Homework 1 Report

Environment setup

The OS used for the tasks is **Linux Kali 4.19.0-kali4-amd64 (2019-03-18)**
The programs used via the Linux environment are:

- dd *v8.30*
- md5sum *v8.30*
- hdparm *v9.58*
- The Sleuth Kit *v4.6.5*
- FTK Imager *v3.1.1 CLI (Aug 24 2012)*

Task 1

1) When we introduce the USB stick, we can see that the device is mounted via the command **lsblk**
So we wipe out the whole disk with the command **dd** (Disk Drop).
Command: *dd if=/dev/zero of=/dev/sdb bs=4096*

```
root@kali:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   25G  0 disk
├─sda1   8:1    0 22.2G  0 part /
└─sda2   8:2    0  2.9G  0 part [SWAP]
sdb      8:16   1   15G  0 disk /media/root/evidence
sr0     11:0    1 1024M  0 rom
root@kali:~# dd if=/dev/zero of=/dev/sdb bs=4096
dd: error writing '/dev/sdb': Operation not permitted
2657279+0 records in
2657278+0 records out
10884210688 bytes (11 GB, 10 GiB) copied, 2216.33 s, 4.9 MB/s
root@kali:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
```

3) We can generate the hash of the wiped device via the **md5sum** program, and also using FTK Imager. We will save these two hashes in a file “wiped-md5.txt”, so we can compare them later.
Command: *./ftkimager devsdb --verify >> wiped-md5.txt*
Hash generated: *8a618d508e269f4b888bc0a937a832ae*
Command: *md5sum dev/sdb >> wiped-md5.txt*
Hash generated: *ea8641a293b187421ff415af90926e52*

```
root@kali:~/forensics# ./ftkimager /dev/sdb --verify >> wiped-md5.txt
AccessData FTK Imager v3.1.1 CLI (Aug 24 2012)
Copyright 2006-2012 AccessData Corp., 384 South 400 West, Lindon, UT 84042
All rights reserved.

Verifying image...
Image verification complete.
root@kali:~/forensics# md5sum /dev/sdb >> wiped-md5.txt
root@kali:~/forensics# cat wiped-md5.txt
f9bd3c6e399e999408d35b9ce7a972f4  /dev/sdb
[MD5]
 Computed hash: 8a618d508e269f4b888bc0a937a832ae
[SHA1]
 Computed hash: cc8a26d525bcb950022c0040cda97527fbd59b93
ea8641a293b187421ff415af90926e52  /dev/sdb
root@kali:~/forensics#
```

4) The two generated hashes are different, whereas they are supposed to be the same. A possible hypothesis can be related to the tools and the way they read the device.

Task 2

1) We generate the MD5 hash value of Disk-2 using md5sum command, and we save the result to a file named “disk2md5.txt”
Command: *md5sum Disk-2 > disk2md5.txt*
We obtain the hash value: *ee9d3a1e82f0a80584b1686a931aca5a*
We can eventually verify this value with FTK Imager, and obtain the same hash value
Command: *./ftkimager Disk-2 --verify*

```
root@kali:~/forensics# md5sum Disk-2 > disk2md5.txt
root@kali:~/forensics# cat disk2md5.txt
ee9d3a1e82f0a80584b1686a931aca5a  Disk-2
root@kali:~/forensics# ./ftkimager Disk-2 --verify
AccessData FTK Imager v3.1.1 CLI (Aug 24 2012)
Copyright 2006-2012 AccessData Corp., 384 South 400 West, Lindon, UT 84042
All rights reserved.

Verifying image...
Image verification complete.
[MD5]
 Computed hash: ee9d3a1e82f0a80584b1686a931aca5a
[SHA1]
 Computed hash: af826f60e3da8ffbbbbd6b324d01a16a6a37a7b9
```

2) We can use FTK Imager to convert the raw image to E01 format. We would then specify additional case related information, such as case number and examiner

