CSI NFS Dynamic Provisioning Lab

> Compatible with Minikube, Kind, or any Kubernetes 1.20+

Prerequisites

- Kubernetes cluster
- `kubectl` installed
- Linux-based system (for NFS setup)
- Internet access for pulling container images

Step 1: Install & Configure NFS Server (On Host or VM)

On your **host** or any reachable machine (e.g. Minikube VM), install NFS server:

```
sudo apt update
sudo apt install nfs-kernel-server -y

sudo mkdir -p /srv/nfs/kubedata
sudo chown nobody:nogroup /srv/nfs/kubedata
sudo chmod 777 /srv/nfs/kubedata

echo "/srv/nfs/kubedata *(rw,sync,no_subtree_check,no_root_squash)" | sudo tee -a
/etc/exports
sudo exportfs -rav
```

Start the service:

```
sudo systemctl enable nfs-server
sudo systemctl start nfs-server
```

> 💡 On Minikube, use `minikube ssh` and perform the above commands inside the VM.

Step 2: Deploy the NFS CSI Driver

Install the CSI NFS driver (official):

```
kubectl apply -k "github.com/kubernetes-csi/csi-driver-
nfs/deploy/kubernetes/overlays/stable?ref=release-1.6"
```

Verify:

```
kubectl get pods -n kube-system -l app=csi-nfs-controller
kubectl get daemonset -n kube-system -l app=csi-nfs-node
```

Step 3: Create a StorageClass

Create a file called `nfs-sc.yaml`:

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: nfs-csi
provisioner: nfs.csi.k8s.io
parameters:
  server: <NFS_SERVER_IP>
  share: /srv/nfs/kubedata
reclaimPolicy: Retain
volumeBindingMode: Immediate
```

> 🔁 Replace `<NFS_SERVER_IP>` with the IP address of the machine where the NFS server is running.

Apply it:

```
kubectl apply -f nfs-sc.yaml
```

Step 4: Create a PVC + Pod using the NFS StorageClass

Create `nfs-pvc-pod.yaml`:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nfs-pvc
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: nfs-csi
  resources:
    requests:
      storage: 1Gi
---
apiVersion: v1
kind: Pod
metadata:
  name: nfs-app
spec:
  containers:
  - name: app
    image: busybox
    command: [ "sh", "-c", "echo 'Hello from NFS!' > /data/hello.txt && sleep
3600" ]
    volumeMounts:
    - mountPath: /data
      name: nfs-vol
  volumes:
  - name: nfs-vol
    persistentVolumeClaim:
      claimName: nfs-pvc
```

Apply it:

```
kubectl apply -f nfs-pvc-pod.yaml
```

Step 5: Verify NFS Volume Mount and File Write

Check logs or exec into pod:

```
kubectl exec -it nfs-app -- cat /data/hello.txt
```

You should see:

```
Hello from NFS!
```

Then on the NFS server:

```
cat /srv/nfs/kubedata/*/data/hello.txt
```

You'll see the same file — confirming dynamic provisioning!

Cleanup

```
kubectl delete -f nfs-pvc-pod.yaml
kubectl delete storageclass nfs-csi
```

What You Learned

- How to set up an **NFS server**
- Install the **NFS CSI driver**
- Define a **StorageClass** that supports dynamic provisioning
- Use a **PVC** and automatically provision NFS-backed **volumes**
- Verify **cross-node shared storage (ReadWriteMany)**

August 22, 2025