1. Create and Test a Kubernetes Pod with an EmptyDir Volume

vi emptydir-pod.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: emptydir-pod
spec:
  containers:
  - name: busybox-container
    image: busybox
    command: ["/bin/sh", "-c", "sleep 3600"]
    volumeMounts:
    - name: emptydir-volume
      mountPath: /data
  volumes:
  - name: emptydir-volume
    emptyDir: {}
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: emptydir-pod
spec:
  containers:
  - name: busybox-container
    image: busybox
    command: ["/bin/sh", "-c", "sleep 3600"]
    volumeMounts:
    - name: emptydir-volume
      mountPath: /data
  volumes:
  - name: emptydir-volume
    emptyDir: {}
```

- **kubectl apply -f emptydir-pod.yaml**
- **kubectl get pods**

```
[root@master ~]# vi emptydir-pod.yml
[root@master ~]# kubectl apply -f emptydir-pod.yml
pod/emptydir-pod created
[root@master ~]# kubectl get pods
NAME           READY   STATUS    RESTARTS   AGE
emptydir-pod   1/1     Running   0          71s
[root@master ~]#
```

- **kubectl exec -it emptydir-pod -- sh**
- **cd /data**
- **echo "This is emptyDir volume test" > test.txt**
- **cat test.txt**

```
[root@master ~]# kubectl exec -it emptydir-pod -- sh
/ # cd data/
/data # echo "This is emptyDir volume test" > test.txt
/data # ls
test.txt
/data # cat test.txt
This is emptyDir volume test
/data #
```

Delete the pod

- **kubectl delete pod emptydir-pod**
- **kubectl get pods**

```
[root@master ~]# kubectl delete pod emptydir-pod
pod "emptydir-pod" deleted from default namespace
[root@master ~]# kubectl get pods
No resources found in default namespace.
[root@master ~]#
```

Recreate the pod

- **kubectl apply -f emptydir-pod.yaml**
- **kubectl exec -it emptydir-pod -- sh**

```
[root@master ~]# kubectl apply -f emptydir-pod.yml
pod/emptydir-pod created
```

- **cd data**
- **ls**

```
[root@master ~]# kubectl exec -it emptydir-pod -- sh
/ # ls
bin    data   dev    etc    home   lib    lib64  proc   root   sys    tmp    usr    var
/ # cd data/
/data # ls
/data #
```

2. Configure a HostPath Volume in Kubernetes and Validate Data Persistence

- **kubectl get nodes**
- **vi hostpath-pod.yml**

```
apiVersion: v1
kind: Pod
metadata:
  name: hostpath-pod
spec:
  containers:
  - name: nginx-container
    image: nginx
    volumeMounts:
    - name: host-storage
      mountPath: /usr/share/nginx/html
  volumes:
  - name: host-storage
    hostPath:
      path: /data/hostpath
      type: DirectoryOrCreate
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: hostpath-pod
spec:
  containers:
  - name: nginx-container
    image: nginx
    volumeMounts:
    - name: host-storage
      mountPath: /usr/share/nginx/html
  volumes:
  - name: host-storage
    hostPath:
      path: /data/hostpath
      type: DirectoryOrCreate
```

- **kubectl apply -f hostpath-pod.yml**
- **kubectl get pods -o wide**

