Retour d'expérience ZFS dans un laboratoire CNRS

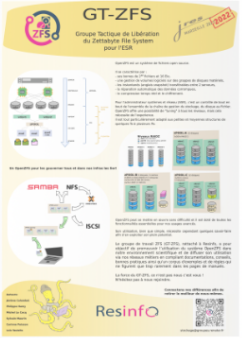
Jérôme COLOMBET

https://homepages.lcc-toulouse.fr/colombet/capitoul_retour_exp_zfs.pdf

29 février 2024

Le GT-ZFS

- **G**roupe de **T**ravail ZFS, rattaché à Resinfo ⇒ son objectif, promouvoir le ZFS dans l'ESR
- 11 membres, répartis sur le territoire avec les infrastructures et les contraintes du terrain
- Le groupe de travail ZFS a pour objectif de fournir :
  - une documentation technique :
    `https://resinfo-gt.pages.in2p3.fr/zfs/doc/index.html`
  - des conseils, des bonnes pratiques, des scripts,
  - des tutoriels, des démonstrations et des formations ...
- Rassembler les usagers autour de la liste : stockage@groupes.renater.fr
- Les communications du GT-ZFS :
  - 2021.10 - Réseau Respire : Retour d'expérience ZFS
  - 2021.12 - JRES 2022 - Marseille : Poster
  - 2023.11 - JoSy ZFS - Lyon
  - 2024.02 - Réseau Capitoul : Retour d'expérience ZFS
  - 2024.12 - JRES 2024 - Rennes : Une démo
  - 2025.xx - Jtech ZFS



L'entité LCC - chimie Toulousaine

- **L**aboratoire de **C**himie de **C**oordination (chimie métaux de transition)
- Proche de la nouvelle attraction touristique Téléo ; Téléphérique Urbain Toulouse
- Le LCC **UPR 8241**, installé sur un campus propre CNRS 205
  - 17 équipes de recherche autour des axes Catalyse, Matériaux et Santé
  - 18 services scientifiques et administratifs en soutien
  - C'est 3 informaticiens pour **280** personnes et un bâtiment de 11000m^2
  - **R**essources **I**nformatiques et **C**alcul **S**cientifique
- Grands équipements techniques **RMN**, **RX**, Spectrométrie IR, Microscopie Electronique
- Volume de données exponentielles, critique et reparti

Rappel du contexte technique

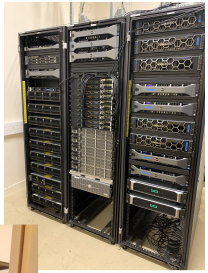


Figure – DC 2013 - 2024

- 3 salles serveurs reparties sur le campus 205 (LCC/IPBS)
- Réseau 10G sur l'ensemble des bâtiments, 600 prises
- Parc utilisateurs multi-os (400 postes)
- Parc scientifique vieux, hétérogène et critique (50 postes)
- 3 clusters hyperviseurs Proxmox VE 8 full ZFS.
- 1 cluster HPC-OAR (Centos 6)
- 1 cluster HPC-SLURM (Almalinux 9)
- 3 stockages centralisés ZFS (production, backup, froid)
- Volumétrie consolidée de **0,5 Po**
  - Production 122To + 38To (bureautique + HPC)
  - Backup 175To, Froid 55To
  - Proxmox 50To, PBS 20To

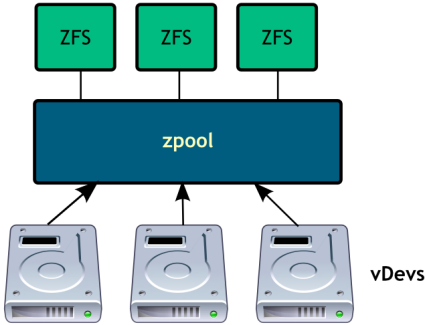
ZFS : Zetta byte File System

- Développé par Sun Microsystems (2004)
- Introduit dans Solaris 10 (2005), licence CDDL = incompatible avec la GNU GPL
- Porté sur Mac OS X, Linux, FreeBSD,...
- Fonctionnalités
  - Snapshots
  - Clones
  - Quotas et réservation d'espace
  - Compression
  - Dé-duplication
  - Export/Import
  - Chiffrement intégré

