## PersistentVolume :-



Here we can see that we have a volume inside the Pod and it is mapped to the Host Machine in path /opt/conf/ directory.

So if any time if we change any thing in host machine or Pod then changes will be reflected vice versa.

But in case if Pod goes down and if the object is created by RS or Deployment Object then it will create the same Pod in second worker machine.

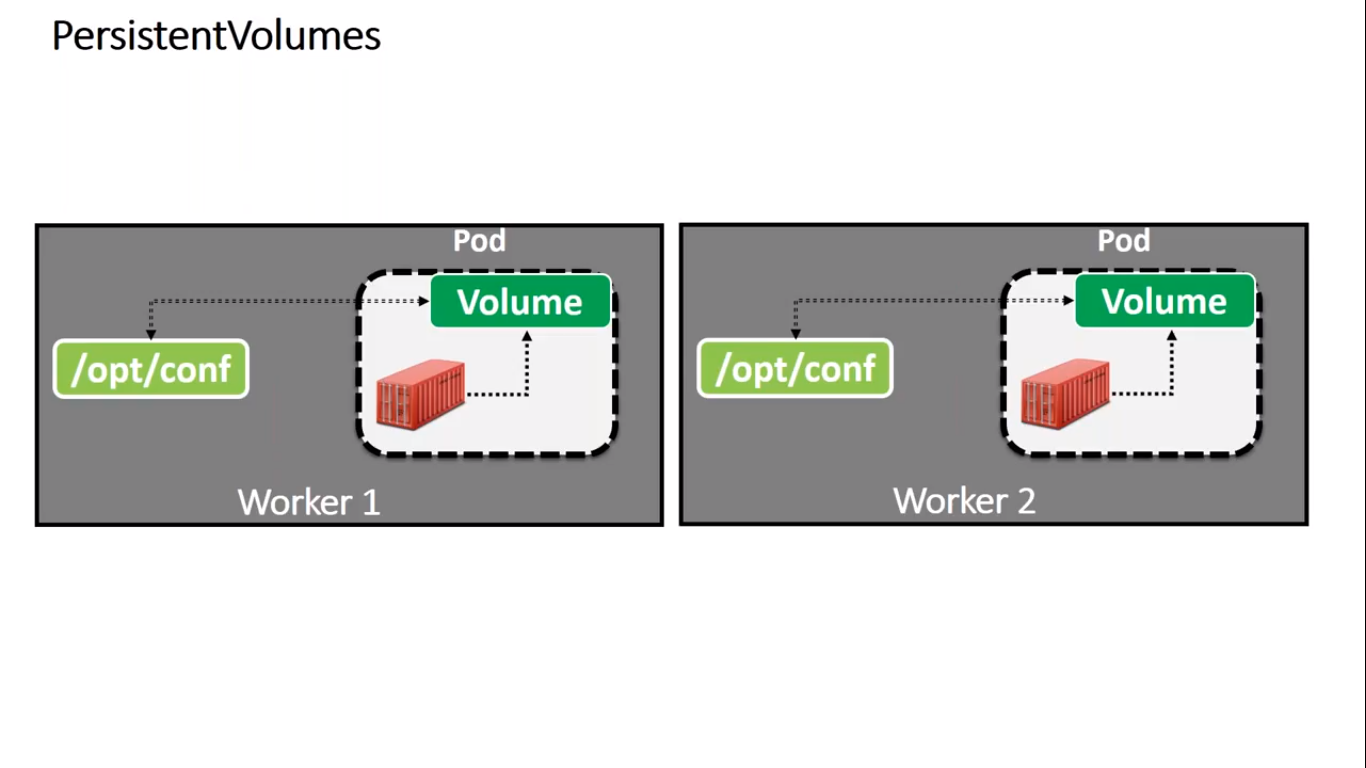
And there also volume is mapped to /opt/conf path in Host Machine.

But the content in /opt/conf in worker 1 will not be same in the content of worker 2.

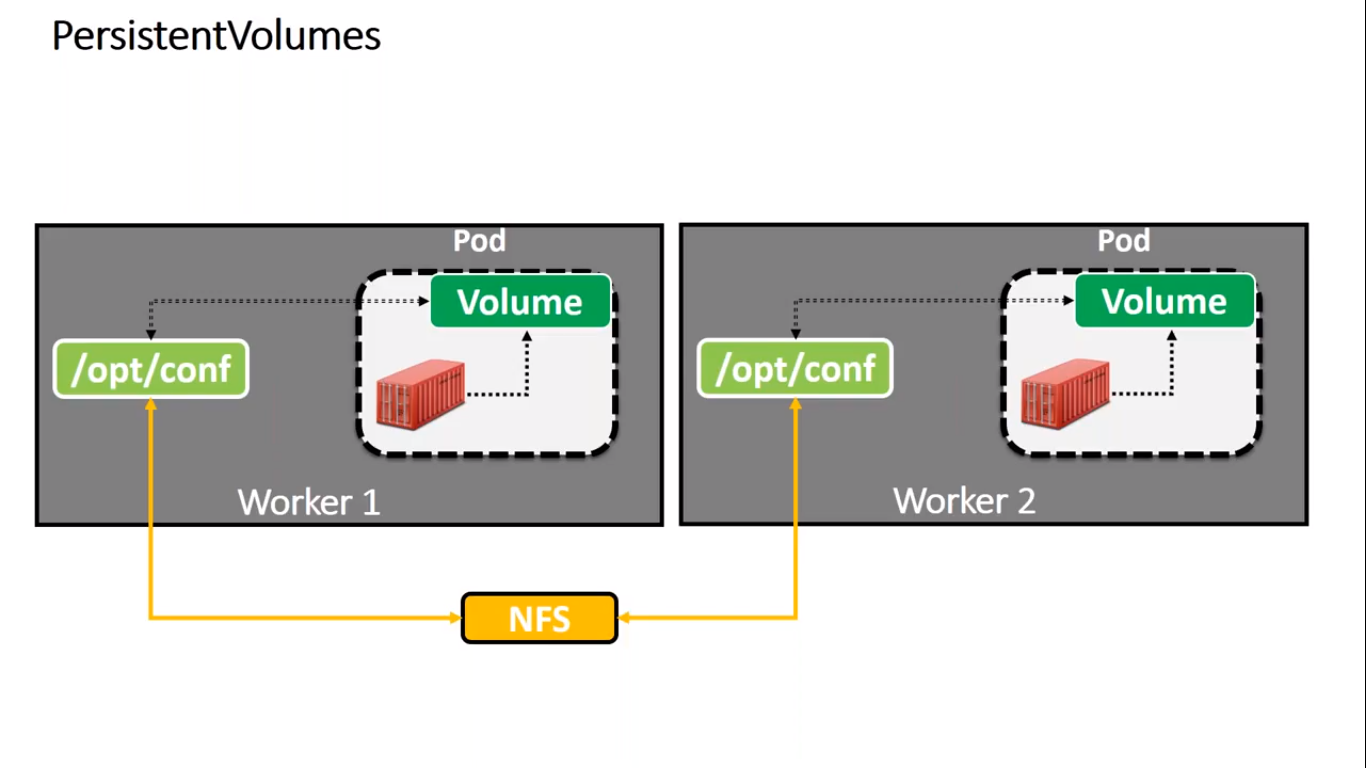
As a result we can lose the Data.