```
[root@master ~]# kubectl apply -f hostpath-pod.yml
pod/hostpath-pod created
[root@master ~]# kubectl get pods -o wide
NAME           READY   STATUS    RESTARTS   AGE   IP           NODE        NOMINATED NODE   READINESS GATES
hostpath-pod   1/1     Running   0          10s   10.244.1.64   worker-01   <none>           <none>
[root@master ~]#
```

- **kubectl exec -it hostpath-pod -- sh**
- **echo "HostPath persistence test" > /usr/share/nginx/html/index.html**

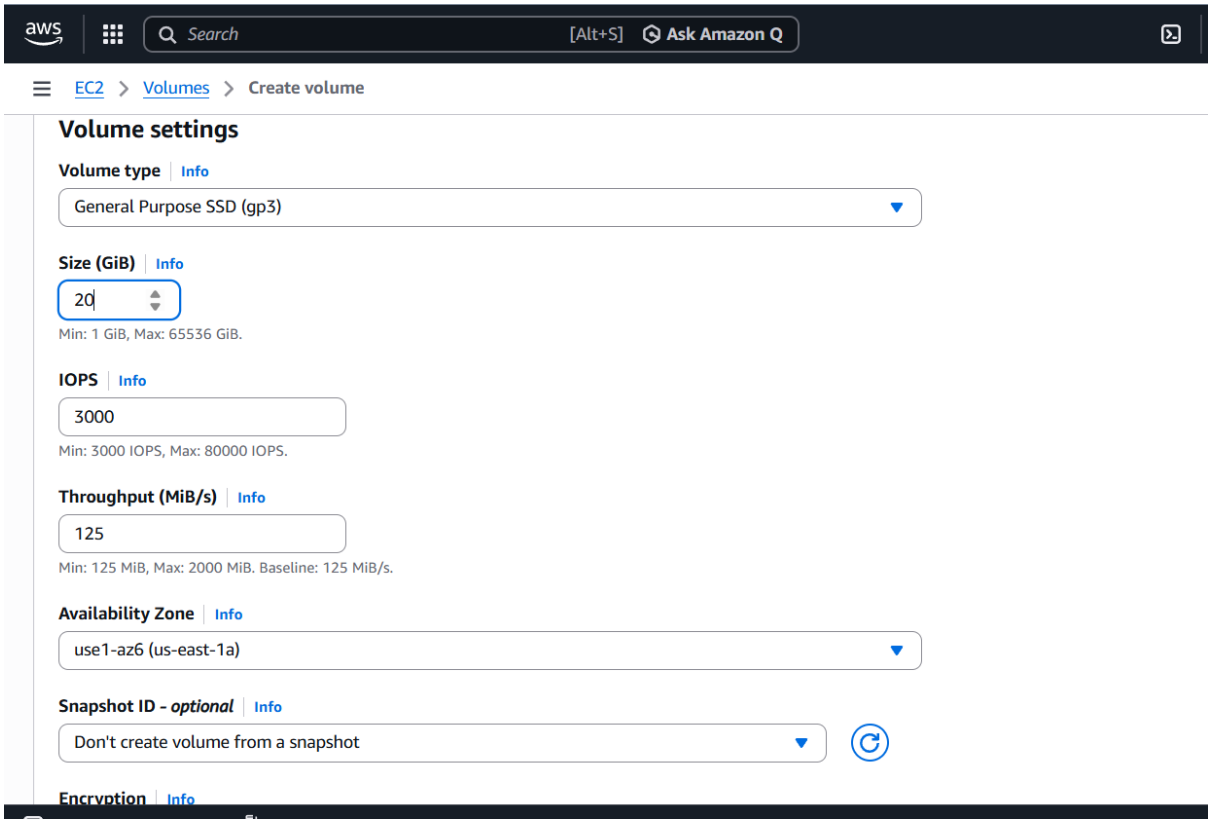
```
[root@master ~]# kubectl exec -it hostpath-pod -- sh
# echo "HostPath persistence test" > /usr/share/nginx/html/index.html
# ls
bin   dev                 docker-entrypoint.sh  home  lib64  mnt  proc  run   srv  tmp  var
boot  docker-entrypoint.d  etc                  lib   media  opt  root  sbin  sys  usr
# exit
```

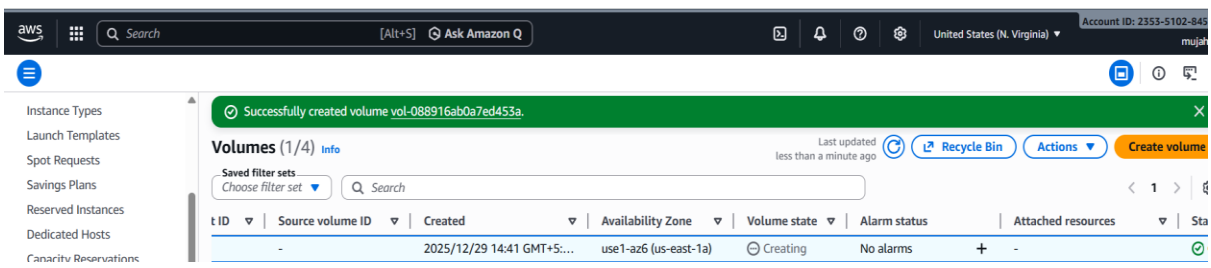
- **kubectl delete pod hostpath-pod**
- **kubectl apply -f hostpath-pod.yaml**
- **kubectl exec -it hostpath-pod -- cat /usr/share/nginx/html/index.html**

