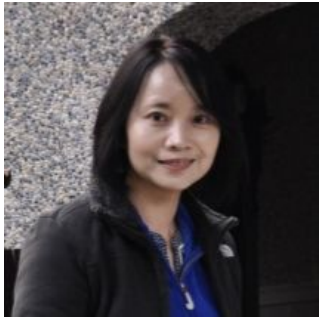
Managing and Protecting Persistent Volumes for Kubernetes

Xing Yang, Huawei and Jay Bryant, Lenovo

Bio

Xing Yang

- Principal Architect at Huawei
- Project and Architecture Lead of OpenSDS
- Core Reviewer in Cinder and Manila since Juno
- Contributor in Kubernetes and Container Storage Interface (CSI)
- IRC and Slack: xyang or xyang1
- GitHub: xing-yang
- Email: xingyang105@gmail.com
- Twitter: @2000Xyang



Bio

Jay Bryant

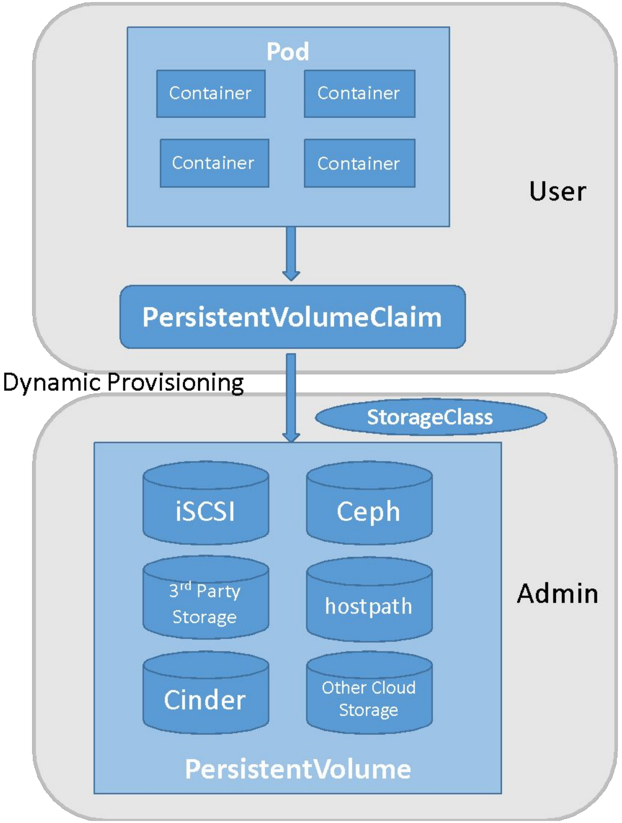
- Cloud Storage Lead at Lenovo
- Core Reviewer in Cinder since Icehouse and current PTL of Cinder
- Stable Maintainer and OSLO and Doc Liaison
- OpenSDS TSC Member
- IRC or Slack: jungleboyj
- GitHub: jsbryant
- Email: jsbryant@electronicjungle.net
- Twitter: @jungleboyj

Agenda

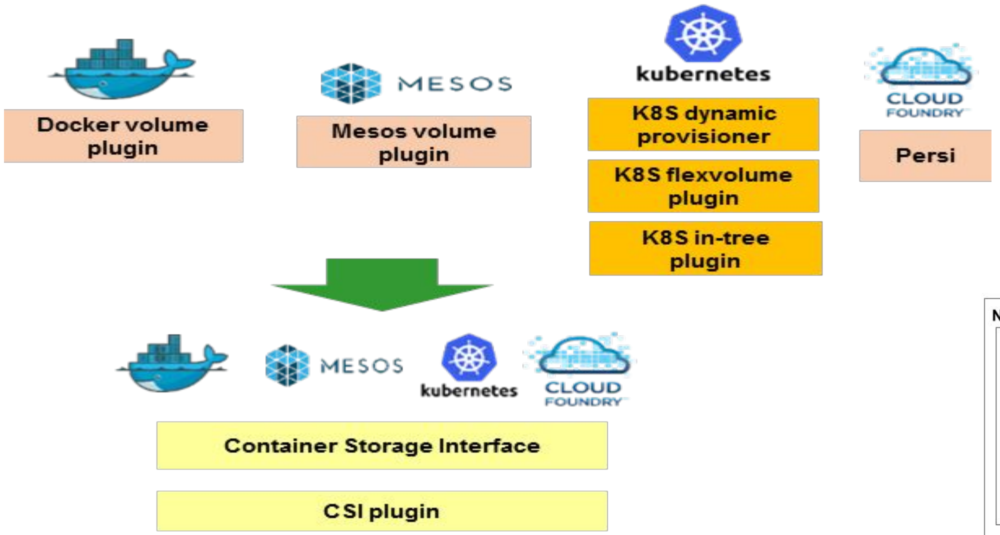
- Kubernetes Persistent Volumes and CSI
- Why Cinder and OpenSDS for Kubernetes?
- Cinder Overview and Cinder stand-alone
- OpenSDS Overview
- Integrate OpenSDS with Cinder
- Provision and Manage Persistent Volumes using OpenSDS and Cinder
- Data Protection for Persistent Volumes
- Disaster Recovery for Persistent Volumes
- Future Integration
- OpenSDS Roadmap for Aruba and Bali Release
- OpenSDS Community
- Demo

Kubernetes Persistent Volumes

- 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.



Container Storage Interface (CSI)



Docker volume plugin

Mesos volume plugin

K8S dynamic provisioner

K8S flexvolume plugin

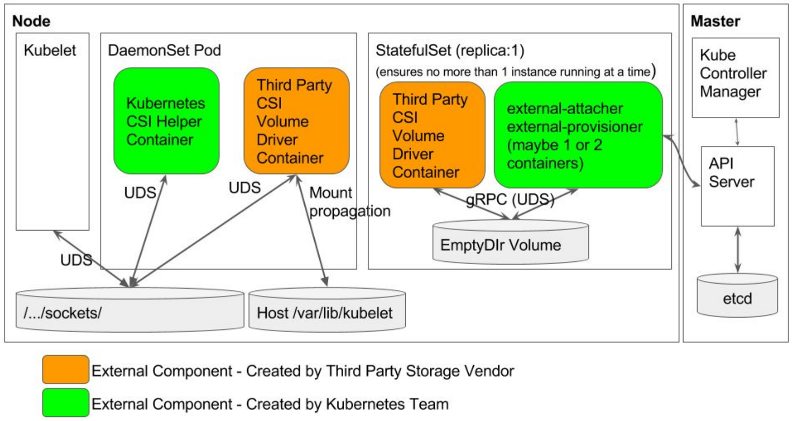
K8S in-tree plugin

Persi

Container Storage Interface

CSI plugin

CSI is an industry standard defined to enable storage vendors to develop a plugin once and have it work across a number of container orchestration systems.



