Table des matières

Rear (Relax-and-Recover)

Dépendances :

```
mkisofs (or genisoimage)
mingetty (rear is depending on it in recovery mode)
syslinux (for i386 based systems)
nfs-utils (when using NFS to store the archives)
cifs-utils (when using SMB to store the archives)
```

Download : http://download.opensuse.org/repositories/Archiving:/Backup:/Rear/Debian_10/amd64/rear_2.6-0_amd64.deb

```
apt-get install mkisofs mingetty syslinux cifs-utils nfs-utils sshfs
dpkg -i rear_2.6-0_amd64.deb
```

- Fichier de conf basique :

```
root@stkoner-pmox2:~# cat /etc/rear/local.conf
# Default is to create Relax-and-Recover rescue media as ISO image
# set OUTPUT to change that
# set BACKUP to activate an automated (backup and) restore of your data
# Possible configuration values can be found in /usr/share/rear/conf/default.conf
#
# This file (local.conf) is intended for manual configuration. For configuration
# through packages and other automated means we recommend creating a new
# file named site.conf next to this file and to leave the local.conf as it is.
# Our packages will never ship with a site.conf.

OUPUT=ISO
BACKUP=NETFS
BACKUP_URL="sshfs://ben@nas/ZP_nas/stkoner-pmox2-rear"
NETFS_KEEP_OLD_BACKUP_COPY=3
BACKUP_PROG_EXCLUDE=( '/tmp/*' '/dev/shm/*' "$VAR_DIR/output/*" "/ZP_vDisks/*" "/ZP_nas/*" "/ZP_ext/*" )
```

⇒ Si vous utiliser un adressage IP fixe, créer les fichiers ci-dessous :

- **/etc/rear/mappings/ip_addresses**

```
eth0 192.268.1.252/24
```

- **/etc/rear/mappings/routes**

```
default 192.168.1.254 eth0
```

⇒ Penser à copier la clé SSH vers la machine distante

⇒ Lancer la sauvegarde avec :

```
rear -v mkbackup
```

```
root@stkoner-pmox2:~# rear -v mkbackup
Relax-and-Recover 2.6 / 2020-06-17
Running rear mkbackup (PID 54505)
Using log file: /var/log/rear/rear-stkoner-pmox2.log
Running workflow mkbackup on the normal/original system
Using backup archive '/tmp/rear.BwoSapPOuUWGjNq/outputfs/stkoner-pmox2/backup.tar.gz'
Using autodetected kernel '/boot/vmlinuz-5.4.143-1-pve' as kernel in the recovery system
Creating disk layout
Overwriting existing disk layout file /var/lib/rear/layout/disklayout.conf
Using guessed bootloader 'GRUB' (found in first bytes on /dev/sda)
Verifying that the entries in /var/lib/rear/layout/disklayout.conf are correct ...
Creating recovery system root filesystem skeleton layout
Copying logfile /var/log/rear/rear-stkoner-pmox2.log into initramfs as '/tmp/rear-stkoner-pmox2-partial-2021-11-15T12:06:02+01:00.log'
Copying files and directories
Copying binaries and libraries
Copying all kernel modules in /lib/modules/5.4.143-1-pve (MODULES contains 'all_modules')
Copying all files in /lib*/firmware/
Symlink '/usr/share/misc/magic' -> '/usr/share/file/magic' refers to a non-existing directory on the recovery system.
It will not be copied by default. You can include '/usr/share/file/magic' via the 'COPY_AS_IS' configuration variable.
Testing that the recovery system in /tmp/rear.BwoSapPOuUWGjNq/rootfs contains a usable system
Creating recovery/rescue system initramfs/initrd initrd.cgz with gzip default compression
Created initrd.cgz with gzip default compression (266267302 bytes) in 40 seconds
Making ISO image
Wrote ISO image: /var/lib/rear/output/rear-stkoner-pmox2.iso (267M)
```

```
Copying resulting files to sshfs location
Saving /var/log/rear/rear-stkoner-pmox2.log as rear-stkoner-pmox2.log to sshfs location
Copying result files '/var/lib/rear/output/rear-stkoner-pmox2.iso /tmp/rear.BwoSapPOuUWGjNq/tmp/VERSION /tmp/rear.BwoSapPOuUWGjNq/tmp/README /tmp/rear.BwoSapPOuUWGjNq/tmp/rear-stkoner-pmox2.log' to /tmp/rear.BwoSapPOuUWGjNq/outputfs/stkoner-pmox2 at
sshfs location
Making backup (using backup method NETFS)
Creating tar archive '/tmp/rear.BwoSapPOuUWGjNq/outputfs/stkoner-pmox2/backup.tar.gz'
Archived 1873 MiB [avg 8127 KiB/sec] OK
WARNING: tar ended with return code 1 and below output:
  ---snip---
  tar: /var/lib/lxcfs: file changed as we read it
  tar: /var/agentx/master: socket ignored
  tar: pve: Warning: Cannot flistxattr: Operation not supported
  ----------
This means that files have been modified during the archiving
process. As a result the backup may not be completely consistent
or may not be a perfect copy of the system. Relax-and-Recover
will continue, however it is highly advisable to verify the
backup in order to be sure to safely recover this system.

Archived 1873 MiB in 237 seconds [avg 8093 KiB/sec]
Exiting rear mkbackup (PID 54505) and its descendant processes ...
Running exit tasks
root@stkoner-pmox2:~#
```

