# CEPH – SUPERVISION ET OPTIMISATIONS

École IN2P3 2017 – Stockage distribué

Mathieu GAUTHIER-LAFAYE

# Comment intervenir

- Une opération à la fois !
  - Une opération peut être l'ajout ou le drainage de l'ensemble des osd d'un serveur
  - Attention à bien prendre en compte vos règles de réplication (datacenter, rack, serveur)
  - Si vous avez 3 réplicas même avec une min_size à 1 évitez de faire une opération sur deux réplicas d'un PG
  - Attendre que le cluster soit revenu en état HEALTH_OK
- Tester avant d'intervenir sur la production avec un environnement le plus représentatif de votre environnement de production.

# Dépannage - Monitor

- Essayez de :
  - Vérifier que les services des moniteurs sont bien lancés
    - systemctl status ceph-mon@<host>
    - Exemple : systemctl status ceph-mon@mon01
  - Vérifier que le serveur est accessible:
    - Connexion SSH
    - Port du moniteur via netcat, telnet
  - Vérifier que l'heure du serveur est bien synchronisée
- Si ceph health / status ne répond toujours  pas :
    - Le nombre de moniteur est insuffisant pour atteindre le quorum
    - Vérifiez le statut des moniteurs via la socket d'administration

# Dépannage – Monitor

- Statut du moniteur via la socket d'administration
  - Connexion au monitor via SSH
  - Utilisation de la commande :
    ceph daemon mon.<id> mon_status
  - Exemple d'information :
    - state : peaon
    - qurum : [1, 2 ]
    - outside_quorum: []
  - État du moniteur :
    - probing : cherche à joindre les autres moniteurs
    - electing : en cours d'élection (court – clock skew?)
    - synchronizing : synchronisation des données avec les autres moniteurs
    - leader : le moniteur est dans le quorum et est à la tête du cluster
    - peon : le moniteur est dans le quorum

# Dépannage - OSD

- Identifiez le(s) serveur(s) et le(s) OSD concerné(s) :
  ceph osd tree
- Vérifiez que les services sont bien lancés
- Vérifiez que les horloges des serveurs sont bien synchronisées
- Vérifiez que les ports sont bien ouverts
- Vérifiez la place disque sur la partition de données du moniteur
- Consultez les journaux /var/log/<cluster>-<name>.log
- Augmentez le niveau de verbosité

# Journaux et débogage

- Verbosité :
  - Niveau de 1 à 20 (laconique à verbeux)
  - Spécifié par la forme : niveau / niveau en mémoire
    Par exemple :
    - 5/10 : niveau de 5 pour les journaux et niveau en mémoire de 10
    - 15 : niveau de 15 pour les journaux et la mémoire
  - Les enregistrements en mémoire sont écrits dans les journaux dans certains cas :
    - Un signal fatal
    - Une assertion est déclenchée
    - Sur demande via la socket d'administration du service (mon, osd, …)

# Journaux et débogage

- Sous systèmes
  - Chacun a son propre niveau de verbosité
  - Chacun a son niveau de verbosité par défaut qui est suffisant en temps normal
  - Exemple de sous système et leurs niveaux verbosités par défaut :
    - default : 0/5
    - mon : 1/5
    - paxos : 0/5
    - osd : 0/5
    - filestore: 1/5
    - crush : 0/5
    - …

# Journaux et débogage

- Configuration générale :
  - Fichier :
    - log file : /var/log/ceph/$cluster-$name.log
    - log max new : nombre maximum de nouveaux fichiers
    - log max recent : nombre maximum d'évenements par fichiers
  - Syslog :
    - log to syslog
    - err to syslog
    - clog to syslog (cluster log)
    - mon cluster log to syslog
- Configuration des sous systèmes
  - debug {subsystem} = {log-level}/{memory-level}
  - Exemple : debug mds balancer = 1/20

# Journaux et débogage

- Application des paramètres :
  - Dans le fichier de configuration du cluster
    Exemple : ceph.conf
  - Injection du paramètre :
    - Par le moniteur :
      ceph tell <daemon-type>.<daemon id or *> injectargs --<name> <value>
    - Depuis le serveur :
      ceph daemon <daemon-type>.<daemon_id or *> config set <name> <value>
    - Exemle:
      - ceph tell osd.1 injectargs --debug-osd 0/5
      - ceph daemon osd.1 config set debug-osd 0/5

# Superviser – Suivre les évènements

- La commande « ceph » permet de suivre les évènements via les arguments :
  - --watch (ou –w) : suit les changements
  - --watch-debug : évènements de débogage
  - --watch-info, --watch-warning, --watch-error

```
cluster b370a29d-9287-4ca3-ab57-3d824f65e339
 health HEALTH_OK
 monmap e1: 1 mons at {ceph1=10.0.0.8:6789/0}, election epoch 2, quorum 0 ceph1
 osdmap e63: 2 osds: 2 up, 2 in
  pgmap v41338: 952 pgs, 20 pools, 17130 MB data, 2199 objects
        115 GB used, 167 GB / 297 GB avail
             952 active+clean

2014-06-02 15:45:21.655871 osd.0 [INF] 17.71 deep-scrub ok
2014-06-02 15:45:47.880608 osd.1 [INF] 1.0 scrub ok
2014-06-02 15:45:48.865375 osd.1 [INF] 1.3 scrub ok
2014-06-02 15:45:50.866479 osd.1 [INF] 1.4 scrub ok
2014-06-02 15:45:01.345821 mon.0 [INF] pgmap v41339: 952 pgs: 952 active+clean;
17130 MB data, 115 GB used, 167 GB / 297 GB avail
2014-06-02 15:45:05.718640 mon.0 [INF] pgmap v41340: 952 pgs: 1
active+clean+scrubbing+deep, 951 active+clean; 17130 MB data, 115 GB used, 167
GB / 297 GB avail
2014-06-02 15:45:53.997726 osd.1 [INF] 1.5 scrub ok
2014-06-02 15:45:06.734270 mon.0 [INF] pgmap v41341: 952 pgs: 1
active+clean+scrubbing+deep, 951 active+clean; 17130 MB data, 115 GB used, 167
GB / 297 GB avail
2014-06-02 15:45:15.722456 mon.0 [INF] pgmap v41342: 952 pgs: 952 active+clean;
17130 MB data, 115 GB used, 167 GB / 297 GB avail
2014-06-02 15:46:06.836430 osd.0 [INF] 17.75 deep-scrub ok
2014-06-02 15:45:55.720929 mon.0 [INF] pgmap v41343: 952 pgs: 1
active+clean+scrubbing+deep, 951 active+clean; 17130 MB data, 115 GB used, 167
GB / 297 GB avail
```