```
[root@master ~]# kubectl delete pod hostpath-pod
pod "hostpath-pod" deleted from default namespace
[root@master ~]# kubectl apply -f hostpath-pod.yml
pod/hostpath-pod created
[root@master ~]# kubectl exec -it hostpath-pod -- cat /usr/share/nginx/html/index.html
HostPath persistence test
[root@master ~]#
```

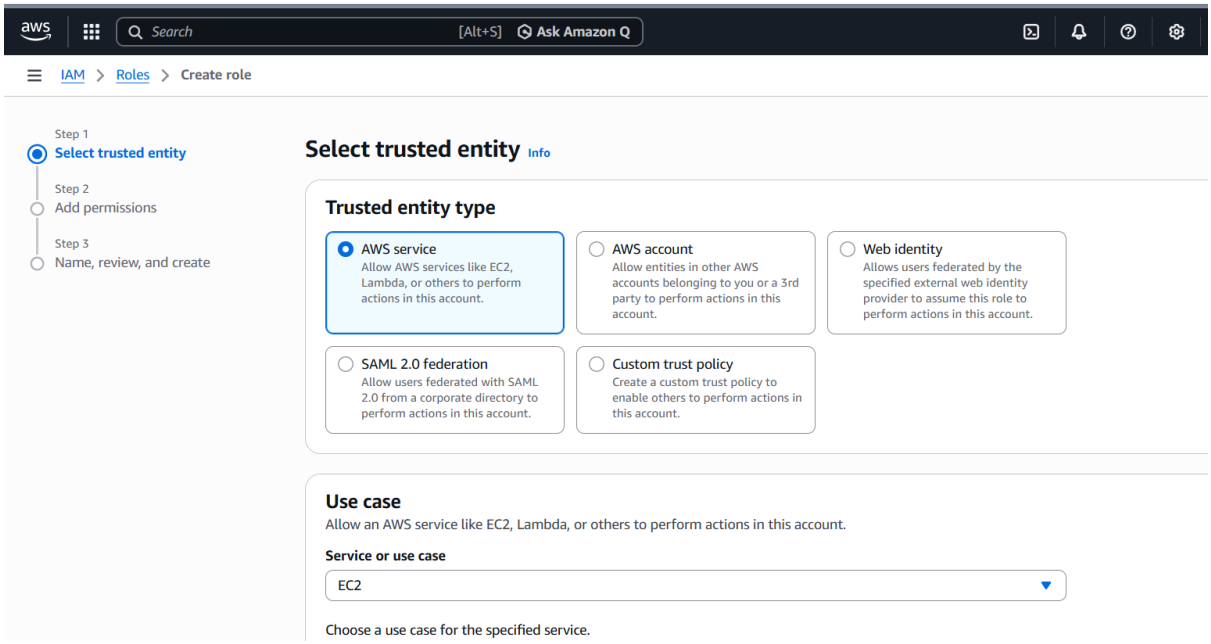
3. Deploy an Amazon EBS Volume Using Persistent Volume and Persistent Volume Claim (PVC)

Go to ec2 and click on volumes create a volume make sure that your volume will be same az of your node machines are there.

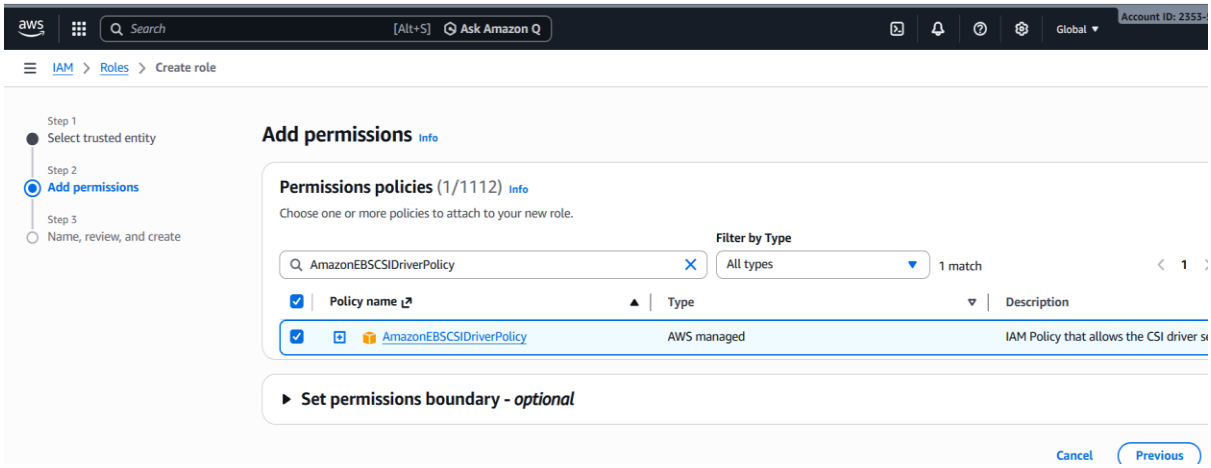




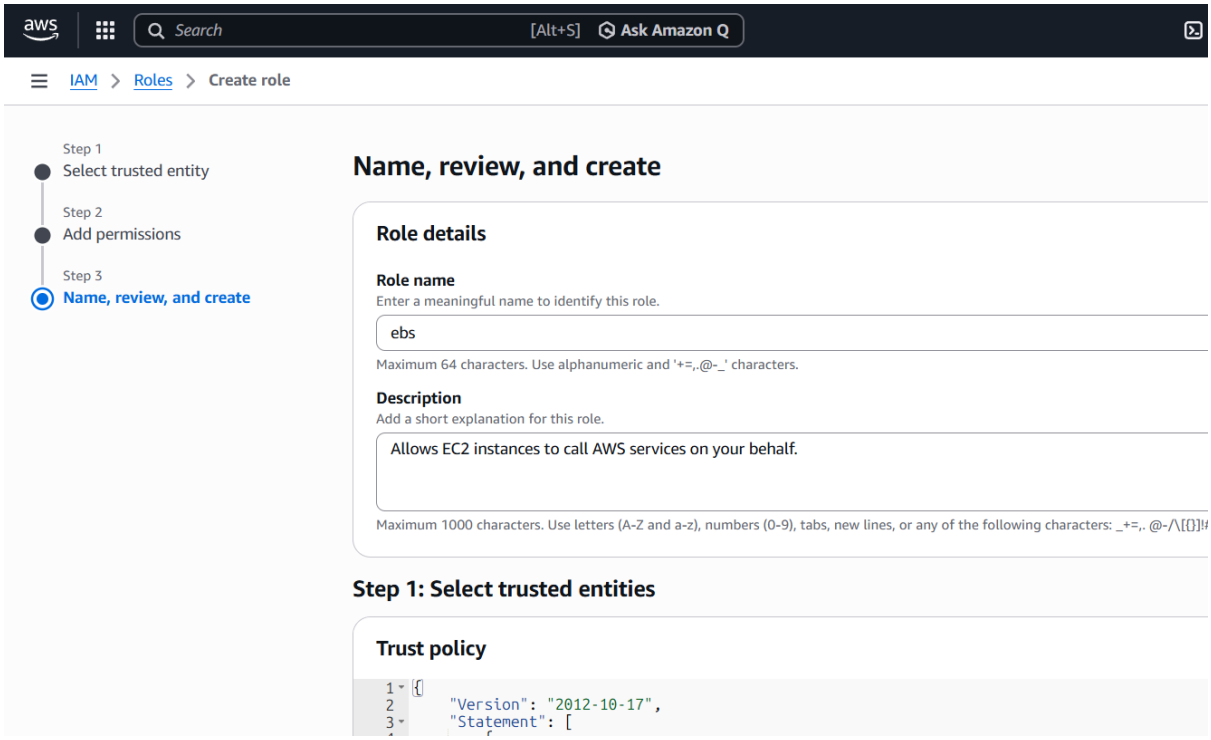
Go to IAM create an role and attach to 3 machines.



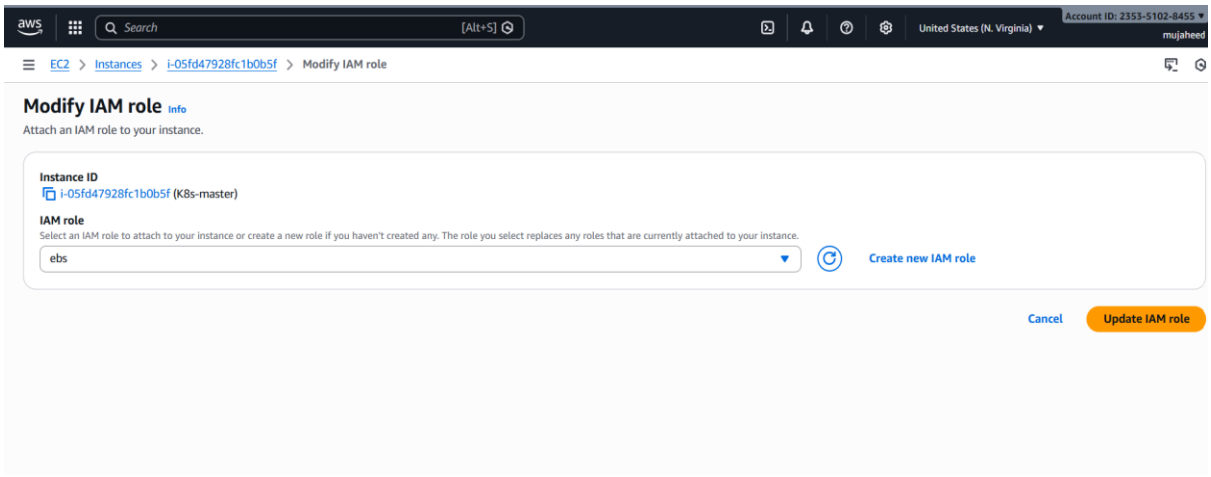
- **AmazonEBSCSIDriverPolicy**



Click on create



Modify IAM role for 3 machines.



In master machine execute this command.

- **git clone https://github.com/kubernetes-sigs/aws-ebs-csi-driver.git**

