CEPH : TP

Yann Dupont

CCIPL / DSIN Université de Nantes

www.univ-nantes.fr

14 Décembre 2016

UNIVERSITÉ DE NANTES

Travaux pratiques

Création d'un cluster CEPH

Création de pools
L'utiliser en RADOS
L'utiliser en RBD
(kRBD, Kvm)
L'utiliser en CephFS
Authentification
Monitoring

Modifier la replication d'un pool
Ajouter un OSD
Changer les règles de placement
Créer un pool en erasure coding
Créer un pool de tiering
Tests de performance
Tolérance à la panne (en provoquer)

Tuning
...

Timing serré !

Création d'un cluster CEPH

En vrai,
pour l'ANF

Désirable

Au moins 3 MON
Au moins 3 MDS
Maximum d'OSD

Bonnes pratiques

MON, MDS, OSD :
machines séparées

MON avec
disque rapide.

MDS avec
disque rapide.

OSD nécessite de la volumétrie
disque dédié, formatage en XFS
Journal sur SSD (mutualisé)

3 VM par cluster
Chaque vm aura mon, mds, osd
(MAUVAISE PRATIQUE)

Déployer 3 VM debian
Créer 3 volumes persistant de 50Go
Accrocher les disques aux VM
Formater les volumes

À faire sur le contrôleur opennebula

Préparation du cluster

Répéter 3 fois

```
oneimage create -d rozostore_image --name YD-dataA3 --type DATABLOC                024G --persistent --prefix vd

onetemplate instantiate 7 --name ceph_A1 --cpu 1 --memory 768                dmin[PRIVATE_666]'
 --net_context --ssh '/home/yann.dupont/.ssh/id_rsa.pub' --d                taA1
```

Repérer ses machines virtuelles

```
onevm list -l IP,USER,NAME
10.100.0.22     yann.dup ceph_A1
10.100.0.23     yann.dup ceph_A2
10.100.0.24     yann.dup ceph_A3
```

le /etc/host

```
alias ceph                -w root@10.100.0.[22-24]"
cephpdsh                0.100.0.22 mona1 mdsa1 osda1' >> /etc/hosts"
cephpd                  10.100.0.23 mona2 mdsa2 osda2' >> /etc/hosts"
ceph                  o '10.100.0.24 mona3 mdsa3 osda3' >> /etc/hosts"
```

C'est FAIT

Hostname cohérent

```
ssh root@10.100.0.22 "ech                /etc/hostname ; /etc/init.d/hostname.sh"
ssh root@10.100.0.23 "e                  > /etc/hostname ; /etc/init.d/hostname.sh"
Ssh root@10.100.0.24                  a3 > /etc/hostname ; /etc/init.d/hostname.sh"
```

Config SSH pour l'installation.

À faire sur le contrôleur opennebula

Générer une clef et la copier sur les mon

```
ssh-keygen -f .ssh/ceph_key

ssh-copy-id -i .ssh/ceph_key.pub root@10.100.0.22
ssh-copy-id -i .ssh/ceph_key.pub root@10.100.0.23
ssh-copy-id -i .ssh/ceph_key.pub root@10.100.0.24
```

nnaitre cette clef

```
E  /config
   na1
    Hostname      10.100.0.22
    User          root
    IdentityFile ~/.ssh/ceph_key
Host mona2
    Hostname      10.100.0.23
    User          root
    IdentityFile ~/.ssh/ceph_key
Host mona3
    Hostname      10.100.0.24
    User          root
    IdentityFile ~/.ssh/ceph_key
```

C'est FAIT

Recopier cett rtout

```
rsync -av .ssh/ceph_key* .ssh/    t@10.100.0.22:/root/.ssh
rsync -av .ssh/ceph_key* .s     root@10.100.0.23:/root/.ssh
rsync -av .ssh/ceph_ke     ig root@10.100.0.24:/root/.ssh
cephpdsh "chown ro     sh/*"
```

Installation de ceph

Choix 1 : ceph-deploy, simple et rapide, mais choix par défaut

Ce qu'on va faire !

