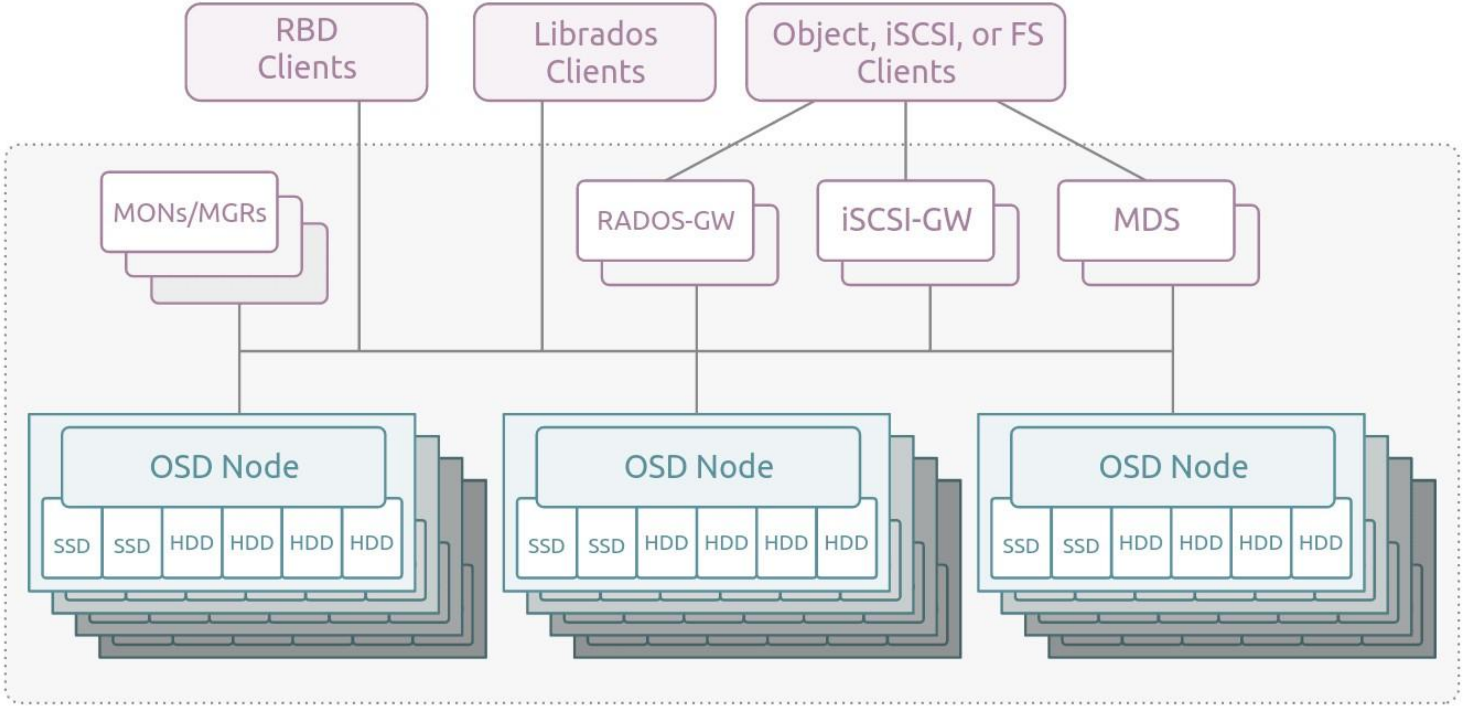
- What is Ceph
  - Open source software define storage
  - Provides
    - Object storage
    - Block storage
    - File system
  - commodity hardware
  - High redundancy
  - Scalable
  - Not simple as Glusterfs