Command: *./ftkimager Disk-2 Disk-2-copy --e01 –case-number HW001 --evidence-number HW001-001 --examiner "Ramy Chemak" --description "E01 image file" --notes "Copy of Disk-2 raw image"*

```
 Computed hash: af826f60e3da8ffbbbbd6b324d01a16a6a37a7b9
root@kali:~/forensics# ./ftkimager Disk-2 Disk-2-copy --e01 --case-number HW001 --evidence-number HW001-01 --examiner "Ramy Chemak" --description "E0
1 image file" --notes "Copy of Disk-2 raw image"
AccessData FTK Imager v3.1.1 CLI (Aug 24 2012)
Copyright 2006-2012 AccessData Corp., 384 South 400 West, Lindon, UT 84042
All rights reserved.

Creating image...
Image creation complete.
root@kali:~/forensics# ls
chrisjonesproject1.pdf  Disk-2  Disk-2-copy.E01  Disk-2-copy.E01.txt  disk2md5.txt  ftkimager  LinuxWriteProtection.pdf
root@kali:~/forensics# du -sh Disk-2-copy.E01
7.0M    Disk-2-copy.E01
root@kali:~/forensics# cat Disk-2-copy.E01.txt | grep MD5
 MD5 checksum:    ee9d3a1e82f0a80584b1686a931aca5a
root@kali:~/forensics#
```

We can notice after the conversion is done, that the file "Disk-2-copy.E01" has been created, with an information file.

Task 3

We use FTK Imager to restore Disk-2 raw image to the USB device, with verification option enabled.
Command: ./ftkimager *Disk-2 /dev/sdb --verify*
We obtain the verification hash value: *ee9d3a1e82f0a80584b1686a931aca5a*
The verification hash value generated match the MD5 hash previously calculated.

```
root@kali: ~/forensics
root@kali:~/forensics# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   25G  0 disk
├─sda1   8:1    0 22.2G  0 part /
└─sda2   8:2    0  2.9G  0 part [SWAP]
sdb      8:16   1   15G  0 disk
sr0     11:0    1 1024M  0 rom
root@kali:~/forensics# ./ftkimager Disk-2 /dev/sdb --verify
AccessData FTK Imager v3.1.1 CLI (Aug 24 2012)
Copyright 2006-2012 AccessData Corp., 384 South 400 West, Lindon, UT 84042
All rights reserved.

Creating image...
Image creation complete.
Verifying image...
Image verification complete.
[MD5]
 Computed hash: ee9d3a1e82f0a80584b1686a931aca5a
 Report hash:   ee9d3a1e82f0a80584b1686a931aca5a
 Verify result: Match
[SHA1]
 Computed hash: af826f60e3da8ffbbbbd6b324d01a16a6a37a7b9
 Report hash:   af826f60e3da8ffbbbbd6b324d01a16a6a37a7b9
 Verify result: Match
root@kali:~/forensics#
```

1) We use **hdparm** program to apply write protection on the USB device previously prepared.
Command: *hdparm -r1 /dev/sdb*

```
root@kali:~# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   25G  0 disk
├─sda1   8:1    0 22.2G  0 part /
└─sda2   8:2    0  2.9G  0 part [SWAP]
sdb      8:16   1   15G  0 disk
sr0     11:0    1 1024M  0 rom
root@kali:~# hdparm -r1 /dev/sdb

/dev/sdb:
 setting readonly to 1 (on)
 readonly      =  1 (on)
root@kali:~#
```