Choix 2 : manuellement : meilleur contrôle, mais plus compliqué

```
Installer les sources de ceph
Installer les paquets
Formatter les volumes en XFS, les monter
Ajouter les journaux, les métas données
```

```
Créer le ceph.conf
Créer les clés de sécurité ceph
Créér l'id du cluster
```

```
Créer les mons
Ajouter les OSD
Ajouter les MDS
```

Choix 3 : docker : rapide à démarrer, mais complexe pour aller en production...

```
docker run -d --net=host -v /etc/ceph:/etc/ceph -e MON_IP=10.100.0.26 -e CEPH_PUBLIC_NETWORK=10.100.0.0/24 ceph/demo
```

Ou faire son Dockerfile

Choix 4 : via un outil d'automatisation (puppet, juju, chef,ansible,salt ...)

Automatisation du choix 2

Choix 5 : On peut aussi compiler ...(2H!!)

Groupes de TP

Les IP ne se suivent
Pas forcément

Salle pc2

```
Groupe A
10.100.0.5 PC2-A1
10.100.0.6 PC2-A2
10.100.0.7 PC2-A3
10.100.0.8 PC2-A4

Groupe B
10.100.0.9  PC2-B1
10.100.0.10 PC2-B2
10.100.0.11 PC2-B3
10.100.0.12 PC2-B4

Groupe C
10.100.0.13 PC2-C1
10.100.0.14 PC2-C2
10.100.0.15 PC2-C3
10.100.0.17 PC2-C4
```

Salle pc3

```
Groupe A
10.100.0.18 PC3-A1
10.100.0.19 PC3-A2
10.100.0.20 PC3-A3
10.100.0.21 PC3-A4

Groupe B
10.100.0.26 PC3-B1
10.100.0.28 PC3-B2
10.100.0.29 PC3-B3
10.100.0.30 PC3-B4

Groupe C
10.100.0.31 PC3-C1
10.100.0.32 PC3-C2
10.100.0.33 PC3-C3
10.100.0.34 PC3-C4
```

Laptops

```
Groupe A
10.100.0.35  LAPTOP-A1
10.100.0.36  LAPTOP-A2
10.100.0.37  LAPTOP-A3
10.100.0.38  LAPTOP-A4

Groupe B
10.100.0.39  LAPTOP-A1
10.100.0.40  LAPTOP-A2
10.100.0.41  LAPTOP-A3
10.100.0.42  LAPTOP-A4

Groupe A
10.100.0.43  LAPTOP-A1
10.100.0.44  LAPTOP-A2
10.100.0.45  LAPTOP-A3
10.100.0.46  LAPTOP-A4
```

Pour être plus confortable et Travailler sur toutes machines En // , utiliser pdsh et un alias Depuis votre compte sur hyper6

```
alias cephpdsh=
"pdsh -w root@10.100.0.[5-8]"
```

```
alias cephpdsh=
"pdsh -w root@10.100.0.1[3-5,7]"
```

Groupes de TP

Vérifiez

Le login depuis hyper6
Avec la clef SSH des TP
Sur vos 4 machines

Les hostnames des 4 machines
Les /etc/hosts

Le ssh direct de
mon1,mon2,mon3

Votre alias pdsh

Installation des paquets binaires nécessaires

Choix 1 : ceph-deploy, simple et rapide, mais choix par défaut

```
cephpdsh "echo deb http://download.ceph.com/debian-jewel jessie main >/etc/apt/sources.list.d/ceph.list"

cephpdsh "apt update && apt upgrade -y"

cephpdsh "apt install -y --force-yes ceph-deploy pdsh"
```