```
[root@master ~]# git clone https://github.com/kubernetes-sigs/aws-ebs-csi-driver.git
Cloning into 'aws-ebs-csi-driver'...
remote: Enumerating objects: 36882, done.
remote: Counting objects: 100% (925/925), done.
remote: Compressing objects: 100% (401/401), done.
remote: Total 36882 (delta 776), reused 530 (delta 522), pack-reused 35957 (from 4)
Receiving objects: 100% (36882/36882), 31.30 MiB | 29.78 MiB/s, done.
Resolving deltas: 100% (21233/21233), done.
[root@master ~]#
```

- **cd aws-ebs-csi-driver**
- **kubectl apply -k deploy/kubernetes/overlays/stable/**

```
[root@master ~]# cd aws-ebs-csi-driver
[root@master aws-ebs-csi-driver]# kubectl apply -k deploy/kubernetes/overlays/stable/
serviceaccount/ebs-csi-controller-sa created
serviceaccount/ebs-csi-node-sa created
role.rbac.authorization.k8s.io/ebs-csi-leases-role created
clusterrole.rbac.authorization.k8s.io/ebs-csi-node-role created
clusterrole.rbac.authorization.k8s.io/ebs-external-attacher-role created
clusterrole.rbac.authorization.k8s.io/ebs-external-provisioner-role created
clusterrole.rbac.authorization.k8s.io/ebs-external-resizer-role created
clusterrole.rbac.authorization.k8s.io/ebs-external-snapshotter-role created
rolebinding.rbac.authorization.k8s.io/ebs-csi-leases-rolebinding created
clusterrolebinding.rbac.authorization.k8s.io/ebs-csi-attacher-binding created
clusterrolebinding.rbac.authorization.k8s.io/ebs-csi-node-getter-binding created
clusterrolebinding.rbac.authorization.k8s.io/ebs-csi-provisioner-binding created
clusterrolebinding.rbac.authorization.k8s.io/ebs-csi-resizer-binding created
clusterrolebinding.rbac.authorization.k8s.io/ebs-csi-snapshotter-binding created
deployment.apps/ebs-csi-controller created
poddisruptionbudget.policy/ebs-csi-controller created
daemonset.apps/ebs-csi-node created
csidriver.storage.k8s.io/ebs.csi.aws.com created
[root@master aws-ebs-csi-driver]#
```

- **kubectl get pods -n kube-system**

```
[root@master aws-ebs-csi-driver]# kubectl get pods -n kube-system
NAME                                  READY   STATUS    RESTARTS        AGE
coredns-66bc5c9577-4g5tm              1/1     Running   14 (63m ago)    12d
coredns-66bc5c9577-sr6l5              1/1     Running   14 (63m ago)    12d
ebs-csi-controller-5d6d9d4bff-2p24q   6/6     Running   0               49s
ebs-csi-controller-5d6d9d4bff-wn8gr   6/6     Running   0               49s
ebs-csi-node-4z7fl                    3/3     Running   0               49s
ebs-csi-node-g97j5                    3/3     Running   0               49s
ebs-csi-node-nd9tk                    3/3     Running   0               49s
etcd-master                           1/1     Running   14 (63m ago)    12d
kube-apiserver-master                 1/1     Running   14 (63m ago)    12d
kube-controller-manager-master        1/1     Running   14 (63m ago)    12d
kube-proxy-594sd                      1/1     Running   14 (63m ago)    12d
kube-proxy-9v7j7                      1/1     Running   14 (63m ago)    12d
kube-proxy-xmsqj                      1/1     Running   14 (63m ago)    12d
kube-scheduler-master                 1/1     Running   14 (63m ago)    12d
[root@master aws-ebs-csi-driver]#
```

vi ebs-pv.yml

apiVersion: v1

kind: PersistentVolume

metadata:

name: ebs-pv

spec:

capacity:

storage: 5Gi

volumeMode: Filesystem

accessModes:

- ReadWriteOnce

persistentVolumeReclaimPolicy: Retain

storageClassName: manual

awsElasticBlockStore:

volumeID: vol-088916ab0a7ed453a

fsType: ext4

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: ebs-pv
spec:
  capacity:
    storage: 5Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: manual
  awsElasticBlockStore:
    volumeID: vol-088916ab0a7ed453a
    fsType: ext4
```

- **kubectl apply -f ebs-pv.yml**
- **kubectl get pv**

```
[root@master ~]# vi ebs-pv.yml
[root@master ~]# kubectl apply -f ebs-pv.yml
persistentvolume/ebs-pv created
[root@master ~]# kubectl get pv
NAME     CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS      CLAIM   STORAGECLASS   VOLUMEATTRIBUTESCLASS   REASON   AGE
ebs-pv   5Gi        RWO            Retain           Available           manual         <unset>                          10s
```

- **vi ebs-pvc.yml**

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: ebs-pvc

spec:

accessModes:

- ReadWriteOnce

storageClassName: manual

resources:

requests:

storage: 5Gi

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: ebs-pvc
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: manual
  resources:
    requests:
      storage: 5Gi
```

- kubectl apply -f ebs-pvc.yml
- kubectl get pvc

