OpenSDS

OSCON 2018

Disaster Recovery and Data Protection for Kubernetes Persistent Volumes

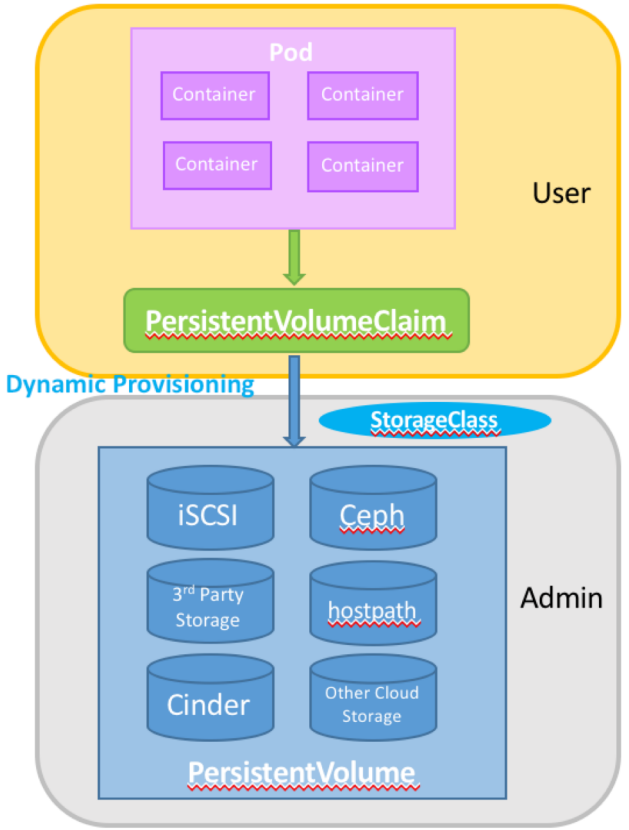
Xing Yang, Principal Architect, Huawei

Agenda

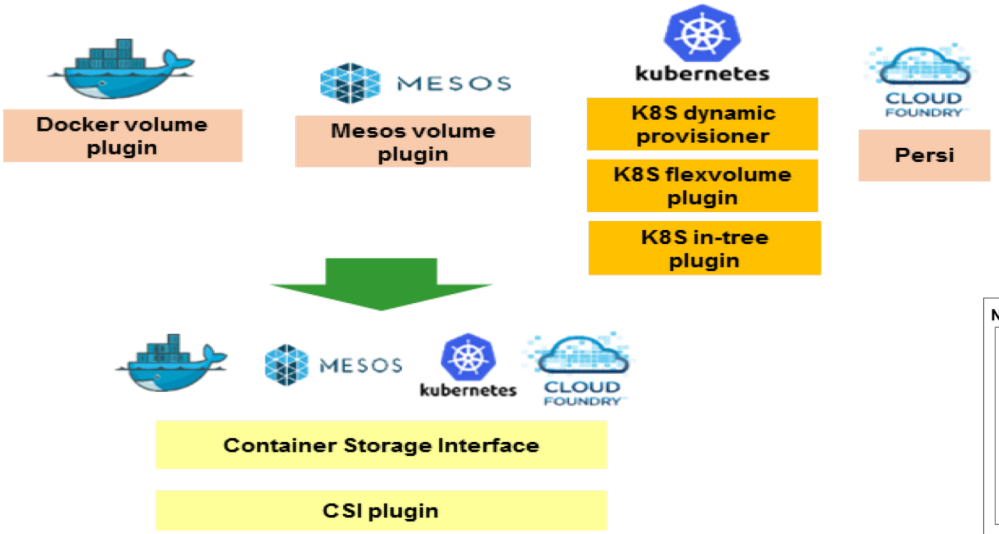
- Kubernetes Persistent Volumes and CSI
- Why OpenSDS for Kubernetes and CSI
- OpenSDS Overview
- Provision and Manage Persistent Volumes
- Disaster Recovery for Persistent Volumes
- Data Protection for Persistent Volumes
- OpenSDS Roadmap for Aruba and Bali Release
- Integration with other ecosystems
  - OpenStack
  - Service broker
- Demo

Kubernetes Persistent Volumes

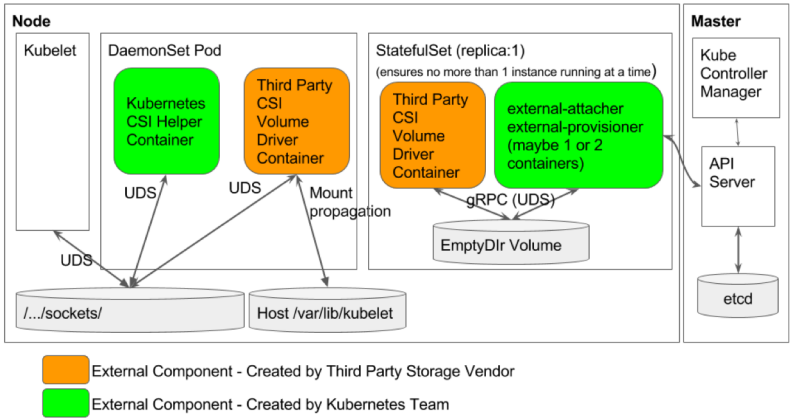
- A PersistentVolume (PV) is a piece of storage in the cluster that has been provisioned by an administrator.
- A PV can be provisioned statically or dynamically.
- A PersistentVolumeClaim (PVC) is a request for storage by a user through a StorageClass.
- A StorageClass provides a way for administrators 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)

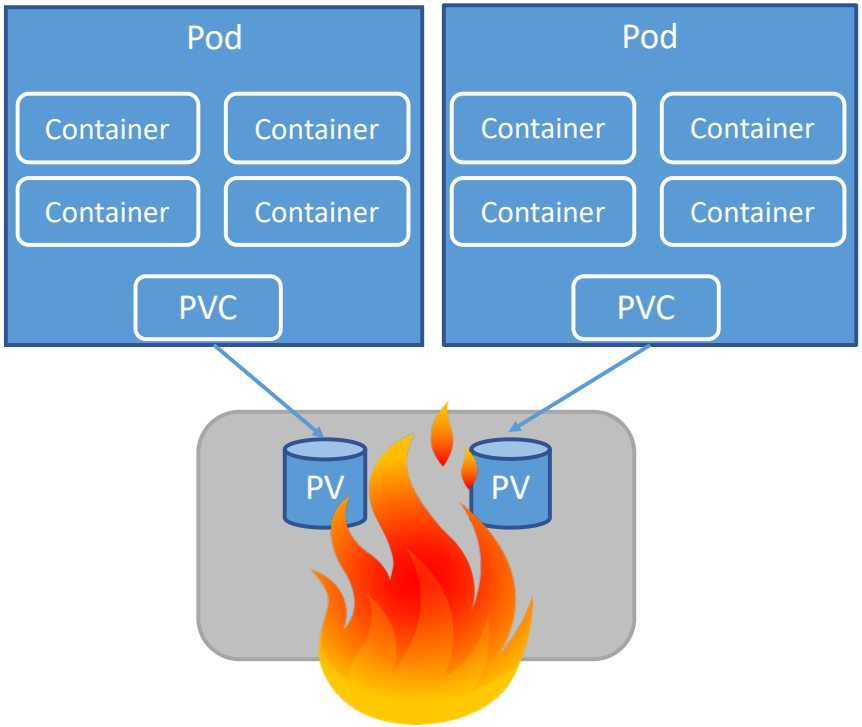


CSI is an industry standard defined to enable storage vendors to develop a plugin once and have it work across a number of container orchestration systems.



