



CEPH, BASES & DÉPLOIEMENT

Yann Dupont

CCIPL / DSIN Université de Nantes

www.univ-nantes.fr

École de stockage IN2P3 – GANIL CAEN- 12-16 juin 2017



UNIVERSITÉ DE NANTES

LES DONNÉES SONT IRREMPLAÇABLES



Une machine en panne se remplace.
Les données perdues le sont à jamais.
Des années de travail peuvent disparaître.

Il s'agit du patrimoine de votre institution.

CEPH ?

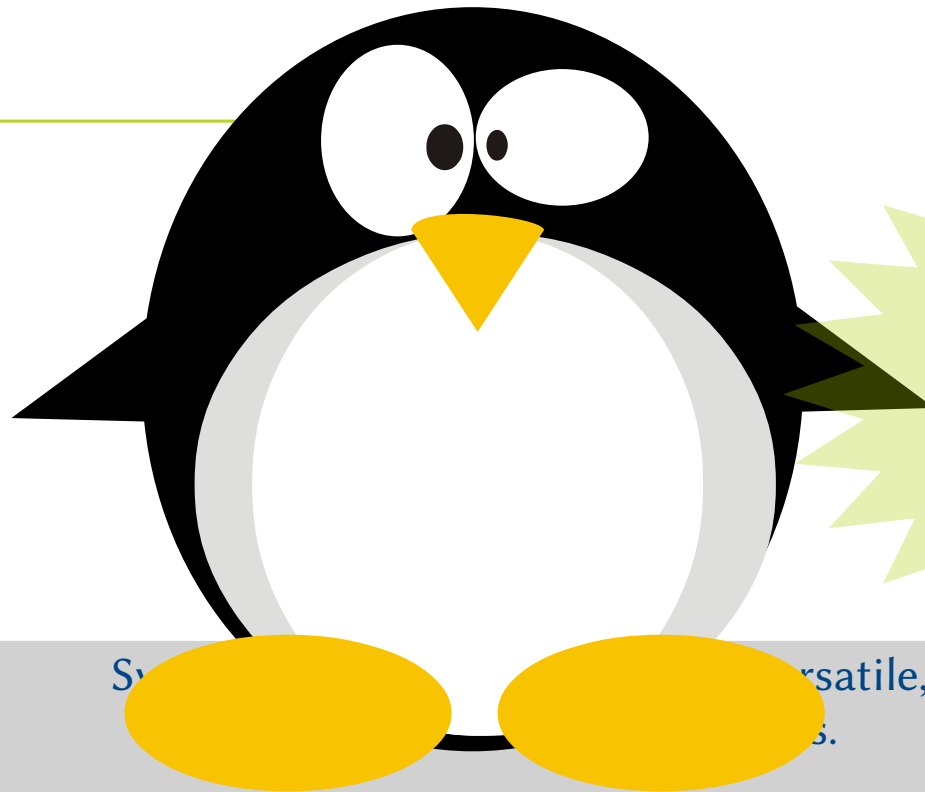


Système de stockage complet et versatile,
utilisable de nombreuses façons.

Architecturé sur un cluster de serveurs standards.
Fournit nativement des interfaces objet, système de fichiers et bloc.

Très utilisé dans le monde open-source : couvre de nombreux cas d'usage.

CEPH ?



Partie serveur
Seulement sous
Linux

(FreeBSD +- OK)

Système versatile,
S.

Architecturé sur un cluster de serveurs standards.
Fournit nativement des interfaces objet, système de fichiers et bloc.

Très utilisé dans le monde open-source : couvre de nombreux cas d'usage.

HISTORIQUE DES VERSIONS

inktank

47 Préversions (!)

07 / 2012 Argonaut (v 0.48)

01 / 2013 Bobtail (v 0.56)

05 / 2013 Cuttlefish (v 0.61)

08 / 2013 Dumpling (v0.72) LTS

11 / 2013 Emperor (v 0.67)

05 / 2014 Firefly (v0.80) LTS

10 / 2014 Giant (v0.87)

04 / 2015 Hammer (v0.94) LTS

11 / 2015 Infernalis (v9.2.z)

04 / 2016 Jewel (v10.2.z) LTS

01 / 2017 Kraken (v11.2.z)

?? / 2017 Luminous (v12.2.z) LTS

Opensource
<https://github.com/ceph>

Tous les 6 mois
LTS tous les ans

Rachat



Majeure

renumérotation

Base de RedHat CEPH storage v2



NUMÉROS DE VERSIONS

Version X.Y.Z

X pair = version à long support (LTS)

Y=0 development / preview

Y=1 beta

Y=2 stable

Z = version mineure

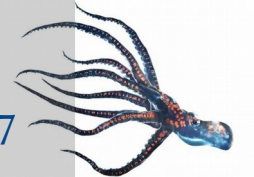
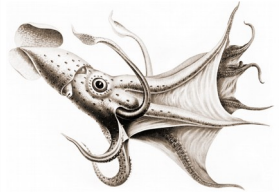
Versions actuelle :

LTS v10.2.7 (11 avril 2017)
(Nom de code Jewel)

Support : (open source)
Jewel 04/2016 → 11/2017

v11.2.0 (Kraken) : 20/01/2017

LTS v12.2.0 : Sans doute juillet
(Nom de code Luminous)



LUMINOUS



UNIVERSITÉ DE NANTES

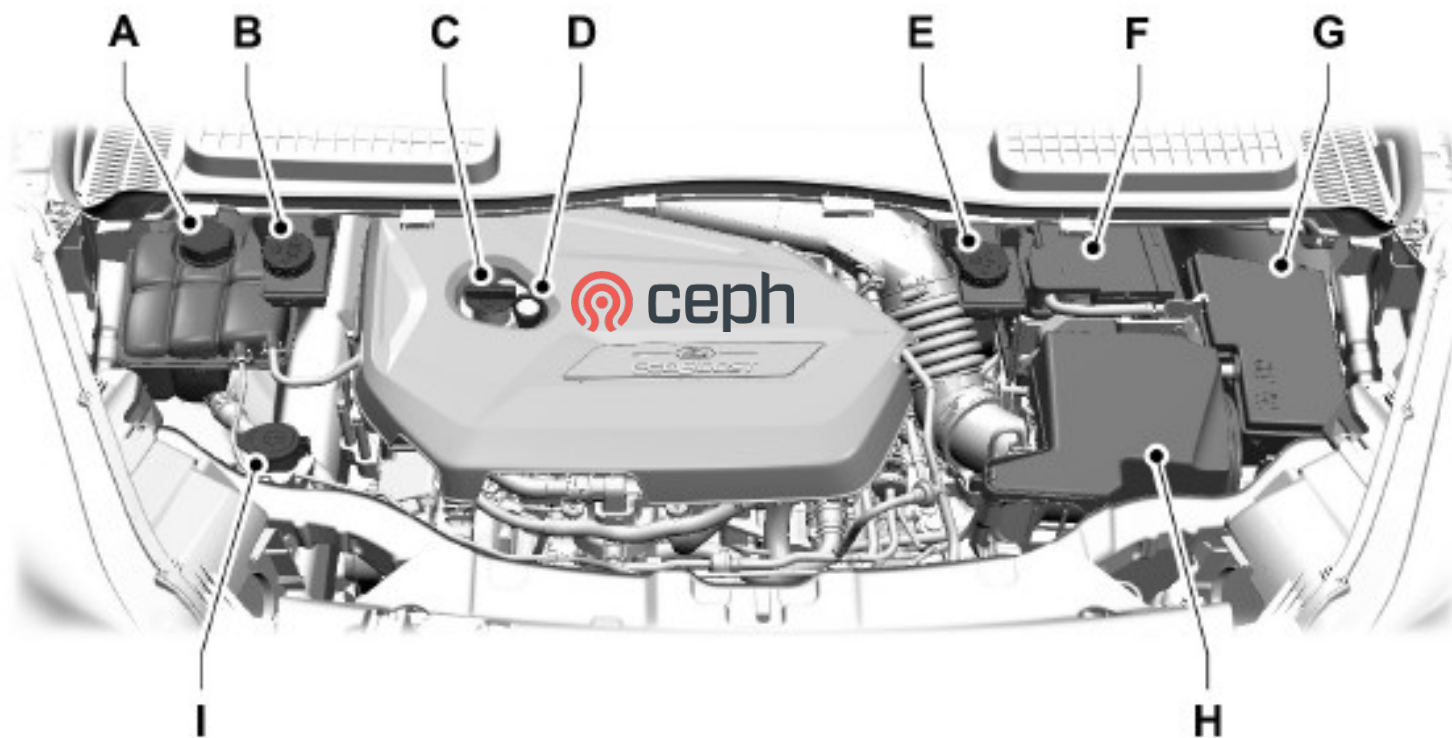
INSTALLATION DE LA SOUCHE LOGICIELLE

```
## exemple debian : /etc/apt/sources.list.d/ceph.list  
  
##ceph  
deb http://download.ceph.com/debian-jewel jessie main  
  
apt update ; apt install ceph
```

Il est aussi possible de compiler (préparez le café ...)

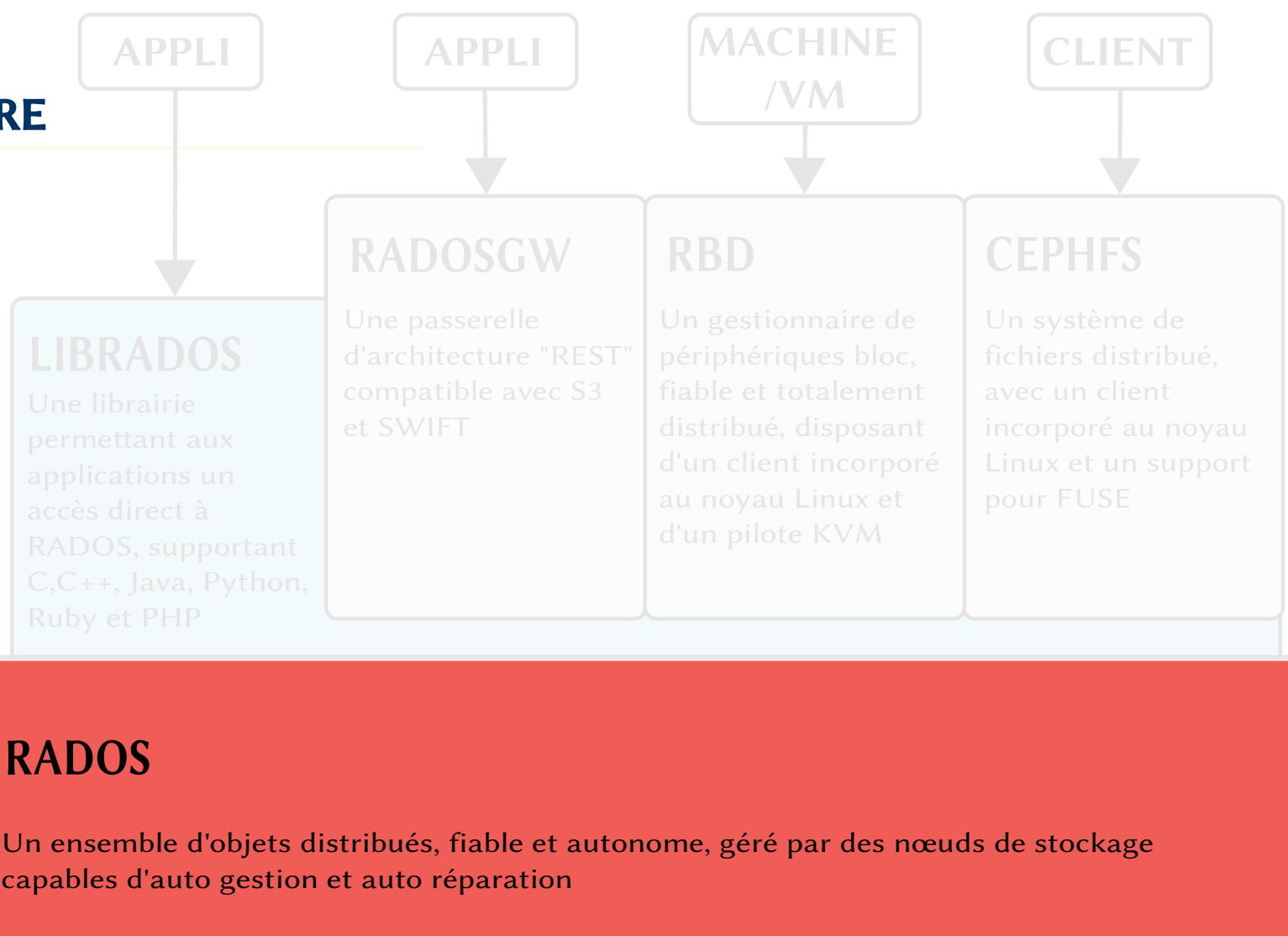


SOUS LE CAPOT ...



ARCHITECTURE

1



ARCHITECTURE

2

APPLI

APPLI

MACHINE
/VM

CLIENT

LIBRADOS

Une librairie permettant aux applications un accès direct à RADOS, supportant C, C++, Java, Python, Ruby et PHP

RADOSGW

Une passerelle d'architecture "REST" compatible avec S3 et SWIFT

RBD

Un gestionnaire de périphériques bloc, fiable et totalement distribué, disposant d'un client incorporé au noyau Linux et d'un pilote KVM

CEPHFS

Un système de fichiers distribué, avec un client incorporé au noyau Linux et un support pour FUSE

RADOS

Un ensemble d'objets distribués, fiable et autonome, géré par des nœuds de stockage capables d'auto gestion et auto réparation

ARCHITECTURE

APPLI

APPLI

MACHINE
/VM

CLIENT

3

LIBRADOS

Une librairie permettant aux applications un accès direct à RADOS, supportant C, C++, Java, Python, Ruby et PHP

RADOSGW

Une passerelle d'architecture "REST" compatible avec S3 et SWIFT

RBD

Un gestionnaire de périphériques bloc, fiable et totalement distribué, disposant d'un client incorporé au noyau Linux et d'un pilote KVM

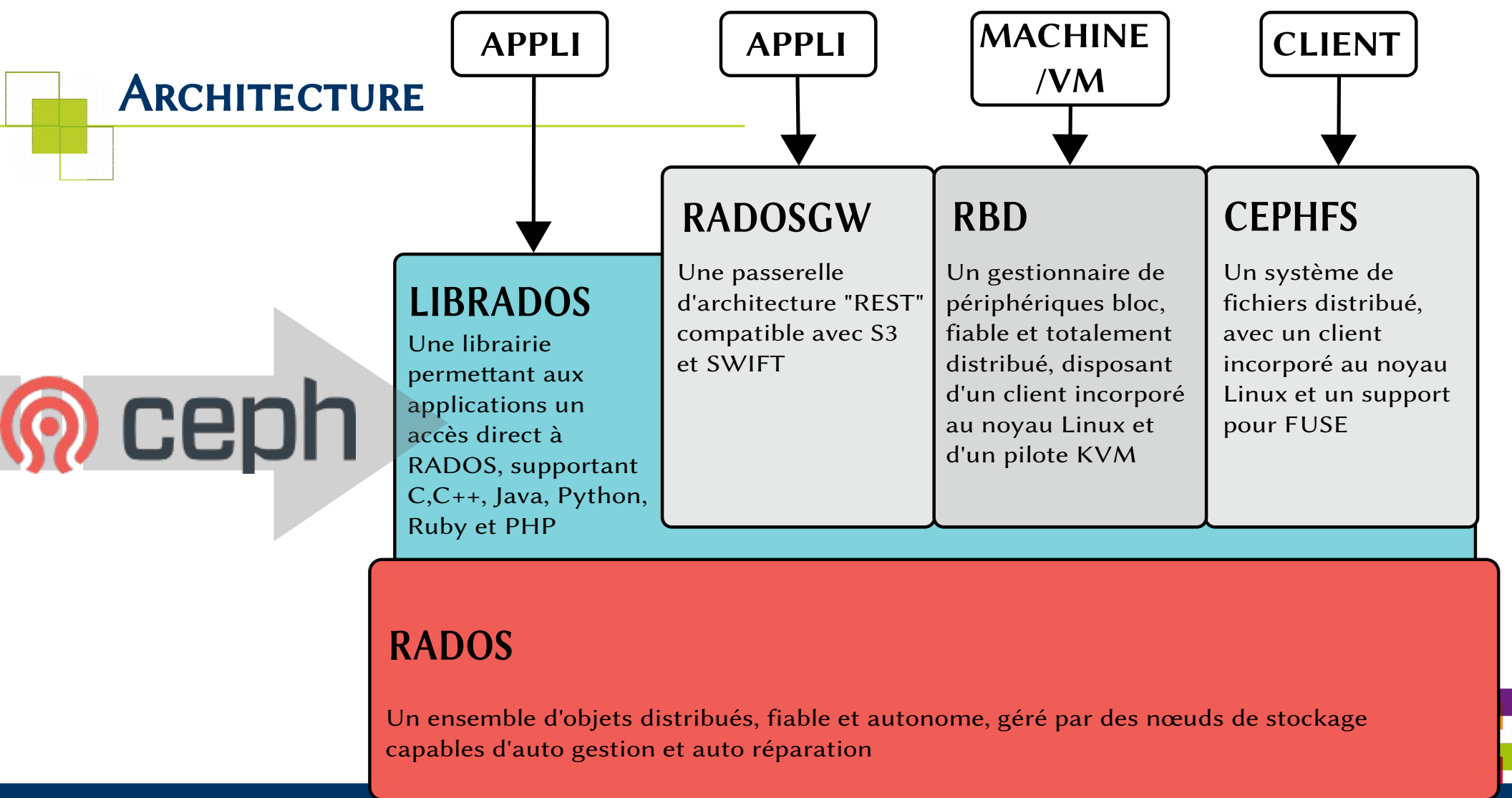
CEPHFS

Un système de fichiers distribué, avec un client incorporé au noyau Linux et un support pour FUSE

RADOS

Un ensemble d'objets distribués, fiable et autonome, géré par des nœuds de stockage capables d'auto gestion et auto réparation

ARCHITECTURE



ARCHITECTURE DE CEPH

Nombre impair



MON
Monitor

Vérifie le bon état du cluster
Assure la communication initiale
avec les clients
Vérifie les droits d'accès
Machine (VM) dédiée conseillée.

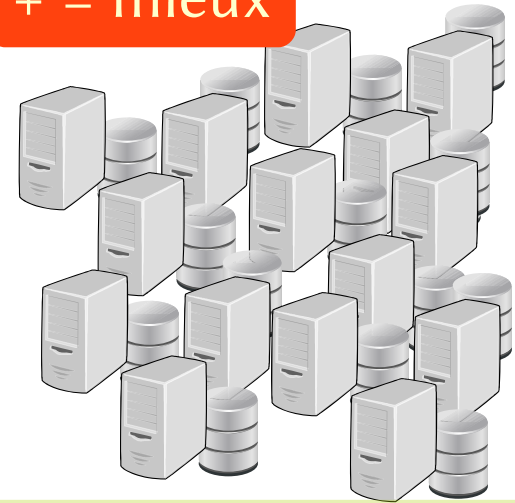
Nombre impair



MDS
Meta Data Server

Gère les méta données
OPTIONNEL
(uniquement pour CephFS)

+ = mieux



OSD
Object Storage Daemon

Stocke les objets
sur filesystem local (XFS)
Communique avec les clients
Débit disque et réseau

ARCHITECTURE DE CEPH

Nombre impair



MON
Monitor

Vérifie le bon état du cluster
Assure la communication initiale
avec les clients
Vérifie les droits d'accès
Machine (VM) dédiée conseillée.