# Superviser – Commandes ceph health

- Les différents statuts :
  - HEALTH_OK : état normal
  - HEALTH_WARN : état dégradé (noout, reconstruction, …)
  - HEALTH_ERR : état à éviter ;)
- Exemples :
  - ceph health
    HEALTH_WARN 1 mons down, quorum 1,2 mon02,mon03
  - ceph health detail
    HEALTH_WARN 1 mons down, quorum 1,2 mon02,mon03
    mon.mon01 (rank 0) addr 172.16.19.162:6789/0 is down (out of quorum)

# Superviser – Commandes ceph status

- C'est la commande la plus utilisée pour consulter l'état de son cluster car elle est assez complète.
- Elle donne des informations utiles sur :
  - L'état du cluster
  - L'état des groupes de placement
  - Les opérations en cours sur le cluster
  - …

# Superviser – Commandes ceph status

```
    cluster 934b93e5-9224-43f0-bbbc-4ffb61d44451
     health HEALTH_WARN
            24 pgs degraded
            24 pgs stuck unclean
            recovery 79/201 objects degraded (39.303%)
     monmap e3: 3 mons at {mon01=172.16.19.162:6789/0,mon02=172.16.19.163:6789/0,
            election epoch 16, quorum 0,1,2 mon01,mon02,mon03
        mgr active: mon03 standbys: mon01, mon02
     osdmap e179: 9 osds: 9 up, 9 in
            flags sortbitwise,require_jewel_osds,require_kraken_osds
      pgmap v615: 288 pgs, 1 pools, 268 MB data, 67 objects
            1146 MB used, 241 GB / 242 GB avail
            79/201 objects degraded (39.303%)
                 264 active+clean
                  24 active+degraded
recovery io 2391 kB/s, 0 objects/s
  client io 1195 kB/s wr, 0 op/s rd, 0 op/s wr
```

# Superviser – Commandes ceph osd tree

```
ID  WEIGHT  TYPE NAME                 UP/DOWN REWEIGHT PRIMARY-AFFINITY
-14 2.39955 root salle
-12 1.38808     room salle1
-15 0.50249         rack baie01
 -2 0.06020             host clapt001
  0 0.06020                 osd.0          up  1.00000          1.00000
 -3 0.44229             host clapt002
  7 0.44229                 osd.7          up  1.00000          1.00000
-16 0.88559         rack baie02
 -8 0.44279             host clapt004
  3 0.44279                 osd.3          up  1.00000          1.00000
 -4 0.44279             host clapt008
  1 0.44279                 osd.1          up  1.00000          1.00000
-13 1.01147     room salle2
-17 1.01147         rack baie03
 -9 0.44279             host clapt005
  4 0.44279                 osd.4          up  1.00000          1.00000
-10 0.44279             host clapt006
  5 0.44279                 osd.5          up  1.00000          1.00000
-11 0.12589             host clapt007
  6 0.12589                 osd.6          up  1.00000          1.00000
 -1 2.39955 root default
 -5 0.88559     replica replica1
 -9 0.44279         host clapt005
  4 0.44279             osd.4              up  1.00000          1.00000
-10 0.44279         host clapt006
  5 0.44279             osd.5              up  1.00000          1.00000
 -6 0.88509     replica replica2
 -3 0.44229         host clapt002
  7 0.44229             osd.7              up  1.00000          1.00000
 -4 0.44279         host clapt008
  1 0.44279             osd.1              up  1.00000          1.00000
 -7 0.62888     replica replica3
 -2 0.06020         host clapt001
  0 0.06020             osd.0              up  1.00000          1.00000
```

# Superviser – Commandes ceph osd pool stat

- Donne des informations par pool :
  - Client IO / vitesse
  - Recovery IO / vitesse
- Diviser en pool permet de mieux surveiller
- $ ceph osd pool stats
  pool rbd id 2
  2/193 objects degraded (1.036%)
  2/193 objects misplaced (1.036%)
  client io 39473 kB/s wr, 0 op/s rd, 9 op/s wr

# Superviser – Commandes ceph pg dump pgs_brief

```
dumped pgs_brief in format plain
pg_stat state    up       up_primary       acting   acting_primary
2.ff    active+clean    [3,5,7] 3        [3,5,7] 3
2.fe    active+clean    [3,4,7] 3        [3,4,7] 3
2.fd    active+clean    [0,4,7] 0        [0,4,7] 0
2.fc    active+clean    [3,7,4] 3        [3,7,4] 3
2.fb    active+clean    [4,7,3] 4        [4,7,3] 4
2.fa    active+clean    [7,3,4] 7        [7,3,4] 7
2.f9    active+clean    [7,4,3] 7        [7,4,3] 7
2.f8    active+clean    [6,4,1] 6        [6,4,1] 6
2.f7    active+clean    [1,3,4] 1        [1,3,4] 1
2.f6    active+clean    [1,4,3] 1        [1,4,3] 1
2.f5    active+clean    [1,5,3] 1        [1,5,3] 1
2.f4    active+clean    [1,3,5] 1        [1,3,5] 1
2.f3    active+clean    [7,6,5] 7        [7,6,5] 7
2.f2    active+clean    [4,7,3] 4        [4,7,3] 4
2.f1    active+clean    [4,7,3] 4        [4,7,3] 4
2.f0    active+clean    [1,5,3] 1        [1,5,3] 1
2.f     active+clean    [1,3,5] 1        [1,3,5] 1
2.ef    active+clean    [7,5,3] 7        [7,5,3] 7
2.ee    active+clean    [7,5,6] 7        [7,5,6] 7
2.ed    active+clean    [7,5,3] 7        [7,5,3] 7
```

