CEPH MANUAL DEPLOYMENT

Развёртывание кластера Ceph в ручном режиме. Производилось на релизе **Luminous** (ceph-2:12.2.2-0.el7), кластер из 3-х машин, на всех запущены по одному OSD, MON, MGR и MDS. Устанавливаем Ceph только на нодах.

Легенда

```
[all]        Выполнить операцию на всех хостах кластера (если ставим какой-то демон - то на всех х
остах, на которых должен быть запущен этот демон)
[one]        Выполнить операцию на одном хосте кластера (на том, который условно назначен "головны
м")
[one, repl]  Выполнить операцию на одном хосте и реплицировать результат на все остальные (наприме
р, скопировать файлик)
{...}        Подставить строку (например, взятую из результата выполнения предыдущей команды)
${..}        Подставить строку или объявленную ранее переменную (для удобства массового выполнения
 команд)
nodename     Имя ноды. Можно подсмотреть, выполнив hostname -s
```

Сам процесс

- ВОРНИНГ: Файлики, получаемые командами вида ***map***, имеют бинарный формат, и просматривать их cat'ом достаточно бессмысленно. В худшем случае вы сломаете себе терминал.

```
[all: установить пакеты]
yum -y install https://download.ceph.com/rpm-luminous/el7/noarch/ceph-release-1-1.el7.noarch.rpm &
& yum -y install ceph

[one: сгенерировать уникальный UID кластера]
uuidgen

;-- Все значения прописываются без {}, пояснения ниже --
;-- {nodenameX} - имя ноды, вывод hostname -s --
;-- {ipX} - локальный IP, например 10.0.0.2 --
;-- {internal network} - локальная сеть, например 10.0.0.0/23 --
;-- {internal OR external network} - указываем локальную сеть, например 10.0.0.0/23 --
[all: сформировать главный конфигурационный файл]
cat > /etc/ceph/ceph.conf << 'EOF'
[global]
fsid = {UUID}
mon initial members = {nodename1}, {nodename2}, {nodename3}
mon host = {ip1}, {ip2}, {ip3}
cluster network = {internal network}
public network = {internal OR external network}
osd journal size = 1024
osd pool default size = 3
osd pool default min size = 2
EOF

[one: сгеренировать временный ключ для мониторов]
ceph-authtool --create-keyring /tmp/ceph.mon.keyring --gen-key -n mon. --cap mon 'allow *'

[one, repl: сгенерировать ключ администратора кластера и распространить его на все ноды]
ceph-authtool --create-keyring /etc/ceph/ceph.client.admin.keyring --gen-key -n client.admin --set
-uid=0 --cap mon 'allow *' --cap osd 'allow *' --cap mds 'allow *' --cap mgr 'allow *'

; cat /etc/ceph/ceph.client.admin.keyring
; cat > /etc/ceph/ceph.client.admin.keyring

[one, repl: сгенерировать ключ для ceph-volume и распространить его на все osd-ноды]
```

```
sudo -u ceph ceph-authtool --create-keyring /var/lib/ceph/bootstrap-osd/ceph.keyring --gen-key -n
client.bootstrap-osd --cap mon 'profile bootstrap-osd'

; cat /var/lib/ceph/bootstrap-osd/ceph.keyring
; cat >/var/lib/ceph/bootstrap-osd/ceph.keyring

;-- Установка MON
[one, repl: импортировать ключи и распространить на все mon-ноды]
ceph-authtool /tmp/ceph.mon.keyring --import-keyring /etc/ceph/ceph.client.admin.keyring
ceph-authtool /tmp/ceph.mon.keyring --import-keyring /var/lib/ceph/bootstrap-osd/ceph.keyring
monmaptool --create --add {nodename1} {ip1} --add {nodename2} {ip2} --add {nodename3} {ip3} --fsid
 {UUID} /tmp/monmap
chmod 0644 /tmp/ceph.mon.keyring /tmp/monmap

; cat /tmp/ceph.mon.keyring
