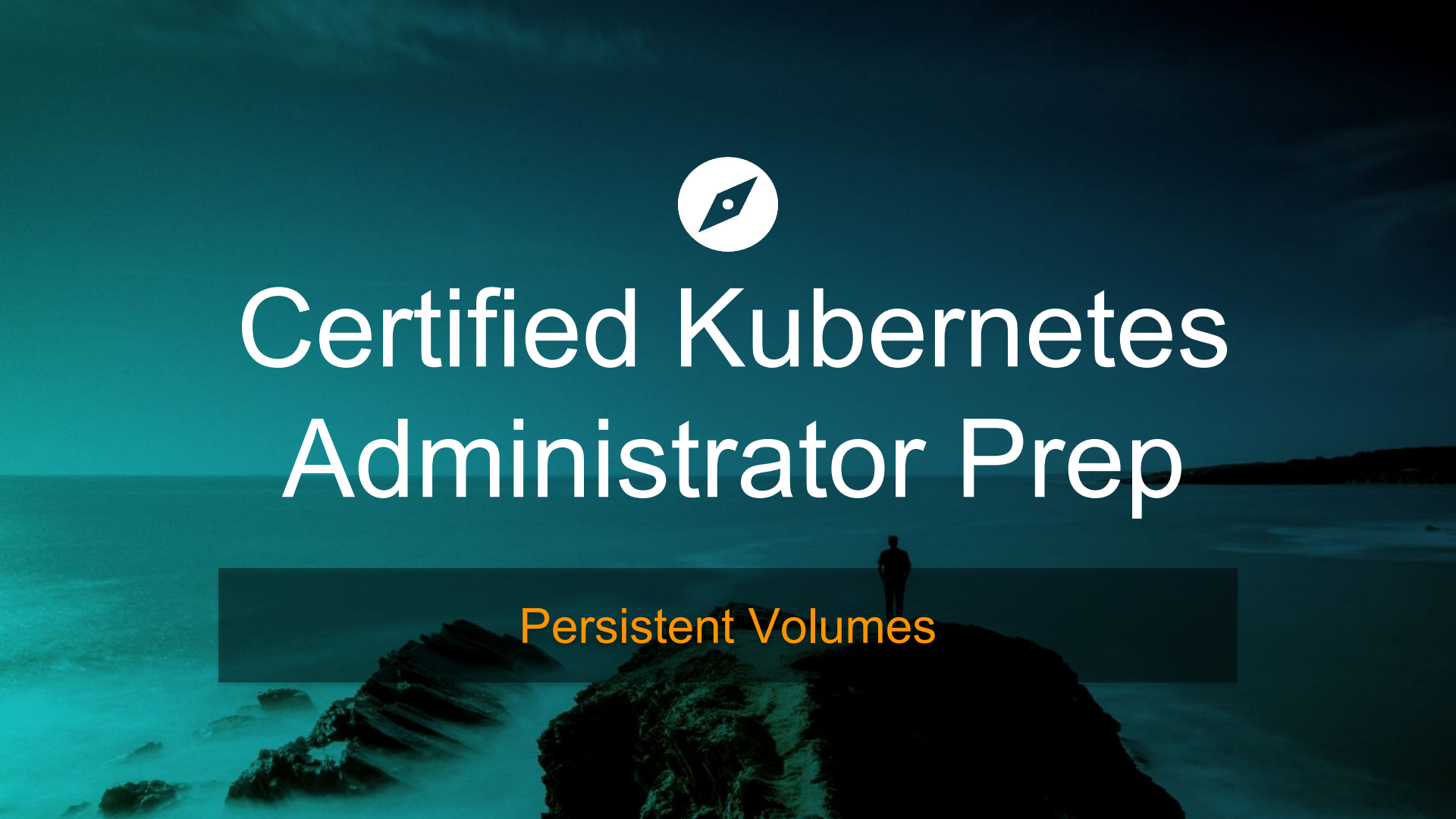

Certified Kubernetes Administrator Prep

Persistent Volumes

Kubernetes and Persistent Volumes

- Native Pod Storage is Ephemeral -- Like a Pod
- What happens when a container crashes:
  - Kubelet restarts it (possibly on another node)
  - File system is re-created from image
  - Ephemeral files are gone
- Docker Volumes
  - Directory on disk
  - Possibly in another container
  - New Volume Drivers

Kubernetes and Persistent Volumes

- Kubernetes Volumes
  - Same lifetime as its pod
  - Data preserved across container restarts
  - Pod goes away -> Volume goes away

Kubernetes and Persistent Volumes

- Kubernetes Volumes
  - Same lifetime as its pod
  - Data preserved across container restarts
  - Pod goes away -> Volume goes away
  - Directory with data
  - Accessible to containers in a pod
  - Implementation details determined by volume types