# Superviser – Commandes ceph osd map

- La commande permet de retrouver les groupes de placement d'un objet
- Syntaxe : ceph osd map <pool> <object>
- Exemple :
  $ ceph osd map rbd rbd_data.3b11abb74b0dc51.00000000000000dc
  osdmap e21545 pool 'rbd' (2) object
  'rbd_data.3b11abb74b0dc51.00000000000000dc' -> pg 2.f642a0ff
  (2.ff) -> up ([3,5,7], p3) acting ([3,5,7], p3)

# Superviser – Commandes ceph osd df

```
ID WEIGHT  REWEIGHT SIZE   USE    AVAIL  %USE VAR  PGS
 0 0.06020  1.00000 63118M  1495M 58394M 2.37 0.90  17
 7 0.44229  1.00000   452G 19667M   410G 4.24 1.61 225
 3 0.44279  1.00000   453G 15421M   415G 3.32 1.26 180
 1 0.44279  1.00000   453G  2799M   427G 0.60 0.23 130
 4 0.44279  1.00000   453G 11237M   419G 2.42 0.92 130
 5 0.44279  1.00000   453G 10565M   420G 2.28 0.86 126
 6 0.12589  1.00000   128G  5103M   117G 3.87 1.47  59
 4 0.44279  1.00000   453G 11237M   419G 2.42 0.92 130
 5 0.44279  1.00000   453G 10565M   420G 2.28 0.86 126
 7 0.44229  1.00000   452G 19667M   410G 4.24 1.61 225
 1 0.44279  1.00000   453G  2799M   427G 0.60 0.23 130
 0 0.06020  1.00000 63118M  1495M 58394M 2.37 0.90  17
 3 0.44279  1.00000   453G 15421M   415G 3.32 1.26 180
 6 0.12589  1.00000   128G  5103M   117G 3.87 1.47  59
              TOTAL  2457G 66289M  2267G 2.63
MIN/MAX VAR: 0.23/1.61  STDDEV: 1.13
```

# Superviser – Commandes ceph tell osd.<id> bench

- Permet d‘effectuer une mesure de performance d‘un OSD
- Syntaxe : ceph tell osd.<id> bench
- Exemple :

```
$ ceph tell osd.0 bench
{
    "bytes_written": 1073741824,
    "blocksize": 4194304,
    "bytes_per_sec": 39531101
}
```

# Superviser – Commandes ceph report

- 246 260 lignes d'informations sur un cluster de production !
- Informations détaillés sur :
  - Le cluster (uuid)
  - Etat du cluster et des services
  - Les moniteurs (quorum)
  - Les OSD
  - Les groupes de placements
- Permet de récupérer énormément d'information en une seule commande.
- La commande peut-être très utile pour écrire un script qui a besoin d'une quantité importante d'informations sur son cluster.

# Superviser – Commandes ceph report


```
{
    "cluster_fingerprint": "cb858b40-5bd2-4392-ba00-ec663022c30f",
    "version": "10.2.3",
    "commit": "ecc23778eb545d8dd55e2e4735b53cc93f92e65b",
    "timestamp": "2017-06-14 20:38:48.341955",
    "tag": "",
    "health": {
        "health": {
            "health_services": [
                {
                    "mons": [
                        {
                            "name": "mon01   ",
                            "kb_total": 5029504,
                            "kb_used": 81852,
                            "kb_avail": 4669124,
                            "avail_percent": 92,
                            "last_updated": "2017-06-14 20:38:11.733363",
                            "store_stats": {
                                "bytes_total": 73279379,
                                "bytes_sst": 0,
                                "bytes_log": 1407128,
                                "bytes_misc": 71872251,
                                "last_updated": "0.000000"
                            },
                            "health": "HEALTH_OK"
                        },
                        {
                            "name": "mon02   ",
                            "kb_total": 5029504,
                            "kb_used": 80880,
                            "kb_avail": 4670096,
```

# Superviser – Commandes Formats de sortie

- Les commandes CEPH permettent de sortir l'information dans plusieurs formats :
  - plain (en général le défaut)
  - json, json-pretty,
  - xml, xml-pretty
- Très utile pour écrire des scripts :
  - Monitoring
  - Métrologie
- Le format « json » ou « xml » est souvent plus verbeux que le format « plain »

# Superviser – Commandes format de sortie

```
root@adm01:/home/user# ceph health
HEALTH_OK
root@adm01:/home/user# ceph health -f json-pretty

{
    "health": {
        "health_services": [
            {
                "mons": [
                    {
                        "name": "mon01",
                        "kb_total": 1886280,
                        "kb_used": 330852,
                        "kb_avail": 1441560,
                        "avail_percent": 76,
                        "last_updated": "2017-06-13 11:55:06.253583",
                        "store_stats": {
                            "bytes_total": 13525770,
                            "bytes_sst": 0,
                            "bytes_log": 2568964,
                            "bytes_misc": 10956806,
                            "last_updated": "0.000000"
                        },
                        "health": "HEALTH_OK"
                    },
                    {
                        "name": "mon02",
                        "kb_total": 1886280,
                        "kb_used": 332088,
                        "kb_avail": 1440324,
```

# Superviser – API REST

- L'API Rest n'est pas lancée par défaut :
  - Il faut ajouter des services systemd pour la démarrer
- Elle permet d'effectuer la majorité des opérations réalisables avec la commande ceph
- Elle propose les mêmes formats de sortie que la commande ceph
- Elle s'authentifie avec un utilisateur pour réaliser les opérations sur le cluster (cephx)
- Elle n'est pas sécurisée (sans serveur web frontend) :
  - Elle ne gère pas d'authentification utilisateur
  - Elle ne gère pas non plus des autorisations (ACL sur les opérations)
  - Il faut filtrer via le réseau avec firewall à minima (iptables/nftables) !
- Elle est réécrite dans la prochaine version (Luminous)…

# Supervision

- Utiliser un outil de supervision automatique (Nagios, Zabbix, …)
- Général :
  - Réseau : connectivité des serveurs (ping, ouverture des ports, …)
- Système :
  - charge
  - mémoire / swap
  - espace disque
  - services importants (ntp, ceph-mon, ceph-osd@…)

# Supervision

- CEPH :
  - État du cluster
  - Vérification des services : mon, osd, mds, rgw
  - Espace libre par pool
  - Latences (apply, commit) des OSD
- Il existe des plugins Nagios (https://github.com/valerytschopp/ceph-nagios-plugins) et Zabbix.

