OpenSDS for CSI - Achieve full potential of stateful application resiliency

Wu Jiajun, Huawei Technologies CO., Ltd

Agenda

- K8s Storage Overview
- CSI Overview
  - Why CSI
  - What is CSI
  - How to implement CSI
- OpenSDS Overview
- Demo

K8s Storage Overview - Stateful Apps

- **SQL Databases**

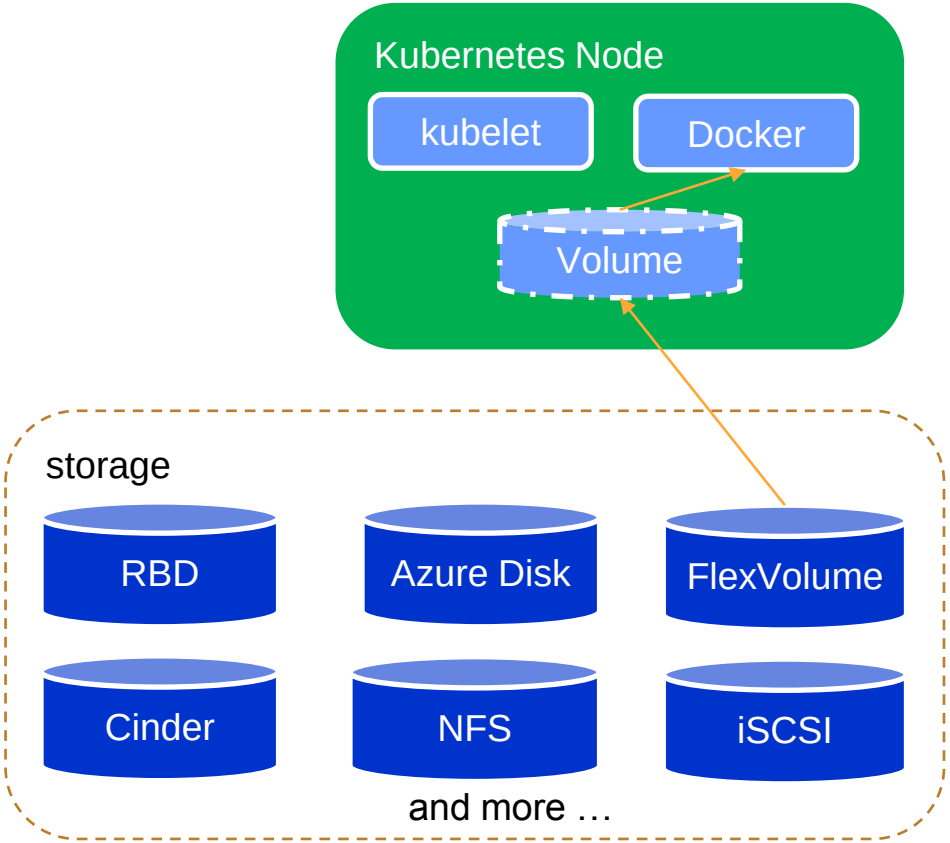

- **NoSQL Databases**


ElasticSearch


- **Time series Databases**

K8s Storage Overview - Volume Plugins

- Reference block device and mounted file system
- Accessible by all containers in a pod
- Lifetime of a volume is same as the pod (or longer)



K8s Storage Overview - Volume Plugins

- awsElasticBlockStore
- azureDisk
- azureFile
- cephfs
- configMap
- csi
- downwardAPI
- emptyDir
- fc (fibre channel)
- flocker
- gcePersistentDisk
- gitRepo
- glusterfs
- hostPath
- iscsi
- local
- nfs
- persistentVolumeClaim
- projected
- portworxVolume
- quobyte
- rbd
- scaleIO
- secret
- storageos
- vsphereVolume

K8s Storage Overview- Volume Plugins Example

- iSCSI volume in a Pod

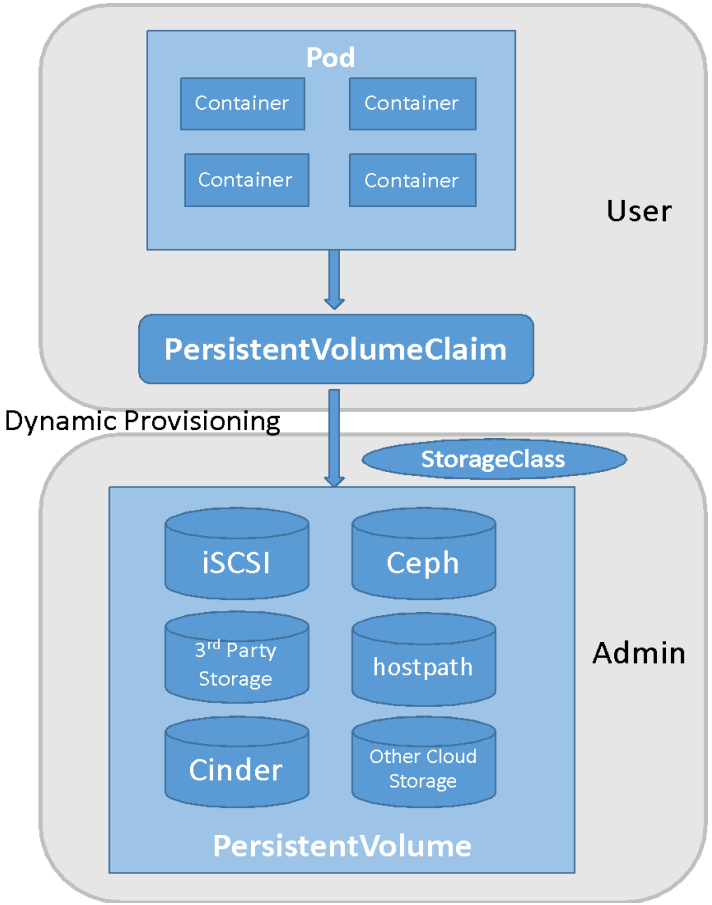
```yaml
---
apiVersion: v1
kind: Pod
metadata:
  name: iscsipd
spec:
  containers:
  - name: iscsipd-rw
    image: kubernetes/pause
    volumeMounts:
    - mountPath: "/mnt/iscsipd"
      name: iscsipd-rw
  volumes:
  - name: iscsipd-rw
    iscsi:
      targetPortal: 10.0.2.15:3260
      portals: ['10.0.2.16:3260', '10.0.2.17:3260']
      iqn: iqn.2001-04.com.example:storage.kube.sys1.xyz
      lun: 0
      fsType: ext4
      readOnly: true
```

