



Серв и гипервизоры

Виталий Филиппов

КУПЛЕНО НА
SKLADCHIK.COM

Серh и гипервизоры

Виталий Филиппов

Разработчик-эксперт в компании CUSTIS, линуксоид. Занимаюсь разработкой на разных языках от node.js до C++, сильно упоролся по Серh-у. :)

Автор статьи «Производительность Серh»



Интеграция Ceph и гипервизоров*



1. Виртуализация — KVM-гипервизоры
Самое простое, ибо в QEMU есть родной userspace драйвер Ceph
2. Виртуализация — Закрытые гипервизоры
3. Kubernetes
4. Демо: OpenNebula во вложенной VM

* гипервизоров/оркестраторов/“Cloud-Native решений”

KVM-оркестраторы

Наиболее популярные:

1. OpenNebula
2. Proxmox VE (есть даже встроенный установщик Ceph)
3. OpenStack
4. oVirt (он же RHEV)
5. CloudStack

KVM-оркестраторы

Полного идеала нет, все немножко безумные

1. OpenNebula — XML в БД, хрупкая bash-интеграция с Ceph, пытаются коммерциализироваться странными способами
2. Proxmox VE — Corosync кластер, до 32 нод
3. OpenStack — очень много компонентов, совсем не легковесный
4. oVirt — UI на GWT (deprecated), Ceph через OpenStack Cinder
5. Но так или иначе поддержка Ceph везде есть

Нюансы

1. Нормальная настройка ВМ. virtio, cache=writeback. Для CI/CD cache=unsafe, rbd cache writethrough until flush=false (из коробки в PVE).
2. ЕС RBD. Поддерживается(*) в OpenNebula 5.8+, не поддерживается в OpenStack и Proxmox. Workaround: задать rbd default data pool
3. Снапшоты. Как сделаны?
Proxmox: rbd snapshot. Плюс: не тормозят. Минус: откат ужасен, медленный и приводит к потере CoW после ребаланса.
OpenNebula: rbd clone. Плюс: быстрый откат. Минус: тормозят, не всегда легко удаляются.
4. Клонирование ВМ. В Serf можно сделать CoW-клон, но ONE и PVE этого не умеют. OpenNebula, правда, легко патчится.

Kubernetes PersistentVolume

1. Сначала были [cephfs-provisioner](#) и [rbd-provisioner](#) (именно они упомянуты на [kubernetes.io](#)) — сейчас устаревшие
2. Сейчас актуален [ceph-csi](#) (Container Storage Interface)
3. Либо [Rook](#) (к чести авторов, это не только “цеф в кубере”), который может сам разворачивать ceph-csi
4. ceph-csi подходит в том числе для KubeVirt (виртуализация через k8s)
5. ceph-csi поддерживает и RBD, и CephFS⁺¹ в 4 вариантах:
krbd, rbd-nbd, cephfs, ceph-fuse

Нюансы PV

1. Ядерные драйвера `krbd/cephfs` могут повесить процесс в D (Uninterruptible sleep), после чего `k8s` даже не сможет убить контейнер. Ну, хотя... у вас же не бывает даунтайма... :)
2. RBD — это RWO волюмы, там кривоватый fencing: [Issue #578](#)
3. Userspace прокси (`rbd-nbd` и `ceph-fuse`) заметно медленнее ядерных драйверов.
Особенно `ceph-fuse`, в котором нет параллелизма на 1 файловом дескрипторе.
`fio -iodepth=16 == fio -iodepth=1`, при этом `fio -numjobs=16 -group_reporting` на должном уровне.
4. ИМХО, предпочтительный вариант — ядерный CephFS.
5. D можно было бы полечить через микроВМ. Kata Containers?

Rook и RGW (S3)

1. Однако PV — скорее для legacy, которому нужна именно ФС, а не S3
2. Поэтому интересно было бы управлять RGW (S3) через k8s.
И такое тоже есть — в Rook!
3. Нет, разворачивать Ceph в k8s не надо (в IT “можно” не значит “нужно”)
4. Rook умеет интегрироваться с внешним кластером Ceph
5. ...и управлять пользователями и бакетами в RGW (!)
ObjectBucketClaim, ObjectStoreUser
6. Таким образом, не так уж и бесполезен

Импорт образа в Серх+OpenNebula извне

1. Без загрузки через Web
2. Сначала импортируем в Серх

```
rbd import --data-pool espool <FILE> rpool/<IMAGE>
```

```
# или (создание отдельно, чтобы использовать data-pool)
```

```
rbd create --data-pool espool -s <SIZE> rpool/<IMAGE>
```

```
qemu-img convert -m 16 -W -p -n -S 4194304 -f входной_формат -O raw rbd:rpool/<IMAGE>
```

1. Потом импортируем в OpenNebula

```
oneimage create --name <IMAGE> --source rpool/<IMAGE> --size 32768 -d <DATASTORE>
```

Настройка прав для ceph-csi CephFS

1. 2 пользователя — один для создания образов, второй для монтирования
2. Для создания образов админ не обязателен (но настройка хитрая):

```
[client.csi]
key = ...
caps mds = "allow * path=/volumes/csi"
caps mgr = "allow module volumes *"
caps mon = "allow r"
caps osd = "allow * tag cephfs data=rfs, allow * pool=fs_meta namespace=csi"
```

1. Команда для создания пользователя:

```
ceph auth get-or-create client.csi mds 'allow * path=/volumes/csi' mgr 'allow module volumes *'
mon 'allow r' osd 'allow * tag cephfs data=rfs, allow * pool=fs_meta namespace=csi'
```

Настройка прав для Rook RGW

1. radosgw-admin пишет напрямую в RADOS
2. Поэтому

```
[client.rook]
  key = ...
  caps mon = "allow r"
  caps osd = "allow * pool=.rgw.root, allow * pool=default.rgw.log, allow *
pool=default.rgw.control, allow * pool=default.rgw.meta, allow * pool=default.rgw.buckets.index,
allow * pool=default.rgw.buckets.data"
```

1. Команда для создания пользователя:

```
ceph auth get-or-create client.rook mon 'allow r' osd 'allow * pool=.rgw.root, allow *
pool=default.rgw.log, allow * pool=default.rgw.control, allow * pool=default.rgw.meta, allow *
pool=default.rgw.buckets.index, allow * pool=default.rgw.buckets.data'
```