- Pas de limites
  - 16 Eo : 16 millions de To max pour des volumes et des fichiers
  - 2^{48} fichiers, soit plus de 280 millions de milliards / volume
- Garantir la sécurité des données (intégrité, disponibilité)
- Administration simplifiée (2 commandes)
- Autorisations POSIX (comme ext4 et Btrfs) et compatible ACL (NFSv4).
- Taille des blocs ajustable (défaut : 128 ko)
- Indépendant du matérielle (HBA)
- Intégré à des solutions (TrueNas, Nexenta,...)

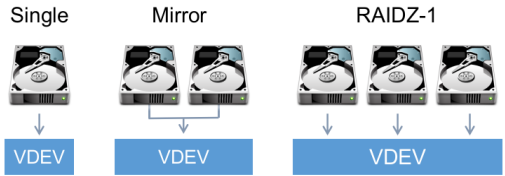
Pool, Fs, Raid-z, DRaid-z

Un pool est un ensemble de périphériques qui fournissent de l'espace pour le stockage et la duplication des données.



zpool est construit à partir de périphériques virtuels **vdevs**, c'est l'unité de base de stockage de données

- disques : entiers ou juste une partition
- fichiers dans un autre système de fichiers
- miroirs disques, partitions ou fichiers
- raidzX : plusieurs disques vs raid5
- draidX : plusieurs disques avec spare logique



Les 2 commandes à connaître

zpool pour la gestion des pools

- Création (create)
- Destruction (destroy)
- Import/Export
- Ajout de stockage (add)
- Visualisation état, performances (iostat)

zfs pour la gestion des systèmes de fichiers

- Création/destruction
- Montage
- Gestion des attributs (quotas, compression, ...)
- Snapshots/Clones
- Sauvegardes

Exemple : installation sur debian

dkms

```
## backports
deb http://deb.debian.org/debian xxxxx-backports main contrib non-free
deb-src http://deb.debian.org/debian xxxxx-backports main contrib non-free
```

dkms

```
$ apt install -t xxxxx-backports dkms spl-dkms -y
$ apt install -t xxxxx-backports zfs-dkms zfsutils-linux -y
```

lister vos disques physiques

```
$ ls -lh /dev/disk/by-id/
scsi-35000039ca82b03b9 -> ../../sdb
scsi-35000039ca82b00f5 -> ../../sde
```

Exemple : creation d'un pool

raidz + logs + cache

```
zpool create hpool raidz scsi-d1 scsi-d2 ... spare scsi-d3 logs/cache ...
```

raidz + group

```
zpool create hpool raidz scsi-d1 scsi-d2 ... raidz scsi-d3 scsi-d4 ...
```

draid - parity,data,children

```
zpool create hpool draid2:3d:1s:6c scsi-d1 scsi-d2 scsi-d3 scsi-d4 ...
```

miroir

```
zpool create rpool mirror scsi-disk1 scsi-disk2
```

Exemple : création de systèmes de fichiers

Création d'un home directory

```
$ zfs create hpool/home
```

Suppression d'un home directory

```
$ zfs destroy hpool/home
```

Définition d'un point de montage

```
$ zfs set mountpoint=/home hpool/home
$ zfs get mountpoint hpool
 hpool mountpoint /hpool default
```

Exemple : quotas

Le quotas permet de limiter la quantité d'espace disque.

Définition d'un quota

```
$ zfs set quota=1T hpool/home
```

Visualisation d'un quota

```
$ zfs get quota hpool/home
hpool/home quota 1T local
```

Suppression d'un quota

```
$ zfs set quota=none hpool/home
```

Quota pour un utilisateur

```
$ zfs set userquota@colombet=400G hpool/home
```

Exemple : snapshots

Pour rappel, les snapshots ne sont pas considérés comme des sauvegardes, car il ne s'agit pas de copies.

