SIT282/SIT703 Computer Forensics and Investigations

Workshop Session 3

Through this session, you will practise four forensic software tools – dd, dcfldd, hashcalc and foremost. By solving forensic tasks, you will be able to acquire dd images, recover files from the images, and verify the images. We will keep using the Ubuntu virtual machine.

Learning Objectives

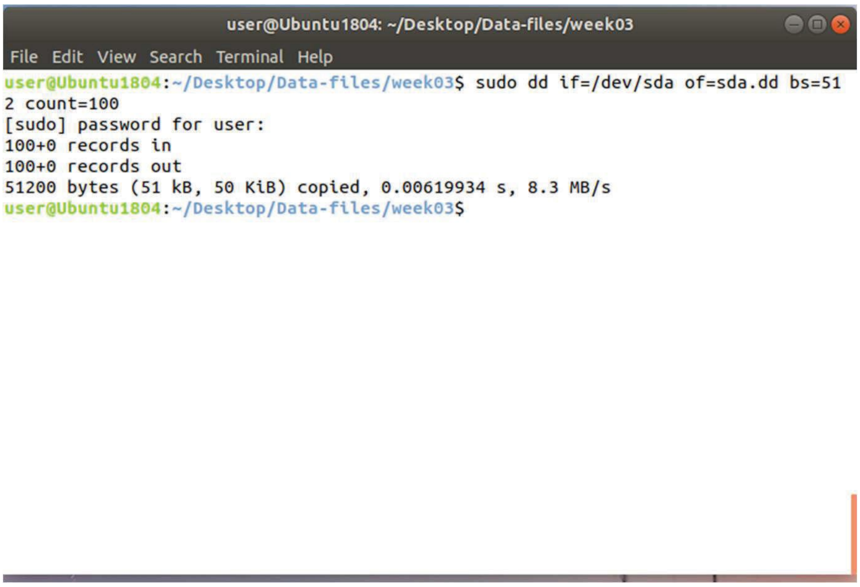
1. Explain the purpose of dd and its usage in digital forensics.
2. Demonstrate that you understand and can use the Linux commands “dd” and “dcfldd”.
3. Describe the major difference between “dd” and “dcfldd”, and explain which one is more suitable for digital forensics acquisition and why.
4. Explain why we need to calculate at least 2 different checksum values using different algorithms for a digital forensic image.
5. Explain whether making a change to a file name will result in the hash value changing.
6. Demonstrate the use of a CLI tool for recovering deleted files.
7. Demonstrate that you can acquire memory for digital forensic analysis.

1. Introducing dd and dcfldd

DiskDump is also known as dd. The tool dd has been used as a Unix/Linux system tool for a very long time. The primary purpose is to convert and copy files. We can use it for acquisition in digital forensics since it provides us with a bit-stream image of the original evidence disk to create the target file for analysis. Before digital forensics became popular, many system admins used dd for various purposes. Even today, many disk imaging tools follow many principles that dd offers.

Launch a “Terminal” and change directory to “~/Desktop/Data-files/week03”. We will use the tool dd to acquire the contents of some disk sectors of the virtual machine. Type command “**sudo dd if=/dev/sda of=sda.dd bs=512 count=100**” to acquire the first 100 sectors of the disk drive of the virtual machine. When prompted for password, type the password “user”.