So we required something like this in which host machine volume folder should be same across all the Host Machine and content as well.

We required not a regular volume so here we required a Volume which is called Persistent Volume.



Here folder is shared across all host machine is persistent.

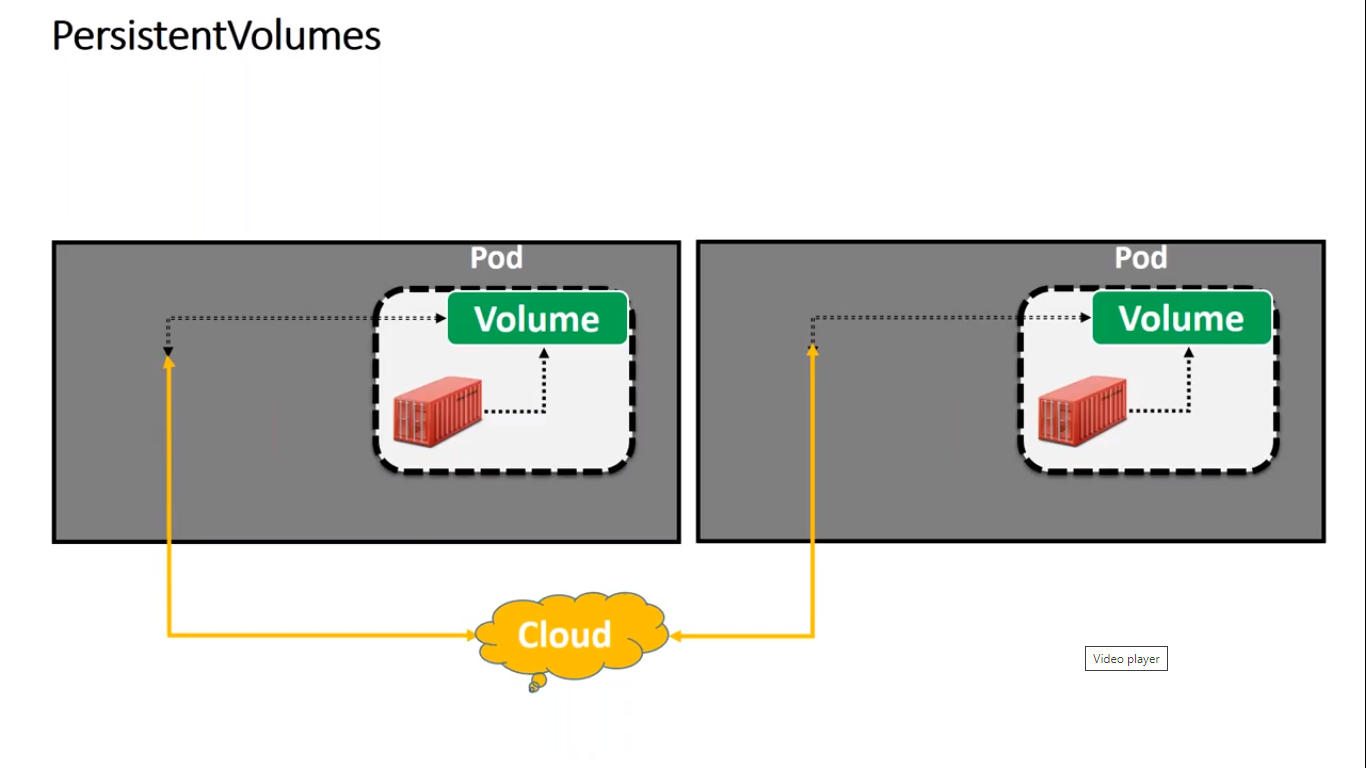


So to achieve it we will be using a Linux sharable server which is called NFS.

This NFS will be installed any where in Linux machine and through it whatever changes are done in one machine will be share to other machine through NFS.