Kubernetes and Persistent Volumes

- Using Volumes
  - Pod spec indicates which volumes to provide for the pod (spec.volumes)
  - Pod spec indicates where to mount these volumes in containers (spec.containers.volumeMounts)
  - Seen from the container's perspective as the file system
  - Volumes cannot mount onto other volumes
  - No hard links to other volumes
  - Each pod must specify where each volume is mounted

Kubernetes Supported Volume Types

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

awsElasticBlockStore

- Mounts an AWS EBS volume to a pod
- EBS volume is preserved when unmounted
- Must be created prior to use
- Nodes must be on AWS EC2 instances in the same region
- Single instance mounting only

Created via a command like:

aws ec2 create-volume --availability-zone=eu-west-la --size=10 --volume-type=gp2

awsElasticBlockStore Example YAML

```
apiVersion: v1
kind: Pod
metadata:
  name: test-ebs
spec:
  containers:
  - image:
k8s.gcr.io/test-webserver
    name: test-container
    volumeMounts:
    - mountPath: /test-ebs
      name: test-volume
  volumes:
  - name: test-volume
    # This AWS EBS volume must
    # already exist.
    awsElasticBlockStore:
      volumeID: <volume-id>
      fsType: ext4
```

azureDisk and azureFile

An azureDisk is used to mount a Microsoft Azure Data Disk into a pod.

An azureFile is used to mount a Microsoft Azure File Volume (SMB 2.1 and 3.0) into a pod.

cephfs

- Allows mounting a CephFS volume to a pod
- Contents of volume are preserved when unmounted
- Must have a Ceph server running
- Share must be exported

csi

- Container Storage Interface
- In-tree CSI volume plugin for volumes on the same node
- Kubernetes 1.9+
  - --feature-gates=CSIPersistentVolume=true
- Metadata fields specify what is used and how
- Driver fields specify the name of the driver
- volumeHandle identifies volume name
- readOnly is supported

downwardAPI

- Mounts a directory and writes data in plan text files

emptyDir

- Created when a pod is assigned to a node
- Exists while pod runs on a particular node
- Initially empty
- Multiple containers can read/write same volume
- Volume can be mounted per container -- same or different mount points
- Pod removed -> volume removed
- Stored on node's local medium
- Optional - set emptyDir.medium = Memory for RAM based tmpfs

emptyDir Sample YAML

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
  - image: k8s.gcr.io/test-webserver
    name: test-container
    volumeMounts:
    - mountPath: /cache
      name: cache-volume
```

```
volumes:
- name: cache-volume
  emptyDir: {}
```

fc (fibre channel)

- Allows existing fc volume to be mounted to a pod
- Single or Multiple Target using targetWWNs
- FC SAN Zoning must be allocated
- LUNs must be masked to target WWN

flocker

- Open source clustered container data volume manager
- Management and orchestration of volumes
- Allows Flocker dataset to be mounted into a pod
- Must be created prior to mounting
- Can be transferred between pods
- Must have Flocker

gcePersistentDisk

- Mounts a GCE persistent disk to pod
- Data preserved if volume unmounted
- Nodes must be on GCE VMs
- Same project and zone as the persistent disk(s)
- Multiple concurrent mounts allowed, but read-only
- Create with a command like:

```
gcloud compute disks create --size=500GB --zone=us-central1-a my-k8s-disk
```

gcePersistentDisk Sample YAML

```
apiVersion: v1
kind: Pod
metadata:
  name: test-pd
spec:
  containers:
  - image: k8s.gcr.io/test-webserver
    name: test-container
    volumeMounts:
    - mountPath: /test-pd
      name: test-volume
  volumes:
```

```
- name: test-volume
  # This GCE PD must already
  #  exist.
  gcePersistentDisk:
    pdName: my-data-disk
    fsType: ext4
```

gitRepo

- Mounts emptyDir and clones a git repository
- YAML Sample:

```
...
  volumes:
  - name: git-volume
    gitRepo:
      repository: "git@somewhere:me/my-git-repository.git"
      revision: "22f1d8406d464b0c0874075539c1f2e96c253775"
```

glusterfs

- Allows GlusterFS volume to be mounted to a pod
- Volume data preserved if volume is unmounted
- Multiple concurrent mounts -- read/write -- are allowed
- Must have GlusterFS

hostPath

- Mounts file or directory from host node's filesystem to a pod
- Field type - Empty string (for backward compatibility) performs no checks
- DirectoryOrCreate -- Created if not present
- Directory -- Directory must exist
- FileOrCreate -- Created if not present
- File -- File must exist
- Socket -- Socket must exist
- CharDevice -- Character device must exist
- BlockDevice -- Block device must exist

hostPath, Warnings!

- hostPath might behave differently on different nodes regardless of pod configuration
- Files and directories created on the host are only writable by root

hostPath Sample YAML