- cinder volume in a Pod

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: cinder-web
spec:
  containers:
    - name: web
      image: nginx
      ports:
        - name: web
          containerPort: 80
          protocol: tcp
      volumeMounts:
        - name: html-volume
          mountPath: "/usr/share/nginx/html"
  volumes:
    - name: html-volume
      cinder:
        # Enter the volume ID below
        volumeID: volume_ID
        fsType: ext4
```

K8s Storage Overview – PV/PVC/SC

- A **PersistentVolume** (PV) is a piece of storage in the cluster that has been provisioned by an administrator.
- A **PersistentVolumeClaim** (PVC) is a request for storage by a user through a StorageClass.
- A **StorageClass** provides a way for administrators to describe the “classes” of storage they offer. Different classes might map to different quality-of-service levels (or ”profiles”) in other storage systems.
- A StorageClass needs to specify a provisioner for dynamic provisioning.



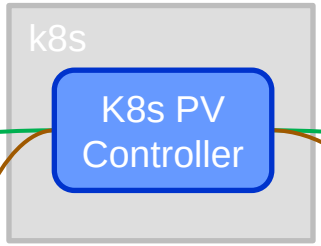
K8s Storage Overview – Persistent Volume types

- GCEPersistentDisk
- AWSElasticBlockStore
- AzureFile
- AzureDisk
- FC (Fibre Channel)
- FlexVolume
- Flocker
- NFS
- iSCSI
- RBD (Ceph Block Device)
- CephFS
- Cinder (OpenStack block storage)
- Glusterfs
- VsphereVolume
- Quobyte Volumes
- HostPath (Single node testing only – local storage is not supported in any way and WILL NOT WORK in a multi-node cluster)
- Portworx Volumes
- ScaleIO Volumes
- StorageOS

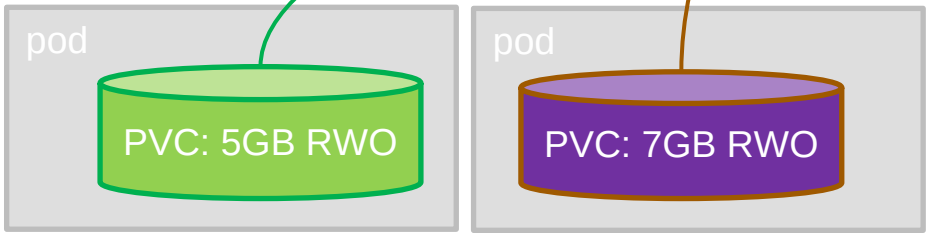
Kubernetes Static Provisioning

User / Developer

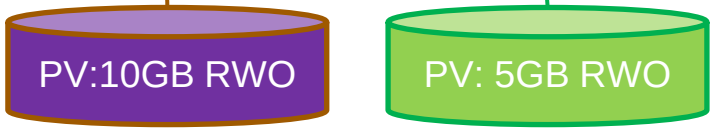
Cluster / IT Admin



storage



PersistentVolumeClaim



PersistentVolume

Kubernetes Static Provisioning

```
ubuntu@k8s-master:~$
ubuntu@k8s-master:~$ kubectl apply -f mysql-pv.yml
persistentvolume "mysql-pv" created
ubuntu@k8s-master:~$
ubuntu@k8s-master:~$ kubectl apply -f mysql-pvc.yml
persistentvolumeclaim "mysql-pvc" created
ubuntu@k8s-master:~$
ubuntu@k8s-master:~$ kubectl get pv,pvc
NAME          CAPACITY   ACCESSMODES   RECLAIMPOLICY   STATUS    CLAIM               STORAGECLASS   REASON    AGE
pv/mysql-pv   1Gi        RWO           Retain          Bound     default/mysql-pvc   nfs                      14s

NAME            STATUS    VOLUME     CAPACITY   ACCESSMODES   STORAGECLASS   AGE
pvc/mysql-pvc   Bound     mysql-pv   1Gi        RWO           nfs            9s
ubuntu@k8s-master:~$
```

K8s Storage Overview – Static Provisioning Example

Pod

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: pod-with-block-volume
spec:
  containers:
    - name: fc-container
      image: fedora:26
      command: ["/bin/sh", "-c"]
      args: [ "tail -f /dev/null" ]
      volumeDevices:
        - name: data
          devicePath: /dev/xvda
  volumes:
    - name: data
      persistentVolumeClaim:
        claimName: block-pvc
```

PVC

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: block-pvc
spec:
  accessModes:
    - ReadWriteOnce
  volumeMode: Block
  resources:
    requests:
      storage: 10Gi
```

PV

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: block-pv
spec:
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteOnce
  volumeMode: Block
  persistentVolumeReclaimPolicy: Retain
  fc:
    targetWWNs: ["50060e801049cfd1"]
    lun: 0
    readOnly: false
```

Kubernetes Dynamic Provisioning

User / Developer

Cluster / IT Admin

OpenSDS

k8s

K8s PV Controller

storage

pod

PVC：1GB RWO
High Performance

OpenSDS
Provisioner

PV: 1GB RWO
High Performance

PersistentVolumeClaim

PersistentVolume

K8s Storage Overview – Dynamic Provisioning E

CHINA OpenInfra Days
IT大咖说 知识共享平台

Pod

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: busy-pod
spec:
  containers:
    - image: busybox
      name: busy-container1
      command: ["/bin/sleep","7d"]
      volumeMounts:
        - name: vol1
          mountPath: /mnt/pv-1
  volumes:
    - name: vol1
      persistentVolumeClaim:
        claimName: "opensds-pvc"
```

StroageClass