**But problem is that NFS is bit slow.**

**So** Next option we do have is **Cloud,** Many Cloud service provider like AWS,Azure.

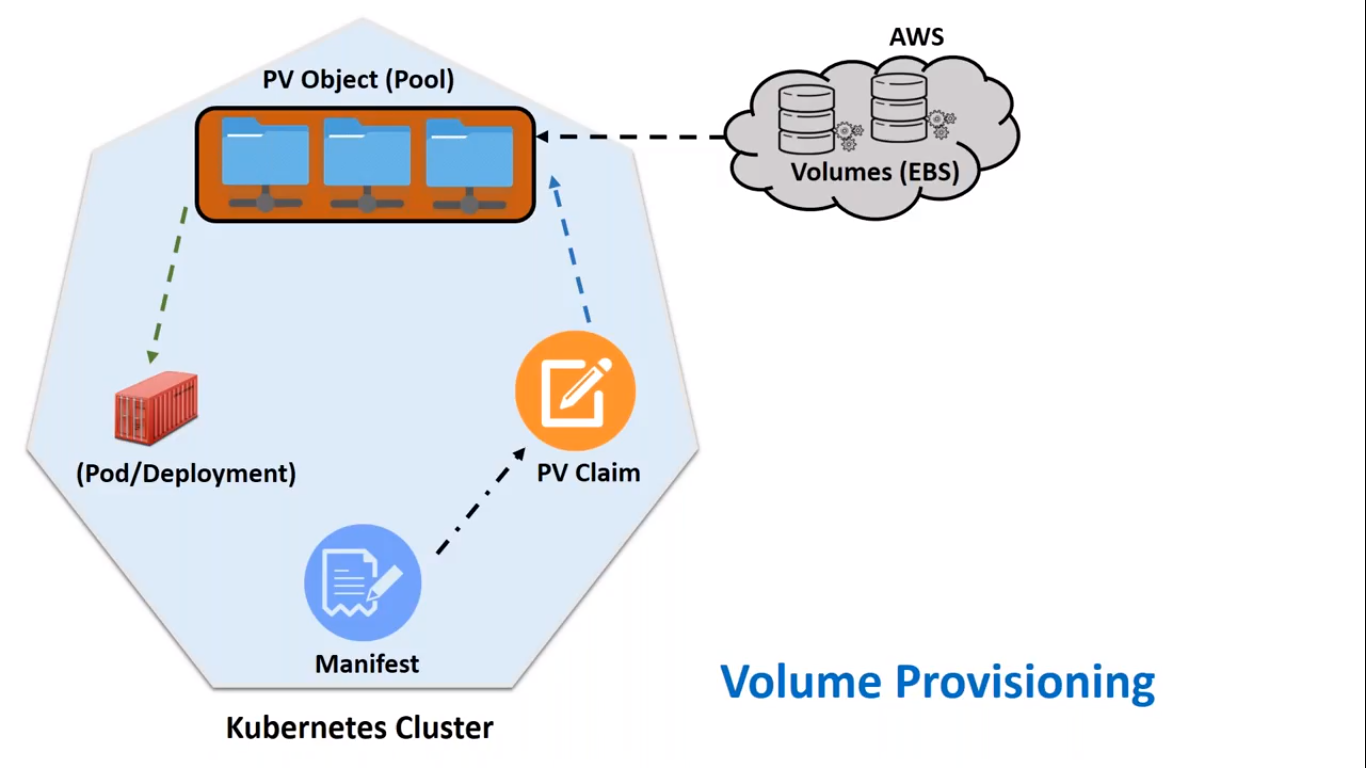


Here if we mapped the Cloud Storage in the cluster. So if any changes happen in any machine will be shared across all the machine.

It will be fast and very much reliable also.

If any Machine goes down then new Pod will be create by any Object like RS and same storage from Cloud will be shared to new Pod.





Here we can see that to achieve Persistent Volume firstly we have to create Storage in AWS .

Based on that Storage in AWS it will create Persistent Volume Object in Ks8 cluster.

There may be group of PV object created in ks8 clusters.

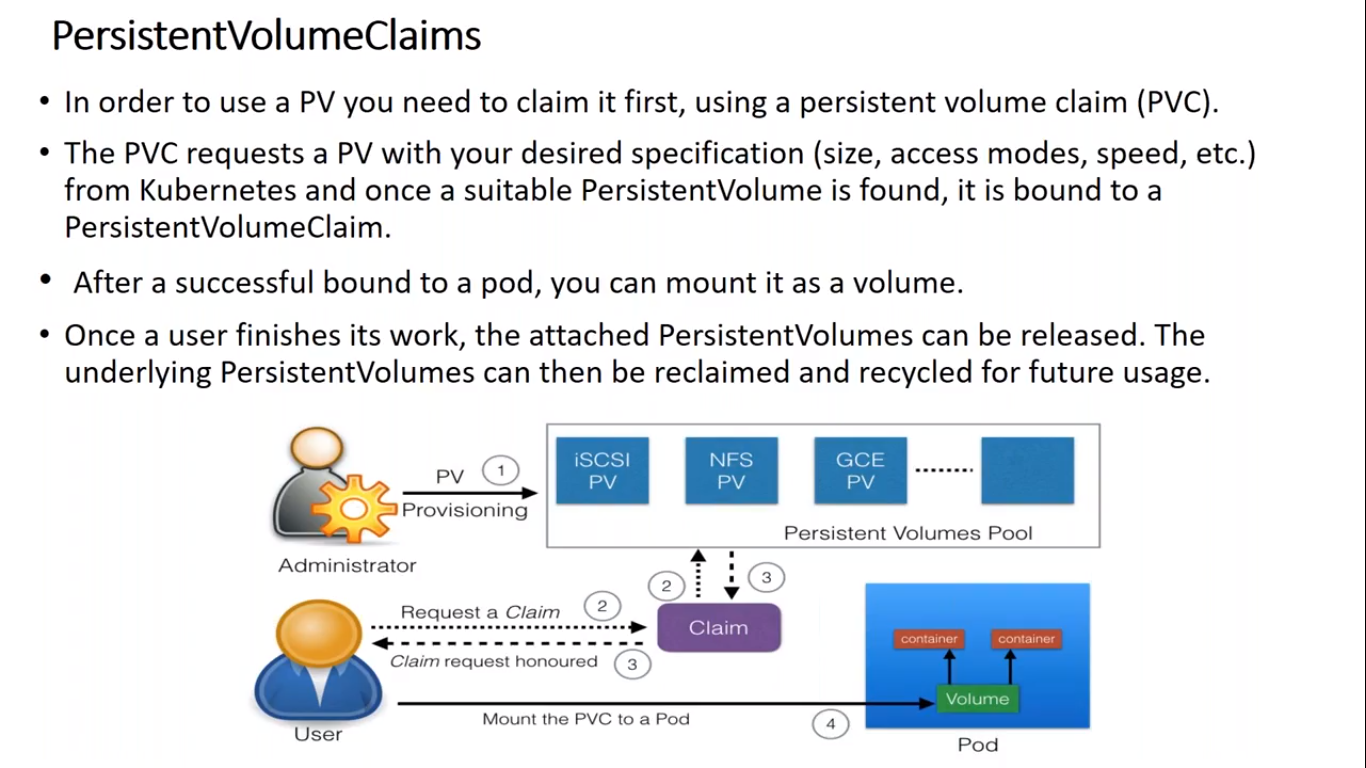
These group of Object is called as pool of PV object.