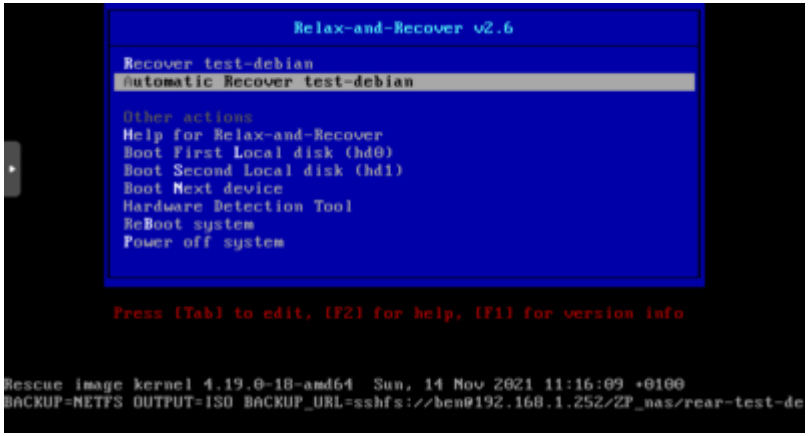
Plusieurs fichiers sont créés :

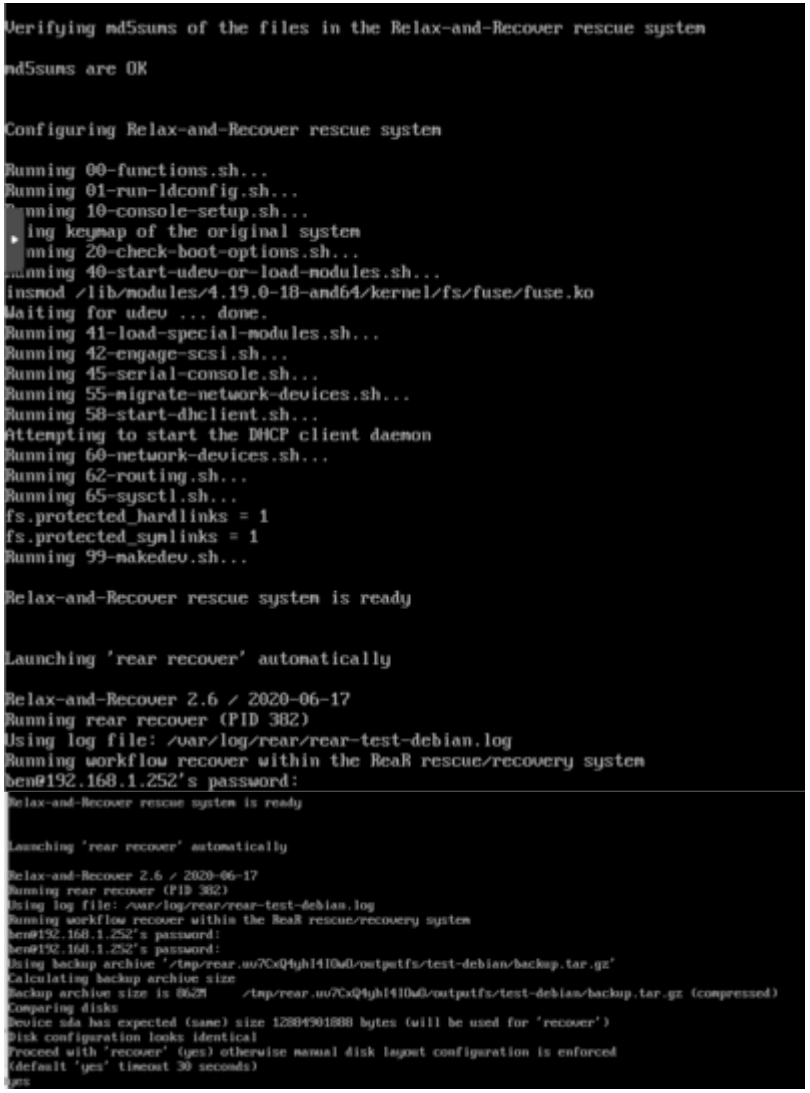
```
root@stkoner-pmox2:~# ssh nas ls -ltr /ZP_nas/stkoner-pmox2-rear/stkoner-pmox2/
total 2198723
-rw------- 1 ben ben  279052288 Nov 15 12:07 rear-stkoner-pmox2.iso
-rw------- 1 ben ben        277 Nov 15 12:07 VERSION
-rw------- 1 ben ben        202 Nov 15 12:07 README