# Cluster

| | | | | | |
|---|---|---|---|---|---|
| ceph_health | OK | 13-06-2017 13:53:23 | 54d 7h 7m 21s | 1/3 | CLUSTER OK: cluster: 16 ok, 0 warning, 0 unknown, 0 critical |
| ceph_mon | OK | 13-06-2017 13:52:35 | 186d 17h 17m 2s | 1/3 | CLUSTER OK: cluster: 3 ok, 0 warning, 0 unknown, 0 critical |
| ceph_osd | OK | 13-06-2017 13:52:47 | 54d 7h 21m 58s | 1/3 | CLUSTER OK: cluster: 47 ok, 0 warning, 0 unknown, 0 critical |
| ceph_osd_perf | OK | 13-06-2017 13:52:59 | 54d 7h 21m 46s | 1/3 | CLUSTER OK: cluster: 47 ok, 0 warning, 0 unknown, 0 critical |

HOST

| | | | | | |
|---|---|---|---|---|---|
| nrpe-ceph_health | OK | 13-06-2017 13:50:23 | 54d 7h 8m 34s | 1/3 | HEALTH OK |
| nrpe-ceph_mon | OK | 13-06-2017 13:49:37 | 54d 7h 19m 22s | 1/3 | MON OK |
| nrpe-ceph_osd_33 | OK | 13-06-2017 13:49:47 | 54d 7h 20m 10s | 1/3 | OSD OK |
| nrpe-ceph_osd_34 | OK | 13-06-2017 13:49:59 | 54d 7h 19m 58s | 1/3 | OSD OK |
| nrpe-ceph_osd_6 | OK | 13-06-2017 13:50:11 | 54d 7h 19m 46s | 1/3 | OSD OK |
| nrpe-ceph_osd_7 | OK | 13-06-2017 13:50:23 | 54d 7h 19m 34s | 1/3 | OSD OK |
| nrpe-ceph_osd_perf_33 | OK | 13-06-2017 13:49:37 | 54d 7h 19m 22s | 1/3 | OSD_PERF OK: osd_33 |
| nrpe-ceph_osd_perf_34 | OK | 13-06-2017 13:49:47 | 4d 9h 52m 52s | 1/3 | OSD_PERF OK: osd_34 |
| nrpe-ceph_osd_perf_6 | OK | 13-06-2017 13:49:59 | 20d 15h 46m 52s | 1/3 | OSD_PERF OK: osd_6 |
| nrpe-ceph_osd_perf_7 | OK | 13-06-2017 13:50:11 | 5d 13h 19m 51s | 1/3 | OSD_PERF OK: osd_7 |
| nrpe-client_version | OK | 13-06-2017 13:48:23 | 54d 7h 16m 33s | 1/3 | NRPE v2.15 |
| nrpe-cpuload | OK | 13-06-2017 13:47:35 | 4d 12h 19m 8s | 1/3 | OK - load average: 7.91, 8.10, 7.56 |
| nrpe-disks_space | OK | 13-06-2017 13:47:47 | 54d 7h 17m 10s | 1/3 | DISK OK - free space: /tmp 2587 MB (99% inode=99%): /var/tmp 4301 MB (99% inode=99%): / 695 MB (40% inode=73%): /boot 299 MB (68% inode=99%): /usr 1349 MB (52% inode=69%): /var 1745 MB (67% inode=97%): /var/log 2087 MB (94% inode=99%): /var/lib/ceph/mon 4215 MB (90% inode=99%): |
| nrpe-ntp_time | OK | 13-06-2017 13:42:59 | 4d 15h 52m 50s | 1/5 | NTP OK: Offset 8.189678192e-05 secs |
| nrpe-services | OK | 13-06-2017 13:48:11 | 27d 7h 23m 49s | 1/3 | SERVICE OK: corosync, ksmtuned, pve-cluster, pve-firewall, pve-ha-crm, pve-ha-lrm, pvedaemon, pvefw-logger, pveproxy, pvestatd, spiceproxy, ceph-mon@clap001, ceph-osd@6, ceph-osd@7, ceph-osd@33, ceph-osd@34 currently running. |
| nrpe-swap | OK | 13-06-2017 13:48:23 | 97d 0h 18m 19s | 1/3 | SWAP OK - 97% free (4946 MB out of 5119 MB) |
| nrpe-total_procs | OK | 13-06-2017 13:47:35 | 54d 7h 17m 22s | 1/3 | PROCS OK: 387 processes |
| nrpe-users | OK | 13-06-2017 13:47:47 | 54d 7h 17m 10s | 1/3 | USERS OK - 0 users currently logged in |
| nrpe-zombie_procs | OK | 13-06-2017 13:47:59 | 54d 7h 16m 58s | 1/3 | PROCS OK: 0 processes with STATE = Z |
| ping-serveurs | OK | 13-06-2017 13:51:11 | 54d 7h 19m 46s | 1/3 | [rtt min/avg/max/mdev = 0.320/0.362/0.443/0.039 ms] |
| resolv-a | OK | 13-06-2017 13:23:23 | 186d 16h 46m 17s | 1/5 | DNS OK - 0,041 seconds response time (clap001.lapp.in2p3.fr. 2505 IN A 10.159.104.20) |
| ssh | OK | 13-06-2017 13:50:35 | 54d 7h 20m 22s | 1/3 | SSH OK - OpenSSH_6.7p1 Debian-5+deb8u3 (protocol 2.0) |