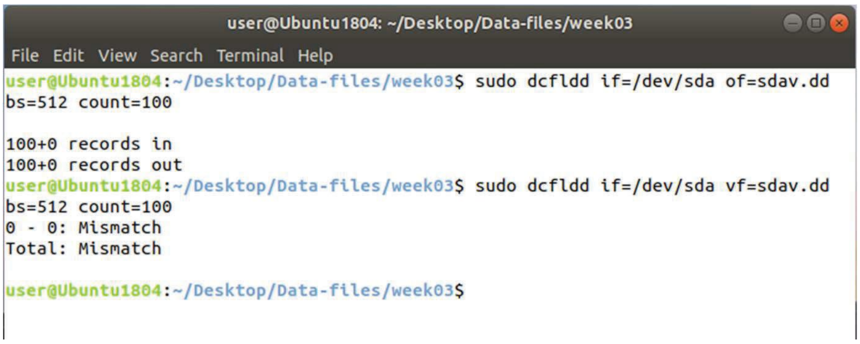
Note: ‘~’ pronounced tilde is a Linux short cut to a user’s home directory. For the path “~/Desktop/Data-files/week03” “~/” is the start of the path to a file or directory below the user’s home directory in this case /home/user. We can use “~/Desktop/Data-files/week03” rather than specifying the full path: /home/user/Desktop/Data-files/week03.


user@Ubuntu1804: ~/Desktop/Data-files/week03
File Edit View Search Terminal Help

```
user@Ubuntu1804:~/Desktop/Data-files/week03$ sudo dd if=/dev/sda of=sda.dd bs=51
2 count=100
[sudo] password for user:
100+0 records in
100+0 records out
51200 bytes (51 kB, 50 KiB) copied, 0.00619934 s, 8.3 MB/s
user@Ubuntu1804:~/Desktop/Data-files/week03$
```

Now, we finished dumping the first image. You can find some good examples of using the tool dd at https://opensource.com/article/18/7/how-use-dd-linux

Because the tool dd is essentially a system tool which lacks some features required by digital forensics such as auto-validating the integrity, we will introduce its sister tool dcfldd. It is almost used in the same manner as dd. When we acquire an image, dcfldd uses the same set of parameters as dd. But when we need to verify the image, we will use the option "vf" instead of "of". In the following screenshot, you can observe that the image "sdav.dd" has been verified with the source. (In case you want to know: dcfldd was written by Nicholas Harbour, who at the time was working for the Department of Defense Computer Forensics Lab (DCFL). Although he is no longer affiliated with the DCFL, Nick still maintains the package. The DCFL does not maintain, support, or have any other affiliation with dcfldd.)


user@Ubuntu1804: ~/Desktop/Data-files/week03
File Edit View Search Terminal Help

```
user@Ubuntu1804:~/Desktop/Data-files/week03$ sudo dcfldd if=/dev/sda of=sdav.dd
bs=512 count=100

100+0 records in
100+0 records out
user@Ubuntu1804:~/Desktop/Data-files/week03$ sudo dcfldd if=/dev/sda vf=sdav.dd
bs=512 count=100
0 - 0: Mismatch
Total: Mismatch

user@Ubuntu1804:~/Desktop/Data-files/week03$
```

We got a "Mismatch" because /dev/sda is currently in use. We should not dump a disk that is in use. Running "dcfldd" on a mounted disk or in-use disk could cause data loss or corruption, as it may modify the contents of the disk. We can scan the disk and check the status of disks by running the following command:

sudo fsck –f /dev/sda (You will see /dev/sda is in use)
sudo fdisk –l

This command will list all the available disk drives and their partitions. If a disk is mounted, its partition(s) will be listed under the “Mounted on” column. To unmount a disk, use the umount command, followed by the mount point of the partition you want to unmount. Once the disk is unmounted, we can safely run dcfldd on the disk without the risk of data loss or corruption.