À faire sur une des VM

Choix 1 : ceph-deploy

Lancer ceph-deploy depuis un des noeuds

```
ssh root@mon1
```

Provisionner les mon initiaux

```
ceph-deploy new mon1 mon2 mon3
```

Installation des paquetages debian

```
ceph-deploy install mon1 mon2 mon3
```

Création effective des mon

```
ceph-deploy mon create-initial
```

Ceph est déja installé et fonctionnel, mais il n'y a pas d'OSD

```
ceph -s
health HEALTH_ERR
no osds
```

Préparer les volumes des OSD (formattage, etc)

```
ceph-deploy osd create mon1:/dev/vdb
ceph-deploy osd create mon2:/dev/vdb
ceph-deploy osd create mon3:/dev/vdb
```

Création optionnelle des mds (pour cephfs)

```
ceph-deploy mds create mds1 mds2 mds3
```

Cluster CEPH fonctionnel !

```
ceph df
GLOBAL:
    SIZE      AVAIL     RAW USED     %RAW USED
    3055G     3055G         100M             0
POOLS:
    NAME     ID     USED     %USED     MAX AVAIL     OBJECTS
    rbd      0         0         0         1018G           0
```

1 pool par défaut : rbd

Créer un client de test

À faire sur le contrôleur opennebula

Ou via l'interface WWW

```
onetemplate instantiate 7 --name ceph_client_A1 --cpu 1 --memory 768 --nic 'oneadmin[PRIVATE_666]'
 --net_context --ssh '/home/yann.dupont/.ssh/id_rsa.pub'
rsync -av .ssh/ceph_key* .ssh/config root@10.100.0.25:/root/.ssh
ssh root@10.100.0.25 "chown root.root ~/.ssh/*"
```

Copier la config et les clefs ceph depuis un des mon (pas bon pour la sécurité !)

```
root@mon1 : rsync -av /etc/ceph 10.100.0.25:/etc
```

Sur la VM cliente

```
ssh root@10.100.0.25
echo deb http://download.ceph.com/debian-jewel jessie main >  /etc/apt/sources.list.d/ceph.list
apt update && apt upgrade && apt install ceph-common
ceph -s
ceph df
mkdir /mnt/rbd
mkdir /mnt/cephfs
```

Attention !!!

Le client est ici administrateur.

“Cette cascade est réalisée par des professionnels. Ne tentez en aucun cas de le reproduire à la maison.”

/etc/ceph/ceph.conf

ceph.conf à l'issue de l'installation

```
cat /etc/ceph/ceph.conf
[global]
fsid = 33e96316-614a-40d8-aea5-
dba146686d5e
mon_initial_members = mona1, mona2, mona3
mon_host =
10.100.0.22,10.100.0.23,10.100.0.24
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx
```

Beaucoup de customisation possible

ceph.client.admin.keyring

```
[client.admin]
    key = AQALHUxY6GaqBRAAaCqUwzQeQ0JIZr4nCCfe8g==
```

NE DEVRAIT PAS ÊTRE PARTAGÉ

Sur toute machine ayant a clef admin

Manipulation des pools

```
root@debian:~# ceph df
GLOBAL:
    SIZE       AVAIL      RAW USED     %RAW USED
    3055G      3050G         5310M          0.17
POOLS:
    NAME      ID     USED      %USED     MAX AVAIL     OBJECTS
    rbd       0      1742M      0.17          1016G         501
root@debian:~# ceph osd pool get rbd size
size: 3
root@debian:~# ceph osd pool get rbd pg_num
pg_num: 64
```

Pool par défaut

Réplicat x3

64 placement groups

Créer un nouveau pool

```
root@debian:~# ceph osd pool create objets_utiles 64 replicated
pool 'objets_utiles' created
root@debian:~# ceph df
GLOBAL:
    SIZE      AVAIL     RAW USED     %RAW USED
    3055G     3050G        5311M          0.17
POOLS:
    NAME              ID     USED      %USED     MAX AVAIL     OBJECTS
    rbd               0      1742M      0.17         1016G         501
    objets_utiles     2          0         0         1016G           0
```

