Café ARAMIS - ZFS

- Filesystem
- Illimité ou presque
- Facile
- Extensible
- Multi OS

Simple, sur

- Pensé pour être simple, facile d'utilisation, directement opérationnel.
- `zpool create monpool raidz2 mfid0pd01 mfid0pd02 mfid0pd03 raidz2 mfid0pd04 mfid0pd05 mfid0pd06 log mirror mfid0pd11 mfid0pd12 cache mfid0pd21 spare mfid0pd07`
- `zfs create monpool/mongrosespace`

  => fs monté sur /monpool/mongrosespace
- Fait pour des volumes conséquents.
- Beaucoup de mémoire, un peu lent en nfs (lié a l'assurance de l'ecriture et de la conformité (checksum) sur disque)

Qu'est-ce que Z-Filesystem

- A la fois gestionnaire de pool volume manager / Filesystem
- RAID =>
  - configuration figée
  - Carte donc passer par le hard pour reparer/changer DD
  - Log ?
  - Un seul device

- ZFS => des Disks et le systeme gère tout.
  - L'espace peut-etre augmenté (si miroir 1To + 2To = 1To, changer 1To en 2To donne 2To)
  - Les filesystem partagent le même pool mais avec leurs propriètés
  - Tout l'espace est disponible pour tous les FS.

Caractéristiques des pools

- Commande unique avec plein d'options : `zpool`
- Creation , liste, état, destruction, réparation, remplacement DD, stats d'utilisations des DD, upgrade :
  - Create, list, status, destroy, scrub,
  - detach, replace, iostats, … , history
- Virtual Devices :
  - Disk, file, mirror, raidz[123], spare, cache, log
- On peut ajouter des VDEV à pool mais pas des disques à un vdev !!!!
- Attention si vdev non redondant dans un pool = securité compromise pour tout le pool

- Property of a pool :

- Possibilité d'exporter un pool => permet de re-cupérer ce pool en réassemblant suffisamment de disques

Choisir ?

- Depend :
  - Nombre de voies controleur disque (cf 6*8 Thumper SUN)
  - Depend log, cache, spare, niveau de raid

    `(un rpool, un datapool ?)`
  - Si carte raid => faire n raid0 (1 par disque) et laisser zfs gerer les disques et les reparer (soft contre hard). Choisir plutot JBOD.
  - RAIDZ1 = perte possible de 1 Disque sur le Vdev
  - RAIDZ2 = perte possible de 2 Disque sur le Vdev ……. 3
  - Si plusieurs controleurs => mixer les disques.
  - Plus de vdev + de perf mais perte de place si sécu.
  - Taille de disques dans un pool peuvent etre differentes => ecriture en % de la taille. (moins striped!)
  - Utiliser des nommages de disque persistants.

Administration

- Remplacement de disque sur machine(hotswap) :
  - `Zpool replace pool dd1 dd2`
  - `Zpool detach pool dd1`

    ou `zpool offline dd1`

  - `Zpool status -x`
  - `Zpool scrub pool` (une fois par mois?)
  - `Zpool upgrade` (-v) (28 freebsd, 32 Solaris10, 3 ? Solaris11)

Les FS

- Se crée sur un pool (crée un dataset) :

  zfs create -o "compress=lz4" monpool/monfs

- df -h :

- Un dataset peut etre utilisé comme n'importe quel Filesystem + Propriètés :

zfs

- zfs get all monpool/monfs :

Propriètés clés

- Mountpoint
-
- Partage nfs :
-
- Compression
-
- Dedup (attention a la memoire)
-
- Quota

- Acl, xattr,

Commandes zfs

- `zfs rename monpool/test monpool/prod`
- `zfs list -t [all, filesystem,snapshot]`
-

Snapshots

- Fait partie du FS (nfs)
- READONLY
- Représente le FS à un instant t
- Instantané (ou presque) fige les inodes :
  - Ce qui change va etre « dupliqué »
  - La taille d'un fs à un instant t+1 =

    snap instant t + ce qui a bougé (donc si destruction d'un fichier qui existait dans le snap => la taille utilisée ne bouge pas)
- Se copie au niveau block (send / recv)

Snapshots

- Facile :

```
zfs snapshot (-r) monpool@_aujourdhui
zfs list -rt snap monpool
zfs snapshot (-r) monpool@demain
   /monpool/monfs/.zfs/snapshots/_aujourdhui
   /monpool/monfs/.zfs/snapshots/_demain
```

- `zfs diff  monpool@_aujourdhui monpool@demain`
- Mettre des dates sinon on sait plus !

Dupliquer / archiver

- Archive dans un fichier :

```
zfs send mypool@_backup1 > /backup/backup1
```

- Duplication de FS :

```
zfs send (-v) mypool@_acopier | zfs recv backup/copie
```

- Distant :

```
zfs snapshot -r mypool/monFS@sauvegarde_1207
```

```
zfs send -v -R -i mypool/monFS@sauve_1206
mypool/monFS@sauve_1207 | ssh replica zfs recv -v mypoolsauve/
```

- Il y a des options subtiles => lire et comprendre la doc ! :
- send : -i, -I, -R ; receive -d -e -F -u -n

Recupération de données

- Rollback :

```
zfs rollback monpool/prod/test@_hier
```

  - Ne peut restaurer que le dernier (sinon -r) mais perte des snap intermédiaires (les snap font partie du FS on vous dit!)

- cp /mesusers/toto/.zfs/snapshot/_hier/monfichier_super_important

  /mesusers/toto/monfichier_retrouvé

  Ca fait partie du fs donc avec les droits de l'utilisateur !

- (zfs set copies=2 monpool/a_pas_perdre)

Clones

- = Snapshots mais read/write
- Basé sur un snap d'un fs

```
zfs clone mesusers/toto@_auj mesusers/tata
```

- => on ne peut detruire le snap tant qu'il a des clones.

```
zfs get origin mesusers/tata
```

- On peut transformer un clone en vrai fs (le détacher du snap initial :

```
zfs promote mesusers/tata
```

Divers

- Acl_nfsv4 sous freebsd pas acl posix.
- C'est lent de décompresser un noyau linux sur NFS => Zil (log) en SSD - EN MIROIR !
- Ne pas supprimer !!
- Arc = cache fs = ca peut améliorer sur la lecture (ssd – Read intensive ?)
-

Questions Diverses