Advance Cyber Security Serial No 09

Assignment 5

shreyyash

2020

April

Contents

1. Assignment 5: Using Disk Duplicator and Mount Command to Prepare a Bootable USB
   1. Introduction………………………………………. 3
   2. Answer 1………………..…………………………. 3
   3. Answer 2………….…….…………………………. 4
   4. Answer 3…………………………………………... 8

List of figures

1.2.1 Figure 1: Partition backup and restore...……………..……………. 3

1.2.2 Figure 2: Formatting of .vhd image File ……………….…………. 4

1.2.2 Figure 3: Mounting a virtual file system ……………………..…… 4

1.2.3 Figure 4: Backing up the MBR…………………………………….. 5

1.2.4 Figure 5: Convert from ASCII to EBCDIC………………………... 5

1.2.5 Figure 6: Convert from Lower to Upper case and vice-versa……... 6

1.3.1 Figure 7: Partition using fdisk command ………………………...... 6

1.3.1 Figure 8: Preparing and naming the ext partition ………………… 7

1.3.1 Figure 9: Preparing and naming the Fat partition…………………. 7

1.3.2 Figure 10: Creating bootable partitions…………………………….. 8

CHAPTER5

Assignment 5: Using Disk Duplicator and Mount

Command to Prepare a Bootable USB

* 1. Introduction

The dd command was practiced on Parrot OS as the base and an 8 GB pendrive was used for the assignments.

* 1. Answer 1

1.2.1 Backup and Restore an HDD or Partition

The following command copied a partion to another partition.

**dd if=/dev/sda of=/dev/sdb**

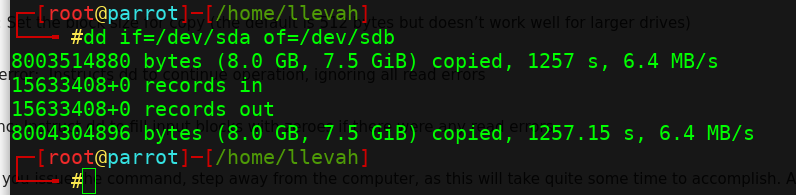
Command usage is as follows

dd - disk duplicator

if - input file

of - output file

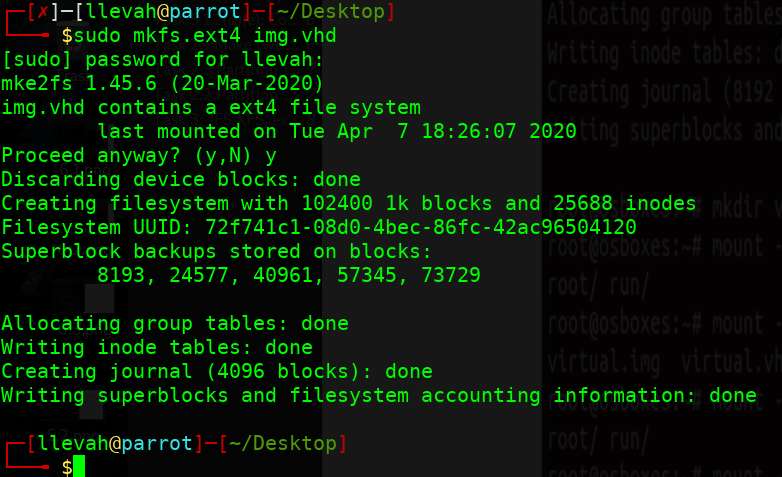
/dev/sdx - source and destination address



*Figure 1: Partition backup and restore*

1.2.2 Virtual Filesystem and ISO of CD /DVD

The Virtual Filesystem can be made by using an .**img** or **.iso** etc file. The command to partition our .vhd file to ext4 since it was made from **/dev/zero** is **makefs.ext4 <source>**.



