**Lesson 06 Demo 01**

**Sharing Data Between Containers in the Same Pod**

**Objective:** To demonstrate data-sharing between containers in the same pod through hostPath volumes for mounting pod files onto the file system of the host node

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

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

Steps to be followed:

1. Configure and launch the pod with the shared volume
2. Interact with the shared volume from both containers
3. Test data persistence and sharing capability

**Step 1: Configure and launch the pod with the shared volume**

1. Open the YAML configuration file with the following command: **nano emptydir.yaml  
     
   A black rectangle with white text

   Description automatically generated**
2. Enter the following code into the **emptydir.yaml** file to define the pod with two containers sharing a volume:  
   **apiVersion: v1**

**kind: Pod**

**metadata:**

**name: container-share-volume**

**spec:**

**containers:**

**- name: container1**

**image: centos:7**

**command:**

**- "bin/bash"**

**- "-c"**

**- "sleep 10000"**

**volumeMounts:**

**- name: container-volume**

**mountPath: "/tmp/xchange"**

**- name: container2**

**image: centos:7**

**command:**

**- "bin/bash"**

**- "-c"**

**- "sleep 10000"**

**volumeMounts:**

**- name: container-volume**

**mountPath: "/tmp/data"**

**volumes:**

**- name: container-volume**

**emptyDir: {}**

**A computer screen shot of a black screen

Description automatically generated**

A computer screen shot of a black screen

Description automatically generated

1. Launch the web service by applying the configuration with the following command:  
   **kubectl apply -f emptydir.yaml**

A screenshot of a computer

Description automatically generated

1. Confirm that the pod is running and the volume is correctly shared by using the following command:  
   **kubectl describe pod container-share-volume**  
     
   **A screenshot of a computer program

   Description automatically generated**

**A computer screen shot of a program

Description automatically generated**

**Step 2: Interact with the shared volume from both containers**

1. Open a shell session in **container1** with the following command:  
   **kubectl exec -it container-share-volume -c container1 -- bash**  
     
   A screen shot of a computer program

   Description automatically generated
2. Within the shell session, use the following commands to navigate to the shared volume and create files:  
   **cd /tmp/xchange  
   touch container1-file{1..10}.txt**  
   A screen shot of a computer

   Description automatically generated
3. List the files to confirm their creation using the following command:  
   **ls**A screen shot of a computer

   Description automatically generated

|  |
| --- |
| **Note:** Exit the shell session with the **exit** command |

1. Open a shell session in **container2** using the following command:  
   **kubectl exec -it container-share-volume -c container2 -- bash**  
     
   A screenshot of a computer program

   Description automatically generated
2. Verify using the following commands that **container2** can see the files created by **container1**:  
   **cd /tmp/data  
   ls**  
   A screen shot of a computer

   Description automatically generated

|  |
| --- |
| **Note:** Remain in the shell session without exiting |

**Step 3: Test data persistence and sharing capability**

1. Use the following command to create additional files in container2:  
   **touch container2-file{1..10}.txt**A screen shot of a computer

   Description automatically generated
2. List the files to confirm their creation with the help of the following command:  
   **ls**  
   A screen shot of a computer

   Description automatically generated
3. Write to a file from **container2** with the help of the following command:  
   **echo “testing from container2” >> container1-file.txt**  
   A screen shot of a computer

   Description automatically generated
4. Read the content of the file to verify the write operation with the help of the following command:  
   **cat container1-file.txt**A screen shot of a computer

   Description automatically generated

|  |
| --- |
| **Note:** Exit the shell session with the **exit** command |

1. Use the following command to return to **container1**:  
   **kubectl exec -it container-share-volume -c container1 – bash**A screen shot of a computer

   Description automatically generated
2. Validate that **container1** can see the changes using the following commands:  
   **cd /tmp/xchange  
   ls**  
     
   A screen shot of a computer screen

   Description automatically generated
3. Use the following command to count the number of files:  
   **ls | wc -l**A computer screen shot of a program

   Description automatically generated
4. Read the contents of the file written by **container2** using the following command:  
   **cat container1-file.txt**  
     
   A screen shot of a computer

   Description automatically generated  
     
   By following these steps, you have successfully implemented data sharing between containers within the same pod using hostPath volumes in Kubernetes.