-rw------- 1 ben ben      98085 Nov 15 12:07 rear-stkoner-pmox2.log
-rw------- 1 ben ben 1964661358 Nov 15 12:11 backup.tar.gz
-rw------- 1 ben ben    6004179 Nov 15 12:11 backup.log
```

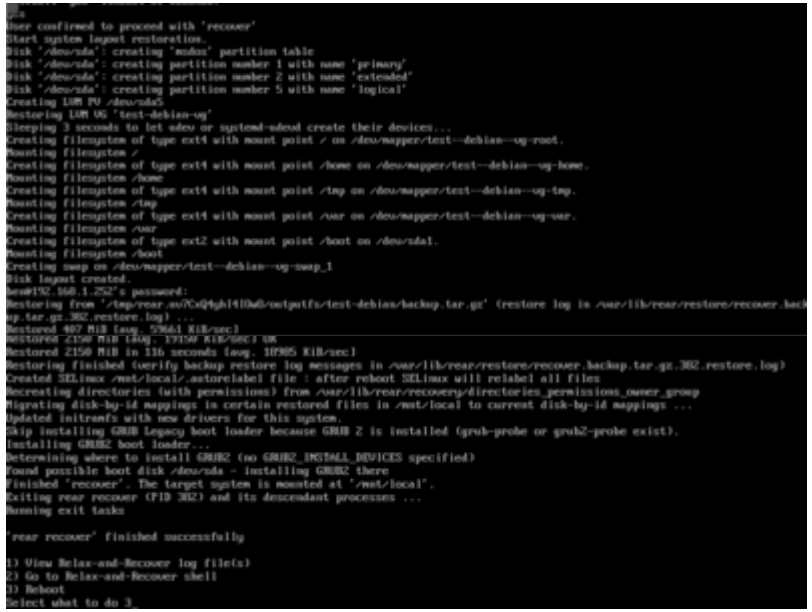
- **rear-stkoner-pmox2.iso** : ISO bootable pour la recovery
- **backup.tar.gz** : contient la sauvegarde OS