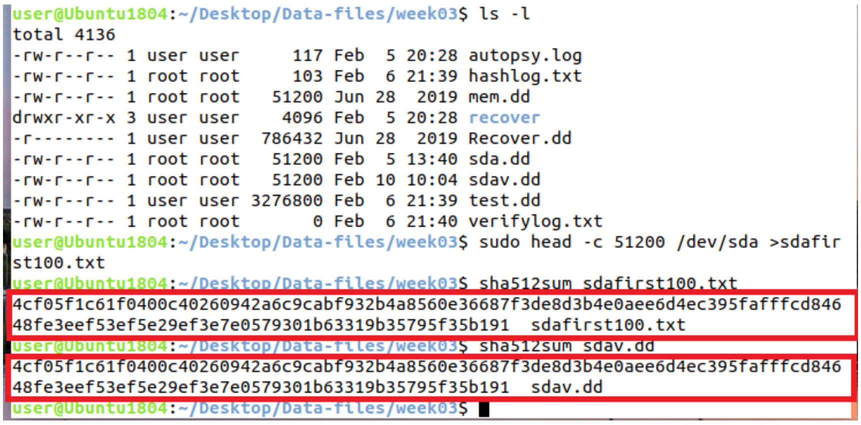
“Mismatch” in the context of dcfldd typically refers to a difference between the hash value of the data that was written to disk and the expected hash value. This could happen for a number of reasons, including:

1. Disk errors: If the disk being written to has errors, this could result in data being written incorrectly, which would cause the hash value to be incorrect.
2. Interrupted write process: If the dcfldd process is interrupted while writing data to disk, the data may not have been fully written and will result in an incorrect hash value.
3. Corrupted data source: If the source of the data being written to disk is corrupted, the resulting hash value will not match the expected value.
4. Incorrect hash value: Finally, it is possible that the expected hash value itself is incorrect.

To troubleshoot a mismatch in the output of dcfldd, it is recommended to check the integrity of the source data and verify that the expected hash value is correct. Additionally, you may want to try using a different disk or storage device to ensure that the issue is not with the disk itself.

In this case, we can compare the hash value (e.g. sha512) of the first 512*100 bytes of /dev/sda and that of the sdav.dd. We cannot directly sha512sum the first 512*100 bytes of /dev/sda, so we first extract that part to a file, then hash.

```
sudo head –c 51200 /dev/sda > sdafirst100.txt
sha512sum sdafirst100.txt
sha512sum sdav.dd
```



```
user@Ubuntu1804:~/Desktop/Data-files/week03$ ls -l
total 4136
-rw-r--r-- 1 user user     117 Feb  5 20:28 autopsy.log
-rw-r--r-- 1 root root     103 Feb  6 21:39 hashlog.txt
-rw-r--r-- 1 root root   51200 Jun 28  2019 mem.dd
drwxr-xr-x 3 user user    4096 Feb  5 20:28 recover
-r-------- 1 user user  786432 Jun 28  2019 Recover.dd
-rw-r--r-- 1 root root   51200 Feb  5 13:40 sda.dd
-rw-r--r-- 1 root root   51200 Feb 10 10:04 sdav.dd
-rw-r--r-- 1 user user 3276800 Feb  6 21:39 test.dd
-rw-r--r-- 1 root root       0 Feb  6 21:40 verifylog.txt
user@Ubuntu1804:~/Desktop/Data-files/week03$ sudo head -c 51200 /dev/sda >sdafir
st100.txt
user@Ubuntu1804:~/Desktop/Data-files/week03$ sha512sum sdafirst100.txt
4cf05f1c61f0400c40260942a6c9cabf932b4a8560e36687f3de8d3b4e0aee6d4ec395fafffcd846
48fe3eef53ef5e29ef3e7e0579301b63319b35795f35b191  sdafirst100.txt
user@Ubuntu1804:~/Desktop/Data-files/week03$ sha512sum sdav.dd
4cf05f1c61f0400c40260942a6c9cabf932b4a8560e36687f3de8d3b4e0aee6d4ec395fafffcd846
48fe3eef53ef5e29ef3e7e0579301b63319b35795f35b191  sdav.dd
user@Ubuntu1804:~/Desktop/Data-files/week03$
```

The hash values match.

Spend a few minutes to read through the information about the tool dcfldd at http://dcfldd.sourceforge.net/

Note: dd can be friend or foe when it comes to cyber security. Consider the following commands and determine what the outcome of using dd in these cases will be.