Stocker & Récupérer un objet

```
root@debian:~#dd if=/dev/urandom of=FichierTireBouchon bs=1M count=16
root@debian:~# md5sum FichierTireBouchon
eb05fa217d1b9569c426b07e92a84854  FichierTireBouchon

root@debian:~#rados put -p objets_utiles ObjetTireBouchon FichierTireBouchon
```

```
root@debian:~# rados ls -p objets_utiles
ObjetTireBouchon
```

```
root@debian:~# rados -p objets_utiles stat ObjetTireBouchon
objets_utiles/ObjetTireBouchon mtime 2016-12-10 19:14:45.000000, size
16777216
```

```
root@debian:~# rados get -p objets_utiles ObjetTireBouchon
FichierTireBouchon2
root@debian:~# md5sum FichierTireBouchon2
eb05fa217d1b9569c426b07e92a84854  FichierTireBouchon2
```

Explications :

```
root@debian:~# ceph osd map objets_utiles ObjetTireBouchon
osdmap e21 pool 'objets_utiles' (2) object 'ObjetTireBouchon' -> pg 2.4384205 (2.5) -> up
([2,1,0], p2) acting ([2,1,0], p2)
```

```
ceph pg dump
```

```
root@mona1:/var/lib/ceph/osd/ceph-0/current/2.5_head# ls -al
total 16396
drwxr-xr-x   2 ceph ceph       72 Dec 10 17:35 .
drwxr-xr-x 260 ceph ceph     8192 Dec 10 17:27 ..
-rw-r--r--   1 ceph ceph        0 Dec 10 17:27 __head_00000005__2
-rw-r--r--   1 ceph ceph 16777216 Dec 10 17:51 ObjetTireBouchon__head_04384205__2

root@mona3:/var/lib/ceph/osd/ceph-2/current/2.5_head# md5sum
ObjetTireBouchon__head_04384205__2
eb05fa217d1b9569c426b07e92a84854  ObjetTireBouchon__head_04384205__2
```

RBD : tests (avec krbd)

```
ceph df
rbd ls
rbd create test1 --size=100G --image-feature layering
rbd info test1
rbd map test1
mkfs.ext4 /dev/rbd0
mount /dev/rbd0 /mnt/rbd
df
ceph df
time cp -avx / /mnt/rbd
df
ceph df
```

2 pools de créés

La commande rbd utilise le pool rbd par défaut

Création d'un volume (idem LUN)