CEPH

- Ceph consists of
  - Monitors (ceph-mon)
    - Cluster state, active and failed nodes, cluster configuration, data placement, manage authentication
  - Managers (ceph-mgr)
    - Maintain cluster runtime metrics, enable dash-boarding
  - Object storage devices (ceph-osd)
    - Store data, replication, recovery, rebalancing
  - Rados gateways (ceph-rgw)
    - Object storage APIs via http/https
  - Metadata servers (ceph-mds)
    - store metadata for ceph FS, mapping filenames and directories

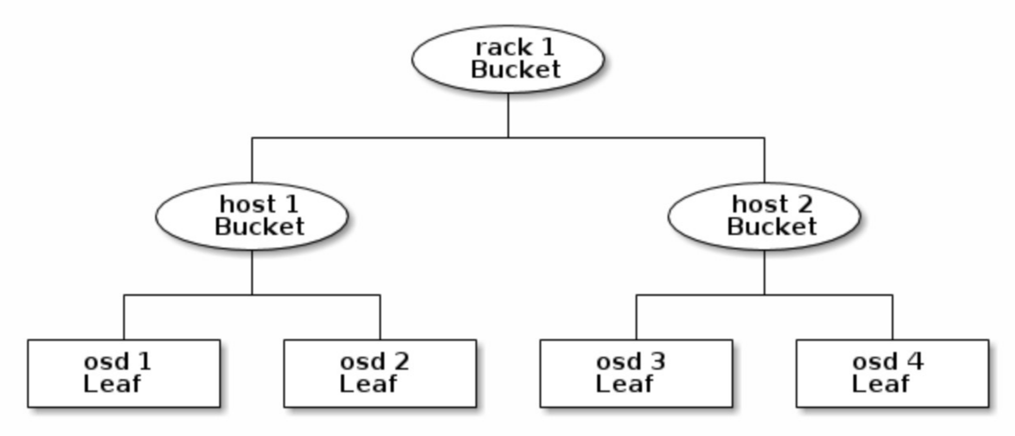
Reliable Autonomic Distributed Object Store

CEPH Cluster



CEPH

- Crush Map
  - Device locations with the hierarchy
  - Ruleset to store data
  - Nodes (buckets)
  - Leaves
  - Helps ceph clients to communicate OSDs directly
  - Helps OSDs to replicate, backfill and recover



CEPH

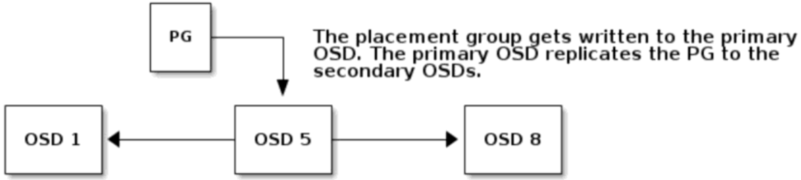
- Crush Algorithm
  - computes storage locations in order to determine how to store and retrieve data
  - allows Ceph clients to communicate with OSDs directly rather than through a centralized server or broker

    Avoids

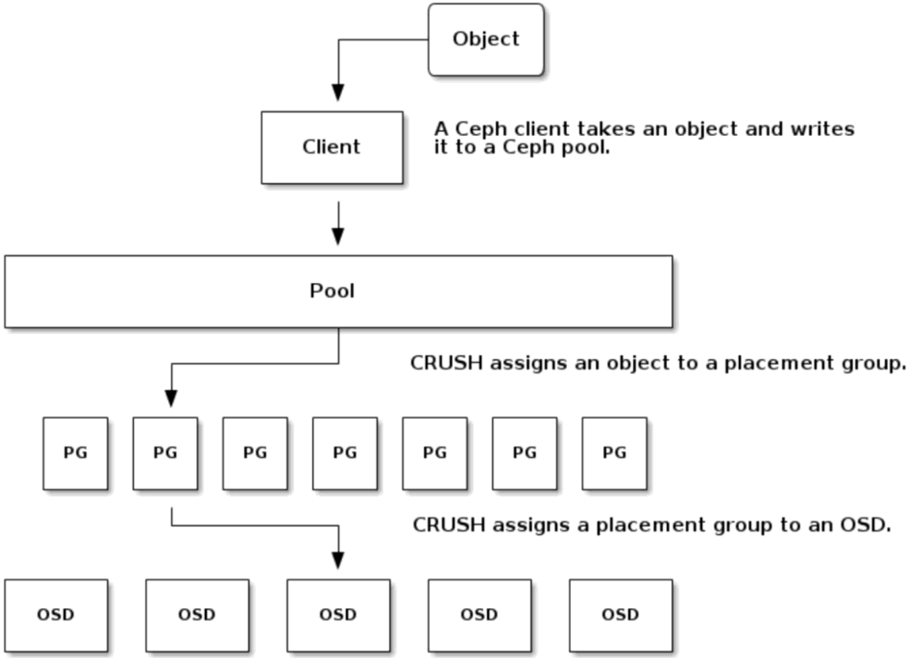
    - a single point of failure
    - a performance bottleneck
    - physical limit to its scalability