WARNING: DO NOT DO THIS ON YOUR OWN DEVICE HARD DISK AS IT WILL BE UNRECOVERABLE!
Unless of course it is your intention to delete everything from the drive.

```
dd if=/dev/zero of=dev/sda1
dd if=/dev/urandom of=dev/sda1
```

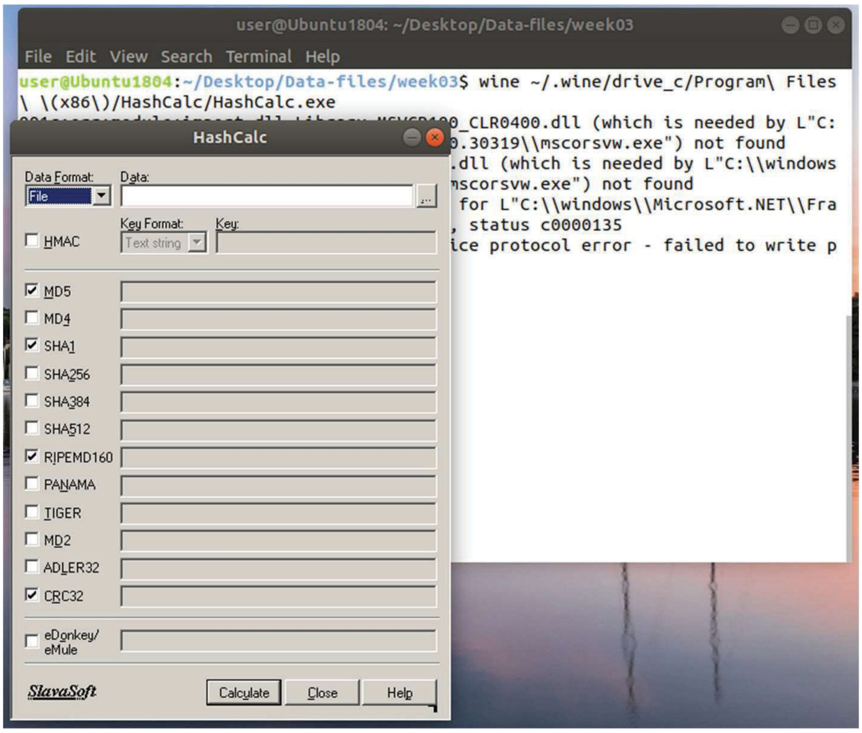
2. Using HashCalc to Get Hash Values

Hashcalc is a powerful tool for calculating hash values of any files. We have installed Hashcalc in the virtual machine. But it is a Windows executable, so we need to use the Windows emulator "wine" to execute the windows executables in Linux. You can read more about the "wine" at https://www.winehq.org. Essentially, it allows us to run many Windows applications in Linux, even some games.

Type the command:
"**wine ~/.wine/drive_c/Program\ Files\ \(x86\)/HashCalc/HashCalc.exe**" and the program will be launched.

NOTE: In the command ensure there is a space:
between wine and ~
after Program\
between the back slashes of Files\ \

