BATCH	:	Batch 85
LESSON	:	**Kubernetes**
DATE	:	14.11.2022
SUBJECT	:	**Kubernetes Volumes**

techproeducation

techproeducation

techproeducation

techproeducation

techproedu

techproeducation.com info@techproeducation.com +1 (917) 768-7466

VOLUMES

Volumes

Persistent Volumes

Storage

Pods by themselves are useful, but many workloads require exchanging data between containers, or persisting some form of data.

For this we have Volumes, PersistentVolumes, PersistentVolumeClaims, and StorageClasses.

Volumes

- → Storage that is tied to the Pod’s Lifecycle.
- → A pod can have one or more types of volumes attached to it.
- → Can be consumed by any of the containers within the pod.
- → Survive Pod restarts; however their durability beyond that is dependent on the Volume Type.

Volume Types

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

🟦 Persistent Volume Supported

Volumes

→ `volumes:` A list of volume objects to be attached to the Pod. Every object within the list must have it's own unique `name`.

→ `volumeMounts:` A container specific list referencing the Pod volumes by `name`, along with their desired `mountPath`.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: volume-example
spec:
  containers:
  - name: nginx
    image: nginx:stable-alpine
    volumeMounts:
    - name: html
      mountPath: /usr/share/nginx/html
      ReadOnly: true
  - name: content
    image: alpine:latest
    command: ["/bin/sh", "-c"]
    args:
      - while true; do
          date >> /html/index.html;
          sleep 5;
        done
    volumeMounts:
    - name: html
      mountPath: /html
  volumes:
  - name: html
    emptyDir: {}
```

Volumes

- → volumes: A list of volume objects to be attached to the Pod. Every object within the list must have it's own unique name.
- → volumeMounts: A container specific list referencing the Pod volumes by name, along with their desired mountPath.

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: volume-example
spec:
  containers:
  - name: nginx
    image: nginx:stable-alpine
    volumeMounts:
    - name: html
      mountPath: /usr/share/nginx/html
      ReadOnly: true
  - name: content
    image: alpine:latest
    command: ["/bin/sh", "-c"]
    args:
      - while true; do
          date >> /html/index.html;
          sleep 5;
        done
    volumeMounts:
    - name: html
      mountPath: /html
  volumes:
  - name: html
    emptyDir: {}
```

Volumes

→ volumes: A list of volume objects to be attached to the Pod. Every object within the list must have it's own unique name.

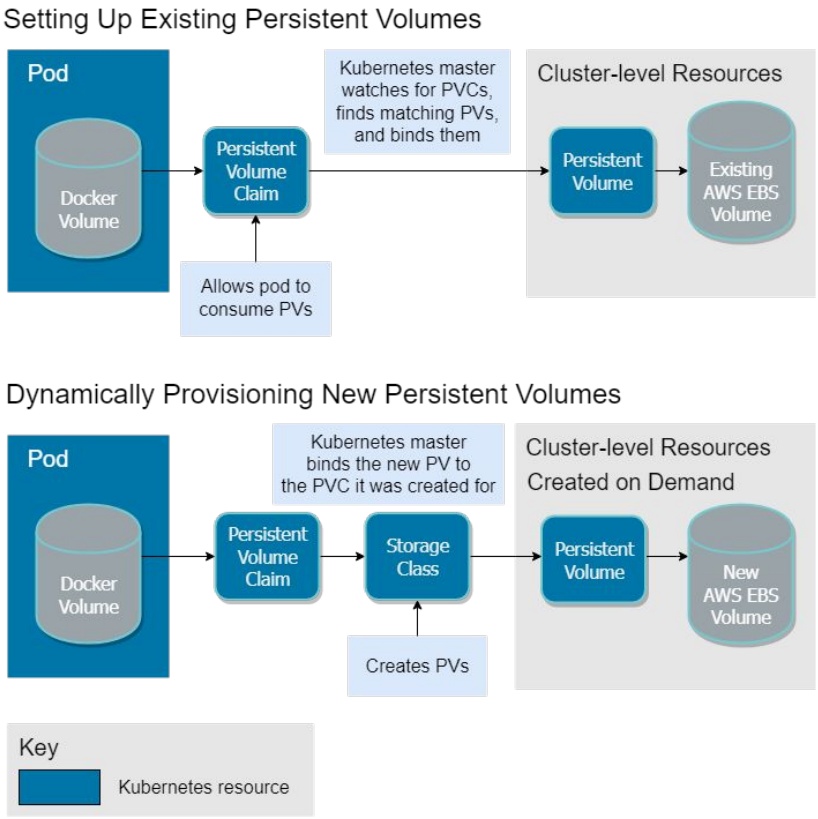
→ volumeMounts: A container specific list referencing the Pod volumes by name, along with their desired mountPath.

```
apiVersion: v1
kind: Pod
metadata:
  name: volume-example
spec:
  containers:
  - name: nginx
    image: nginx:stable-alpine
    volumeMounts:
    - name: html
      mountPath: /usr/share/nginx/html
      ReadOnly: true
  - name: content
    image: alpine:latest
    command: ["/bin/sh", "-c"]
    args:
      - while true; do
          date >> /html/index.html;
          sleep 5;
        done
    volumeMounts:
    - name: html
      mountPath: /html
  volumes:
  - name: html
    emptyDir: {}
```

