KubeCon CloudNativeCon Europe 2019

VMware SIG

Introduction to the CSI driver

David vonThenen and Steven Wong
VMware

vmware

Abstract

Hidden slide during presentation – retain for deck publication

The Container Storage Interface (CSI) is a specification designed to enable persistent storage volume management, using a plugin maintained independently of Kubernetes. Kubernetes CSI support recently advanced to GA. In the longer term, there is a plan to deprecate existing legacy storage plugins. New storage-related functionality, such as snapshot support, is now being targeted for CSI only.

The “in-tree” vSphere storage plugin remains fully supported at this time. but users running Kubernetes on vSphere may wish to change to CSI to gain new features. At some point. migration to the CSI plugin is projected to become mandatory.

This session will explain and demonstrate deployment, configuration and use of the new vSphere CSI driver. We will also cover the roadmap for new functionality including snapshots, migration and other topics (e.g. interaction with scheduling and zones).

Speakers

David vonThenen

Cloud Native Engineer
VMware Cloud Native Applications Business Unit

Community participant:

- Kubernetes
- Cloud Provider
- Cluster API

Kubernetes VMware SIG member
Kubernetes Cloud Provider SIG member
Kubernetes Cluster Lifecycle SIG member

Steven Wong

Open Source Community Relations Engineer
VMware Cloud Native Applications Business Unit

Community participant:

- Kubernetes
- Container Storage Interface
- CNCF Storage Working Group

Kubernetes Storage SIG member
Kubernetes VMware SIG chair
Kubernetes IoT and Edge Working Group lead

Agenda

Background - external volume mounts in Kubernetes

Why? How it works

Legacy Storage Plugins (in-tree)

CSI

What is it? Why replace the legacy implementation? roadmap

The CSI Driver for vSphere

Zone support, installation, configuration

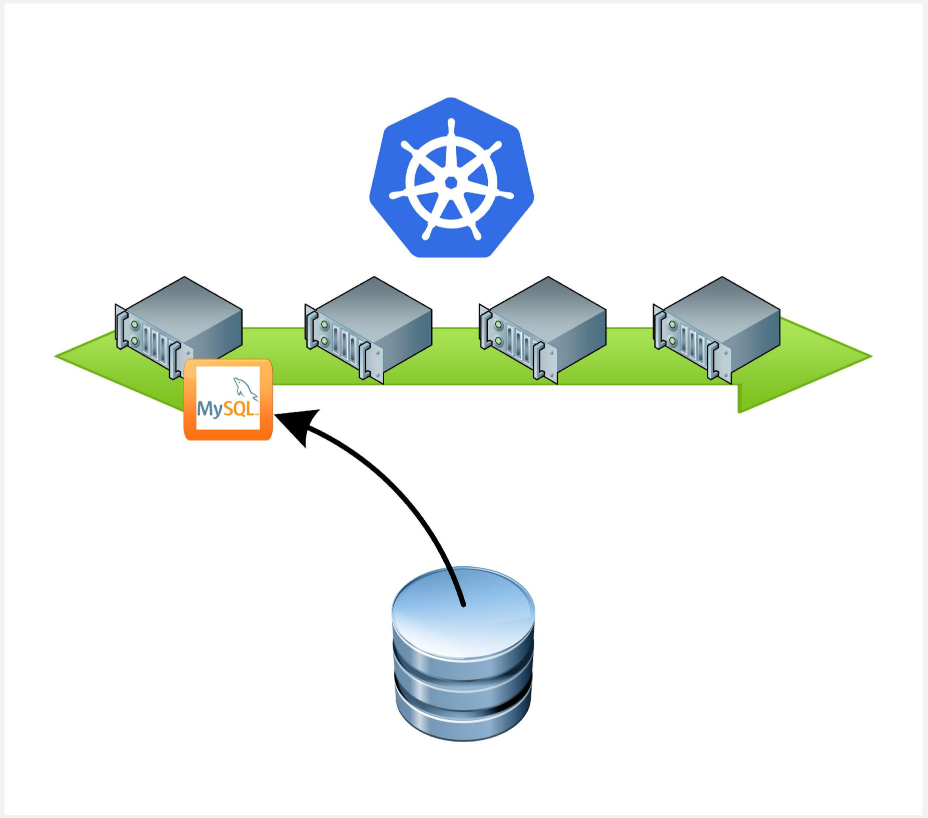
Operation

Best practices, troubleshooting

Migration

External Volume Mounts

How they work, Why the are useful.