Node

Kubelet

DaemonSet Pod

Kubernetes CSI Helper Container

Third Party CSI Volume Driver Container

StatefulSet (replica:1)
(ensures no more than 1 instance running at a time)

Third Party CSI Volume Driver Container

external-attacher external-provisioner (maybe 1 or 2 containers)

UDS

UDS

UDS

Mount propagation

gRPC (UDS)

EmptyDIr Volume

/.../sockets/

Host /var/lib/kubelet

Master

Kube Controller Manager

API Server

etcd

🟧 External Component - Created by Third Party Storage Vendor

🟩 External Component - Created by Kubernetes Team

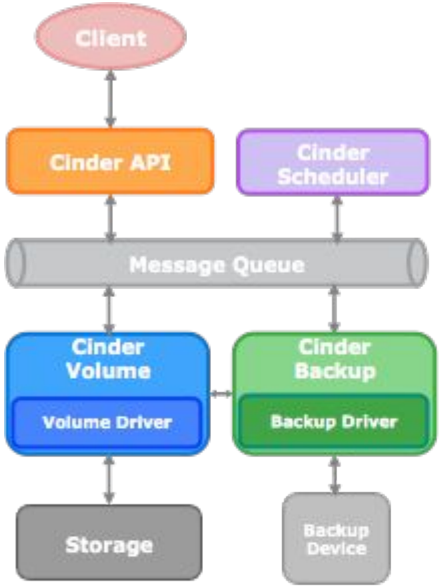
Source: https://github.com/kubernetes/community/blob/master/contributors/design-proposals/storage/container-storage-interface.md

Why Cinder and OpenSDS for Kubernetes

- Storage functionalities in Kubernetes are still evolving.
- Cinder and OpenSDS can provide additional storage functionalities for Kubernetes.
- Provide unified control for traditional cloud and cloud native environment.

Cinder Overview

- Mission statement: To implement services and libraries to provide on demand, self-service access to Block Storage resources. Provide Software Defined Block Storage via abstraction and automation on top of various traditional backend block storage devices.
- 70+ drivers in Cinder currently.



Cinder Stand-alone

- Containerized Cinder services
- Deploys using docker-compose
- Uses noauth option
- Allows Cinder to provide block storage service outside of OpenStack

Cinder Lib

- Cinder Library is a Python library that allows storage drivers to be used outside of Cinder
- Removed DBMS, message broker, Cinder API, scheduler, and volume manager layers
- Currently in Alpha status
- https://github.com/Akrog/cinderlib

