**StorageOS Volume Guide**

As a simple first use of StorageOS with Kubernetes following the example below will create a PersistentVolumeClaim (PVC) and schedule a Pod to mount the PersistentVolume (PV) provisioned by the PVC.

**Creating the PersistentVolumeClaim**

1. You can find the latest files in the StorageOS example deployment repository
2. git clone https://github.com/storageos/deploy.git storageos

PVC definition

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: my-vol-1

annotations:

volume.beta.kubernetes.io/storage-class: fast

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 5Gi

The above PVC will dynamically provision a 5GB volume using the fast StorageClass. This StorageClass was created during the StorageOS install and causes StorageOS to provision a PersistentVolume.

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: my-vol-1

labels:

storageos.com/replicas: "1"

annotations:

volume.beta.kubernetes.io/storage-class: fast

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 5Gi

The above PVC has the storageos.com/replicas label set. This label tells StorageOS to create a replica for the volume that is created. For the sake of keeping this example simple the unreplicated volume will be used.

1. Move into the examples folder and create a PVC using the PVC definition above.
2. $ cd storageos
3. $ kubectl create -f ./k8s/examples/pvc.yaml

You can view the PVC that you have created with the command below

$ kubectl get pvc

NAME STATUS VOLUME CAPACITY ACCESS MODES STORAGECLASS AGE

my-vol-1 Bound pvc-f8ffa027-e821-11e8-bc0b-0ac77ccc61fa 5Gi RWO fast 1m

1. Create a pod that mounts the PVC created in step 2.
2. $ kubectl create -f ./k8s/examples/debian-pvc.yaml

The command above creates a Pod that uses the PVC that was created in step 1.

apiVersion: v1

kind: Pod

metadata:

name: d1

spec:

containers:

- name: debian

image: debian:9-slim

command: ["/bin/sleep"]

args: [ "3600" ]

volumeMounts:

- mountPath: /mnt

name: v1

volumes:

- name: v1

persistentVolumeClaim:

claimName: my-vol-1

In the Pod definition above the volume v1, which references the PVC created in step 2, is mounted in the pod at /mnt. In this example a debian image is used for the container but any container image with a shell would work for this example.

1. Confirm that the pod is up and running
2. $ kubectl get pods
3. NAME READY STATUS RESTARTS AGE
4. d1 1/1 Running 0 1m
5. Execute a shell inside the container and write some contents to a file
6. $ kubectl exec -it d1 -- bash
7. root@d1:/# echo "Hello World!" > /mnt/helloworld
8. root@d1:/# cat /mnt/helloworld
9. Hello World!

By writing to /mnt inside the container, the StorageOS volume created by the PVC is being written to. If you were to kill the pod and start it again on a new node, the helloworld file would still be avaliable.

If you wish to see more use cases with actual applications please see our [Use Cases](https://docs.storageos.com/docs/usecases/kubernetes/index) documentation.