Directory, accessible by all containers in pod

Volume Plugins Define

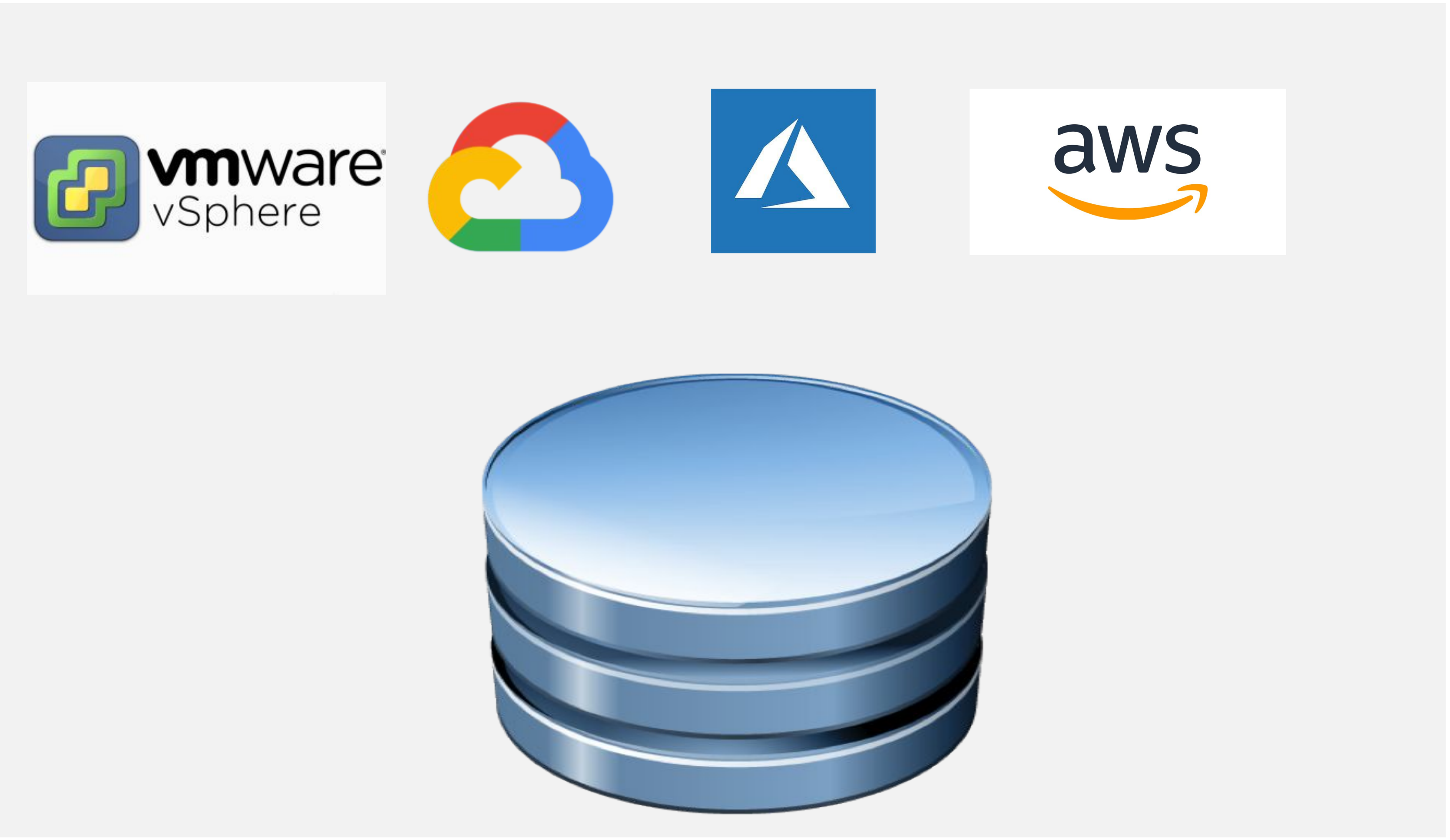
- How directory is setup
- Medium that backs it
- Contents of the directory

Data lifetime can survive beyond the pod, or the host worker node.

Volumes size not limited by physical capacity of a single node.