OSD

Nombre impair

Pour éviter le

Split BRAIN

+ = mieux



OSD

Object Storage Daemon

OPTIMISE
(uniquement pour Ceph)

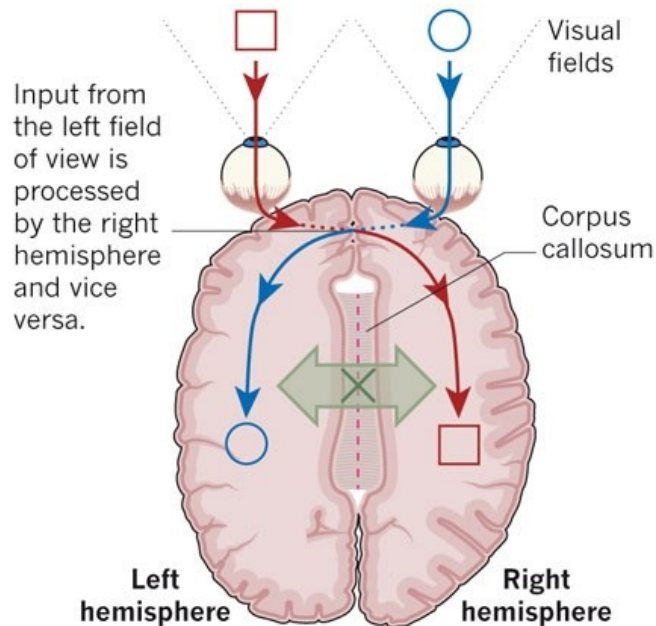
Gère les objets
sur filesystem local (XFS)
Communique avec les clients
Débit disque et réseau

SPLIT BRAIN (SITE NATURE.COM)

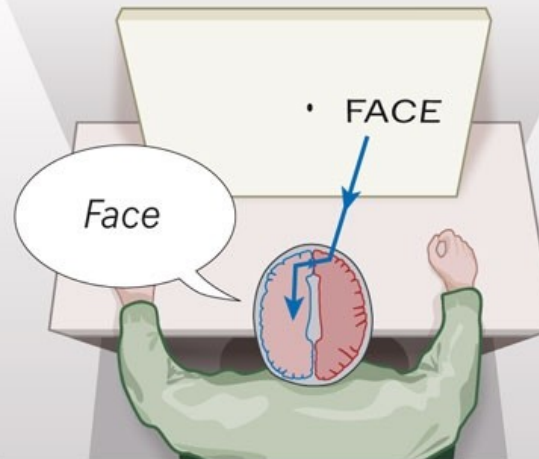
OF TWO MINDS

Experiments with split-brain patients have helped to illuminate the lateralized nature of brain function.

Split-brain patients have undergone surgery to cut the corpus callosum, the main bundle of neuronal fibres connecting the two sides of the brain.

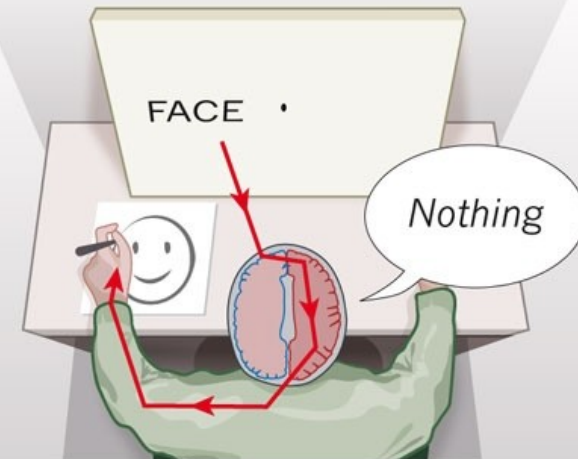


A word is flashed briefly to the right field of view, and the patient is asked what he saw.



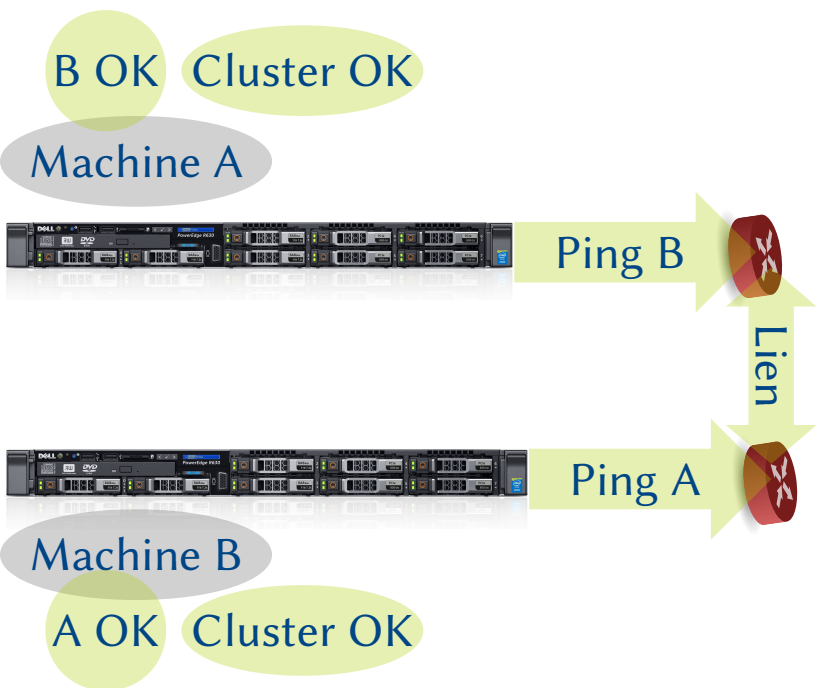
Because the left hemisphere is dominant for verbal processing, the patient's answer matches the word.

Now a word is flashed to the left field of view, and the patient is asked what he saw.



The right hemisphere cannot share information with the left, so the patient is unable to say what he saw, but he can draw it.

SPLIT BRAIN CLUSTER (1)



Les machines A et B se contrôlent régulièrement et se synchronisent.

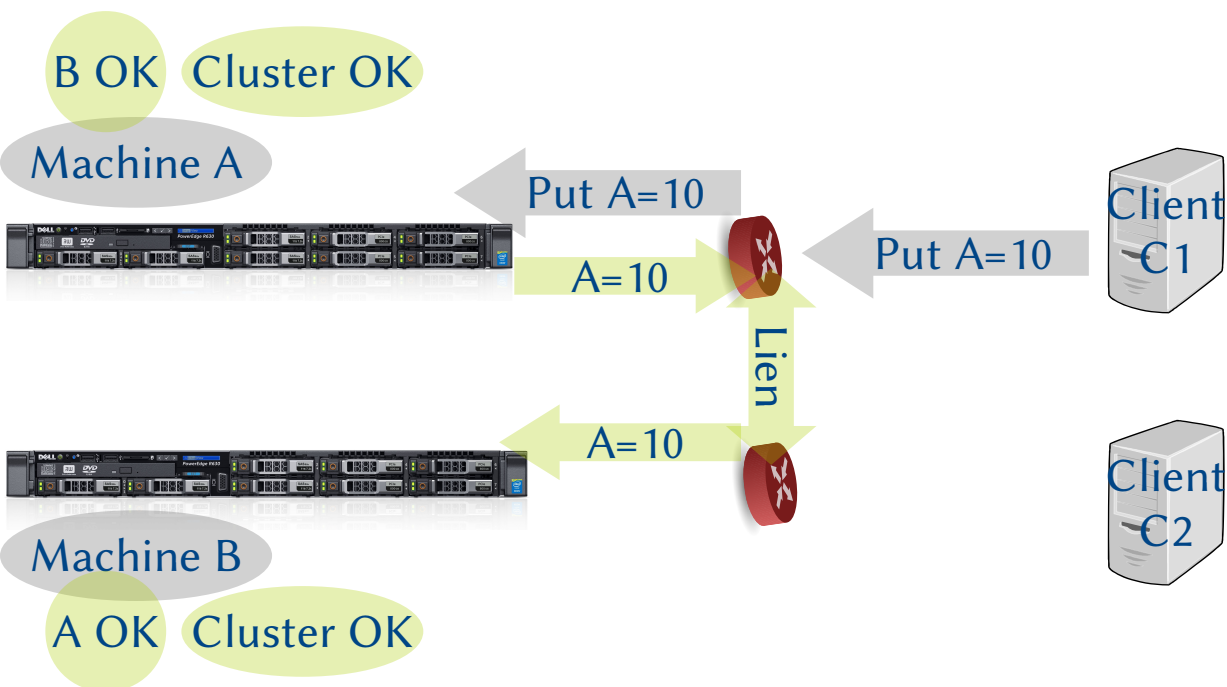
Dans Ceph,
Les mon se surveillent entre eux
Les osd se surveillent entre eux
(et previennent les mons)



Dans Ceph,
PAXOS est utilisé pour gérer la
Cohérence du cluster.

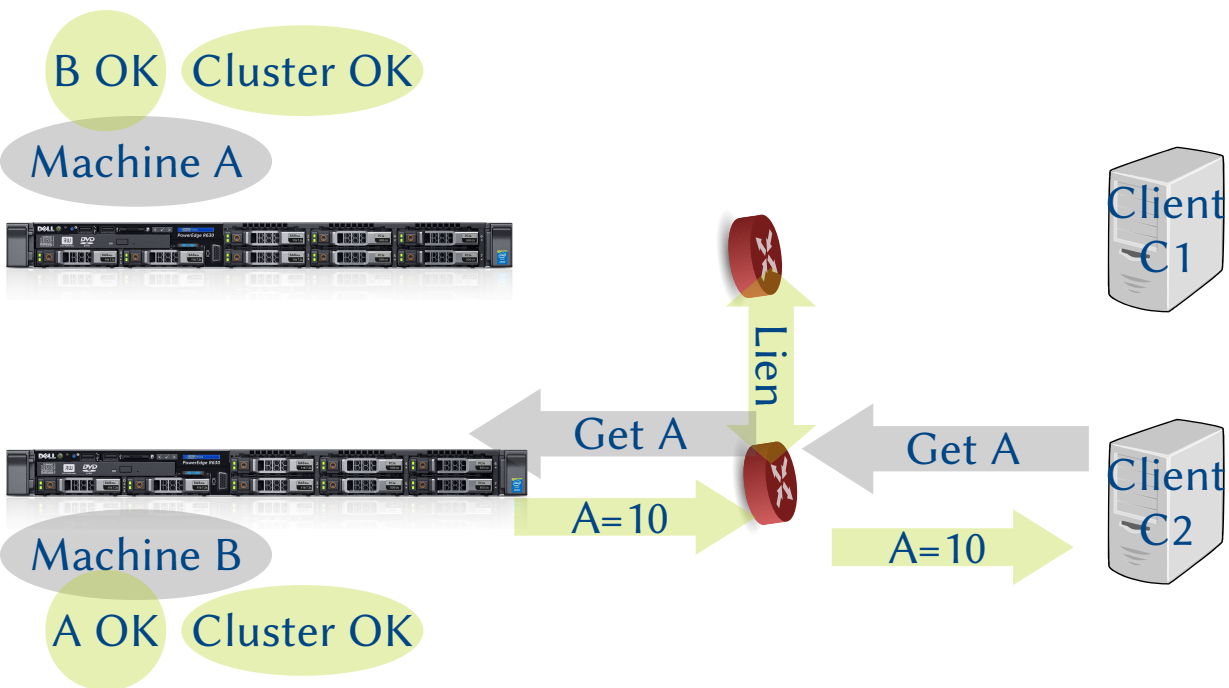
SPLIT BRAIN CLUSTER (2)

Lorsque un client modifie une valeur, les machines se synchronisent

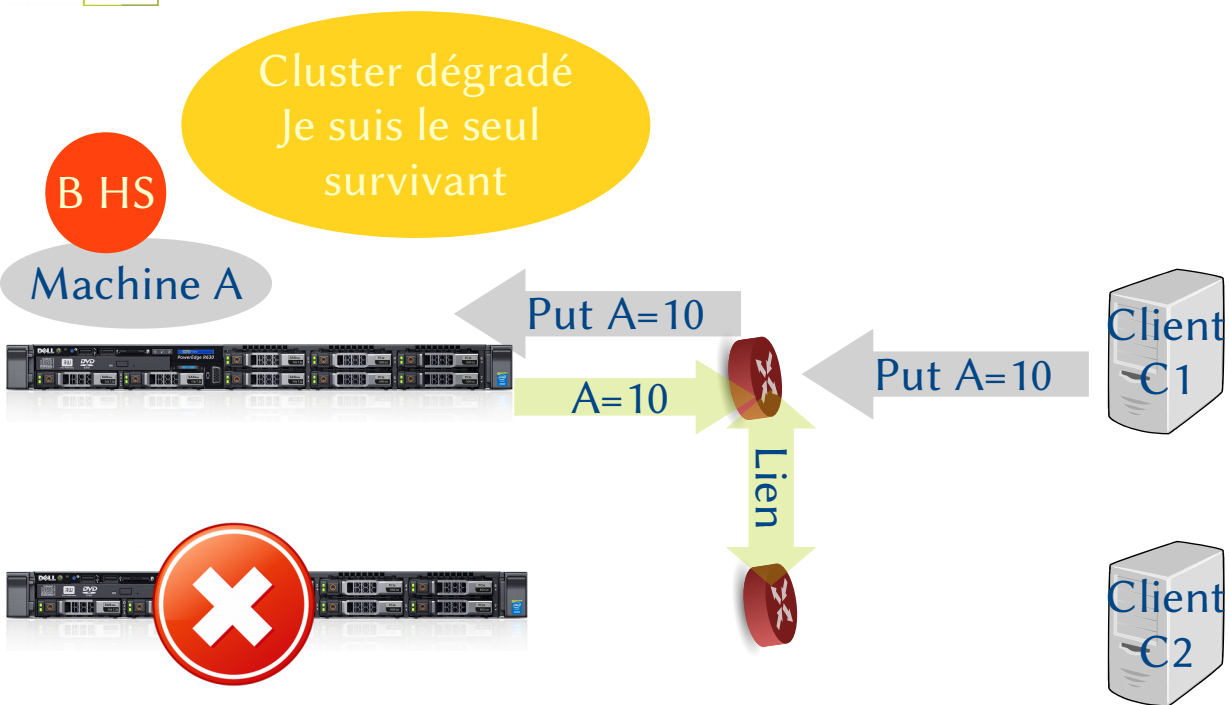


SPLIT BRAIN CLUSTER (3)

Un autre client obtiendra une valeur cohérente



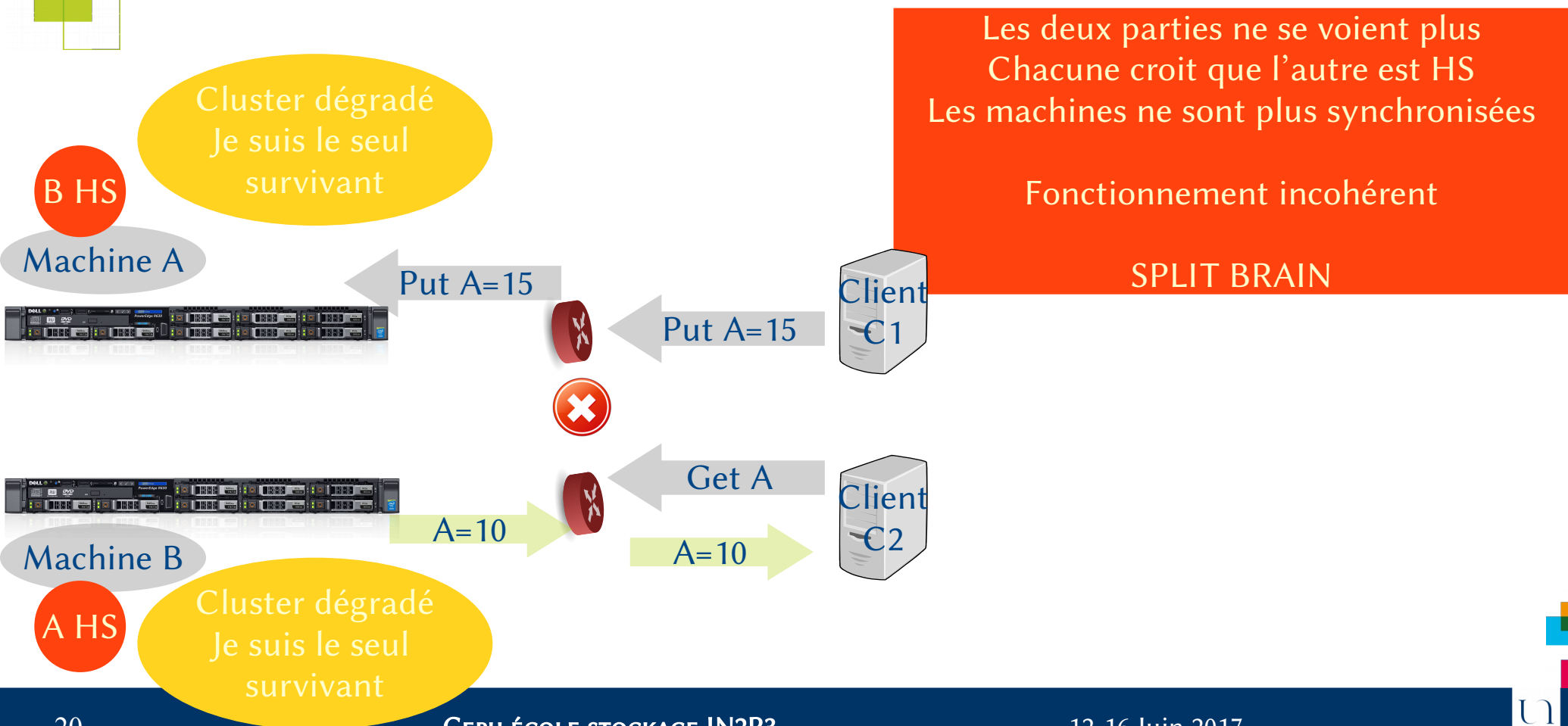
SPLIT BRAIN CLUSTER (4)



