Bash script backup

From AudioWiki

Contents

BEWARE!

You can't use as name of a snapshot next machine's names:

- hq, hq-s, hq-root
- lab0, lab1, lab2, lab3
- comp0, comp1, comp2, comp3
- lvrealm, lvapache, lvmp0, realm, apache

because you will destroy a machine!!!

Use conditions

In this section it will show how to make changes in the backup script and so get the script which makes operations we want.

How to perform a machines backup only

If you want to perform a machines and home folder backup you must use next command:

```
$ ./backup.sh
```

How to perform a machines backup only

If you only want to perform a machines backup you must use next command:

```
$ ./backup.sh -m m
```

How to perform a home backup only

If you only want to perform a home folder backup you must use next command:

```
$ ./backup.sh -m h
```

Structure of usb hard disk drive

The structure of backup directories is:

```
name-YYYY-MM-DD    name-YYYY-MM-DD-old
```

Where **name** can be:

- A machine: lab0, lab1, lab2, lab3, comp0, comp1, comp2 or comp3.
- Home folder, then name=home.

And:

- YYYY: refers to year of the backup.
- MM: refers month of the backup.
- DD: refers day of the backup.
- **old** extension: when a machine has a backup done and then the system begins to make another one, the last backup (format: name-YYYY-MM-DD) will be renamed to old backup (format: name-YYYY-MM-DD-old) and if another old version backup of this machine exists, it will be removed.

Constraints

You must maintain the structure explained above because if you don't do it backup won't be done.

Allowed cases:

- Folder where you want to place backup is empty.
- Folder where you want to place backup has backup of some machines but other machines don't have a backup in this folder. For example:

```
comp0-2013-11-07-old
comp0-2013-11-09
lab1-2013-11-07-old
lab1-2013-11-09
lab3-2013-11-07-old
lab3-2013-11-09
home-2013-11-07-old
home-2013-11-09
```

- Folder where you want to place backup has backup of a machine but hasn't a old backup or viceversa. A situation which we have last backup of a machine but not its old backup:

```
comp0-2013-11-07-old
comp0-2013-11-09
lab1-2013-11-09
lab3-2013-11-07-old
lab3-2013-11-09
```

```
home-2013-11-07-old
home-2013-11-09
```

A situation which we have old backup of amachine but not its last backup:

```
comp0-2013-11-07-old
comp0-2013-11-09
lab1-2013-11-07-old
lab3-2013-11-07-old
lab3-2013-11-09
home-2013-11-07-old
home-2013-11-09
```

Error cases:

- There are two or more last backup for the same machine or home folder. This situation will provoque backup only for that machine or home folder won't be done. The rest of machines will have a backup. For example:

```
comp0-2013-11-07-old
comp0-2013-11-09
comp1-2013-11-07-old
comp1-2013-11-09
comp1-2013-11-19
...
lab3-2013-11-09
home-2013-11-07-old
home-2013-11-09
```

In this case all machines and hme folder will have a backup except comp1 machine.

- The same situation is applied to old backup. If there are two or more old backup for the same machine or home folder, that machine or home folder won't have a backup. For example:

```
comp0-2013-11-07-old
comp0-2013-11-09
comp1-2013-11-07-old
comp1-2013-11-08-old
comp1-2013-11-09
...
lab3-2013-11-09
home-2013-11-07-old
home-2013-11-09
```

