Create Shared Volume with NFS

Steps to Build an NFS Server

1. Install NFS Server on Linux (VM or Kubernetes Node)

If you're using a VM or any Linux-based system as the NFS server, you first need to install the NFS server package.

For Ubuntu/Debian:

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

2. Create a Directory to Share

Create a directory that will be shared over NFS. For example:

```
sudo mkdir -p /mnt/data
sudo chmod 777 /mnt/data
sudo chown nobody:nogroup /mnt/data
```

3. Configure NFS Exports

You'll need to configure the NFS exports to specify which directories you want to share and with which permissions.

Edit the NFS exports file:

```
sudo nano /etc/exports
```

Add the following line to the file, allowing any IP to access the directory (replace `*` with specific IP ranges for security):

```
/mnt/data *(rw,sync,no_subtree_check)
```

- `rw`: Allows read-write access.
- `sync`: Ensures data is written to disk before returning a response.
- `no_subtree_check`: Prevents checking the file system for each request (to improve performance).

4. Export the Shared Directory

After editing the `/etc/exports file`, export the shared directories with the following command:

```
sudo exportfs -a
```

To confirm that the export was successful, run:

```
sudo exportfs -v
```

5. Start NFS Server

Start and enable the NFS server to run on boot:

For Ubuntu/Debian:

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

6. Check if the NFS Server Service is Running

Log in to the NFS server and verify that the required services are active.

For most systems:

```
sudo systemctl status nfs-server
```

Test Port Accessibility

Ensure that the required NFS ports are open on the NFS server and accessible from the client.

Check the NFS Ports:

```
sudo rpcinfo -p
```

Look for services like `nfs`, `mountd`, and `portmapper` in the output.

Example:

```
   program vers proto   port  service
    100000    4   tcp    111  portmapper
    100005    1   udp  20048  mountd
    100003    3   tcp   2049  nfs
```

7. Open Ports in Firewall (if applicable)

If you're using a firewall, open the necessary ports to allow NFS traffic. The typical ports used for NFS are `2049` (NFS) and `111` (RPC).

For UFW (Ubuntu firewall):

```
sudo ufw allow from any to any port nfs
```

8. Test the NFS Share

From another machine (which will act as the client), test the NFS share.

Mount the NFS share to a local directory (replace `nfs-server-ip` with the IP of the NFS server):

```
sudo mount -t nfs nfs-server-ip:/mnt/data /mnt
```

You should now be able to read and write files to the `/mnt/data` directory over NFS.

Check the IP Address on the NFS Server

If you have direct access to the NFS server, run the following command to find its IP address:

Use `ip` Command:

```
ip addr show
```

- Look for the network interface connected to your local network (e.g., eth0, ens0, or similar).
- Example output:

```
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    inet 192.168.1.10/24 brd 192.168.1.255 scope global eth0
```

Use `hostname -I`:

```
hostname -I
```

This will display all IP addresses assigned to the server.

Steps to Expose NFS in Kubernetes

Now that you have the NFS server set up, you can use it as shared storage in your Kubernetes cluster.

1. Create the Persistent Volume (PV)

Create a `PersistentVolume` resource in Kubernetes that points to your NFS server. Here's an example YAML:

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: nfs-pv
spec:
  capacity:
    storage: 10Gi
  accessModes:
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  nfs:
    path: /mnt/data  # Path to the directory on NFS server
    server: <nfs-server-ip>  # Replace with the NFS server's IP address
```

2. Create the Persistent Volume Claim (PVC)

Next, create a `PersistentVolumeClaim` (PVC) to request the shared volume:

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nfs-pvc
spec:
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi
```

3. Use the PVC in a Pod

Finally, mount the `PersistentVolumeClaim` into your pod as follows:

```
apiVersion: v1
kind: Pod
metadata:
  name: nfs-client-pod
spec:
  containers:
    - name: nfs-client-container
      image: busybox
      command: ["sleep", "3600"]
      volumeMounts:
        - mountPath: "/mnt/data"
          name: nfs-volume
  volumes:
    - name: nfs-volume
      persistentVolumeClaim:
        claimName: nfs-pvc
```

4. Apply the Configuration

Apply the `PersistentVolume`, `PersistentVolumeClaim`, and Pod configurations:

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

5. Verify the Setup

Once the Pod is running, verify that the NFS share is mounted inside the container:

```
kubectl exec -it nfs-client-pod -- df -h
```

You should see `/mnt/data` mounted and showing the space usage of the NFS share.

Remove Unused NFS

To remove or delete an unused NFS export entry that you no longer need, you can follow these steps:

Identify the Export Entry

First, identify the export entry you want to remove by running `exportfs -v` with `sudo`:

```
sudo exportfs -v
```

This command will list all the NFS exports currently configured on your system.