Persistent Volumes

- A PersistentVolume (PV) represents a storage resource.
- PVs are a cluster wide resource linked to a backing storage provider: NFS, GCEPersistentDisk, RBD etc.
- Generally provisioned by an administrator.
- Their lifecycle is handled independently from a pod
- CANNOT be attached to a Pod directly. Relies on a PersistentVolumeClaim

Persistent Volume Types



Setting Up Existing Persistent Volumes

Pod

Docker Volume

Persistent Volume Claim

Allows pod to consume PVs

Kubernetes master watches for PVCs, finds matching PVs, and binds them

Cluster-level Resources

Persistent Volume

Existing AWS EBS Volume

Dynamically Provisioning New Persistent Volumes

Pod

Docker Volume

Persistent Volume Claim

Storage Class

Creates PVs

Kubernetes master binds the new PV to the PVC it was created for

Cluster-level Resources Created on Demand

Persistent Volume

New AWS EBS Volume

Key

🟦 Kubernetes resource

PersistentVolumeClaims

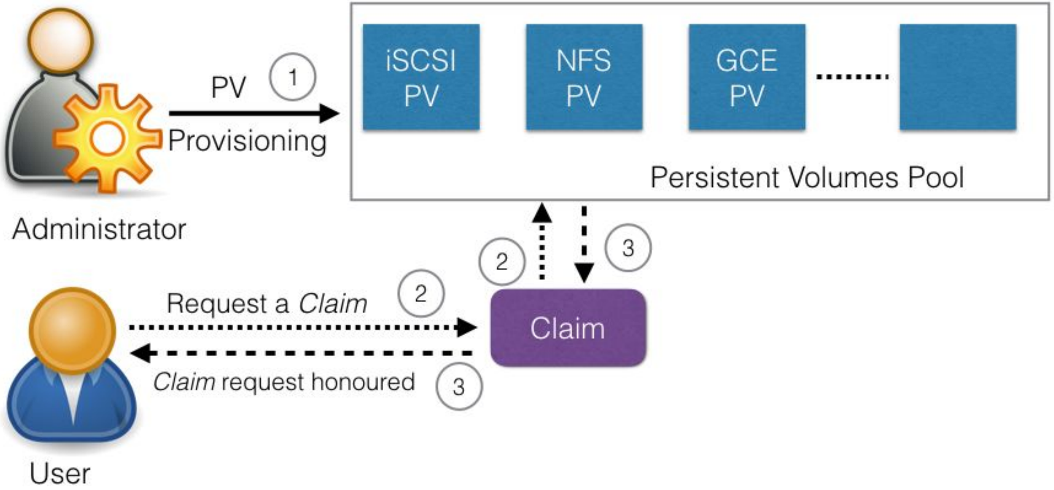
→ A PersistentVolumeClaim (PVC) is a namespaced request for storage.