2) We use FTK Imager to acquire data from the USB stick and save to a E01 image file called "evidence". We also enable the verification option.
Command: *./ftkimager /dev/sdb evidence --e01 --verify --case-number HW001 –evidence-number HW001-002 --description "E01 image of evidence 2" --examiner "Ramy Chemak" --notes "USB image acquired after Disk-2 file was restored to device"*

```
root@kali:~/forensics# ./ftkimager /dev/sdb evidence --e01 --verify --case-number HW001 --evidence-number HW001-02 --description "E01 image of eviden
 2" --examiner "Ramy Chemak" --notes "USB image acquired after Disk-2 file was restored to device"
AccessData FTK Imager v3.1.1 CLI (Aug 24 2012)
Copyright 2006-2012 AccessData Corp., 384 South 400 West, Lindon, UT 84042
All rights reserved.

Creating image...
Image creation complete.
Verifying image...
Image verification complete.
[MD5]
 Computed hash: af3740b0ab19d6b7a6fe91414edbb793
 Image hash:    3b45eced7c1dbb685d0d908efcfdebf0
 Report hash:   3b45eced7c1dbb685d0d908efcfdebf0
 Verify result: Mismatch
[SHA1]
 Computed hash: cfa0b11e042c8c88e8d68844efdb0d13d0a86797
 Image hash:    884d242c1f02ad792043e546824f4f32f5348f1c
 Report hash:   884d242c1f02ad792043e546824f4f32f5348f1c
 Verify result: Mismatch
root@kali:~/forensics# lsblk
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0   25G  0 disk
├─sda1   8:1    0 22.2G  0 part /
└─sda2   8:2    0  2.9G  0 part [SWAP]
sdb      8:16   1   15G  1 disk
sr0     11:0    1 1024M  0 rom
root@kali:~/forensics# ls
chrisjonesproject1.pdf  Disk-2-copy.E01      disk2md5.txt  evidence.E01.txt  LinuxWriteProtection.pdf
Disk-2                  Disk-2-copy.E01.txt  evidence.E01  ftkimager         wiped-md5.txt
root@kali:~/forensics#
```

Once the data acquisition is done, we can verify via the MD5 hash values if the acquisition was successful.

Task 4

1) We use Sleuthkit to analyse the E01 image acquired previously from the USB evidence.
Command: *mmls Disk-2-copy.E01*
Partitioning type used is GPT
We can see 4 allocated partitions on the device, in addition to the 3 meta volumes.

```
root@kali:~/forensics# mmls Disk-2-copy.E01
GUID Partition Table (EFI)
Offset Sector: 0
Units are in 512-byte sectors

      Slot      Start        End          Length       Description
000:  Meta      0000000000   0000000000   0000000001   Safety Table
001:  -------   0000000000   0000002047   0000002048   Unallocated
002:  Meta      0000000001   0000000001   0000000001   GPT Header
003:  Meta      0000000002   0000000033   0000000032   Partition Table
004:  000       0000002048   0000114687   0000112640
005:  001       0000114688   0000526335   0000411648
006:  002       0000526336   0000727039   0000200704
007:  003       0000727040   0002095103   0001368064
008:  -------   0002095104   0002097151   0000002048   Unallocated
```