OpenSDS Overview - Core Projects

SUSHI

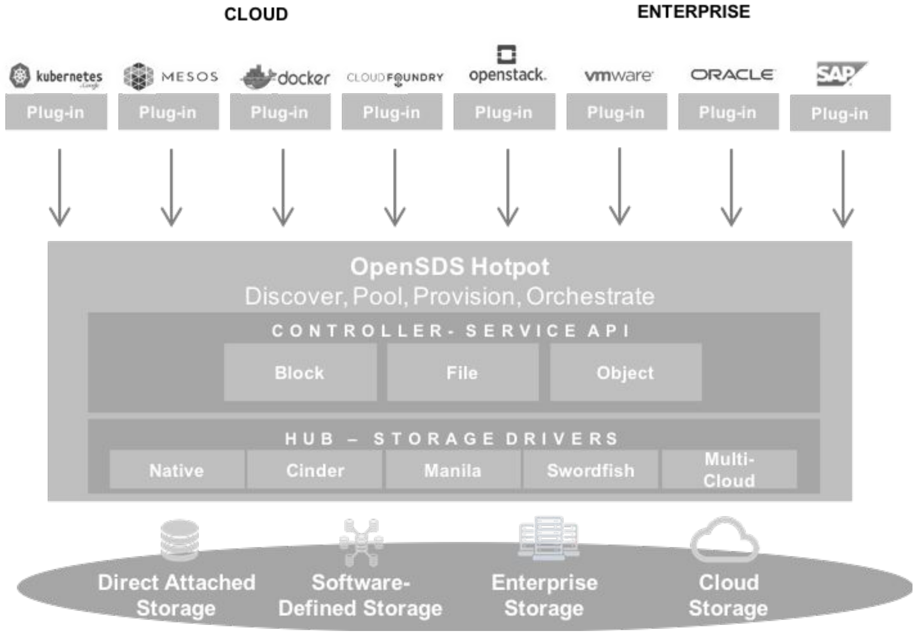
The Northbound Plug-ins Project ▶

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



CLOUD

ENTERPRISE

Plug-in | Plug-in | Plug-in | Plug-in | Plug-in | Plug-in | Plug-in | Plug-in

OpenSDS Hotpot

Discover, Pool, Provision, Orchestrate

CONTROLLER- SERVICE API

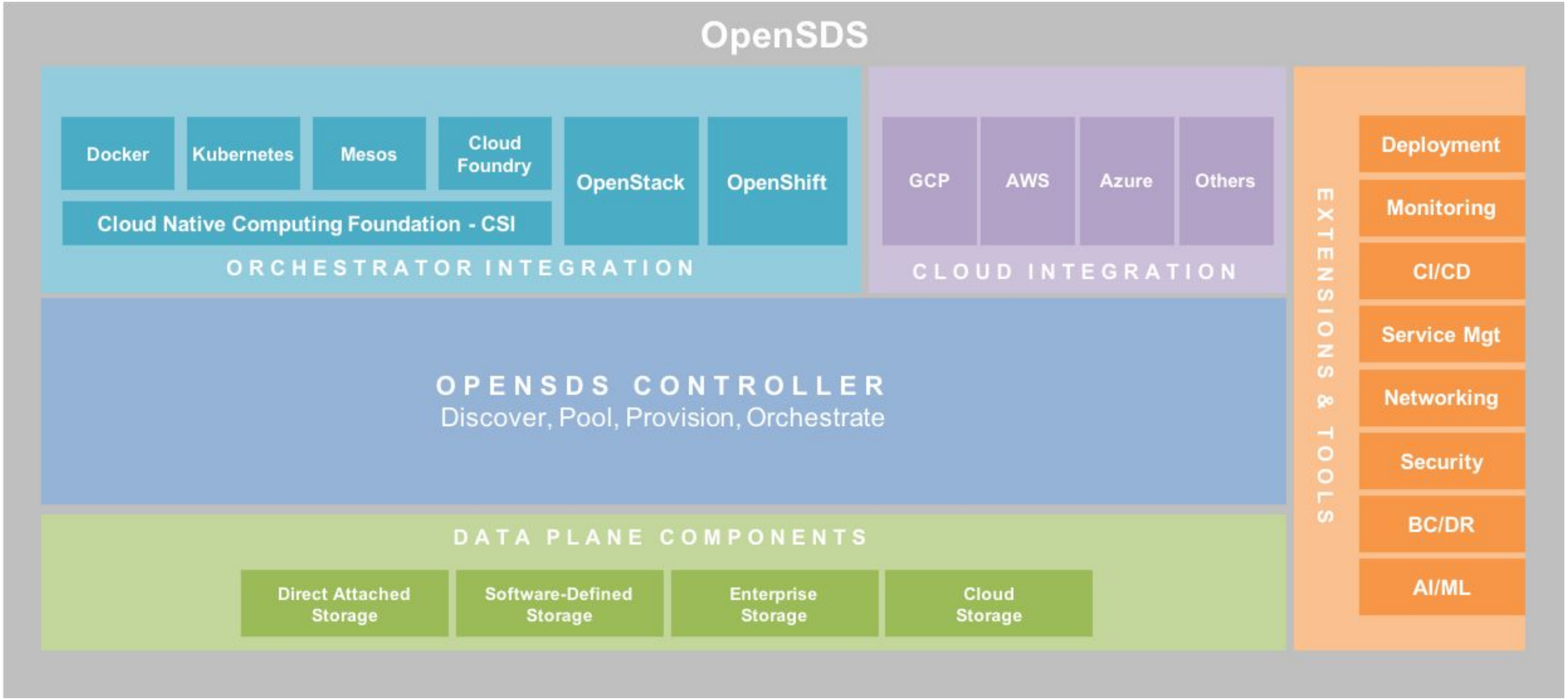
Block | File | Object

HUB – STORAGE DRIVERS

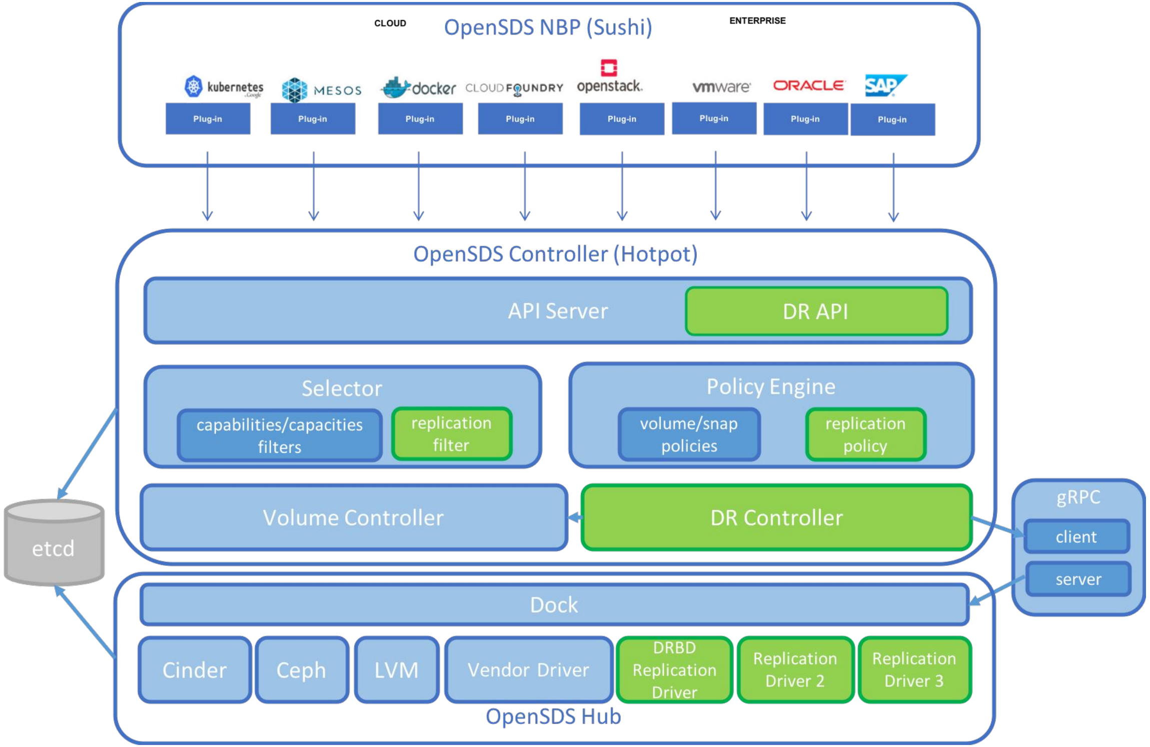
Native | Cinder | Manila | Swordfish | Multi-Cloud

Direct Attached Storage | Software-Defined Storage | Enterprise Storage | Cloud Storage

OpenSDS Overview - Project Framework



OpenSDS Overview - Architecture

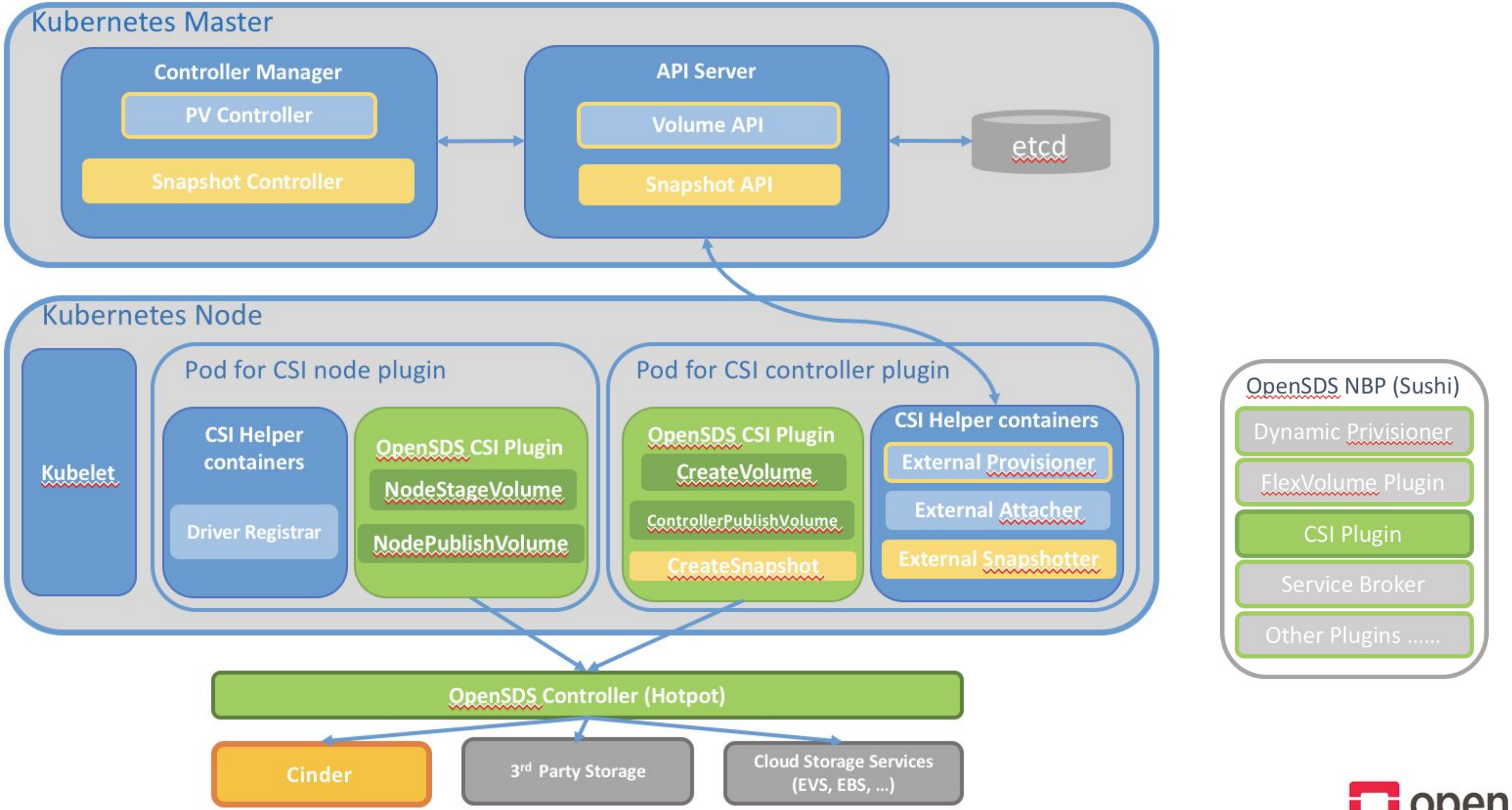


Integrate OpenSDS with Cinder

- OpenSDS uses Cinder to provision storage
  - OpenSDS southbound volume driver for Cinder
  - Cinder in OpenStack deployment, Cinder standalone, or Cinder lib

openSDS

Provision and Manage Persistent Volumes using OpenSDS and Cinder



Kubernetes Master

Controller Manager

PV Controller

Snapshot Controller

API Server

Volume API

Snapshot API

etcd

Kubernetes Node

Kubelet

Pod for CSI node plugin

CSI Helper containers

Driver Registrar

OpenSDS CSI Plugin

NodeStageVolume

NodePublishVolume

Pod for CSI controller plugin

OpenSDS CSI Plugin

CreateVolume

ControllerPublishVolume

CreateSnapshot

CSI Helper containers

External Provisioner

External Attacher

External Snapshotter

OpenSDS Controller (Hotpot)