```yaml
apiVersion: storage.k8s.io/v1beta1
kind: StorageClass
metadata:
  name: opensds
provisioner: opensds/nbp-provisioner
parameters:
  kubernetes.io/availabilityZone: default
  kubernetes.io/type: ext4
```

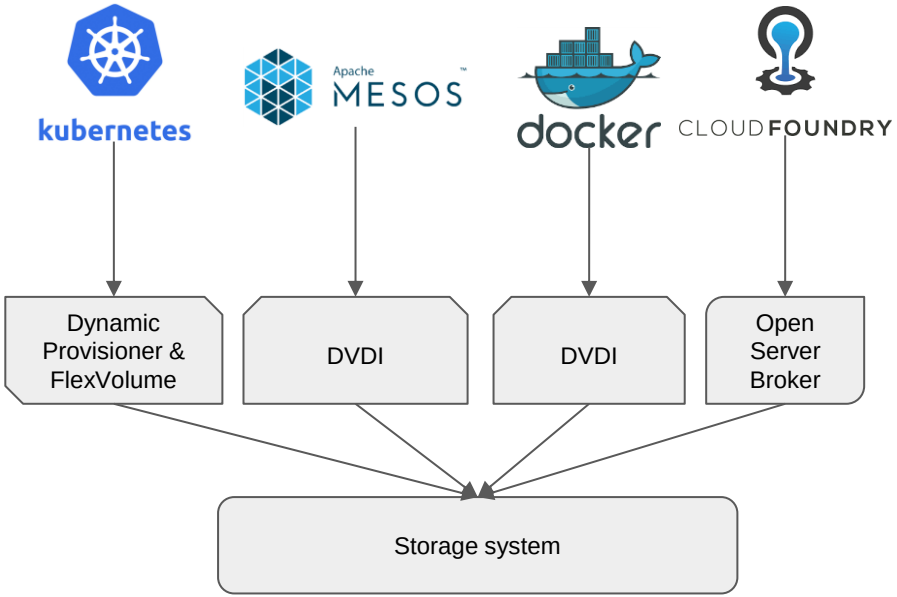
PVC

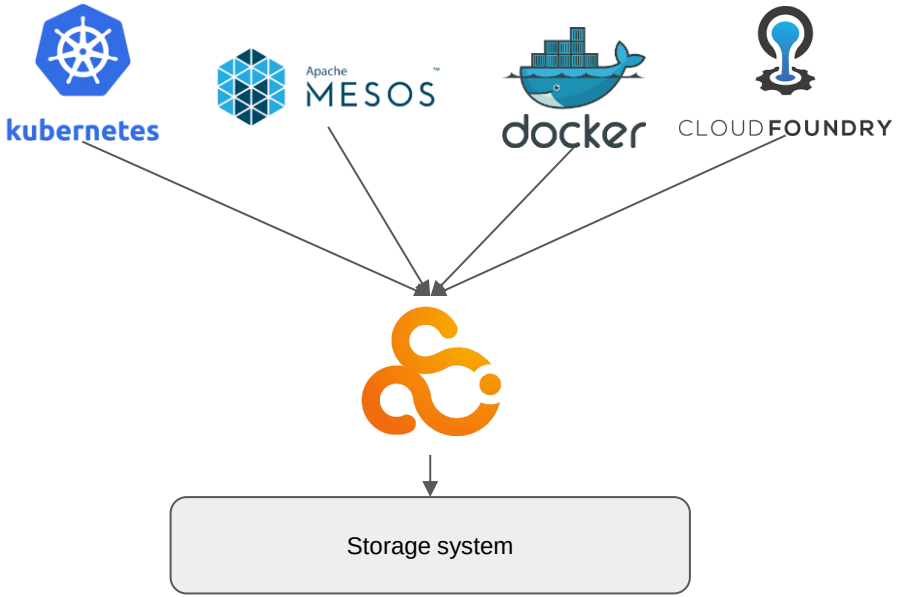
```yaml
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: opensds-pvc
  namespace: default
  annotations:
    volume.beta.kubernetes.io/storage-class: "opensds"
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

What the problem?

- Volume plugin development is tightly coupled and dependent on Kubernetes **releases**.
- Kubernetes developers/community are responsible for **testing** and **maintaining** all volume plugins, instead of just testing and maintaining a stable plugin API.
- **Bugs** in volume plugins can crash critical Kubernetes components, instead of just the plugin.
- Volume plugins get full **privileges** of kubernetes components (kubelet and kube-controller-manager).
- Plugin developers are forced to make plugin **source code** available, and can not choose to release just a binary.

CSI Overview - Why CSI?





- Maintaining all volume plugins is awesome
- SP need to implement several volume plugins for each CO
- High cost on learning different container storage standards

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.

What is CSI?

- Full Name

  Container Storage Interface

- Objectives

  Enable storage vendors (SP) to develop a plugin once and have it work across a number of container orchestration (CO) systems

CSI Overview

Controller service

- ControllerGetCapabilities
- **CreateVolume**
- **DeleteVolume**
- **ControllerPublishVolume**
- **ControllerUnpublishVolume**
- ListVolumes
- GetCapacity
- ValidateVolumeCapabilities

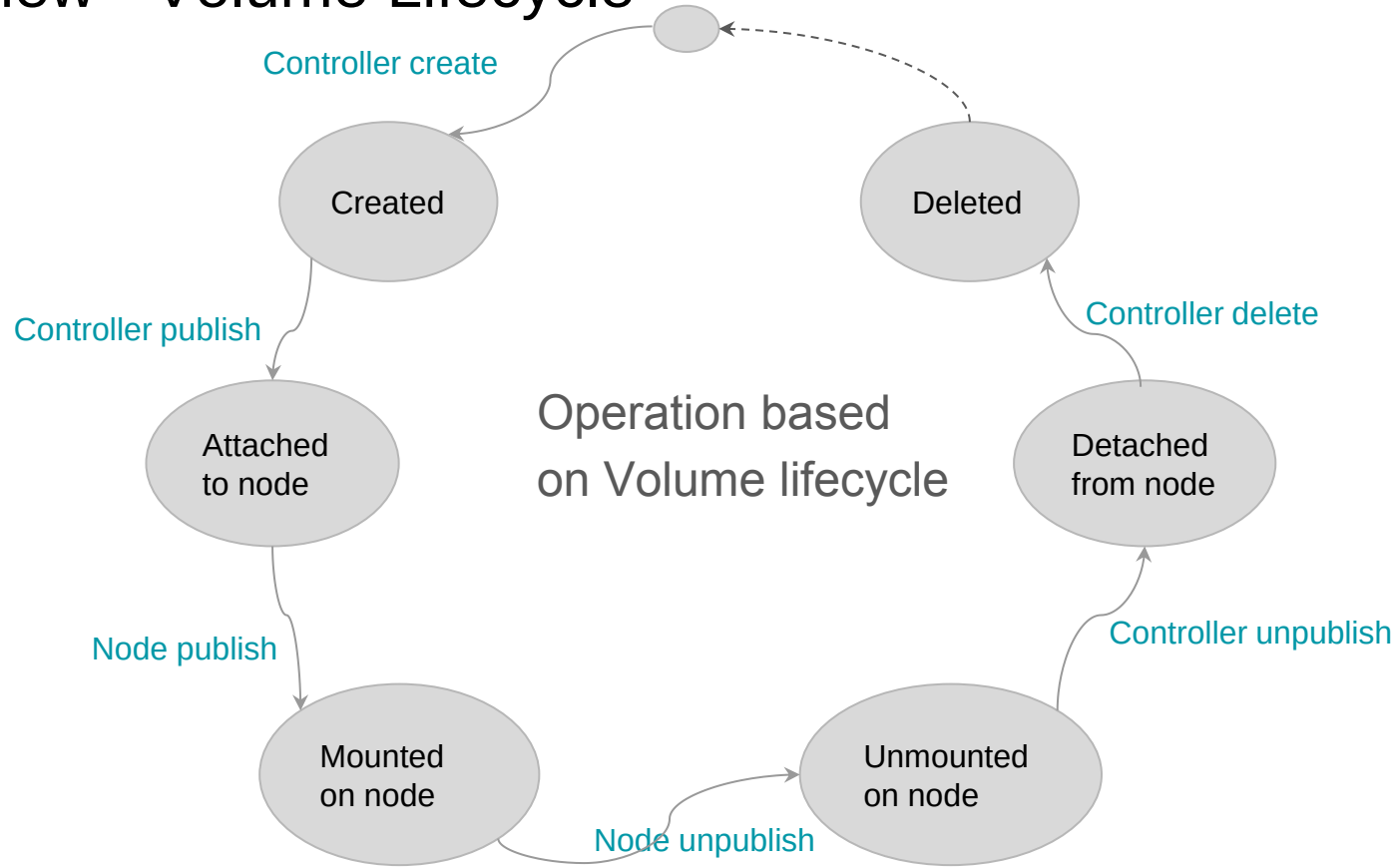
Node service

- NodeGetCapabilities
- **NodePublishVolume**
- **NodeUnpublishVolume**
- NodeStageVolume
- NodeUnstageVolume
- NodeGetId