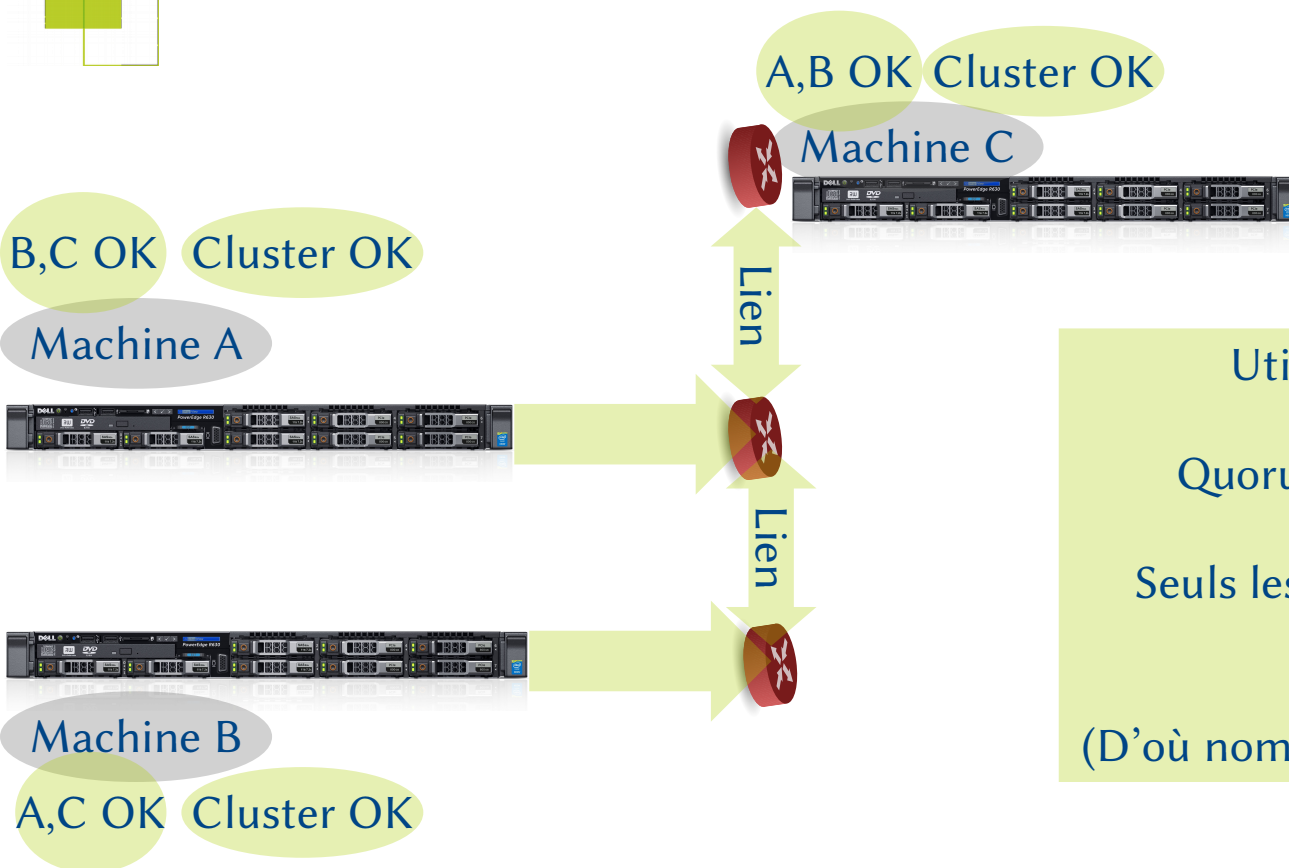
Ce système fonctionne bien en cas de panne matérielle.

Lorsque B est réparée, elle se resynchronisera.

SPLIT BRAIN CLUSTER (5)



SPLIT BRAIN CLUSTER (6)

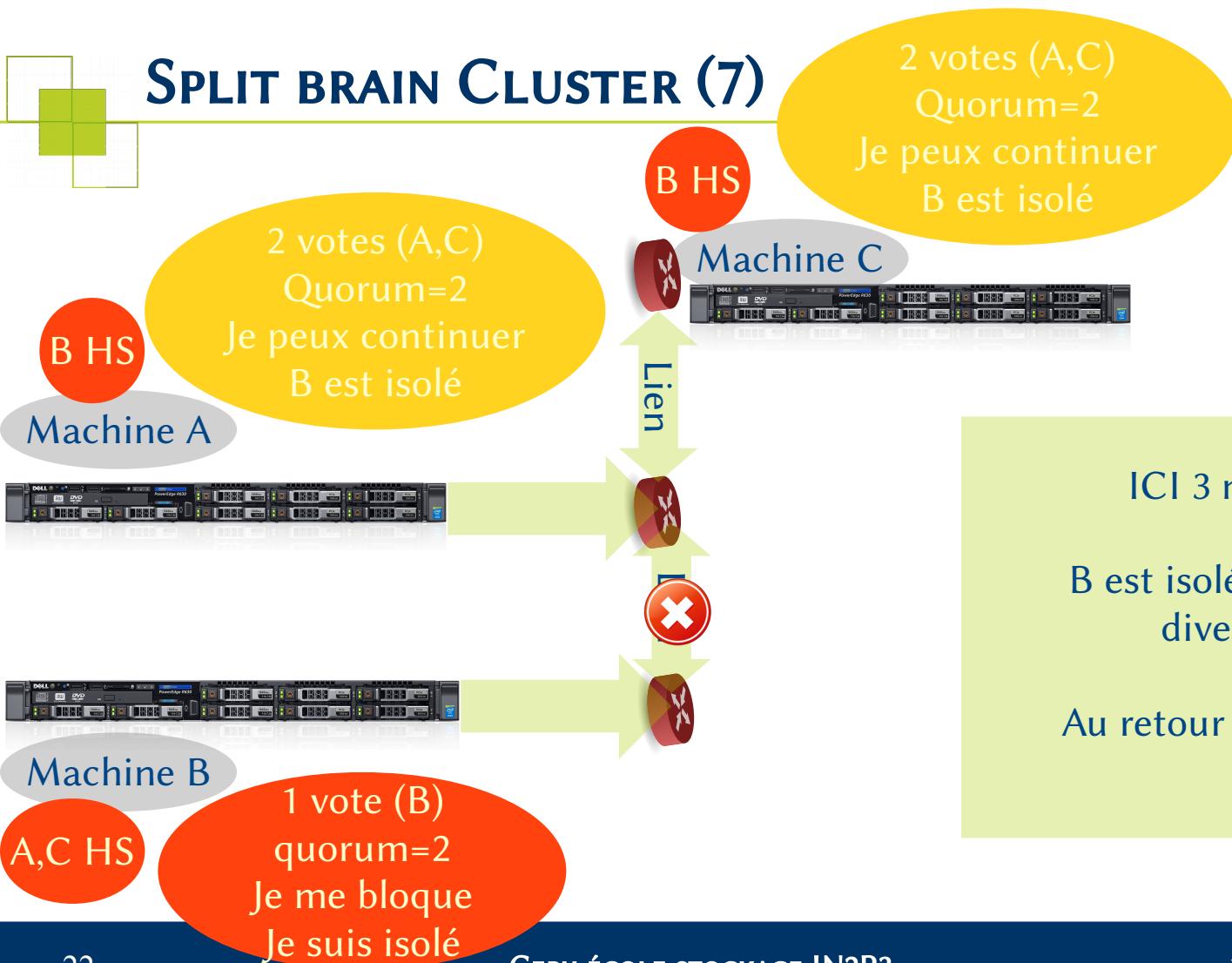


Utiliser un système de Quorum :
Basé sur des votes ;
Quorum = majorité absolue des votes.

Seuls les systèmes satisfaisant ce quorum
sont valides.

(D'où nombre impair de machines : imblocable)

SPLIT BRAIN CLUSTER (7)



ICI 3 machines, donc quorum=2

B est isolé et va se bloquer pour ne pas diverger et être incohérente.

Au retour du réseau, la machine B va se resynchroniser.

RESYNCHRONISATION

Chaque opération est versionnée
Les machines doivent être synchronisées (NTP)

Une machine qui a été isolée ou en panne
va pouvoir utiliser ces transactions pour se
resynchroniser.

```
ceph-mon-lmb-C-1:~# ceph -s
cluster 046b934b-f8c9-42b3-8e16-22e05c7379bf
health HEALTH_OK
monmap e9: 3 mons at
{a=172.20.107.85:6789/0,b=172.20.106.85:6789/0,d=172.20.108.85:6789/0}, election
epoch 2044, quorum 0,1,2 b,a,d
mdsmap e238: 1/1/1 up {0=0=up:active}
osdmap e47004: 12 osds: 12 up, 12 in
pgmap v70989076: 6600 pgs, 17 pools, 10230 GB data, 2853 kobjects
```

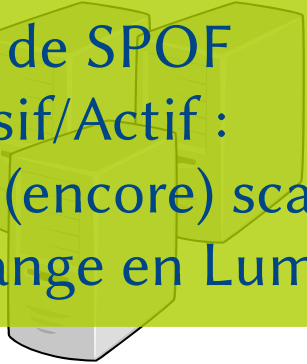
ARCHITECTURE DE CEPH



pas de SPOF
scalable

The diagram shows three server icons. Two are positioned behind a semi-transparent green box containing the text 'pas de SPOF scalable'. A third server icon is in front of the box.

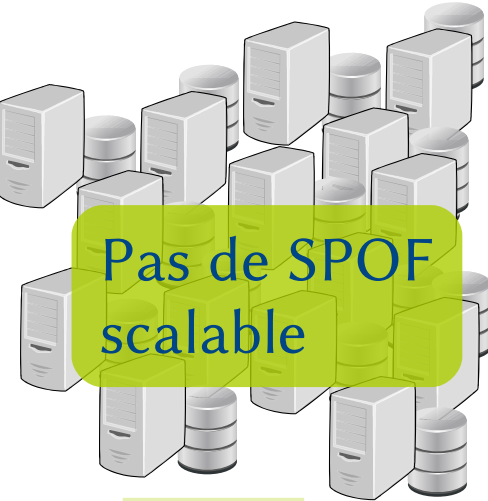
MON



Pas de SPOF
Passif/Actif :
pas (encore) scalable
(change en Luminous)

The diagram shows three server icons. Two are positioned behind a semi-transparent green box containing the text 'Pas de SPOF Passif/Actif : pas (encore) scalable (change en Luminous)'. A third server icon is in front of the box.

MDS



Pas de SPOF
scalable

The diagram shows a cluster of server and disk icons. A semi-transparent green box containing the text 'Pas de SPOF scalable' is overlaid on the cluster.

OSD

Démons en espace utilisateur Linux : de simples programmes

CRÉATION D'UN CLUSTER CEPH

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)

Va changer avec bluestore

En vrai,
pour cette semaine

On utilisera des VM
Peut être plusieurs démons/VM
(MAUVAISE PRATIQUE)
(Sync global régulier)

Pratique pour manipuler
Pas représentatif des performances

DES MACHINES « PRO ! »



Performances,
Fiabilité

Assurées !



DIMENSIONNER SON ARCHITECTURE

Réplication synchrone
(la partie la plus lente ralentit le reste)

Débit != Latence

Performance : tous les éléments comptent

Réseau :

Bonne carte et bon driver (éviter bnx2x !)

Iperf3 entre les nœuds pour vérifier le débit

Bon switch pour baisser les latences

Rdma supporté – Infiniband, omnipath ?

Utiliser des contrôleurs non bloquants

1 SSD sur 4 pour les OSD (journaux)

OSD sur 7,2k, 15k ou SSD : dépend du profil

PAS DE RAID

Beaucoup d'OSD
Être vigilant sur la configuration

≥ 1 cœur / OSD

cœurs rapides : utiles pour Erasure coding

Utile pour OSD sur SSD, (cpu facteur limitant)

Utilisé pour les processus OSD (512 Mo $\rightarrow$ 4 Go)
Et cache I/O par les kernel Linux

MACHINE TYPIQUE OSD VOLUMÉTRIQUE

R720/R730/R740xd (marché matinfo)



plus de RAID Hardware, Mode Jbod
Des SSD SLC partagés (Journaux) ?
Disques 8, 10 To 7200 Rpm SAS NL
12 OSD ou 16 OSD / machines
Capacité Brute : > 100 To

2 interfaces réseau (10 Gbit/s)
2 CPU avec cœurs, au moins 1/OSD...)
De la mémoire
Des canaux disques non bloquants

MACHINE TYPIQUE OSD RAPIDE

R630/R640 (marché matinfo)



Mode Jbod

Des SSD SLC SAS 12G ou Nvme
Ou des sas 2,5'' 1 To 15k RPM
Capacité Brute : $\sim < 10$ To

2/4 interfaces 10 ou 40 Gbit/s ou..
CPU rapide (privilégier le Mhz)
De la mémoire
Des canaux disques non bloquants

MIXER OSD CAPACITIFS ET RAPIDES ?



Faisable
Et
Intéressant.

Cf Crush.

MACHINES TYPIQUES MON/MDS



Machines virtuelles suffisantes ? ...

Pas nécessité de disques rapides ? ...

Attention en cas de cluster non nominal !

(quand le cluster n'est pas en bon état, le mon stocke de nombreuses informations)

Attention à la reconstruction ! Le mon est très sollicité.

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
Formater les volumes en XFS, les monter
Ajouter les journaux, les métas données

Créer le ceph.conf
Créer les clés de sécurité ceph
Créer 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!!), puis passer au choix 2 : Installer à la main.

TUNER SES MACHINES (CÔTÉ SERVEUR)

 **ceph** sollicite beaucoup tous les compartiments du système.

Kernel : Faire le bon choix entre kernel récent et stable : 4.4 est notre choix (en deadline, plus en CFQ) (4.9 est à qualifier) – CFQ servait à descendre la priorité des scrubs (Pre-Jewel)

4.12 : multiqueue + I/O scheduler : à évoluer pour prochain kernel LTS dans le cas des SSD : **Kyber**



Un bug du kernel peut avoir des résultats désastreux, BIEN les qualifier.

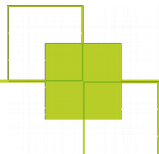
Un kernel trop ancien peut sévèrement pénaliser les performances, la gestion des disques (XFS)

XFS : formater avec les derniers utilitaires (format V5, qui implique `crc=1`, `finobt=1`)

Tuner la mémoire (favoriser inodes/dentries dans `sysctl.conf`): `vm.vfs_cache_pressure = 10`

Utilisation de rbd et cephfs implique des objets de 4 Mo ,
pour éviter la fragmentation, monter xfs avec `allocsize=4M`

Bonnes cartes réseau (*du moins, bon pilotes*) ...



TOPOLOGIE RÉSEAU

Objectifs : Avoir le meilleur débit

Mais aussi baisser la latence au maximum entre les clients et les OSD (entre eux)

3 réseaux séparés

1 pour l'administration (Gbit/s suffisant)

1 pour le réseau public (entre clients et OSD) (10 Gbit/s ou + conseillés)

1 privé (pour les communications inter OSD) (10 Gbit/s ou + conseillés)

MTU 9000 sur le privé. Également sur le public si possible.

Si Bonding : utiliser un mode actif/actif (lacp, xmit_hash_policy=layer3+4)

Un routage est possible pour les réseau publics et privés

MAIS nécessite un bon commutateur/routeur (pour ne pas pénaliser la latence)

CLIENTS

Mode	Objet	Bloc	FS
Kernel Linux	X	Oui : krbd	Oui : cephfs
User (FUSE)	X	OUI	OUI
User (KVM/Qemu)	X	Oui, via librbd	X
Via Applicatif	Oui : via R. GW (Mode S3/swift) Exemple : Nextcloud	Oui : Nfs ganasha Oui : Iscsi	?

FAIRE ÉVOLUER SON CLUSTER

En taille : simplement ajouter des OSD au cluster : Les triplets des PG sont redistribués
→ Ce qui entraîne un mouvement important des données.

Le faire de façon symétrique (par exemple, si 3 DC → 3 OSD)

En performance : Ajouter des machines plus spécialisées (SSD , disques différents, CPU...)
(et utiliser des racines différentes).

Luminous devrait permettre une typologie des OSD et (à priori?) permettrait de ne plus avoir à spécifier des racines différentes.

FAIRE ÉVOLUER SON CLUSTER

D'une version LTS vers LTS+1 (Hammer->Jewel → Luminous)

D'une version stable (LTS ou non) vers +1 (Infernalis → Jewel → Kraken → Luminous)

Avoir tous les éléments en phase extrêmement conseillé !

```
ceph-mon-lmb-A-1:~# ceph tell mon.* version
```

```
mon.b: ceph version 10.2.5 (c461ee19ecbc0c5c330aca20f7392c9a00730367)
```

```
mon.a: ceph version 10.2.5 (c461ee19ecbc0c5c330aca20f7392c9a00730367)
```

```
ceph tell osd.* version
```

```
osd.0: "version": "ceph version 10.2.5 (c461ee19ecbc0c5c330aca20f7392c9a00730367)"
```

```
osd.1: "version": "ceph version 10.2.5 (c461ee19ecbc0c5c330aca20f7392c9a00730367)"
```

Toujours tenir compte de sa topologie et son domaine de panne (par ex : DC par DC)

Upgrader les paquetages

Upgrade les MON. Attendre stabilisation.

Upgrader les OSD. Attendre stabilisation.

Upgrader les MDS. Attendre stabilisation.

Lire les releases notes !

CEPH.CONF

Une partie globale

Une partie spécifique pour les mon, mds, osd... (serveurs)

Une partie spécifique pour les clients

```
[global]
fsid = 18857a8a-f828-462a-9214-5db66def3806
auth_cluster_required = cephx
auth_service_required = cephx
auth_client_required = cephx

public network = 172.20.106.0/24,172.20.107.0/24, 172.20.108.0/24
cluster network = 172.20.112.0/24, 172.20.113.0/24, 172.20.114.0/24

[mon]
mon debug dump transactions = false
mon compact on start = true
mon initial members = a,b,c

[mon.a]
host = ceph-mon-lmb-D3-1
mon_addr = 172.20.107.89:6789

[mon.b]
```

CEPH.CONF

[osd]

```
osd data = /CEPH/D3.$id
osd_journal = /var/lib/ceph/osd/ceph-$id/journal-$id
osd_journal size = 5000
journal aio = true
```

;; Un peu d'optimisation d'I/O

```
osd op threads = 8
osd disk threads = 2
filestore op threads = 4
```

```
filestore journal writeahead = true ; default pour XFS, mais pour LXC
filestore flusher = false ;
filestore max sync interval = 15;
```

```
osd mkfs type = xfs
```

[osd.0]

```
host = ceph-osd-lmb-D3-1
```

[osd.1]

```
host = ceph-osd-cha-D3-1
```

[osd.2]

CEPH.CONF

[client]

rd cache = true

rd cache size = 1024 Mib

admin socket = /var/run/ceph/\$cluster-\$type.\$id.\$pid.\$cctid.asok

debug_auth = 0/0

debug_buffer = 0/0

debug_context = 0/0

debug_crypto = 0/0

debug_finisher = 0/0

debug_ms = 0/0

debug_objectcacher = 0/0

debug_objecter = 0/0

debug_rados = 0/0

debug_rbd = 0/0

debug_striper = 0/0

debug_tp = 0/0

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

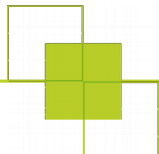
64 = nombre de PG du pool

Replicated = façon dont les données sont stockées

Beaucoup de paramètres par défaut

Mode répliqué : chaque OSD (3 par défaut) ont une copie complète de la donnée

Mode Erasure : comme Raid 5 (en simplifiant)



REPLIQUÉ VS ERASURE

Codage différent de l'information.