Cinder

3rd Party Storage

Cloud Storage Services (EVS, EBS, …)

OpenSDS NBP (Sushi)

Dynamic Privisioner

FlexVolume Plugin

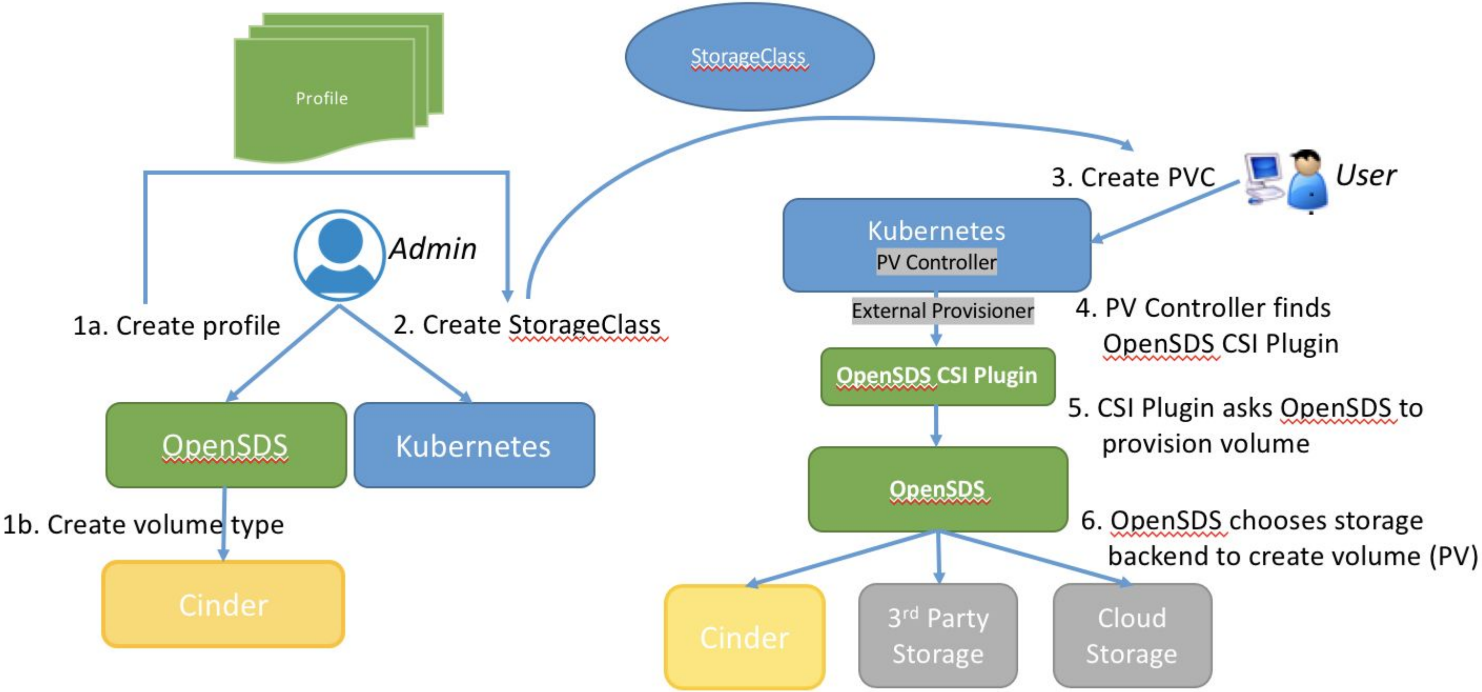
CSI Plugin

Service Broker

Other Plugins ……

openstack.

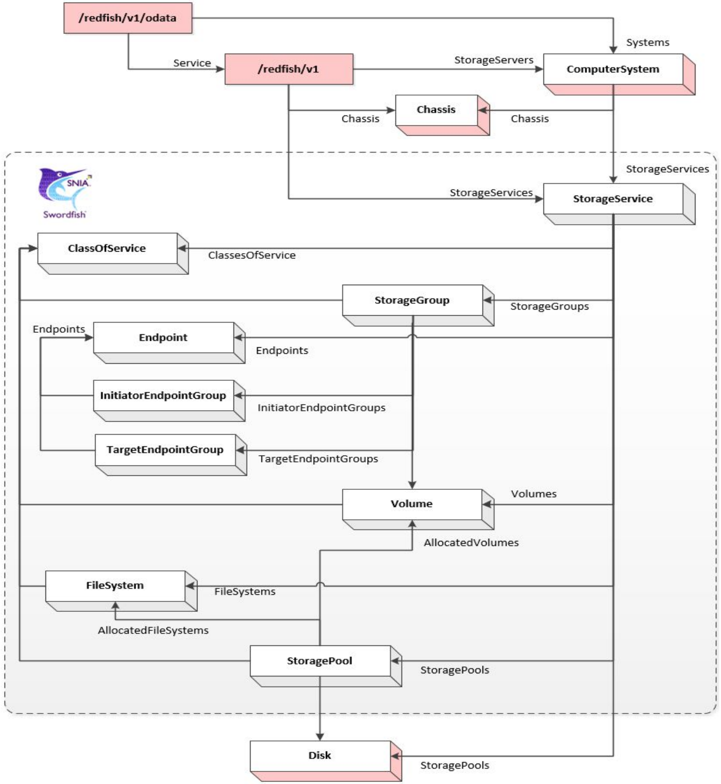
Mapping OpenSDS Profile and Cinder Volume Type to K8S StorageClass



