


Review 3

Stateful applications



1

ConfigMaps

Session 3 - Review

- Externalized data stored within kubernetes.
- Can be referenced through several different means:
 - environment variable
 - a command line argument (via env var)
 - injected as a file into a volume mount
- Can be created from a manifest, literals, directories, or files directly.
- Imperative style:
 - `$ kubectl create configmap literal-example --from-literal="city=Brussels" --from-literal=state=Belgium`
 - `$ kubectl create configmap file-example --from-file=cm/city --from-file=cm/state`

```

apiVersion: v1
kind: ConfigMap
metadata:
  name: manifest-example
data:
  state: Belgium
  city: Brussels
  content: |
    Look at this,
    its multiline!

```

M.Romdhani, 2020

2

Session 3 - Review

Secrets

- **type:** There are three different types of secrets within Kubernetes:
 - **docker-registry** - credentials used to authenticate to a container registry
 - **generic/Opaque** - literal values from different sources
 - **tls** - a certificate based secret
- **data:** Contains key-value pairs of base64 encoded content.
- **Imperative style:**
 - `$ kubectl create secret generic literal-secret --from-literal=username=administrator --from-literal=password=password`
 - `kubectl create secret generic file-secret --from-file=secret/username --from-file=secret/password`

```
apiVersion: v1
kind: Secret
metadata:
  name: manifest-secret
type: Opaque
data:
  username: S3ViZXJuZXRlcw==
  password: cGFzc3dvcmQ=
```

M.Romdhani, 2020
3

3

Session 3 - Review

emptyDir & hostPath

- **EmptyDir**
 - An emptyDir volume is first created when a Pod is assigned to a Node, **and exists as long as that Pod is running on that node.**
 - When a Pod is restarted or removed, the data in the emptyDir is lost forever.
- **HostPath**
 - A hostPath volume mounts a file or directory from the **node's filesystem into the Pod.** You can specify whether the file/directory must already exist on the node or should be created on pod startup.

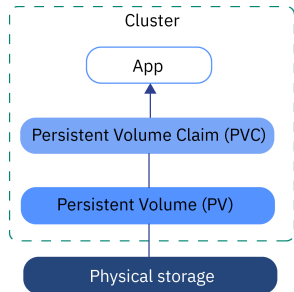
M.Romdhani, 2020
4

4

Session 3 - Review

PVs vs. PVCs

- **Persistent Volume (PV) –**
 - It is a piece of network storage that has been provisioned by the administrator. It's a resource in the cluster which is independent of any individual pod that uses the PV.
- **Persistent Volume Claim (PVC)**
 - It is a request for storage by a user that can be fulfilled by a PV.
- **PVs and PVCs are independent from Pod lifecycles and preserve data through restarting, rescheduling, and even deleting Pods.**



The diagram illustrates the storage stack within a Kubernetes cluster. A dashed green box labeled 'Cluster' contains three components: 'App' (a white rounded rectangle), 'Persistent Volume Claim (PVC)' (a blue rounded rectangle), and 'Persistent Volume (PV)' (a blue rounded rectangle). Arrows indicate the flow of data: from 'App' to 'PVC', and from 'PVC' to 'PV'. Below the cluster box, outside the cluster boundary, is a dark blue rounded rectangle labeled 'Physical storage'. An arrow points from 'PV' down to 'Physical storage'.

5

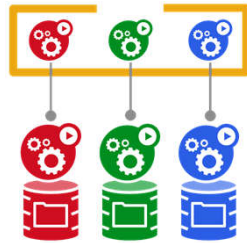
M.Romdhani, 2020

5

Session 3 - Review

Statefulsets

- **Tailored to managing Pods that must persist or maintain state.**
 - Replicated Relational DBs
 - No SQL BDs
- **StatefulSet Deployments provide:**
 - **Stable, unique network identifiers**
 - **Stable, persistent storage**
 - **Ordered, graceful deployment and scaling**
 - **Ordered, automated rolling updates**



The diagram shows three StatefulSet pods, each represented by a colored gear icon (red, green, and blue) with a unique identifier (1, 2, and 3) inside. Below each gear is a corresponding colored database icon (red, green, and blue). The pods are connected to their respective databases by lines, indicating that each pod has its own persistent storage.

6

M.Romdhani, 2020

6