CSU, College of Engineering and Computer Science
Department of Computer Science
CSC 153 – Computer Forensics Principles and Practice

Catherine Nguyen
CSC 153
10/11/2020

Activity 3: Linux Data Acquisition

Introduction

Data Acquisition is the process of copying data. For digital forensics, it's the task of collecting digital evidence from electronic media. There are two types of data acquisition: static acquisition and live acquisition. In this practice, you'll perform a static data acquisition using the Linux dd and dcfldd commands. Operations in this guide are performed in CAINE Linux. You may also use Kali Linux for the same practice.

Objectives

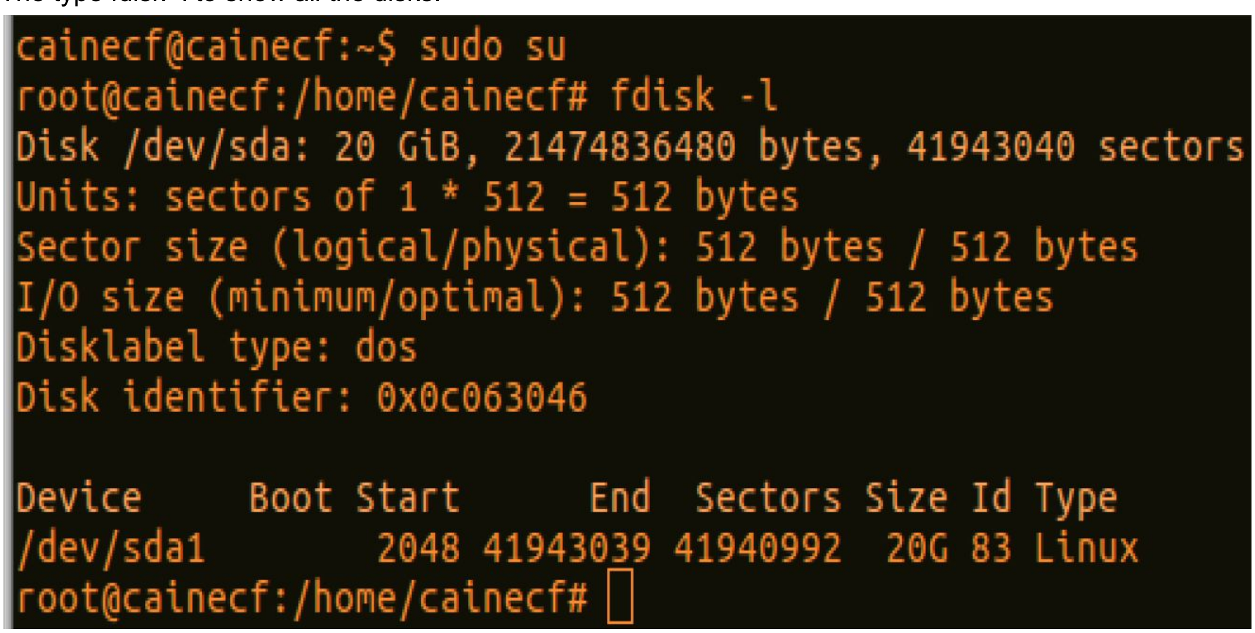
- Get familiar with the data acquisition process.
- Prepare a target drive for data acquisition.
- Use Linux data acquisition tool to acquire data from a USB drive.
- Validate the acquired data.

Tasks

- Part 1: prepare the target drive-by zeroing-out the target drive and creating a new file system on it;
- Part 2: perform a data acquisition by acquiring data from evidence drive to target drive;
- Part 3: validate the acquisition.

Part 1. Prepare the target drive

We start off by getting into the mate terminal and logging into the root account. I used 'sudo su'.
The type fdisk -l to show all the disks.



```
cainecf@cainecf:~$ sudo su
root@cainecf:/home/cainecf# fdisk -l
Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x0c063046

Device     Boot Start      End  Sectors Size Id Type
/dev/sda1        2048 41943039 41940992  20G 83 Linux
root@cainecf:/home/cainecf#
```

Fig 1: No disk to be found

Next, we need to connect the USB and rerun fdisk -l.

```
root@cainecf:/home/cainecf# fdisk -l
Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x0c063046

Device     Boot Start      End  Sectors Size Id Type
/dev/sda1        2048 41943039 41940992  20G 83 Linux


Disk /dev/sdb: 1.9 GiB, 1992294400 bytes, 3891200 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x021627d5

Device     Boot Start     End Sectors  Size Id Type
/dev/sdb1  *       32 3891199 3891168  1.9G  6 FAT16
root@cainecf:/home/cainecf#
```

Fig 2: The drive is not plugged in.

Now we need to zero out the drive using ‘dd if=/dev/zero of=/dev/sdb status=progress’. This shows the progress at which it will take to zero out.

```
root@cainecf:/home/cainecf# dd if=/dev/zero of=/dev/sdb status=progress
1991582208 bytes (2.0 GB, 1.9 GiB) copied, 2042 s, 975 kB/s
dd: writing to '/dev/sdb': No space left on device
3891201+0 records in
3891200+0 records out
1992294400 bytes (2.0 GB, 1.9 GiB) copied, 2044.01 s, 975 kB/s
root@cainecf:/home/cainecf#
```

Fig: 3 Zeroed out the drive using the dd command.

Next, we have to create a partition using ‘fdisk /dev/sdb’.

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-3891199, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-3891199, default 3891199):

Created a new partition 1 of type 'Linux' and of size 1.9 GiB.