Création d'un snapshot

```
$ zfs snapshot -r hpool/home@zfs-auto-snap-$(date +%Y-%m-%d-%H%M%S)
$ zfs list -H -o name -t snapshot
hpool/home@zfs-auto-snap-2024-02-22-070000

$ ls -l .zfs/snapshot/
```

Restauration d'un snapshot

```
$ zfs rollback hpool/home@zfs-auto-snap-2024-02-22-070000
```

Exemple : exportation d'un pool

Utile pour la gestion des supports amovibles ou en HA via @IP flottante

Exporter un pool

```
$ zpool export hpool
```

Importer un pool

```
$ zpool import hpool
```

Renommer un pool

```
$ zpool import hpool mypool
$ zpool status
$ zpool export mypool
```

Exemple : externaliser une copie ZFS manuellement via SSH

- disposer de trois copies de vos données
- stocker ces copies sur deux supports différents
- conserver une copie de la sauvegarde hors site

Externaliser un snapshot via zfs send over ssh

```
$ zfs snapshot -r localpool/home@snapshot-new
$ zfs list -H -o name -t snapshot
localpool/home@snapshot-old
localpool/home@snapshot-new
$ zfs send -i localpool/home@snapshot-old localpool/home@snapshot-new| pv
-b | ssh myserver "mbuffer -m 1G | zfs receive distantpool/home -F"
```

Le ZFS au LCC

LCC : Proxmox - PVESR Storage Replication

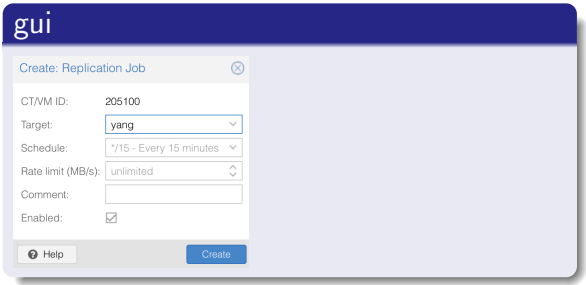
Gestion DR14 - Campus 205 - LCC + IPBS Synchronisation des VMs GTB / CA

cli

```
$ pvesr create-local-job 205100-0 yin
$ pvesr list
JobID Target Schedule Rate Enabled
205099-0 local/yang 21:00 - yes
205100-0 local/yang */15 - yes
205101-0 local/yin */20 - yes

$ pvesr delete 205100-0 --force
```

gui



En…	Guest	J…	Target	Status ↑	Last Sync	Dur…	Next S…	Sched…	Com
✔	205099	0	yang	✔ OK	2024-02-27 21:00:03	20m…	2024-…	21:00	
✔	205100	0	yang	✔ OK	2024-02-28 09:45:00	14.9s	2024-…	*/15	
✔	205101	0	yin	✔ OK	2024-02-28 09:40:04	16.8s	2024-…	*/20	

🔗: https://pve.proxmox.com/pve-docs/chapter-pvesr.html

LCC : Proxmox - ZFS over iSCSI

sur stockage distant

créer un dataset dans le pool tank

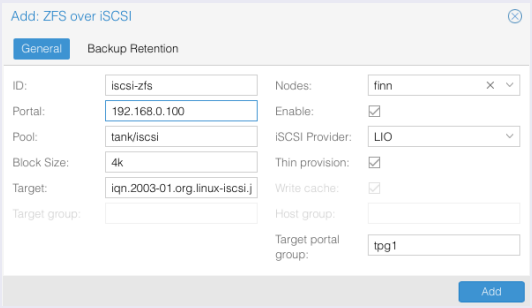
```
$ zfs create tank/iscsi
```

créer une target et les acls

```
$ apt install targetcli-fb -y
$ targetcli
/> cd iscsi
/> create
Created target
iqn.2003-01.org.linux-iscsi.nas.x8664:sn.2c0c3e76710e
/> cd
iqn.2003-01.org.linux-iscsi.nas.x8664:sn.2c0c3e76710e/tpg1/acls
/> create iqn.1993-08.org.debian:01:1ae0ad6ebb5f
```

sur cluster proxmox

```
$ cat /etc/iscsi/initiatorname.iscsi
iqn.1993-08.org.debian:01:1ae0ad6ebb5f
$ ssh-keygen -f /etc/pve/priv/zfs/192.168.0.100_id_rsa
$ ssh-copy-id -i /etc/pve/priv/zfs/192.168.0.100_id_rsa.pub
root@192.168.0.100
```