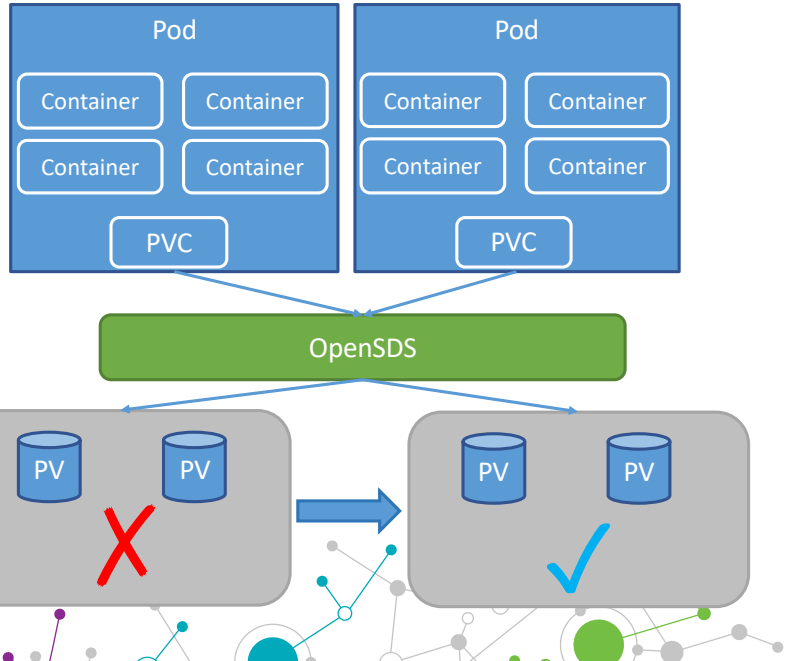
Source: https://github.com/kubernetes/community/blob/master/contributors/design-proposals/storage/container-storage-interface.md

What Happens When Disaster Strikes



Why OpenSDS for Kubernetes and CSI

- Storage functionalities in Kubernetes and CSI are still evolving.
- OpenSDS can provide additional storage functionalities such as data protection and disaster recovery.
- Provide unified control for traditional cloud and cloud native environment.