; cat >/tmp/ceph.mon.keyring
; scp /tmp/monmap

[all: настроить мониторы]
sudo -u ceph mkdir /var/lib/ceph/mon/ceph-${nodename}
sudo -u ceph ceph-mon --mkfs -i ${nodename} --monmap /tmp/monmap --keyring /tmp/ceph.mon.keyring
sudo -u ceph touch /var/lib/ceph/mon/ceph-${nodename}/done
systemctl enable --now ceph-mon@${nodename}

; -- Установка MGR -- произвести на всех mon-нодах
[all]
sudo -u ceph mkdir /var/lib/ceph/mgr/ceph-${nodename}
ceph auth get-or-create mgr.${nodename} mon 'allow profile mgr' osd 'allow *' mds 'allow *' | sudo
 -u ceph tee /var/lib/ceph/mgr/ceph-${nodename}/keyring
systemctl enable --now ceph-mgr@${nodename}

; -- Настройка OSD -- полуавтоматический вариант
; -- !!! Создаются pv, vg, lv. При таком способе скормить Цефу готовый lv, скорее всего, не получи
тся.
[all]
ceph-volume lvm prepare --data /dev/{block device or disk partition} --bluestore
ceph-volume lvm list
ceph-volume lvm activate {osd id} {osd fsid}

; -- Установка MDS --
; -- ${id} - произвольное значение, на каждой ноде присваиваем уникальное значение id=1a, id=2b, i
d=3c --
[all]
sudo -u ceph mkdir /var/lib/ceph/mds/ceph-${id}
sudo -u ceph ceph-authtool --create-keyring /var/lib/ceph/mds/ceph-${id}/keyring --gen-key -n mds.
${id}
ceph auth add mds.${id} osd "allow rwx" mds "allow" mon "allow profile mds" -i /var/lib/ceph/mds/c
eph-${id}/keyring
systemctl enable --now ceph-mds@${id}

; -- Создание пула для CephFS --
; при создании пулов надо иметь в виду, что рекомендованное количество групп расположения (placeme
nt groups, pg)
; на небольших сетапах должно быть около 128 (см. скрипт ниже)
[one]
ceph osd pool create cephfs_data 32
ceph osd pool create cephfs_metadata 32
ceph fs new cephfs cephfs_metadata cephfs_data
ceph fs ls
ceph mds stat
ceph osd lspools

; -- Создание пула под rbd (для PV требуется отдельный пул, где kube - название пула) --
; -- Пулам можно давать произвольные, понятные названия, в зависимости от применения --
ceph osd pool create kube 32
ceph osd pool application enable kube kubernetes ; -- ставим метку -- начиная с Luminous обязатель
но, иначе HEALTH_WARN
```

```
; -- Мы создаем разных пользователей для CephFS и для RBD --
;
; -- Создание пользователя для RBD --
ceph auth get-or-create client.{username} mon 'allow r, allow command "osd blacklist"' osd 'allow
rwx pool=kube'
;
; -- Создание пользователя для CephFS --
; -- В данном примере, {data path} - это каталог куда будет писать приложение, например /redis или
 /mysql --
ceph auth get-or-create client.{username} mon 'allow r' mds 'allow r, allow rw path=/{data path}'
osd 'allow rw pool=cephfs_data' ; <-- !!! здесь должен быть пул cephfs, а не kube

; -- Тестовое подключение выполняется под пользователем CephFS
mount.ceph IP_or_Hostname_Node:/ /mnt/cephfs -o name={username},secret={key}

; -- Важно создать каталог приложения (/redis, /mysql...), подключившись под учетной записью admin
 --
; -- Ограничение сделано для того, чтобы одно приложение не смогло убить или испортить данные друг
ого --
; -- Ключ можно посмотреть в конфиге /etc/ceph/ceph.client.admin.keyring --
mount.ceph IP_or_Hostname_Node:/ /mnt/cephfs -o name=admin,secret=AQAcU7JaXXXXXXXXXXXXXXXXXmQ==

; -- Посмотреть список созданных пулов можно командой --
ceph osd lspools
```

