Kubernetes CSI Demo Guide

This repository contains a fully-automated Kubernetes demo script showcasing the following capabilities:

- **CSI Driver Integration**
- **PersistentVolume (PV)** and **PersistentVolumeClaim (PVC)** usage
- **VolumeSnapshot** creation
- **PVC Cloning** from snapshots
- **Application Deployment** and **Service Exposure** using NodePort
- **NGINX Web Server Integration with Persistent Storage**

📁 **Repository Structure**

```
├── deployment/
│   ├── nginx-pvc.yaml
│   ├── nginx-deployment.yaml
│   └── nginx-service.yaml
├── snapshot/
│   ├── volumesnapshotclass.yaml
│   └── volumesnapshot.yaml
├── clone/
│   ├── clone-pvc-from-snapshot.yaml
│   ├── nginx-deployment-clone.yaml
│   └── nginx-service-clone.yaml
├── webpages/
│   └── index.html (or multiple .html files)
├── vmstore-csi-demo.sh
├── README.md
```

How to Run the Demo

Prerequisites

- A working Kubernetes cluster with CSI snapshot and clone support
- kubectl configured
- CSI Snapshot CRDs installed
- NodePort access enabled
-

Steps to Execute

Create your own namespace ::

kubectl create ns tapas
kubectl config set-context --current --namespace=tapas

This will execute all the data in the namespace that you have created.

NOTE:Change the directory to /demo/webserver/
NOTE:mv websites to webpages as we have all the data pull from webpages

chmod +x webserver-cp-demo-v2.sh
./webserver-cp-demo-v2.sh

What Happens in the Script

1. **PVC Creation** – via nginx-pvc.yaml
2. **NGINX App Deployment** – mounts the PVC
3. **NodePort Service Exposure**
4. **Random HTML Copy** into the NGINX pod —> If you have multiple HTML sites in the folder /webpages it will randomly pull websites.
5. **PVC Snapshot Creation**
6. **Deployment from Snapshot (Cloning)**
7. **Access URLs** printed for both original and cloned apps

1. Deploying Persistent Volume and App

kubectl apply -f deployment/nginx-pvc.yaml

- **What it does**: Applies a YAML that defines a PersistentVolumeClaim (PVC).
- **Purpose**: This PVC is dynamically bound to a PersistentVolume (PV) provisioned by the **CSI (Container Storage Interface) driver**.
- **Sales Pitch**: Demonstrates **dynamic provisioning**—no manual storage setup needed!

kubectl apply -f deployment/nginx-deployment.yaml

- **What it does**: Deploys the NGINX container using the PVC.
- **Link**: The pod's volumeMounts refer to the PVC created in the last step.
- **Technical Note**: This creates a pod with the app storing its web files on persistent storage.

kubectl apply -f deployment/nginx-service.yaml

- **What it does**: Exposes NGINX with a **NodePort** service.
- **Use Case**: Enables external access to the app via a URL/IP.

2. Check Pod Status

kubectl get pods | grep -E ngin*

- **What it does**: Filters pods related to nginx.
- **Support Tip**: Use this to confirm pod status (e.g., Running, CrashLoopBackOff).

3. Copy HTML Content to NGINX Pod

kubectl cp index.html <namespace>/<nginx-pod>:/usr/share/nginx/html/index.html

- **Actual Execution**: The script dynamically detects:
  - Namespace (default or set in kubeconfig)
  - Pod name with label app=nginx
- **Result**: The static HTML file is injected into the pod's mounted volume.
- **Technical Point**: Since the PVC is used, the copied file persists across pod restarts.

4. Snapshot the PVC

kubectl apply -f snapshot/volumesnapshotclass.yaml
kubectl apply -f snapshot/volumesnapshot.yaml

- **What it does**:
  - First command registers the snapshot capability (usually per CSI driver).
  - Second creates a snapshot of the PVC in use by the NGINX pod.
- **Purpose**: Acts like a restore point or backup image of the current disk state.
- **Sales Angle**: Enables **instant rollbacks** and **dev/test cloning** without data loss.

5. View the Original App Service

kubectl get pods,svc -o wide | grep ngin*

- **Why it matters**:
  - Shows pod IP, node, and service port.
  - Script parses this and prints the web access URL like http://<node-ip>:<nodeport>.
- **Support Help**: Useful when helping users access services deployed via NodePort.

6. Clone the App from the Snapshot

kubectl apply -f clone/clone-pvc-from-snapshot.yaml

- **What it does**: Creates a new PVC from the existing VolumeSnapshot.
- **Key CSI Feature**: This PVC is a **restored version** of the original data.

kubectl apply -f clone/nginx-deployment-clone.yaml
kubectl apply -f clone/nginx-service-clone.yaml

- **Deploys** a new pod and service using the **cloned volume**.
- **Result**: This creates a **separate NGINX instance** with the same data

as the original.