Repliqué X 3 = 3 OSD détiennent une copie intégrale du fichier

Erasure 2+1 = 2 OSD détiennent 50 % du fichier , 1 OSD détient une transformée mathématique qui peut reconstruire une partie manquante.

Exemple : Parité en Raid5 (Fonction XOR sur code binaire)

Repliqué X 3 = Très Sûr
Très rapide
Très consommateur d'espace

Erasure Coding 2+1 moins sûr
Plus lent en écriture
Moins consommateur d'espace

STOCKER UN OBJET

Ceci est un objet
(utile)



STOCKER UN OBJET DANS RADOS

Où ?
Comment ?

Un cluster CEPH présente des pools de stockage.
Ils sont créés au besoin. (1 de base).

Chaque pool dispose de ses caractéristiques,
ses droits d'accès,
sa règle de placement d'objets (CRUSH).



Stockage dans
pool objets_utiles

Pools

GLOBAL:

SIZE	AVAIL	RAW USED	%RAW USED
43483G	16986G	26496G	60.94

POOLS:

NAME	ID	USED	%USED	MAX AVAIL	OBJECTS
mirrors	3	3341G	7.68	7646G	863776
cloud-perso	4	1296G	2.98	5097G	465044
os-patrons	6	209G	0.48	5097G	67193
os-dsin-prv	7	907G	2.09	5097G	262322
logs	9	2583G	5.94	5097G	666408
data-dsin	11	432G	1.00	5097G	111371
data-tiers	12	180G	0.41	5097G	46168
objets-utiles	13	21234M	0.05	5097G	5379
esxi-backup	14	8973M	0.02	5097G	2916

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
```

CRUSH

Où et comment placer les objets
dans le pool ?

Le faire simplement et rapidement ?

Qui fait le choix ?



RÈGLES CRUSH

Description de la hiérarchie du matériel

```
root default
  datacenter lmb
    room lombarderie-ltp
      host abouriou
        vm ceph-osd-lmb-C-1-1
          osd.0 up 1
        vm ceph-osd-lmb-C-1-2
          osd.1 up 1
      host magon
        vm ceph-osd-lmb-C-3-1
          osd.8 up 1
        vm ceph-osd-lmb-C-3-2
          osd.9 up 1
  datacenter loi
  [...]
```

Règles de placement des objets

```
rule rbd {
  ruleset 2
  type replicated
  min_size 1
  max_size 10
  step take default
  step chooseleaf firstn 0 type datacenter
  step emit
}
```

Tout est stocké globalement dans le cluster
Tout le monde (y compris le client) dispose de la dernière version.

GROUPES DE PLACEMENT

Un pool est découpé en groupes de placement (PG).

1 PG contient la liste des OSD contenant les copies de l'objet.

Carte des PG change lors de la modification de la topologie (algorithme).

Le client PLACE ses objets.

```
ceph osd pool get objets-utiles pg_num  
pg_num: 1024
```

Le pool objets-utiles dispose de 1024 PG

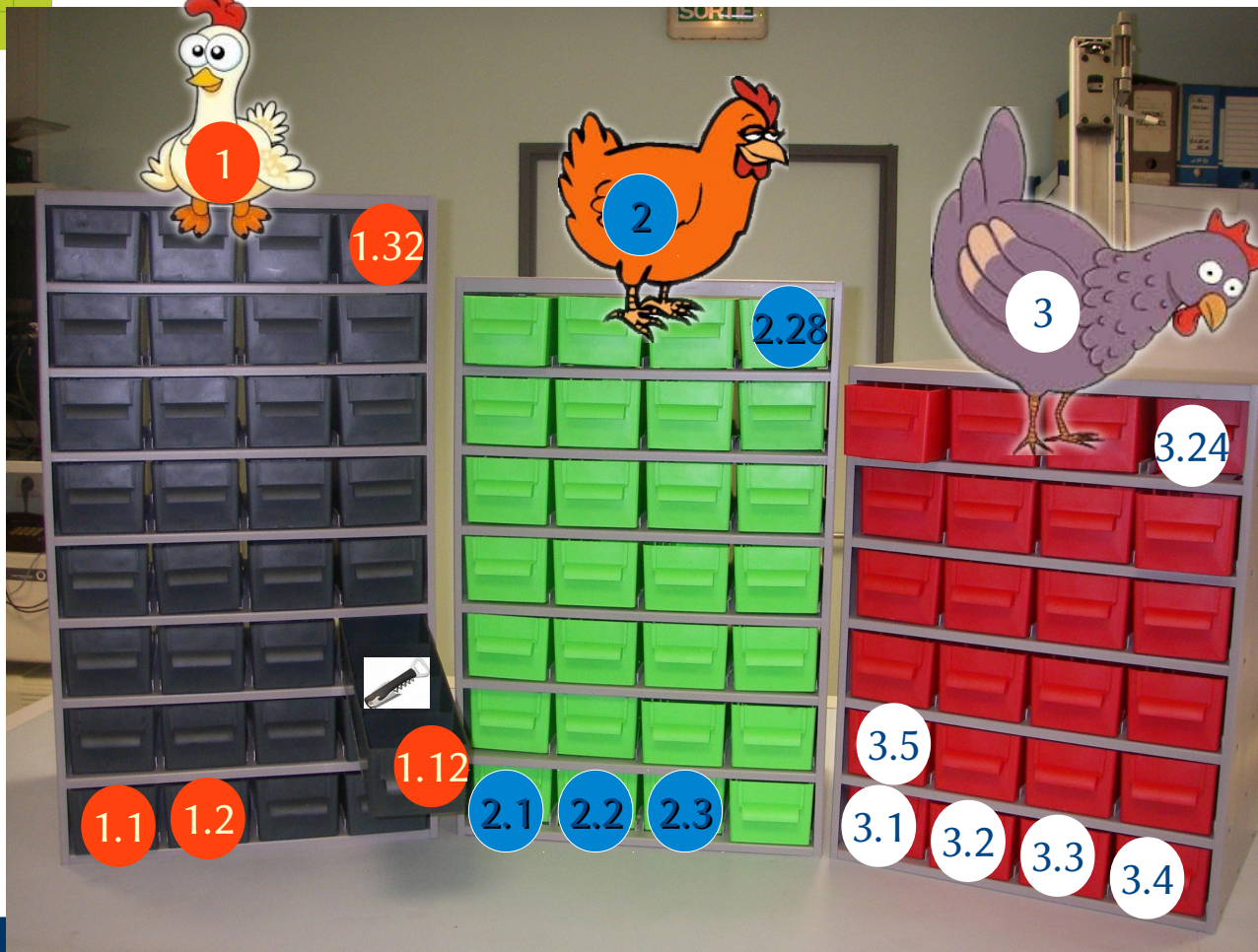
```
ceph osd pool get objets-utiles size  
Size: 3
```

Le pool os-patterns est configuré pour dupliquer 3 fois l'objet

```
ceph pg dump ( 13.3fb → [3,4,8] )
```

Le PG 3fb du pool 13 (objets-utiles) utilisera les serveurs de stockage (OSD) 3, 4 et 8

Pools & PG



Chaque pool dispose de caractéristiques différentes :

Nombre de PG (32,28,24)

Taille

Placement des données...

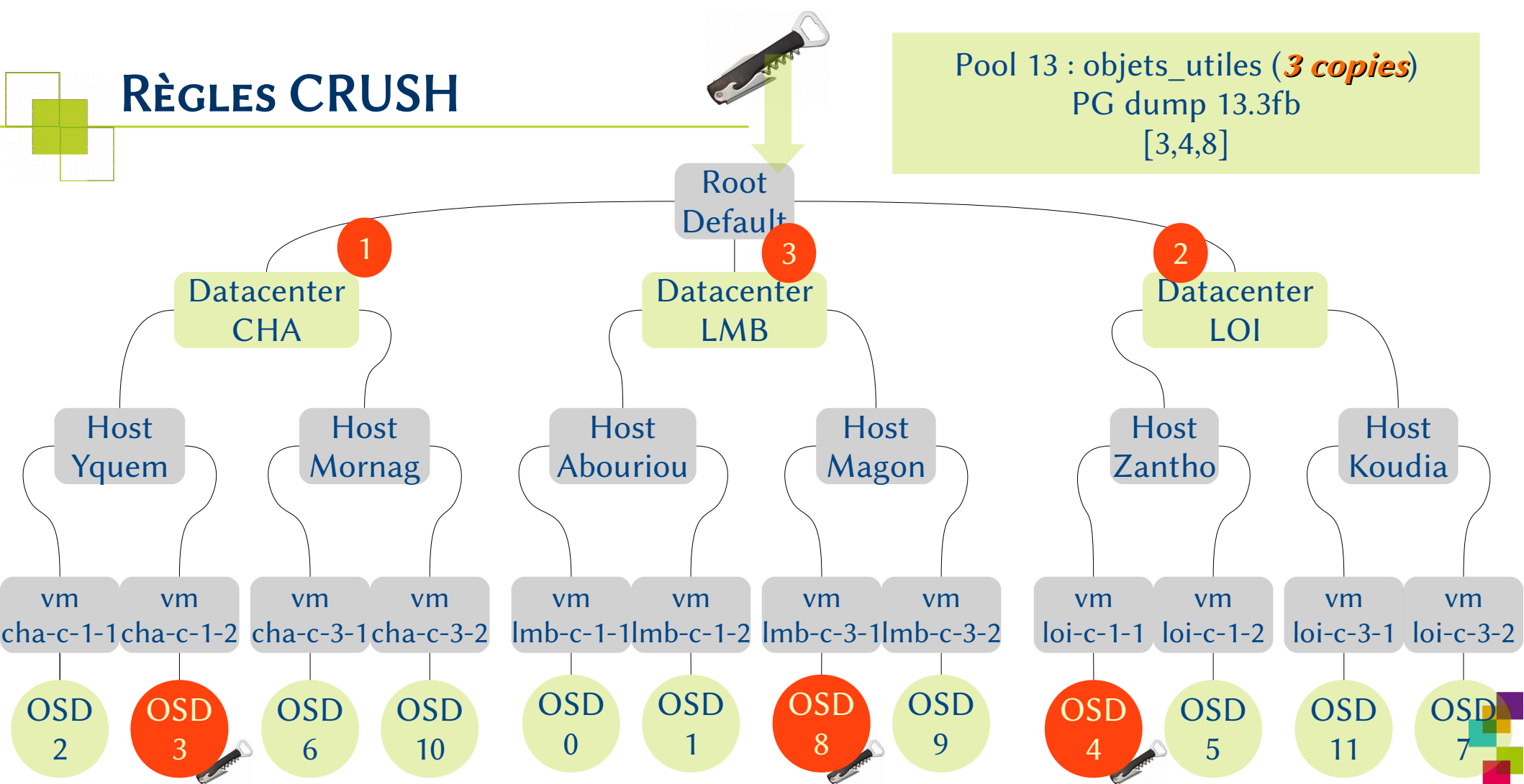
Un objet est placé dans
Le PG d'un pool

Chaque PG va contenir de
nombreux objets

Si beaucoup d'objets à stocker :
avoir plus de PG dans un pool

permet d'avoir moins d'objets par
PG

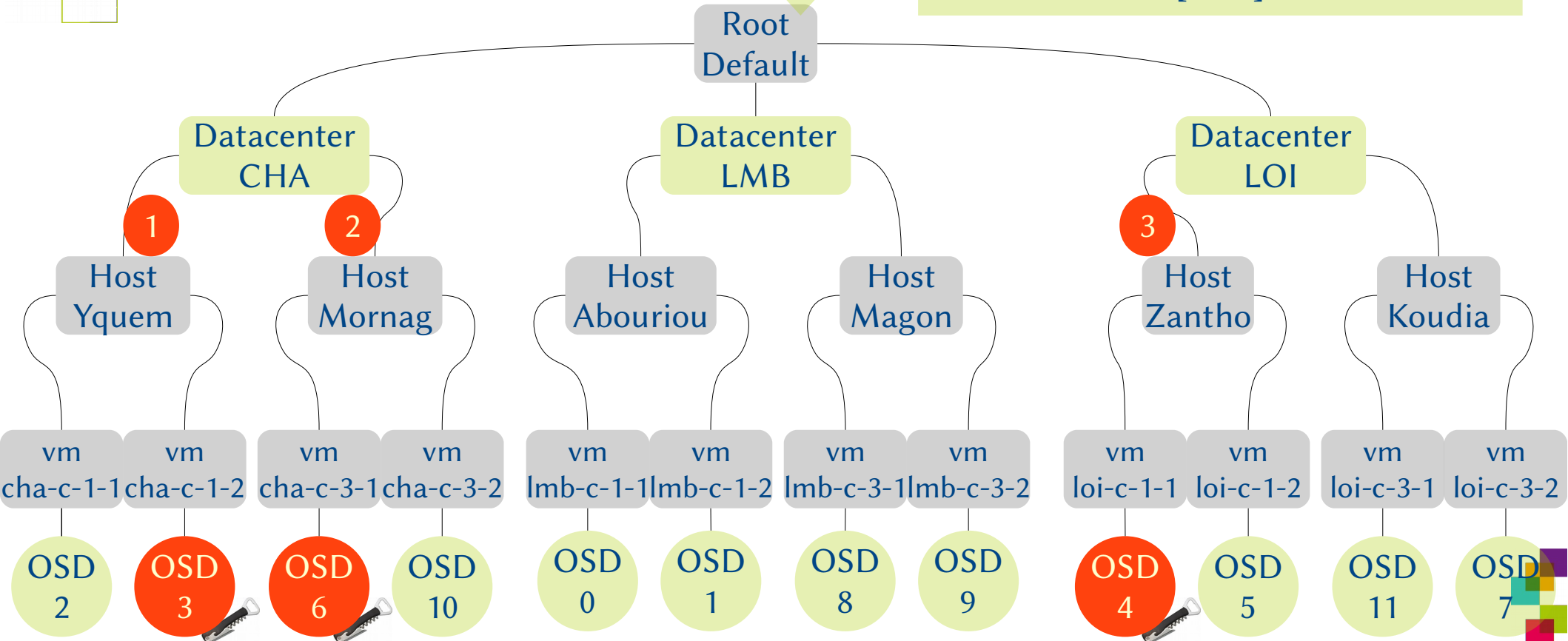
RÈGLES CRUSH



RÈGLES CRUSH



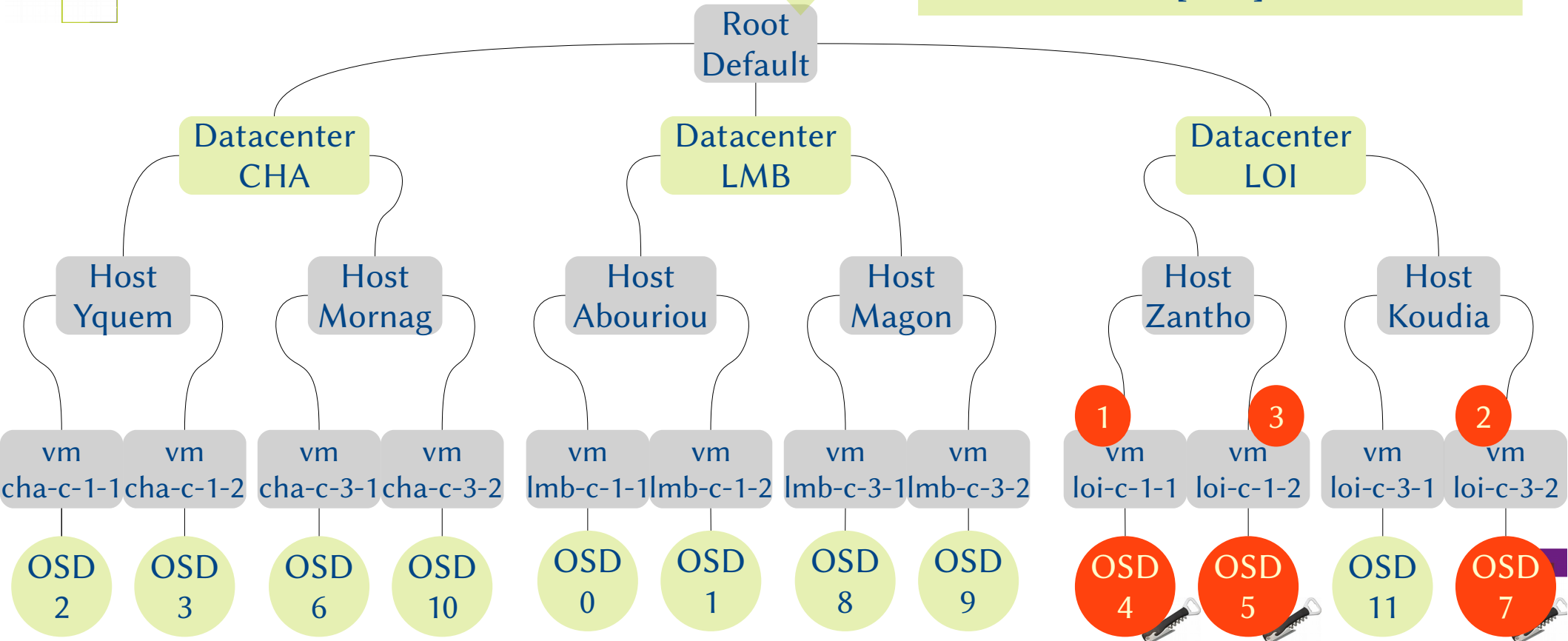
Si le choix se fait sur host
PG dump 13.3fb
[3,6,4]



RÈGLES CRUSH



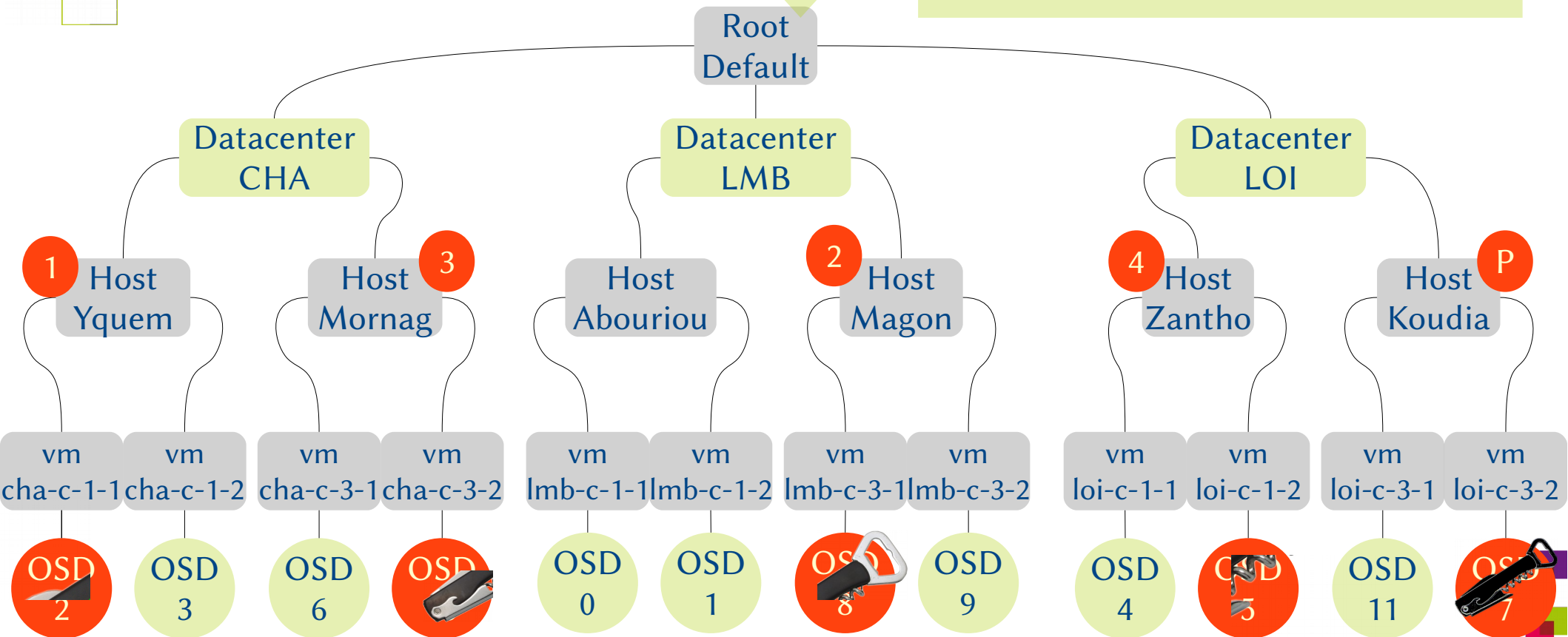
Si le choix se fait sur vm
PG dump 13.3fb
[4,7,5]



ERASURE CODING



Ici, 4+1 (objet découpé en 4)
+ d'axes, **moins de place occupé**
Placement au niveau des hosts



SUR LES OSD