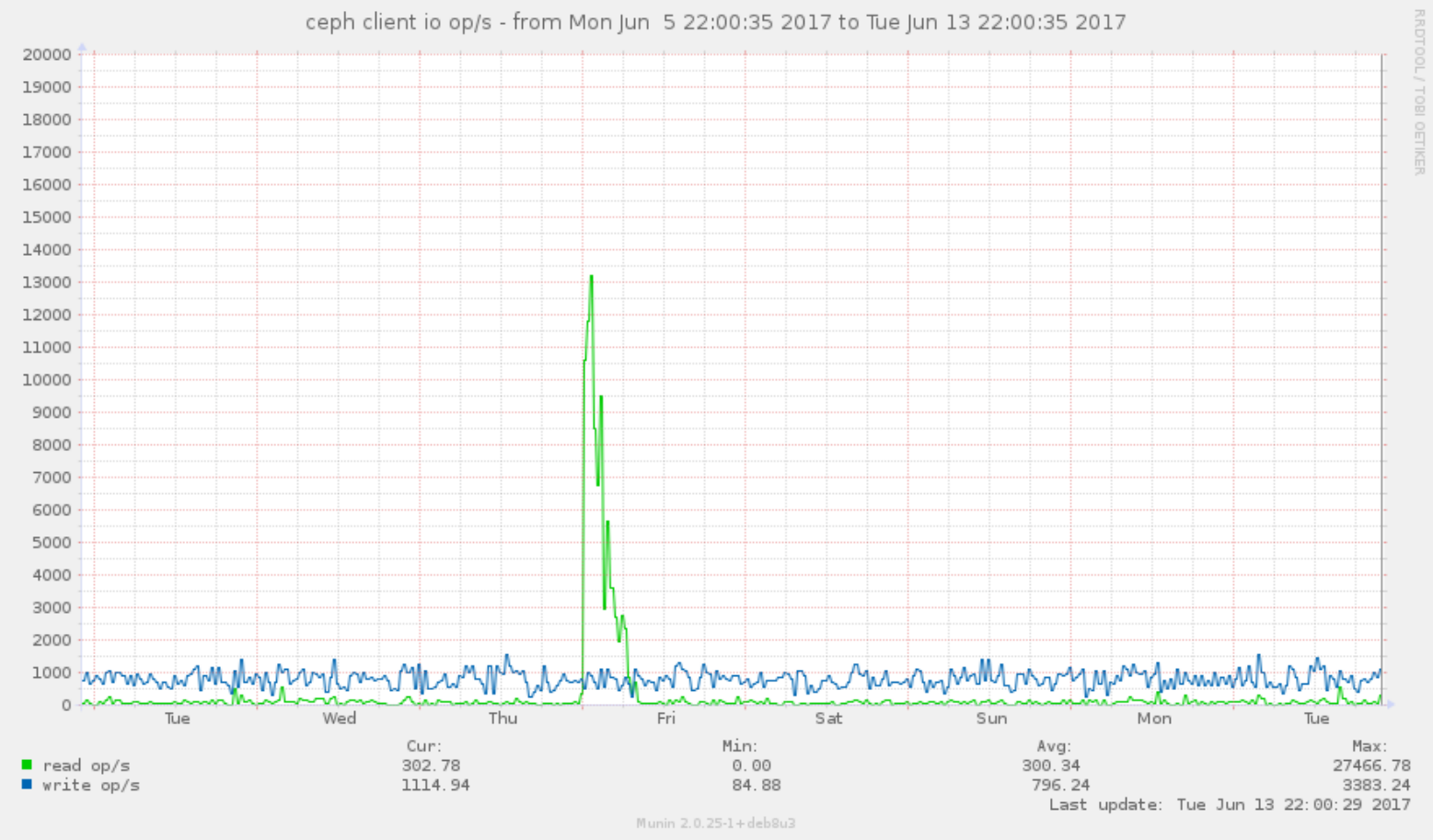
# Métrologie – Métriques

- Métriques intéressantes :
  - Client read / write
  - Client iop/s read / write
  - Recovery read / write
  - Recovery iops read / write
  - Espace occupé / disponible
  - Latence d'un OSD (apply / commit)
  - Benchmark d'un OSD
  - Consommation des processus CEPH
  - *Mon count* / Osd count