OpenSDS Overview – Linux Foundation Project

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

OpenSDS Overview – Core Projects

SUSHI

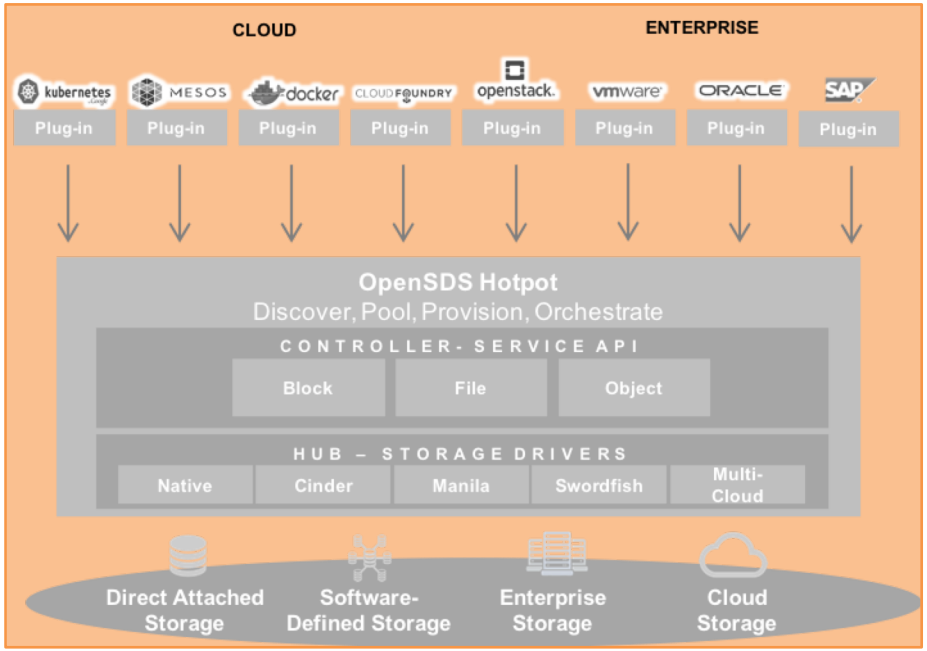
The Northbound Plug-ins Project

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

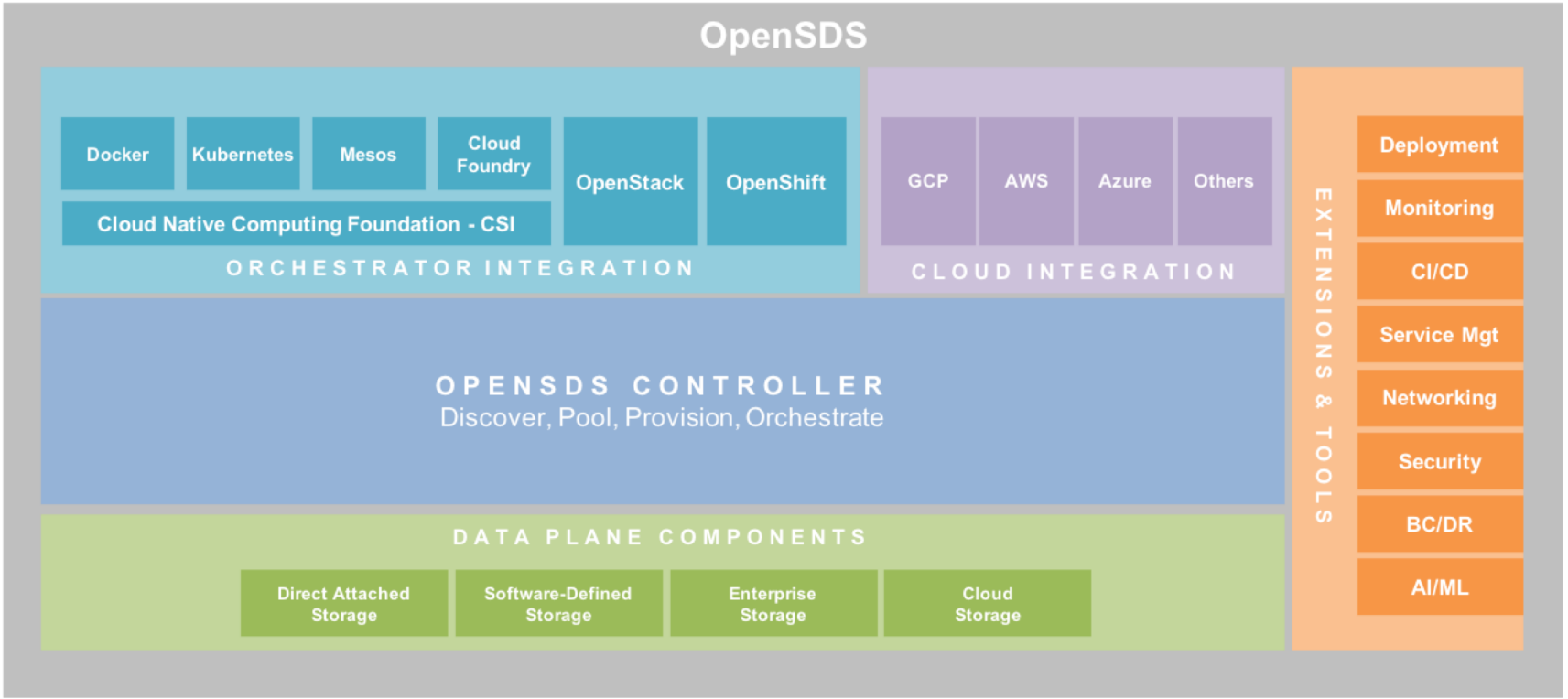
HOTPOT

The Storage Controller Project

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



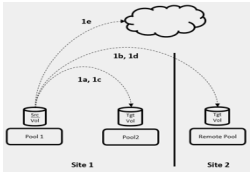
OpenSDS Overview – Project Framework



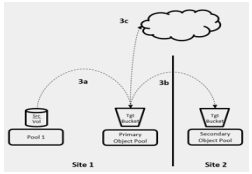
OpenSDS Overview – Connecting Storage and Everything Else