Policy Driven SPDM

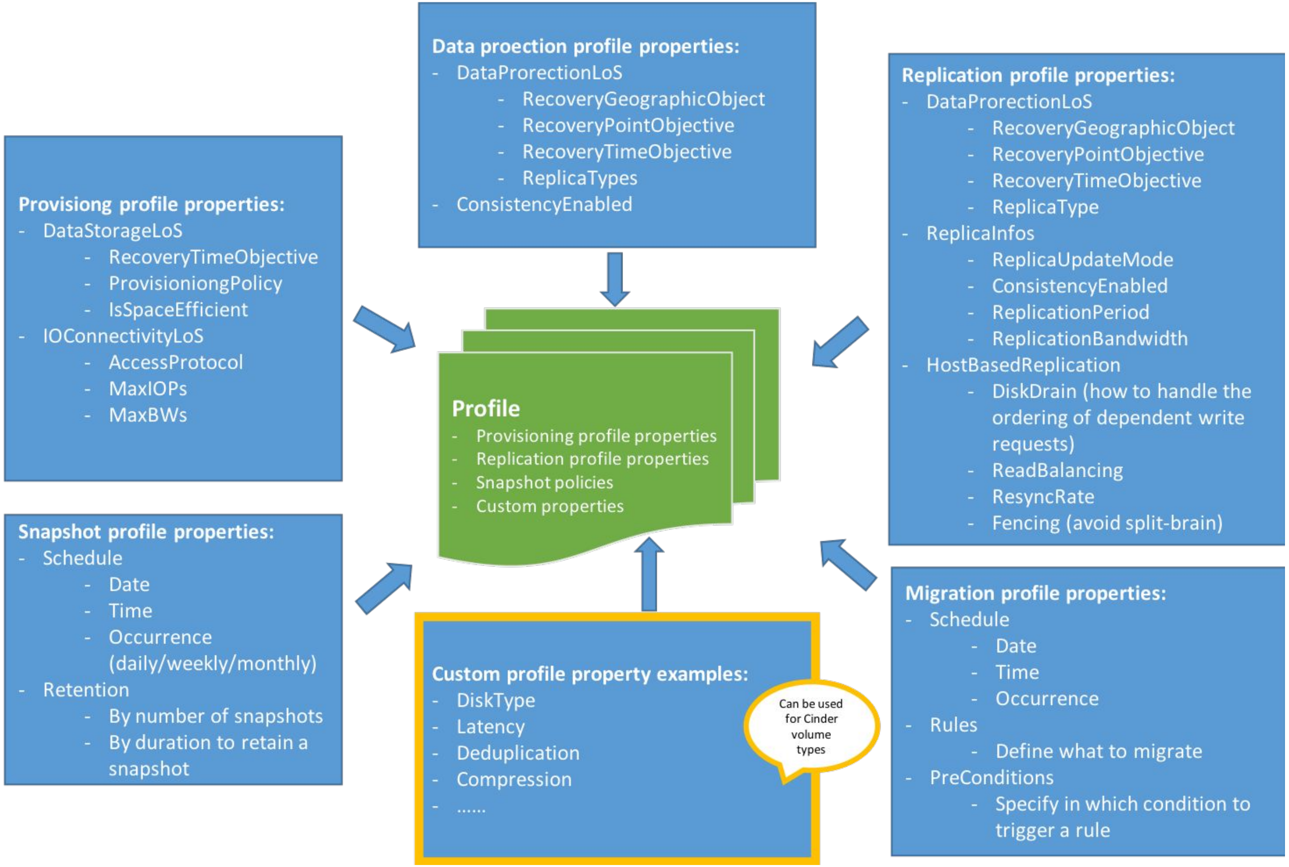
openSDS
SNIA
Swordfish
DMTF
Redfish

- OpenSDS profile is based on Swordfish specification.
- The SNIA Swordfish™ specification helps to provide a unified approach for the management of storage and servers in hyperscale and cloud infrastructure environments, supported by multiple storage vendors.
- An extension of the DMTF (Distributed Management Task Force) Redfish specification.
  - Redfish is designed by the DMTF's Scalable Platforms Management Forum (SPMF) to create and publish an open industry standard specification and schema for management of scalable platform hardware. It is a RESTful interface over HTTPS in JSON format based on OData v4.



Source: Swordfish_v1.0.5_Specification

Profile Definitions



Data proection profile properties:
- DataProrectionLoS
    - RecoveryGeographicObject
    - RecoveryPointObjective
    - RecoveryTimeObjective
    - ReplicaTypes
- ConsistencyEnabled

Provisiong profile properties:
- DataStorageLoS
    - RecoveryTimeObjective
    - ProvisioniongPolicy
    - IsSpaceEfficient
- IOConnectivityLoS
    - AccessProtocol
    - MaxIOPs
    - MaxBWs

Replication profile properties:
- DataProrectionLoS
    - RecoveryGeographicObject
    - RecoveryPointObjective
    - RecoveryTimeObjective
    - ReplicaType
- ReplicaInfos
    - ReplicaUpdateMode
    - ConsistencyEnabled
    - ReplicationPeriod
    - ReplicationBandwidth
- HostBasedReplication
    - DiskDrain (how to handle the ordering of dependent write requests)
    - ReadBalancing
    - ResyncRate
    - Fencing (avoid split-brain)

Profile
- Provisioning profile properties
- Replication profile properties
- Snapshot policies
- Custom properties

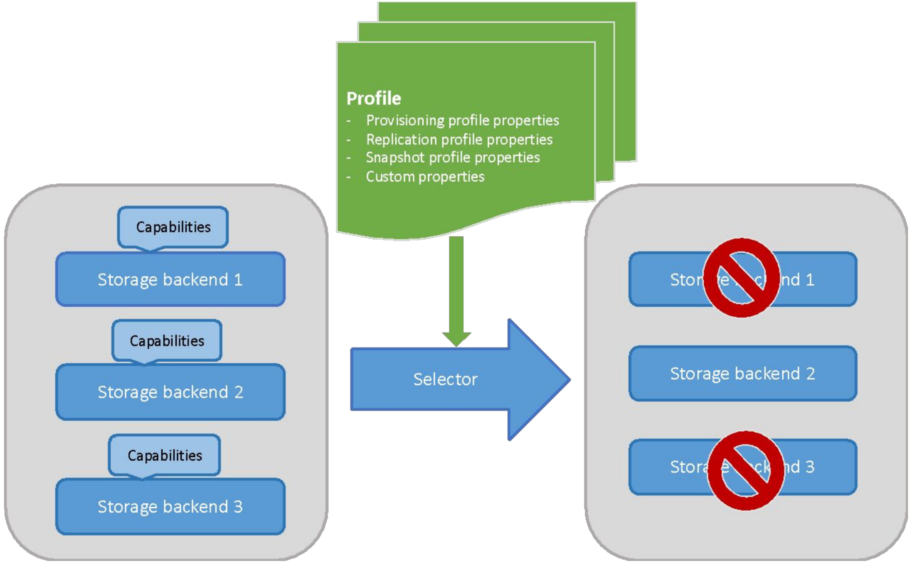
Snapshot profile properties:
- Schedule
    - Date
    - Time
    - Occurrence (daily/weekly/monthly)
- Retention
    - By number of snapshots
    - By duration to retain a snapshot

Custom profile property examples:
- DiskType
- Latency
- Deduplication
- Compression
- ……

Can be used for Cinder volume types

Migration profile properties:
- Schedule
    - Date
    - Time
    - Occurrence
- Rules
    - Define what to migrate
- PreConditions
    - Specify in which condition to trigger a rule

Mapping Profiles to Capabilities



Profile Example

DATA STORAGE

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

DATA PERFORMANCE

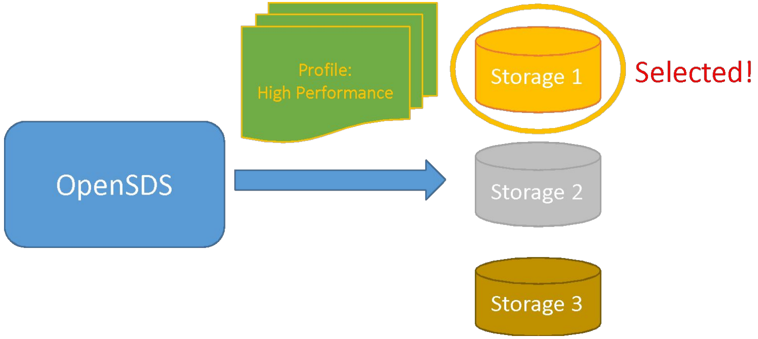
- **IOConnectivitLoS**
  - AccessProtocol (iSCSI, FC, RBD …)
  - MaxIOPS
  - MaxBWS