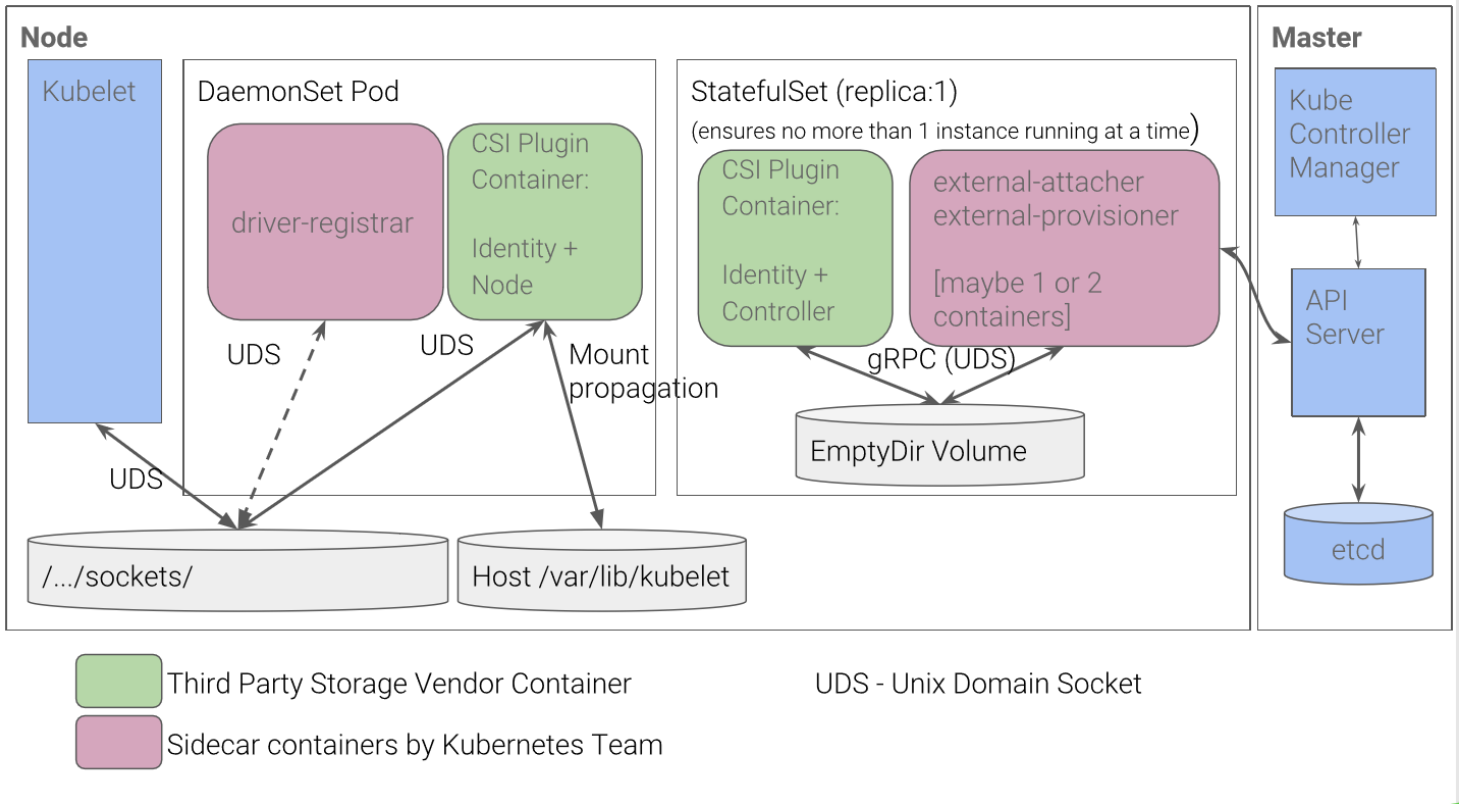
Identity service

- GetPluginCapabilities
- GetPluginInfo
- Probe

CSI Overview - Volume Lifecycle



Recommended Deployment



Source: https://github.com/kubernetes/community/blob/master/contributors/design-proposals/storage/container-storage-interface.md

Common external controller and node plugins implemented by k8s team

- External-Provisioner

  https://github.com/kubernetes-csi/external-provisioner

- External-attacher

  https://github.com/kubernetes-csi/external-attacher

- Driver-register

  https://github.com/kubernetes-csi/driver-registrar

OpenSDS Overview - Core Projects

SUSHI

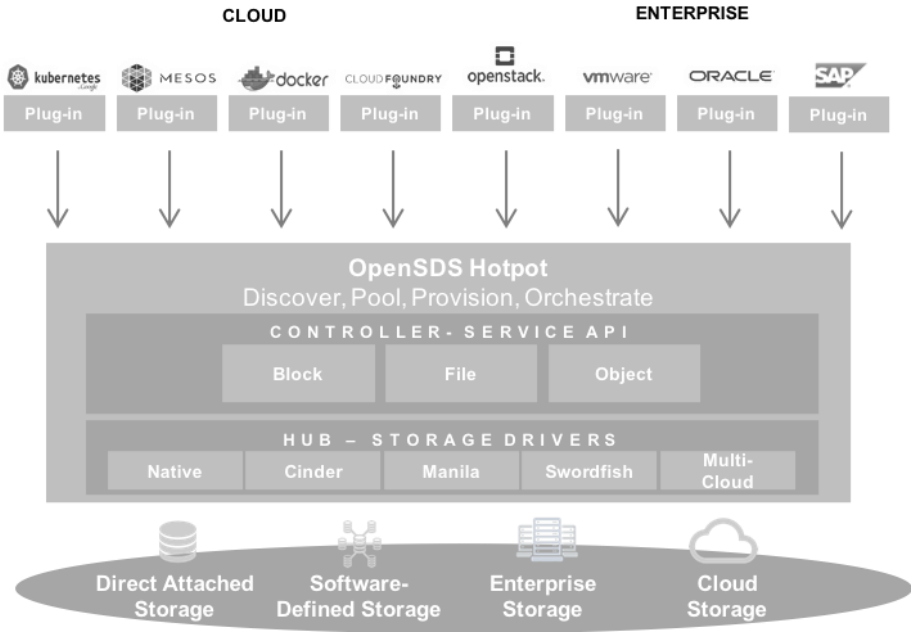
The Northbound Plug-ins Project

Common plug-ins to enable OpenSDS storage services for cloud and application frameworks

