Lesson 3 : Select a Storage Class for an Application

Match applications with storage classes that provide storage services to satisfy application requirements.

Storage Class Selection

- Type of storage
- Enable dynamic storage provisioning (on demand)
- Cluster Admin determine StorageClass
- Default storage class

- Use cases:
  - Development
  - Testing
  - Production
- Storage tier
  - Cost
  - Performance
  - Reliability
  - other functions

Kubernetes and Application Responsibilities

- Application responsible for data integrity, confidentiality and consistency
  - concurrent accesss to shared volume
  - data replicated across region
- Example :
  - PVC of iSCSI LUN configured as RWO (single pod access) but developer forcibly mount to two pods of same host.
  - Problematic depends on applications
- Single-node access (RWO) or shared access (RWX)
  - do not ensure files can be shared safely and reliably

Use Cases for Storage Classes

Storage volume modes A storage class with block volume mode support can increase performance for applications that can use raw block devices. Consider using a storage class with Filesystem volume mode support for applications that share files or that provide file access.

Quality of Service (QoS) levels A Solid State Drive (SSD) provides excellent speed and support for frequently accessed files. Use a lower cost and a slower hard drive (HDD) for files that are accessed less often.

Administrative support tier A production-tier storage class can include volumes that are backed up often. In contrast, a development-tier storage class might include volumes that are not configured with a backup schedule.

Kubernetes matches PVCs with best available PV

- PV that is not bound to any PVC
- PV that at least as larges as the requested size in PVC
- PV that matches the other settingsspecified in PVC
  - Access Mode
  - Storage Class
- If PVC can’t satisfied all criterias, enter pending state

Create a Storage Class

```
apiVersion: storage.k8s.io/v1 (1)
kind: StorageClass (2)
metadata:
  name: io1-gold-storage (3)
  annotations: (4)
    storageclass.kubernetes.io/is-default-class: 'false'
    description:'Provides RWO and RWOP Filesystem & Block volumes'
    ...
parameters: (5)
  type: io1
  iopsPerGB: "10"
    ...
provisioner: kubernetes.io/aws-ebs (6)
reclaimPolicy: Delete (7)
volumeBindingMode: Immediate (8)
allowVolumeExpansion: true (9)
```

```
[user@host ~]$ oc create -f <storage-class-filename.yaml>
```

1. (required) The current API version.
2. (required) The API object type.
3. (required) The name of the storage class.
4. (optional) Annotations for the storage class.
5. (optional) The required parameters for the specific provisioner; this object differs between plug-ins.
6. (required) The type of provisioner that is associated with this storage class.
7. (optional) The selected reclaim policy for the storage class.
8. (optional) The selected volume binding mode for the storage class.

Verify Storage Classes creation

Use the `oc get storageclass` command to view the storage class options that are available in a cluster.

```
[user@host ~]$ oc get storageclass
```

A regular cluster user can view the attributes of a storage class by using the `describe` command. The following example queries the attributes of the storage class with the name `lvms-vg1`.

```
[user@host ~]$ oc describe storageclass lvms-vg1
IsDefaultClass:        No
Annotations:           description=Provides RWO and RWOP Filesystem & Block volumes
Provisioner:           topolvm.io
Parameters:            csi.storage.k8s.io/fstype=xfs,topolvm.io/device-class=vg1
AllowVolumeExpansion:  True
MountOptions:          <none>
ReclaimPolicy:         Delete
VolumeBindingMode:     WaitForFirstConsumer
Events:                <none>
```

Storage Class Usage

- The YAML manifest file

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: my-block-pvc
spec:
  accessModes:
  - RWO
  volumeMode: Block
  storageClassName: <storage-class-name>
  resources:
    requests:
      storage: 10Gi
```

$ oc create –f <manifest file>

$ oc set volume \
deployment/<deployment-name> \
--add –name <volume-name> \
--claim-name my-block-pvc \
--mount-path /var/tmp

Guided Exercise: Provision Persistent Data Volumes

You should be able to:

- Deploy a MySQL database with persistent storage from a PVC.
- Identify the PV that backs the application.
- Identify the storage provisioner that created the PV