CEPH

- Placement Groups
  - Object to placement group
  - Placement group to OSD
  - Many PGs to each OSD
  - Evenly distributed



The placement group gets written to the primary OSD. The primary OSD replicates the PG to the secondary OSDs.



A Ceph client takes an object and writes it to a Ceph pool.

CRUSH assigns an object to a placement group.

CRUSH assigns a placement group to an OSD.

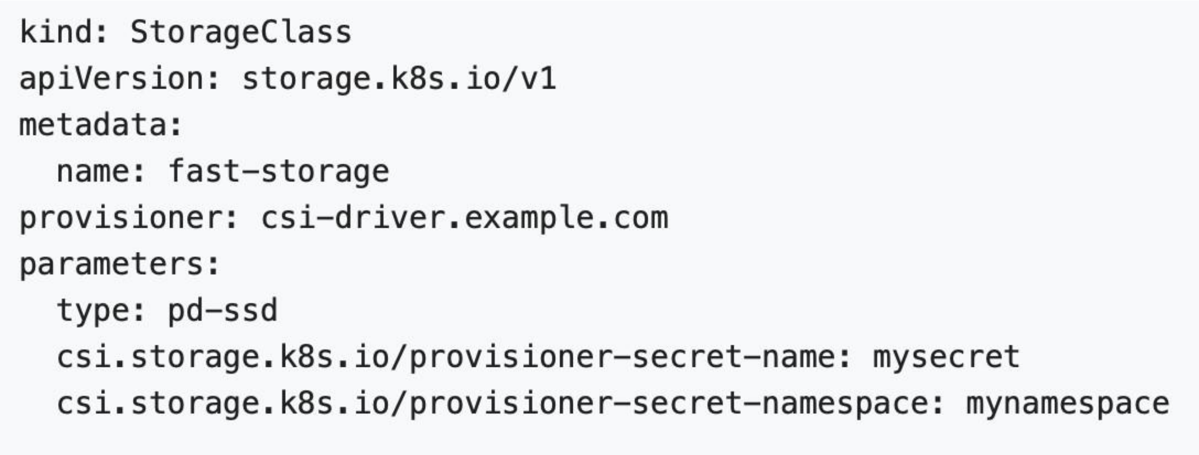
CSI - Container Storage Interface

- Standard for exposing arbitrary block and file storage systems to containerized workloads on K8s
- Third-party storage providers can write and deploy plugins exposing new storage systems
  - Many storage options of K8s users
  - Be more secure and reliable