OpenSDS OSCON 2018

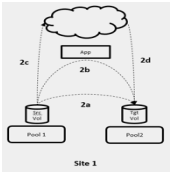
- Storage orchestration



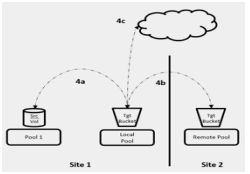
Data Replication



Data Protection

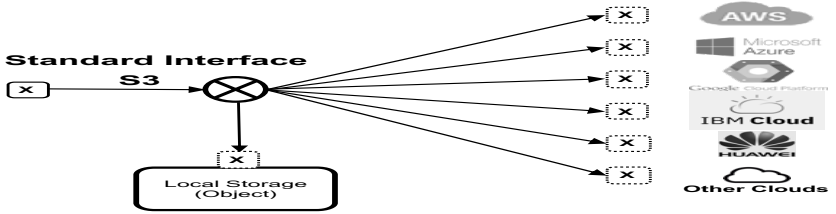


Data Migration



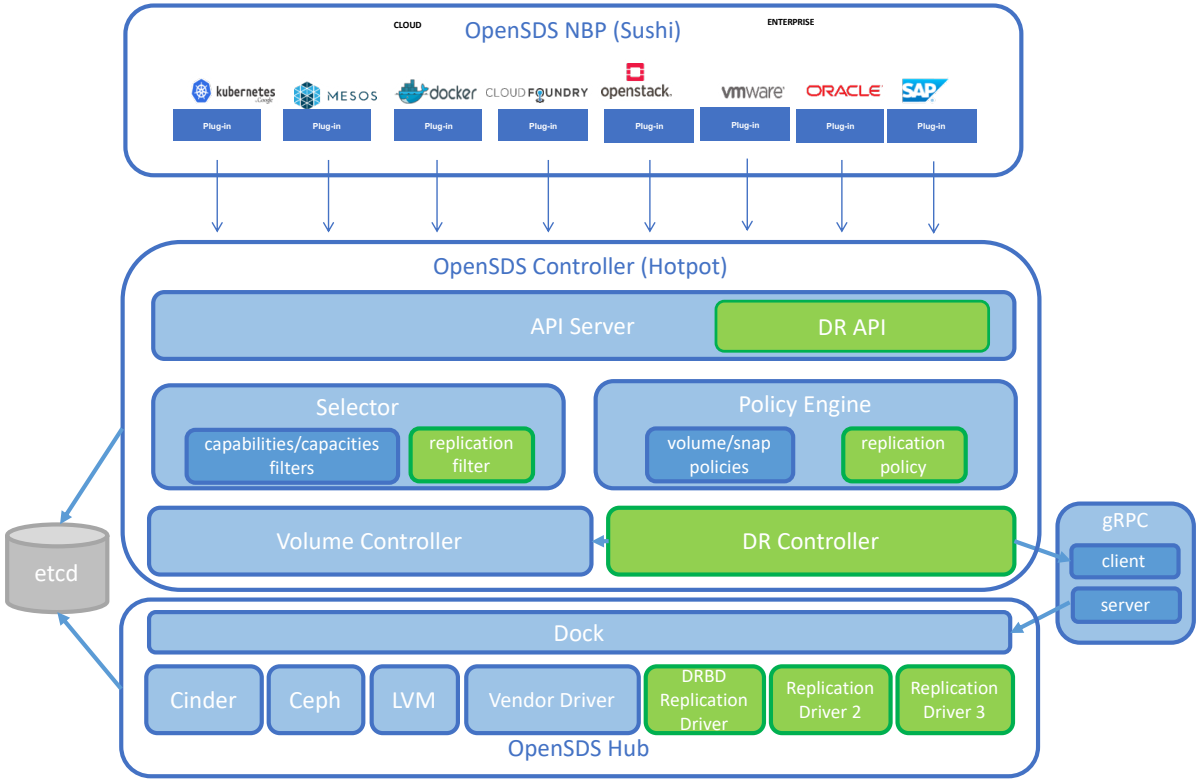
Data Lifecycle

- Storage multi-cloud

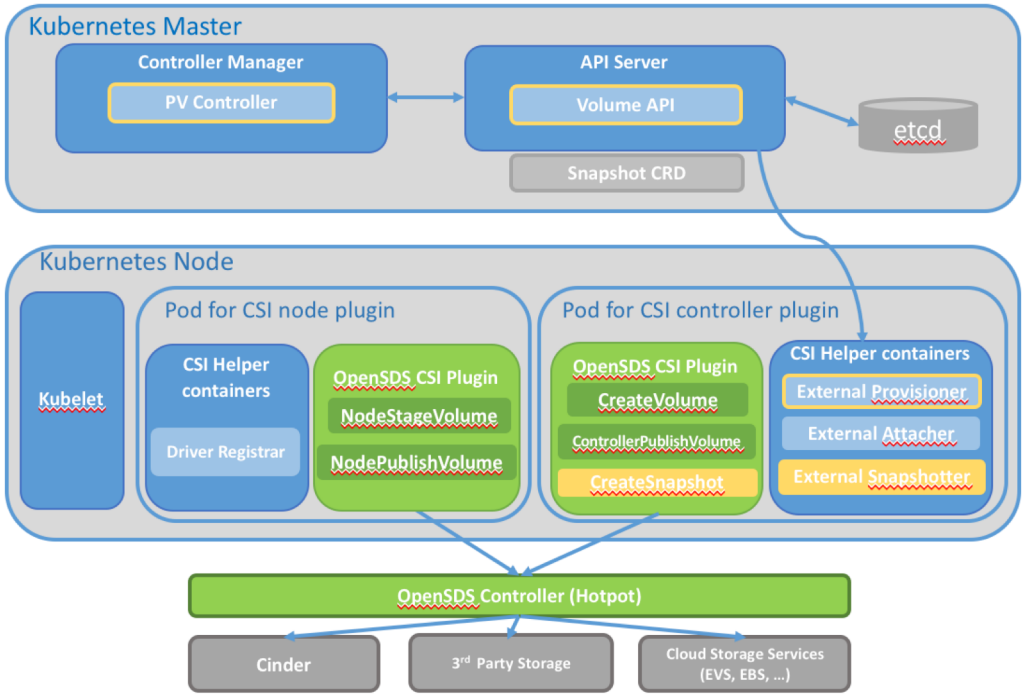


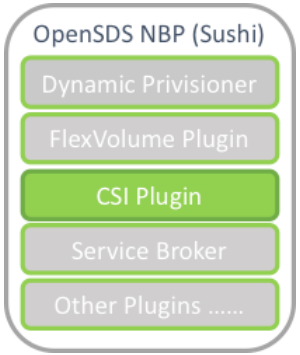
- Storage Intelligence

OpenSDS Overview - Architecture

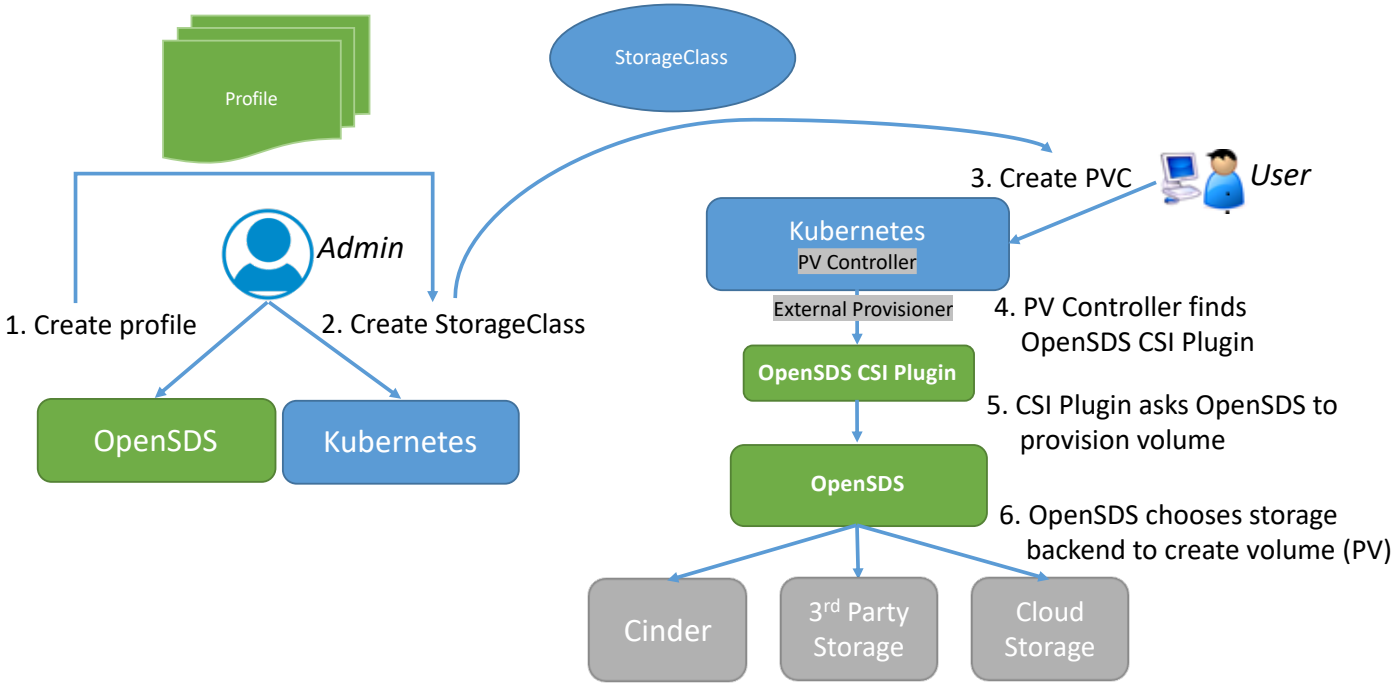


Provision and Manage Persistent Volumes using OpenSDS



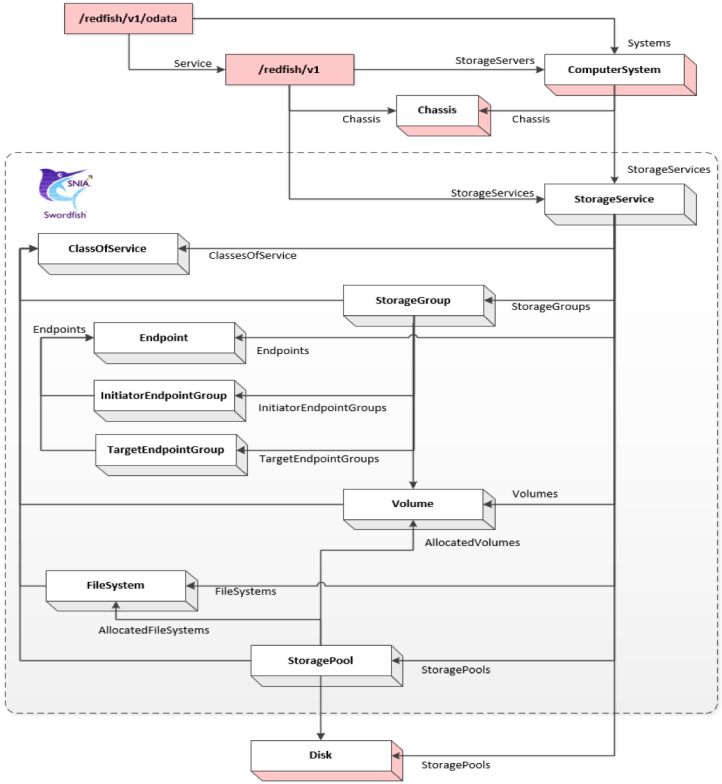


Profiles: Mapping OpenSDS Profile to K8S StorageClass



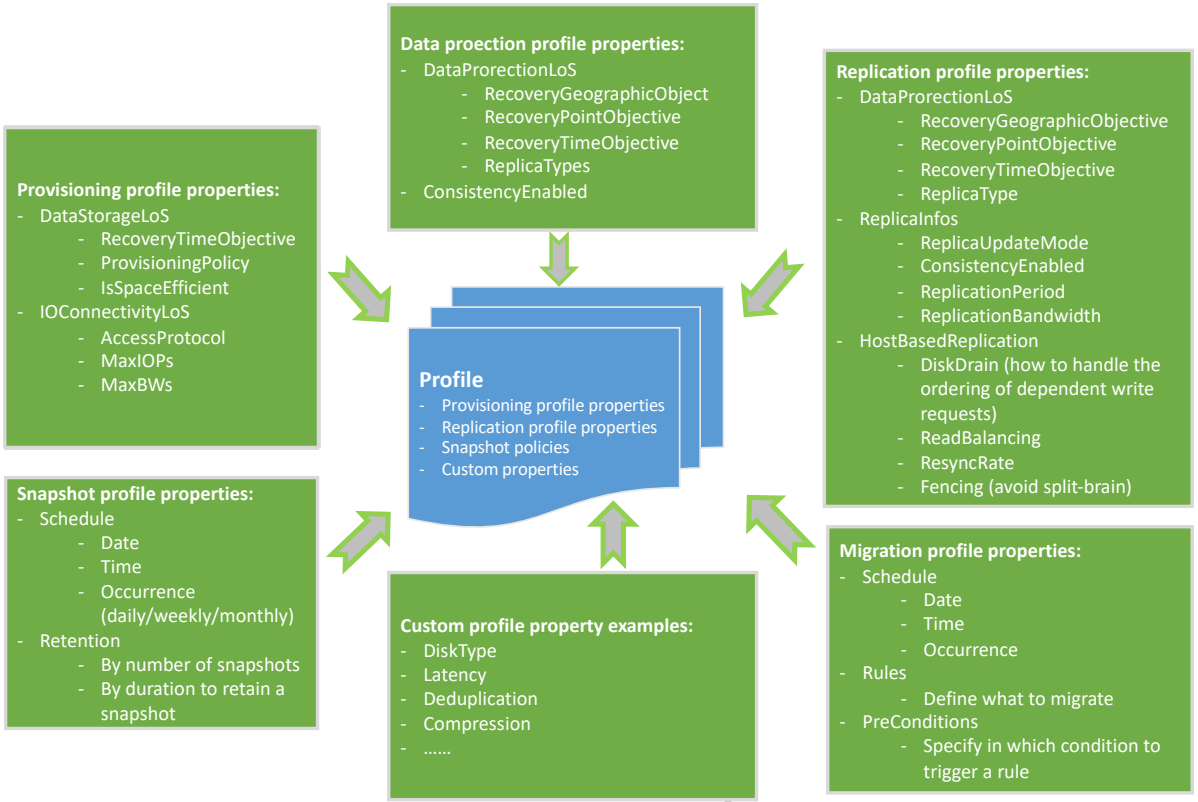
Profiles: Policy Driven SPDM