Informations sur le volume créé

On le mappe sur la machine

Remarquer que les 100G n'ont pas étés alloués

Mais qu'ici l'allocation (x3) a eu lieu

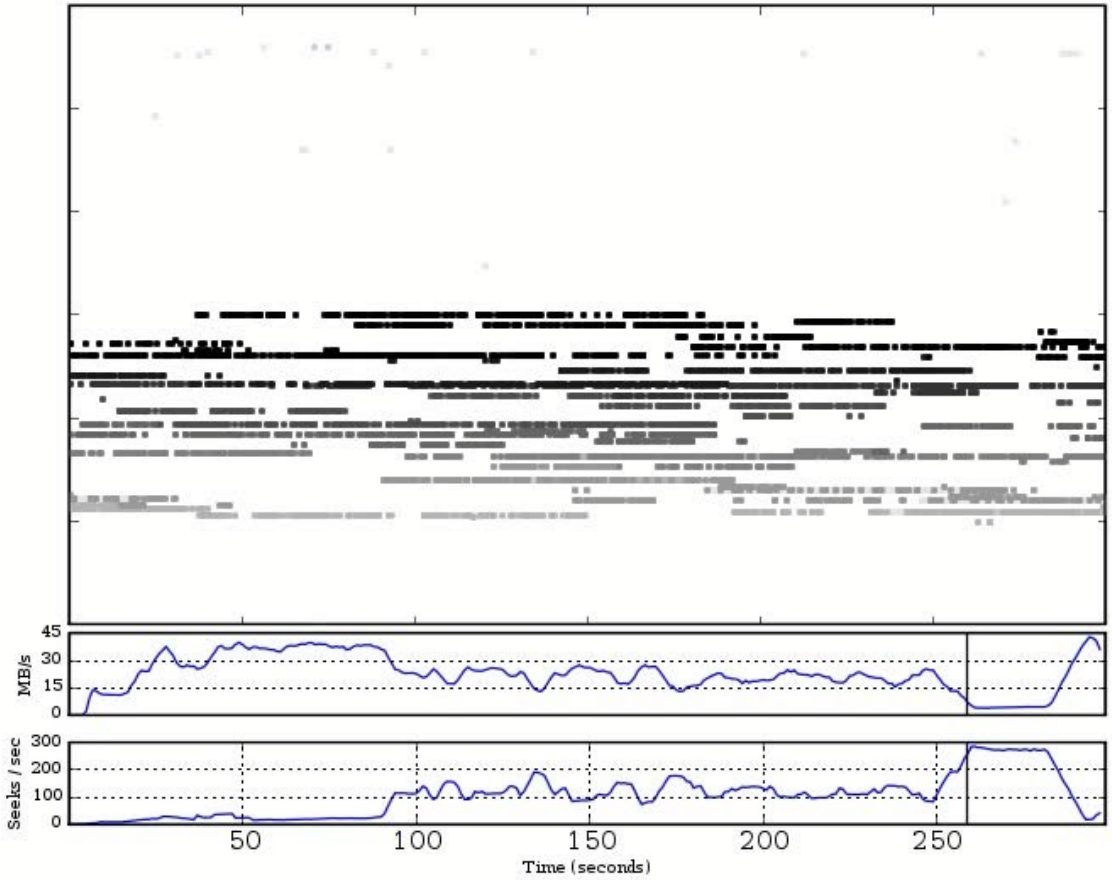
RBD : tests (avec krbd)

```
time rm -rf /mnt/rbd
ceph df
time cp -avx / /mnt/rbd
```

Répéter plusieurs fois.

Allocation à la volée.
Supression depuis fs != suppression de ceph
Nécessite trim (uniquement virtio-scsi dans kvm)

Allocations sur un disque



RBD : SNAPSHOTS, CLONE

```
ceph df
rbd snap test1@snap1
rbd snap ls test1
rbd snap protect test1@snap1
ceph df
rbd clone test1@snap1 test2
ceph df
rbd map test2
mount /dev/rbd1 /mnt/2
```

Le Snapshot n'utilise pas de place

Le clone non plus. Attention à l'UUID

Sur toute machine ayant a clef admin

CEPHFS

Création des pools nécessaires

```
ceph osd pool create ANF_cephfs_data 64 replicated
ceph osd pool create ANF_cephfs_metadata 64 replicated
```

Création du système de fichiers

```
ceph fs new ANF_fs ANF_cephfs_metadata ANF_cephfs_data
ceph fs ls
  name: ANF_fs, metadata pool: ANF_cephfs_metadata,
  data pools: [ANF_cephfs_data ]
ceph fs set_default ANF_fs
```

Sur la VM cliente

Extraire la clef admin du keyring, ajouter le client de montage

```
ceph auth get client.admin | head -2 | tail -1 > /etc/ceph/admin.secret
apt install ceph-fs-common
```

Montage du fs sur le client

```
mount -t ceph mds1,mds2,mds3:/ /mnt/cephfs -o name=admin,secretfile=/etc/ceph/admin.secret
```

RBD VS CEPHFS

Place libre reportée

```
df -h
[..]
/dev/rbd0             99G    1.2G   93G   2% /mnt/rbd
10.100.0.24:/        3.0T    9.5G  3.0T   1% /mnt/cephfs
```