🔗: https://homepages.lcc-toulouse.fr/colombet/debian-nas-san-open-source-avec-support-zfs/

LCC : SaMBa4 - restaurer une versions précédentes

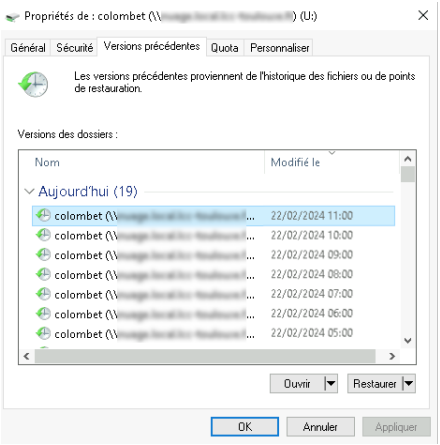
cron snapshots toutes les heures

```
$ zfs snapshot -r hpool/home@zfs-auto-snap-$(date
+%Y-%m-%d-%H%M%S)
$ zfs list -H -o name -t snapshot
hpool/home@zfs-auto-snap-2021-01-05-070000
$ zfs list -H -o name -t snapshot
```

/etc/samba/smb.conf ⇒ shadowcopy

```
shadow: snapdir = .zfs/snapshot
shadow: sort = desc
shadow: format = -%Y-%m-%d-%H%M%S
shadow: snapprefix = ^zfs-auto-snap
shadow: delimiter = -20

vfs objects = shadow_copy2
```



LCC : SaMBa4 - gestion des quotas

quota zfs + quota utilisateur colombet

```
$ zfs get quota hpool/home
hpool/home quota 15T local


$ zfs get -H userused@colombet hpool/home
hpool/home userused@colombet 367G local


$ zfs get -H userquota@colombet hpool/home
hpool/home userquota@colombet 400G local
```

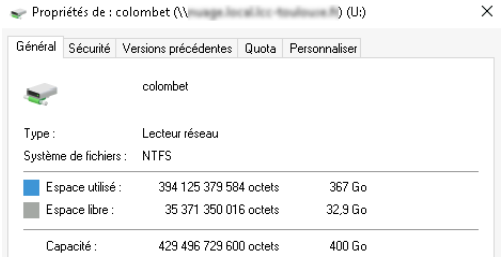
/etc/samba/smb.conf ⇒ script bash quota zfs

```
get quota command = /opt/scripts/samba_quotazfs.sh
```

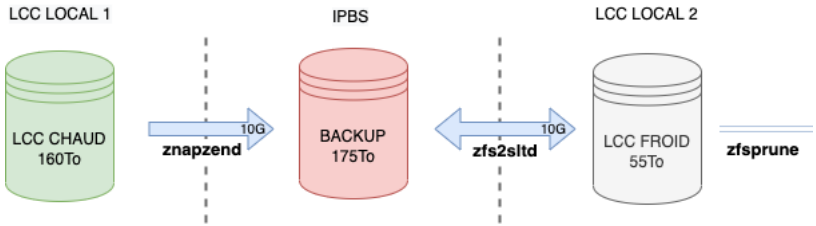
```
#!/bin/bash
#username=$1

smbpath=${PWD}
username="${smbpath:6:${#smbpath}-1}"

if [ ! -z  "$username" ]; then
  smbpath=${PWD}
  dataset=`/bin/df -l ${smbpath} | /usr/bin/tail -n 1 | /usr/bin/awk '{ print $1 };'`
  infoused=`/sbin/zfs get -Hp userused@$username $dataset`
  infoquota=`/sbin/zfs get -Hp userquota@$username $dataset`
  usedbytes=`echo ${infoused}| /usr/bin/awk '{ printf "%.f", $3/1024 };';`
  quotabytes=`echo ${infoquota}| /usr/bin/awk '{ if ( $3 == "none" ) { print "0"} else { printf "%.f", $3/1024 }  };'`
  echo 2 $usedbytes $quotabytes $usedbytes $quotabytes $quotabytes
  #info=`/sbin/zfs userspace -Hpo name,used,quota $dataset | /usr/bin/grep -i ${username}`
  #info=`/bin/more /tmp/quotazfs-home | /usr/bin/grep -i ${username}`
  #usedbytes=`echo ${info}| /usr/bin/awk '{ printf "%.f", $2/1024 };';`
fi
exit
```