```
ceph osd map test-ANF ObjetTireBouchon
osdmap e15976 pool 'test-ANF' (18) object 'ObjetTireBouchon'
-> pg 18.4384205 (18.5) -> up ([2,3,4], p2) acting ([2,3,4], p2)
```

```
ceph osd tree
```

```
-1 31.85030 root default
```

```
-5 10.61696   datacenter lmb
```

```
-13 10.61667   datacenter loi
```

```
-12 10.61667   room loire-presidence
```

```
-11 10.61667   host koudia
```

```
-10 10.61667   vm ceph-osd-loi-B-1
```

```
 2 3.53899      osd.2          up 1.00000      1.00000
```

```
 5 3.53899      osd.5          up 1.00000      1.00000
```

SUR LES OSD

```
root@koudia:/var/lib/lxc/10-ceph-osd-loi-B-1/rootfs/CEPH/B.2/current/18.5_head# ls -al
total 16444
drwxr-xr-x  2 64045 64045    86 juin  12 11:16 .
drwxr-xr-x 554 64045 64045 32768 juin  12 11:15 ..
-rw-r--r--  1 64045 64045    0 juin  12 11:15 __head_00000005__12
-rw-r--r--  1 64045 64045 16777216 juin  12 11:16 ObjetTireBouchon__head_04384205__12
```

```
root@koudia:/var/lib/lxc/10-ceph-osd-loi-B-1/rootfs/CEPH/B.2/current/18.5_head# ls -al
total 16444
drwxr-xr-x  2 64045 64045    86 juin  12 11:16 .
drwxr-xr-x 554 64045 64045 32768 juin  12 11:15 ..
-rw-r--r--  1 64045 64045    0 juin  12 11:15 __head_00000005__12
-rw-r--r--  1 64045 64045 16777216 juin  12 11:16 ObjetTireBouchon__head_04384205__12
```

SUR LES OSD

```
xattr -l ObjetTireBouchon__head_04384205__12
user.cephos.spill_out:
0000 30 00 0.

user.ceph._:
0000 0F 08 F5 00 00 00 04 03 31 00 00 00 00 00 00 00 .....1.....
0010 10 00 00 00 00 4F 62 6A 65 74 54 69 72 65 42 6F 75 ....ObjetTireBou

user.ceph.snapset:
0000 02 02 19 00 00 00 00 00 00 00 00 00 00 00 01 00 .....
0010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....

user.ceph._@1:
0000 FF .

root@koudia:/var/lib/lxc/10-ceph-osd-loi-B-1/rootfs/CEPH/B.2/current/18.5_head# ls -al
total 16444
drwxr-xr-x 2 64045 64045 86 juin 12 11:16 .
drwxr-xr-x 554 64045 64045 32768 juin 12 11:15 ..
-rw-r--r-- 1 64045 64045 0 juin 12 11:15 __head_00000005__12
-rw-r--r-- 1 64045 64045 16777216 juin 12 11:16 ObjetTireBouchon__head_04384205__12
```



CRUSH (L'OUTIL)

Placement des données non uniformes

Straw2
Tunables

Nouveautés de Luminous

DISTRIBUTION DES DONNÉES NON UNIFORME

```
ceph-mon-lmb-A-1:~# ceph df
```

```
GLOBAL:
```

SIZE	AVAIL	RAW USED	%RAW USED
66469G	24553G	41915G	63.06

```
POOLS:
```

NAME	ID	USED	%USED	MAX AVAIL	OBJECTS
data	0	0	0	6360G	0
metadata	1	0	0	6360G	0
rbd	2	0	0	6360G	1
data-kvm-tiers	33	2107G	24.89	6360G	540584
ECisa2p1	50	22024G	63.39	12721G	5648511
HOTECisa2p1	51	818G	11.41	6360G	209939

```
ceph report > ceph_report.json
```

DISTRIBUTION DES DONNÉES NON UNIFORME

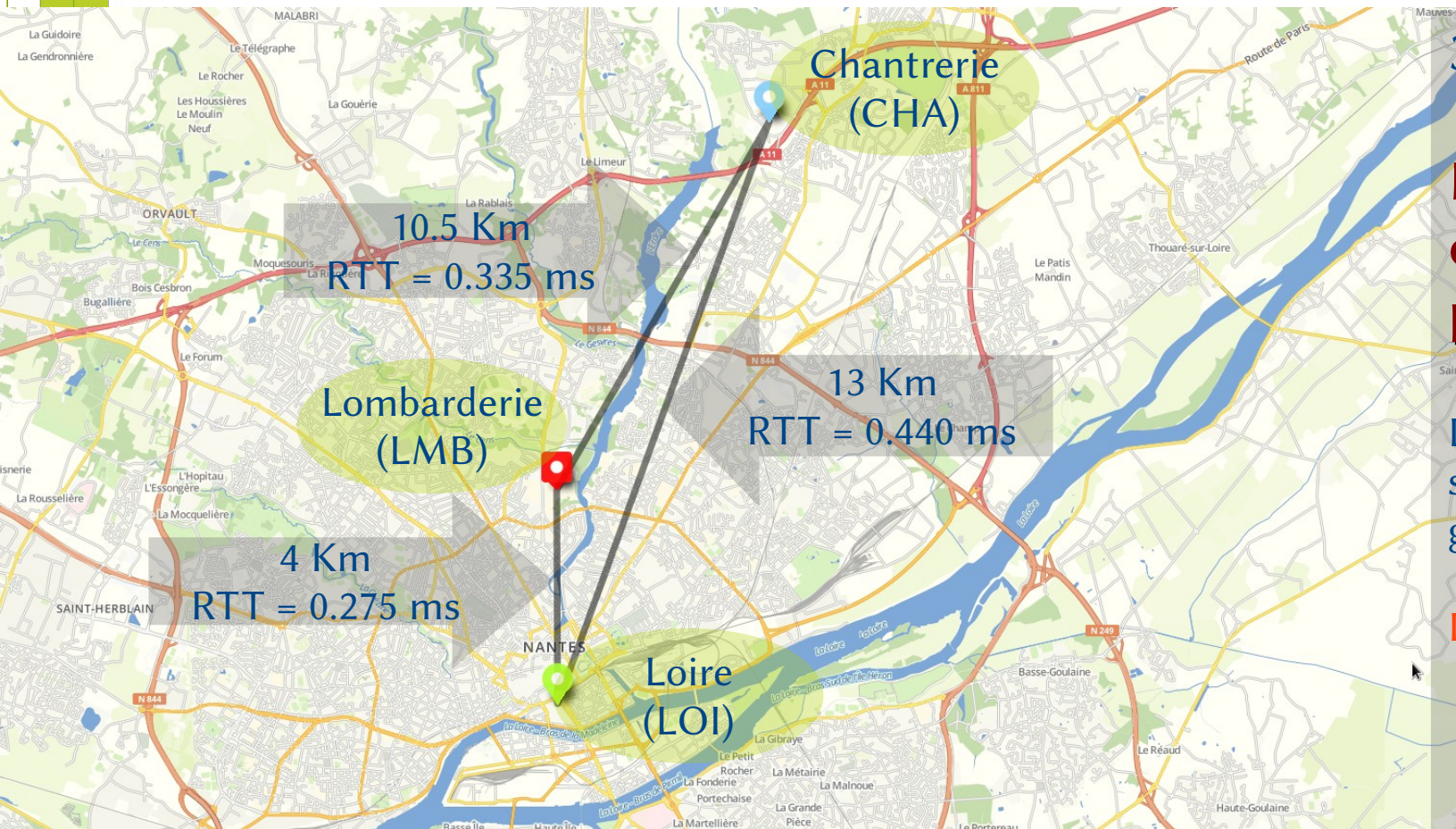
```
crush analyze --crushmap ceph_report.json --pool 3
```

	~id~	~weight~	~PGs~	~over/under filled %~
~name~				
cloud3-1363	-6	419424	1084	7.90
cloud3-1364	-7	427290	1103	7.77
cloud3-1361	-4	424668	1061	4.31
cloud3-1362	-5	419424	1042	3.72
cloud3-1359	-2	419424	1031	2.62
cloud3-1360	-3	419424	993	-1.16
cloud3-1396	-8	644866	1520	-1.59
cloud3-1456	-11	665842	1532	-3.94
cloud3-1397	-9	644866	1469	-4.90
cloud3-1398	-10	644866	1453	-5.93

Worst case scenario if a host fails:

	~over filled %~
~type~	
device	30.15
host	10.53
root	0.00

SÉCURITÉ DES DONNÉES.



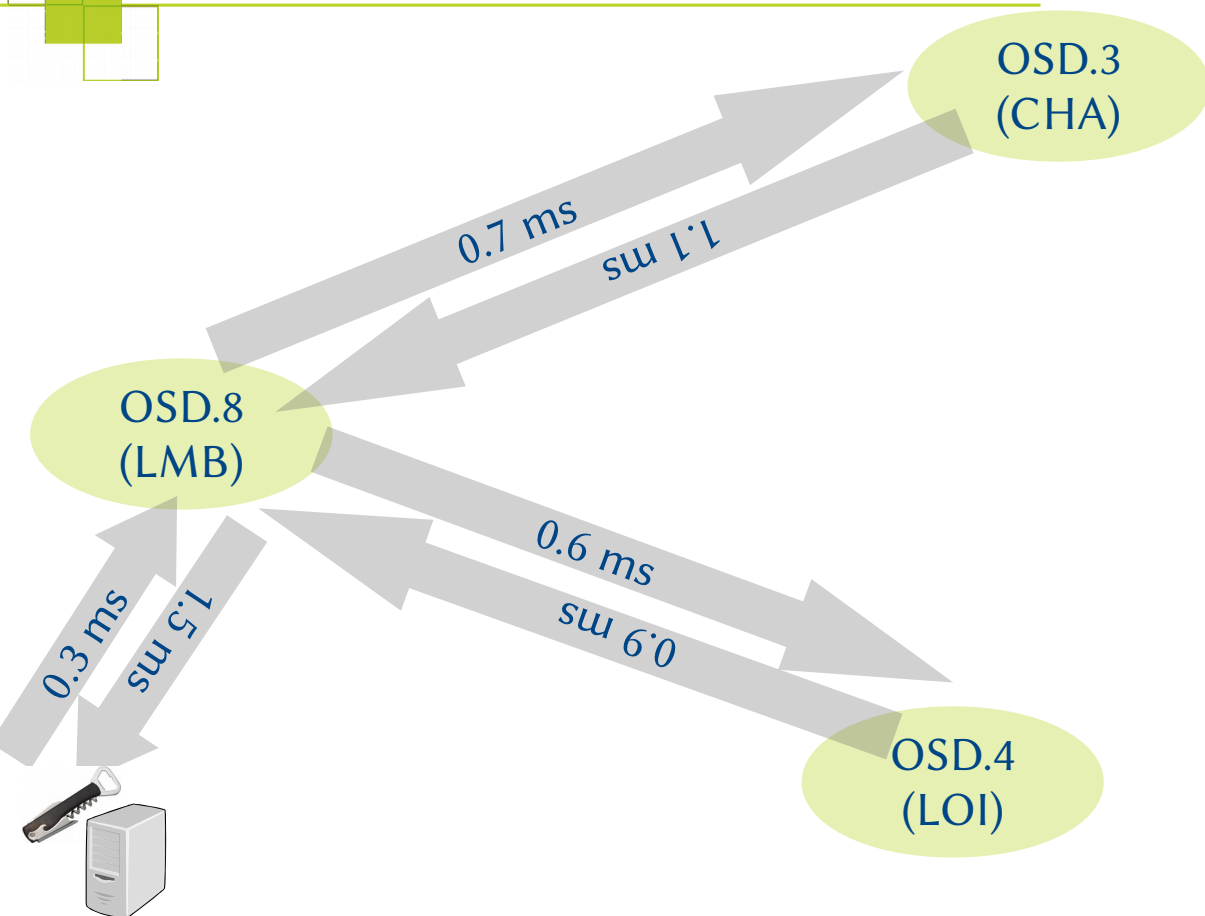
3 salles machines

**Notion de
domaine de
panne**

Données stockées
sur 3 points
géographiques

Latence vs Performance

LATENCE



>1,5 ms par transaction

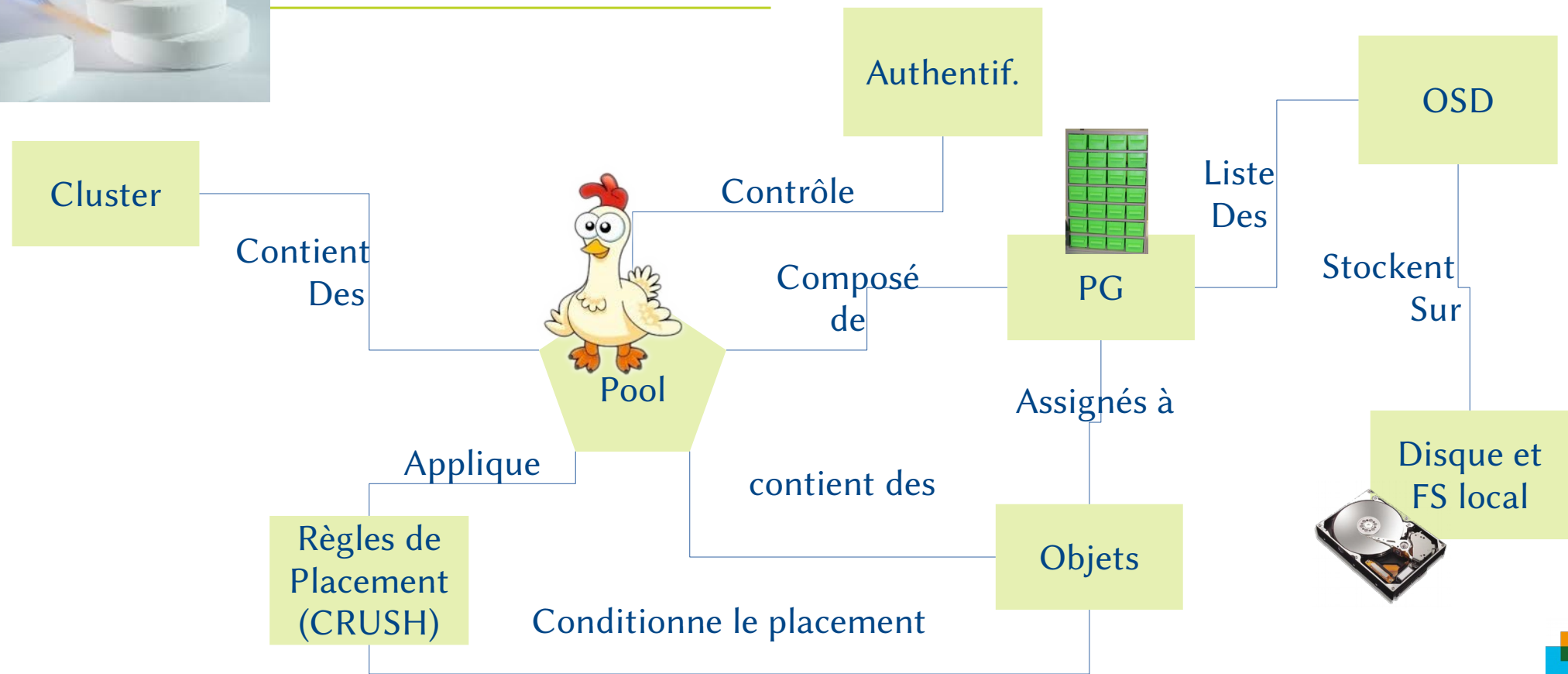
~ 600 iops

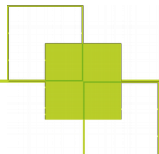
Mais par thread.

Plus de threads = plus d'I/O mécaniquement



RÉCAPITULATIF





INTERFACE OBJET (LIBRADOS, RADOSGW)

Nécessite Mon, OSD

Rados, librados = mode natif.

Objet stocké de façon monolithique ou découpé en blocs d'une certaine taille.

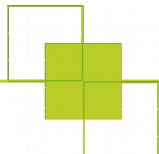
Utilitaire rados pour interagir directement (cf TP)

Clients : outils cloud, applications WWW

RESTful (web) service: Swift, S3

implémenté via l'outil RadosGW (couche de translation utilisant un serveur WWW embarqué)

Une grande partie de S3 est implanté.



INTERFACE BLOC RBD (RADOS BLOCK DEVICE)

Nécessite Mon, OSD

Émule des périphériques de type bloc. San Virtuel.

Gère les snapshots, copy on write, import, export, miroir asynchrone, thin provisioning

Performant, peut remplacer des baies SAN.

Clients :

Krbd, Fuse, Librbd.

Pas de client Windows ou Vmware.

Support librbd pour KVM → Windows peut quand même en bénéficier.

Support iSCSI via un gateway (ne faisant pas partie de CEPH) : il existe plusieurs solutions.

Comme n'importe quel périphérique bloc, peut aussi être réexporté par samba ou NFS.

VOLUMES RBD ?

Volume segmenté en blocs de 4 Mo,
alloués à la demande
assignés à un PG (Placement Group)

PG distribués selon règles (CRUSH)
de placement du pool

ICI: règles de placement simples

Root =datacenter
Size = 3

Pool 6, volume RBD
Os-templates/debian8

“map” pré-calculée,
Distribuée aux clients
Pas de serveur de metadonnées
Accès simple & direct

VOLUMES RBD ?

On continue à stocker des objets dans un pool.

Les objets = blocs de 4 Mo de données.

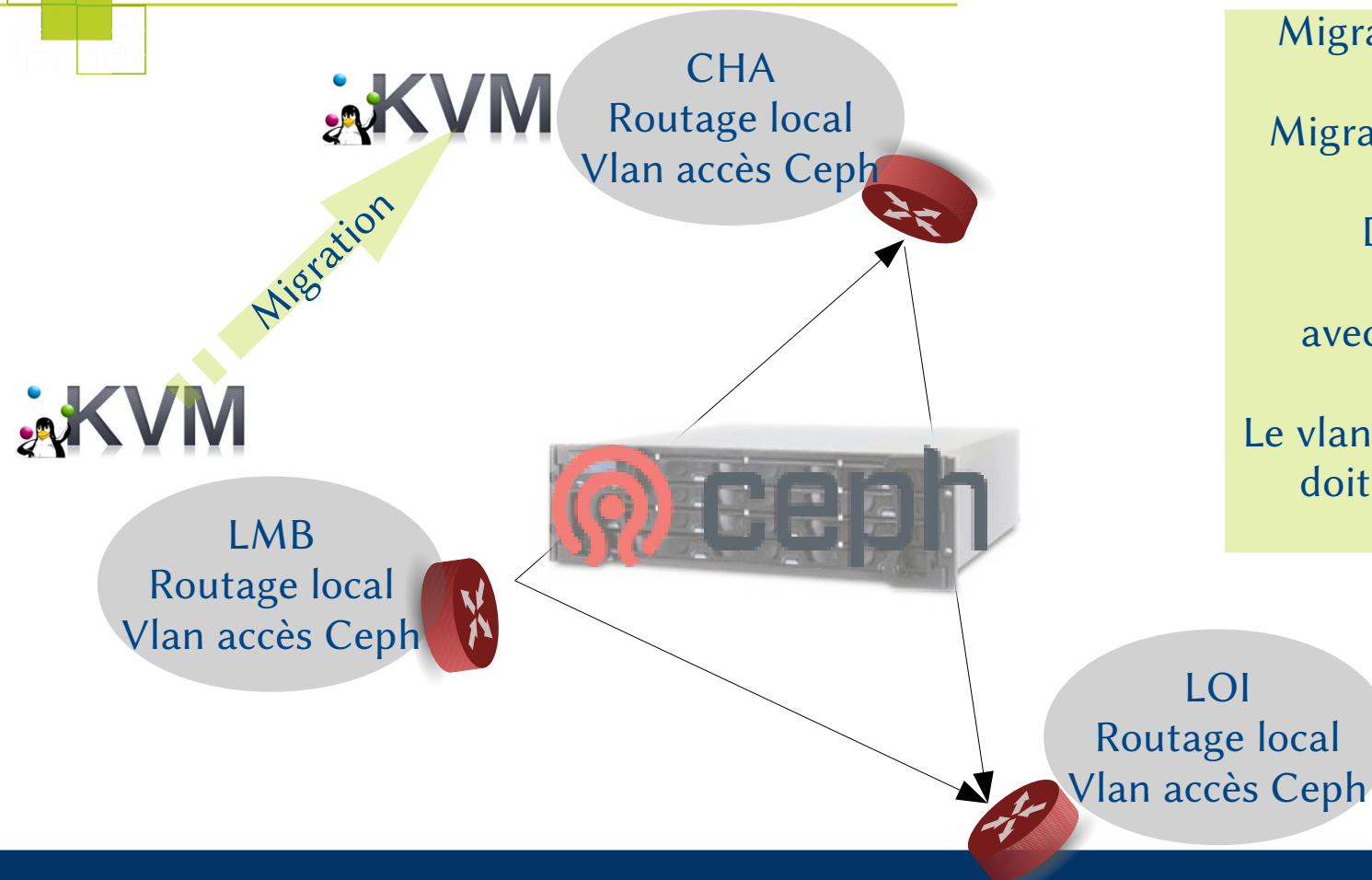
(1000 secteurs de type AF, 8000 de type standard)

Allocation de l'objet lors de sa 1ere écriture

Possible de créer autant de volumes que nécessaire

Attention à la sur-réservation !

MIGRATION À CHAUD



Migration à chaud d'une VM:

Migration des seuls processus

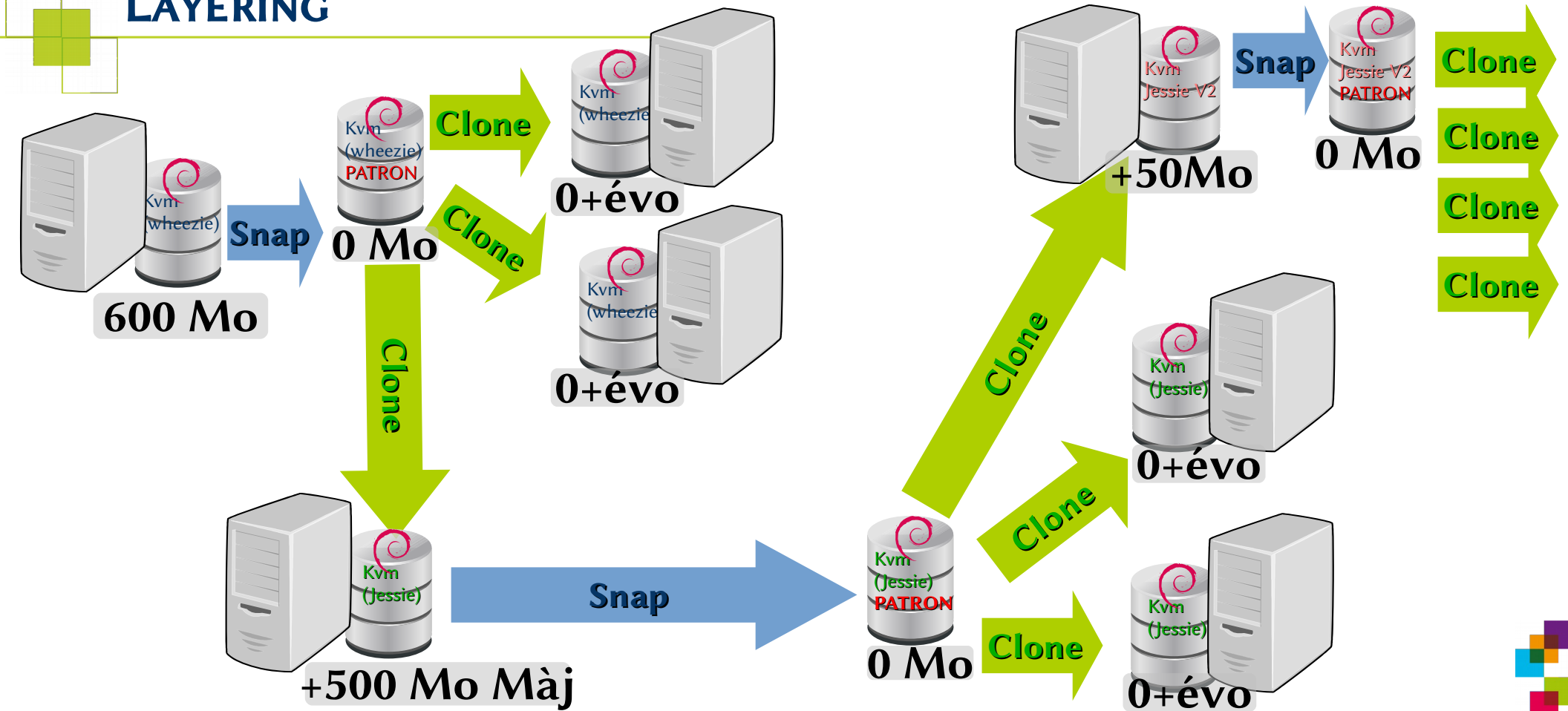
Données immobiles,
accessibles
avec la même performance

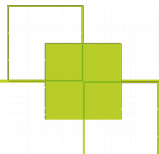
Le vlan d'accès aux clusters Ceph
doit être disponible sur les
serveurs hôtes

Il est possible de faire des snapshots globaux
(au niveau RADOS / pool)
Où au niveau de l'Image RBD (attention, l'un ou l'autre)

Dans RBD les snapshots peuvent être clonés (y compris dans un autre pool)
Et deviennent des nouvelles images à part entière, comme un calque
Docker procède aussi ainsi (overlayFS)

LAYERING





FEATURES

Par défaut Jewel active certaines features pour les volumes RBD :
deep-flatten,fast-diff,object-map,exclusive-lock

Cela les rend incompatibles avec krbd (sauf noyau très récent)
Mais ne pas les désactiver aveuglément, car compatibles libRBD

Il est possible de mettre à jour librbd sans mettre à jour KVM.

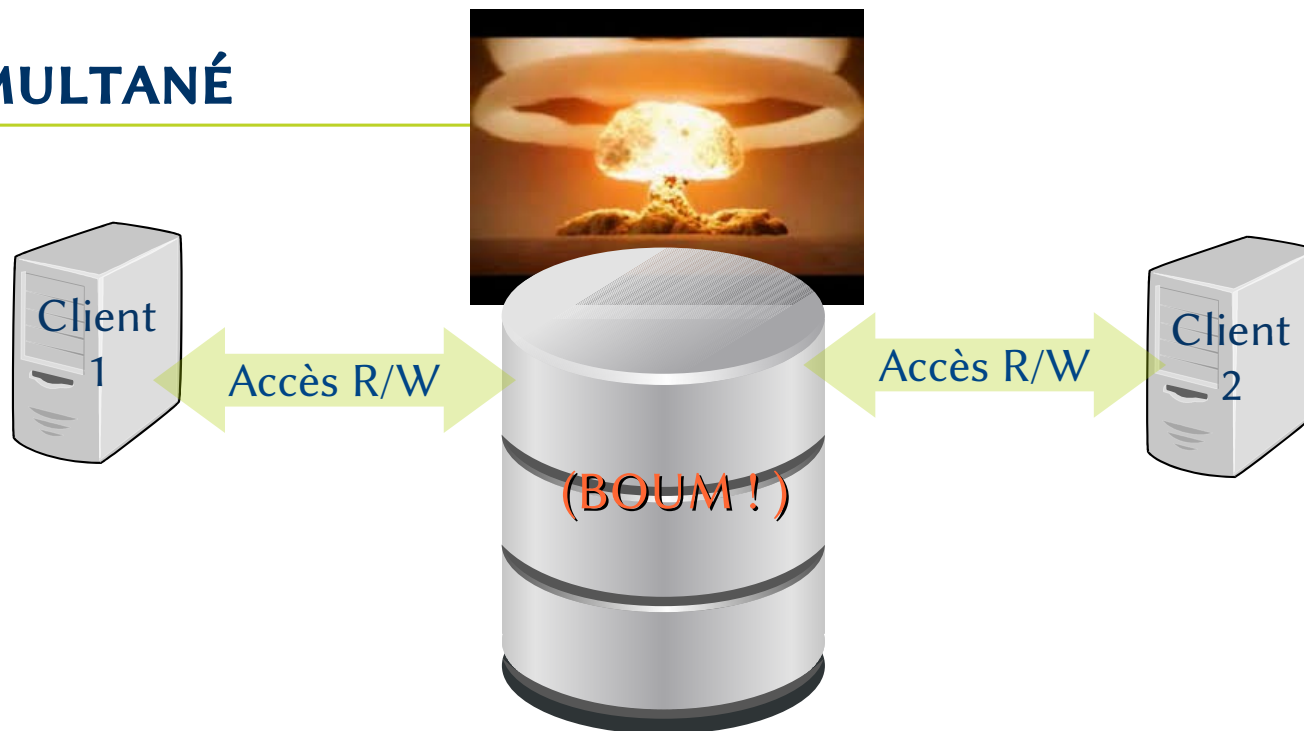
Gains en fonctionnalité ET performances (on doit redémarrer ou migrer la VM).

Exclusive-lock : Empêche une image d'être montée en simultané sur 2 VM

Object-map : Optimise la recherche des objets entre « Layers » (et suppression volumes)

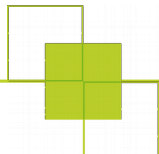
Fast-diff & Deep Flatten permettent d'être plus efficace dans les exports Rados.

ACCÈS SIMULTANÉ



Activer Exclusive-lock : Empêche une image d'être montée en simultanée sur 2 VM
Ou utiliser un FS cluster sur l'image : Ocfs2, Gfs2...

Ou utiliser Cephfs (ou un appli utilisant le mode objet)



INTERFACE FILE SYSTEM DISTRIBUÉ: CEPHFS

Nécessite Mon, OSD, *MDS*

Très longtemps considéré comme insuffisant pour la production.
Déclaré stable depuis Jewel.
Outil de check/réparation disponible.

Mais pas forcément scalable : Support multi-mds déconseillé pour Jewel (mode actif passif : cf TP)
Pour Luminous, Actif/Actif stable

Et pas forcément performant : benchmarks ?

Fichiers découpés en blocs de 4 Mo (comme RBD).

Support Linux kernel ou fuse.

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
```

