Storages

Система хранения работает через драйверы (CSI плагины) или локально на нодах. Второй вариант неудобный. Далее первый вариант. Разработчик обычно предоставляет плагины в виде Helm чартов или yaml установщиков. Они устанавливаются в виде набора подов в namespace kube-system. Список плагинов Для тестов можно использовать встроенный драйвер, OpenEbs. К вопросу выбора драйвера, архитектуры хранилища и безопасности необходимо подходить очень серьезно.

Процесс запроса ресурсов: Pod Volume - PVC - SC - CSI Plugin

Storage Classes

ресурсы в storage.k8s.io/v1 группе. Неизменяемый. Для обновления нужно удалить и создать. SC access mode:

- ReadWriteOnce - один PVC может подключиться в режиме чтения/записи
- ReadWriteMany - несколько PVC. Файловые и объектные хранилища обычно поддерживают, блоковые нет.
- ReadOnlyMany - несколько PVC в режиме чтения.

Все PV должны подключиться в одинаковом режиме.

Reclaim policy (политика восстановления)

Политики восстановления сообщают Kubernetes, что делать с PV и связанным с ним внешним хранилищем, когда его PVC будет запущен.

- Delete. Удаление PVC приведет к удалению PV и внешнего хранилища.
- Retain. Безопаснее, но нужно самостоятельно удалять ресурсы.

Volume binding mode

Момент создания бакета. Immediate - сразу же, WaitForFirstConsumer - при подключении первого.

Примеры

Локальное хранилище (устаревшее)

На воркере

```
sudo mkdir -p /mnt/disks/ssd1
sudo chmod 777 /mnt/disks/ssd1  # Для упрощения примера
```

Настройка PV

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: example-local-pv
  labels:
    type: local
spec:
  capacity:
    storage: 10Gi
  volumeMode: Filesystem
  accessModes:
    - ReadWriteOnce
  persistentVolumeReclaimPolicy: Retain
  storageClassName: local-storage
  local:
    path: /mnt/disks/ssd1
  nodeAffinity:
    required:
      nodeSelectorTerms:
        - matchExpressions:
            - key: kubernetes.io/hostname
              operator: In
              values:
                - <node-name>  # Замените на имя узла, где находится директория
```

Настройка PVC

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: example-local-pvc
spec:
  accessModes:
    - ReadWriteOnce
  storageClassName: local-storage
```

```
  resources:
    requests:
      storage: 10Gi
```

Проверка связи

```
kubectl get pv
kubectl get pvc
```

Использование в Pod

```
apiVersion: v1
kind: Pod
metadata:
  name: example-pod
spec:
  containers:
    - name: example-container
      image: nginx
      volumeMounts:
        - mountPath: "/usr/share/nginx/html"
          name: local-storage
  volumes:
    - name: local-storage
      persistentVolumeClaim:
        claimName: example-local-pvc
```

Удаление ресурсов

```
kubectl delete pod example-pod
kubectl delete pvc example-local-pvc
kubectl delete pv example-local-pv
```

Хранилище Yandex.cloud

S3 aws - совместимое хранилище.

1. Настройка доступа через консоль

Установить консоль yc

```
curl -sSL https://storage.yandexcloud.net/yandexcloud-yc/install.sh | bash
source ~/.bashrc
```

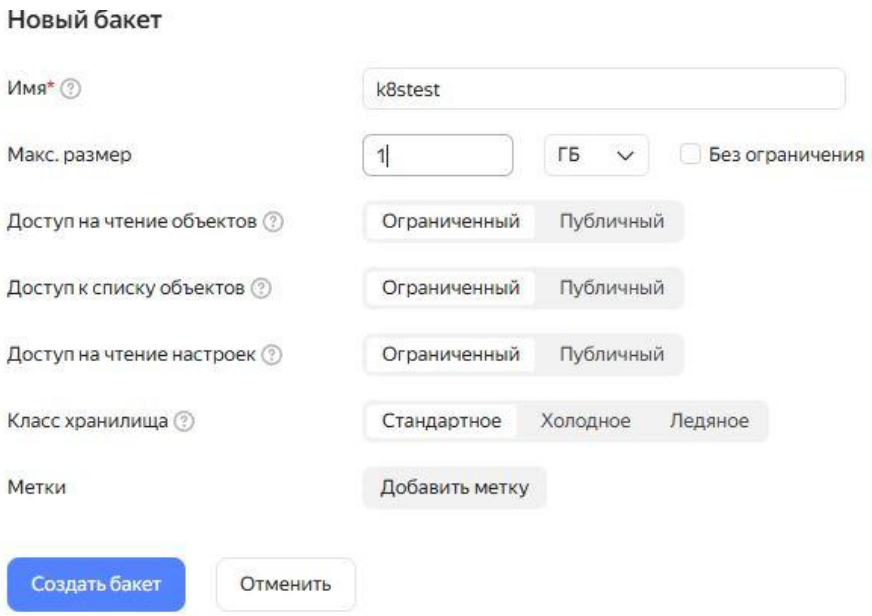
Запросить новый OAuth токен. Время жизни токена 1 год.

Инициализировать консоль

```
yc init
```

2. Настройка доступа к бакету

Через web-консоль создать бакет.



Выяснить folder_id созданного аккаунта

```
yc storage bucket get k8stest

name: k8stest
folder_id: b...q
anonymous_access_flags:
  read: false
  list: false
  config_read: false
default_storage_class: STANDARD
versioning: VERSIONING_DISABLED
max_size: "1073741824"
created_at: "2025-03-23T06:35:58.715069Z"
```

Создать сервисный аккаунт для доступа к бакету, в выводе будет id аккаунта

```
yc iam service-account create --name k8stest --output key.json
```

Файл key.json понадобится далее.