As it's a GPT partitioning, we can display the protective MBR with the **hexdump** command, in order to learn more about about the present partitions.
Command: *hexdump -C -s 0 -n 640 Disk-2-copy.E01*

```
root@kali:~/forensics# hexdump -C -s 0 -n 640 Disk-2-copy.E01
00000000  45 56 46 09 0d 0a ff 00  01 01 00 00 00 68 65 61  |EVF..........hea|
00000010  64 65 72 00 00 00 00 00  00 00 00 00 00 d8 00 00  |der.............|
00000020  00 00 00 00 00 cb 00 00  00 00 00 00 00 00 00 00  |................|
00000030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000050  00 00 00 00 00 0d 04 ff  fc 78 01 6d 8a c1 0e 82  |.........x.m....|
00000060  30 10 05 cf af 5f b1 3f  50 b2 5b ca 81 a3 01 12  |0...._.?P.[.....|
00000070  bd 7a f1 bc 31 ad 36 58  4a 50 31 fc bd 46 af 5e  |.z..1.6XJP1..F.^|
00000080  26 93 c9 88 c9 9a 26 73  c6 04 45 c0 03 ba a2 ac  |&.....&s..E.....|
00000090  c8 78 62 c6 62 f6 27 66  c1 97 f6 23 03 0b a5 ac  |.xb.b.'f...#....|
000000a0  97 40 31 dd 02 8e 9a 37  ea ae 21 eb 88 ae cc 1b  |.@1....7..!.....|
000000b0  95 48 7d ba 8f d6 d1 a2  af df 89 5d 7f a8 e1 2b  |.H}........]...+|
000000c0  69 2b b6 0c c7 d2 92 30  79 92 9a 7c 43 ae f9 97  |i+.....0y..|C...|
000000d0  18 d1 bc 01 2b 5e 25 66  68 65 61 64 65 72 00 00  |....+^%fheader..|
000000e0  00 00 00 00 00 00 00 00  a3 01 00 00 00 00 00 00  |................|
000000f0  cb 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
00000100  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00000120  d9 03 9e f1 78 01 6d 8a  c1 0e 82 30 10 05 cf af  |....x.m....0....|
00000130  5f b1 3f 50 b2 5b ca 81  a3 01 12 bd 7a f1 bc 31  |_.?P.[......z..1|
00000140  ad 36 58 4a 50 31 fc bd  46 af 5e 26 93 c9 88 c9  |.6XJP1..F.^&....|
00000150  9a 26 73 c6 04 45 c0 03  ba a2 ac c8 78 62 c6 62  |.&s..E......xb.b|
00000160  f6 27 66 c1 97 f6 23 03  0b a5 ac 97 40 31 dd 02  |.'f...#.....@1..|
00000170  8e 9a 37 ea ae 21 eb 88  ae cc 1b 95 48 7d ba 8f  |..7..!......H}..|
00000180  d6 d1 a2 af df 89 5d 7f  a8 e1 2b 69 2b b6 0c c7  |......]...+i+...|
00000190  d2 92 30 79 92 9a 7c 43  ae f9 97 18 d1 bc 01 2b  |..0y..|C.......+|
000001a0  5e 25 66 64 69 73 6b 00  00 00 00 00 00 00 00 00  |^%fdisk.........|
```