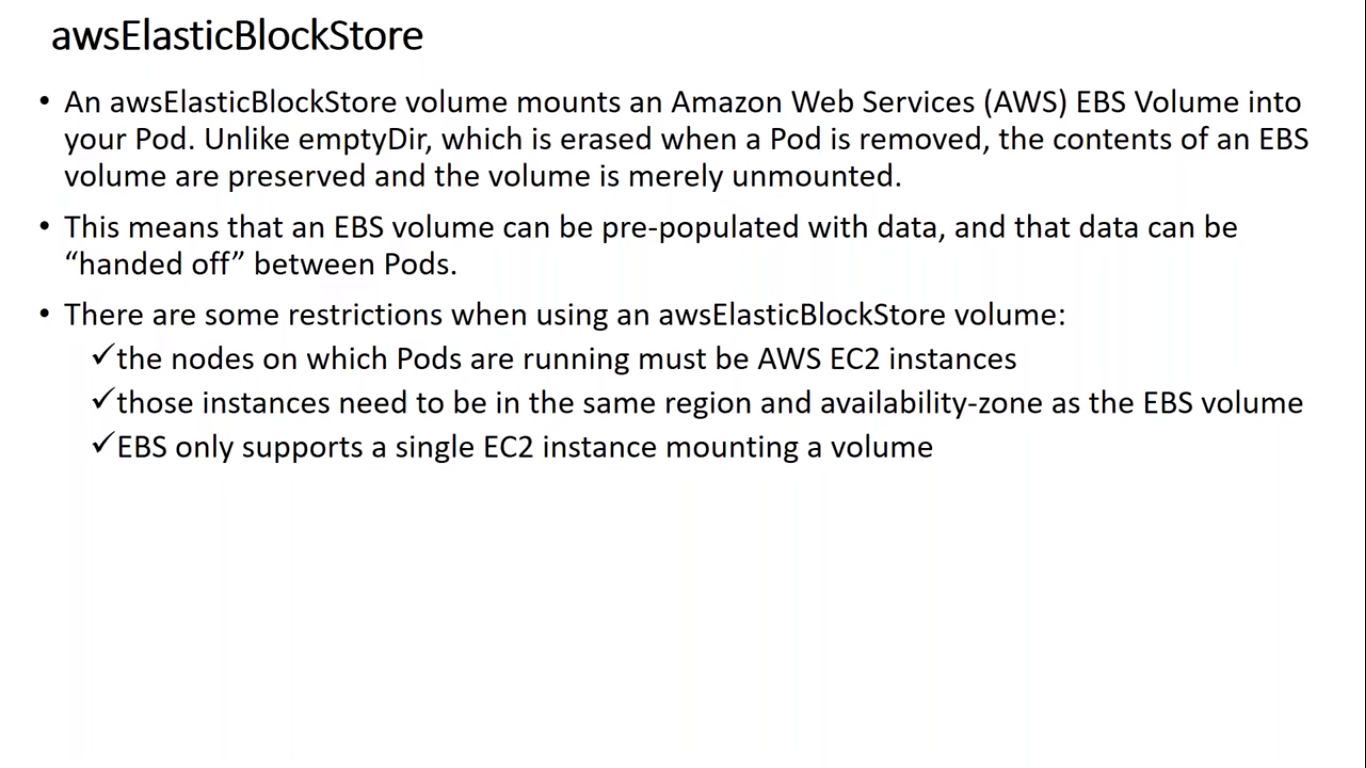
Now here question arises which Object we need to choose.

To achieve it we have to create one more object called PV Claim. it will decide based on our need which storage object will be selected and similarly a Pod will be created by mapping to this storage object.

Whatever will be store in Pod same will be stored in AWS PV object.

This is similar like our Medical Insurance Policy Claim there also to claim we have to raise claim and based on our need it will select the hospitals and claimed our policy.