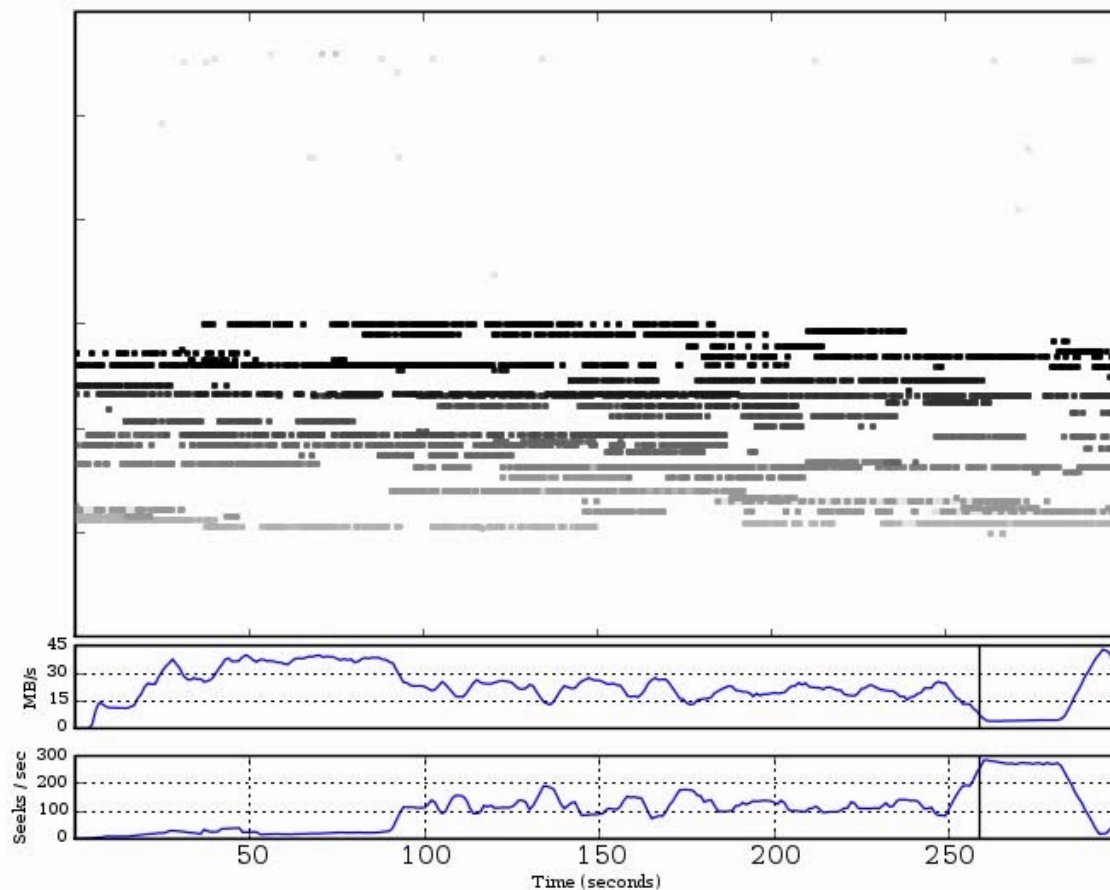
```
root@debian:/# time cp -ax / /mnt/cephfs/
```

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

Attention : sur Jewel
Luminous devrait être meilleur.



ALLOCATIONS SUR UN DISQUE



Exemple des allocations d'un système de fichiers EXT4 au fil du temps

(Benchmark de 5 minutes).

COMMUNICATION ENTRE FS ET SYSTÈME BLOC SOUS JACENT

Sur la vidéo, les blocs alloués apparaissent en noir.

Ceph répertorie les nouveaux blocs alloués et crée des objets RBD à chaque nouveau besoin
(blocs de 4Mo émulant des secteurs disques)

Allocation à la volée, consommation à l'utilisation.

Il est possible de provisionner des volumes de 200 To sans les avoir physiquement.

→ Quand le système de fichiers efface un fichier, il va juste le déréférencer de sa table d'inodes (ou équivalent)

→ Le système bloc sous-jacent n'a aucun moyen de savoir que les blocs alloués ne sont plus utilisés !

Le FS peut ainsi être vide, alors que le volume CEPH est alloué à 100 %

Le problème est le même avec les SSD ou Les baies intelligentes (compellent).

Il faut INFORMER le système bloc sous-jacent
À l'aide d'une commande spéciale : TRIM

→ Monter le fs en mode discard (lent)

Ou utiliser périodiquement fstrim.