FAQ : http://relax-and-recover.org/documentation/faq

- **Test de restauration**




Verifying md5sums of the files in the Relax-and-Recover rescue system
md5sums are OK
Configuring Relax-and-Recover rescue system
Running 00-functions.sh...
Running 01-run-ldconfig.sh...
Running 10-console-setup.sh...
ing keymap of the original system
nning 20-check-boot-options.sh...
Running 40-start-udev-or-load-modules.sh...
insmod /lib/modules/4.19.0-18-amd64/kernel/fs/fuse/fuse.ko
Waiting for udev ... done.
Running 41-load-special-modules.sh...
Running 42-engage-scsi.sh...
Running 45-serial-console.sh...
Running 55-migrate-network-devices.sh...
Running 58-start-dhclient.sh...
Attempting to start the DHCP client daemon
Running 60-network-devices.sh...
Running 62-routing.sh...
Running 65-sysctl.sh...
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
Running 99-makedev.sh...
Relax-and-Recover rescue system is ready
Launching 'rear recover' automatically
Relax-and-Recover 2.6 / 2020-06-17
Running rear recover (PID 382)
Using log file: /var/log/rear/rear-test-debian.log
Running workflow recover within the ReaR rescue/recovery system
ben@192.168.1.252's password:
Relax-and-Recover rescue system is ready
Launching 'rear recover' automatically
Relax-and-Recover 2.6 / 2020-06-17
Running rear recover (PID 382)
Using log file: /var/log/rear/rear-test-debian.log
Running workflow recover within the ReaR rescue/recovery system
ben@192.168.1.252's password:
ben@192.168.1.252's password:
Using backup archive '/tmp/rear.uv7CxQ4yhl4lOwO/outputfs/test-debian/backup.tar.gz'
Calculating backup archive size
Backup archive size is 862M /tmp/rear.uv7CxQ4yhl4lOwO/outputfs/test-debian/backup.tar.gz (compressed)
Comparing disks
Device sda has expected (same) size 12884901888 bytes (will be used for 'recover')
Disk configuration looks identical
Proceed with 'recover' (yes) otherwise manual disk layout configuration is enforced
(default 'yes' timeout 30 seconds)
yes



Backup réseau

Sauvegarder dans un fichier :

```
dd if=/dev/hda bs=1k conv=sync,noerror | gzip -c | ssh user@hostname "gzip -d | dd of=/backup/system.img bs=1k"
dd if=/dev/md0 |gpg -e -r 'cleGPG' - | ncftpput -c -u login -p password hostname system/boot.img.gpg
```

Restaurer à partir d'un fichier :

```
dd if=/backup/system.img bs=1k | gzip -c | ssh user@hostname "gzip -d | dd of=/dev/hda bs=1k"
ncftpget -u login -p password hostname system/boot.img.gpg | gpg -d 'cleGPG' - | dd of=/dev/md0
```

Dupliquer un OS :

```
dd if=/dev/hda bs=1k conv=sync,noerror | gzip -c | ssh user@hostname "gzip -d | dd of=/dev/hda bs=1k"
```

Avec find, cpio and co

```
find /boot -mount -depth |cpio -ova -H crc |gzip | ssh root@server9000982 'cat>/mnt/backup_3696/boot_3696.gz'
find /boot -mount -depth |cpio -ova -H crc |gzip | ssh -q root@host "gunzip |cpio -idvum --absolute-filenames"
find /boot -mount -depth |cpio -ova -H crc |gzip | ssh -q root@host "gunzip |cd /tmp ; cpio -idvum"
```