Efficience sur petits fichiers (copie de 35k fichiers, 1,2 Go)

```
root@debian:/#  time cp -ax / /mnt/rbd

real 0m32.595s
user 0m0.216s
sys  0m2.488s

root@debian:/# time cp -ax / /mnt/cephfs/

real 3m30.366s
user 0m0.492s
sys  0m3.908s
```

Authentifications

```
ceph df
ceph auth list
```

```
Rbd ls rbd
rados -p ANF_cephfs_data ls | head
```

```
ceph auth list
ceph auth get-or-create-key client.ANF mon 'allow r' osd 'allow rwx pool=rbd, allow rwx pool=objets_utiles'
```

AQB5cE1Y1O/JDBAAVBtnxzCMu3y30eBXJzlBkw==

Sur la VM cliente

```
cat /etc/ceph/ceph.client.ANF.keyring
[client.ANF]
    key = AQB5cE1Y1O/JDBAAVBtnxzCMu3y30eBXJzlBkw==
mv ceph.client.admin.keyring ceph.client.admin.keyring.old
```

```
ceph df
ceph auth list
```

La VM cliente n'a plus de droits admin

```
ceph df --id=ANF
ceph auth list --id=ANF
```

```
rbd ls rbd --id=ANF
rados -p ANF_cephfs_data –id=ANF  ls | head
```

Monitoring

Donne l'état du cluster en temps réel ou continu

```
ceph -s, ceph -w
```

Scruter /var/log/ceph :

monclient: _check_auth_rotating possible clock skew, rotating keys expired way too early (before 2016-12-11 16:52:00.157565)

Détails de santé du cluster

```
ceph health
HEALTH_WARN mon.mona1 low disk space; mon.mona2 low disk space; mon.mona3 low disk space
```

Donne la hiérachie des objets, indique les osd up / down

```
ceph osd tree
```

Sur un OSD

Socket d'administration dans /var/run/ceph/ceph-osd.xxx.asok

```
ceph --admin-daemon /var/run/ceph/ceph-osd.2.asok perf dump
ceph daemonperf /var/run/ceph/ceph-osd.2.asok
```

Changer les règles d'un pool

ceph -w en simultané

Passer un pool en réplicat 2, le repasser à 3

```
ceph osd pool get rbd size
ceph osd pool set rbd size 2
[Attendre]
ceph osd pool set rbd size 3
```

Augmenter les pg d'un pool

```
ceph osd pool get rbd pg_num
ceph osd pool get rbd pgp_num
ceph osd pool set rbd pg_num 128
[Attendre]
ceph osd pool set rnd pgp_num 128
```

Désactiver / Activer les scrubs

En cas de reconstruction, évite de saturer davantage

```
ceph osd set noscrub
ceph osd set nodeep-scrub
ceph -s
```

```
ceph osd unset noscrub
ceph osd unset nodeep-scrub
ceph -s
```

Ajouter un OSD (1)

```
oneimage create -d rozostore_image --name YD-dataA4 --type DATABLOCK --size 1024G --persistent --prefix vd

onetemplate instantiate 7 --name ceph_A4 --cpu 1 --memory 768 --nic 'oneadmin[PRIVATE_666]'
 --net_context --ssh '/home/yann.dupont/.ssh/id_rsa.pub' --disk=1,YD-dataA4
```

ssh-copy-id -i .ssh/ceph_key.pub root@10.100.0.27

EDIT .ssh/config
[..]
Host mon3
[..]
Host osd4
  Hostname 10.100.0.27
  User root
  IdentityFile ~/.ssh/ceph_key

Peupler le /etc/host

```
alias cephpdsh="pdsh -w root@10.100.0.[22-25,27]"
###cephpdsh "echo '10.100.0.25 clienta1' >> /etc/hosts"
cephpdsh "echo '10.100.0.27 osda4' >> /etc/hosts"
```

rsync -av .ssh/ceph_key* .ssh/config root@10.100.0.27:/root/.ssh
cephpdsh "chown root.root ~/.ssh/*"