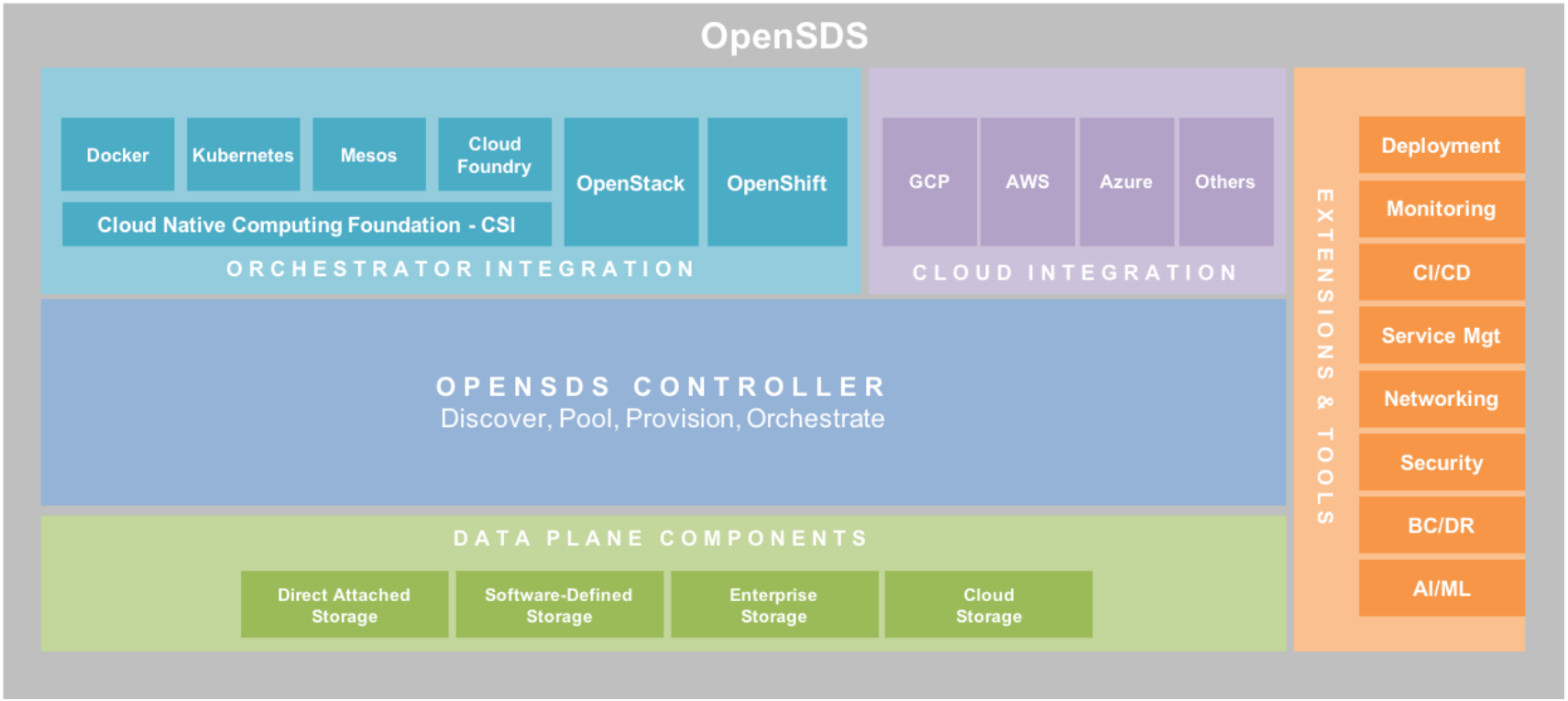
HOTPOT

The Storage Controller Project

Single control for block, file, and object services across storage on premise and in clouds