Scritp

```
#!/bin/bash

#############################################################################################################
# backup.sh - script that performs a lv backup of all voice group machines and user's home.
#
# To backup machines and home folder in the same action usage: ./backup.sh
# To backup machines only usage: ./backup.sh -m m
# To backup home folder only usage: ./backup.sh -m h
#
# Author: Jesús Campos Muñoz
# Complaints: jesus.campos@tsc.upc.edu
#############################################################################################################
```

```
echo "starting ... " $(/bin/date +%H:%M-%Y-%m-%d)

#Directory which holds the machine we want to backup.
pSnapshot=/dev/virt0/
#Directory where mount machine's backup.
pSnapshotMount=/mnt/snapshots/
#USB HDD
#USB hard disk directory (where to mount usb device)
pUsb=/mnt/usb/
#USB hard disk device
pUsbDevice=/dev/sdb1/
#Copy extensions
dateFormat=[-][0-9][0-9][0-9][0-9][-][0-9][0-9][-][0-9][0-9]
oldExtension="-old"
#Snapshot size
snapshotSize=300M
#Machines' name we want to backup: lab0, lab1, ..., comp0, comp1, ...
machinesGroup="lab0 lab1 lab2 lab3 comp0 comp1 comp2 comp3"

#Arrange of usb direcories
prepareUsb(){

 #Start preparing USB device.
 if [ -d $pUsb$machine$dateFormat ]; then

  vars=
  #Testing if .old version of backup exists it willbe erased and the actually backup version will be rename t
  if [ -d $pUsb$machine$dateFormat$oldExtension ]; then
   /bin/rm -rf $pUsb$machine$dateFormat$oldExtension

  else
   echo OLD::.old version doesnt exist
  fi

  vars=$(/bin/ls -d $pUsb$machine*)

  #Rename actual backup to .old backup.
  /bin/mv  $vars $vars$oldExtension

 #If doesn't exist a no .old backup version we don't mind about .old version
 else
  echo REGULAR_VERSION::backup doesnt exist
 fi

 #End preparing USB device.
}

#MACHINES: lab0, lab1, .... copm0, comp1
#These functions make the snapshot, mount the snapshot, backup the system and umount the snapshot.
work(){
 #Snapshot's name.
 snp="Snapshot"
 snapshotName=$machine$snp

 prepareUsb

 #Creating folder for current backup. This poitn is diferent in backup home, instead of using mahcine name i
 /bin/mkdir $pUsb$machine-$(/bin/date +%Y-%m-%d)

 #Creating snapshot machine.
 /sbin/lvcreate -s -L$snapshotSize -n $pSnapshot$snapshotName $pSnapshot$machine

 #mount /dev/virt0/machineSnapshot /mnt/snapshots/machine
 /bin/mount $pSnapshot$snapshotName $pSnapshotMount$machine

 #Creating a new folder with current date
 echo "********* starting " $machine " information copy *********"

 /bin/cp -Rp $pSnapshotMount$machine/* $pUsb$machine$dateFormat
 echo "********* finished " $machine " information copy *********"

 #Umounting snapshot
 /bin/umount $pSnapshotMount$machine

 #Se elimina el snapshot
 /sbin/lvremove $pSnapshot$snapshotName -f

}

backupMachines(){
```

```
 #Call function for every machine.
 for machine in $machinesGroup
 do
  echo "================================" $machine "backup starting ================================"
  work
  echo "================================" $machine "backup finished ================================"
  echo "finishing ... " $(/bin/date +%H:%M-%Y-%m-%d)
  echo
 done
}


#USERs HOME
#These functions make the snapshot(kpartx), mount the snapshot, backup the system and umount the snapshot.
workHome(){
 #Snapshot's name.
 snp="Snapshot"
 snapshotName=$typePartition$snp

 #Change machine's name for using correctly prepareUsb function.
 machine="home"

 prepareUsb

 machine="hq-s"

 #Creating folder for current home folder backup
 /bin/mkdir $pUsb$typePartition-$(/bin/date +%Y-%m-%d)

 #Creating snapshot machine.
 /sbin/lvcreate -s -L$snapshotSize -n $pSnapshot$snapshotName $pSnapshot$machine

 #Mounting snapshot
 #kpartx -a /dev/virt0/homeSnapshot
 /sbin/kpartx -a $pSnapshot$snapshotName
 #mount /dev/mapper/virt0-homeSnapshot8 /mnt/snapshots/home/
 /bin/mount $pKpartSnapshotMount$snapshotName$numPartition $pSnapshotMount$typePartition

 #Creating a new folder with current date
 echo "********* starting " $machine " information copy *********"

 /usr/bin/rsync -rptgoD $pSnapshotMount$typePartition/* $pUsb$typePartition$dateFormat
 echo "********* finished " $machine " information copy *********"

 ##Umounting snapshot
 #umount /mnt/snapshots/home/
 /bin/umount $pSnapshotMount$typePartition
 #umount /mnt/snapshots/homeSnapshot
 /sbin/kpartx -d $pSnapshot$snapshotName

 #Removing snapshot
 if /sbin/lvremove -f $pSnapshot$snapshotName 2>&1|/bin/grep  -q "Failed"
  then
   if /sbin/lvremove -f $pSnapshot$snapshotName 2>&1|/bin/grep  -q "Failed"
    then
     echo "Error in lvremove " $pSnapshot$snapshotName

     recipient="root@localhost"

     message="Backup error"

     subject="Error in lvremove " $pSnapshot$snapshotName " in home users backup at " $(/bin/date +%H:%M-%Y-%

     echo "$message" | /usr/bin/mail -s "$subject" $recipient
   fi

 fi

}

backupHome(){
 echo "starting home backup ... " $(/bin/date +%H:%M-%Y-%m-%d)
 #Users' home backup
 machine="hq-s"
 pKpartSnapshotMount=/dev/mapper/virt0-
 typePartition="home"
 numPartition="8"
 echo "================================ " $machine"/"$typePartition " backup starting ======================
 workHome
 echo "================================ " $machine"/"$typePartition " backup finished ======================
```

```
 echo
 echo
}



###SCRIPT STARTING
#Mounting USB hard disk
/bin/mount $pUsbDevice $pUsb

MODE="all"

while getopts m: option
do
       case "${option}"
       in
               m) MODE=${OPTARG};;
       esac

       if [ ${option} = "?" ]
       then
               MODE="none"
       fi
done

if [ $MODE = "all" ]
then
 backupMachines
 backupHome

elif  [ $MODE = "h" ]
then
 #Action: backupHome.
 backupHome

elif  [ $MODE = "m" ]
then
 #Action: backupMachines.
 backupMachines

elif  [ $MODE = "none" ]
then
       echo "With -m option parameter m or h is needed."
else
       echo "Unrecognised parameter."
fi

#Unmounting USB hard disk
/bin/umount $pUsb

echo "FINISHED " $(/bin/date +%H:%M-%Y-%m-%d)
```