Command (m for help):


Command (m for help): p

Disk /dev/sdb: 1.9 GiB, 1992294400 bytes, 3891200 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x5a2f562e

Device     Boot Start     End Sectors  Size Id Type
/dev/sdb1        2048 3891199 3889152  1.9G 83 Linux
```

Fig 4: The partition is created.

Next is to change this new partition to windows 95 FAT32 file system. Then run ‘fdisk -l’ to see all the drives again. Use ‘t’ to change the partition type. Then use ‘l’ to list all the known partition types.

```
Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Synching disks.

root@cainecf:/home/cainecf# fdisk -l
Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x0c063046

Device     Boot Start      End  Sectors Size Id Type
/dev/sda1        2048 41943039 41940992  20G 83 Linux


Disk /dev/sdb: 1.9 GiB, 1992294400 bytes, 3891200 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x5a2f562e

Device     Boot Start     End Sectors  Size Id Type
/dev/sdb1        2048 3891199 3889152  1.9G  c W95 FAT32 (LBA)
root@cainecf:/home/cainecf#
```

Fig 5: Changed the new partition into W95 FAT32.

Lastly, for this part. We need to format a FAT file system from Linux, with ‘mkfs.msdos –vF32 /dev/sdb1’.

```
root@cainecf:/home/cainecf# mkfs.msdos -vF32 /dev/sdb1
mkfs.fat 3.0.28 (2015-05-16)
/dev/sdb1 has 62 heads and 62 sectors per track,
hidden sectors 0x0800;
logical sector size is 512,
using 0xf8 media descriptor, with 3889152 sectors;
drive number 0x80;
filesystem has 2 32-bit FATs and 8 sectors per cluster.
FAT size is 3791 sectors, and provides 485192 clusters.
There are 32 reserved sectors.
Volume ID is 004242ad, no volume label.
root@cainecf:/home/cainecf#
```

Fig 6: Formatting the FAT file system from Linux.

Part 2. Perform Data Acquisition

Next is to plug in the second USB drive and show them with the command ‘fdisk -l’. After that, we need to perform data acquisition.

```
root@cainecf:/home/cainecf# fdisk -l
Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x0c063046

Device     Boot Start      End  Sectors Size Id Type
/dev/sda1        2048 41943039 41940992  20G 83 Linux


Disk /dev/sdb: 1.9 GiB, 1992294400 bytes, 3891200 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x5a2f562e

Device     Boot Start     End Sectors  Size Id Type
/dev/sdb1        2048 3891199 3889152  1.9G  c W95 FAT32 (LBA)


Disk /dev/sdc: 1.9 GiB, 1992294400 bytes, 3891200 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x028e887c

Device     Boot Start     End Sectors  Size Id Type
/dev/sdc1  *       32 3891199 3891168  1.9G  6 FAT16
root@cainecf:/home/cainecf#
```

Fig 7: The evidence drive is the /dev/sdc1

Next is to mount the target drive partition then change the default directory to the target drive. To mount run ‘mount -t vfat /dev/sdb1 /mnt/sdb1’. After that we change the dir ‘cd /mnt/sdb1/’ and make the file case1 with ‘mkdir case1’ The to calculate the hash of the evidence drive we run the MD5 algorithm. ‘Md5sum /dev/sdc |tee /mnt/sdb1/case1/pre-imagesource.md5.txt’

```
root@cainecf:/home/cainecf# mkdir /mnt/sdb1
root@cainecf:/home/cainecf# mount -t vfat /dev/sdb1 /mnt/sdb1
root@cainecf:/home/cainecf# md5sum /dev/sdc1 |tee /mnt/sdb1/case1/pre-imagesourc
e.md5.txt
58a003798388a9e718691551665710a8  /dev/sdc1
root@cainecf:/home/cainecf#
```

Fig 8: Verifying the acquired data

Part 3: Validate the data acquired

Next we run 'dcfldd if=/dev/sdc of=/mnt/sdb1/case1/image1.dd conv=noerror,sync hash=md5 hashwindow=0 hashlog=/mnt/sdb1/case1/post-imagesource.md5.txt' this acquires the data from the evidence drive.

```
root@cainecf:/mnt/sdb1# dcfldd if=/dev/sdc of=/mnt/sdb1/case1/image1.dd conv=noerror,sync
sh=md5 hashwindow=0 hashlog=/mnt/sdb1/case1/post-imagesource.md5.txt
60416 blocks (1888Mb) written.
60649+0 records in
60648+0 records out
```

Fig 9: Validation of acquired data.

Post-Activity questions.

1. What are the two broad categories of acquisition?

Static Acquisition, Live Acquisition.

2. What is a live storage acquisition and when is it used?

- Live storage acquisition = data collected from the local ps or over a running network. Is not repeatable because the OS alters the data continuously.
- Use it when the computer can't be shut down.

3. Which command should be used to check the disks available on the current system? You only need to state the command name, not the entire command string.

- fdisk, in the activity we used 'fdisk -l'.

4. The mkfs -t command does what?

- Makes a file system a certain type.

5. Which drive should be 'zeroed out', the source evidence drive, or the target drive?

- The target drive.

6. What is the purpose of 'zeroing out' before a storage acquisition is performed?

- The make sure there is nothing else in the flash drive that could tamper with the evidence. For example, software, malware.

7. When you issue the command dd if=/dev/zero of=/dev/sdb What does the string "/dev/sdb" represent?

- This command zeroes out the target drive before the evidence is copied into the drive. The '/dev/sbd/ represents the target drive.

8. The md5sum /dev/sdc command does what? Why is it used?

- Generates a hash of the evidence before we copy it, to compare with the hash of the copy. This allows us to validate the evidence to make sure it is the same.

9. How many times should the md5sum command be used at least in one acquisition?

- One time for the pre-image source when hashing the evidence drive. Another time once for the post-image source after the acquisition.

10. Instead of using "dd", what other commands can you use to perform data acquisition in Linux?

- We used 'dcfldd'. It is an enhanced version of 'dd'.