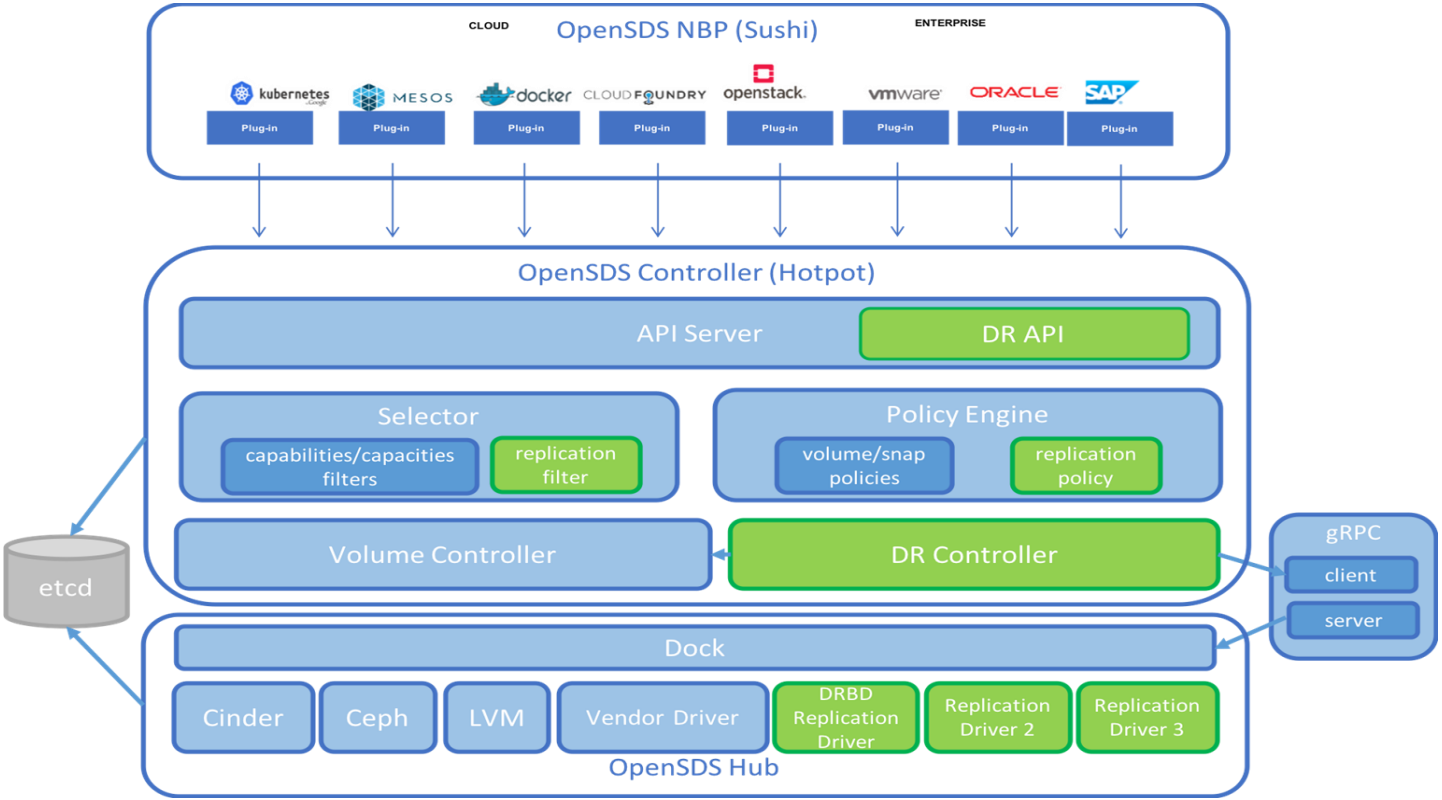


OpenSDS Overview - Project Framework

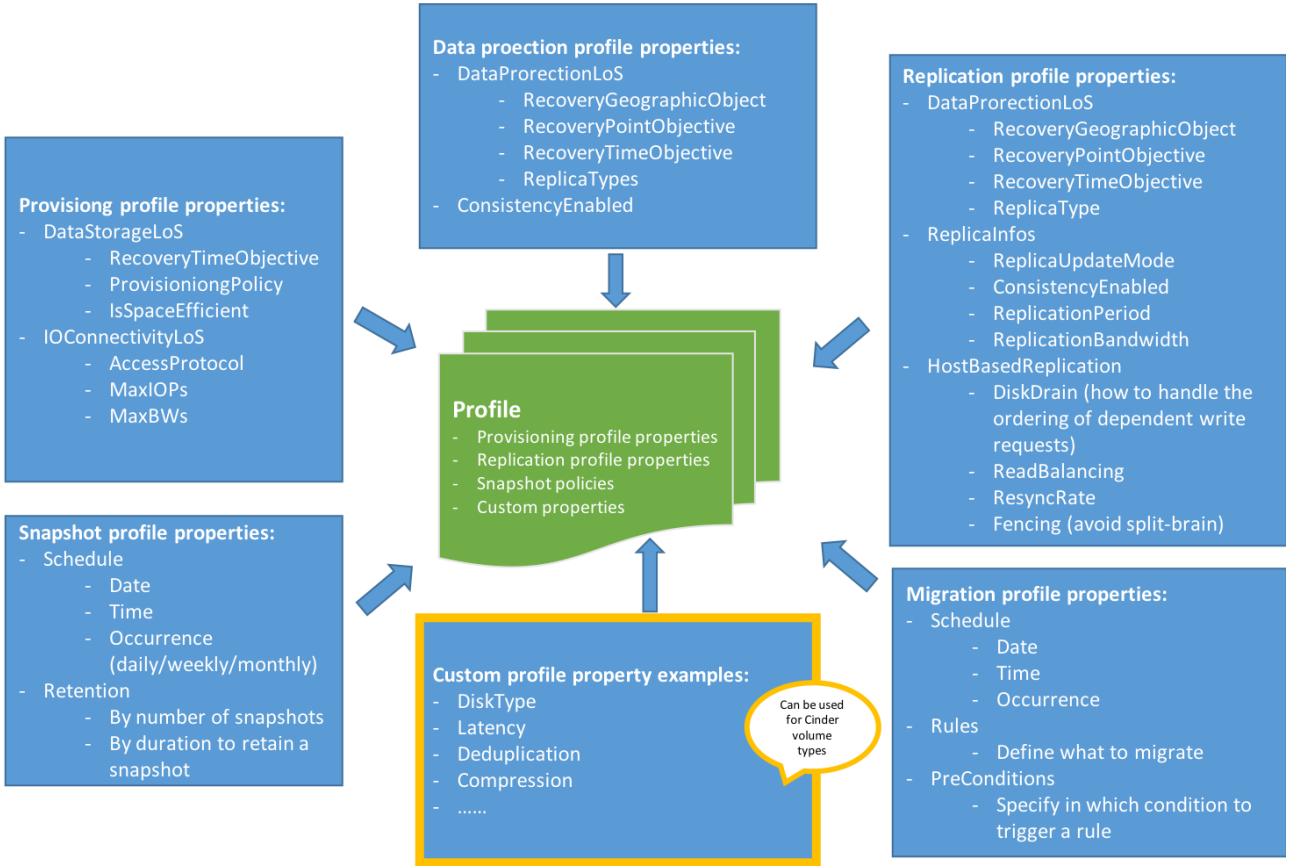


OpenSDS Overview - Architectu

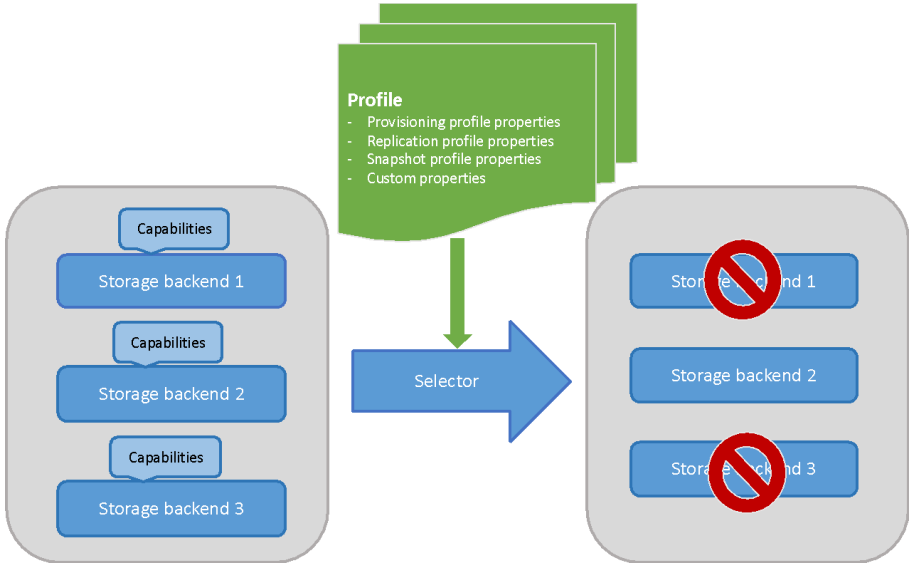
CHINA OpenInfra Days
IT大咖说 知识共享平台



Profile Definitions



Mapping Profiles to Capabilities



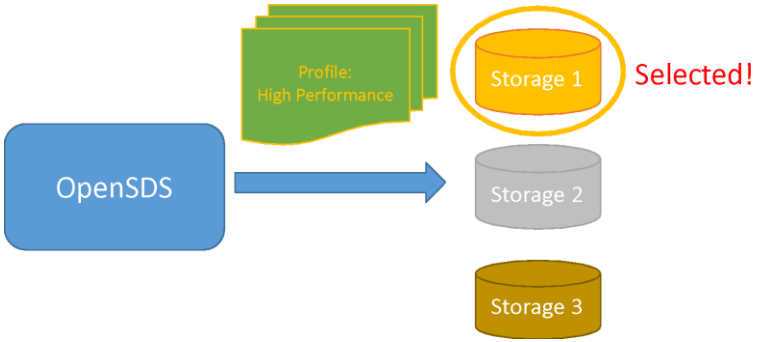
Profile Example

DATA STORAGE

- **DataStorageLoS**
  - RecoveryTimeObjective (Immediate, Nearline, Offline, Online)
  - ProvisioningPolicy (thin, thick)
  - IsSpaceEfficient (true, false)