7. Access the Cloned App

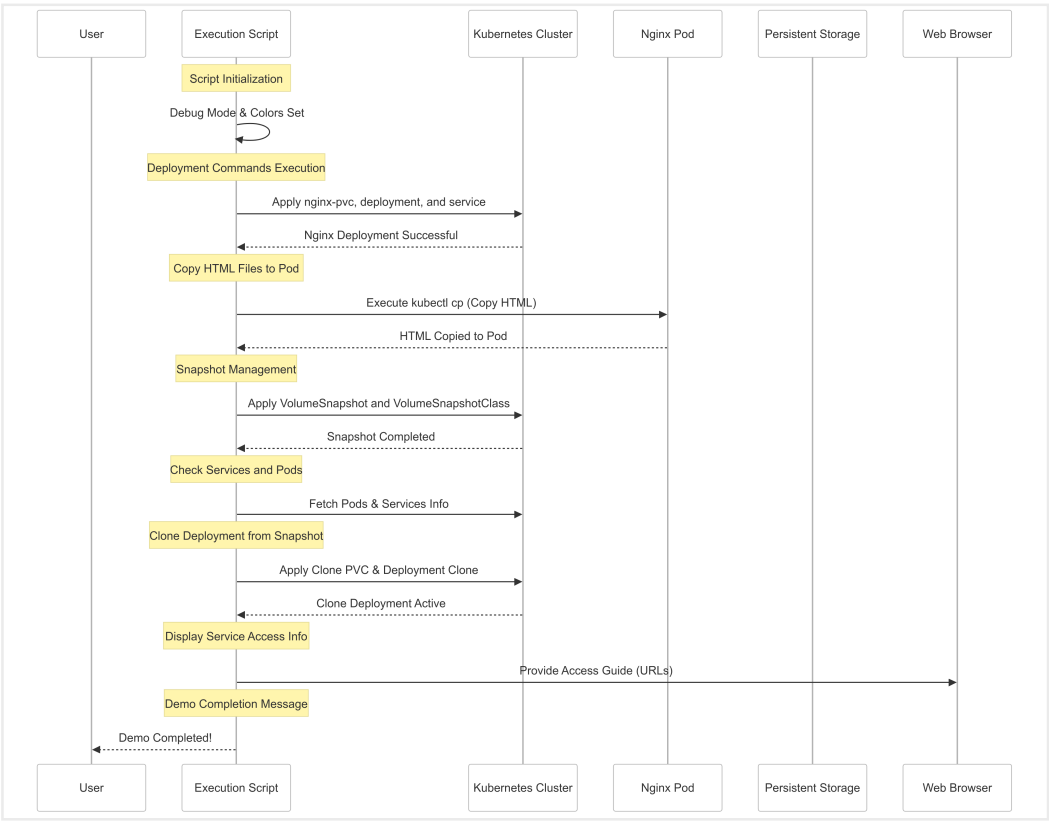
```
kubectl get pods,svc -o wide | grep clone*
```

- **Output**: Shows pod and service for the cloned app.
- **Script Enhancement**:
  - Extracts the pod's node and service's NodePort.
  - Prints: http://<node-ip>:<nodeport> for easy access.
- **Sales/Support Benefit**: Easy demo of cloning and data consistency with CSI.

8. Dynamic Access Resolution

```
display_access_guide "nginx" "nginx-service"
display_access_guide "clone" "nginx-clone-service"
```

- These functions:
  - Get pod's node name and IP.
  - Pull NodePort of the service.
  - Construct a usable web URL.
- **No Manual Lookup Needed**!


User
Execution Script
Kubernetes Cluster
Nginx Pod
Persistent Storage
Web Browser
Script Initialization
Debug Mode & Colors Set
Deployment Commands Execution
Apply nginx-pvc, deployment, and service
Nginx Deployment Successful
Copy HTML Files to Pod
Execute kubectl cp (Copy HTML)
HTML Copied to Pod
Snapshot Management
Apply VolumeSnapshot and VolumeSnapshotClass
Snapshot Completed
Check Services and Pods
Fetch Pods & Services Info
Clone Deployment from Snapshot
Apply Clone PVC & Deployment Clone
Clone Deployment Active
Display Service Access Info
Provide Access Guide (URLs)
Demo Completion Message
Demo Completed!
User
Execution Script
Kubernetes Cluster
Nginx Pod
Persistent Storage
Web Browser