```
[root@master ~]# vi ebs-pvc.yml
[root@master ~]# kubectl apply -f ebs-pvc.yml
persistentvolumeclaim/ebs-pvc created
[root@master ~]# kubectl get pvc
NAME      STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
ebs-pvc   Bound    ebs-pv   5Gi        RWO            manual         <unset>                 13s
[root@master ~]#
```

vi ebs-pod.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: ebs-pod
spec:
  containers:
    - name: app
      image: nginx
      volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: ebs-storage
  volumes:
    - name: ebs-storage
      persistentVolumeClaim:
        claimName: ebs-pvc
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: ebs-pod
spec:
  containers:
    - name: app
      image: nginx
      volumeMounts:
        - mountPath: /usr/share/nginx/html
          name: ebs-storage
  volumes:
    - name: ebs-storage
      persistentVolumeClaim:
        claimName: ebs-pvc
```

- **kubectl apply -f ebs-pod.yml**
- **kubectl get pod ebs-pod**

```
[root@master ~]# kubectl get pod ebs-pod
NAME      READY   STATUS    RESTARTS   AGE
ebs-pod   1/1     Running   0          10s
[root@master ~]# kubectl get pod ebs-pod -o wide
NAME      READY   STATUS    RESTARTS   AGE   IP            NODE        NOMINATED NODE   READINESS GATE
ebs-pod   1/1     Running   0          25s   10.244.2.61   worker-02   <none>           <none>
[root@master ~]#
```

- **kubectl exec -it ebs-pod -- /bin/bash**
- **echo "Hello from EBS" > /usr/share/nginx/html/index.html**

```
[root@master ~]# kubectl exec -it ebs-pod -- /bin/bash
root@ebs-pod:/# echo "Hello from EBS" > /usr/share/nginx/html/index.html
root@ebs-pod:/# exit
exit
[root@master ~]#
```

- **kubectl delete pod ebs-pod**
- **kubectl apply -f ebs-pod.yml**

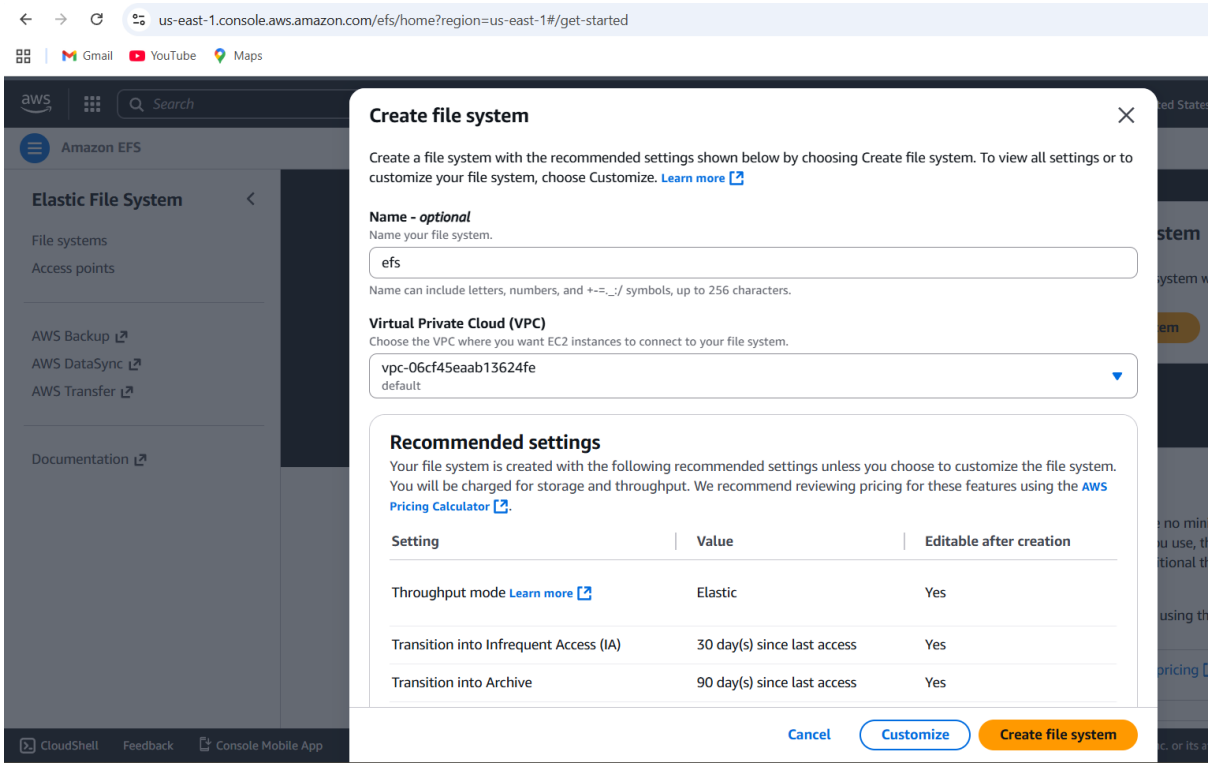
```
[root@master ~]# kubectl delete pod ebs-pod
pod "ebs-pod" deleted from default namespace
[root@master ~]# kubectl apply -f ebs-pod.yml
pod/ebs-pod created
```