DATA PERFORMANCE

- **IOConnectivitLoS**
  - AccessProtocol (iSCSI, FC, RBD …)
  - MaxIOPS
  - MaxBWS

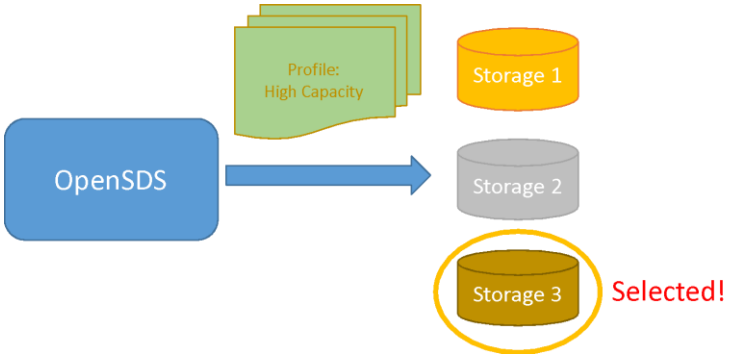


REPLICATION

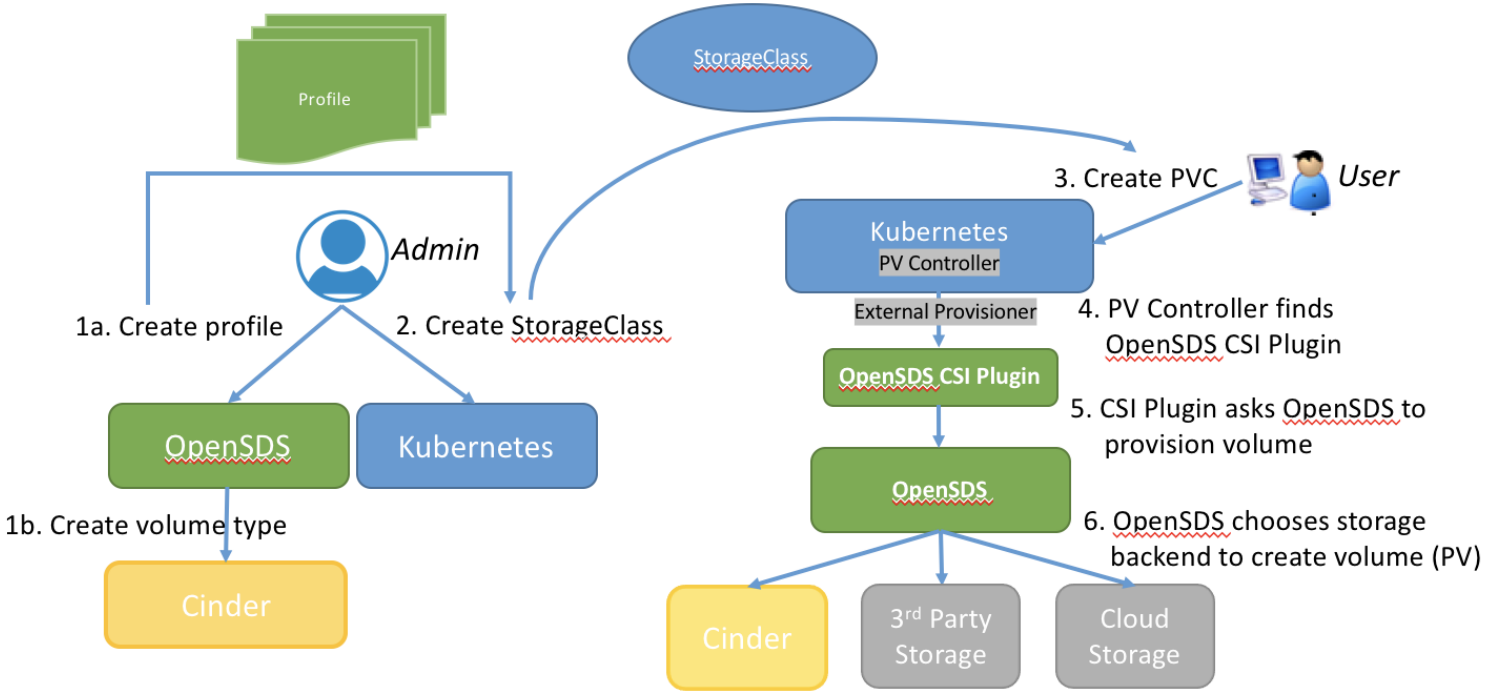
- **DataProtectionLoS**
  - RecoveryGeographicObjective
  - RecoveryTimeObjective
  - RecoveryPointObjective
  - ReplicaType
- **ReplicaInfos**
  - ReplicationUpdateMode
  - ConsistencyEnabled
  - ReplicationPeriod
  - ReplicationBandwidth

