Cuando un OSD se corrompe por ruptura del disco

1. Usar los siguientes comandos para saber como esta configurado el arbol de OSD y los pool que hay en cada uno.

```
ceph osd tree
ceph pg dump
```

Es importante ver que no este comprometido el ratio de replicacion porque de estarlo se debe incrementar la capacidad lo antes posible.

1a. Luego de detectar el que se desea eliminar se debe quitarle el peso que tiene de almacenaje.

```
ceph osd crush reweight osd.7 0
```

2. Sacarlo de la lista de OSD activos. osd out {osd-num}

```
ceph osd out 7
```

3. Esperar que el proceso de reorganizacion del cluster termine. Se puede monitorear con el comando 'ceph pg stat'. El proceso acaba cuando todo est· 'active+clean'

4. Detener los OSD y desmontarlo del sistema de archivos: stop ceph-osd id={osd-num}

```
stop ceph-osd id={osd-num}
ceph-osd stop 7
umount /var/lib/ceph/osd/ceph-7
```

5. NUEVO Eliminar el OSD del mapa CRSUH: ceph osd crush remove {name}

```
ceph osd crush remove osd.7
```

6. Eliminar las llaves de autentificacion: ceph auth del osd.{osd-num}

```
ceph auth del osd.7
```

7. Eliminar el OSD del mapa de OSD's: ceph osd rm {osd-num}

```
ceph osd rm 7
```

8. En este punto ya todo está como si el OSD no hubiera existido. Se puede proceder a crear el nuevo.

Addicionando un nuevo espacio

1 Crear el OSD logicamente

```
ceph osd create
```

2 Crear el punto donde se va a montar el OSD

```
mkdir /var/lib/ceph/osd/ceph-7
```

3 Formatear el OSD (Se asume que ya esta la particion creada. De no estarlo se recomienda hacerlo con el parted)

```
mkfs -t xfs /dev/sdd1
```

4 Montar la particion en el lugar asignado

```
mount -o rw,relatime,inode64,logbsize=256k,delaylog,allocsize=4M /dev/sdd1
/var/lib/ceph/osd/ceph-7
```

5 Crear las llaves de autentificacion

```
ceph-osd -i 7 --mkfs --mkkey
```

6 Adicionar las llaves y las reglas de acceso del OSD creado

```
ceph auth add osd.7 osd 'allow *' mon 'allow rwx' -i
/var/lib/ceph/osd/ceph-7/keyring
```

7 Adicionar el OSD al mapa del CRUSH: ceph osd crush add {id-or-name} {weight} [{bucket-type}={bucket-name} ...]

```
ceph osd crush add osd.7 1.45 host=compute1
```

8 Iniciar el proceso del OSD

```
start ceph-osd id=7
```

9 Comprobar que se halla integrado al esquema de replicacion y que se este replicando todo.

```
ceph -w
```

ref: https://blueprints.launchpad.net/fuel/+spec/ceph-osd-remove
http://docs.ceph.com/docs/master/rados/operations/add-or-rm-osds/

Exportar el mapa CRUSH

```
ceph osd getcrushmap -o salida
crushtool -d salida -o salida.decompile
```

Asignado un grupo de discos a un pool

Normalmente cuando se adiciona OSD a una estrucutra ya en funcionamiento se ve asi:

```
$ ceph osd tree
# id    weight  type name       up/down reweight
-1      0.18    root default
-2      0.03            host ceph1
0       0.03                    osd.0   up      1
3       0.03                    osd.3   up      1
-3      0.03            host ceph2
1       0.03                    osd.1   up      1
4       0.03                    osd.4   up      1
-4      0.03            host ceph3
2       0.03                    osd.2   up      1
5       0.03                    osd.5   up      1
```

Pero si se tienen discos tipo SSD y discos SATA es comprensible que se deseen usar en casos separados. Para eso deben estar en pools distintos. Una ayuda muy clara se puede ver en: http://www.sebastien-han.fr/blog/2014/08/25/ceph-mix-sata-and-ssd-within-the-same-box/