- 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.

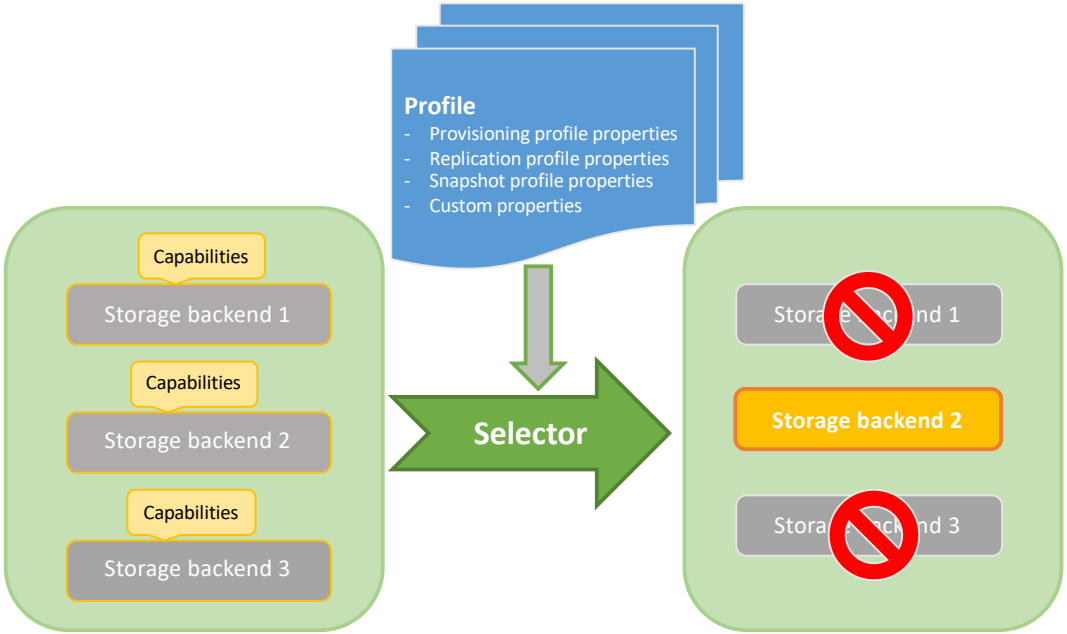


Source: Swordfish_v1.0.5_Specification

Profiles: Definitions



Profiles: Mapping Profiles to Capabilities



Provision: StorageClass with Profile Parameter

OpenSDS OSCON 2018

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
```

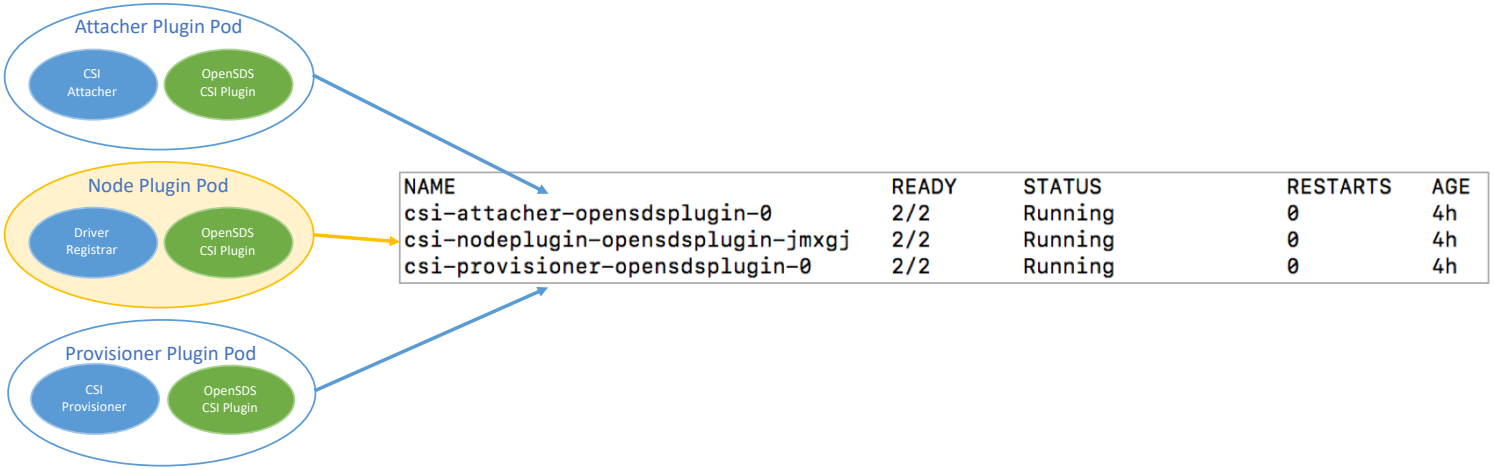
Note: profile parameter can be profile id or name

Provision: 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`:



Provision: 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`

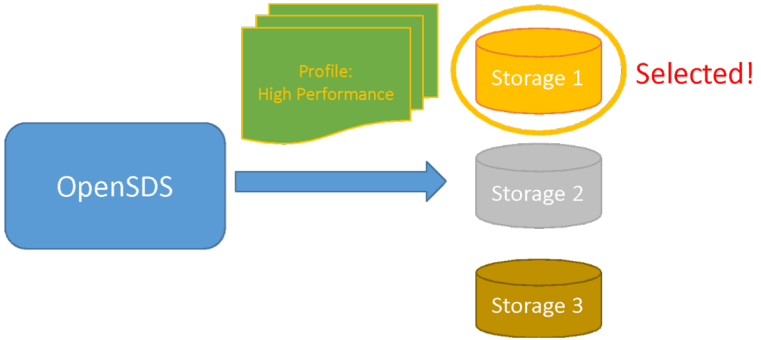
```
root@nginx:/# mount | grep html
/dev/sda on /var/lib/www/html type ext4 (rw,relatime,data=ordered)
```

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
```