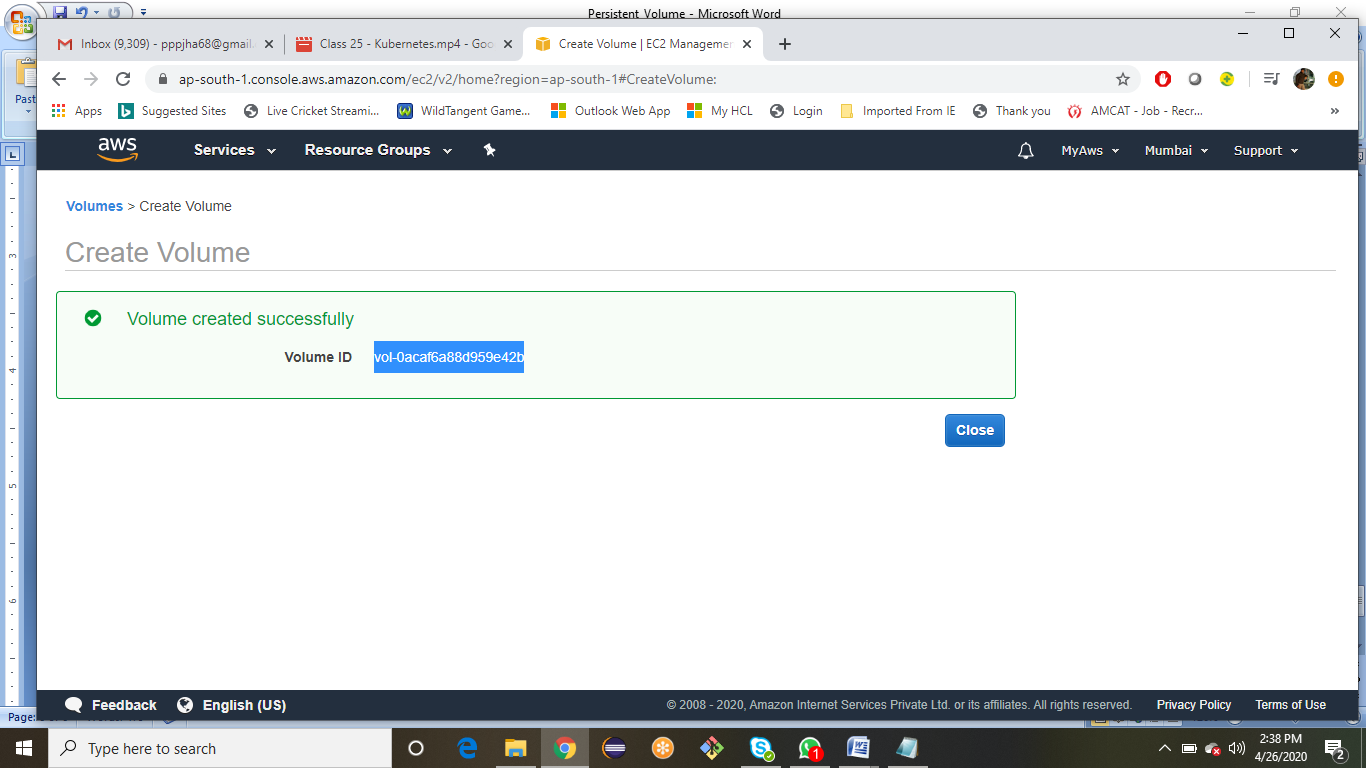




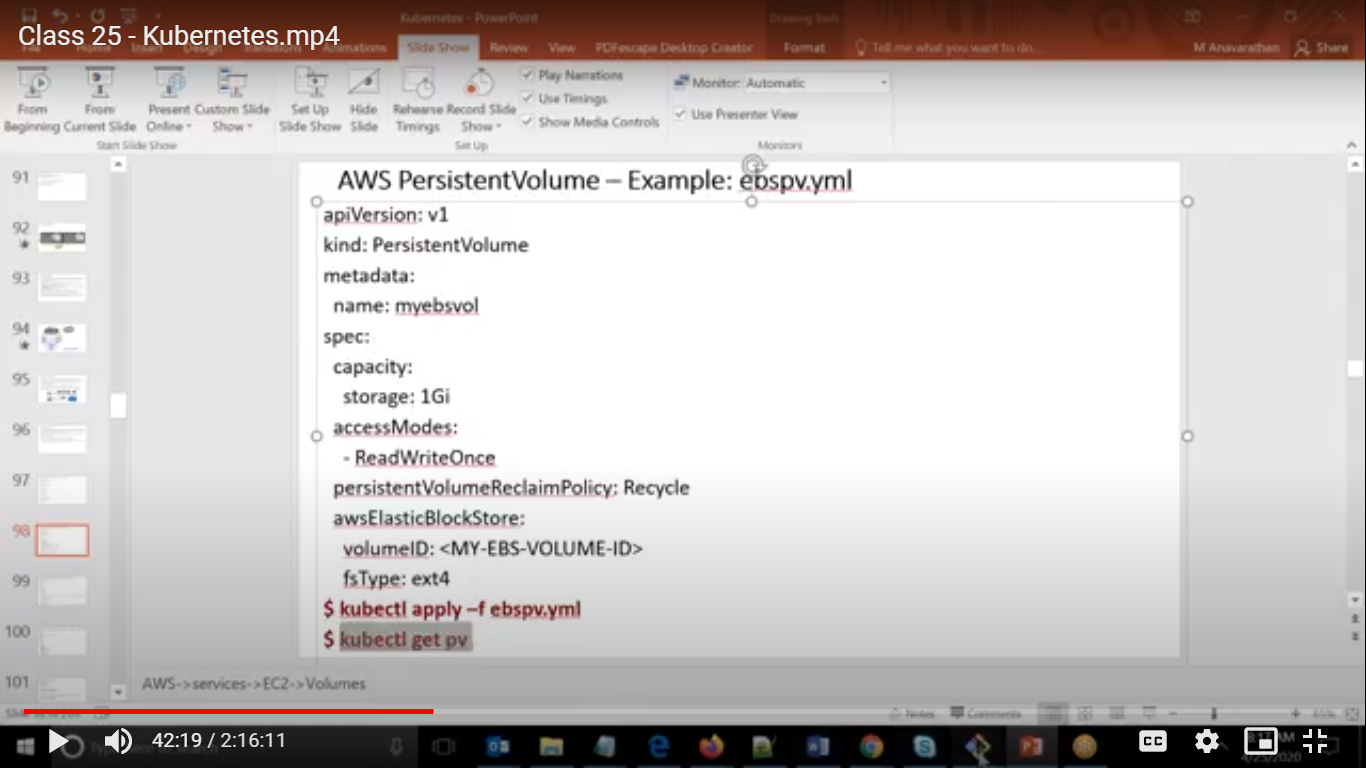
AWS has a service called ElasticBlockStore.