→ Satisfies a set of requirements instead of mapping to a storage resource directly.

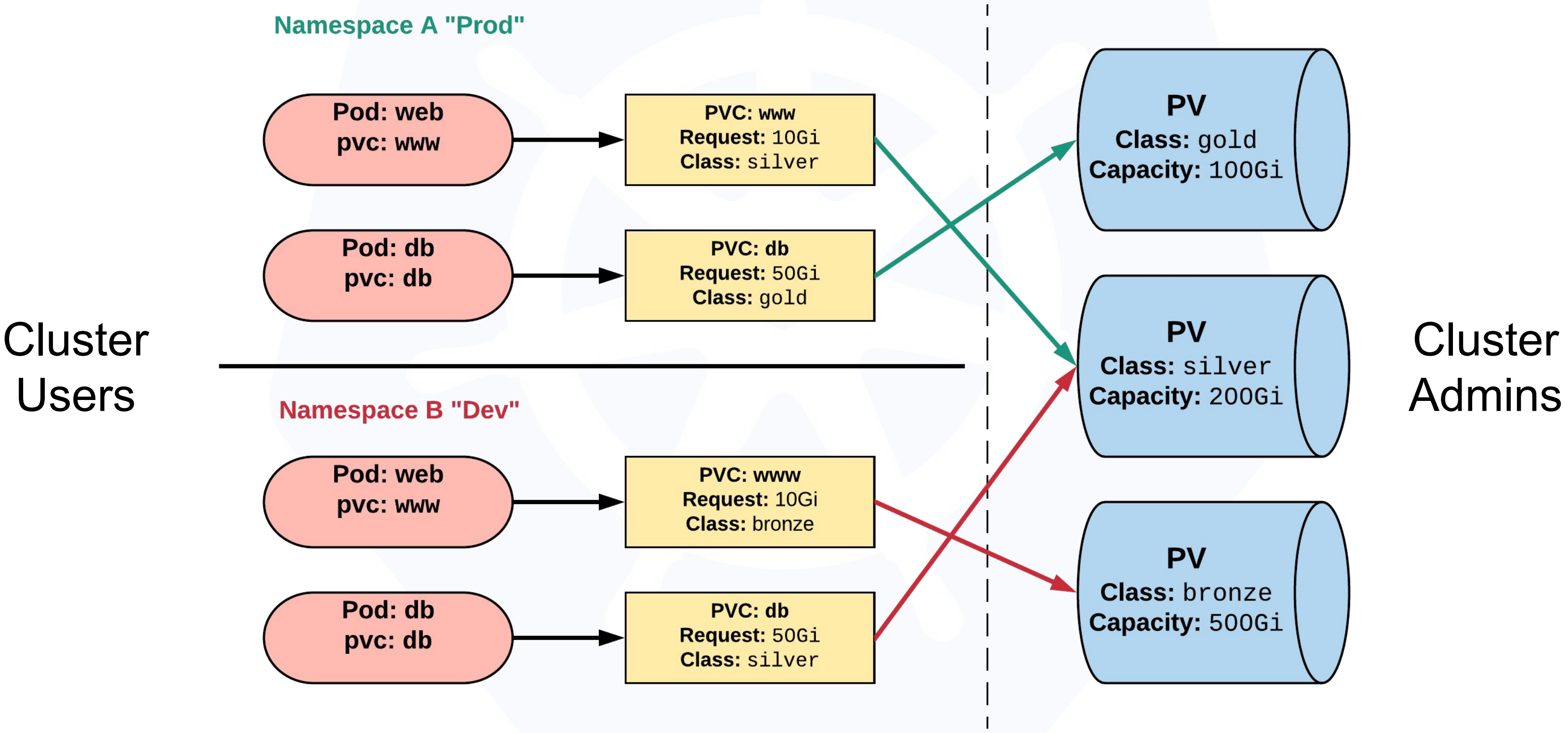
→ Ensures that an application’s ‘*claim*’ for storage is portable across numerous backends or providers.