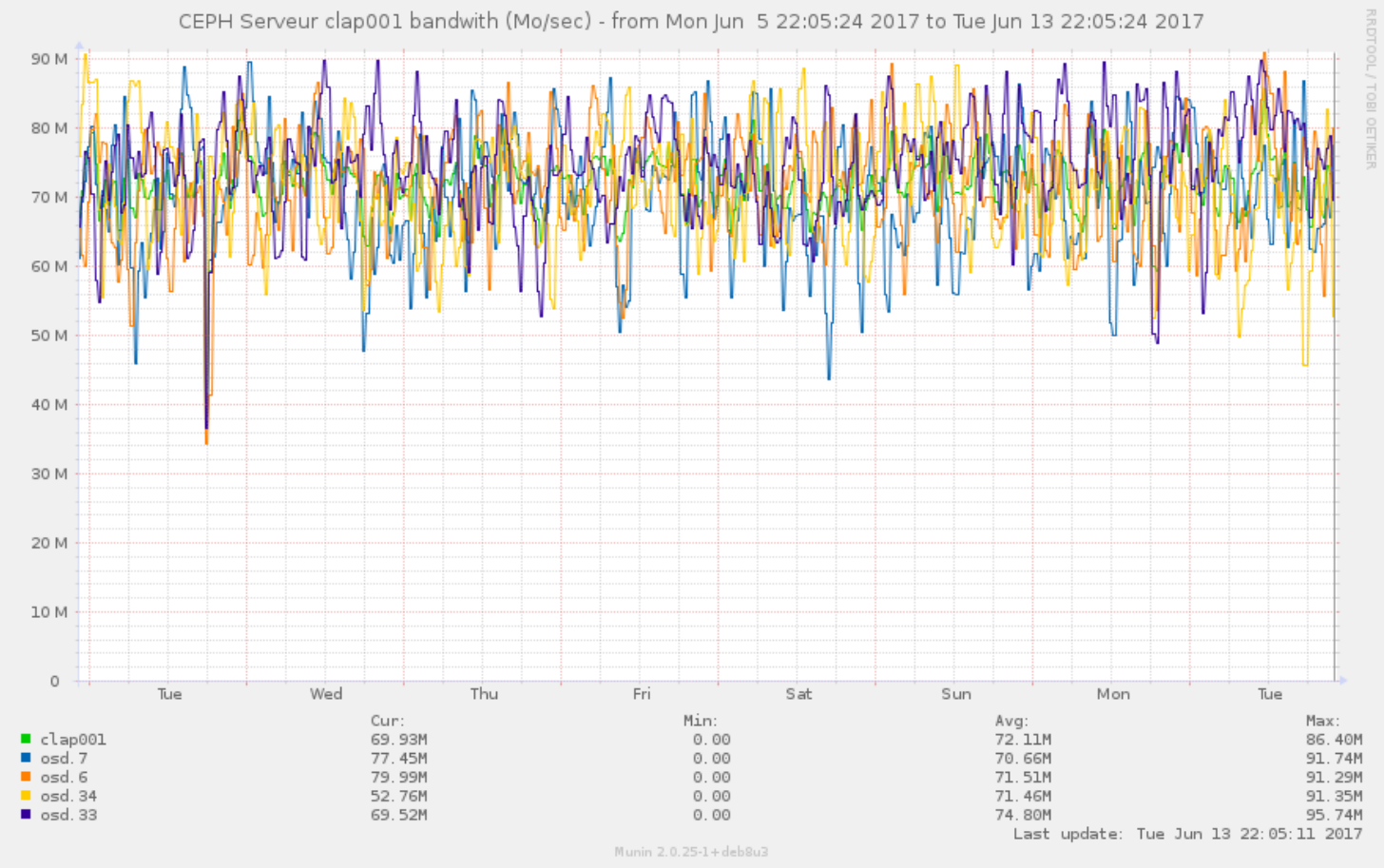
# Métrologie – Métriques

- Commandes intéressantes :
  - ceph osd pool stats --format json-pretty
  - ceph osd status --format json-pretty
  - ceph osd perf
  - ceph osd report
  - ceph tell osd.<id> bench

# Métrologie – Métriques



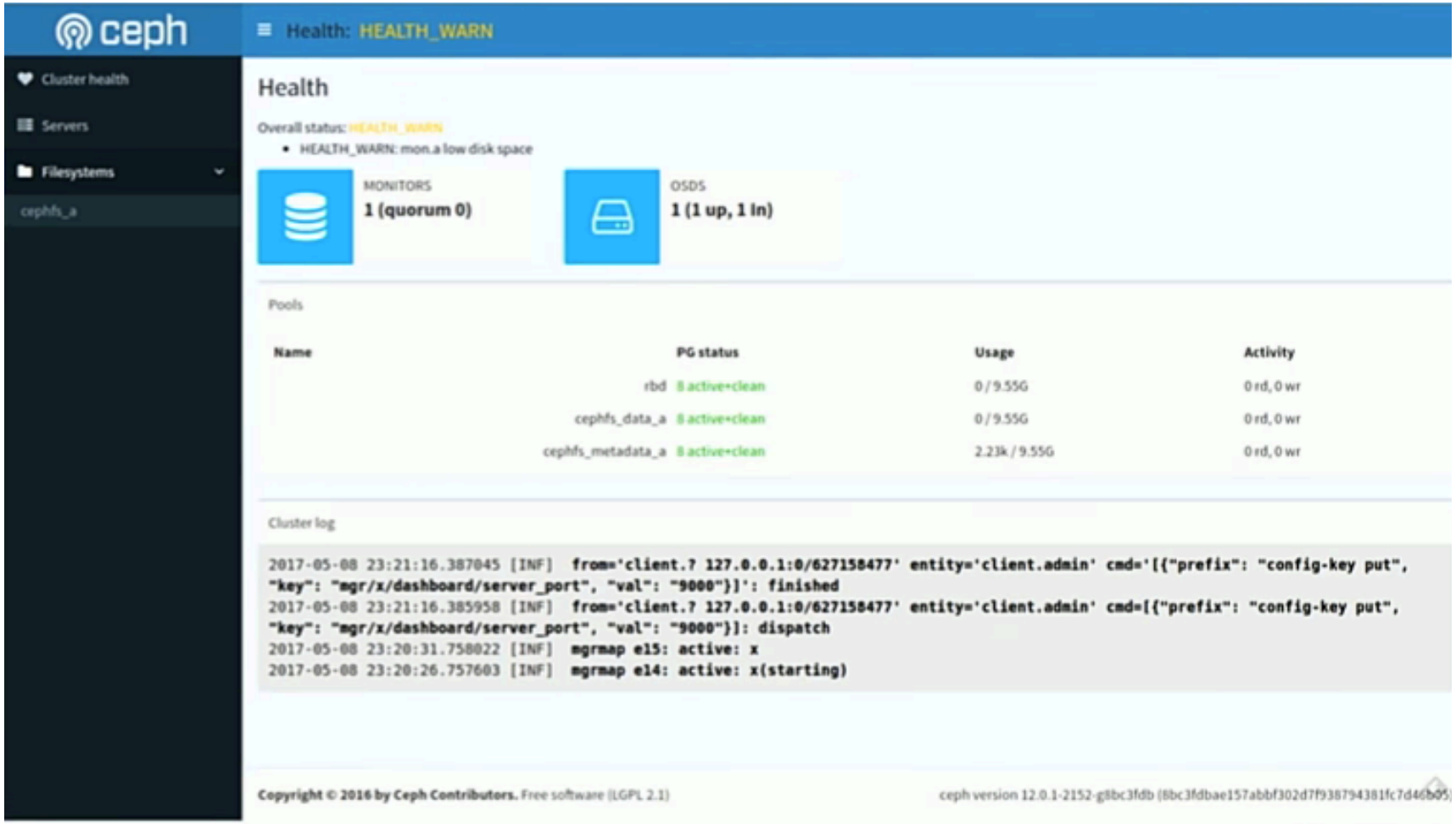
# Métrologie – Métriques



# Supervision - Dashboard

- Il existe plusieurs Dashboards pour CEPH :
  - Calamari (admin et supervision)
  - Inkscope (admin et supervision)
  - Ceph-dash
- La prochaine version « Luminous » introduit un Dashboard dans le service « mgr » :
  - Pour le moment un équivalent à « ceph status » web mais des améliorations à venir !