Backup simple

```
dd if=/dev/sda of=/var/mksysb/$DATE/mbr.`hostname`.`date +%d%m%Y`.sda bs=512 count=1
dd if=/dev/sdb of=/var/mksysb/$DATE/mbr.`hostname`.`date +%d%m%Y`.sdb bs=512 count=1
dd if=/dev/md0 of=/var/mksysb/$DATE/mbr.`hostname`.`date +%d%m%Y`.md0 bs=512 count=1

find /boot -print |cpio -ovc |gzip -c > boot.`hostname`.`date +%d%m%Y`.cpio.gz
find / -xdev -print |cpio -ovc |gzip -c > root.`hostname`.`date +%d%m%Y`.cpio.gz
find /usr -print |cpio -ovc |gzip -c > usr.`hostname`.`date +%d%m%Y`.cpio.gz
find /var -xdev -print |grep -v "/var/cache/apt" |cpio -ovc |gzip -c > var.`hostname`.`date +%d%m%Y`.cpio.gz
```

⇒ pour restaurer le MBR.

Backup / Restore system rapide

⇒ sans LVM

dest : machine accueillant le backup
source : machine à backuper

```
dd if=/dev/sda1 bs=4k conv=sync,noerror,notrunc | gzip -c | ssh root@dest"dd of=/mnt/boot.img.gz bs=4k"
dd if=/dev/sda2 bs=4k conv=sync,noerror,notrunc | gzip -c | ssh root@dest "dd of=/mnt/root.img.gz bs=4k"
```

- Boot sur live CD + recréer les partitions avec fdisk puis à partir de dest :

```
dd if=/mnt/root.img.gz conv=sync,noerror,notrunc bs=4k | ssh root@source"gzip -d | dd of=/dev/sda2 bs=4k"
dd if=/mnt/boot.img.gz conv=sync,noerror,notrunc bs=4k | ssh root@source "gzip -d | dd of=/dev/sda1 bs=4k"
```

Encore un script de sauvegarde

```
#!/bin/bash

set -ux

NFS=X.X.X.X:/var/rhel6/mksysb/
LOCAL_PATH=/tmp/backup.$$
LOG=/var/log/mksysb.log
DEVICE=$1
FS="usr opt var boot"
DATE="date '+%d%m%Y %H:%M:%S'"
HOSTNAME=`hostname |awk '{print tolower($0)}' |awk -F "." '{print $1}'`


# montage du nfs
echo "`eval $DATE` : Montage du NFS distant ..."
mkdir -p ${LOCAL_PATH}
mount $NFS/$HOSTNAME ${LOCAL_PATH}

# backup de la table de partition
echo "`eval $DATE` : Backup de la table de partition ..."
sfdisk -d /dev/$DEVICE > ${LOCAL_PATH}/ptable.$HOSTNAME

# backup du MBR
echo "`eval $DATE` : Backup du MBR ..."
dd if=/dev/$DEVICE of=${LOCAL_PATH}/mbr.$HOSTNAME bs=512 count=1

# backup du VG
for i in `vgs|grep -v VSize|awk '{print $1}'`
do
echo "`eval $DATE` : Backup du $i ..."
vgcfgbackup -d -v $i --file ${LOCAL_PATH}/$i.$HOSTNAME
chmod 644 ${LOCAL_PATH}/$i.$HOSTNAME
done

# Copie de fichiers utiles
echo "`eval $DATE` : Backup de fichiers systeme ..."
fdisk -l > $LOCAL_PATH/fdisk.$HOSTNAME
cat /etc/fstab > $LOCAL_PATH/fstab.$HOSTNAME
> $LOCAL_PATH/pvdisplay.$HOSTNAME
for i in `pvs|grep -v PSize|awk '{print $1}'`
do
pvdisplay >> $LOCAL_PATH/pvdisplay.$HOSTNAME
done

# backup des FS
echo "`eval $DATE` : Backup des FS ..."
for i in $FS
do
FSREN=`echo $i|sed "s%/%_%g"`
find /$i -xdev -print |grep -v mksysb |cpio -ovc |gzip -c > ${LOCAL_PATH}/$FSREN.$HOSTNAME.cpio.gz
done

find / -xdev -print | egrep -v "/var|/usr|/opt|/boot|moteurs|oracle" |cpio -ovc |gzip -c > ${LOCAL_PATH}/root.$HOSTNAME.cpio.gz
find /dev -print |cpio -ovc |gzip -c > ${LOCAL_PATH}/dev.$HOSTNAME.cpio.gz
```

```
echo "`eval $DATE` : Demontage du NFS distant ..."
sleep 1
umount ${LOCAL_PATH}
rmdir ${LOCAL_PATH}
```

