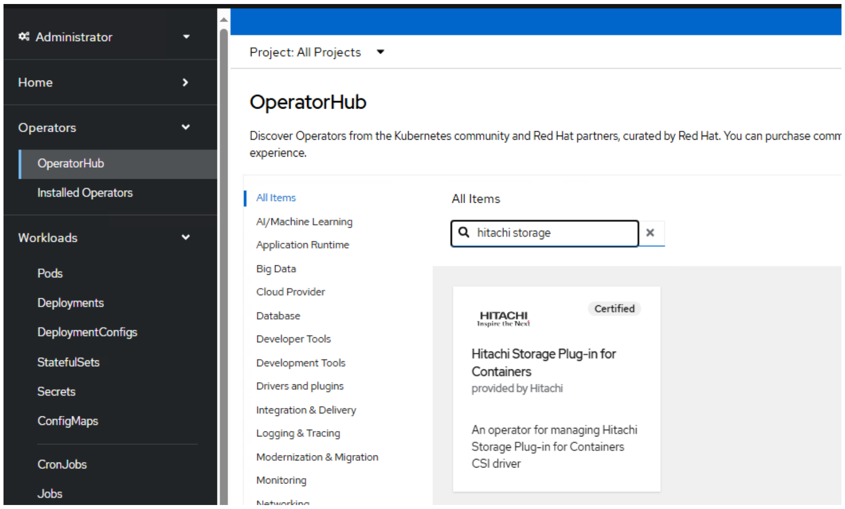
Openshift HITACHI Storage Operatör Kurulumu ve Konfigurasyonu

Öncelikle Openshift Cluster üzerinden "Hitachi Storage" diyerek operatör kurulumu yapıyoruz.



Project: All Projects

OperatorHub

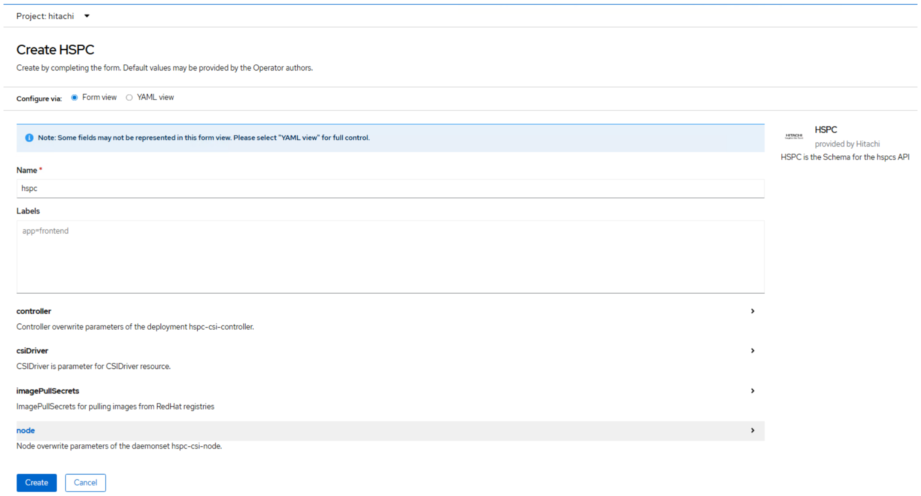
Discover Operators from the Kubernetes community and Red Hat partners, curated by Red Hat. You can purchase comm
experience.

hitachi storage

Hitachi Storage Plug-in for Containers
provided by Hitachi

An operator for managing Hitachi Storage Plug-in for Containers CSI driver

The next step is to create the HSPC Instance, this can be done using the Operator Details. Select the Hitachi Storage Plug-in for Containers, and then click Create Instance on the Operator Details. Click Create.



Project: hitachi

Create HSPC

Create by completing the form. Default values may be provided by the Operator authors.

Configure via: ● Form view ○ YAML view

Note: Some fields may not be represented in this form view. Please select "YAML view" for full control.