Kubernetes Persistent Volume abstraction

Enable portable stateful applications and services



identical interface everywhere

Pods consume from storage classes

- Typically described by desired outcome or class of service (e.g. fast, cheap, etc.)
- An administrator can map classes to the most appropriate backing store for the cloud.
- Applications are portable – can be moved unchanged

Kubernetes volume abstraction

Enable portable stateful applications and services

identical interface everywhere

Pods consume from storage classes

- Typically described by desired outcome or class of service (e.g. fast, cheap, etc.)
- An administrator can map classes to the most appropriate backing store for the cloud.
- Applications are portable – can be moved unchanged

Original in-tree design

Issues

Volume plug-ins were either:

Built directly into the Kubernetes release

- Could not be patched or enhanced independent of a full Kubernetes release
- Resulted in undesirable bloat of Kubernetes itself – most users need only a tiny number of the plug-ins, yet all are part of the release.
- Volume plugin code runs as a privileged component of Kubernetes itself – security and stability risk

A “flex driver”

- Run like an exec’d CLI call, limiting feature set, and complicating install

The in-tree version of the vSphere plugin

“Project Hatchway”

Released in 2017, production worthy and stable today
docs: https://vmware.github.io/vsphere-storage-for-kubernetes/documentation/

- Supports a Kubernetes 1.9+ cluster spanning multiple ESXi clusters, Datacenters and vCenters - but storage must be accessible from these
- Supports VMFS, NFS, vSAN, VVOLs
- Supports vSphere 6.5, 6.7

BUT:

- Like all in-tree plug-ins, feature set has been frozen since end of 2018.
- Slated for eventual deprecation

The Container Storage Interface (CSI)

Moved to GA in Kubernetes 1.13

CONTAINER STORAGE INTERFACE

CSI is a cross-orchestrator open source project independent of Kubernetes

- Feature enablement typically follows the path:
  - Enabling a Kubernetes interface definition,
  - Independent implementation by the CSI project and associated implementations of out of tree CSI storage plugins

CSI for vSphere 1st release

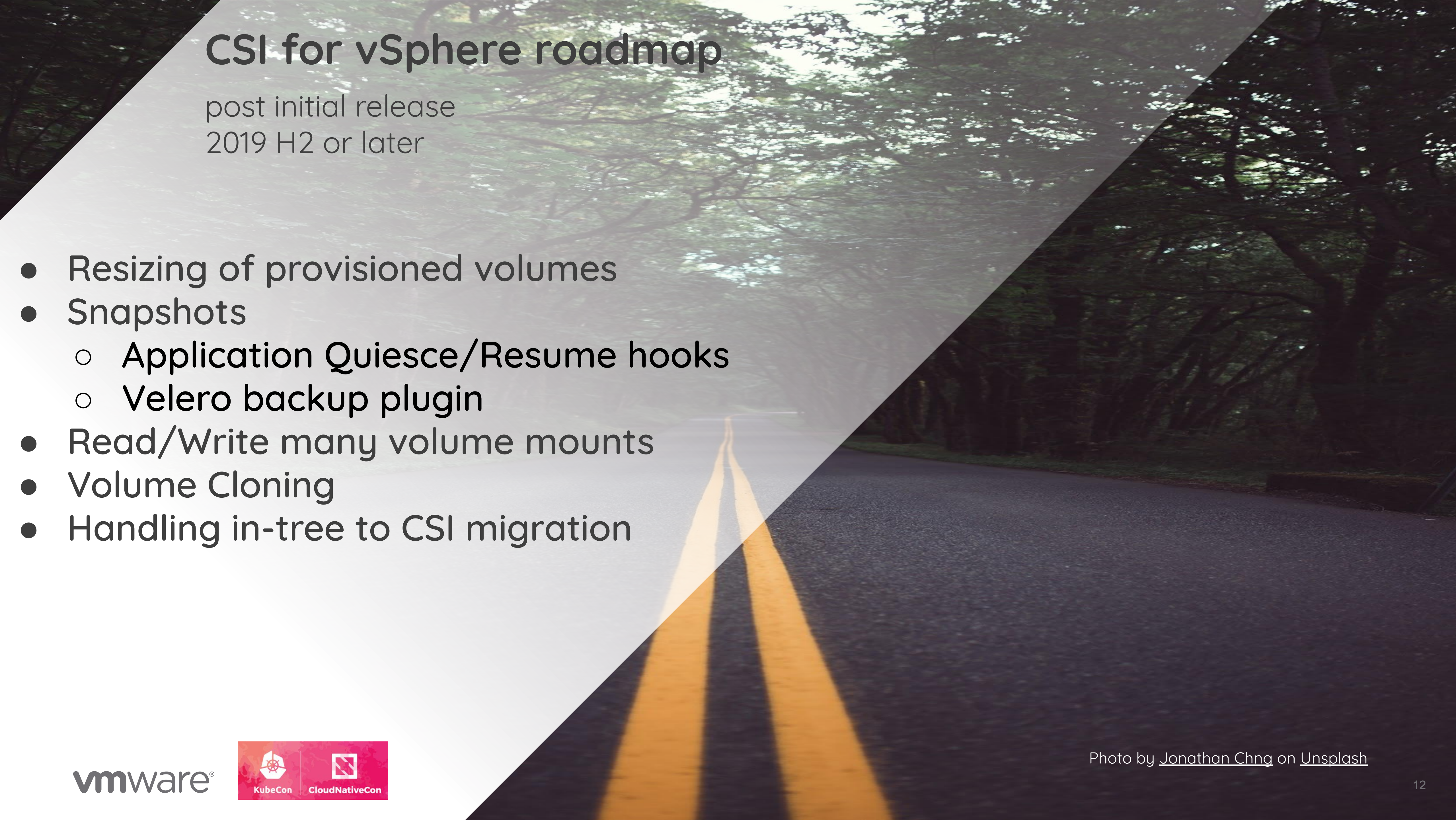
- Alpha - available now (version 0.2.0)
- Beta - June 2019
- GA - July 2019