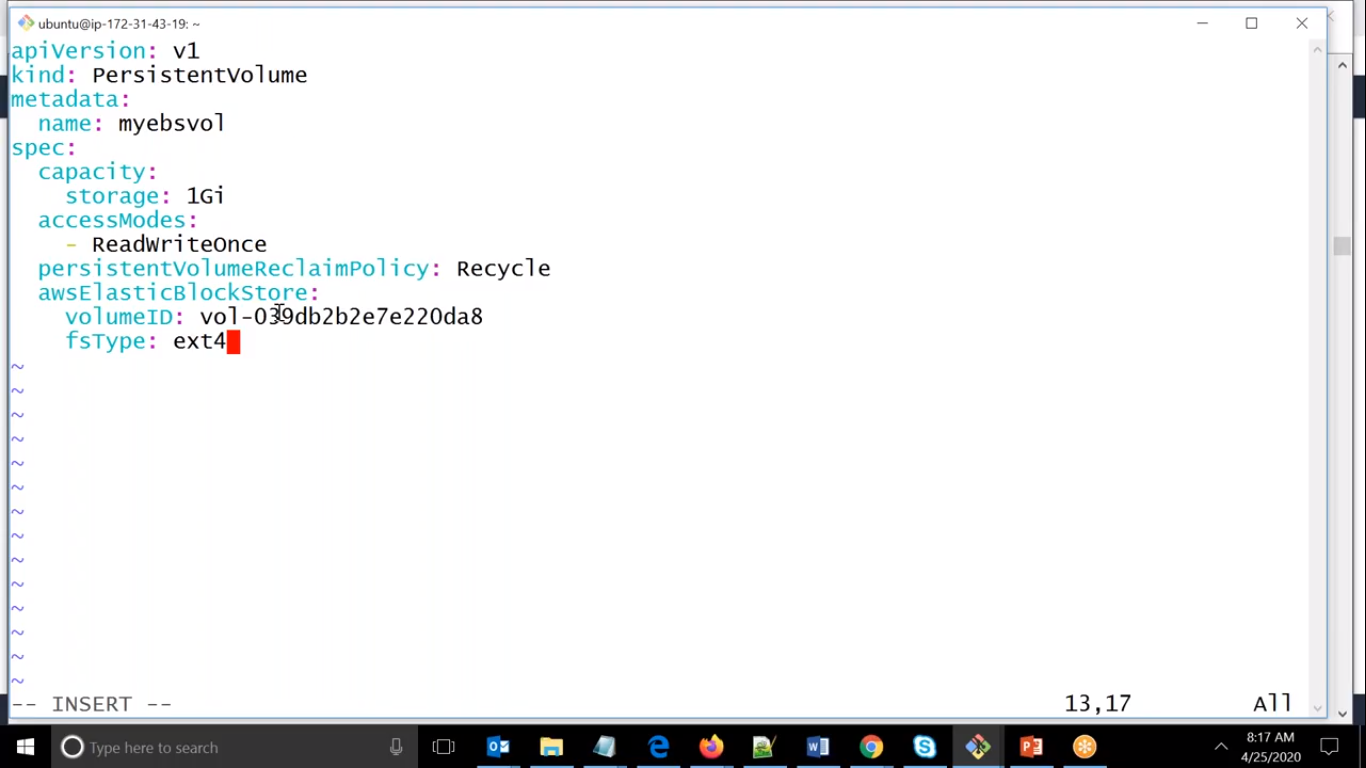
To use it we have some conditions.

1. The nodes on which Pods are running must be AWS EC2 instances.
2. Both should be in same region.
3. EBS only supports a single EC2 instance mounting a volume.



Here we have created AWS EBS.





Here we can see that we are creating PV object.

Here Kind is PV.

Storage is 1 Gi, i.e. we need only 1 GB storage.

Volume ID is the Storage id :- vol-0acaf6a88d959e42b .