# Supervision - Dashboard



# Optimisations – Architectures

- Bonnes pratiques d'architectures :
  - Séparer les moniteurs et les serveurs OSD.
    - Les moniteurs font des « sync » régulièrement et dégradent les performances des OSD
      - Sur les anciennes version de CEPH (< v0.55)
      - Les nouvelles versions utilisent le syscall "syncfs" introduit dans Linux 2.6.39
  - Ne pas mélanger les hyperviseurs avec les serveurs qui sont responsable du stockage
- Attention aux « single point of failure » (SPOF) en fonction de la topologie :
  - Exemple : si toute l'infrastructure CEPH est dans une seule baie => redonder le commutateur

# Optimisations – Réseau

- Il est recommandé d'utiliser des réseaux 10gbit/s
  - pour le réseau de réplication
  - pour le réseau client du cluster
- Jumbo Frame (MTU: 9000)
  - Sur le réseau de réplication
  - Sur le réseau client à utiliser **en fonction de la faisabilité**
- Pour un petit cluster un réseau 1gbit/s fonctionne
  - Mais attention au temps de réplication en cas de mouvements des données

# Optimisations – Paramètres Kernel

- Dans le cas où vous avez beaucoup d'OSD par serveur, il est recommandé d'augmenter le nombre maximal possible de threads au maximum :
  - kernel.pid_max = 4194303
- Tunning réseaux (> 10 Gbit/s):
  - Tailles des buffers, …

# Optimisation – Système de fichiers

- Le système de fichiers utilisé par FileStore à une incidence importante sur les **performances** et la **fiabilité** de CEPH
- BTRFS a été pendant longtemps recommandé à partir du certaine version de Linux (en fonction de l'interprétation de la documentation) :
  - Ceux qui ont essayé ont eu pas mal de problèmes de fiabilité
- EXT4 est déconseillé à cause d'une limitation dans la taille des attributs étendus.
  - Pour l'utiliser, il faut ajouter à la configuration de CEPH:
    - osd max object name len = 256
    - osd max object namespace len = 64
  - Au LAPP-LAPTh, nous avons constaté une baisse de latence par rapport à XFS
- XFS est le système de fichiers le plus testé **et le seul système de fichiers recommandé**.

# Optimisations – BlueStore

- De meilleures performances :
  - Pour la version « Luminous » les développeurs annoncent des performances :
    - 2x plus rapide pour les disques durs
    - 1,5x plus rapide pour les SSD
- Enregistrent des checksums des données (crc32, xxhash…)
- Possibilité d'activer la compression (zlib, snappy)
  - Activable sur un pool ou globalement
- Technology Preview sur Kraken
- Sera utilisé par défaut à partir de Luminous
  - Les développeurs le considèrent stable

# Optimisations – Scheduler IO

- Il existe plusieurs scheduler IO dans le kernel Linux
  - **CFQ** : Met les demandes d'IO dans les files d'attente par processus et alloue des tranches de temps pour chaque file d'attente (défault)
  - **Deadline** : Définit des dates limites aux demandes d'IO et les met dans des files d'attente qui sont triées selon leurs dates limites.
  - **NOOP** : L'ordonnanceur NOOP fonctionne en plaçant toutes les requêtes d'E/S dans une simple file FIFO, et n'implémente que la fusion de requêtes. Il suppose que les performances d'E/S sont ou seront optimisées au niveau du périphérique de stockage ou du contrôleur de périphérique.
  - **MultiQueue** : Deadline, **Kyber** (Linux 4.12).

# Optimisations – Scheduler IO

- Les résultats varient en fonction :
  - du système de fichiers utilisé
  - de la technologie sous jacente (Contrôleur avec cache writeback ou JBOD)
  - du type de disque (disque dur, SSD, …)
  - de l'usage (mon, osd, …)
- **Il peut être intéressant de faire un benchmark avec la configuration que vous avez choisie pour déterminer le Scheduler le plus adapté**.

# Optimisations – Journal

- Les journaux des OSD peuvent être placés sur des disques différents des données.
- Sur les disques durs les premiers secteurs sont plus rapides que les derniers (placer le journal en tête)
- On peut également utiliser des disques rapides de type SSD
  - Il faut prendre des SSD « write intensive ».
  - 1 SSD tous les 3 à 4 OSD sinon c'est contre productif (en fonction du SSD)
- On peut mettre plusieurs journaux sur disque dédié au journaux
  - Attention en cas de problème avec un disque dans ce cas là plusieurs OSD sont affectés (RAID1 ? Coût…)
- La taille des journaux peut également influer sur les performances. Elle peut se calculer avec la formule suivante :
  - expected throughput = min(disque_throughtput)
  - osd journal size = {2 * (expected throughput * filestore max sync interval)}
  - Exemple avec un disque de 100 Mo / sec:
    - 2 * 100 * 5 => 1000