Basicamente se debe: 1. Exportar el CRUSH map actual

ceph osd getcrushmap -o salida crushtool -d salida -o salida.decompile

Se va a ver que todos los osd estan dentro de los 3 hosts

2. Editarlo

La idea es crear 3 nuevos host “virtuales” que incluyan cada uno los tipos de discos que se desean separar y luego poner estos osd bajo un root diferente. En el caso de hacer esto en una plataforma que ya este trabajando lo mejor es no modificar el nombre del root default ni de los pools que estan creados bajo el y adicionar los nuevos discos/osd/root/pools en una estrucutra nueva que se pueda asociar desde cero.

OSD SATA DECLARATION ## host ceph1-sata {

```
id -2   # do not change unnecessarily
# weight 0.000
alg straw
hash 0  # rjenkins1
item osd.0 weight 1.000
```

} host ceph2-sata {

```
id -3   # do not change unnecessarily
# weight 0.000
alg straw
hash 0  # rjenkins1
item osd.2 weight 1.000
```

} host ceph3-sata {

```
id -4   # do not change unnecessarily
# weight 0.000
alg straw
hash 0  # rjenkins1
item osd.1 weight 1.000
```

}

OSD SSD DECLARATION

host ceph1-ssd {

```
id -22    # do not change unnecessarily
# weight 0.000
alg straw
hash 0  # rjenkins1
item osd.3 weight 1.000
```

} host ceph2-ssd {

```
id -23    # do not change unnecessarily
# weight 0.000
alg straw
hash 0  # rjenkins1
item osd.4 weight 1.000
```

} host ceph3-ssd {

```
id -24    # do not change unnecessarily
# weight 0.000
alg straw
hash 0  # rjenkins1
item osd.5 weight 1.000
```

}

SATA ROOT DECLARATION

root sata {

```
id -1   # do not change unnecessarily
# weight 0.000
alg straw
hash 0  # rjenkins1
item ceph1-sata weight 1.000
item ceph2-sata weight 1.000
item ceph3-sata weight 1.000
```

}

SATA ROOT DECLARATION

root ssd {

```
id -21    # do not change unnecessarily
# weight 0.000
alg straw
hash 0  # rjenkins1
item ceph1-ssd weight 1.000
item ceph2-ssd weight 1.000
item ceph3-ssd weight 1.000
```

} a ## # SSD RULE DECLARATION ##

rules rule ssd { ruleset 0 type replicated min_size 1 max_size 10 step take ssd step chooseleaf firstn 0 type host step emit }

SATA RULE DECLARATION

rule sata { ruleset 1 type replicated min_size 1 max_size 10 step take sata step chooseleaf firstn 0 type host step emit }

3. Reinsertarlo

$ crushtool -c salida.decompiled -o salida2.compiled $ sudo ceph osd setcrushmap -i salida2.compiled

4.comprobar que el arbol este dividido en 2.

ceph@ceph-admin:~$ ceph osd tree # id weight type name up/down reweight -21 0.09 root sata -22 0.03 host ceph1-sata 3 0.03 osd.3 up 1 -23 0.03 host ceph2-sata 4 0.03 osd.4 up 1 -24 0.03 host ceph3-sata 5 0.03 osd.5 up 1 -1 0.09 root ssd -2 0.03 host ceph1-ssd 0 0.03 osd.0 up 1 -3 0.03 host ceph2-ssd 1 0.03 osd.1 up 1 -4 0.03 host ceph3-ssd 2 0.03 osd.2 up 1

