Operators

» etcd Operator

Create the Custom Resource Definition (CRD) for the Etcd Operator:

```
cat > etcd-operator-crd.yaml<<EOF
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  name: etcdclusters.etcd.database.coreos.com
spec:
  group: etcd.database.coreos.com
  names:
    kind: EtcdCluster
    listKind: EtcdClusterList
    plural: etcdclusters
    shortNames:
    - etcdclus
    - etcd
    singular: etcdcluster
  scope: Namespaced
  version: v1beta2
  versions:
  - name: v1beta2
    served: true
    storage: true
EOF
```

```
oc create -f etcd-operator-crd.yaml
```

Verify the CRD was successfully created:

```
oc get crd
```

```
oc get etcdcluster
```

Create a dedicated Service Account responsible for running the Etcd Operator:

```
cat > etcd-operator-sa.yaml<<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  name: etcd-operator-sa
EOF
```

```
oc create -f etcd-operator-sa.yaml
```

Verify the Service Account was successfully created:

```
oc get sa
```

Create the Role that the `etcd-operator-sa` Service Account will need for authorization to perform actions against the Kubernetes API:

```
cat > etcd-operator-role.yaml<<EOF
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: etcd-operator-role
rules:
- apiGroups:
  - etcd.database.coreos.com
  resources:
  - etcdclusters
  - etcdbackups
  - etcdrestores
  verbs:
  - '*'
- apiGroups:
  - ""
  resources:
  - pods
  - services
  - endpoints
  - persistentvolumeclaims
  - events
  verbs:
  - '*'
- apiGroups:
  - apps
  resources:
  - deployments
  verbs:
  - '*'
  -  *
- apiGroups:
  - ""
  resources:
  - secrets
  verbs:
  - get
EOF
```

```
oc create -f etcd-operator-role.yaml
```

Verify the Role was successfully created:

```
oc get roles
```

Create the RoleBinding to bind the `etcd-operator-role` Role to the `etcd-operator-sa` Service Account:

```
cat > etcd-operator-rolebinding.yaml<<EOF
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: etcd-operator-rolebinding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: etcd-operator-role
subjects:
- kind: ServiceAccount
  name: etcd-operator-sa
  namespace: myproject
EOF
```

```
oc create -f etcd-operator-rolebinding.yaml
```

Verify the RoleBinding was successfully created:

```
oc get rolebindings
```

Create the Deployment containing the Etcd Operator container image:

```
cat > etcd-operator-deployment.yaml<<EOF
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  labels:
    name: etcdoperator
  name: etcd-operator
spec:
  replicas: 1
  selector:
    matchLabels:
      name: etcd-operator
  template:
    metadata:
      labels:
        name: etcd-operator
    spec:
      containers:
      - command:
        - etcd-operator
        - --create-crd=false
        env:
        - name: MY_POD_NAMESPACE
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.namespace
        - name: MY_POD_NAME
          valueFrom:
            fieldRef:
              apiVersion: v1
              fieldPath: metadata.name
        image: quay.io/coreos/etcd-operator@sha256:c0301e4686c3ed4206e370b42de5a3bd2229b9fb4906cf85f3f30650424abec2
        imagePullPolicy: IfNotPresent
        name: etcd-operator
      serviceAccountName: etcd-operator-sa
```

```
EOF
```

```
oc create -f etcd-operator-deployment.yaml
```

Verify the Etcd Operator Deployment was successfully created:

```
oc get deploy
```

Verify the Etcd Operator Deployment pods are running:

```
oc get deploy
```

Open a new terminal window to follow Etcd Operator logs in real-time:

```
export ETCD_OPERATOR_POD=$(oc get pods -l name=etcd-operator -o jsonpath='{.items[0].metadata.name}' -o jsonpath='{.items[0].metadata.
name}')
oc logs $ETCD_OPERATOR_POD -f
```

Observe the leader-election lease on the Etcd Operator Endpoint:

```
oc get endpoints etcd-operator -o yaml
```

Create the `EtcdCluster` Custom Resource (CR):

```
cat > etcd-operator-cr.yaml<<EOF
apiVersion: etcd.database.coreos.com/v1beta2
kind: EtcdCluster
metadata:
  name: example-etcd-cluster
spec:
  size: 3
  version: 3.2.13
EOF
```

```
oc create -f etcd-operator-cr.yaml
```

Verify the cluster object was created:

```
oc get etcdclusters
```

Watch the pods in the Etcd cluster get created:

```
oc get pods -l etcd_cluster=example-etcd-cluster -w
```

Verify the cluster has been exposed via a ClusterIP service:

```
oc get services -l etcd_cluster=example-etcd-cluster
```

Let’s now create another pod and attempt to connect to the etcd cluster via `etcdctl` :

```
oc run etcdclient --image=busybox busybox --restart=Never -- /usr/bin/tail -f /dev/null
```

Access the pod:

```
kubectl exec -it etcdclient /bin/sh
```

Install the Etcd Client:

```
wget https://github.com/coreos/etcd/releases/download/v3.1.4/etcd-v3.1.4-linux-amd64.tar.gz
tar -xvf etcd-v3.1.4-linux-amd64.tar.gz
cp etcd-v3.1.4-linux-amd64/etcdctl .
```

Set the etcd version and endpoint variables:

```
export ETCDCTL_API=3
export ETCDCTL_ENDPOINTS=example-etcd-cluster-client:2379
```

Attempt to write a key/value into the Etcd cluster:

```
./etcdctl put operator sdk
./etcdctl get operator
```

Exit out of the client pod:

```
exit
```

Let’s change the size of the Etcd `example-etcd-cluster` CR. The Etcd Operator pod will detect the CR `spec.size` change and modify the number of pods in the cluster:

```
oc patch etcdcluster example-etcd-cluster --type='json' -p '[{"op": "replace", "path": "/spec/size", "value":5}]'
```

Let’s change the version of our `example-etcd-cluster` CR. The etcd-operator pod will detect the CR `spec.version` change and create a new cluster with the newly specified image:

```
oc patch etcdcluster example-etcd-cluster --type='json' -p '[{"op": "replace", "path": "/spec/version", "value":3.2.13}]'
```

In another session delete a pod from the cluster and watch recovery in real-time.

```
export EXAMPLE_ETCD_CLUSTER_POD=$(oc get pods -l app=etcd -o jsonpath='{.items[0].metadata.name}' -o jsonpath='{.items[0].metadata.nam
e}')
oc delete pod $EXAMPLE_ETCD_CLUSTER_POD
```

Delete your Etcd cluster:

```
kubectl delete etcdcluster example-etcd-cluster
```

Delete the Etcd Operator:

```
kubectl delete deployment etcd-operator
```

Delete the Etcd CRD:

```
oc delete crd etcdclusters.etcd.database.coreos.com
```