Remove the Export Entry

To remove an NFS export entry, you use the `exportfs` command with the `-u` option followed by the path of the export. Here’s the general syntax:

```
sudo exportfs -u [options] export_path
```

- Replace `export_path` with the path you want to remove from the NFS exports.
- `-u` option is used to unexport (remove) the specified export path.

For example, if you see an export like `/mnt/nfs_share *(rw,fsid=0,sync,no_subtree_check)` that you want to remove, you would use:

```
sudo exportfs -u /mnt/nfs_share
```

Restart NFS Service

Restart NFS Service: After removing the export entry, restart the NFS server to apply changes:

```
sudo systemctl restart nfs-server
```

Verify Removal

After running the `exportfs -u` command, you can verify that the export entry has been removed by running `exportfs -v` again:

```
sudo exportfs -v
```

The entry you removed should no longer appear in the list of NFS exports.

Additional Notes

- Make sure to review the list of NFS exports carefully before removing any entry to avoid unintentionally deleting important configurations.
- Changes made with `exportfs` are immediate and affect NFS clients accessing the exports. Ensure any necessary changes are communicated if NFS exports are actively used.

By following these steps, you can effectively remove an unused NFS export entry from your system using `exportfs` on Ubuntu or any Linux distribution that supports NFS.

Uninstall NFS Service

To uninstall NFS (Network File System) on Ubuntu, you need to remove the NFS server components and any associated configuration. Here’s how you can do it:

Uninstall NFS Server Components

1. **Stop NFS Services**: First, stop the NFS server services to prevent them from running:

   ```
   sudo systemctl stop nfs-server
   ```

2. **Remove NFS Packages**: Use `apt` to remove the NFS server packages. The specific package names might vary depending on your Ubuntu version and NFS setup. Typically, you may have packages like `nfs-kernel-server`:

   ```
   sudo apt purge nfs-kernel-server
   ```

3. **Clean Up Configuration Files**: After purging the packages, you can clean up any remaining configuration files:

   ```
   sudo apt autoremove
   ```

Uninstall NFS Client Components (Optional)

If you also want to remove NFS client components:

1. **Stop NFS Client Services**:

   ```
   sudo systemctl stop nfs-client.target
   ```

2. **Remove NFS Client Packages**: Again, the package names might vary, but typically it includes `nfs-common`:

   ```
   sudo apt purge nfs-common
   ```

3. **Clean Up Configuration Files**:

   ```
   sudo apt autoremove
   ```

Verify Removal

After performing the above steps, NFS should be uninstalled from your system. You can verify by checking if NFS-related services are stopped and if the packages are no longer installed:

```
sudo systemctl status nfs-server     # Check NFS server status (should show inactive or not found)
sudo systemctl status nfs-client     # Check NFS client status (if applicable)
dpkg -l | grep nfs                   # Check if NFS packages are listed (should be empty)
```

Additional Considerations

- **Configuration Files**: If you have customized configuration files (`/etc/exports` for NFS server or `/etc/fstab` for NFS mounts), you may want to manually remove or restore them as needed.
- **Data and Shares**: Uninstalling NFS does not delete any data stored on NFS shares. Ensure you have backed up any critical data before uninstalling if necessary.

By following these steps, you can uninstall NFS server and client components from your Ubuntu system effectively. Adjust the package names as needed based on your specific installation.

Connect a Node to NFS Server

1. **Install NFS Client Utilities (if not already installed)**: Ensure that NFS client utilities are installed on the client node. Install them if needed:

   ```
   sudo apt update
   sudo apt install nfs-common
   ```

2. **Mount NFS Share**: Create a directory on the client node where you want to mount the NFS share (e.g., `/mnt/nfs_client`):

   ```
   sudo mkdir -p /mnt/nfs_client
   ```

3. **Mount the NFS Share**: Mount the NFS share from the server to the client directory:

   ```
   sudo mount -t nfs server_ip:/mnt/nfs_share /mnt/nfs_client
   ```

   Replace `server_ip` with the IP address of your NFS server.
4. **Verify Mount**: Check that the NFS share is mounted correctly:

   ```
   mount | grep nfs
   ```

5. **Automount NFS Share (Optional)**: If you want the NFS share to be mounted automatically on boot, you can add an entry to `/etc/fstab` on the client node:

   ```
   server_ip:/mnt/nfs_share /mnt/nfs_client nfs defaults 0 0
   ```

   Save the file and run:

   ```
   sudo mount -a
   ```

6. **Reload `daemon`**

   ```
   systemctl daemon-reload
   ```

Testing the NFS Mount

- Create a file on the NFS share from the client node to ensure read and write permissions are correctly set up:

```
echo "Test file" | sudo tee /mnt/nfs_client/test.txt
```

- Check if the file appears on the NFS server at `/mnt/nfs_share`.

By following these steps, you should be able to connect another node (client) to your NFS server and access shared directories. Adjust the IP addresses, paths, and configuration options (`rw`, `sync`, etc.) according to your specific setup and security requirements.