CSI - Container Storage Interface

- How to deploy CSI driver?
  - Author provides way to do

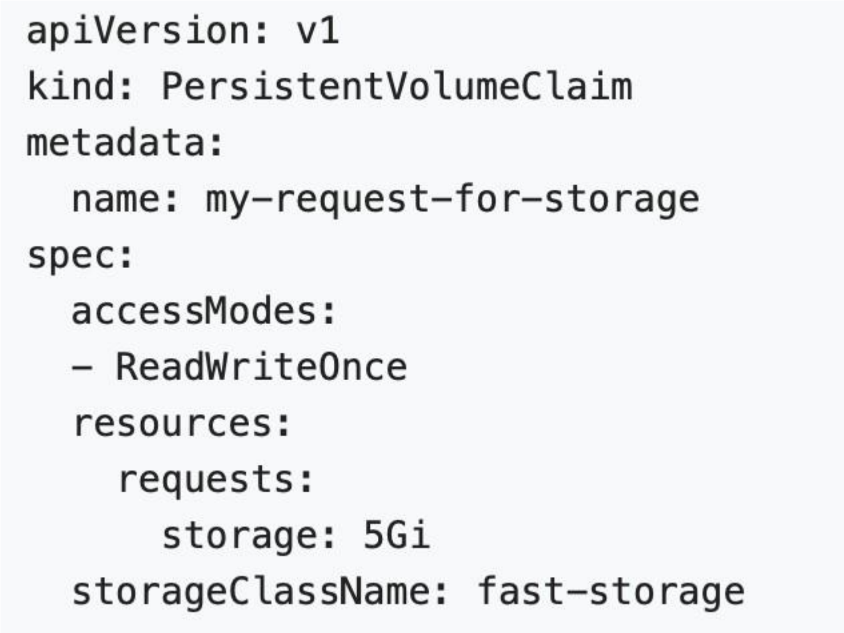
  How to use volume?
  - Storage class



```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: fast-storage
provisioner: csi-driver.example.com
parameters:
  type: pd-ssd
  csi.storage.k8s.io/provisioner-secret-name: mysecret
  csi.storage.k8s.io/provisioner-secret-namespace: mynamespace
```