HSPC
provided by Hitachi
HSPC is the Schema for the hspcs API

Name *
hspc

Labels
app=frontend

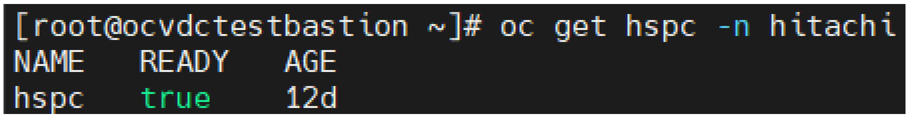
controller
Controller overwrite parameters of the deployment hspc-csi-controller.

csiDriver
CSIDriver is parameter for CSIDriver resource.

imagePullSecrets
ImagePullSecrets for pulling images from RedHat registries

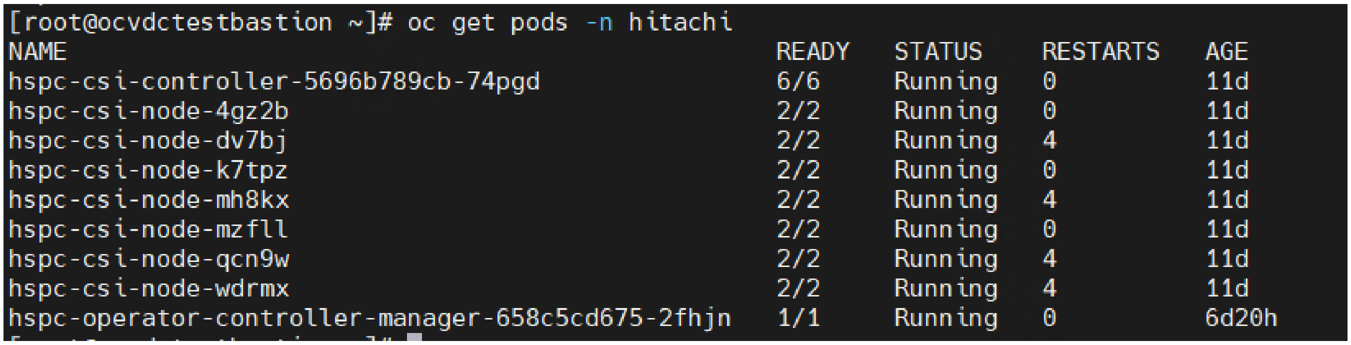
node
Node overwrite parameters of the daemonset hspc-csi-node.

Sonrasında hspc in başarılı bir şekilde ayakta olduğuna bakıyoruz.



```
[root@ocvdctestbastion ~]# oc get hspc -n hitachi
NAME    READY   AGE
hspc    true    12d
```

Pod larında ayakta olduğu aşağıdaki gibi görüyoruz.



```
[root@ocvdctestbastion ~]# oc get pods -n hitachi
NAME                                              READY   STATUS    RESTARTS   AGE
hspc-csi-controller-5696b789cb-74pgd              6/6     Running   0          11d
hspc-csi-node-4gz2b                               2/2     Running   0          11d
hspc-csi-node-dv7bj                               2/2     Running   4          11d
hspc-csi-node-k7tpz                               2/2     Running   0          11d
hspc-csi-node-mh8kx                               2/2     Running   4          11d
hspc-csi-node-mzfll                               2/2     Running   0          11d
hspc-csi-node-qcn9w                               2/2     Running   4          11d
hspc-csi-node-wdrmx                               2/2     Running   4          11d
hspc-operator-controller-manager-658c5cd675-2fhjn   1/1     Running   0          6d20h
```

Sonrasında aşağıda storage kullanıcı adı, şifre ve Storage API Url in bulunduğu bir secret yapıyoruz.

```
apiVersion: v1
kind: Secret
metadata:
 name: secret-vsp-01
 namespace: hitachi
type: Opaque
data:
 url: aHR0cHM6Ly8xOTIuMTY4LjE5OC4xMi91aS8jL3ZvbHVtZXM7dD0xNzU2NzI1NTYxOTg0
 user: b3BlbnNoaWZ0
 password: T28yMDI1
```