Persistent Volume Claim

Persistent Volumes Claim (PVC)



Persistent Volumes and Claims



PersistentVolume

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfsserver
spec:
  capacity:
    storage: 50Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
    - ReadWriteMany
  persistentVolumeReclaimPolicy:
Delete
  storageClassName: slow
  mountOptions:
    - hard
    - nfsvers=4.1
  nfs:
    path: /exports
    server: 172.22.0.42
```

- **`capacity.storage:`** **The total amount of available storage.**
- **`volumeMode:`** **The type of volume, this can be either `Filesystem` or `Block`.**
- **`accessModes:`** **A list of the supported methods of accessing the volume. Options include:**
  - `ReadWriteOnce`
  - `ReadOnlyMany`
  - `ReadWriteMany`
  - `ReadWriteOncePod`

PersistentVolume

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfsserver
spec:
  capacity:
    storage: 50Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Delete
  storageClassName: slow
  mountOptions:
    - hard
    - nfsvers=4.1
  nfs:
    path: /exports
    server: 172.22.0.42
```

- `persistentVolumeReclaimPolicy:` The behaviour for PVC’s that have been deleted. Options include:
    - `Retain` - manual clean-up
    - `Delete` - storage asset deleted by provider.
- `storageClassName:` Optional name of the storage class that PVC’s can reference. If provided, ONLY PVC’s referencing the name consume it.
- `mountOptions:` Optional mount options for the PV.

PersistentVolumeClaim

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-sc-example
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: slow
```

- **`accessModes:`** The selected method of accessing the storage. This MUST be a subset of what is defined on the target PV or Storage Class.
  - `ReadWriteOnce`
  - `ReadOnlyMany`
  - `ReadWriteMany`
- **`resources.requests.storage:`** The desired amount of storage for the claim
- **`storageClassName:`** The name of the desired Storage Class

PVs and PVCs with Selectors

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-selector-example
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 1Gi
  selector:
    matchLabels:
      type: hostpath
```

```
kind: PersistentVolume
apiVersion: v1
metadata:
  name: pv-selector-example
  labels:
    type: hostpath
spec:
  capacity:
    storage: 2Gi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: "/mnt/data"
```

