[zfs, libvirt]

ZFS

```
apt install zfsutils-linux
```

```
sudo zfs list
```

create filesystem 'lab' on 'LXD' pool

```
sudo zfs create LXD/lab
sudo zfs set mountpoint=/lab LXD/lab
```

```
sudo zfs destroy LXD/lab
```

create pool

Pool can be created on disk or partition. In case of partition this is an example

```
Number  Start (sector)    End (sector)  Size       Code  Name
   1         1026048        74426367   35.0 GiB    8300  Linux filesystem
   2            2048         1026047   500.0 MiB   EF02
   3        74426368        76474367   1000.0 MiB  8200
   4        76474368       976756735   429.3 GiB   BF01  # <------- zfs
   5       976756736       976773119   8.0 MiB     BF07  # ???
```

```
zpool create lxd /dev/sda4
```

libvirt

add ZFS pool to libvirt

```
# create zfs filesystem
zfs create rpool/libvirt

virsh pool-define-as --name zfspool --source-name rpool --type zfs
virsh pool-start zfspool
virsh pool-autostart zfspool
```

create volume (not possibile in virt-manager)

```
virsh vol-create-as --pool zfspool --name maas2 --capacity 10G
```

destroy volume

```
virsh vol-delete --pool zfspool maas2
```

virtualbox

create volume

```
zfs create -V 50G rpool/win7
```

create vmdk file that point /dev/zvol/rpool/win7 (/opt/vms/win7.vmdk keeps little)

```
VBoxManage internalcommands createrawvmdk -filename /opt/vms/win7.vmdk -
rawdisk /dev/zvol/rpool/win7
```

now use /opt/vms/win7.vmdk file as virtual hard disk for virtualbox guest

tuning

pool on SSD https://storagetuning.wordpress.com/2011/12/01/zfs-tuning-for-ssds/

/etc/systctl.conf

```
vfs.zfs.l2arc_noprefetch=0
```

zfs root from live system

```
zpool export rpool
zpool import -R /mnt rpool

umount everything inside /mnt

zfs mount rpool/ROOT/ubuntu


zfs set devices=off rpool
mount --rbind /dev  /mnt/dev
mount --rbind /proc /mnt/proc
mount --rbind /sys  /mnt/sys
chroot /mnt /bin/bash --login

... work here

exit
mount | grep -v zfs | tac | awk '/\/mnt/ {print $3}' | xargs -i{} umount -lf
```

```
{}
zpool export rpool
reboot
```

remote replication

in origin make a snapshot of volume

```
zfs snap storage/cimiteriali@snap1
```

destination volume cannot be exists

```
zfs snap storage/cimiteriali@snap1
zfs send -R storage/cimiteriali@snap1 | pv | ssh zfs1 zfs recv -F -u
rpool/cimiteriali
# zfs destroy storage/cimiteriali@snap1
```

on destination destroy snapshot to have volume

```
zfs destroy rpool/cimiteriali@snap1

# volsize ?
zfs get volsize,reservation rpool/cimiteriali
zfs set volsize=20G rpool/cimiteriali
zfs get volsize,reservation rpool/cimiteriali
```

incremental backup

consider a FS rpool/test

```
FS=rpool/test

# make a first snapshot
zfs snapshot rpool/test@snap01
```

if files are added to FS used space of FS grow

```
sync && zfs list -t all -r $FS
GROW --->rpool/test      200M   138G   200M  /test
         rpool/test@01    64K      -   100M  -
```

if files are changed in FS used space of SNAP grow

```
sync && zfs list -t all -r $FS
         rpool/test      200M   138G   200M  /test
GROW --->rpool/test@01   100M      -   100M  -
```

===⇒ CRYPTOLOCKER grows SNAP

check crypto locker every day

```
LIMIT_MB=100
FS=rpool/test

USED=$(zfs get -Hp used $FS@01 | cut -f3)
if [ $USED -gt $(($LIMIT_MB*1000*1000)) ]; then
   echo "CRYPTOLOCKER detected"

fi
```