PersistentVolume (PV) and PersistentVolumeClaim (PVC)

Persistent Volumes

A PersistentVolume (PV) is a storage resource in the cluster that has been provisioned by an administrator or dynamically provisioned using Storage Classes.

Static Provisioning:

A cluster administrator creates several PVs. They carry the details of the real storage, which is available for use by cluster users.

awsElasticBlockStore:

Before you can use an EBS volume with a Pod, you need to create it.

```
aws ec2 create-volume \--availability-zone=eu-west-1a \--size=100 \--volume-type=gp2PersistentVolume spec:
```

Here,

```
---
apiVersion: v1
gcePersistentDisk: ~
kind: PersistentVolume
metadata:
  name: test-volume
spec:
  accessModes:
    - ReadWriteOnce
  awsElasticBlockStore:
    fsType: ext4
    volumeID: ~
  capacity:
    storage: 100Gi
  storageClassName: ebs-disk
```

Before creating a PersistentVolume, you must create the PD.

```
gcloud beta compute disks create --size=200GB my-data-disk \--region us-central1 \ --replica-zones
us-central1-a,us-central1-b
```

PersistentVolume spec:

```
---
apiVersion: v1
kind: PersistentVolume
metadata:
  name: test-volume
spec:
  accessModes:
    - ReadWriteOnce
  capacity:
    storage: 200Gi
  gcePersistentDisk:
    fsType: ext4
    pdName: my-data-disk
  storageClassName: gcp-disk
```

Check persistent Volumes

```
kubectl get pv
NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECLASS
REASON AGE
test-volume 200Gi RWO Delete Available gcp-disk 6s
```

azureDisk:

Before creating a PersistentVolume, you must create a virtual disk in Azure.

PersistentVolume spec:

```
---
apiVersion: v1
```

```
kind: PersistentVolume
metadata:
  name: test-volume
spec:
  accessModes:
    - ReadWriteOnce
  azureDisk:
    diskName: test.vhd
    diskURI: "https://someaccount.blob.microsoft.net/vhds/test.vhd"
  capacity:
    storage: 500Gi
  storageClassName: azure-disk
```

azureFile:

You will need to create a Kubernetes secret that holds both the account name and key.

```
kubectl create secret generic azure-secret \ — from-literal=azurestorageaccountname=< … > \ —
from-literal=azurestorageaccountkey=< … >
```

Before creating a PersistentVolume, create Azure Files share.

PersistentVolume spec:

```
---
apiVersion: v1
kind: PersistentVolume
metadata:
  name: sample-storage
spec:
  accessModes:
    - ReadWriteMany
  azureFile:
    readOnly: false
    secretName: azure-secret
    shareName: k8stest
  capacity:
    storage: 10Gi
  persistentVolumeReclaimPolicy: Retain
  storageClassName: azure-file-share
```

NFS:

Before creating a PersistentVolume, You will need NFS server details.

PersistentVolume spec:

```
---
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfs
spec:
  accessModes:
    - ReadWriteMany
  capacity:
    storage: 1Mi
  nfs:
    path: /
    server: nfs-server.mydomain.com
  storageClassName: nfs
```

Dynamic Provisioning:

When none of the static PVs match a user's PersistentVolumeClaim, the cluster may try to dynamically provision a volume, especially for the PVC.

This provisioning is based on StorageClasses, the PVC must request a storage class and the administrator must have created and configured that class for dynamic provisioning to occur.

StorageClasses:

Volume implementations are configured through StorageClass resources.

If you set up a Kubernetes cluster on GCP, AWS, Azure, or any other cloud platform, a default StorageClass creates for you which uses the standard persistent disk type.

List storage class:

AWS:

```
kubectl get storageclass
NAME PROVISIONER AGE
default (default) kubernetes.io/aws-ebs 3d
```

GCP:

```
kubectl get storageclass
NAME PROVISIONER AGE
standard (default) kubernetes.io/gce-pd 3d
```

StorageClass Configuration:

```
---
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: standard
provisioner: kubernetes.io/aws-ebs
reclaimPolicy: Retain
volumeBindingMode: Immediate
```

Capacity:

Generally, a PV will have a specific storage capacity. This is set using the PV’s capacity attribute.

Currently, storage size is the only resource that can be set or requested.

Provisioner:

Storage classes have a provisioner that determines what volume plugin is used for provisioning PVs.

Reclaim Policy:

It can be either Delete or Retain. Default is Delete.

Volume Binding Mode:

The volumeBindingMode field controls when volume binding and dynamic provisioning should occur. Immediate is default and specifying the WaitForFirstConsumer mode.