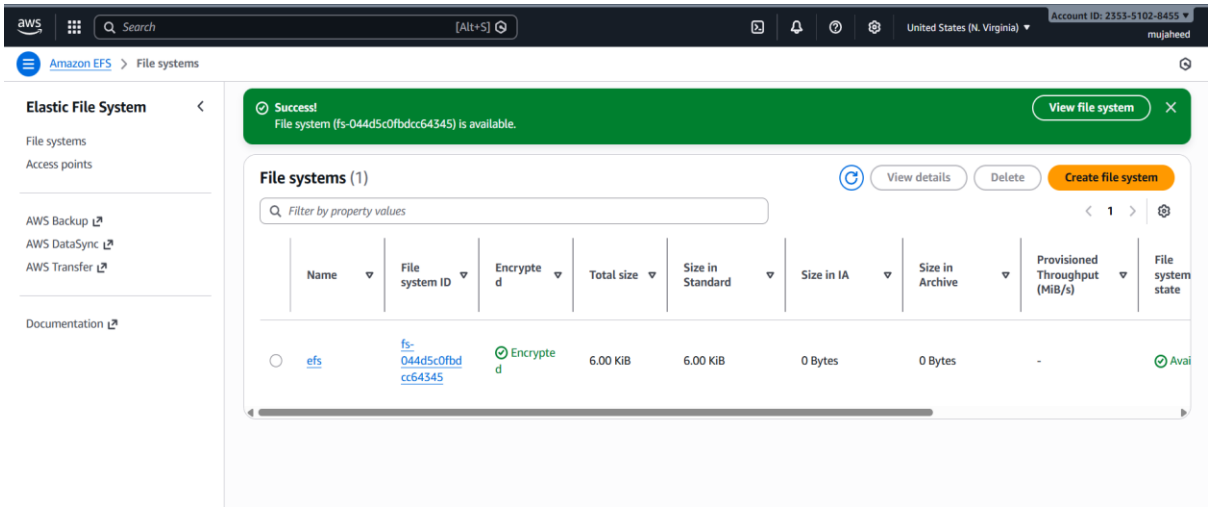
- **kubectl exec -it ebs-pod -- /bin/bash**
- **cat /usr/share/nginx/html/index.html**

```
[root@master ~]# kubectl exec -it ebs-pod -- /bin/bash
root@ebs-pod:/# cat /usr/share/nginx/html/index.html
Hello from EBS
root@ebs-pod:/#
```

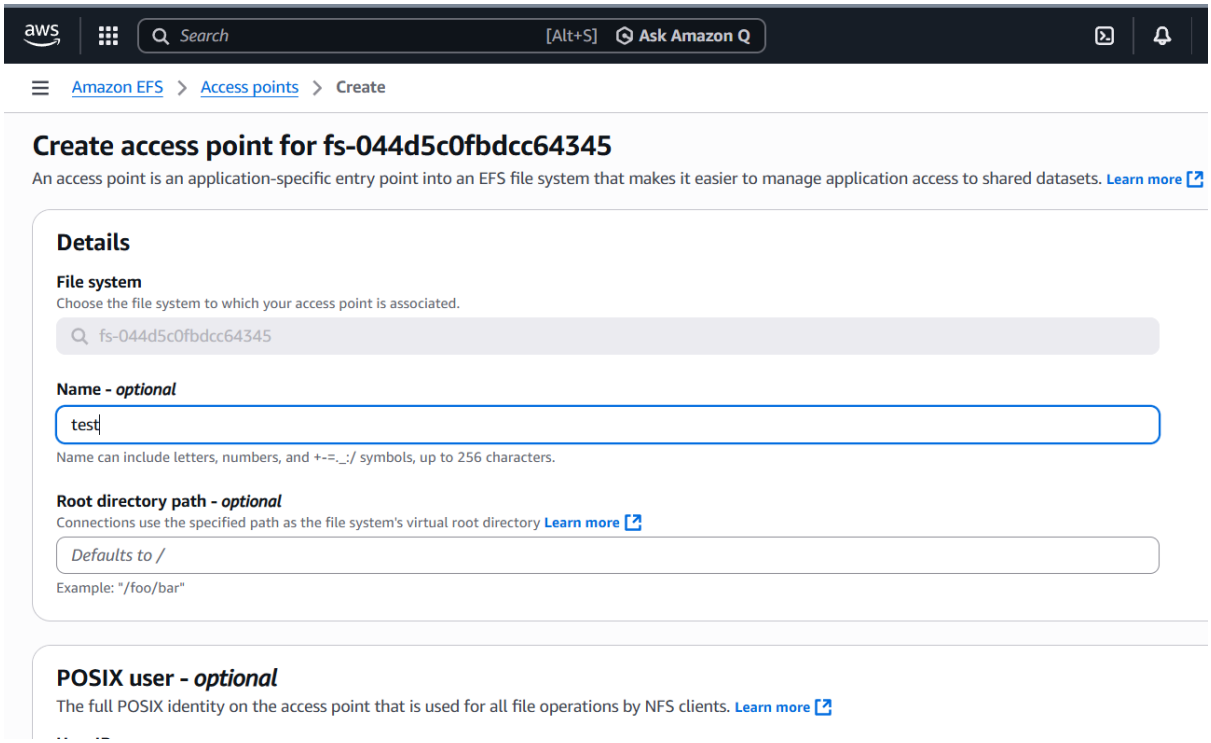
4. Set Up an Amazon EFS Volume and Attach it to Multiple Pods