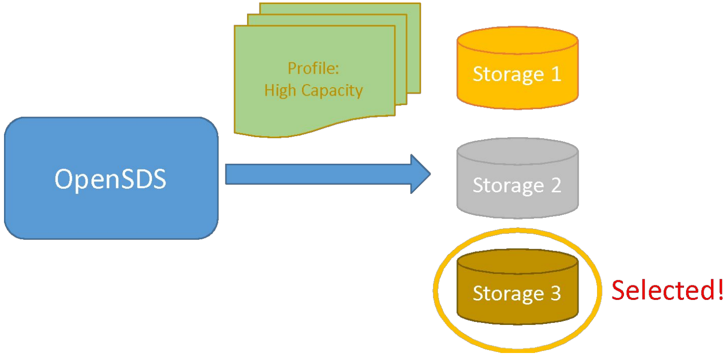
REPLICATION

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

SNAPSHOT

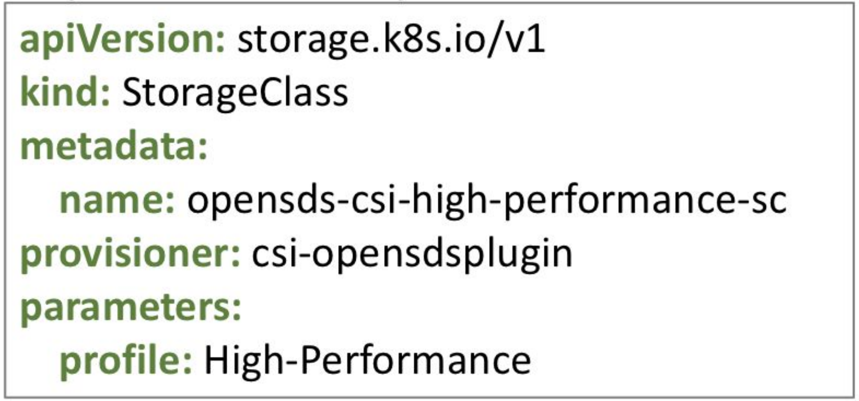
- **Schedule**
  - Date
  - Time
  - Occurrence (daily/weekly/monthly)
- **Retention**
  - By number of snapshots
  - By duration to retain a snapshot





StorageClass with Profile Parameter

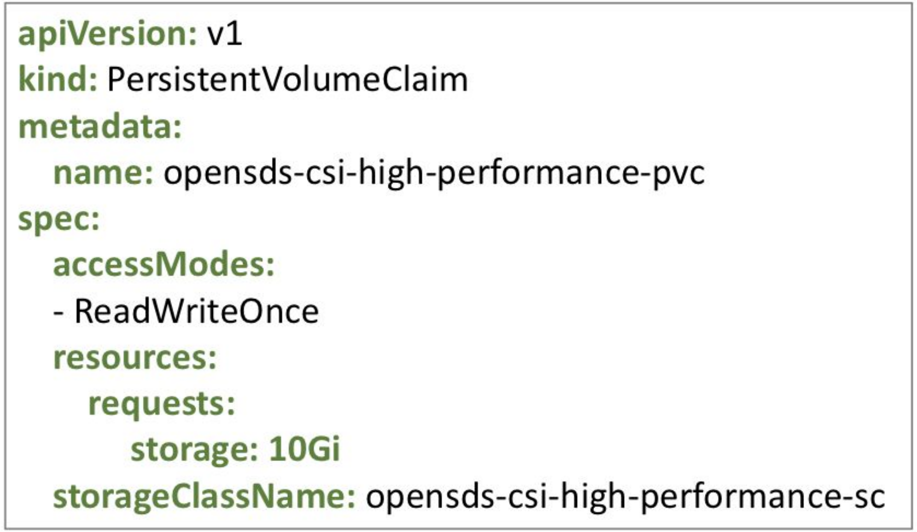
HighPerformanceSC.yaml



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

Note: profile parameter can be profile id or name

HighPerformancePVC.yaml



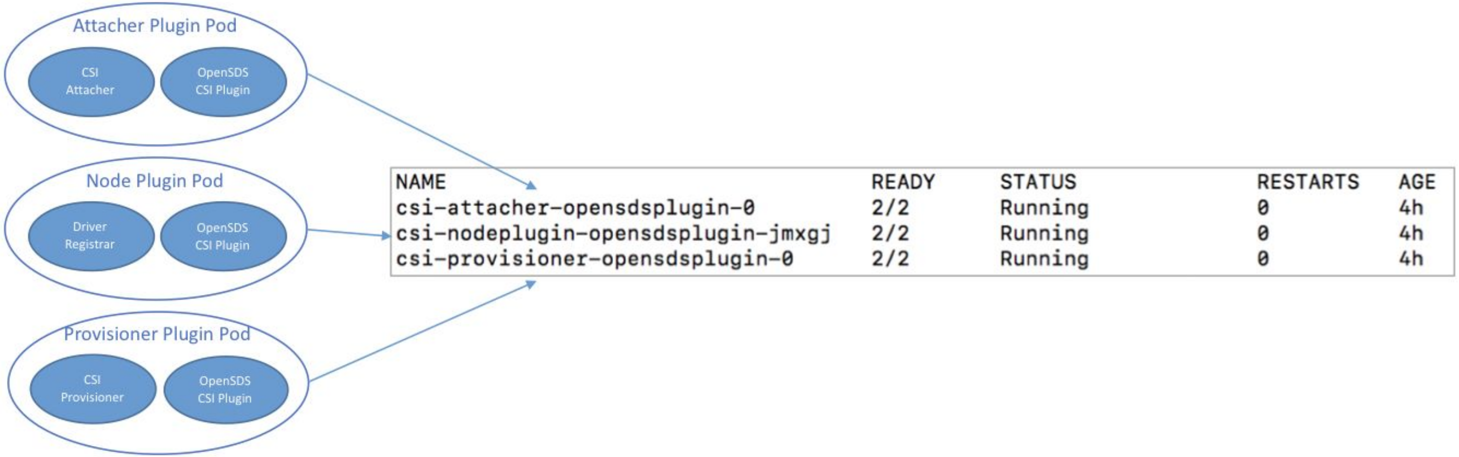
```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: opensds-csi-high-performance-pvc
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: opensds-csi-high-performance-sc
```

Running OpenSDS CSI Plugin

- Create OpenSDS CSI plugin pods:

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

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



```
NAME                                  READY   STATUS      RESTARTS   AGE
csi-attacher-opensdsplugin-0          2/2     Running     0          4h
csi-nodeplugin-opensdsplugin-jmxgj    2/2     Running     0          4h
csi-provisioner-opensdsplugin-0       2/2     Running     0          4h
```

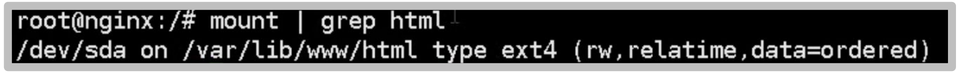
Using OpenSDS Volume

- 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`

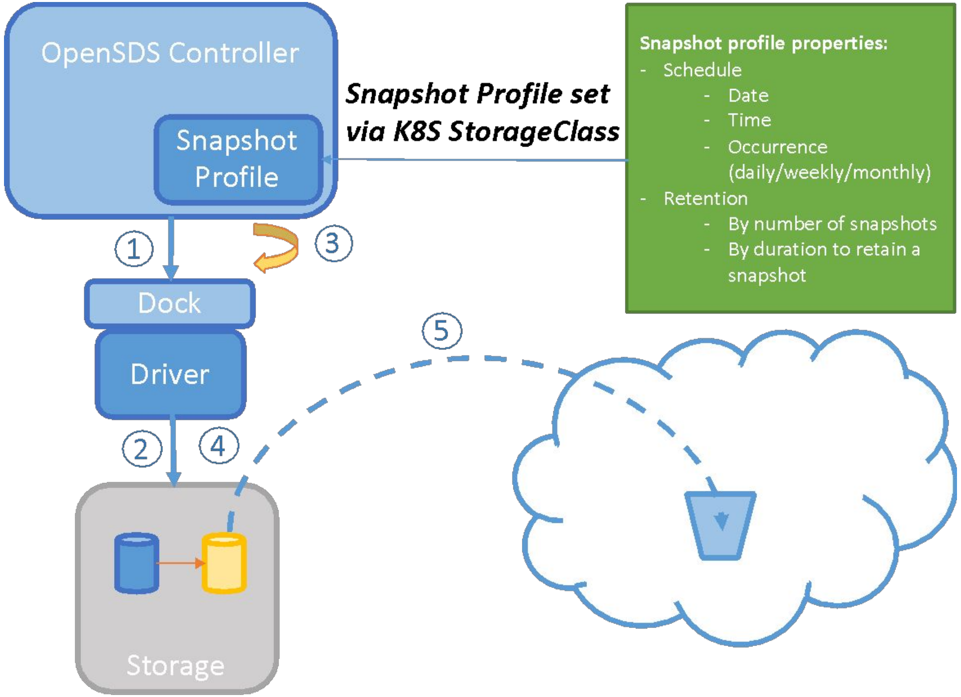


```
root@nginx:/# mount | grep html
/dev/sda on /var/lib/www/html type ext4 (rw,relatime,data=ordered)
```