*Figure 2: Formatting of .vhd image File before mounting*

The partition is mounted using

**mount –t auto –o loop <source> <dest>.**



*Figure 3: Mounting a virtual file system*

Command description is as follows

mkfs - make filesystem

ext4 - type of filesystem

mount - mount the filesystem

-t auto - type of filesystem, auto

-o - options

loop - use file as mount device

1.2.3 Copy raw region like MBR

The MBR can be copied using the command

**dd if=<source> of=<destn> bs=512 count=1**

Command usage is as follows

dd - disk duplicator

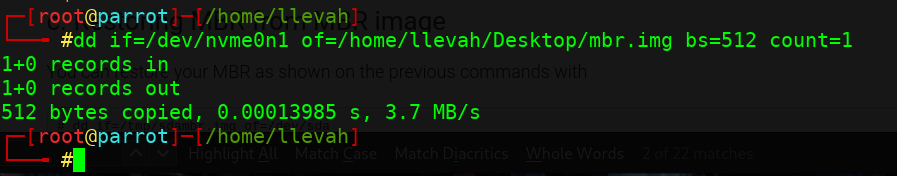
<source> - MBR source

<destn> - destination to save

bs - block size

count - no of times to repeat bs

# The product of **bs** and **count** should be greater than or equal to the size of data to copy. In this case, MBR is generally of size 512 bytes, thus the below shown values have been taken.



*Figure 4: Backing up the MBR*

1.2.4 Convert data format from ASCII to EBCDIC

The conversion from one data format to the other can be done by the following command.

**dd if=<source> of=<destn> conv=<destn file format>**

The command usage is as follows

dd - disk duplicator

conv - convert file format

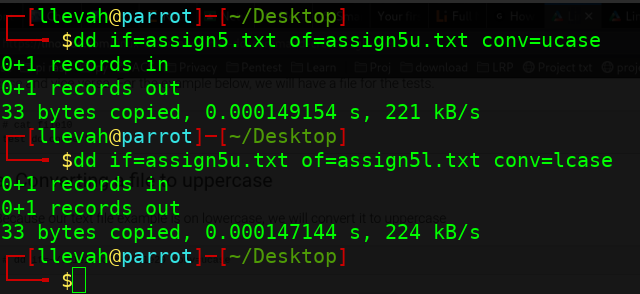


*Figure 5: Convert from ASCII to EBCDIC*

1.2.5 Lower Case to Upper Case and Vice-versa

The command to convert from one case to another is as follows.

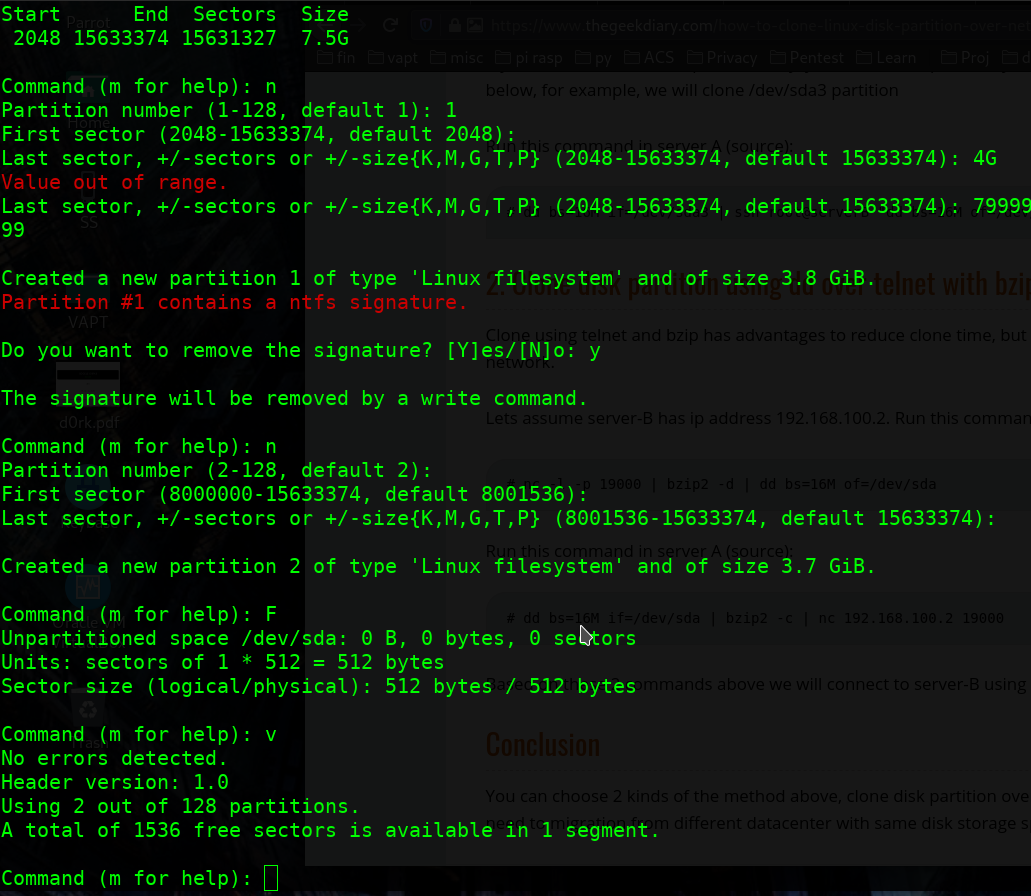
**dd if=<source> of=<destn> conv=<ucase/lcase>.**