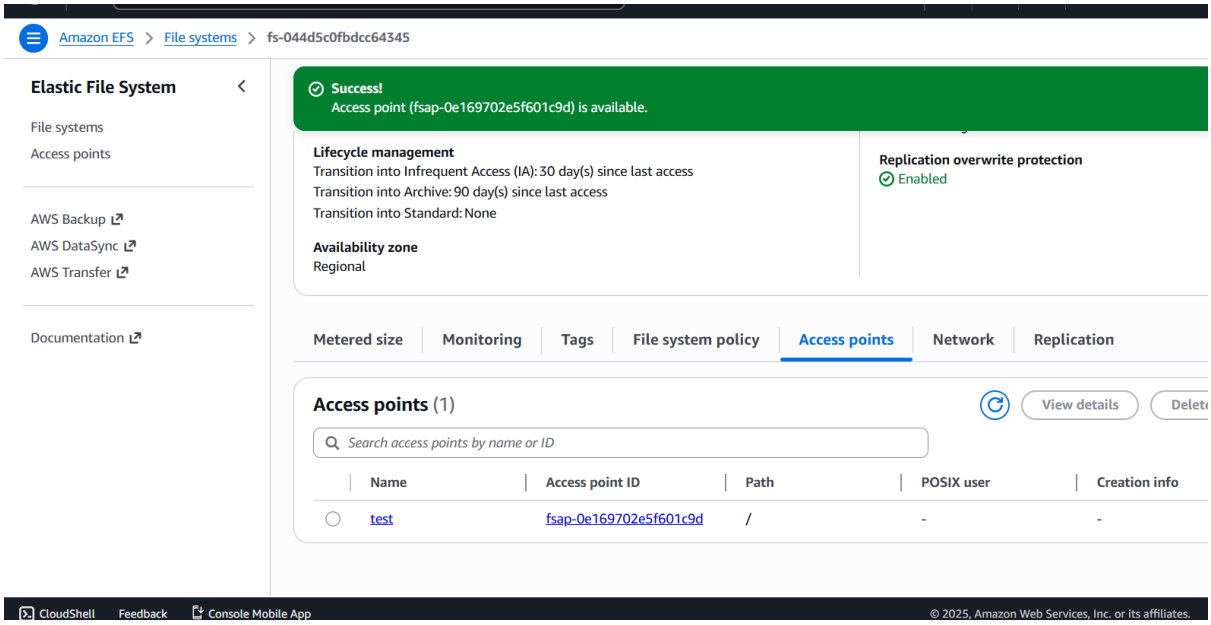
Go to AWS EFS and create a EFS





Create access point to your efs





Execute this command in master machine.

- **git clone https://github.com/kubernetes-sigs/aws-efs-csi-driver.git**
- **cd aws-efs-csi-driver/deploy/kubernetes/overlays/stable/**
- **kubectl apply -k .**

```
[root@master ~]# git clone https://github.com/kubernetes-sigs/aws-efs-csi-driver.git
Cloning into 'aws-efs-csi-driver'...
remote: Enumerating objects: 38679, done.
remote: Counting objects: 100% (2087/2087), done.
remote: Compressing objects: 100% (343/343), done.
remote: Total 38679 (delta 1896), reused 1746 (delta 1744), pack-reused 36592 (from 3)
Receiving objects: 100% (38679/38679), 35.45 MiB | 35.45 MiB/s, done.
Resolving deltas: 100% (20966/20966), done.
[root@master ~]# cd aws-efs-csi-driver/deploy/kubernetes/overlays/stable/
[root@master stable]# kubectl apply -k .
# Warning: 'bases' is deprecated. Please use 'resources' instead. Run 'kustomize edit fix' to
tically.
serviceaccount/efs-csi-controller-sa created
serviceaccount/efs-csi-node-sa created
clusterrole.rbac.authorization.k8s.io/efs-csi-external-provisioner-role created
clusterrole.rbac.authorization.k8s.io/efs-csi-external-provisioner-role-describe-secrets crea
clusterrole.rbac.authorization.k8s.io/efs-csi-node-role created
rolebinding.rbac.authorization.k8s.io/efs-csi-provisioner-binding-describe-secrets created
clusterrolebinding.rbac.authorization.k8s.io/efs-csi-node-binding created
clusterrolebinding.rbac.authorization.k8s.io/efs-csi-provisioner-binding created
deployment.apps/efs-csi-controller created
daemonset.apps/efs-csi-node created
csidriver.storage.k8s.io/efs.csi.aws.com created
[root@master stable]#
```

- **vi efs-sc.yml**

apiVersion: storage.k8s.io/v1

kind: StorageClass

metadata:

name: efs-sc

provisioner: efs.csi.aws.com

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: efs-sc
provisioner: efs.csi.aws.com
```

- **kubectl apply -f efs-sc.yml**

```
[root@master ~]# vi efs-sc.yml
[root@master ~]# kubectl apply -f efs-sc.yml
storageclass.storage.k8s.io/efs-sc created
```

- **vi efs-pv.yml**

apiVersion: v1

kind: PersistentVolume

metadata:

name: efs-pv

spec:

capacity:

storage: 5Gi

volumeMode: Filesystem

accessModes:

- ReadWriteMany

persistentVolumeReclaimPolicy: Retain

storageClassName: efs-sc

csi:

driver: efs.csi.aws.com

volumeHandle: fs-015d047384d2351b9::fsap-083503fedc8c5a29a

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: efs-pv
spec:
  capacity:
    storage: 5Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  storageClassName: efs-sc
  csi:
    driver: efs.csi.aws.com
    volumeHandle: fs-015d047384d2351b9::fsap-083503fedc8c5a29a
```

- kubectl apply -f efs-pv.yml

```
[root@master ~]# vi efs-pv.yml
[root@master ~]# kubectl apply -f efs-pv.yml
persistentvolume/efs-pv created
[root@master ~]#
```

- vi efs-pvc.yml

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: efs-pvc

spec:

accessModes:

- ReadWriteMany

storageClassName: efs-sc

resources:

requests:

storage: 5Gi

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: efs-pvc
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: efs-sc
  resources:
    requests:
      storage: 5Gi
```

- **kubectl apply -f efs-pvc.yml**
- **kubectl get pvc**

```
[root@master ~]# kubectl apply -f efs-pvc.yml
persistentvolumeclaim/efs-pvc created
[root@master ~]# kubectl get pvc
NAME      STATUS   VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEA
efs-pvc   Bound    efs-pv   5Gi        RWX            efs-sc         <unset>
[root@master ~]#
```

- **vi efs-pod.yml**

```
apiVersion: v1
kind: Pod
metadata:
  name: efs-pod1
spec:
  containers:
    - name: my-app
      image: nginx
      volumeMounts:
        - name: efs-volume
          mountPath: /efs-mount
  volumes:
    - name: efs-volume
      persistentVolumeClaim:
        claimName: efs-pvc
```

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: efs-pod1
spec:
  containers:
    - name: my-app
      image: nginx
      volumeMounts:
        - name: efs-volume
          mountPath: /efs-mount
  volumes:
    - name: efs-volume
      persistentVolumeClaim:
        claimName: efs-pvc
```