root@ip-172-31-46-102:/home/ubuntu# vi pv.yml

root@ip-172-31-46-102:/home/ubuntu# kubectl apply -f pv.yml

persistentvolume/myebsvol created

root@ip-172-31-46-102:/home/ubuntu# kubectl get pv

NAME CAPACITY ACCESS MODES RECLAIM POLICY STATUS CLAIM STORAGECLASS REASON AGE

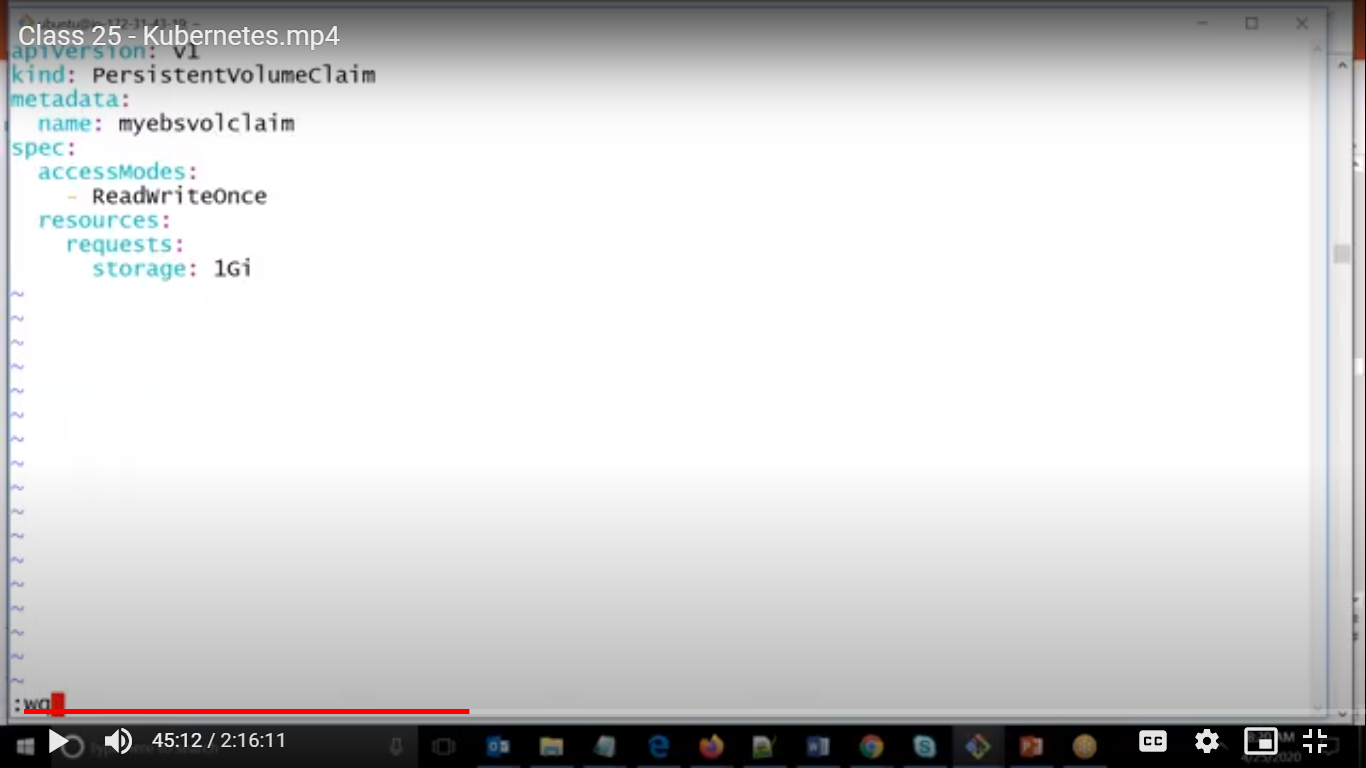
myebsvol 1Gi RWO Recycle Available 55s

root@ip-172-31-46-102:/home/ubuntu#

Here we have created PV object and correspondingly EBS storage.

We can check it by using kubectl get pv.

Now we need to create PVC object.

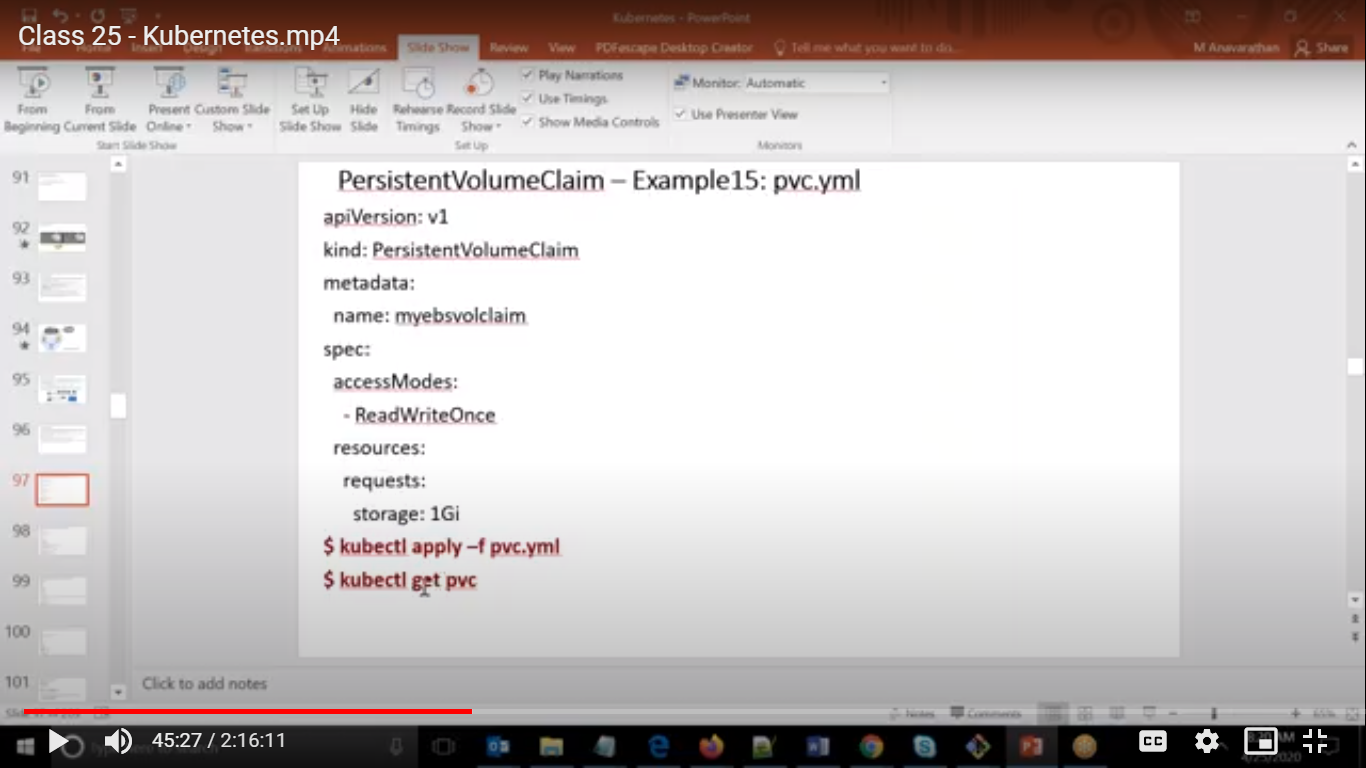


Here we have version is V1

Kind is PVC

accessModes :- Readwriteonce it means that it will search in PV object whichever PV object is having this mode then it will matches to that one.