Добавить роль для созданного бакета сервис аккаунту

```
yc resource-manager folder add-access-binding <идентификатор_каталога> \
  --role <роль> \
  --subject serviceAccount:<идентификатор_сервисного_аккаунта>
```

3. Установка CSI плагина для yandex.cloud

```
git clone https://github.com/deckhouse/yandex-csi-driver.git
cd yandex-csi-driver
```

В самом git сказано, что запускать нужно из папки deploy/1.17 установив 2 переменные. У меня это не заработало.

В папке yandex-csi-driver/charts/yandex-csi-controller установить serviceAccountJSON и folderID

В файле csidriver.yaml заменить apiVersion: storage.k8s.io/v1beta1 на apiVersion: storage.k8s.io/v1

Дополнительно:

```
# Список сервис аккаунтов
yc iam service-account list
# Детализация информации по аккаунту
yc iam service-account get <идентификатор_аккаунта>
# Удаление аккаунта
yc iam service-account delete <идентификатор_аккаунта>
```

И нечего с yandex не получилось.

Настройка MinIO

Настройка внешнего MinIO

Настройка авторизации

Создать Secret. Ключ и Секрет не в Base64, а как указано при создании бакета.

```
apiVersion: v1
kind: Secret
metadata:
  namespace: kube-system
  name: csi-s3-secret
stringData:
  accessKeyID: XP5...Ih
  secretAccessKey: klz...zo
  # For AWS set it to "https://s3.<region>.amazonaws.com"
  endpoint: http://192.168.1.194:9000
  # If not on S3, set it to ""
  region: ""
```

Применить Secret

```
kubectl apply -f 1-minio-credentials.yaml
kubectl get secret
```

Установка csi драйвера

Проверить установленные csi драйверы, поставить драйвер

```
git clone https://github.com/ctrox/csi-s3.git
cd csi-s3/deploy/kubernetes
kubectl apply -f .
```

Вот только какого-то хера в данной директории не было yaml для создания драйвера)
Добавляем драйвер

```
apiVersion: storage.k8s.io/v1
kind: CSIDriver
metadata:
  name: ch.ctrox.csi.s3-driver
spec:
  attachRequired: false
  podInfoOnMount: true
```

Обязательно в списке драйверов должен появиться драйвер

```
kubectl get csidrivers.storage.k8s.io
NAME                     ATTACHREQUIRED   PODINFOONMOUNT   TOKENREQUESTS
```

```
REQUIRESREPUBLISH   MODES        AGE
ch.ctrox.csi.s3-driver   false            true             <unset>
false               Persistent   36m
```

И все поды csi должны быть запущены

```
kubectl --namespace kube-system get pods | grep csi
csi-attacher-s3-0               1/1     Running   0             9h
csi-provisioner-s3-0            2/2     Running   0             9h
csi-s3-f5v74                    2/2     Running   0             9h
```

Настройка класса и PVC

Добавляем StorageClass

```
---
kind: StorageClass
apiVersion: storage.k8s.io/v1
metadata:
  name: csi-s3
provisioner: ch.ctrox.csi.s3-driver
parameters:
  # specify which mounter to use
  # can be set to rclone, s3fs, goofys or s3backer
  #mounter: rclone
  mounter: "s3fs"
  otherOpts: "-o allow_other -o uid=0 -o gid=0 -o umask=000"
  # to use an existing bucket, specify it here:
  bucket: bucketone
  path: "pvc-fbd999ab-1ba7-4b07-922d-51270e6028d9"
  csi.storage.k8s.io/provisioner-secret-name: csi-s3-secret
  csi.storage.k8s.io/provisioner-secret-namespace: kube-system
  csi.storage.k8s.io/controller-publish-secret-name: csi-s3-secret
  csi.storage.k8s.io/controller-publish-secret-namespace: kube-system
  csi.storage.k8s.io/node-stage-secret-name: csi-s3-secret
  csi.storage.k8s.io/node-stage-secret-namespace: kube-system
  csi.storage.k8s.io/node-publish-secret-name: csi-s3-secret
  csi.storage.k8s.io/node-publish-secret-namespace: kube-system
  sslVerify: "false"
reclaimPolicy: Retain
volumeBindingMode: Immediate
```

mounter rclone на финальном этапе отказался создавать файлы.

В директории бакета была создана директория pvc-fbd999ab-1ba7-4b07-922d-51270e6028d9/csi-fs Не знаю почему. но при команде ls (проверка) он находился внутри нее.

Добавляем PVC

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-s3-pvc
  namespace: default
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: csi-s3
```

Проверка:

```
apiVersion: v1
kind: Pod
metadata:
  name: minio-test-pod
spec:
  containers:
  - name: app
    image: alpine
    command: ["sleep", "infinity"]
    volumeMounts:
    - name: minio-storage
      mountPath: /mnt/minio
  volumes:
  - name: minio-storage
    persistentVolumeClaim:
      claimName: csi-s3-pvc
      readOnly: false
```

Команда

```
kubectl exec -it minio-test-pod -- sh -c "echo 'New test' > /mnt/minio/test2.txt && ls /mnt/minio"
```

должна вывести созданный файл в списке.

Основные команды

Команда	Доп. пар.	Описание
kubectl get csidrivers.storage.k8s.io		Список CSI-драйверов
kubectl get storageclass -o wide		Какой PROVISIONER используется
kubectl get pods -n kube-system | grep csi		Поды CSI-драйверов. Какие поды отвечают за CSI
kubectl get daemonsets -n kube-system		Где работает CSI-драйвер

Revision #7
Created 22 March 2025 16:20:07 by Admin
Updated 28 March 2025 00:29:55 by Admin