- **kubectl apply -f efs-pod.yml**
- **kubectl get pods -o wide**

```
[root@master ~]# kubectl get pods -o wide
NAME      READY   STATUS    RESTARTS   AGE   IP           NODE        NOMINATED NODE   READI
efs-pod   1/1     Running   0          16m   10.244.1.80  worker-01   <none>           <none
```

- **kubectl exec -it efs-pod – sh**
- **ls**

```
[root@master ~]# kubectl exec -it efs-pod -- sh
# ls
bin   dev                docker-entrypoint.sh  etc   lib    media  opt   root  sbin  sys  usr
boot  docker-entrypoint.d  efs-mount            home  lib64  mnt    proc  run   srv   tmp  var
```

- **cd efs-amount**
- **touch pod1**

```
# cd efs-mount
# touch pod1
# exit
[root@master ~]#
```

Create an another pod

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: efs-pod1
spec:
  containers:
    - name: my-app
      image: nginx
      volumeMounts:
        - name: efs-volume
          mountPath: /efs-mount
  volumes:
    - name: efs-volume
      persistentVolumeClaim:
        claimName: efs-pvc
```

- **kubectl apply -f efs-pod .yml**
- **kubectl get pods -o wide**

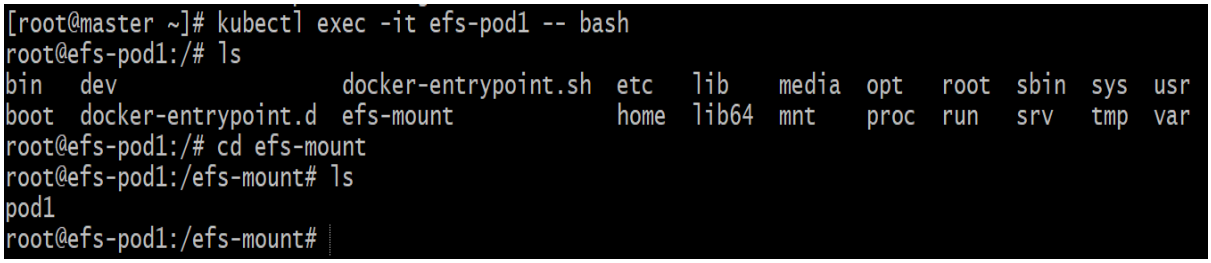
```
[root@master ~]# vi efs-pod.yml
[root@master ~]# kubectl apply -f efs-pod.yml
pod/efs-pod1 created
[root@master ~]# kubectl get pods -o wide
NAME       READY   STATUS    RESTARTS   AGE   IP            NODE        NOMINATED NODE   READINESS GATES
efs-pod    1/1     Running   0          36m   10.244.1.80   worker-01   <none>           <none>
efs-pod1   1/1     Running   0          8s    10.244.2.72   worker-02   <none>           <none>
[root@master ~]#
```

Now the pod has been created in worker-02

Login into the second pod.

- **kubectl exec -it efs-pod1 -- bash**
- **cd efs-mount**
- **ls**

you should see the file which you have created in the node1 machine pod.



5. Implement and Test Liveness and Readiness Probes in a Kubernetes Pod

Liveliness:

- **vi liveness.yml**

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
    - name: my-app
      image: nginx            # Image name
      ports:
        - containerPort: 80


      livenessProbe:
```

httpGet: # HTTP GET request (can also use tcpSocket or exec)

path: / # Health check path

port: 80

initialDelaySeconds: 15 # Wait 15 seconds before first liveness check

periodSeconds: 10 # Check every 10 seconds

```
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
  - name: my-app
    image: nginx              #Image Name
    ports:
    - containerPort: 80
    livenessProbe:
      httpGet:                #get request or we
        path: /                #health check path
        port: 80
      initialDelaySeconds: 15    #It will wait
      periodSeconds: 10
~
```

- **kubectl apply -f liveness.yml**
- **kubectl get pods**

make any error it will restart

- **kubectl exec -it probe-test-pod -- rm -f /usr/share/nginx/html/index.html**

```
[root@master ~]# kubectl get pod my-pod
NAME     READY   STATUS    RESTARTS   AGE
my-pod   1/1     Running   0          58s
[root@master ~]# kubectl exec -it probe-test-pod -- rm -f /usr/share/nginx/html/index.html
```

```
[root@master ~]# kubectl get pod my-pod
NAME      READY   STATUS    RESTARTS      AGE
my-pod    1/1     Running   1 (39s ago)   3m9s
[root@master ~]#
```

Readiness:

- **vi readiness.yml**

apiVersion: v1

kind: Pod

metadata:

name: my-pod

spec:

containers:

- name: my-app

image: nginx

ports:

- containerPort: 80

readinessProbe:

httpGet:

path: /index.html

port: 80

initialDelaySeconds: 10

periodSeconds: 5

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: my-pod
spec:
  containers:
    - name: my-app
      image: nginx
      ports:
        - containerPort: 80

      readinessProbe:
        httpGet:
          path: /index.html
          port: 80
        initialDelaySeconds: 10
        periodSeconds: 5
```

- kubectl apply -f readiness.yml
- kubectl get pods

```
[root@master ~]# kubectl get pods
NAME             READY   STATUS      RESTARTS   AGE
efs-pod          0/1     Completed   0          94m
efs-pod1         0/1     Completed   0          58m
readiness-test   1/1     Running     0          68s
```

Make changes to get error

- kubectl exec -it readiness-test -- rm /usr/share/nginx/html/index.html
- kubectl get pods

the pod was not ready

```
[root@master ~]# kubectl exec -it readiness-test -- rm /usr/share/nginx/html/index.html
[root@master ~]# kubectl get pods
```

```
[root@master ~]# kubectl get pods
NAME             READY   STATUS      RESTARTS   AGE
efs-pod          0/1     Completed   0          95m
efs-pod1         0/1     Completed   0          59m
readiness-test   0/1     Running     0          2m10s
[root@master ~]#
```