Container Storage Interface for Kubernetes

Anil Degwekar
AGENDA

• CSI – an overview
• Why CSI?
• CSI in Kubernetes
• The GoCSI framework
• Developing a CSI driver
CSI – an overview

• CSI = Container Storage Interface
  – Standard initially promoted by Container Orchestrator vendors
  – Now adopted widely by the storage industry

• Aims to reduce effort – only one driver for all Container Orchestrators
## CSI Plugin Components

### Controller Service
- CreateVolume
- DeleteVolume
- ListVolumes
- ControllerPublishVolume
- ControllerUnpublishVolume
- ValidateVolumeCapabilities
- GetCapacity
- CreateSnapshot
- DeleteSnapshot
- ListSnapshots
- ControllerGetCapabilities

### Node Service
- NodeStageVolume
- NodeUnstageVolume
- NodePublishVolume
- NodeUnpublishVolume
- NodeGetVolumeStats
- NodeGetInfo
- NodeGetCapabilities

### Identity Service
- GetPluginInfo
- GetPluginCapabilities
- Probe(ProbeRequest)
Why CSI?

- Widely adopted by industry
- Open source
- Making rapid progress – already at rev. 1.0
- Recently added support for snapshots
CSI adoption

<table>
<thead>
<tr>
<th>Container Orchestrator</th>
<th>Version</th>
<th>CSI version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kubernetes</td>
<td>1.10</td>
<td>0.2</td>
</tr>
<tr>
<td></td>
<td>1.13</td>
<td>0.3, 1.0</td>
</tr>
<tr>
<td>OpenShift</td>
<td>3.11</td>
<td>0.2</td>
</tr>
<tr>
<td>PKS</td>
<td>1.4</td>
<td>1.0</td>
</tr>
<tr>
<td>Mesos</td>
<td>1.6</td>
<td>0.2</td>
</tr>
<tr>
<td>Cloud Foundry</td>
<td>2.5</td>
<td>0.3</td>
</tr>
</tbody>
</table>
CSI in Kubernetes

• Kubernetes 1.13 added support for CSI 1.0

• Kubernetes has extra features which complement CSI
  – Storage Classes which provide parameters to the CSI drivers
  – Ability to encrypt credentials via ‘secrets’
  – Multiple CSI drivers can co-exist
  – Can dynamically start the node service on newly spawned nodes
  – Ensures that only one controller service runs at a time
Kubernetes + CSI: How does it work?

Kubernetes Core
- Master
- Node
  - Kubelet

External Components (Kubernetes team)
- Driver registrar
- External provisioner
- External attacher

External Components (array vendor)
- CSI Identity
- CSI Controller
- CSI Node

External storage
Storage Class, PVC and PV

• **Storage Class**
  – Created by administrators
  – Indicates different classes of storage available
  – Maps to QoS, replication, compression, backup or some arbitrary policy
  – Specifies a Name, name of plugin, Reclaim Policy, Volume Binding Mode + some (opaque to Kubernetes) parameters

• **Persistent Volume**
  – Persistent storage required for a container
  – Bound to a PVC

• **Persistent Volume Claim**
  – Claim for a Persistent Volume
  – Specifies a storage class name, access mode, type and size of the volume
The GoCSI framework

• History of GoCSI
  – Started as part of the RexRay open source project
  – Is maintained even today by community of developers
  – Is up to date with CSI 1.0 Specification

• What it offers today
  – Provides common code (gRPC server)
  – Provides skeleton for CSI plugin interfaces implementation
  – CSC: CLI tool which emulates all CSI RPCs

• Drivers using GoCSI
  – Dell EMC VxFlex OS
  – Dell EMC XtremIO
  – Others
Developing a CSI driver

• Step 1: Create driver framework using GoCSI

• Step 2: Develop a Go library for your storage system

• Step 3: Implement Array specific logic in the controller and node plugins
Typical CSI driver architecture

Helm charts for installation

CSI Driver

GoCSI Framework

CSI Controller Plugin

CSI Node Plugin

Go library for storage array
Things to watch out for

- Idempotency
- String limitation (128 characters)
- Performance and scalability considerations
- Kubernetes version vs CSI version
- Storage classes not available in other Container Orchestrators
A peek into what is coming

• Topology
• Block (Raw) access
• Replication
Dell EMC and CSI

• Published multiple CSI drivers
  – VxFlex OS
  – XtremIO

• Others are under development