RBD : USAGE DIRECT AVEC KVM (LIBRBD, VIRTIO-SCSI)

```
<disk type='network' device='disk'>
  <driver name='qemu' type='raw' cache='writeback' discard='unmap'/>
  <auth username='nfs-gw'>
    <secret type='ceph' uuid='260ee1cc-c10a-44c0-a708-6f466c8adb2b'/>
  </auth>
  <source protocol='rbd' name='NFS/IRTS'>
    <host name='172.20.106.86' port='6789'/>
    <host name='172.20.107.86' port='6789'/>
    <host name='172.20.108.86' port='6789'/>
  </source>
  <target dev='sdb' bus='scsi'/>
</controller type='scsi' index='0' model='virtio-scsi'>
</controller>
```



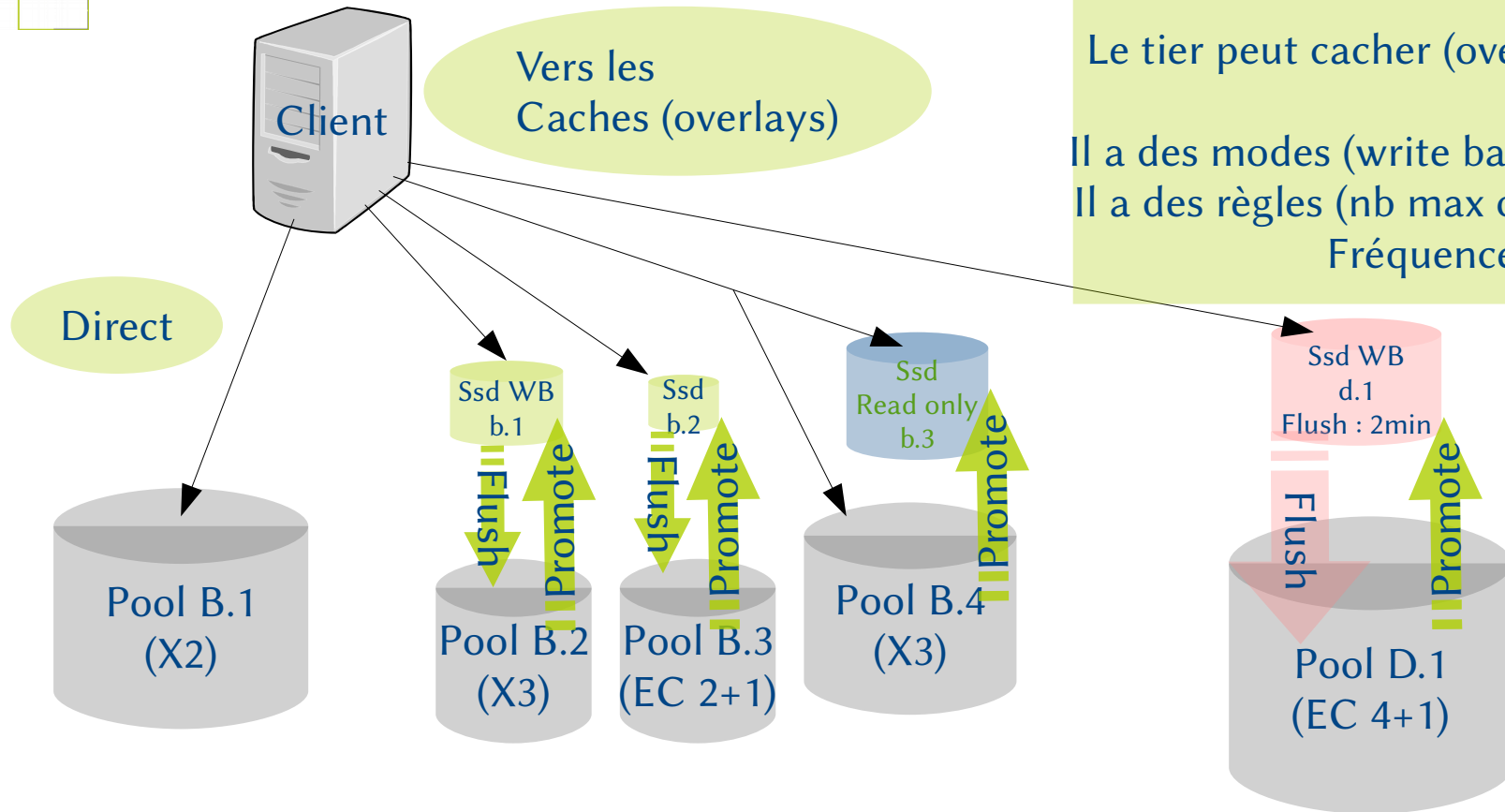
Support du trim/discard
Pour l'invité

Possibilité de fstrim
ou mount -o discard

Synchronisation de
l'allocation par le FS
Et l'allocation par la
couche bloc.

Virtio-scsi + lent que virtio
(10% ?)

TIER, CACHE



Tout pool peut être tier d'un autre.
sans nécessité de similarité
(taille $\neq$, réplicat X2 $\neq$ EC 4+1, racine $\neq$)

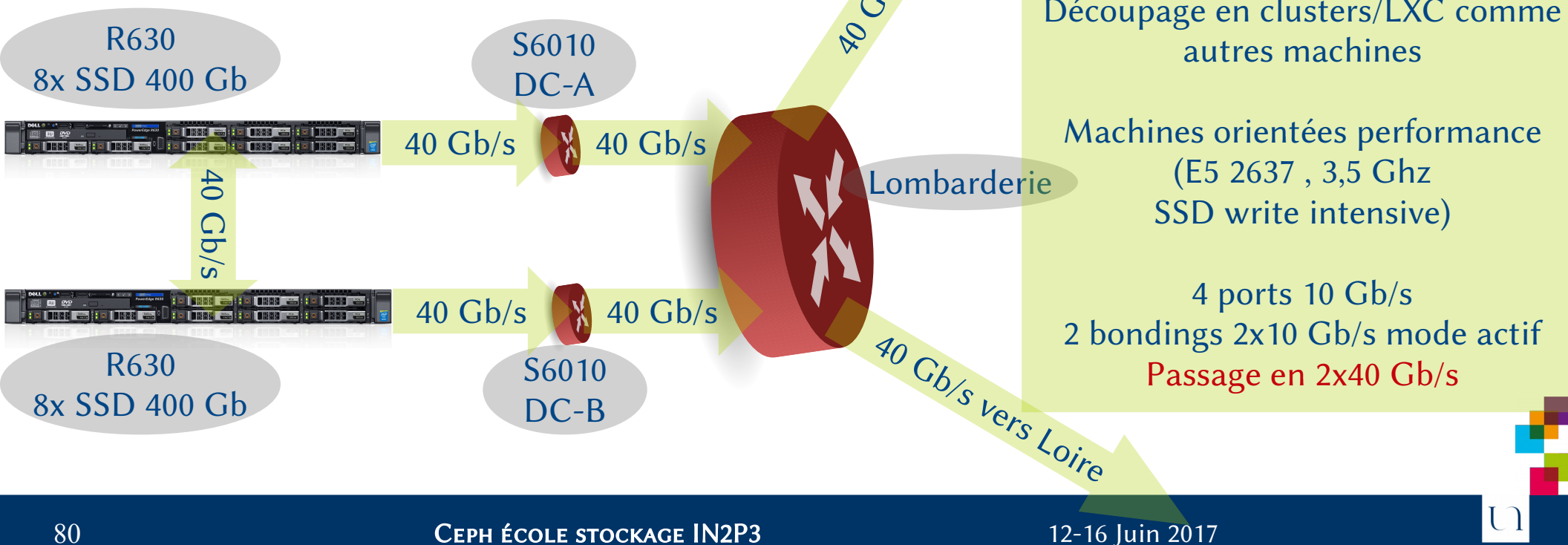
Le tier peut cacher (overlay) le pool d'origine

Il a des modes (write back, forward, read-only)

Il a des règles (nb max objet, taux remplissage)

Fréquence flush...

SETUP MACHINES SSD



RISQUES ASSOCIÉS

Adhérence des objets au cache : jamais flushé si mauvais réglage des règles.

En cas de crash du cache, peut-on récupérer les pools sous-jacents ?

Erase Coding : Pas mieux que 2+1 pour garantir une tolérance à la panne.
(sur notre setup)



PAS à mettre sur tous les POOLS.

CACHE SSD : VALABLE OU PAS ?

« Votre kilométrage peut varier ! »

Le cache n'est pas toujours conseillé dans les docs de ceph.

BUG (#19773) QUI SERA CORRIGÉ EN 10.2.8

```
-OPTION(osd_tier_promote_max_objects_sec, OPT_U64, 5 * 1024*1024)
-OPTION(osd_tier_promote_max_bytes_sec, OPT_U64, 25)
+OPTION(osd_tier_promote_max_objects_sec, OPT_U64, 25)
+OPTION(osd_tier_promote_max_bytes_sec, OPT_U64, 5 * 1024*1024)
```

qui inverse 2 paramètres du « throttling » du cache
et qui le rend peu utile sans cette correction.

Exemple d'utilisation sur une plateforme d'IaaS (OpenNebula).

[Video].



OSD LENTS ET RAPIDES

ceph-mon-lmb-l1-1:~# ceph osd tree

ID	WEIGHT	TYPE	NAME	UP/DOWN	REWEIGHT	PRIMARY-AFFINITY
----	--------	------	------	---------	----------	------------------

-17	1.08600	root	ssd			
-20	0.54300	rack	DC-1-UNIV-A7			
-18	0.54300	host	fleurie			
-15	0.54300	vm	ceph-osd-lmb-l1-ssd1			
3	0.18100	osd.3		up	1.00000	1.00000
4	0.18100	osd.4		up	1.00000	1.00000
5	0.18100	osd.5		up	1.00000	1.00000
-19	0.54300	rack	DC-1-UNIV-B7			
-16	0.54300	host	chiroubles			
-14	0.54300	vm	ceph-osd-lmb-l1-ssd2			
6	0.18100	osd.6		up	1.00000	1.00000
7	0.18100	osd.7		up	1.00000	1.00000
8	0.18100	osd.8		up	1.00000	1.00000

-1 173.87997 root **default**

-5 57.95999 datacenter lmb

-4 57.95999 room lombarderie-ltp

-3 57.95999 host waimea

-2 57.95999 vm ceph-osd-lmb-l1-1

0	7.24500	osd.0		up	1.00000	1.00000
---	---------	-------	--	----	---------	---------

9	7.24500	osd.9		up	1.00000	1.00000
---	---------	-------	--	----	---------	---------

-9 57.95999 datacenter cha

-8 57.95999 room chantrerie-local-telephonie

-7 57.95999 host akarua

-6 57.95999 vm ceph-osd-cha-l1-1

1	7.24500	osd.1		up	1.00000	1.00000
---	---------	-------	--	----	---------	---------

16	7.24500	osd.16		up	1.00000	1.00000
----	---------	--------	--	----	---------	---------

1.00000

-13 57.95999 datacenter loi

-12 57.95999 room loire-presidence

-11 57.95999 host mahi

-10 57.95999 vm ceph-osd-loi-l1-1

2	7.24500	osd.2		up	1.00000	1.00000
---	---------	-------	--	----	---------	---------

23	7.24500	osd.23		up	1.00000	1.00000
----	---------	--------	--	----	---------	---------

OSD LENTS ET RAPIDES

```
ceph-mon-lmb-l1-1:~# ceph osd pool get opennebula crush_ruleset
crush_ruleset: 0
ceph-mon-lmb-l1-1:~# ceph osd pool get HOT-opennebula crush_ruleset
crush_ruleset: 1
```

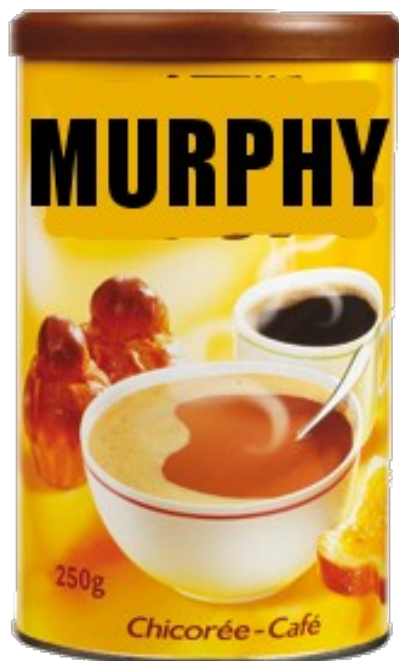
```
# rules
rule replicated_ruleset {
    ruleset 0
    type replicated
    min_size 1
    max_size 10
    step take default
    step chooseleaf firstn 0 type datacenter
    step emit
}
rule replicated_ssd_ruleset {
    ruleset 1
    type replicated
    min_size 1
    max_size 10
    step take ssd
    step chooseleaf firstn 0 type rack
    step emit
}
```

Le principe est
d'avoir plusieurs
Racines différentes

D'appliquer des
Règles crush dessus

Et appliquer ces
Règles crush spécifiques
Sur des pools

L'AMI MURPHY



QUELQUES MOTS DE SAGESSE

Une volumétrie importante !

**Démarrer plusieurs clusters CEPH
(soit physique, soit virtuels)**

Exemple : cluster de sauvegarde

Des besoins différents
Des administrateurs différents
Des versions différentes



ÉVITER LE DÉSASTRE

SCRUB automatique pour le cluster
(patrouille sur les PG et fait des tests de cohérence)
Ne **PAS** utiliser size=2

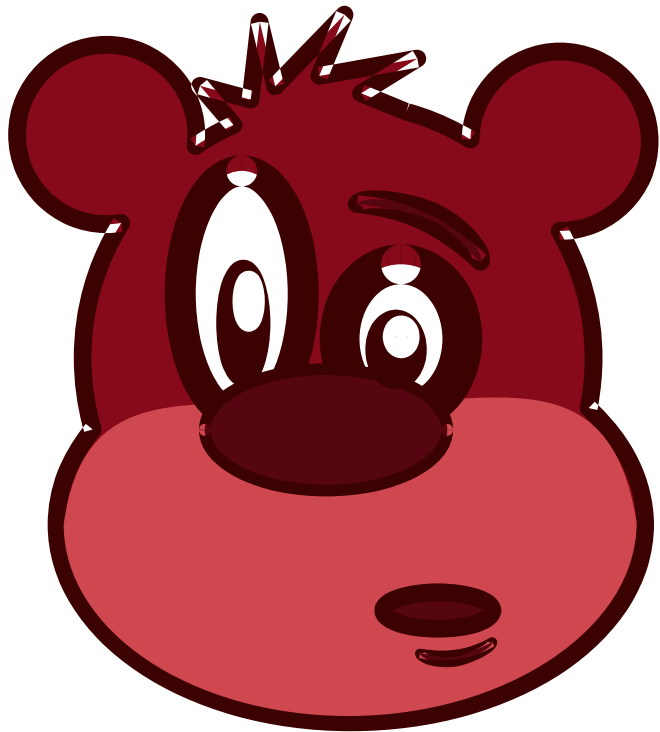
Utiliser les outils sous-jacent du FS des OSD
(mkfs.xfs -m crc=1,finobt=1)

Utiliser des noyaux éprouvés
Une distribution à jour
Des versions de Ceph à jour (mais pas trop)

Monitoring
Lire les logs
Avoir une bonne connaissance du système sous-jacent
Lire les listes de diffusion



ÉVITER LE DÉSASTRE



SAUVEGARDER !
Risque n°1 :
Erreur Humaine !

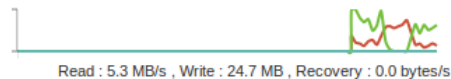


Ceph status

MONITORING

Cluster Health **OK** |rados Gateway | MDS | Misc.

Cluster health at 10:16:14



OK

history

2015-11-18 10:14:33 : OK

Monitors status

b

a

d

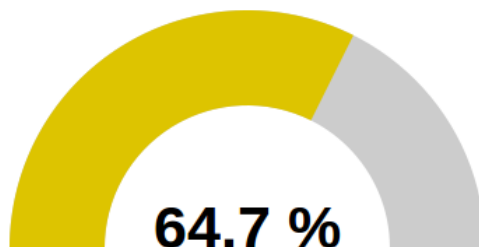
10432 Placement groups

- active+clean+scrubbing+deep
- active+clean
- active+clean+scrubbing



100%

Avail. capacity :
115.3 TB



64.7 %

211.6 TB used
on 326.8 TB

117 OSD

	UP	DOWN
IN	117	0
OUT	0	0
Full	near	full

pools

clean	unclean
9	0

MDSs

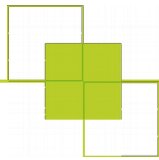
up:in	1	1
up:standby		
max		1

Vue inkscope

Répliqua x3

~75 To
utiles

9 pools



QUELQUES CONSEILS

S'abonner aux listes de diffusion

Prendre le temps de suivre quelques présentations

Admettre le fait que les versions changent tous les ans.

KRAKEN

Format disque bluestore stabilisé
Erasure coding sans Tiers pour RBD
Scrub en pause pendant les phases de
reconstruction / rééquilibrage des données
...

V11.2.0 → Stable non LTS
Kraken = support limité



LUMINOUS (12.2.0)

Bluestore stabilisé

(Formatage spécifique du disque, checksums, compression, journaux plus petits)

Plus d'obligation de mettre en place un système de cache/tier pour les pools d'erasure coding

Nouveau protocole réseau 'AsyncMessenger' plus efficace par défaut

Meilleur temps de détection des pannes d'OSD (pour éviter les freezes)

En cas de reconstruction, les scrubs sont automatiquement dépriorisés

Rados GW NFSv3

....

RETOUR D'EXPÉRIENCE À NANTES



Production à Nantes depuis début 2013, tests depuis 2011...
Utilisation du mode BLOC, pas CephFS.

Déploiement en clusters de containers.

Pour une présentation plus complète de Ceph, notre infrastructure, son évolution :
cf Jres 2013, Meetup Openstack Paris, Ceph Days Paris 2014, Bio Ouest 2014, Cargo Days 2015...
TutoJres 18, ANF CNRS

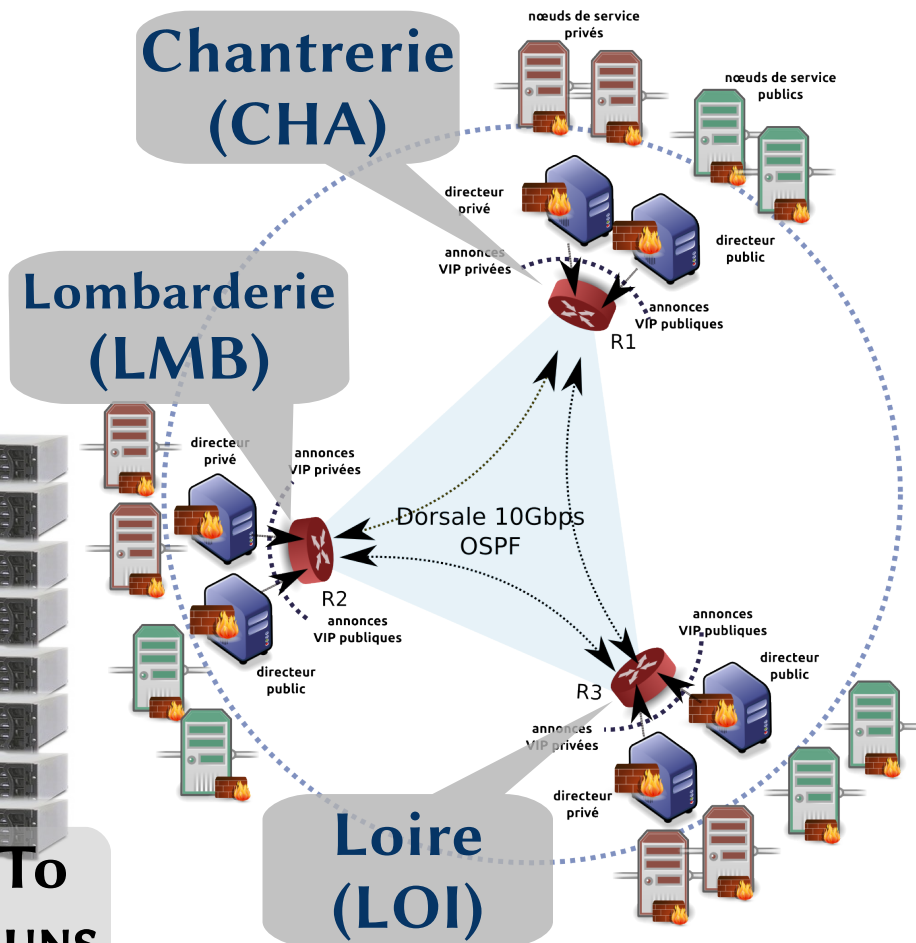
Et voir l'évolution de la plateforme (et des transparents !!)