Disaster Recovery: Replication Profile

- RecoveryTimeObjective
- RecoveryPointObjective
- RecoveryGeographicObjective
- ReplicaType
  - Mirror
- ReplicationUpdateMode
  - Sync, Async, Active, Adaptive
- ConsistencyEnabled
- ReplicationPeriod
- ReplicationBandwidth



Disaster Recovery: Replication Example

OpenSDS OSCON 2018

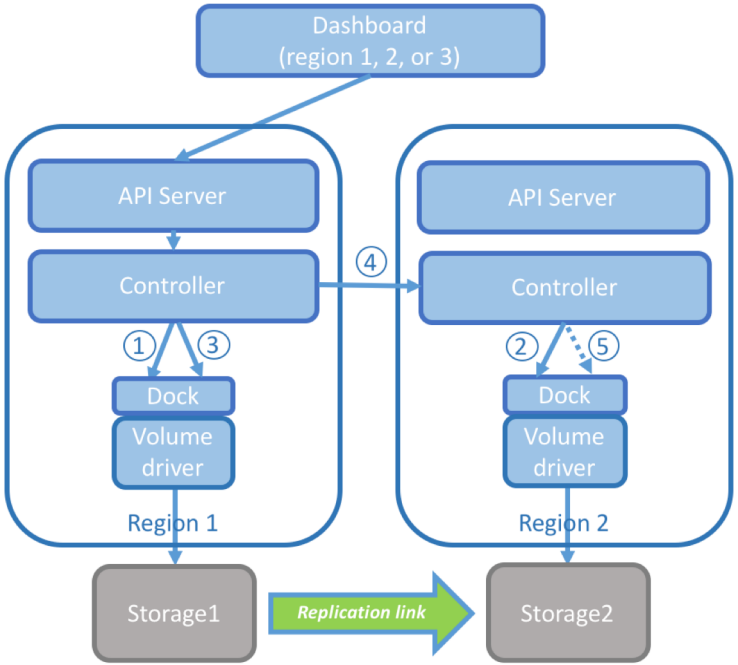
ReplicationSC.yaml

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
    name: opensds-csi-replication-sc
provisioner: csi-opensdsplugin
parameters:
    profile: replication-profile
    enableReplication: “true”
```

ReplicationPVC.yaml

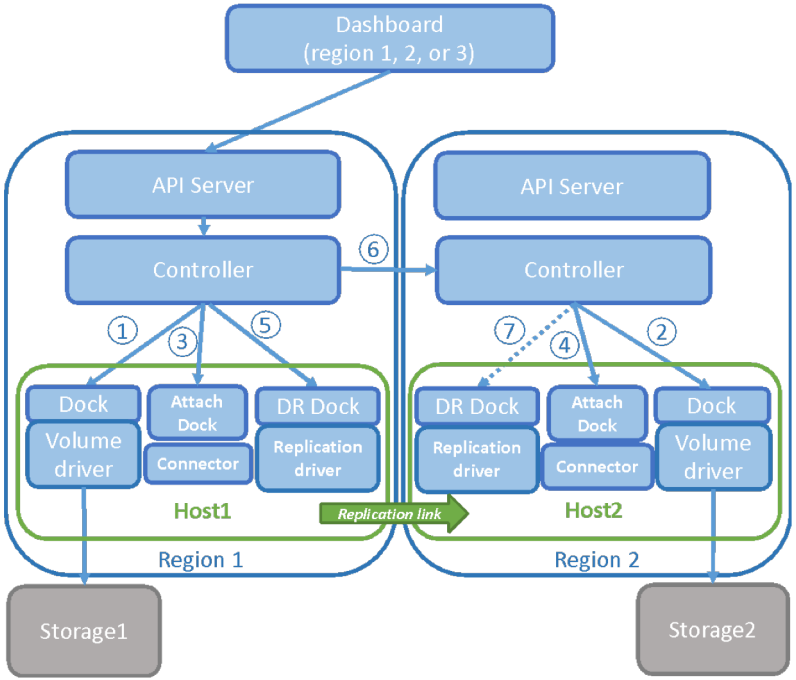
```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
    name: opensds-csi-replication-pvc
spec:
    accessModes:
    - ReadWriteOnce
    resources:
        requests:
            storage: 10Gi
    storageClassName: opensds-csi-replication-sc
```

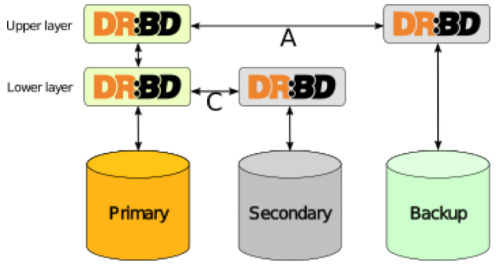
Disaster Recovery: Array-based Replication



- Create source and target volume
- Create replication on the arrays

Disaster Recovery: Host-based Replication





Source: http://docs.linbit.com/docs/users-guide-9.0/

- Create source and target volume
- Attach volumes
- Create replication on the hosts

Disaster Recovery: Replication Functionalities