*Figure 6: Convert from Lower to Upper case and vice-versa*

1.3 Answer 2

1.3.1 The mount point of the pen drive was checked using **lsblk** command and found to be **/dev/sda**. The pen drive was then partitioned into two partitions using **fdisk /dev/sda.** The screen shot of the same is given below



*Figure 7: Partition using fdisk command*

The formatting of the partition was carried out using the command

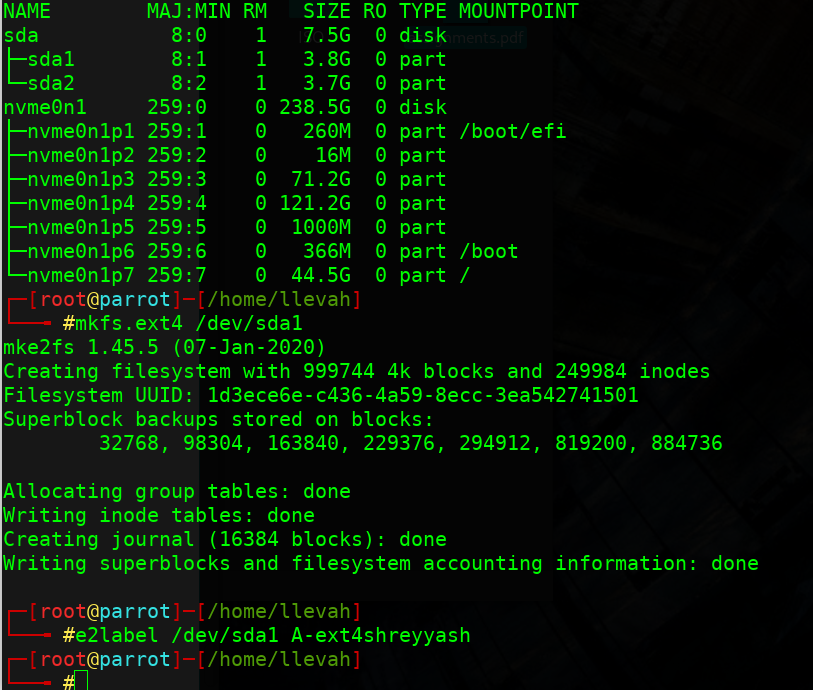
**mkfs.<filesystem> <path>**

Naming as per format given was carried out using the command

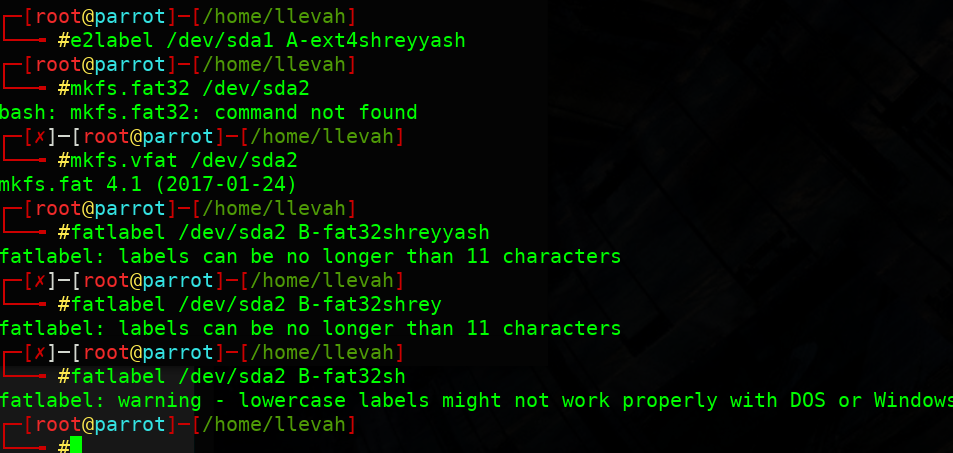
**e2label <path> <name>** #for ext3

**fatlabel <path> <name>** #for fat partition, label must be shorter

than 12 chars



*Figure 8: Preparing and naming the ext partition*



*Figure 9: Preparing and naming the Fat partition*

1.3.2 Creating Bootable Partitions

The size of Parrot ISO being larger than 4GB, two partitions of dissimilar sizes were created using **fdisk** command. The sequence of commands for writing and naming was as follows