À faire sur mona1

Ajouter un OSD (2)

```
root@mon1:~# ceph-deploy install osd4
ceph-deploy osd create osd4:/dev/vdb
ceph -w
```

Les données sont en train de migrer sur le 4 eme OSD

```
root@mona1:~# ceph osd map objets_utiles ObjetTireBouchon
osdmap e258 pool 'objets_utiles' (2) object 'ObjetTireBouchon'
-> pg 2.4384205 (2.5) -> up ([3,2,1], p3) acting ([3,2,1], p3)
```

Même PG, mais OSD différents : était [2,1,0]

```
root@mona1:~# ceph df
```

200 Go bruts.

Règles crush.

```
root@mon1:~# ceph osd getcrushmap -o crushmap
got crush map from osdmap epoch 285
```

```
crushtool -d crushmap -o crushmap.txt
EDIT crushmap.txt
```

```
ceph osd crush add-bucket salle1 room
ceph osd crush add-bucket salle2 room
ceph osd crush add-bucket salle3 room
ceph osd crush move salle1 root=default
ceph osd crush move salle2 root=default
ceph osd crush move salle3 root=default
ceph osd crush move mon1 root=default room=salle1
ceph osd crush move mon2 root=default room=salle2
ceph osd crush move mon3 root=default room=salle3
ceph osd crush move mon4 root=default room=salle1
ceph osd tree
```

Les changements dans la hiérarchie engendrent un fort déplacement des données.

Règles crush.

```
root@mon1:~# ceph osd getcrushmap -o crushmap
root@mon1:~# crushtool -d crushmap -o crushmap.txt
root@mon1:~# EDIT crushmap.txt
```

Ajout d'une nouvelle règle en éditant La crushmap

```
rule replicat_salle {
        ruleset 1
        type replicated
        min_size 1
        max_size 10
        step take default
        step chooseleaf firstn 0 type room
        step emit
}
```

```
root@mon1:~# crushtool -c crushmap.txt -o crushmap2
root@mon1:~# ceph osd setcrushmap -i crushmap2
```

La règle existe dans le cluster mais est Non appliquée.

```
root@mon1:~# ceph osd pool set rbd crush_ruleset 1
set pool 0 crush_ruleset to 1
```

Elle l'est !

Créer un pool d'erasure coding

Plugin isa intel meilleur que jerasure (défaut)

```
ceph osd erasure-code-profile set p2p1ANF plugin=isa k=2 m=1 ruleset-failure-domain=room
ceph osd crush rule create-erasure r2p1AND p2p1ANF
```

```
ceph osd pool create poolEC 32 erasure p2p1ANF
ceph df
```

Place disponible 2x supérieure dans ce pool

```
rbd create poolEC/test --size=1G
2016-12-11 20:16:33.118333 7f025000cd40 -1
librbd: error adding image to directory: (95)
Operation not supported
rbd: create error: (95) Operation not supported
```

Pool EC pas directement utilisable par RBD ou cephFS

```
rados put -p poolEC ObjetTireBouchon FichierTireBouchon
rados get -p poolEC ObjetTireBouchon FichierTireBouchon2
```

Mais utilisable en RADOS.
Raison : pas de support d'écriture partielle

Ajouter un tiers répliqué au dessus

Permet d'utiliser un pool EC sur rbd, cephfs

Peut être intéressant en terme de performance (pool chaud, pool froid)

```
root@debian:~# ceph osd pool create HOT 32 replicat_salle
root@debian:~# ceph df
```

précise la règle de placement des données

```
root@debian:~# ceph osd tier add poolEC HOT
pool 'HOT' is now (or already was) a tier of 'poolEC'
root@debian:~# ceph osd tier cache-mode HOT writeback
set cache-mode for pool 'HOT' to writeback
root@debian:~# ceph osd pool set HOT hit_set_type bloom
set pool 9 hit_set_type to bloom
root@debian:~# ceph osd tier set-overlay poolEC HOT
overlay for 'poolEC' is now (or already was) 'HOT'
```