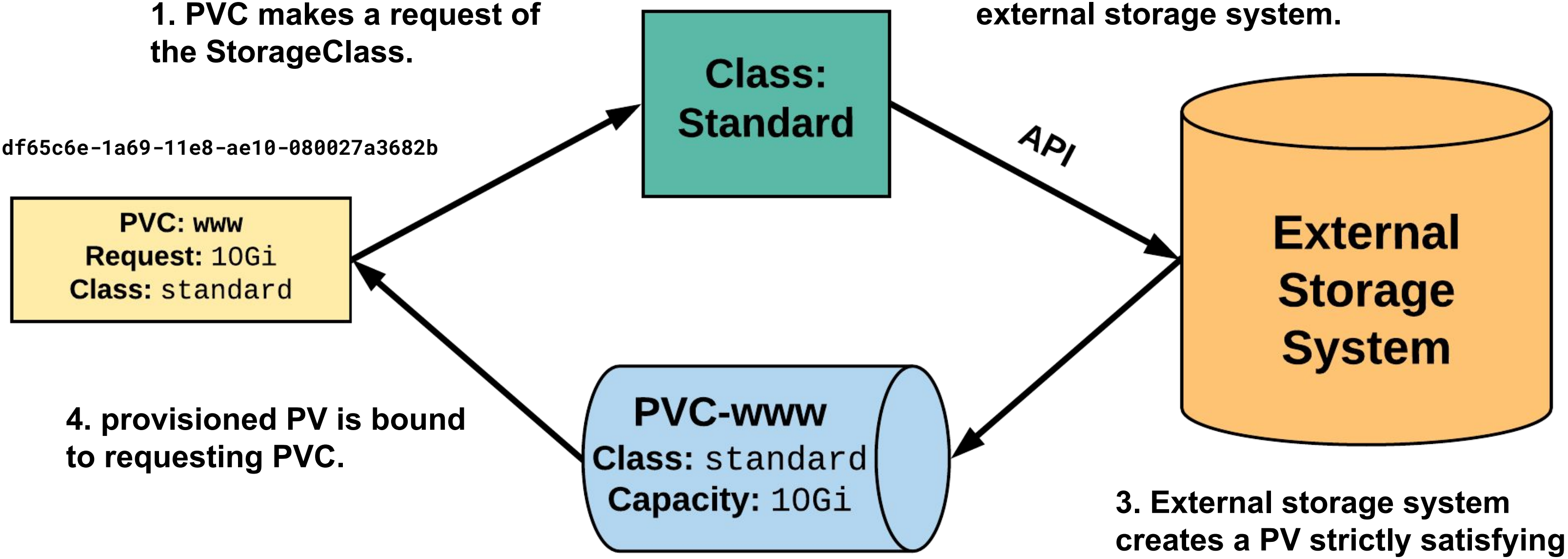
StorageClass

- Storage classes are an abstraction on top of an external storage resource (PV)
- Work hand-in-hand with the external storage system to enable dynamic provisioning of storage
- Eliminates the need for the cluster admin to pre-provision a PV

StorageClass

1. PVC makes a request of the StorageClass.

2. StorageClass provisions request through API with external storage system.



uid: 9df65c6e-1a69-11e8-ae10-080027a3682b

4. provisioned PV is bound to requesting PVC.

3. External storage system creates a PV strictly satisfying the PVC request.

pv: pvc-9df65c6e-1a69-11e8-ae10-080027a3682b

StorageClass

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: standard
provisioner: kubernetes.io/gce-pd
parameters:
  type: pd-standard
  zones: us-central1-a, us-central1-b
reclaimPolicy: Delete
```

➔ `provisioner:` Defines the '*driver*' to be used for provisioning of the external storage.

➔ `parameters:` A hash of the various configuration parameters for the provisioner.

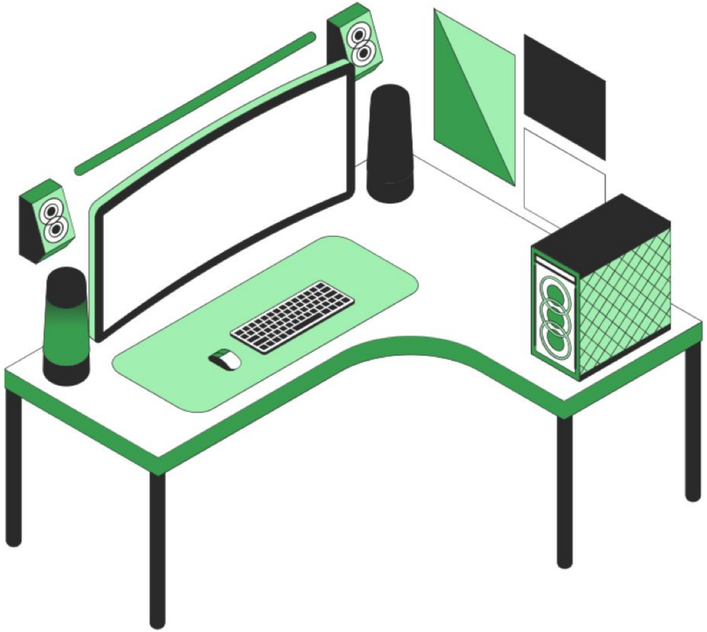
➔ `reclaimPolicy:` The behaviour for the backing storage when the PVC is deleted.

- `Retain` - manual clean-up
- `Delete` - storage asset deleted by provider

Available StorageClasses

- AWSElasticBlockStore
- AzureFile
- AzureDisk
- CephFS
- Cinder
- FC
- Flocker
- GCEPersistentDisk
- Glusterfs
- iSCSI
- Quobyte
- NFS
- RBD
- VsphereVolume
- PortworxVolume
- ScaleIO
- StorageOS
- Local

🟦 Internal Provisioner



Do you have any questions?

Send it to us! We hope you learned something new.