**mkfs.ext3 /dev/sda1**

**e2label /dev/sda1 Parrot**

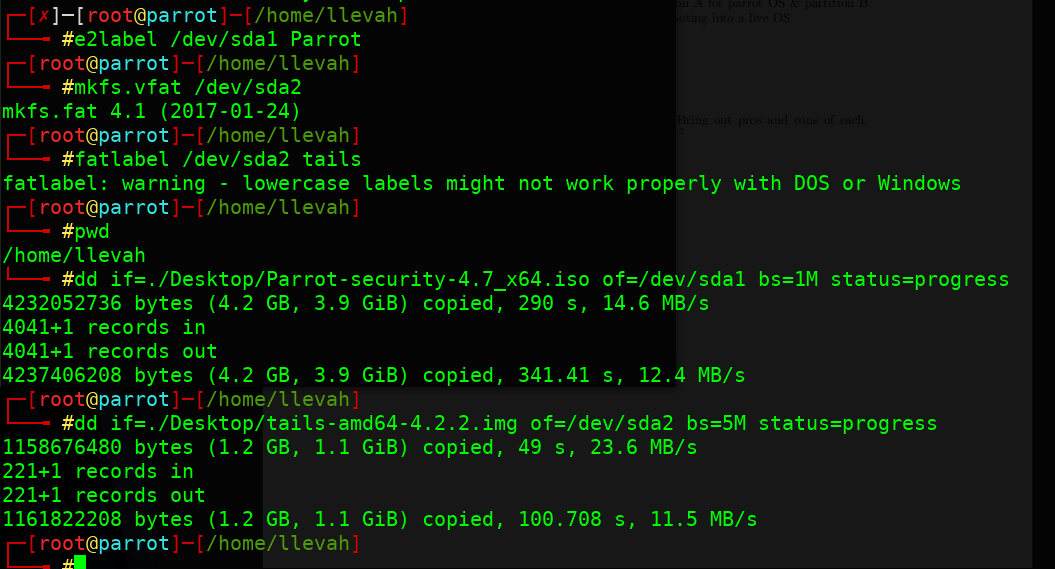
**mkfs.vfat /dev/sda2**

**fatlabel /dev/sda2 tails**

**dd if=./Desktop/Parrot…x64.iso of=/dev/sda1**

**dd if=./Desktop/tails...2.img of=/dev/sda2**

The screen shots of the same are given below



*Figure 10: Creating bootable partitions*

1.4 Answer 4

1.4.1 Study of JFS, XFS, ext3 and ext4 Filesystems

1.4.2 XFS File System

XFS introduced in 1993 by silicon graphics

Pros

* Excels in parallel i/o
* Extremely scalable
* Delayed allocation reducing fragmentation
* Supports journaling

Cons

* No support for snapshots
* Internal log for journaling, creates contention for disk
* Can only be shrunk by Backing up, Resizing and Restoring
* No Checksum protection

1.4.3 JFS File System

64-bit Journaling File System created by IBM in 1990

Pros

* Inherent support for journaling
* High i/o performance using allocation groups

Cons

* Poor compression
* Problem in implementation of Journal writes

1.4.4 Ext2 File System

64 bit file system introduced by Remy Card in 1993

Pros

* Ordered mode of journaling
* Internal structure resistant to crashing

Cons

* Slower as compared to others
* Data prone to loss in case of crash
* Issues due to backward compatibility to ext2
* No checksum in journal

1.4.5 Ext4 File System

Pros

* Support vol upto 1 Exbibyte (260 bytes)
* Backward compatible
* Delayed allocation
* Metadata Checksum
* Transparent encryption

Cons

* No secure deletion
* Potential data loss due to delayed allocation

1.4.6 Final Remarks

Ext4 File System is the most robust and file system with the least issues. The issues ext4 does have can be mitigated with some effort. Thus ext4 is the preferred file system.