**Lesson 06 Demo 06**

**Configuring Pod Using NFS-Based PV and PVC**

**Objective:** To configure a pod using NFS-based PersistentVolume (PV) and PersistentVolumeClaim (PVC) for more efficient storage management

**Tools required:** kubeadm, kubectl, kubelet, and containerd

**Prerequisites:** A Kubernetes cluster should already be set up (refer to the steps provided in Lesson 02, Demo 01 for guidance)

Steps to be followed:

1. Configure the NFS kernel server
2. Set the permissions
3. Configure the NFS common on client machines
4. Create PersistentVolume
5. Create PersistentVolumeClaim
6. Create the deployment for MySQL

**Step 1: Configure the NFS kernel server**

1. Create a directory on the **worker-node-1** using the following command:

**sudo mkdir /mydbdata**

**A black background with white text

Description automatically generated**

1. Install the NFS kernel server on the machine:

**sudo apt install nfs-kernel-server**

**A screenshot of a computer

Description automatically generated**

**Step 2: Set the permissions**

1. On the **worker-node-1**, open the exports file in the /etc directory using the following command:

**sudo nano /etc/exports**

****

1. Inside the file, append the following code:

**/mydbdata \*(rw,sync,no\_root\_squash)**

**A screen shot of a computer program

Description automatically generated**

1. Use the **cat** command to view the file:

**A screenshot of a computer program

Description automatically generated**

1. Export all shared directories defined in the **/etc/exports** file using the following command:

**sudo exportfs -rv**

**A screen shot of a computer program

Description automatically generated**

1. Make the folder publicly accessible by changing its owner user and group using the following command:

**sudo chown nobody:nogroup /mydbdata/**

**A screenshot of a computer program

Description automatically generated**

1. Assign full permissions to read, write, and execute files in this directory using the following command:

**sudo chmod 777 /mydbdata/**

**A screenshot of a computer

Description automatically generated**

1. Restart the NFS kernel server to apply the changes using the following command:  
   **sudo systemctl restart nfs-kernel-server**  
     
   A screenshot of a computer program

   Description automatically generated
2. Retrieve the internal IP of the node where the NFS server is installed using the following command:

**ip a**

**A screenshot of a computer program

Description automatically generated**After running this command, look for the relevant IP address in the output. This IP will be used to associate the PV with the NFS server.

|  |
| --- |
| **Note**: Save the IP address to use in the next steps |

**Step 3: Configure the NFS common on client machines**

|  |
| --- |
| **Note:** Perform the below steps on each worker node intended for sharing |

1. Run the following command to install the NFS common package:

**sudo apt install nfs-common**

**A screenshot of a computer program

Description automatically generated**

1. Execute the following commands to refresh the NFS common service and verify its status:  
   **sudo rm /lib/systemd/system/nfs-common.service**  
   **sudo systemctl daemon-reload**  
     
   A screenshot of a computer

   Description automatically generated
2. Restart the NFS client service and check its status using the following commands:

**sudo systemctl restart nfs-common**

**sudo systemctl status nfs-common**

A screen shot of a computer program

Description automatically generated  
  
A computer screen shot of a program

Description automatically generated

|  |
| --- |
| **Note:** These steps are to be performed on both worker nodes as shown in the screenshots above. |

**Step 4: Create PersistentVolume**

1. On the **master** node, create the YAML file using the following command:  
   **nano** **pv.yaml**  
     
   A black and white screen with white text

   Description automatically generated
2. Add the following code to the **pv.yaml** file:  
   **apiVersion: v1**

**kind: PersistentVolume**

**metadata:**

**name: test**

**labels:**

**app: wordpress**

**spec:**

**capacity:**

**storage: 10Gi**

**accessModes:**

**- ReadWriteMany**

**nfs:**

**server: YOUR\_NFS\_SERVER\_IP\_HERE**

**path: "/mydbdata"  
  
A screenshot of a computer

Description automatically generated**

|  |
| --- |
| **Note:** Replace **YOUR\_NFS\_SERVER\_IP\_HERE** with the internal IP of the NFS server from step **2.8** as shown in the screenshot above |