Encore un script de restore

```
#!/bin/bash

#set -x

[[ $# -ne 1 ]] && echo "Indiquer la machine en parametre"  && exit

vgchange -an

####### Variable a modifier pour rajouter des FS specifiques a restaurer #######
FS=""

BASE_FS="usr opt var"
LOCAL_PATH=/tmp/restore/$1
VG=`grep -w "/" ${LOCAL_PATH}/fstab.$1 |awk '{print $1}'|awk -F "/" '{print $4}'|awk -F "-" '{print $1}'`
UUID=`grep -A7 $VG ${LOCAL_PATH}/pvdisplay.$1|awk '/UUID/ {print $NF}'`
FAKEROOT=/tmp/fakeroot
DEVICE=`grep -w table ${LOCAL_PATH}/ptable.$1|awk -F "/dev/" '{print $NF}'`
BOOTDEVICE=${DEVICE}1

mkdir -p $FAKEROOT

# Restore de la table de partition
sfdisk --force /dev/$DEVICE < ${LOCAL_PATH}/ptable.$1

# Restore du VG

echo y | pvcreate -ff --norestorefile --uuid $UUID /dev/${DEVICE}2
vgcfgrestore --file ${LOCAL_PATH}/$VG.$1 $VG
vgchange -ay $VG

# Creation des FS et du device de swap

awk -v vg=$V -v fakeroot=$FAKEROOT '
        /vg/ {print "mkfs."$3" "$1}' ${LOCAL_PATH}/fstab.$1 |grep -v swap |sh

size=`grep -A 10 swap ${LOCAL_PATH}/$VG.$1|awk '/extent_count/ {print $3}'`
lvcreate -l $size -n lv_swap $VG
mkswap -f /dev/$VG/lv_swap

# Montage et restore de la racine
ROOT=`grep -w "/" ${LOCAL_PATH}/fstab.$1 |awk '{print $1}'`
BOOT=`grep -w "/boot" ${LOCAL_PATH}/fstab.$1 |awk '{print $1}'`
mount $ROOT $FAKEROOT

cd $FAKEROOT
gzip -dc ${LOCAL_PATH}/root.$1.cpio.gz |cpio -iv --no-absolute-filenames
gzip -dc ${LOCAL_PATH}/dev.$1.cpio.gz |cpio -iv --no-absolute-filenames

FSTYPE=`awk '/boot/ {print $3}' ${LOCAL_PATH}/fstab.$1`
mkfs.${FSTYPE} /dev/$BOOTDEVICE
mkdir -p $FAKEROOT/boot
mount /dev/$BOOTDEVICE $FAKEROOT/boot

gzip -dc ${LOCAL_PATH}/boot.$1.cpio.gz |cpio -iv --no-absolute-filenames

# Montage des autres FS
for i in $FS ${BASE_FS}
do
mkdir -p $FAKEROOT/$i
done

awk -v vg=$V -v fakeroot=$FAKEROOT '
        /vg/ {print "mount "$1" "fakeroot$2}' ${LOCAL_PATH}/fstab.$1 |grep -v swap |sh


cd $FAKEROOT
```

```
for i in $FS ${BASE_FS}
do
gzip -dc ${LOCAL_PATH}/$i.$1.cpio.gz |cpio -iv --no-absolute-filenames
done

# Reinstallation de grub
echo "chroot $FAKEROOT /bin/sh -c \"mount /proc ; mount /sys ; grub-install /dev/$DEVICE\"" | sh

# Mise a jour fstab
sed -i "/UUID/d" $FAKEROOT/etc/fstab
echo "/dev/$BOOTDEVICE  /boot   $FSTYPE    defaults        1 2" >> $FAKEROOT/etc/fstab

echo;echo "### Restauration terminee ###"
```