OSCON 2018

Create Replication:
osdsctl replication create <primary volume id> <secondary volume id> [flags]

Flags:

-d, --description string	the description of created replication
-h, --help	help for create
-n, --name string	the name of created replication
-p, --primary_driver_data string	the primary replication driver data of created replication
-m, --replication_mode string	the replication mode of created replication, value can be sync/async
-t, --replication_period int	the replication period of created replication, the value must be greater than 0
-s, --secondary_driver_data string	the secondary replication driver data of created replication

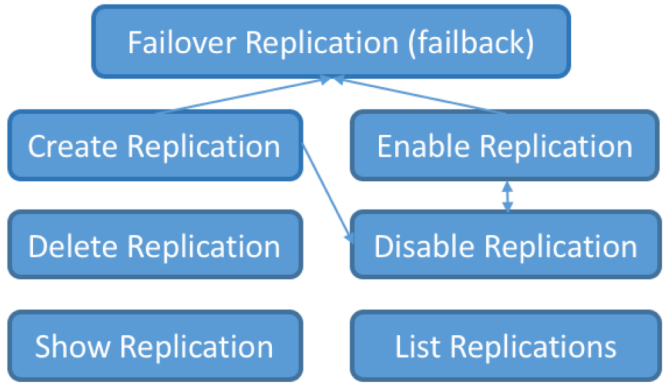
Enable Replication:
osdsctl replication enable <replication id>

Disable Replication:
osdsctl replication disable <replication id>

Failover Replication:
osdsctl replication failover <replication id> [flags]

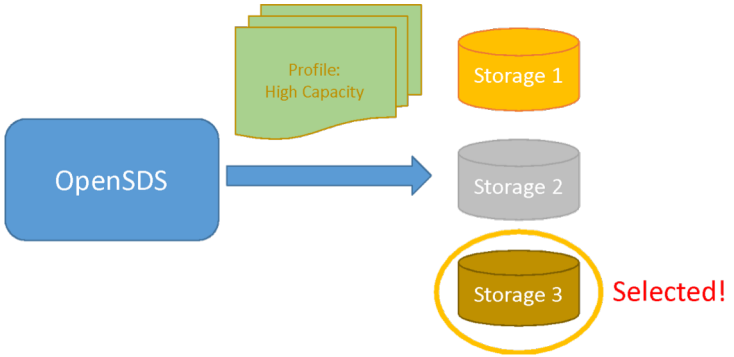
Flags:

-a, --allow_attached_volume	whether allow attached volume when failing over replication
-h, --help	help for failover
-s, --secondary_backend_id string	the secondary backend id of failover replication

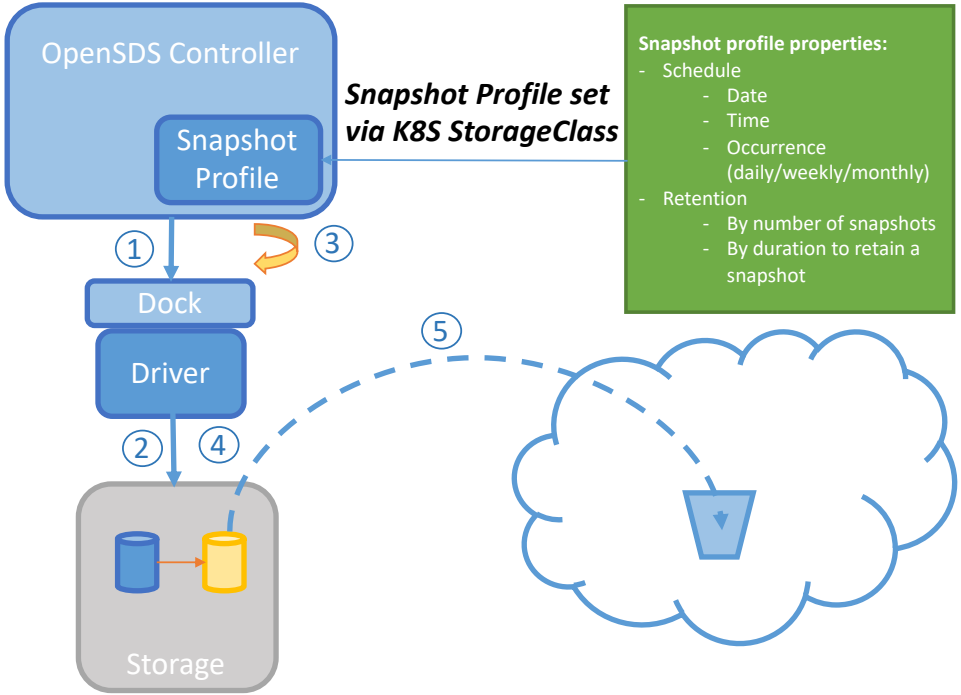


Data Protection: Snapshot Profile