nginx.yaml

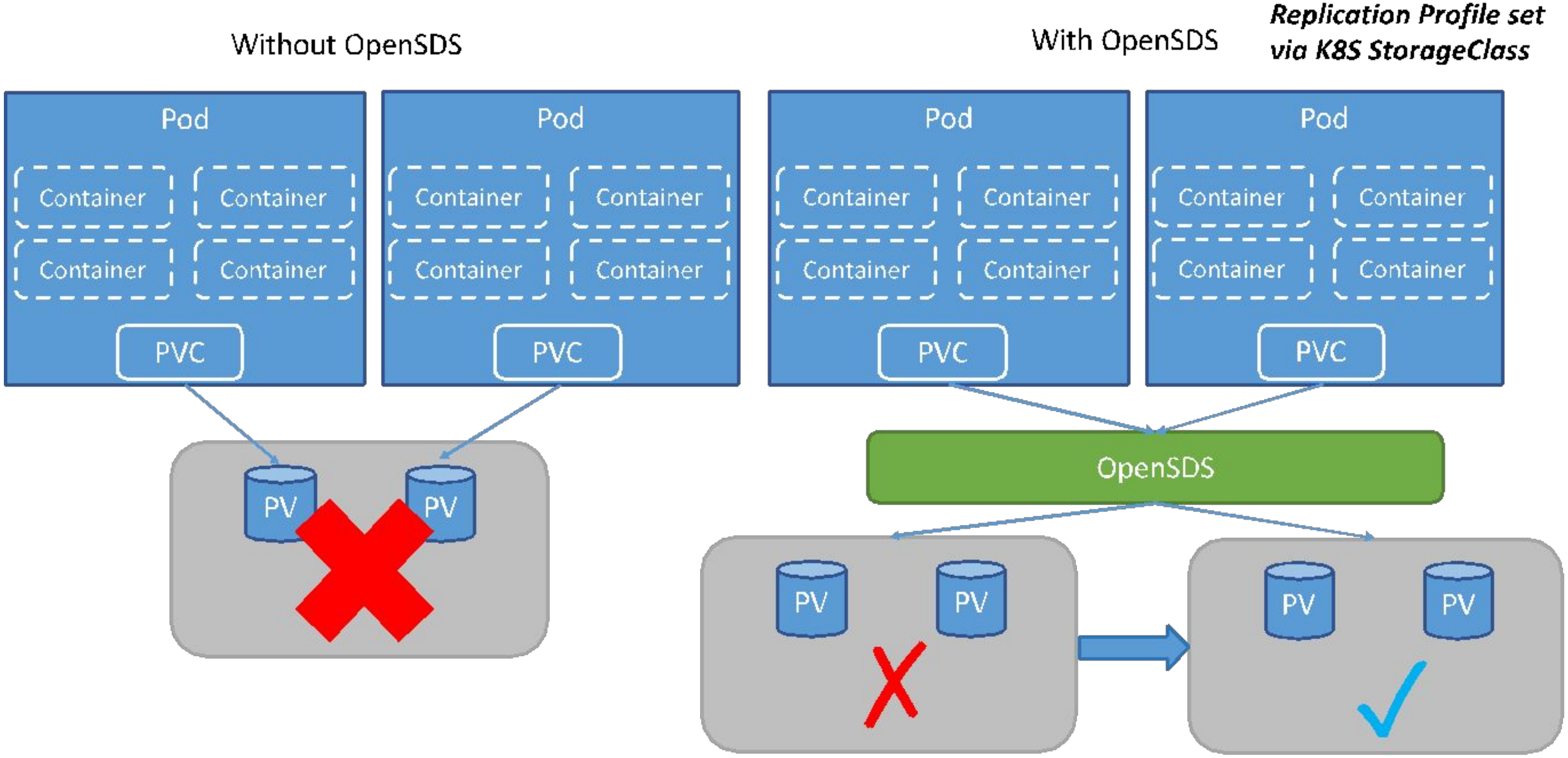
```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - image: nginx
   imagePullPolicy: IfNotPresent
   name: nginx
   ports:
   - containerPort: 80
    protocol: TCP
   volumeMounts:
     - mountPath: /var/lib/www/html
      name: csi-data-opensdsplugin
  volumes:
  - name: csi-data-opensdsplugin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

Data Protection for Persistent Volumes

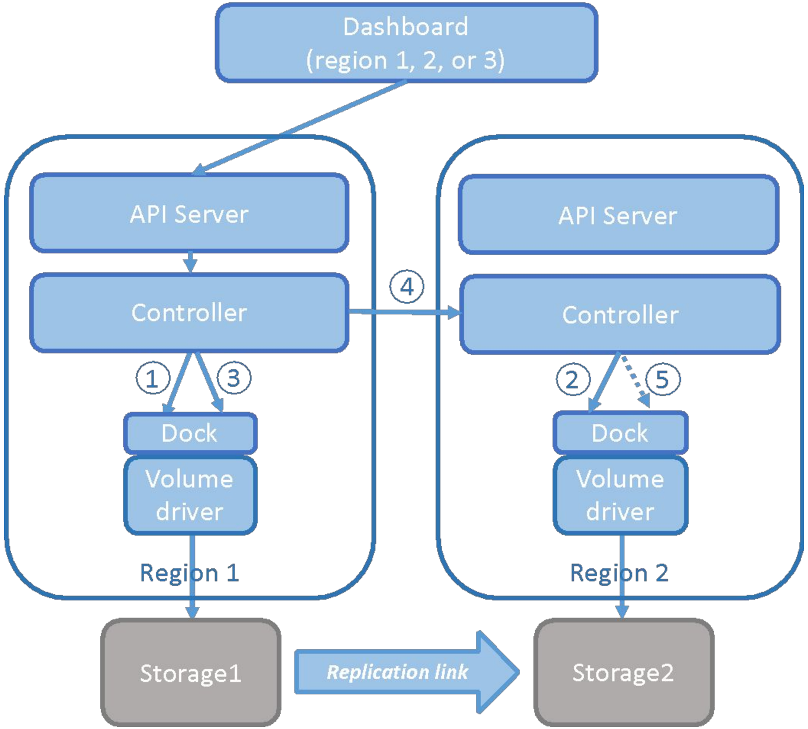

OpenSDS Controller
Snapshot Profile
Snapshot Profile set via K8S StorageClass
Snapshot profile properties:
- Schedule
- Date
- Time
- Occurrence (daily/weekly/monthly)
- Retention
- By number of snapshots
- By duration to retain a snapshot
1
3
Dock
5
Driver
2
4
Storage

1. Controller asks driver to create a volume.
2. Driver creates a volume on the storage backend.
3. Controller periodically asks driver to create a snapshot based on policies defined in the Snapshot Profile.
4. Driver creates a snapshot on the storage backend.
5. Driver uploads the snapshot to an object store on premise or in the cloud based on the snapshot profile.

Disaster Recovery for Persistent Volumes

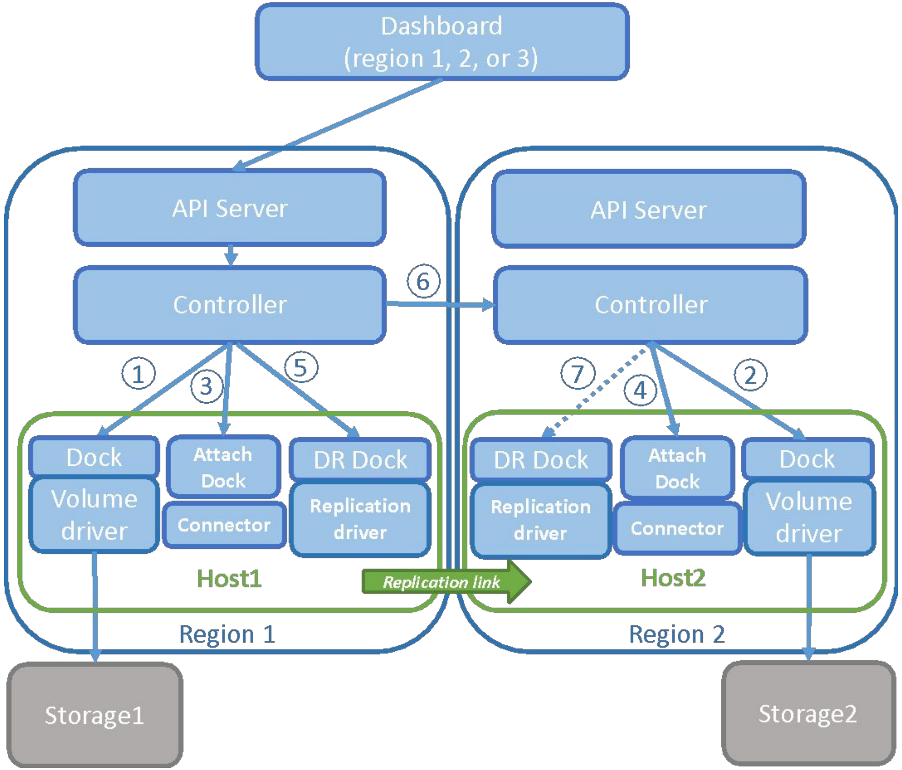


Array-based Replication



1. Creates source volume
   - Creates entry in db
   - Creates volume on Storage1.
2. Creates target volume
   - Creates entry in db
   - Creates volume on Storage2
3. Creates source replication
   - Creates entry in db
   - Creates replication relationship on Storage1 and Storage2
4. Controller 1 communicates with controller 2 to create target replication
5. Controller 2 creates entry in db

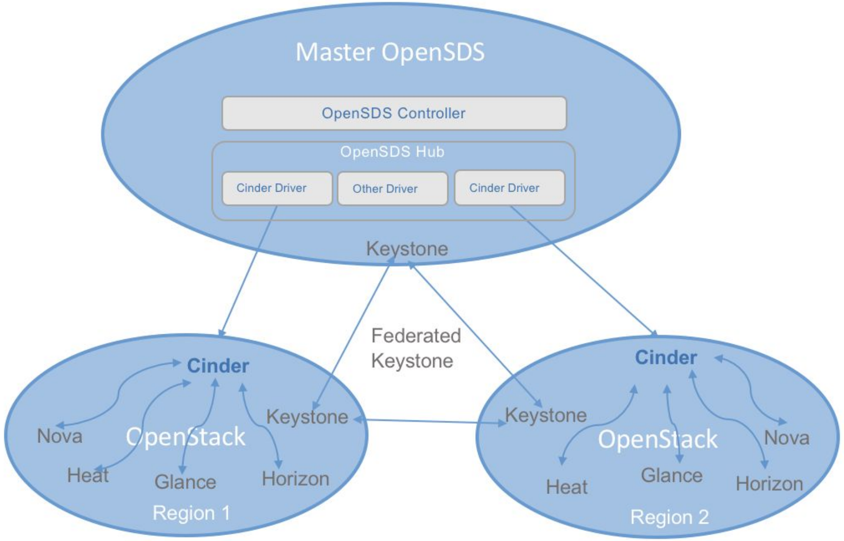
Host-based Replication



1. Creates source volume
   - Creates entry in db
   - Creates volume on Storage1
2. Creates target volume
   - Creates entry in db
   - Creates volume on Storage2
3. Attach source volume to Host1
   - Update volume entry in db with host info
4. Attach target volume to Host2
   - Update volume entry in db with host info
5. Controller 1 Creates source replication
   - Creates entry in db
   - Creates replication relationship on Host1 and Host2 (Host1 is primary)
6. Controller 1 communicates with controller 2 to create target replication
