ADMINISTRATION IT

LAB 01 - LINUX BACKUP

Authors : Nair Alic & Adam Zouari

TASK 1: PREPARE THE BACKUP DISK

1.Before plugging the disk in examine the special files in the /dev directory that represent hard disks. List all files called

- /dev/hd*
- /dev/sd*

Which disks and which partitions on these disks are visible?

We can see 1 disk (sda) and 5 partitions (sda1/sda2/.../sda5) here.

Which partitions are mounted? Use the command mount without parameters to find out.

We can see that only sda2, sda3 and sda5 are mounted.**

2.Attach the disk to your computer.

Consult again the special files in /dev. Which new files appeared? These represent the disk and its partitions you just attached.

We can see that a new file sdb appeared.

3.Create a partition table on the disk and create two partitions of equal size using the parted tool.

You can consult Gentoo Linux Documentation — Preparing the Disks as a reference on how to use parted.

- Using superuser privileges invoke parted with a single parameter which is the special file representing the disk. Be careful not to confuse the special file for the disk (ending in a letter) and for the partitions (ending in a number).

- Display the existing partitions with the print command. If the disk is completely blank you will get an error message about a missing disk label.

- Use the mktable command to create a partition table (overwriting any existing one). It should have Master Boot Record (MBR) layout (i.e. label type msdos).

- Display the free space with the command print free (roughly the size of the disk minus some overhead). Write the value down.

  We have 1074 MB of size available

- Use the mkpart command to create the partitions.

  - The first partition will

    - be a primary partition
    - have a file system type of fat32
    - start at 0
    - end at about half the free space.

  - The second partition will

    - be a primary partition
    - have a file system type of ext4
    - start at half the free space
    - end at the free space.

- Quit parted and verify that there are now two special files in /dev that correspond to the two partitions.

4. Format the two partitions using the mkfs command.

- The first partition should have the file system type vfat.
- The second partition should have the file system type ext4.

5. Create two empty directories in the /mnt directory as mount points, called backup1 and backup2. Mount the newly created file systems in these directories.

6. How much free space is available on these filesystems? Use the df command to find out. What does the -h option do?

```
osboxes@osboxes:~$ df -h /mnt/backup1
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb1       512M  4.0K  512M   1% /mnt/backup1
osboxes@osboxes:~$ df -h /mnt/backup2
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb2       488M  2.3M  456M   1% /mnt/backup2
osboxes@osboxes:~$
```

Here we can see that we have 512 MB for backup1 and 456 MB for backup2 folder. Option -h print the sizes in powers of 1024, which are more readable for us.

TASK 2: PERFORM BACKUPS USING TAR AND ZIP

In this task you will perform different backup tasks. For each write a quick reference that you would consult as a system administrator when you have forgotten the exact invocation of the command. Do this using the `tar` command and then using the `zip` command.

The backup tasks are the following:

- Do a backup of a user's home directory to the backup disk (VFAT partition). Create a compressed archive. Do the files in the archive have a relative path so that you can restore them later to any place?

```
tar -cvzf osboxes-backup.tar.gz /home/osboxes
```

```
zip -r /mnt/backup1/osboxes-backup-zip.zip
```

Yes, they have relative path. "tar" don't store absolute paths in this case. We can restore wherever we want.

- List the content of the archive.

```
tar -tvf osboxes-backup.tar.gz
```

```
unzip -l osboxes-backup-zip.zip
```

- Do a restore of the archive to a different place, say `/tmp`.

```
tar -xvzf osboxes-backup.tar.gz -C /tmp/restoreBackup
```

```
unzip /mnt/backup1/osboxes-backup-zip.zip
```

- Do an incremental backup that saves only files that were modified after, say, September 23, 2016, 10:42:33. Do this only for `tar`, not for `zip`.

  - Use the `find` command to determine the files that should be included in the backup.
  - Use `tar`'s `-T` option to read the names of the files to be archived from a file

```
touch -d "23 Sep 2016 10:42:33" /tmp/backupDate
```

```
find /home/osboxes -newer /tmp/backupDate -type f >
/tmp/newFilesToBackup
```

```
tar -cvf /mnt/backup1/incrementalBackup.tar.gz -T /tmp/newFilesToBackup
```

TASK 3: BACKUP OF FILE METADATA

In this task you will examine how well the backup commands preserve file metadata. Consult the man pages and perform tests using `tar` and `zip` and examine whether you can restore:

- the last modification time
- the permissions
- the owner

In the lab report describe the tests you did and their results.

We did some tests and here is what we find out for fat32 and ext4 partitions :

We have made few backups using tar and zip. In the table below you'll find out the results made on each partitions. Here we have the results for backup1 which is the **fat32** partition. (If OK it means the metadatas are preserved, NOK means not preserved.)

Metadata	TAR	ZIP
the last modification	OK	OK

the permissions	NOK	NOK
the owner	NOK	NOK

In order to test, we created some files and we changed few things like owner, group before the backup.

Here are the results for backup2 which is on **ext4** partitions :

Metadata	TAR	ZIP
the last modification	OK	OK
the permissions	OK	OK
the owner	NOK	NOK

We can see that the owner always change. The user who untar or unzip the backup becomes the new owner. The last modification time and permissions are kept.

There are options in case we need to keep the owner. For untar the options **—same-owner** will do the job. For unzip the options is **-X**. This is the case on linux filesystems only.

TASK 4: SYMBOLIC AND HARD LINKS

In this task you will examine whether the backup commands preserve symbolic and hard links. Consult the man pages and perform tests using tar and zip. In the lab report describe the tests you did and their results.

All **FAT** filesystems doesn't support symbolic/hard links so we only tested on the **ext4** partitions :

Metadata	TAR	ZIP
hard links	OK	NOK
symbolic links	OK	NOK

We can see that basic zip command don't preserve the links but store the file referred to. For UNIX systems, the **—symlinks** option can be used to preserve only symbolic links. For tar, the links are preserved.