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

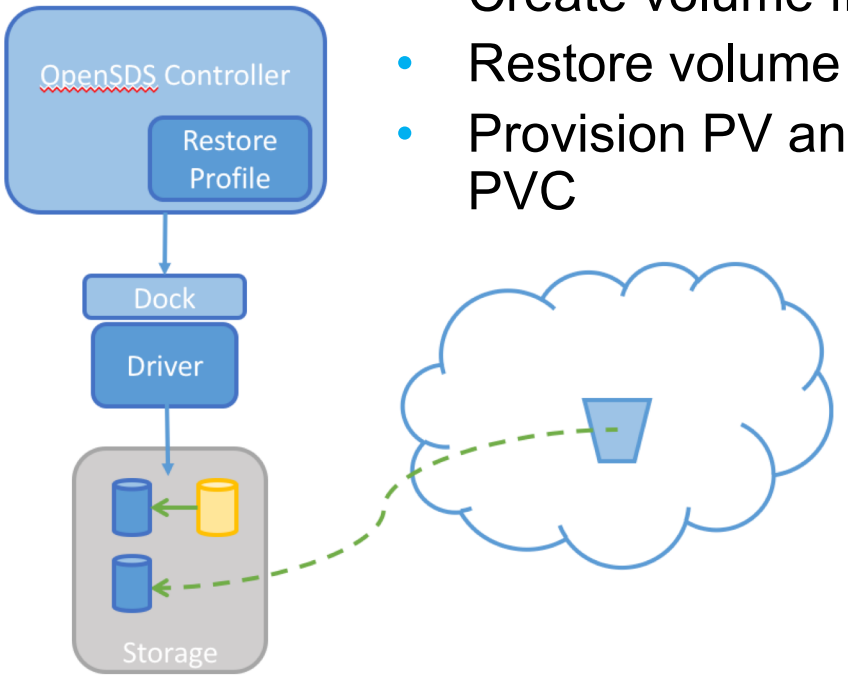


Data Protection: Periodic Snapshotting



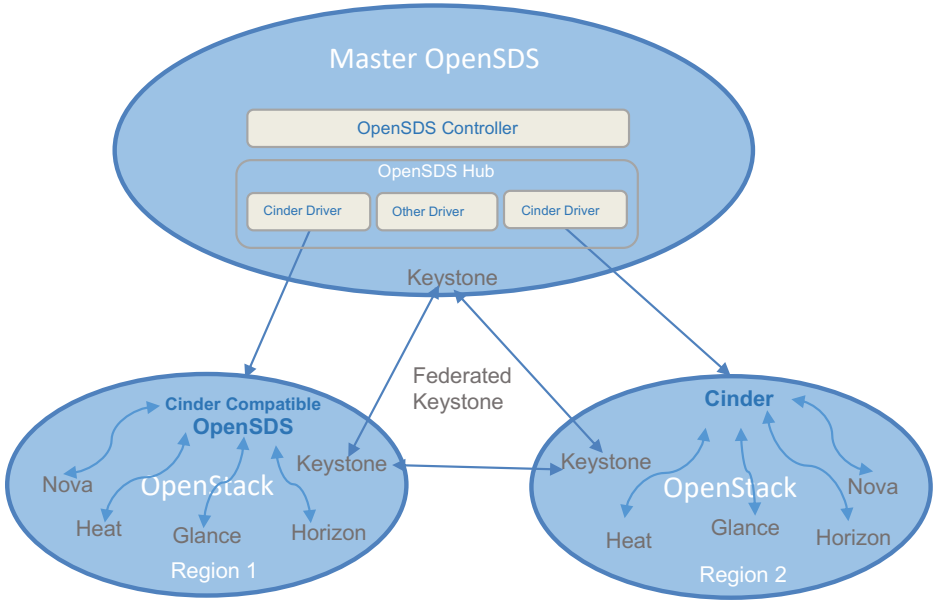
- Take snapshots periodically based on snapshot profile
- Upload snapshots to object store on-premise or in the cloud

Data Protection: Restore



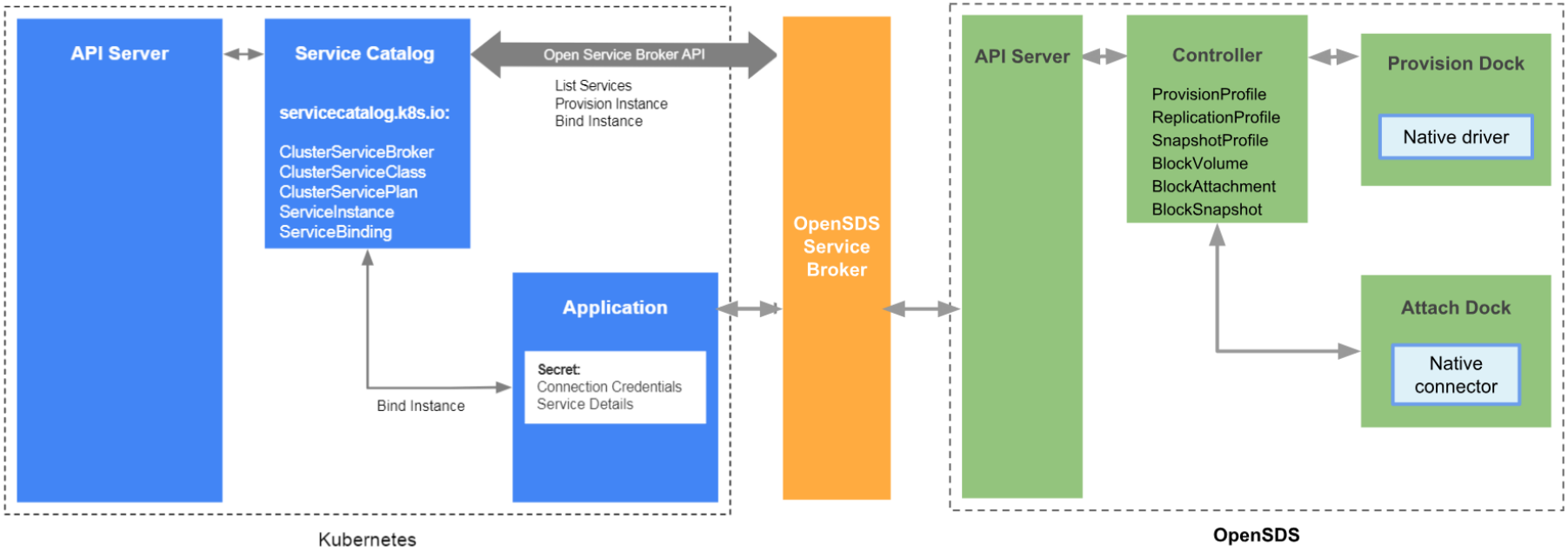
- Create volume from snapshot
- Restore volume from backup
- Provision PV and bind with PVC

OpenStack Integration



- OpenSDS uses Cinder to provision storage
  - OpenSDS southbound volume driver for Cinder
  - Cinder in OpenStack deployment, Cinder standalone, or Cinder lib
- Cinder compatible API adapter
- Multi-OpenStack (Bali)

Service Broker Integration



OpenSDS Roadmap

OpenSDS OSCON 2018

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

Governance

OpenSDS OSCON 2018

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

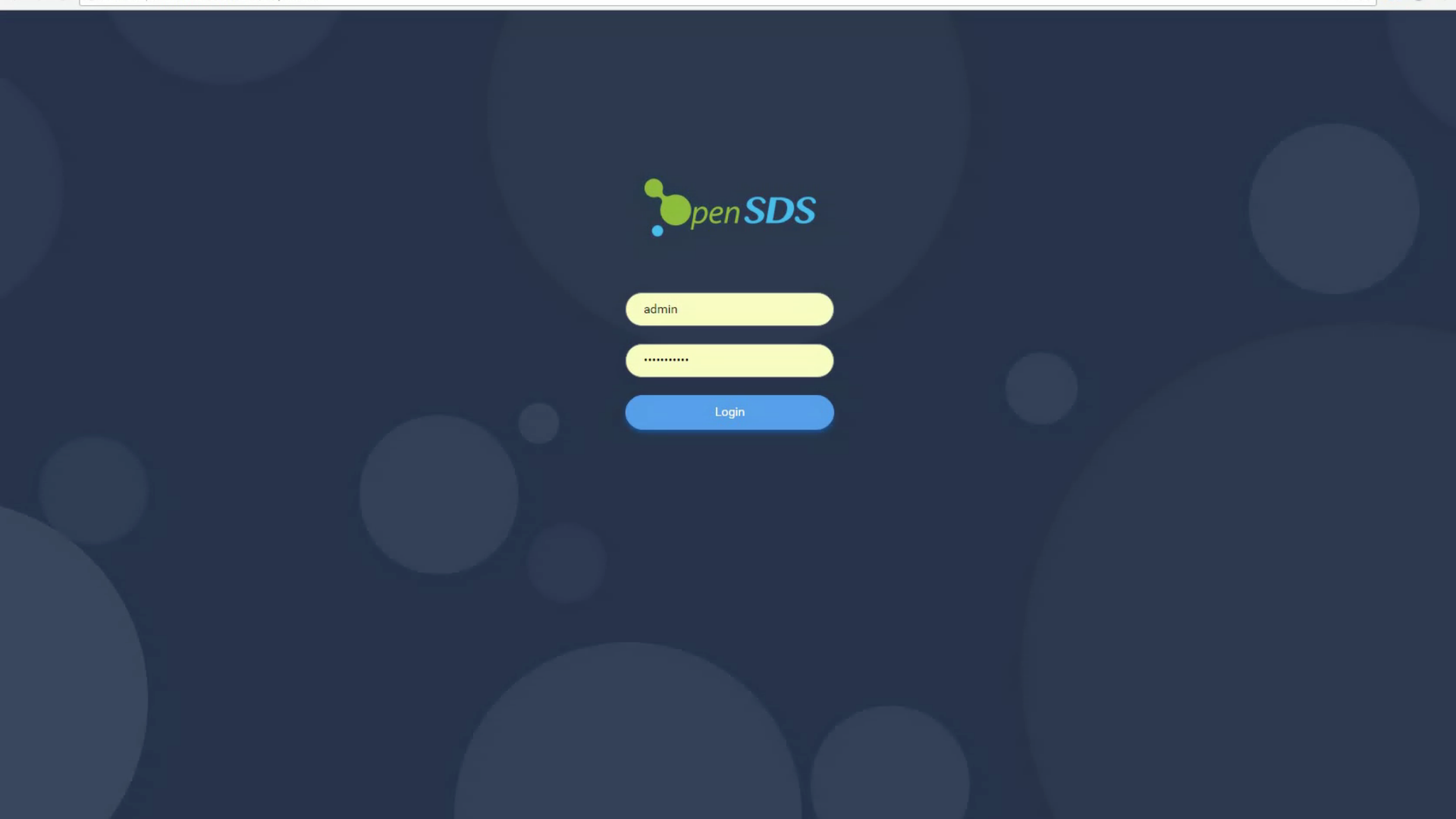
Yusuke Sato
Yahoo Japan, Infrastructure Lead

Kei Kusunoki
NTT Communications, Storage Architect

Yuji Yazawa
Toyota ITC, Group Lead

Demo

- Array-based replication: Failover storage provisioned by OpenSDS CSI plugin


penSDS
admin
Login

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

Thank You

@opensds_io

OpenSDS

OSCON 2018