Features

- VM independent volume management (FCD)
- Kubernetes clusters can straddle mutli-vCenter, multi-Datacenter
- Provision from multiple datastores or datastore clusters
- Conventional + “raw” mounts
- Zone support

https://github.com/kubernetes-sigs/vsphere-csi-driver

vmware® KubeCon CloudNativeCon



CSI for vSphere roadmap

post initial release
2019 H2 or later

- Resizing of provisioned volumes
- Snapshots
  - Application Quiesce/Resume hooks
  - Velero backup plugin
- Read/Write many volume mounts
- Volume Cloning
- Handling in-tree to CSI migration

vmware KubeCon CloudNativeCon

Photo by Jonathan Chng on Unsplash

Operation of the CSI driver for vSphere

Installation, Configuration, Best Practices

Installation

- https://github.com/kubernetes-sigs/vsphere-csi-driver/blob/master/docs/deploying_csi_vsphere_with_rbac.md

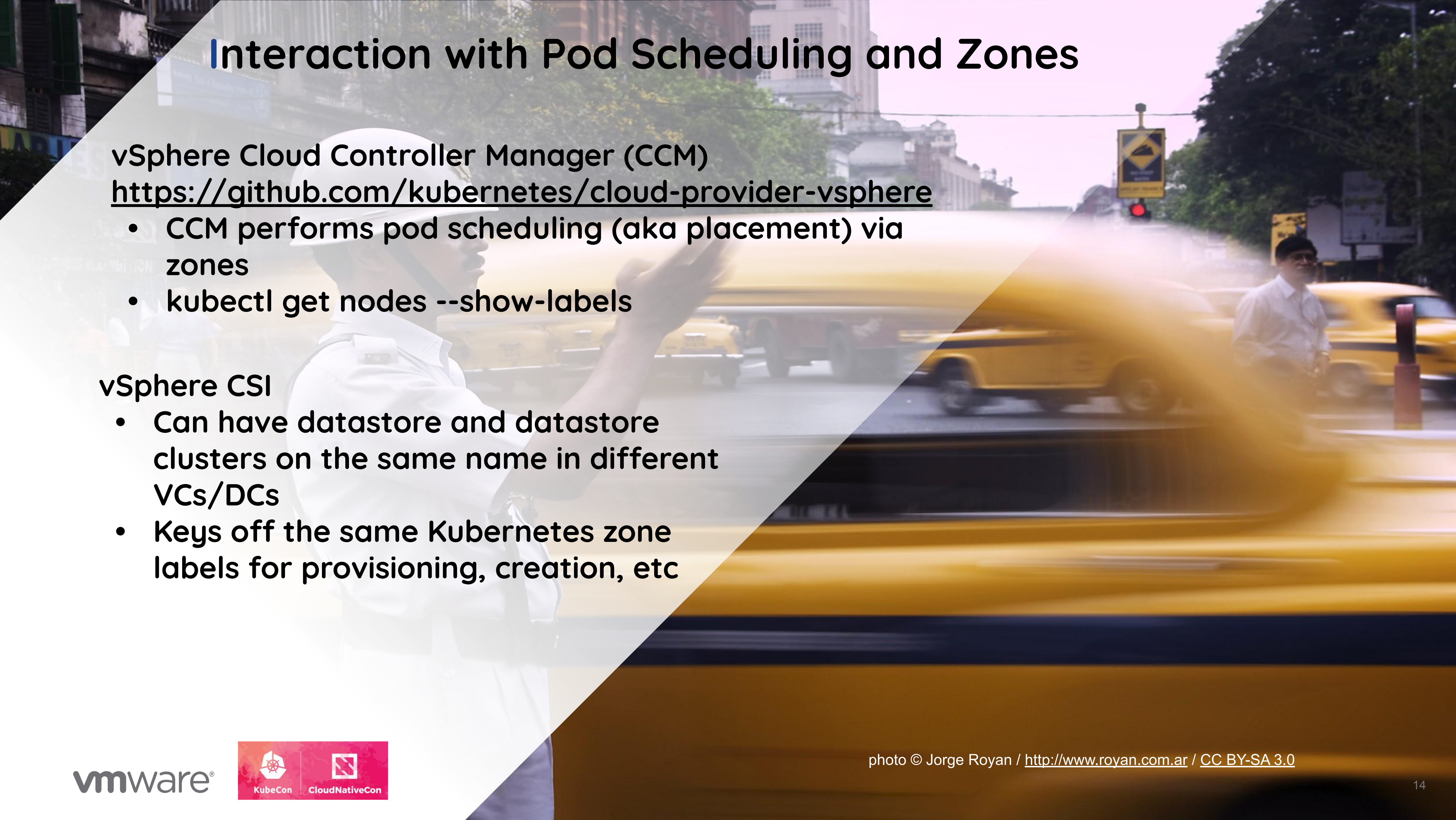
Configuration

- Requires vsphere.conf identical format to in-tree
- Provides YAML within the repo to deploy
- https://github.com/kubernetes-sigs/vsphere-csi-driver/tree/master/manifests/csi

NOTE: CSI Zone Support

- Requires vSphere Cloud Controller Manager
- https://github.com/kubernetes/cloud-provider-vsphere/blob/master/docs/deploying_ccm_and_csi_with_multi_dc_vc_aka_zones.md

Interaction with Pod Scheduling and Zones



vSphere Cloud Controller Manager (CCM)
https://github.com/kubernetes/cloud-provider-vsphere

- CCM performs pod scheduling (aka placement) via zones
- kubectl get nodes --show-labels

vSphere CSI

- Can have datastore and datastore clusters on the same name in different VCs/DCs