7. Controller 2 creates entry for target replication in db

Future Integration

- Multi-OpenStack
  - Use Federated Keystone or Multi-region Keystone
- Multi-Cloud Control



OpenSDS Roadmap v0.14

2017H2
ZEALAND
Storage For Kubernetes

- Kubernetes FlexVolume
- Vol CRUD
- Standalone Cinder Integration
- CSI Support
- Ceph, LVM

2018H1
ARUBA
Storage Orchestration

- OpenStack
- Replication
  Array-Based, Host-Based
- Dashboard
- Virtual Pools
- Storage Profiles
- NVMeoF preview
- Enumeration
- Block Storage
  - Ceph
  - LVM
  - IBM: XIV, Storwize, SVC
  - Huawei: Dorado

2018H2
BALI
Storage Multi-Cloud

- Data Migration
  Offline, Online*
- Monitoring
- Multi-OpenStack
- S3 Object
- Multi-Cloud Control
- NVMeoF
- Storage Groups
  Snapshots, Replication
- CSI
  Mesos*, Docker*
- Swordfish
  Dell-EMC, NetApp

2019H1
CAPRI
Storage Intelligence

- Analytics
- Lifecycle
- Data Protection
- File Share

2019H2++

- Performance
- Optimization
- Tiering
- Security
- Sharing
- Networking
- SCM

https://github.com/opensds

Governance

Technical Steering Committee

Steven Tan, Chairman
Huawei, VP & CTO Cloud Solution

Rakesh Jain, Vice-Chair
IBM, Research Engineer and Architect

Allen Samuels
Western Digital, R&D Engineering Fellow

Anjaneya "Reddy" Chagam
Intel, Chief SDS Architect

Jay Bryant
Lenovo, Cloud Storage Lead

End-User Advisory Committee

Cosimo Rossetti
Vodafone, Lead Storage Architect

Yusuke Sato
Yahoo Japan, Infrastructure Lead

Kei Kusunoki
NTT Communications, Storage Architect

Yuji Yazawa
Toyota ITC, Group Lead

Supporting Organizations

An industry-wide open source project for software-defined storage management

openSDS

THE LINUX FOUNDATION

DELL EMC

HITACHI Inspire the Next

HUAWEI

vodafone

YAHOO! JAPAN

NTT Communications

WD Western Digital®

Oregon State UNIVERSITY OSU

IBM

TOYOTA InfoTechnology Center Co., Ltd.

FUJITSU

OUR NEWEST MEMBER

LINBIT

Join Us

- Repos: https://github.com/opensds
- Slack: https://opensds.slack.com
- Mailing list: https://lists.opensds.io
- Weekly meetings:

  https://github.com/opensds/design-specs/blob/master/README.md#opensds-technical-meetings

Demo

- Provision storage using OpenSDS CSI plugin with stand-alone Cinder

Thank You

@opensds_io