Requesting storage of 1 GB.



Here we have created PVC object.

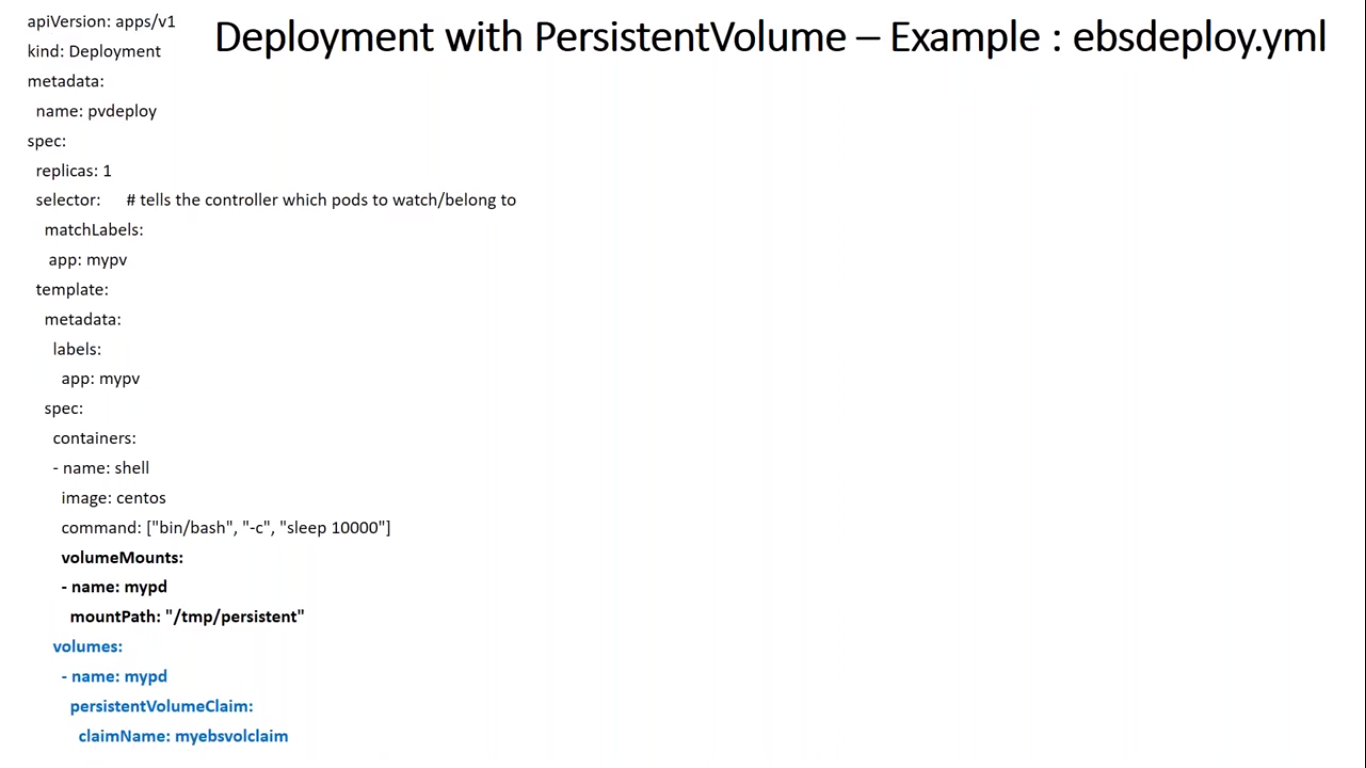
root@ip-172-31-46-102:/home/ubuntu# vi pvc.yml

root@ip-172-31-46-102:/home/ubuntu# kubectl apply -f pvc.yml

persistentvolumeclaim/myebsvolclaim created

To see it we will be using kubectl get pvc

Now we will create a deployment object so that if any pod goes down it will recreate it.



Here we can see that Kind is Deployment.

spec for Deployment Object, and name is mypv which should matches to the name of pod as well.

Then we have template for pod which is having its spec.

Here we are using volumes as PVC and its name.

Here we have Volume Mounts name in pod is identical to name in AWSEBS storage name I.e. mypd.

So whatever we will store in pod will be shared to the ebs volume and sometime if it is goes down

it will create again new pod and ebs storage will be share to that pod and in this way we will have same contents.

root@ip-172-31-46-102:/home/ubuntu# vi ebsdeploy.yml

root@ip-172-31-46-102:/home/ubuntu# kubectl apply -f ebsdeploy.yml

deployment.apps/pvdeploy created

root@ip-172-31-46-102:/home/ubuntu# kubectl get deploy

NAME READY UP-TO-DATE AVAILABLE AGE

pvdeploy 1/1 1 1 34s

root@ip-172-31-46-102:/home/ubuntu# kubectl get rs

NAME DESIRED CURRENT READY AGE

pvdeploy-559d7dc768 1 1 1 45s

root@ip-172-31-46-102:/home/ubuntu# kubectl get pod

NAME READY STATUS RESTARTS AGE

pvdeploy-559d7dc768-zkm4f 1/1 Running 0 55s

Here we can see it created pod which is mapped to the EBS storage volume.