- Keys off the same Kubernetes zone labels for provisioning, creation, etc

Troubleshooting the CSI driver for vSphere

Examining logs, where to reach out for assistance

- Getting assistance
  - Kubernetes Slack - #sig-vmware
  - Kubernetes VMware SIG (bi-weekly meetings)
  - GitHub (Issues) - https://github.com/kubernetes-sigs/vsphere-csi-driver

- Debugging, Logs, etc
  - Controller
    - kubectl logs vsphere-csi-controller-0 --namespace=kube-system -c vsphere-csi-controller
  - Node
    - kubectl logs vsphere-csi-node-<random> --namespace=kube-system -c vsphere-csi-node
  - CCM (if using zones)
    - kubectl get nodes --show-labels
    - kubectl logs vsphere-cloud-controller-manager-<random> --namespace=kube-system
- Additional logs
  - CSI sidecar containers, kubelet, and etc

Migration from legacy to CSI for vSphere

Migration from In-Tree to External CSI Implementation

- Migration process is currently in development
  - Will have formal process at the time of GA
  - Additional questions, feedback, input, etc using channels in previous slide
- Migration will be tricky
  - In-Tree used vanilla VMDKs
  - External CSI Driver uses First Class Disks (FCD). Requires vSphere 6.5+
    - Several caveats on feature availability between 6.5 to 6.7u2
  - Full feature support on next vSphere update (6.7u3+)
- Personal recommendations

Thank You

Contacts:

This deck: bit.ly/2JBwMWR

Join SIG VMware

- Slack channel: https://kubernetes.slack.com/messages/sig-vmware
- List: https://groups.google.com/forum/#!forum/kubernetes-sig-vmware
- Zoom meetings (join mailing list group for schedule)

David vonThenen

@dvonthenen

Steven Wong

@cantbewong