Интеграция с k8s / RBD

- Если на ноде, входящей в кластер kubernetes, не планируется поднимать кластер ceph, а только нужна возможность его использования, то ограничиваемся первым шагом вышеприведённой инструкции (где **yum install**).

Проверяем права пользователя RBD

Важно проверить, что права созданного пользователя выглядят аналогично приведенным ниже для пользователя mongouser:

```
#ceph auth get client.mongouser

exported keyring for client.mongouser
[client.mongouser2]
    key = AQAXXXXXXXXXXXXXXc1k0IQ==
    caps mon = "allow r, allow command "osd blacklist""
    caps osd = "allow rwx pool=mongodb"
```

Создаем секреты

Секрет, который позволит kubernetes управлять ceph (создавать/удалять диски), создаем в системном NS

```
ceph auth get-key client.admin
kubectl create secret generic ceph-secret --type="kubernetes.io/rbd" --from-literal=key='AQAcU7JaU
4NALBAyyyyyyyy==' --namespace=kube-system
```

Секрет для доступа пода к RBD создаем в NS приложения

```
ceph auth get-key client.{username}
kubectl create secret generic ceph-secret-username --type="kubernetes.io/rbd" --from-literal=key='
AQAcU7Jaxxxxxxx==' --namespace=app
```

Секрет для доступа пода к CephFS создаем в NS приложения
Обратите внимание в данном случае тип секрета не указываем, потому что подключаем напрямую, а не через механизм Claim

```
ceph auth get-key client.{username}
kubectl create secret generic ceph-fs-secret-username --from-literal=key='AQAcU7Jaxxxxxxx==' --nam
espace=app
```

Создаем Storage-Class (на каждый pool)

Обязательно создавать на каждый Pool в Ceph создаем свой StorageClass!
Важные поля - name: имя указывается в PVC, чтобы оно знало где создавать тома

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: kube
provisioner: kubernetes.io/rbd
parameters:
  monitors: <monitor-1-ip>:6789, <monitor-2-ip>:6789, <monitor-3-ip>:6789
  adminId: admin
  adminSecretName: ceph-secret
  adminSecretNamespace: "kube-system"
  pool: kube
  userId: {username}
  userSecretName: ceph-secret-username
```

Создаем PersistentVolumeClaim

PVC создается в namespace приложения.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: prometheus-3gb
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 3Gi
  storageClassName: kube_rbd
```

На этом настройка завершена, далее можно создавать Pod или Deployment и монтировать туда RBD.

Пример: Pod / PVC

```
mongodb-pod-pvc.yaml
apiVersion: v1
kind: Pod
metadata:
  name: mongodb
spec:
```

```
  containers:
  - image: mongo
    name: mongodb
    volumeMounts:
    - name: mongodb-data
      mountPath: /data/db
    ports:
    - containerPort: 27017
      protocol: TCP
  volumes:
  - name: mongodb-data
    persistentVolumeClaim:
      claimName: prometheus-3gb
```

Важно указать mountPath: /data/db, так как это монтирование по-умолчанию для mongodb.

Пример: Deployment / PV / PVC

Создаем деплоймент с PV и PVC

```
apiVersion: extensions/v1beta1
kind: Deployment
spec:
  template:
    spec:
      containers:
      - command:
        volumeMounts:
        - name: cephfs
          mountPath: /{path}
        - name: data-rbd
          mountPath: /data/rbd
      volumes:
      - name: cephfs
        cephfs:
          monitors:
          - {nodename}
          path: /{namespace}/{app name}
          user: {user}
          secretRef:
            name: ceph-fs-secret-username
      - name: data-rbd
          persistentVolumeClaim:
            claimName: prometheus-3gb
```

Права доступа к volumes

По-умолчанию устанавливаются права 755 на каталог:

```
root@mongodb:/# ls -l /data/db/ -d
drwxr-xr-x. 5 mongodb root 4096 Apr  3 14:22 /data/db/
```

Часто приложениям этого недостаточно, требуется выставить права 777 или поставить корректную группу. Kubernetes позволяет установить группу.
Нужно выставить:

```
securityContext:
    fsGroup: 999
```

Где 999 это GID.

В итоге описание пода будет выглядеть таким образом:

```
apiVersion: v1
kind: Pod
metadata:
  name: mongodb
spec:
  securityContext:
    runAsUser: 1000
    fsGroup: 1000
  containers:
  - image: mongo
    name: mongodb
    volumeMounts:
    - name: mongodb-data
      mountPath: /data/db
    ports:
    - containerPort: 27017
      protocol: TCP
  volumes:
  - name: mongodb-data
    persistentVolumeClaim:
      claimName: mongodb-1gb
```

После мы увидим что GID был изменен с root на mongodb:

```
root@mongodb:/# ls -l /data/db -d
drwxrwsr-x. 6 mongodb mongodb 4096 Apr  6 14:56 /data/db
```

Боротьба с блокировками

В том случае, если по недомыслию для rbd был создан юзер без "allow command "osd blacklist"", то, скорее всего, поды, использующие rbd, не будут проходить ContainerCreating, ругаясь в describe на заблокированные тома. Для исправления следует

- переделать права пользователя (в данном примере -- kube)

```
ceph auth get client.kube
ceph auth caps client.kube mon 'allow r, allow command "osd blacklist"' osd 'allow rwx pool=ku
be'
ceph auth get client.kube
```

- удалить пользователя можно командой (в данном примере --kube)

```
ceph auth del client.kube
```

- удалить мёртвые блокировки (в данном примере пул 'kube')

```
rbd -p kube ls
```

```
rbd -p kube lock list {device}
rbd lock remove kube/{device} {ID} {Locker}
```

- перескейлить пострадавший deployment или statefulset

Перезагрузка узла

При ребуте узла, на котором смонтированы поды с rbd дисками из-за бага в systemd (на 10.04.2018) сервер виснет. помогает железная кнопка.
Поэтому перед плановой перезагрузкой обязательно делаем drain

```
kubectl drain node-1 --force --grace-period=10 --ignore-daemonsets=true
```

Опционально на узле можно проверить что все rbd отмонтированы:

```
mount | grep rbd
```