LCC : Stockage - Gestion du cycle des données (ZnapZend et zfs2sltd)



lcc chaud

```
# znapzendzetup create
--mbuffer=/usr/bin/mbuffer
--mbuffersize=1G
--tsformat=zfs-auto-snap-%Y-%m-%d-%H%M%S
SRC '7d=>1h' hpool/home DST '15d=>1h'
myuser@x.x.x.x:backup/home

# znapzendzetup list
```

lcc backup

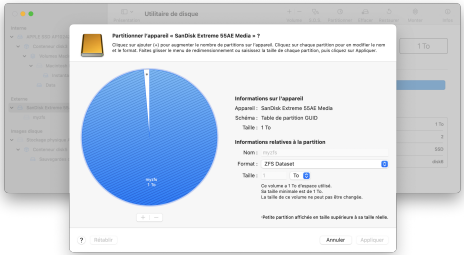
```
# zfs2sltd -h myuser@x.x.x.x -p 22 -s
backup/home -d froid/home -f
zfs-auto-snap -t sync
```

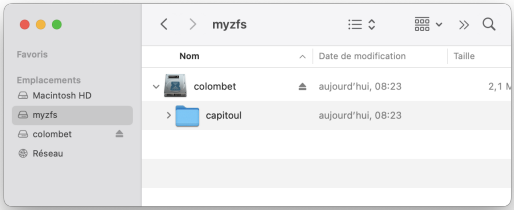
lcc froid

```
# zfs-prune-snapshots 1Y froid
```

🔗: ZnapZend (backup with mbuffer and ssh support) : https://github.com/oetiker/znapzend/
🔗: ZFS Simple Script Local To Distant (zfs2sltd) : https://gitlab.in2p3.fr/jerome.colombet/
🔗: ZFS Prune Snapshots : https://github.com/bahamas10/zfs-prune-snapshots

LCC : et nos macOS (brew install openzfs)






```
$ ls -all /dev/disk*
$ zpool create myzfs disk4
$ zfs create -o encryption=on -o keylocation=prompt -o
keyformat=passphrase myzfs/colombet
$ zfs umount myzfs/colombet
$ zfs export myzfs
```

```
$ zfs import myzfs
$ zfs load-key -r myzfs/colombet
Enter passphrase for 'myzfs/colombet':
1 / 1 key(s) successfully loaded
$ zfs mount myzfs/colombet
```

🔗: `https://openzfsonosx.org`

Pour conclure : ZFS est un système de stockage pour l'ESR

- Il est à l'aise dans des petites comme dans des grandes structures
- Il est simple à administrer (zpool & zfs)
- Il est performant et compatible avec les constructeurs de Matinfo5
- C'est une technologie mûre, en pleine expansion et à promouvoir dans l'ESR
- Il faut respecter les pré-requis
- Oui il a de la concurrence mais c'est un autre sujet : ext4, btrfs, ceph ...
- Il faut penser au GT-ZFS :
  - `https://resinfo-gt.pages.in2p3.fr/zfs/doc/index.html`
  - stockage@groupes.renater.fr

Questions

: jerome.colombet@lcc-toulouse.fr
: https://homepages.lcc-toulouse.fr/colombet/
: https://gitlab.in2p3.fr/jerome.colombet/