1. Apply the configuration defined in **pv.yaml** using the following command:  
   **kubectl apply -f pv.yaml  
     
   A screen shot of a computer

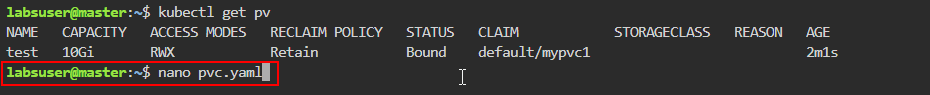
   Description automatically generated**
2. List all the **PVs** in the cluster using the following command:

**kubectl get pv**

**A screenshot of a computer screen

Description automatically generated**

**Step 5: Create PersistentVolumeClaim**

1. Create the YAML file using the following command:  
   **nano pvc.yaml**
2. Add the following code to the **pvc.yaml** file:

**apiVersion: v1**

**kind: PersistentVolumeClaim**

**metadata:**

**name: mypvc1**

**labels:**

**app: wordpress**

**spec:**

**accessModes:**

**- ReadWriteMany**

**resources:**

**requests:**

**storage: 6Gi**

**A computer screen with a black background

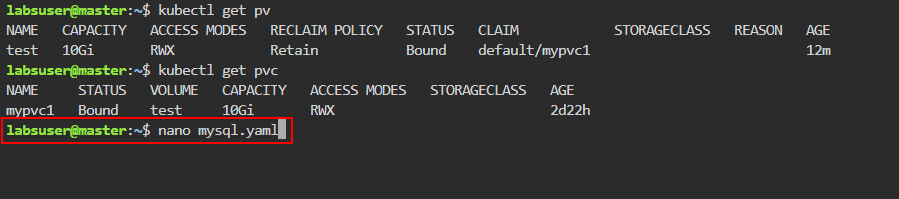
Description automatically generated**

1. Apply the configuration defined in **pvc.yaml** using the following command:  
   **kubectl apply -f pvc.yaml**  
     
   A black screen with white text

   Description automatically generated
2. List all the **PVs** and **PVCs** in the cluster using the following commands:  
   **kubectl get pv  
   kubectl get pvc**  
   A black background with white text

   Description automatically generated

**Step 6: Create the deployment for MySQL**

1. Create the YAML file using the following command:  
   **nano mysql.yaml**
2. Add the following code to the **mysql.yaml** file:

**apiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: test-mysql**

**labels:**

**app: wordpress**

**spec:**

**selector:**

**matchLabels:**

**app: wordpress**

**tier: mysql**

**strategy:**

**type: Recreate**

**template:**

**metadata:**

**labels:**

**app: wordpress**

**tier: mysql**

**spec:**

**containers:**

**- image: mysql:5.6**

**name: mysql**

**env:**

**- name: MYSQL\_ROOT\_PASSWORD**

**value: password**

**ports:**

**- containerPort: 3306**

**name: mysql**

**volumeMounts:**

**- name: myvol1**

**mountPath: /var/lib/mysql**

**volumes:**

**- name: myvol1**

**persistentVolumeClaim:**

**claimName: mypvc1**

**A computer screen with a black background

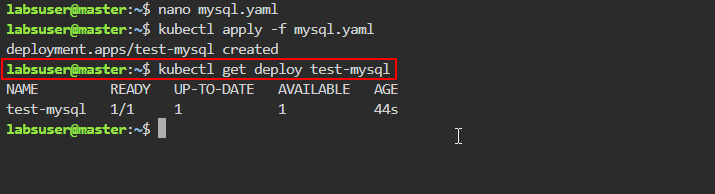
Description automatically generated**

**A computer screen shot of a black screen

Description automatically generated**

1. Apply the configuration defined in **mysql.yaml** using the following command:  
   **kubectl apply -f mysql.yaml**  
     
   A black screen with white text

   Description automatically generated
2. Check the status of deployment using the following command:

**kubectl get deploy test-mysql**  
  


1. Check the status of the pod using the following command:

**kubectl get pod -l app=wordpress**  
  
A screen shot of a computer

Description automatically generated

|  |
| --- |
| **Note:** Save the **name** of the pod for the next step |

1. View detailed information about the pod using the following command:

**kubectl describe pod <pod-name>**

**A screenshot of a computer

Description automatically generated**

A screenshot of a computer program

Description automatically generated

|  |
| --- |
| **Note:** Replacethe **<pod-name>** with the **name** of your pod, as shown in the screenshots above |

By following these steps, you have successfully configured a Kubernetes pod using   
NFS-based PV and PVC for efficient storage management.