```
...
  volumes:
  - name: test-volume
    hostPath:
      # directory location on host
      path: /data
      # this field is optional
      type: Directory
```

iSCSI

- Allows an existing iSCSI volume to be mounted to a pod
- Volume data preserved if volume is unmounted
- Must have an iSCSI provider
- Multiple read only concurrent connections allowed
- Only one writer at a time

local

- Alpha, requires “PersistentLocalVolumes” and “VolumeScheduling” feature gates
- Allows local mounted storage to be mounted to a pod
- Statically created PersistentVolume
- Kubernetes is aware of the volume’s node constraints
- Must be on the node
- Not suitable for all applications
- 1.9+ - Volume binding can be delayed until pod scheduling

nfs

- Allows existing NFS share to be mounted to a pod
- Volume data preserved if volume unmounted
- Must have an NFS server

persistentVolumeClaim

- Used to mount a PersistentVolume into a pod
- Users can stake a claim to durable storage without knowing implementation details

projected

- Volume types currently projected (subject to expansion!):
  - secret
  - downwardAPI
  - configMap

projected Sample YAML

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: volume-test
spec:
  containers:
  - name: container-test
    image: busybox
    volumeMounts:
    - name: all-in-one
      mountPath: "/projected-volume"
      readOnly: true
```

```yaml
volumes:
- name: all-in-one
  projected:
    sources:
    - secret:
        name: mysecret
        items:
          - key: username
            path: my-group/my-username
```

projected Sample YAML

```
downwardAPI:
 items:
   - path: "labels"
     fieldRef:
       fieldPath: metadata.labels
   - path: "cpu_limit"
     resourceFieldRef:
       containerName: container-test
       resource: limits.cpu
```

```
configMap:
  name: myconfigmap
  items:
    - key: config
      path: my-group/my-config
```

portworxVolume

- Elastic block storage layer
- Storage on a server
- Capabilities tiers
- Aggregates capacity
- Runs in-guest in VMs or on bare metal Linux nodes
- Can be created dynamically or pre-provisioned

rbd

- Allows Rados Block Device volume to be mounted to a pod
- Volume data preserved when volume is unmounted
- Ceph cluster is required
- Multiple concurrent read-only connections allowed
- Single writer only

scaleIO

- Software-based storage platform
- Allows ScaleIO volumes to be mounted to pods
- Must have existing ScaleIO cluster
- Volumes must be pre-created

secret

- Used to pass sensitive information to pods
- Stored using the Kubernetes API
- Mount secrets as files for use by pods
- Volumes are backed by tmpfs so secrets are never written to non-volatile storage
- Secrets must be created in Kubernetes API prior to use

storageos

- Allows existing StorageOS volume to be mounted to a pod
- Runs as a container in the K8s environment
- Data can be replicated
- Provisioning and compression can improve utilization and reduce cost
- Provides block storage to containers via file system
- Requires 64-bit Linux
- Free Developer License available!
- StorageOS container must run on each node that accesses StorageOS volumes and contributes capacity

vsphereVolume

- Kubernetes with vSphere Cloud Provider must be configured
- Used to mount vSphere VMDK to a pod
- Volume data is preserved when volume is unmounted
- Supports both VMFS and VSAN
- Can share subvolumes

FlexVolume plugin

- For when storage vendors create custom plugins without adding it to the K8s repo
- Enables users to mount vendor volumes to a pod
- Vendor plugin implemented using a driver
- Drivers must be installed in correct path on each node

Mount Propagation

- Alpha feature as of Kubernetes 1.8
- Allows for sharing volumes mounted by one container to other containers in the same pod
- Other pods in the same node

Mount Propagation

- Alpha feature as of Kubernetes 1.8
- Allows for sharing volumes mounted by one container to other containers in the same pod
- Other pods in the same node
- --feature-gates MountPropagation=true
- mountPropagation subfield:
  - HostToContainer - Container gets subsequent mounts to this volume (default)
  - Bidirectional - HostToContainer, plus host sees subsequent mounts made by container

Mount Propagation

- Might need this on a pod using FlexVolume driver
- Can be dangerous! (privileged containers only)
- Familiarity with Linux Kernel Behavior strongly recommended!
- Any volume mounts created by containers in pods must be unmounted by the containers upon termination.

Conclusion

- Might need this on a pod using FlexVolume driver
- Can be dangerous! (privileged containers only)
- Familiarity with Linux Kernel behavior strongly recommended!
- Any volume mounts created by containers in pods must be unmounted by the containers upon termination.