SNAPSHOT

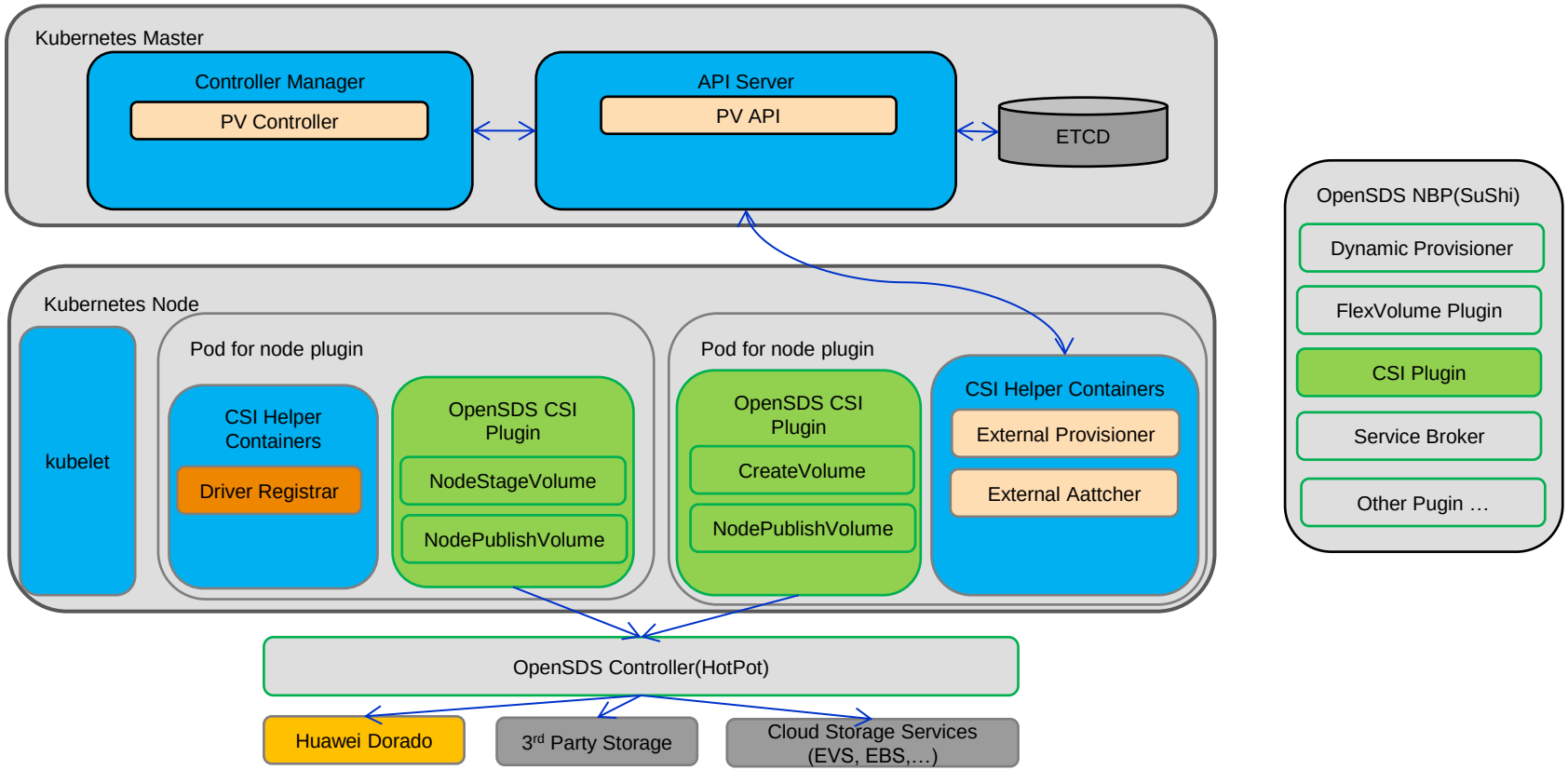
- **Schedule**
  - Date
  - Time
  - Occurrence (daily/weekly/monthly)
- **Retention**
  - By number of snapshots
  - By duration to retain a snapshot



Demo - Mapping OpenSDS Profile to K8S StorageClass



Demo - Provision and Manage Persistent Volumes using



StorageClass with Profile Parameter

HighPerformanceSC.yaml

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: opensds-csi-high-performance-sc
provisioner: csi-opensdsplugin
parameters:
  profile: High-Performance
```

HighPerformancePVC.yaml

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: opensds-csi-high-performance-pvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: opensds-csi-high-performance-sc
```

Note: profile parameter can be profile id or name

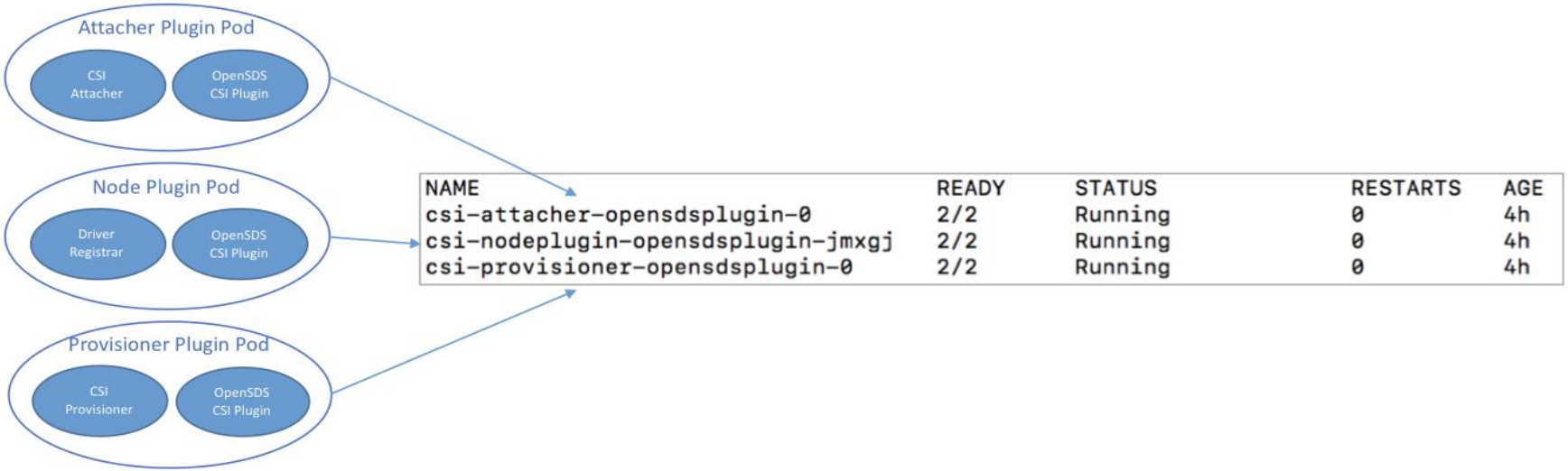
Demo - Running OpenSDS CSI Plu...

CHINA OpenInfra Days
IT大咖说 知识共享平台

- Create OpenSDS CSI plugin pods:

  `kubectl create -f csi/server/deploy/kubernetes`

- Three pods can be found by `kubectl get pod`:



Demo Using OpenSDS Volume

- Create nginx application

  kubectl create -f

  csi/server/examples/kubernetes/nginx.yaml

- An OpenSDS volume is mounted at */var/lib/www/html*.

  docker exec -it <nginx container id> /bin/bash

```
root@nginx:/# mount | grep html
/dev/sda on /var/lib/www/html type ext4 (rw,relatime,data=ordered)
```

nginx.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - image: nginx
    imagePullPolicy: IfNotPresent
    name: nginx
    ports:
    - containerPort: 80
      protocol: TCP
    volumeMounts:
      - mountPath: /var/lib/www/html
        name: csi-data-opensdsplugin
  volumes:
  - name: csi-data-opensdsplugin
    persistentVolumeClaim:
      claimName: opensds-csi-high-performance-pvc
      readOnly: false
```

References

- Container Storage Interface Spec
- CSI in Kubernetes Design Proposal
- What Is The Container Storage Interface (CSI)

Thank you!