HOT devient un pool tiers, writeback

Bloom = algorithme du cache/tier

HOT masque le pool sous-jacent

```
root@debian:~# rbd create poolEC/test --image-feature layering --size=100G
root@debian:~# rbd map poolEC/test
/dev/rbd1
root@debian:~# mkfs.xfs -m crc=1,finobt=1 /dev/rbd1
root@debian:~# mount /dev/rbd1 /mnt/EC
```

Corrections d'erreurs

Sur un OSD

```
root@mon2:/var/lib/ceph/osd/ceph-1/current/2.5_head# rm ObjetTireBouchon__head_04384205__2
```

Sur le client

```
rados get -p objets_utiles ObjetTireBouchon FichierTireBouchon2 ; md5sum FichierTireBouchon2
eb05fa217d1b9569c426b07e92a84854  FichierTireBouchon2
```

```
ceph pg scrub 2.5
instructing pg 2.5 on osd.2 to scrub
```

```
ceph pg scrub 2.5
instructing pg 2.5 on osd.2 to scrub
```

```
2016-12-11 15:09:35.174012 osd.2 [INF] 2.5 scrub starts
2016-12-11 15:09:35.176571 osd.2 [ERR] 2.5 shard 1 missing 2:a0421c20:::ObjetTireBouchon:head
2016-12-11 15:09:35.176754 osd.2 [ERR] 2.5 scrub 1 missing, 0 inconsistent objects
2016-12-11 15:09:35.176761 osd.2 [ERR] 2.5 scrub 1 errors
```

```
ceph pg repair 2.5
2016-12-11 15:11:31.192588 osd.2 [INF] 2.5 repair starts
2016-12-11 15:11:31.252131 osd.2 [ERR] 2.5 shard 1 missing 2:a0421c20:::ObjetTireBouchon:head
2016-12-11 15:11:31.252239 osd.2 [ERR] 2.5 repair 1 missing, 0 inconsistent objects
2016-12-11 15:11:31.252256 osd.2 [ERR] 2.5 repair 1 errors, 1 fixed
```

Panne de serveurs

Kill violent d'OSD, de MON

Au niveau d'openNebula, « Shutter un serveur »

Relancer, vérifier...

Crush tunables, support des options

Noyau 3.16 de base.

```
root@debian:~# rbd create rbd/test2 --size=500G
root@debian:~# rbd map rbd/test2
rbd: sysfs write failed
RBD image feature set mismatch. You can disable features unsupported by the kernel with "rbd
feature disable".
```

Cluster Jewel
Noyau client ancien
Upgrader le noyau ou limiter les features

```
root@debian:~# umount -a
root@debian:~# ceph osd crush tunables optimal
```

Raffinement dans le placement des données
Déplace beaucoup de données.

```
root@debian:~# mount /dev/rbd0 /mnt/rbd
[Ne rend pas la main]
```

Le noyau 3.16 ne peut accepter ces règles
LibRBD est plus souple (pour KVM)

Compatibilité krbd

Noyau 4.8 compatible, mais pas toutes
Les options

```
root@debian:~# uname -a
Linux 4.8.10-dsiun-dl-160725 #201 SMP Fri Nov 25 11:37:30 UTC 2016 x86_64 GNU/Linux
```

```
root@debian:~# rbd map test1
/dev/rbd0
root@debian:~# rbd map test2
rbd: sysfs write failed
RBD image feature set mismatch. You can disable features unsupported by the kernel with "rbd
feature disable".
In some cases useful info is found in syslog - try "dmesg | tail" or so.
rbd: map failed: (6) No such device or address
```

```
root@debian:~# rbd info test2
features: layering, exclusive-lock, object-map, fast-diff, deep-flatten
root@debian:~# rbd feature disable test2 exclusive-lock, object-map, fast-diff, deep-flatten
root@debian:~# rbd map test2
/dev/rbd1
```

Merci !

Questions ?