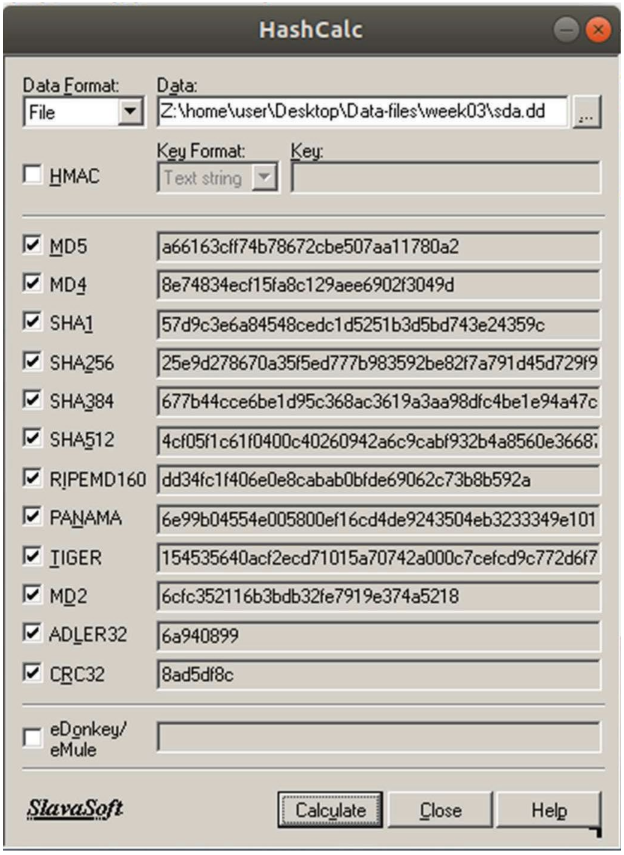
\ is used to escape space, "(" and ")" because of the folder name "Program Files (x86)". You can also use single or double quotes, e.g.,
wine ~/.wine/drive_c/"Program Files (x86)"/HashCalc/HashCalc.exe or
wine ~/.wine/drive_c/'Program Files (x86)'/HashCalc/HashCalc.exe


user@Ubuntu1804: ~/Desktop/Data-files/week03
File Edit View Search Terminal Help
HashCalc
Calculate
Close
Help
SlavaSoft

```
user@Ubuntu1804:~/Desktop/Data-files/week03$ wine ~/.wine/drive_c/Program\ Files
\ \(x86\)/HashCalc/HashCalc.exe
                                         0_CLR0400.dll (which is needed by L"C:
                                         0.30319\\mscorsvw.exe") not found
                                         .dll (which is needed by L"C:\\windows
                                         mscorsvw.exe") not found
                                          for L"C:\\windows\\Microsoft.NET\\Fra
                                         , status c0000135
                                         ice protocol error - failed to write p
```

Data Format: File
Data:
□ HMAC Key Format: Text string Key:
☑ MD5
□ MD4
☑ SHA1
□ SHA256
□ SHA384
□ SHA512
☑ RIPEMD160
□ PANAMA
□ TIGER
□ MD2
□ ADLER32
☑ CRC32
□ eDonkey/ eMule

We tick all the hash algorithm boxes, and then select the image “sda.dd” acquired in Section 1 for “Data” field. Click the “Calculate” button. The results are:



Data Format: File
Data: Z:\home\user\Desktop\Data-files\week03\sda.dd
□ HMAC Key Format: Text string Key:
☑ MD5 a66163cff74b78672cbe507aa11780a2
☑ MD4 8e74834ecf15fa8c129aee6902f3049d
☑ SHA1 57d9c3e6a84548cedc1d5251b3d5bd743e24359c
☑ SHA256 25e9d278670a35f5ed777b983592be82f7a791d45d729f9
☑ SHA384 677b44cce6be1d95c368ac3619a3aa98dfc4be1e94a47c
☑ SHA512 4cf05f1c61f0400c40260942a6c9cabf932b4a8560e3668
☑ RIPEMD160 dd34fc1f406e0e8cabab0bfde69062c73b8b592a
☑ PANAMA 6e99b04554e005800ef16cd4de9243504eb3233349e101
☑ TIGER 154535640acf2ecd71015a70742a000c7cefcd9c772d6f7
☑ MD2 6cfc352116b3bdb32fe7919e374a5218
☑ ADLER32 6a940899
☑ CRC32 8ad5df8c
□ eDonkey/ eMule

Spend a few minutes to calculate the hash values for the image “sdav.dd” and see whether they match.

