

□□nfs-storageClass

- storageClass

--	--	--	--	--

k8scephNFS

- node nfs

```

namespace

```

- ☐PV/☐PVC☐☐☐PV☐☐☐NFS,☐☐PVC☐☐PVC☐☐PV

POD PV PVC rancher

storageClass PV/PVC

- 

- **00-nfs-storage-rabc.yaml**

```
apiVersion: v1
kind: Namespace
metadata:
  # replace with namespace where provisioner is deployed
  # [ ] namespace
  name: nfs-storage
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: nfs-client-provisioner
  # replace with namespace where provisioner is deployed
```

```

# []namespace
namespace: nfs-storage
---
kind: ClusterRole
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: nfs-client-provisioner-runner
rules:
  - apiGroups: [""]
    resources: ["nodes"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["persistentvolumes"]
    verbs: ["get", "list", "watch", "create", "delete"]
  - apiGroups: [""]
    resources: ["persistentvolumeclaims"]
    verbs: ["get", "list", "watch", "update"]
  - apiGroups: ["storage.k8s.io"]
    resources: ["storageclasses"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["events"]
    verbs: ["create", "update", "patch"]
---
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: run-nfs-client-provisioner
subjects:
  - kind: ServiceAccount
    name: nfs-client-provisioner
    # replace with namespace where provisioner is deployed
    # []namespace
    namespace: nfs-storage
roleRef:
  kind: ClusterRole
  name: nfs-client-provisioner-runner
  apiGroup: rbac.authorization.k8s.io
---
kind: Role

```

```

apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: leader-locking-nfs-client-provisioner
  # replace with namespace where provisioner is deployed
  # [ ] namespace
  namespace: nfs-storage
rules:
  - apiGroups: [""]
    resources: ["endpoints"]
    verbs: ["get", "list", "watch", "create", "update", "patch"]
  ---
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: leader-locking-nfs-client-provisioner
  # replace with namespace where provisioner is deployed
  # [ ] namespace
  namespace: nfs-storage
subjects:
  - kind: ServiceAccount
    name: nfs-client-provisioner
    # replace with namespace where provisioner is deployed
    # [ ] namespace
    namespace: nfs-storage
roleRef:
  kind: Role
  name: leader-locking-nfs-client-provisioner
  apiGroup: rbac.authorization.k8s.io

```

• 01-nfs-storage-provisioner.yaml

[] API [] PV/PVC []

[]

```

# env:
# - PROVISIONER_NAME [ ] storageClass [ ]
# - NFS_SERVER [ ] nfs [ ]
# - NFS_PATH [ ] nfs [ ]

```

```
kind: Deployment
apiVersion: apps/v1
metadata:
  name: nfs-client-provisioner
spec:
  replicas: 1
  selector:
    matchLabels:
      app: nfs-client-provisioner
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: nfs-client-provisioner
    spec:
      serviceAccountName: nfs-client-provisioner
      containers:
        - name: nfs-client-provisioner
          # image: registry.k8s.io/sig-storage/nfs-subdir-external-provisioner:v4.0.2
          image: k8s.dockerproxy.com/sig-storage/nfs-subdir-external-provisioner:v4.0.2
          volumeMounts:
            - name: nfs-client-root
              mountPath: /persistentvolumes
          env:
            - name: PROVISIONER_NAME
              value: k8s-sigs.io/nfs-subdir-external-provisioner
            - name: NFS_SERVER
              # value: <YOUR NFS SERVER HOSTNAME>
              value: nfs-share.vpclub.io
            - name: NFS_PATH
              # value: /var/nfs
              value: /data/share
      volumes:
        - name: nfs-client-root
          nfs:
            # server: <YOUR NFS SERVER HOSTNAME>
            server: nfs-share.vpclub.io
            path: /data/share
```

```
# PV[ ]hfs[ ]
# [ ]hosts
# echo "192.168.5.10      nfs-share.vpclub.io" >> /etc/hosts
```