# Optimisations – Primary Affinity

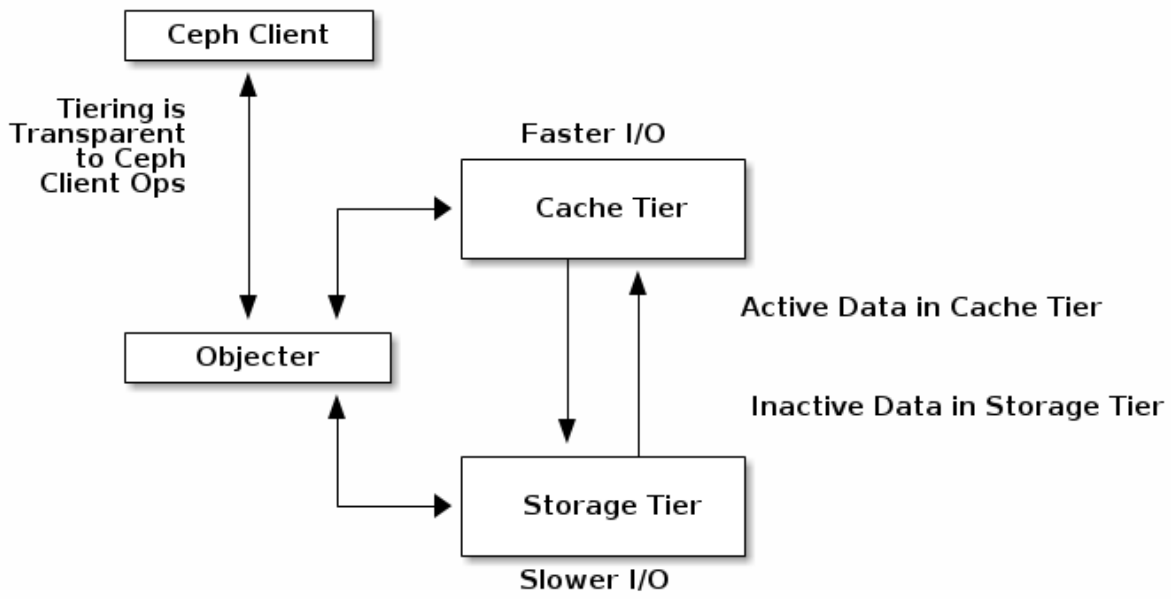
- C'est un paramétrage qui est utile dans le cas où le matériel est hétérogène
  - On peut envisager de l'utiliser aussi pour privilégier une localisation
- Il permet de privilégier les OSD qui ont les poids les plus fort dans le choix de l'OSD primaire.
- Les OSD primaires sont utilisés en premier dans la lecture des données.
- On peut à partir de benchmark des disques définir un paramétrage par OSD.

# Optimisations – Primary Affinity

- Il faut l'activer spécifiquement dans le ceph.conf
  - mon osd allow primary affinity = true
- Le paramètre ce définit par OSD avec un chiffre entre 0 et 1
  - attention 0 : l'OSD n'est utilisé primaire.
- La commande pour changer le primary affinity d'un OSD:
  - Exemple : ceph osd primary-affinit osd.0 0.55

# Optimisation – Cache tiering

- Le cache tiering utilise un pool de cache (hot) au dessus d'un pool de stockage (cold)
- C'est la seul solution pour faire de l'erasure coding pour du RBD
  - La prochaine version (Luminous) permet de faire du RBD sur de l'erasure coding



# Optimisation – Cache tiering

- Difficile de faire des benchmarks pour vérifier l'efficacité
- Les développeurs mettent en garde que tous les scénarios d'utilisation ne donnent pas de une bonne performance
  - Scénario qui fonctionne :
    - Rados Gateway où il y a beaucoup d'opérations de lecture sur des objets écrits récemment.
  - Scénarios où les performances sont discutées :
    - RBD avec de l'erasure coding (très demandé)
    - RBD en réplication
- L'empilement des couches est un risque supplémentaire à prendre en compte lors qu'on définit son architecture.

# Optimisations – Réplication

- Les réplications peuvent dégrader fortement les performances d'un cluster CEPH
- Il existe des paramètres qui permettent de limiter leurs effets :
  - osd max backfills : le nombre maximum de réplication (ou reconstruction) autorisé vers ou depuis un osd.
  - osd recovery max active : le nombre de réplication (ou reconstruction) actives par OSD en même temps (redémarrage des OSD nécessaire).
- Il est intéressant de connaitre les valeurs qu'on peut utiliser pour son cluster en fonction de l'urgence de la réplication et de la dégradation qu'on tolère et de définir des paramètres par défaut.
  - Pour un petit cluster avec peu d'OSD, on peut définir 1 aux deux paramètres.
- Le paramètre « osd_max_backfills » peut-être changé sans redémarrer le service de l'OSD
  - ceph tell osd.* injectargs '--osd_max_backfills=1'
  - ceph daemon osd.0 config set 'osd max backfills' 1

# Optimisations – Tunables

- Les nouvelles version de CEPH améliorent les algorithmes de placement des données
- Pour des raisons de compatibilité ascendante, les clients utilisant une vieille version de l'algorithme doivent pouvoir continuer à trouver les données.
- Il y a donc un paramètre pour choisir la version de l'algorithme utilisée par le cluster
- Ce paramètre se change quand tous les clients ont été mis à jour vers la nouvelle version
- Les « tunables » dérivent du nom des versions de CEPH : firefly, hammer, …
- Il y a plusieurs algorithmes pour le placement de données (straw, straw2, …)

# Optimisations – QOS

- Actuellement il n'y a pas de fonctionnalités dans CEPH qui permettent de limiter les IO au niveau des clients
  - Prévu dans une release future (**M**imics)
- QEMU/KVM permet de définir des limites :
  - Limite de bande passante en lecture et/ou écriture (bytes/sec)
  - Limite en terme d'IO (iops/sec)
- Il peut être intéressant en terme de qualité de service de définir des classes de disques qui permettent de garantir qu'une machine ne va pas être responsable d'un déni de service.

# Bibliographie


- Documentation CEPH :
  - Officielle : http://docs.ceph.com/docs/master/
  - Red Hat CEPH Storage : https://access.redhat.com/documentation/en/red-hat-ceph-storage/
- Nouveautés de Luminous et feuille de route : https://www.youtube.com/watch?v=NqOFWGUvA9A
- Scheduler IO:
  - http://ceph.com/community/ceph-bobtail-performance-io-scheduler-comparison/