Sonrasındaki aşağıdaki gibi bir StorageClass oluşturulur.

```
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: sc-vsp-01-virtualization
  annotations:
    description: Provides RWO and RWX Block volumes suitable for Virtual Machine disks
provisioner: hspc.csi.hitachi.com
parameters:
  serialNumber: '810732'
  poolID: '0'
  portID: 'CL1-D,CL2-D,CL3-A,CL4-A'
  connectionType: nvme-fc
  nvmSubsystemID: "1"
  storageEfficiency: "CompressionDeduplication"
  storageEfficiencyMode: "Inline"
  mapOptions: 'krbd:rxbounce'
  mounter: rbd
  imageFormat: '2'
  imageFeatures: 'layering,deep-flatten,exclusive-lock,object-map,fast-diff'
  csi.storage.k8s.io/fstype: ext4
  csi.storage.k8s.io/node-publish-secret-name: "secret-vsp-01"
  csi.storage.k8s.io/node-publish-secret-namespace: "hitachi"
  csi.storage.k8s.io/provisioner-secret-name: "secret-vsp-01"
  csi.storage.k8s.io/provisioner-secret-namespace: "hitachi"
  csi.storage.k8s.io/controller-publish-secret-name: "secret-vsp-01"
  csi.storage.k8s.io/controller-publish-secret-namespace: "hitachi"
  csi.storage.k8s.io/node-stage-secret-name: "secret-vsp-01"
  csi.storage.k8s.io/node-stage-secret-namespace: "hitachi"
  csi.storage.k8s.io/controller-expand-secret-name: "secret-vsp-01"
  csi.storage.k8s.io/controller-expand-secret-namespace: "hitachi"
reclaimPolicy: Delete
allowVolumeExpansion: true
volumeBindingMode: Immediate
```

Sonrasında üzerinde hitachi csi ların bulunduğu node lar için bir label ekliyoruz. Bu label üzerinden bu node ları yöneteceğiz. Yapımızda 4 adet fiziksel node var bunlar için aşağıdaki gibi label tanımı yapıyoruz.

```
 oc label node ocvdctestworker1.ocvdctest.itservices.local hitachi-map=true
 oc label node ocvdctestworker4.ocvdctest.itservices.local hitachi-map=true
 oc label node ocvdctestworker5.ocvdctest.itservices.local hitachi-map=true
 oc label node ocvdctestworker6.ocvdctest.itservices.local hitachi-map=true
```

Sonrasında aşağıda görüldüğü üzere sadece bu node lara MachineConfig yapmak için bir MachineConfigPool oluşturuyoruz.

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfigPool
metadata:
  name: example
spec:
  machineConfigSelector:
    matchExpressions:
      - key: machineconfiguration.openshift.io/role
        operator: In
        values:
          - worker
          - hitachi-workers
  maxUnavailable: 1
  nodeSelector:
    matchLabels:
      hitachi-map: 'true'
```

Sonrasında bu node lara multipath konfigurasyonu yapılması gerekmektedir, o nedenle ilk olarak bu node larda geçerli olacak şekilde bir Machine Config oluşturuyoruz.

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
 name: hitachi-workers-enable-multipath-conf
 labels:
 machineconfiguration.openshift.io/role: hitachi-workers
spec:
 config:
 ignition:
 version: 3.2.0
 storage:
 files:
 - path: /etc/multipath.conf
 mode: 400
 filesystem: root
 contents:
 source: data:text/plain;charset=utf-8;base64,
ZGVmYXVsdHMgewogICAgICAgIHVzZXJfZnJpZW5kbHlfbmFtZXMgeWVzCiAgICAgICAgZmluZF9tdWx0a
XBhdGhzIHllcwp9CgpibGFja2xpc3Qgewp9Cg==
 verification: {}
 systemd:
 units:
 - name: multipathd.service
 enabled: true
 state: started
osImageURL: ""
```

Son olarak node lara bağlanıp, her node da aşağıda görüldüğü üzere hostnqn id sini resetliyoruz.

```
oc debug node/<node-name>
```

```
chroot /host

nvme gen-hostnqn > /etc/nvme/hostnqn
```

Sonrasında Snapshot alınabilmesi için aşağıdaki VolumeSnapshotClass ı yaratıyoruz.

```
apiVersion: snapshot.storage.k8s.io/v1
kind: VolumeSnapshotClass
metadata:
 name: snapshotclass-hitachi
driver: hspc.csi.hitachi.com
deletionPolicy: Delete
parameters:
 poolID: "0"
 csi.storage.k8s.io/snapshotter-secret-name: "secret-vsp-01"
 csi.storage.k8s.io/snapshotter-secret-namespace: "hitachi"
```