Persistent Volumes

» Persistent Volume (AWS)

Storage Class

First, create a new StorageClass resource.

```
cat > aws-ebs.yaml <<EOF
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: general
provisioner: kubernetes.io/aws-ebs
EOF
```

Create the new storage class.

```
kubectl create -f aws-ebs.yaml
```

Persistent Volume Claim

Define a Persistent Volume Claim resource.

```
cat > volume-persistent-claim.yaml <<EOF
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: volume-persistent-claim
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 5Gi
  storageClassName: general
EOF
```

Create the Persistent Volume Claim.

```
kubectl create -f volume-persistent-claim.yaml
```

Using Persistent Volumes

Create a new deployment that uses the persistent volume.

```
cat > volume-persistent-deployment1.yaml <<EOF
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: volume-persistent-deployment1
spec:
  template:
    metadata:
      labels:
        app: volume-persistent1
    spec:
      containers:
      - name: volume-persistent1
        image: busybox
        stdin: true
        tty: true
        volumeMounts:
        - mountPath: /persistent
          name: persistent-data
      volumes:
      - name: persistent-data
        persistentVolumeClaim:
          claimName: volume-persistent-claim
EOF
```

Create the deployment and view it's detail.

```
kubectl create -f volume-persistent-deployment1.yaml
kubectl get pods -l app=volume-persistent1
POD=$(kubectl get pods -l app=volume-persistent1 --output=jsonpath={.items..metadata.name})
kubectl attach -it $POD
ls -la /
ls -la /persistent
echo "Hello, World!" > /persistent/hello.txt
exit
```

Now delete the first deployment.

```
kubectl delete deployment volume-persistent-deployment1
```

Create a new deployment that will use the same persistent volume.

```
cat > volume-persistent-deployment2.yaml <<EOF
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: volume-persistent-deployment2
spec:
  template:
    metadata:
      labels:
        app: volume-persistent2
    spec:
      containers:
      - name: volume-persistent2
        image: busybox
        stdin: true
        tty: true
        volumeMounts:
        - mountPath: /persistent
          name: persistent-data
      volumes:
      - name: persistent-data
        persistentVolumeClaim:
          claimName: volume-persistent-claim
EOF
```

Create the deployment and view it’s detail. Verify the file still exists.

```
kubectl create -f volume-persistent-deployment2.yaml
kubectl get pods -l app=volume-persistent2
POD=$(kubectl get pods -l app=volume-persistent2 --output=jsonpath={.items..metadata.name})
kubectl attach -it $POD
ls -la /
ls -la /persistent
cat /persistent/hello.txt
exit
```

Clean Up

Delete the resources when you are finished.

```
kubectl delete deployment volume-persistent-deployment2
kubectl delete -f volume-persistent-claim.yaml
```

View any remaining resources.

```
kubectl get all
```