5. Creacion de los pools $ceph osd pool create ssd 128 128 pool 'ssd' created

$ceph osd pool create sata 128 128 pool 'sata' created

6. Assignar reglas a los pools. Los valores de las reglas son los definidos anteriormente en el CRUSH map.

$ceph osd pool set ssd crush_ruleset 0 set pool 8 crush_ruleset to 0

$ceph osd pool set sata crush_ruleset 1 set pool 9 crush_ruleset to 1

7. Comprobar que los nuevos pools se hallan creado y que esten asociados con las reglas adecuadas. ceph osd dump

$ ceph osd dump (fijarse en las lineas que dicen pool 3 y pool 4)

epoch 35 fsid 531b4820-2257-4f3b-b12b-e1f6827ecce5 created 2016-06-02 12:45:52.763566 modified 2016-06-02 16:08:00.149899 flags pool 0 'data' replicated size 2 min_size 1 crush_ruleset 0 object_hash rjenkins pg_num 64 pgp_num 64 last_change 1 flags hashpspool crash_replay_interval 45 stripe_width 0 pool 1 'metadata' replicated size 2 min_size 1 crush_ruleset 0 object_hash rjenkins pg_num 64 pgp_num 64 last_change 1 flags hashpspool stripe_width 0 pool 2 'rbd' replicated size 2 min_size 1 crush_ruleset 0 object_hash rjenkins pg_num 64 pgp_num 64 last_change 1 flags hashpspool stripe_width 0 pool 3 'ssd' replicated size 2 min_size 1 crush_ruleset 0 object_hash rjenkins pg_num 128 pgp_num 128 last_change 33 flags hashpspool stripe_width 0 pool 4 'sata' replicated size 2 min_size 1 crush_ruleset 1 object_hash rjenkins pg_num 128 pgp_num 128 last_change 34 flags hashpspool stripe_width 0 max_osd 6 osd.0 up in weight 1 up_from 4 up_thru 31 down_at 0 last_clean_interval [0,0) 10.12.1.151:6800/2927 10.12.253.50:6800/2927 10.12.253.50:6801/2927 10.12.1.151:6801/2927 exists,up c21dd057-11af-44a9-a84c-8ef8db73bc7d osd.1 up in weight 1 up_from 8 up_thru 31 down_at 0 last_clean_interval [0,0) 10.12.1.152:6800/2823 10.12.253.51:6800/2823 10.12.253.51:6801/2823 10.12.1.152:6801/2823 exists,up 6b224e75-e552-4b04-ab4a-fa7764ebfaa7 osd.2 up in weight 1 up_from 12 up_thru 31 down_at 0 last_clean_interval [0,0) 10.12.1.153:6800/5369 10.12.253.52:6800/5369 10.12.253.52:6801/5369 10.12.1.153:6801/5369 exists,up 4a105b3c-6628-47a5-9eeb-922be576ee55 osd.3 up in weight 1 up_from 16 up_thru 34 down_at 0 last_clean_interval [0,0) 10.12.1.151:6803/4036 10.12.253.50:6802/4036 10.12.253.50:6803/4036 10.12.1.151:6804/4036 exists,up 369d2b64-fd96-4ec9-a956-acc2f0c47cdd osd.4 up in weight 1 up_from 20 up_thru 34 down_at 0 last_clean_interval [0,0) 10.12.1.152:6803/3890 10.12.253.51:6802/3890 10.12.253.51:6803/3890 10.12.1.152:6804/3890 exists,up 649fc093-6dff-403c-928f-2e07f2d02e63 osd.5 up in weight 1 up_from 24 up_thru 34 down_at 0 last_clean_interval [0,0) 10.12.1.153:6803/6458 10.12.253.52:6802/6458 10.12.253.52:6803/6458 10.12.1.153:6804/6458 exists,up dbef13f9-4fc4-4ed6-aa35-fd757dfe17ab

8. Para saber si todo funciona bien y si solo se estan usando los discos adecuados en un sistema que no se este usando se puede crear un block-device y montarlo en algun cliente. Usarlo mientras se comprueba con comandos atop pro ejemplo que discos se usan y cuales no. El resultado debe ser que solo se usen o los discos SATA o los discos SSD segun el pool que se use.

TODO: Estudiar con mas detalle una configuracion mas granular en el CRUSH map. Ver los step y los type