CSI - Container Storage Interface

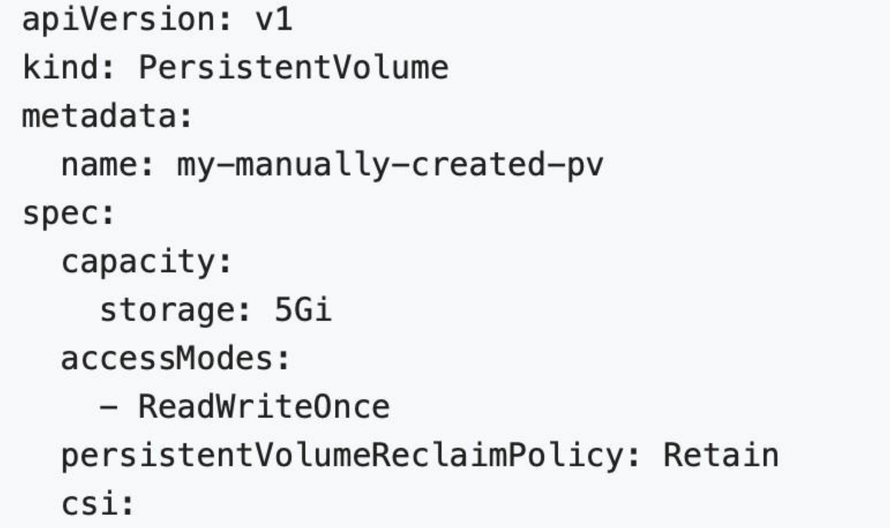
- How to use volume?
  - Volume claim



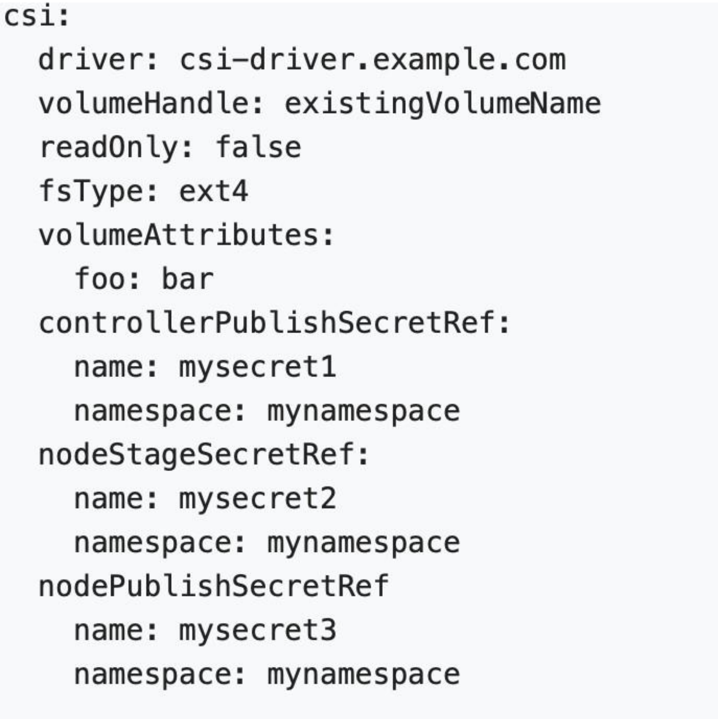
```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-request-for-storage
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
  storageClassName: fast-storage
```

CSI - Container Storage Interface

- How to use volume?
  - PersistentVolume



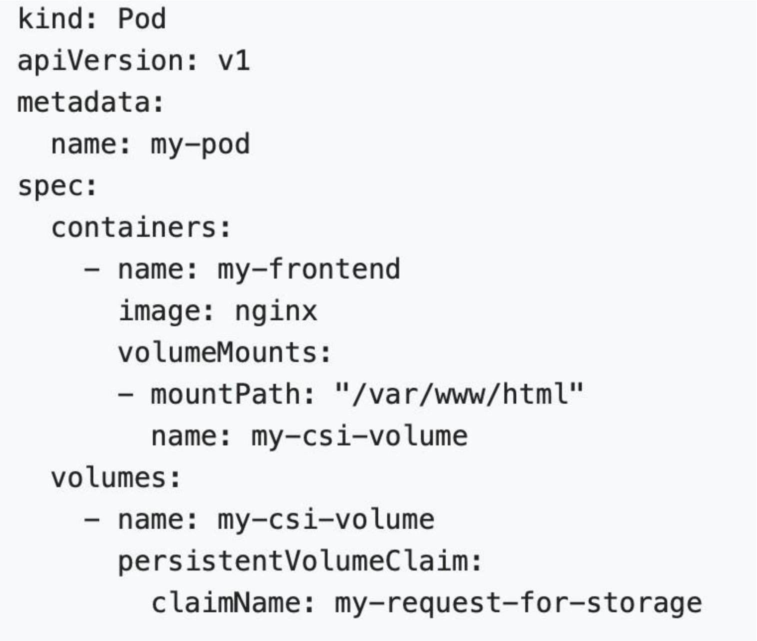
```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: my-manually-created-pv
spec:
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  csi:
```



```yaml
csi:
  driver: csi-driver.example.com
  volumeHandle: existingVolumeName
  readOnly: false
  fsType: ext4
  volumeAttributes:
    foo: bar
  controllerPublishSecretRef:
    name: mysecret1
    namespace: mynamespace
  nodeStageSecretRef:
    name: mysecret2
    namespace: mynamespace
  nodePublishSecretRef
    name: mysecret3
    namespace: mynamespace
```

CSI - Container Storage Interface

- How to use volume?
  - Volume mount



```
kind: Pod
apiVersion: v1
metadata:
  name: my-pod
spec:
  containers:
    - name: my-frontend
      image: nginx
      volumeMounts:
      - mountPath: "/var/www/html"
        name: my-csi-volume
  volumes:
    - name: my-csi-volume
      persistentVolumeClaim:
        claimName: my-request-for-storage
```