ÉTAT DÉBUT 2012

Des services majoritairement redondés et distribués

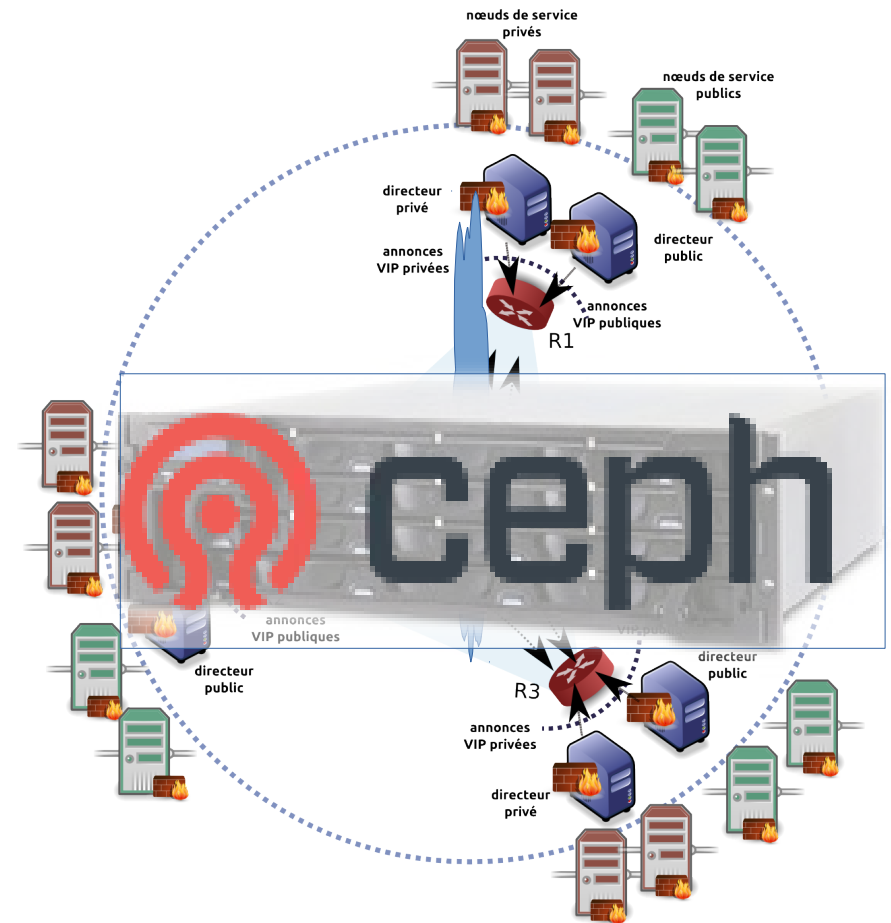
Stockage SAN trop centralisé

> 500 To
> 100 LUNS



ESPACE DE STOCKAGE DISTRIBUÉ

Après une longue quête,
validation de CEPH
début 2012



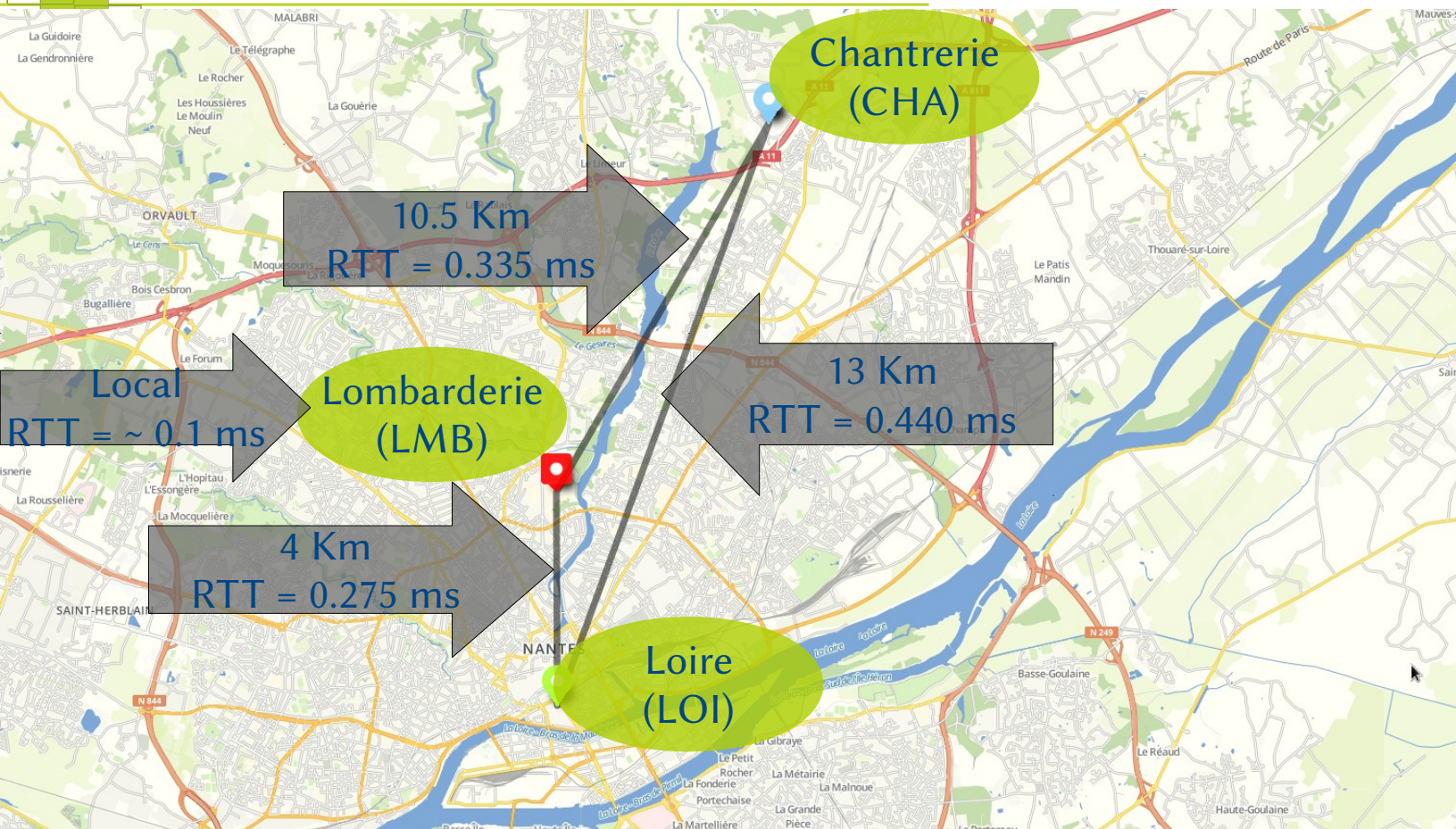
OBJECTIF GLOBAL

DSIN = PRODUCTION, pour ~ 40.000 personnes
Tout type de données & services hébergés

- 1) Fiabilité
- 2) Fiabilité
- 3) Fiabilité
- 4) Tolérance à la panne
- 5) Tolérance à la panne de cerveau
- ...
- 10) Pérennité
- 11) Volumétrie
- 12) Économie
- 13) Performance (pas de HPC!)

Solution à un problème posé dès 2004 !
Pierre angulaire du travail à venir
(IaaS, Cloud)

TOPOLOGIE RÉSEAU À NANTES



3 points de présence

Fibre noire
(Allumée
par la DSIN)

40 Gbit/s
entre sites

1 Datacenter

MISE EN ŒUVRE OPTIMALE

- 2011/2012 : aucun guide de mise en œuvre...
 - Expériences heureuses puis malheureuses !
 - Erreurs de débutant...
 - Peinture pas encore sèche !
- Partage d'expérience
 - Liste de diffusion
 - Canal IRC
 - Meetups, Ceph Days, Ceph Breizh
 - Jres, OpenStack summit
 - Blogs
- 2017 : Plus simple, commun



BUGS & DÉCEPTIONS (2012)

Cluster

Peu d'OSD très volumineux (baies SAN) :

1 unique GROS cluster CEPH

Journaux sur disques : mauvaise idée, Leeeent !

Kernel

Bug mémoire virtuelle, crash fréquent des OSD

Filesystems

BTRFS (Lent, se fige, plante...)

XFS (1 Bug sévère et dévastateur) (Vite corrigé)

Cluster presque plein + bugs + effet domino à la reconstruction = « On casse l'incassable »

Un désastre peut arriver
Choix architecturaux pour atténuer cela



NE PAS METTRE TOUS SES ŒUFS DANS LE MÊME PANIER

Une volumétrie importante

**Des besoins différents
Des administrateurs différents**

**Démarrer plusieurs clusters CEPH
mais de façon virtuelle**

**(Utilisation de virtualisation,
conteneurs LXC)**



DÉPLOIEMENTS (DEPUIS FIN 2013)



1 OSD = 1 Disk = 1 LXC
plus de RAID Hardware
2 SSD SLC partagés (Journaux)
12 OSD / machines
Capacité Brute : 48 To

Déploiements symétriques
par plaque.

3 machines identiques.

ÉVOLUTION (Mi 2015)



LXC D

LXC A

1 OSD = 1 Disk

1 seul LXC par cluster

Pas de SSD. Journal en tête des disques.

24 OSD / machines

Capacité Brute : ~ 24To

«Seulement» 5 LXC max/machine.
Beaucoup d'axes. Orientation
performances en lecture
(Racines IaaS)

ÉVOLUTION (DÉBUT 2016 /2017)



OSD.1

OSD.2

OSD.3

OSD.1

Libre

Libre

Libre

LXC D

LXC A

1 OSD = 1 Disk

1 seul LXC par cluster

SSD Write intensive 12G SAS (Sandisk)

8 OSD / machines

Capacité Brute : ~ 3,2To

Mellanox 40 Gb/s.

Sera décliné avec des SSD

Read intensive , moins cher pour

Certains clusters.

MACHINES DÉPLOYÉES. CAPACITÉ BRUTE GLOBALE

Gen 5
Pour fin 2017
(Perf lecture)

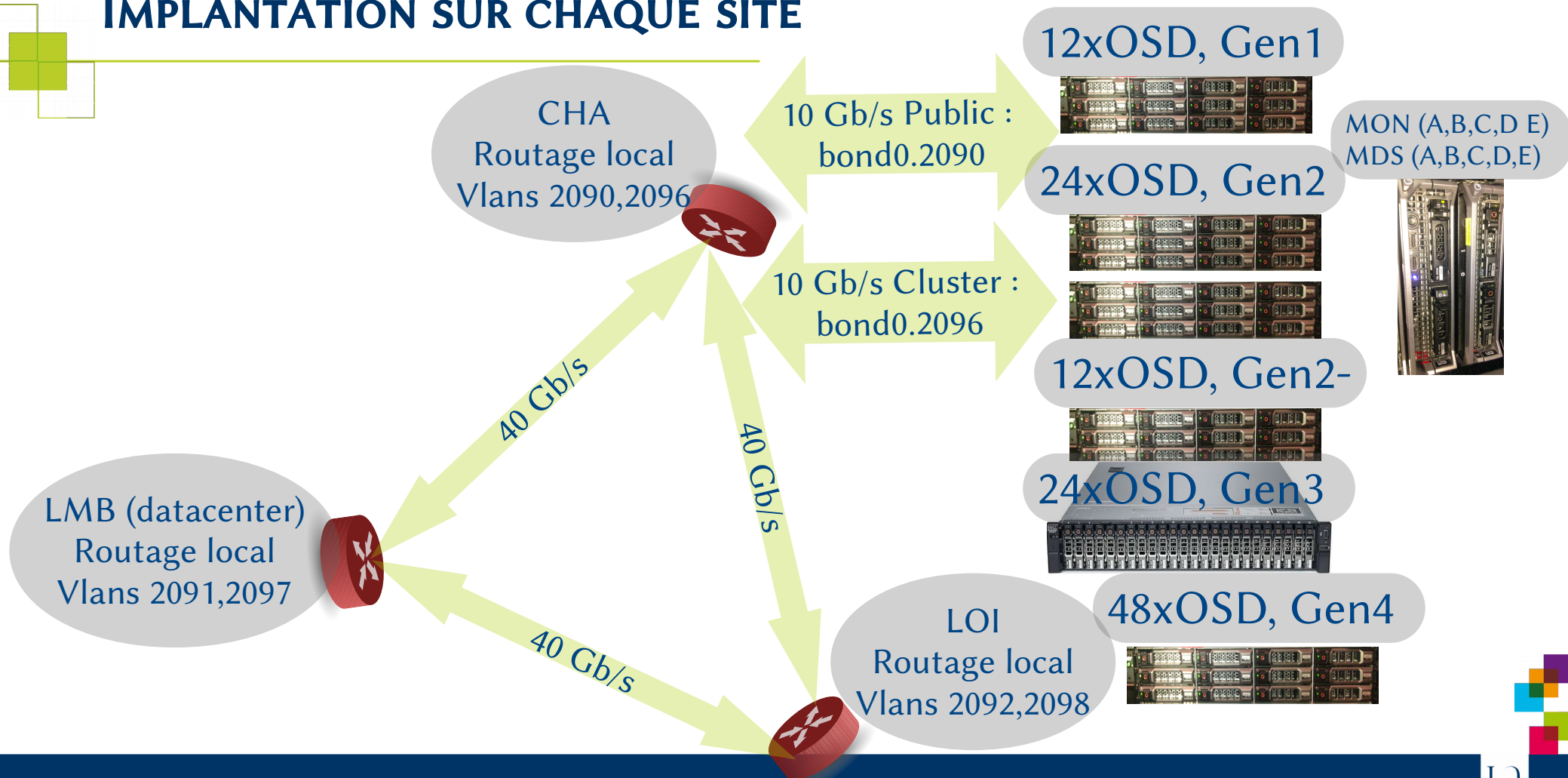
Gén	Destination	Nb	Taille	Version	Volume
1	Mixé	3	12 x 3 To 2 SSD MLC	R720xd Juin 2013	108
2	Mixé/Perf	6	12 x 4 To 2 SSD SLC	R720xd, Nov 2013 Juin 2014	288
2--	Stockage de masse,	3	12x2 (utilisés), 0 SSD	R720xd	72
2-	BUDGET	3	12x4 (utilisés), 0 SSD	Juin 2014	144
3	Perf lecture	3	24x1 (2,5" sas 10k) pas de SSD.	R720xd Nov 2014	72
4	Stockage de masse	9 +9	16 x 8 To Pas de SSD.	R730xd Nov 2015 Nov 2016	1152 (!) + 1152 (!)
4c	PERF !! Cache, écriture, latence.	2	8x400 Gb SSD write intensive	R630 Nov 2015	6,4 cache



CLUSTERS CEPH ET DESTINATION (USAGE * VOLUME * ADMIN)

Nom	Usage	Depuis	Taille	Version
A	Tiers (labos de recherche)	07/2015	Pourrait être très importante	Stable LTS (Jewel)
B	Incubation	03/2013	Petit	Stable, Jewel → Kraken
C C2	Production vSAN systèmes, racines, images	05/2014	Moyenne	Stable LTS (Firefly → Jewel)
D D2,D3	Backups	01/2013 11/2015	Très importante	Stable LTS : Firefly → D2 : Hammer D3 : Jewel
E	Expérimental	06/2014	Petit	Versions de développement instables
F	Data IaaS, Data Vsan	04/2016	Moyen à important	Stable LTS : Jewel
G	Sécurité, logs archives video-surveillance	05/2016	Important	Stable LTS : Jewel
H	Home dirs (cloud)	05/2016	Pourrait être très importante	Stable LTS : Jewel
I	IaaS (Racines, Images)	05/2016	Importante	Stable LTS : Jewel

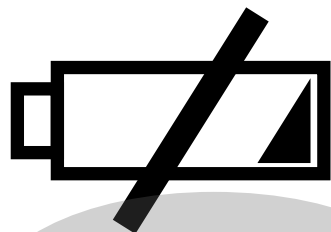
IMPLANTATION SUR CHAQUE SITE



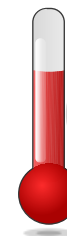
TOLÉRANCE À LA PANNE : TESTÉ !



(Fibres!)
CHA



Panne d'onduleur
et disjoncteur
LOI



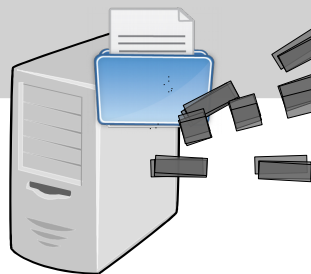
Panne climatiseur
& arrêt d'urgence
LMB

Les problèmes sur machine physique sont rares
En général, une salle machine complète est impactée
3 répliquas permettent d'éviter une reconstruction

CONNEXIONS RÉSEAU, SCALABILITÉ

Les clients initient des
connexions directes avec les OSD
= Scalabilité

Beaucoup de connexions,
stresse les firewalls



10 Gbits/s pour de
Bonnes performances

TP.



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 déjà installé et fonctionnel, mais il n'y a pas d'OSD

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

Préparer les volumes des OSD (formatage, 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

```
onetranslate 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

NE DEVRAIT PAS ÊTRE
PARTAGÉ

ceph.client.admin.keyring

```
[client.admin]
key = AQALHUxY6GaqBRAAAcQUwzQeQ0JIZr4nCCfe8g==
```

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

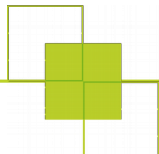
```
root@debian:~#dd if=/dev/urandom of=FichierTireBouchon bs=1M count=16
root@debian:~# md5sum FichierTireBouchon
eb05fa217d1b9569c426b07e92a84854  FichierTireBouchon
```

```
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

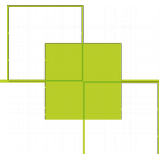
Création d'un volume (idem LUN)

Informations sur le volume créé

On le mappe sur la machine

Remarquer que les 100G n'ont pas été alloués

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

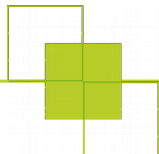


RBD : TESTS (AVEC KRBD)

```
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)



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



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
```

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'
AQB5cE1Y10/JDBAAVBtnxzCMu3y30eBXJzlBkw==
```

Sur la VM cliente

```
cat /etc/ceph/ceph.client.ANF.keyring
[client.ANF]
    key = AQB5cE1Y10/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 hiérarchie des objets, indique les osd up / down

```
ceph osd tree
```

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
```

Sur un OSD

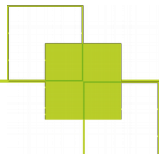
CHANGER LES RÈGLES D'UN POOL

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 rbd 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
```

Peupler le /etc/hosts

```
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/*"
```

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

AJOUTER UN OSD (2)

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

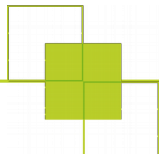
Les données sont en train de migrer sur le 4 eme OSD

```
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
```

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

La règle existe dans le cluster mais est
Non appliquée.

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

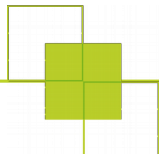
```
rados get -p objets_utiles ObjetTireBouchon FichierTireBouchon2 ; md5sum FichierTireBouchon2
eb05fa217d1b9569c426b07e92a84854 FichierTireBouchon2
```

```
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 ?

Crédits : Opencliparts / Openstreetmap / Ceph.com