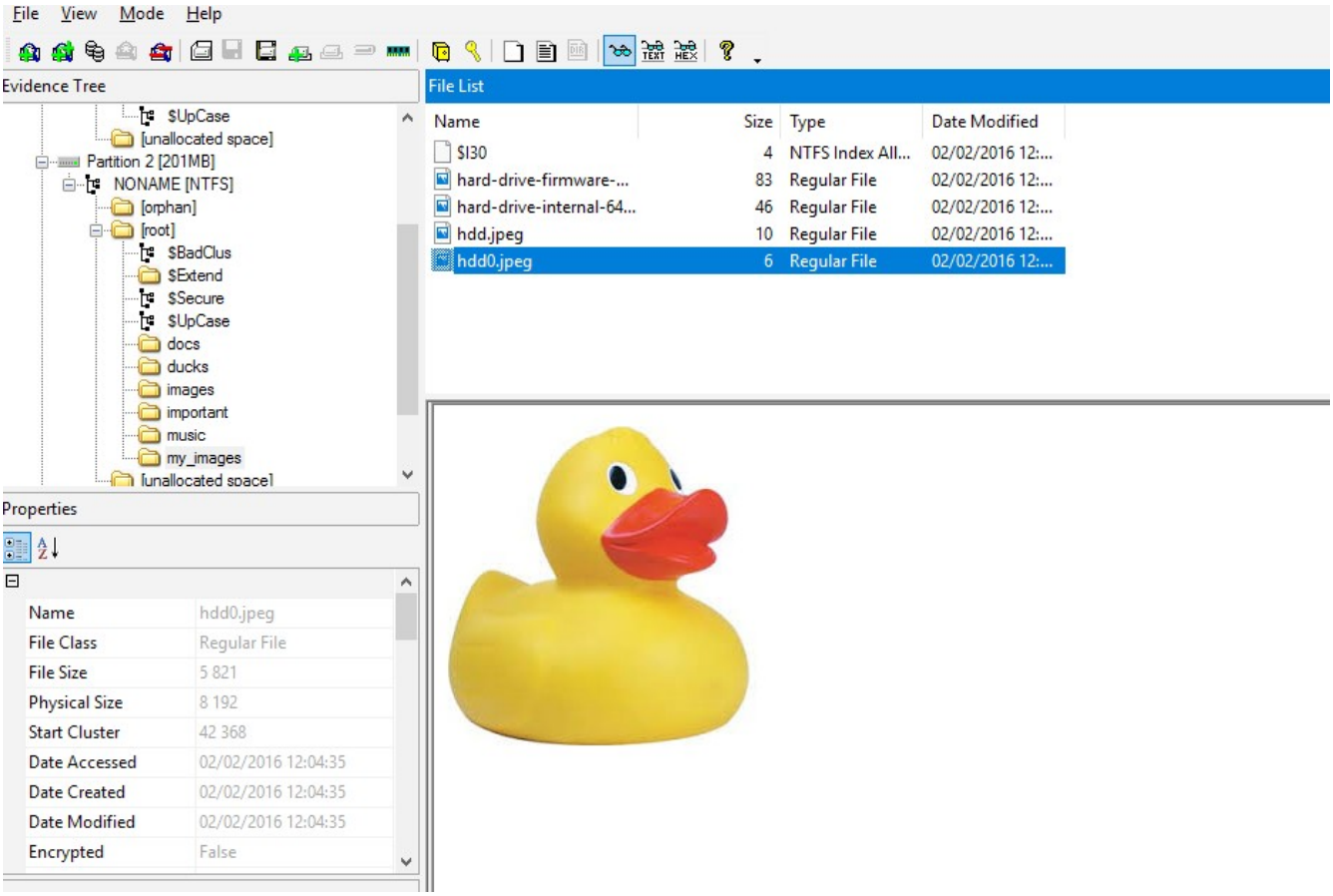
We can also display the GPT header relying on general information collected on the device. According to the general analyse, the GPT table start with an offset of 1 sector (512 bytes) and has a length of 1 sector too.
By using the **fsstat** utility, we can recognize the NTFS file system present on the device. Using FTK Imager on Windows platform later, we can see that the fourth partition is formatted with FAT32.
Command: *fsstat -o 2048 Disk-2-copy.E01*

```
root@kali:~/forensics# fsstat -o 2048 Disk-2-copy.E01
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: NTFS
Volume Serial Number: 39887A935B003F7E
OEM Name: NTFS
Version: Windows XP

METADATA INFORMATION
--------------------------------------------
First Cluster of MFT: 4
First Cluster of MFT Mirror: 7039
Size of MFT Entries: 1024 bytes
Size of Index Records: 4096 bytes
Range: 0 - 27
Root Directory: 5

CONTENT INFORMATION
--------------------------------------------
Sector Size: 512
Cluster Size: 4096
```

2) By using FTK Imager on Windows platform, we can find out more information about the device, as well as stored files.
We can find among other files, the picture of the duck. The picture is stored in the second partition, which is a NTFS partition.
The file size is of 5821 bytes. The creation date (on the device) as well as the last accessed and modified date is 02/02/2016. The picture is located under [root2]/ducks/hdd0.jpeg, whereas [root2] refers to the root folder of the second partition.


File View Mode Help
Evidence Tree
$UpCase
[unallocated space]
Partition 2 [201MB]
NONAME [NTFS]
[orphan]
[root]
$BadClus
$Extend
$Secure
$UpCase
docs
ducks
images
important
music
my_images
[unallocated space]
File List
Name Size Type Date Modified
$I30 4 NTFS Index All... 02/02/2016 12:...
hard-drive-firmware-... 83 Regular File 02/02/2016 12:...
hard-drive-internal-64... 46 Regular File 02/02/2016 12:...
hdd.jpeg 10 Regular File 02/02/2016 12:...
hdd0.jpeg 6 Regular File 02/02/2016 12:...
Properties
Name hdd0.jpeg
File Class Regular File
File Size 5 821
Physical Size 8 192
Start Cluster 42 368
Date Accessed 02/02/2016 12:04:35
Date Created 02/02/2016 12:04:35
Date Modified 02/02/2016 12:04:35
Encrypted False