• 02-nfs-storage-class.yaml

provisioner[]PROVISIONER_NAME[]

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: nfs-client-root-ns-pvname
  # or choose another name, must match deployment's env PROVISIONER_NAME'
  # [ ] provisioner pod [ ] PROVISIONER_NAME
provisioner: k8s-sigs.io/nfs-subdir-external-provisioner
parameters:
  # [ ]hfs[ ]
  # "${.PVC.namespace}/${.PVC.name}"
  # "${.PVC.namespace}/${.PVC.annotations.nfs.io/storage-path}"
  pathPattern: "${.PVC.namespace}/${.PVC.name}"
  # PV[ ]delete [ ]retain[ ]
  onDelete: retain
# [ ] Retain - [ ]Recycle - [ ]Delete - [ ] PVC [ ] volume [ ] Delete
reclaimPolicy: Retain

---

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: nfs-client-root-ns-customer
  # or choose another name, must match deployment's env PROVISIONER_NAME'
  # [ ] provisioner pod [ ] PROVISIONER_NAME
provisioner: "k8s-sigs.io/nfs-subdir-external-provisioner"
parameters:
  # [ ]hfs[ ]
  # "${.PVC.namespace}/${.PVC.name}"
```

```

# "${.PVC.namespace}/${.PVC.annotations.nfs.io/storage-path}"
pathPattern: "${.PVC.namespace}/${.PVC.annotations.nfs.io/storage-path}"
# [PVC]delete [ ]retain[ ]
onDelete: retain
# [PVC]false[ ] onDelete [ ] onDelete [ ]
archiveOnDelete: "false"
# [ ] Retain - [ ]Recycle - [ ]Delete - [ ] PVC [ ] volume [ ] Delete
reclaimPolicy: Retain

---

apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: nfs-client-root
  # or choose another name, must match deployment's env PROVISIONER_NAME'
  # [ ] provisioner pod [ ] PROVISIONER_NAME
provisioner: k8s-sigs.io/nfs-subdir-external-provisioner
parameters:
  # [ ]nfs[ ]
  # "${.PVC.namespace}/${.PVC.name}"
  # "${.PVC.namespace}/${.PVC.annotations.nfs.io/storage-path}"
  pathPattern: "${.PVC.annotations.nfs.io/storage-path}"
  # [PVC]delete [ ]retain[ ]
  onDelete: retain
  # [ ] Retain - [ ]Recycle - [ ]Delete - [ ] PVC [ ] volume [ ] Delete
  reclaimPolicy: Retain

```

- []

```

kubectl apply -f 00-nfs-storage-rabc.yaml
kubectl apply -f 01-nfs-storage-provisioner.yaml
kubectl apply -f 02-nfs-storage-class.yaml

```

- **III PVC**

- 
- 

```
pathPattern: "${.PVC.namespace}/${.PVC.name}"
```

				PVC					
--	--	--	--	-----	--	--	--	--	--

nfs-client-root-ns-pvname

pathPattern: "\${PVC.namespace}/\${PVC.annotations.nfs.io/storage-path}"

```
(namespacefs.io/storage-path , rancher PVC PVC YML
```

```

nfs-client-root-ns

```

[illegible]

```
pathPattern: "${PVC.annotations.nfs.io/storage-path}" nfs-client-root
```

• 03-nfs-storage-customer.yaml

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: pvc-park-nsdsj
  # namespace: nfs-storage
  annotations:
    nfs.io/storage-path: "park-tianwei" # not required, depending on whether this annotation
was shown in the storage class description
spec:
  storageClassName: nfs-client-root
  # storageClassName: nfs-client-root-ns-customer
  # storageClassName: nfs-client-root-ns-pvname
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 10Gi

# 创建
# 创建storage-class
# rancher 创建PVC
# kubectl apply -f 03-nfs-storage-customer.yaml
```

- 创建PVC
- pv nfs nfs
- PVC PVC centos PVC

#26

12 2023 16:39:06

21 2024 18:01:59