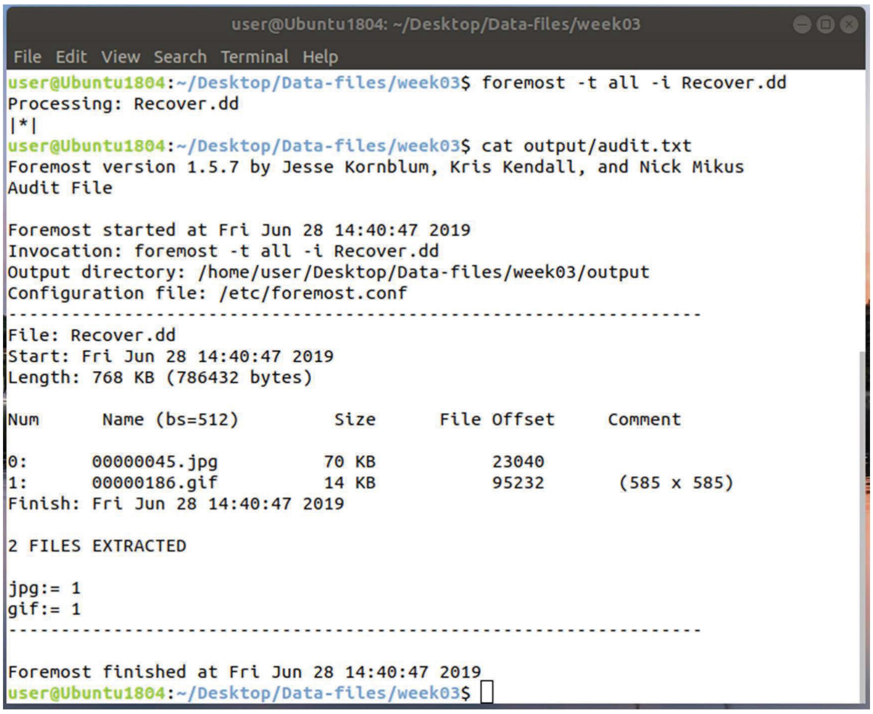
3. Recovering Files Using Foremost

One of the important tasks for the digital forensic investigators is to recover deleted files. With a dd image, we can use file recovery (or file carving) tools. Foremost is a simple tool (not the best, not the worst). We now show how to use it to quickly recover files from a dd image. (More advanced file recovery tools will be introduced later.)

NOTE: Check to see whether the “output” directory has already been created in your week03 folder. If yes, remove the directory and its contents. Type “**rm –r output/**”
-r specifies to remove directories and their contents recursively

Type the command “**foremost -t all -i Recover.dd**” and you will see two recovered files in the newly created folder “output”:

NOTE: type “man foremost” to view the descriptions for option –t and –i in the command.


user@Ubuntu1804: ~/Desktop/Data-files/week03
File Edit View Search Terminal Help

```
user@Ubuntu1804:~/Desktop/Data-files/week03$ foremost -t all -i Recover.dd
Processing: Recover.dd
|*|
user@Ubuntu1804:~/Desktop/Data-files/week03$ cat output/audit.txt
Foremost version 1.5.7 by Jesse Kornblum, Kris Kendall, and Nick Mikus
Audit File

Foremost started at Fri Jun 28 14:40:47 2019
Invocation: foremost -t all -i Recover.dd
Output directory: /home/user/Desktop/Data-files/week03/output
Configuration file: /etc/foremost.conf
------------------------------------------------------------------
File: Recover.dd
Start: Fri Jun 28 14:40:47 2019
Length: 768 KB (786432 bytes)

Num      Name (bs=512)         Size      File Offset     Comment

0:      00000045.jpg          70 KB          23040
1:      00000186.gif          14 KB          95232       (585 x 585)
Finish: Fri Jun 28 14:40:47 2019

2 FILES EXTRACTED

jpg:= 1
gif:= 1
------------------------------------------------------------------

Foremost finished at Fri Jun 28 14:40:47 2019
user@Ubuntu1804:~/Desktop/Data-files/week03$ 
```