Digital Forensics Masterclass Labsheet

File System Forensics

Fraser Brown - Software Engineering MEng

Contents

1 Lab Overview

The learning objective for this lab is for participants to gain practical experience in digital forensics specifically file system analysis. During the lab you will gain experience in the following tools, environments and tasks:

- Kali Linux[1]
- The Sleuth Kit (TSK)[2] file system analysis tool kit.
- Forensic image creation.
- Automated Data Carving using Foremost
- Identifying Attributes about file systems.
- Deleted file recover procedure.
- Searching for evidence on a file system.

As an *optional* bit of fun for the lab, I have added a capture the flag (CTF) style element. For each task there is a set goal, that may be finding a word or phrase in unallocated space, the cryptographic hash of a recovered file etc. After each task is completed please enter the given evidence as an argument to the `flagfound` script as follows:

```
$  ./flagfound "YOUR EVIDENCE HERE"
```

2 Removable Media Resource Credits

The USB resources we will be using in today's lab are from the National Institute of Standards and Technology (NIST) as part of their Computer Forensic Reference Data Sets (CFReDS)[3]. We will be focusing on a subsection of this case using "Removable Media #2", in order to investigate material covered in the lecture. The scenario section below will set the scene.

If you are interested in exploring this scenario more the whole case can be found here:

`https://www.cfreds.nist.gov/data_leakage_case/data-leakage-case.html`

[1] https://www.kali.org/
[2] https://www.sleuthkit.org/
[3] https://www.cfreds.nist.gov/

3 Environment Setup

Installing Kali Linux in Virtual Box

Before your digital investigation can begin you are required to create a lab environment, for this we will be using a virtual machine (VM) instance of Kali Linux. Your user accounts should have been given access to virtual box in the university lab (EM2.50) on machines `linxu02-linux30`. We are using a virtual machine to not only give you complete control over the operating system, but to also make sure this lab can be carried out safely. Below we will show you how to create a Kali Linux VM.

1. Open a terminal window and enter the following command to open Virtual Box:
   ```
   $ virtualbox &
   ```
2. Follow the instructions in: `df-loading-kali.pdf` document.
3. Start your Kali VM

4 Kali Linux Information

Login Information

username: `root` password: `toor`

Note About Root

In Kali Linux you are logged in as the root user of the system, therefore the operating system assumes you know what you are doing. Double check the commands you are entering especially if they are related to `rm`.

5 Lab Scenario

A client has contacted your digital forensics firm stating there has been an attempt to leak information from the company. The person accused of said information leakage has had various office technology collected, said items have been split among your hypothetical team. You have been assigned removable media # 2 - a 4GB USB stick. The following tasks will aid in evidence and hypothesis determination for the client.

6 Task 0: Create a Case Directory for Investigation

Within your home directory create the a `cases` directory which contains a directory called `001` to represent the case number you are working on. Make `cases/001/` this your current working directory in a terminal window. Finally within `cases/001/` create two more directories called `images` and `output`.

7 New Task 1: Download The Forensic Image

IMPORTANT: The original Task 1 (creating forensic image) is not feasible on the university machines due to the time it takes. Making a forensic image of the USB sticks takes far to long. On my own laptop VM it was completed in 5 minutes however the lab machines took 20+ min to get 20% through. Therefore, instead please run the following commands from a terminal within `cases/001/images/` directory:

```
# wget http://www2.macs.hw.ac.uk/~fmb30/digital-forensics-
    masterclass/cfreds_2015_data_leakage_rm%232.7z
# 7z x cfreds_2015_data_leakage_rm#2.7z
# rm cfcfreds_2015_data_leakage_rm#2.7z
```

You should now see the file `cfcfreds_2015_data_leakage_rm#2.dd` in the folder `cases/001/images/`. This is a forensic image of the USB drive in question. If you wish to do the **original** Task 1 and manually create a forensic image in your spare time, please feel free with your own USB sticks. *The old version of Task 1 can be found at the end of the document.* I apologise for creating forensic images not being doable within the lab time.

Please continue the lab from Task 2.

8 Task 2: Automated Data Carving

Data Carving is a process where files are recovered from a disk image based on common information such as file headers, footers and data structures. These can be viewed with a hex editor and changed if needed. Performing data carving for large forensic images can be rather

tedious if done by hand, tools such as **foremost** have been developed to help digital forensic investigators automating this process.

We will use **foremost** initially on our forensic image to perform automated data carving and file recovery, displaying tools that are used in industry. In later tasks we execute recovery methods by hand to reinforce what is going on at lower levels.

The following command will recover all files of various types such as `png, gif, docx, mp3` and output them into sub directories based on those file types.

```
# foremost -i [path-to-image] -o output/foremost/ -v
```

Get Your Flag:

Optional: *Generate a* ***sha1*** *hash of the* `output/formost/audit.txt` *file and enter it into the* `findflag` *program to receive your flag.*

9 Task 3: File System Information Gathering using TSK

Now that we have a forensic image from task 1 we can start to analyse the evidence, using The Sleuth Kit (TSK) command line tools.

Commands To Gather Information About The Media:

- **mmls**:
  `mmls` displays the layout of the partitions in a volume system, which include partition tables and disk labels.
- **fsstat**:
  `fsstat` displays the details associated with a file system. (requires partition starting point in sectors)

Find the following Information:

Using the tools above (`mmls, fsstat`) and files from task 1 (.dd, .info) find the following information (write it down you will need to reference it later):

1. What is the staring point of the File System partition (in sectors)?
2. What is the cluster size of the file system?
3. What is the sector size of the file system?
4. Given that a cluster contains consecutive sectors, how many sectors are in one cluster?
5. Can you confirm the file system type is fat32 using a TSK tool or other information you have gathered file?

Get Your Flag:

Optional: *Enter your answer to question 4 into the* `findflag` *program to receive your flag.*

10 Task 4: Searching Unallocated Space for Evidence

Our client has informed us that the person under investigation for information leaking worked on projects relating to NASA. This task will involve attempting to recover any files relating to the string "NASA". We will do so by searching unallocated space for evidence.

Commands To Search Unallocated Space for Evidence:

- **`blkls`**:

  `blkls` opens the named image(s) and copies file system data units (blocks). By default, `blkls` copies the contents of unallocated data blocks.
- **`fsstat`**:

  `fsstat` displays the details associated with a file system. (requires partition starting point in sectors)
- **`strings`**:

  `strings` print strings of printable characters in files.
- **`blkcalc`**:

  `blkcalc` converts between unallocated disk unit numbers and regular disk unit numbers
- **`ifind`**:

  `ifind` Find the meta-data structure that has allocated a given disk unit or file name
- **`istat`**:

  `istat` displays the uid, gid, mode, size, link number, modified, accessed, changed times, and all the *disk units* a structure has allocated.

Task Walkthrough

Task 4.1: Retrieve unallocated space from disk image:

In order to efficiently search through the unallocated space in our forensic image we must first extract a copy of all unallocated space in our forensic image. The command below:

```
# blkls -o [file-system-starting-location] [path-to-image] >
output/unallocated.blkls
```

Task 4.2: Extract Searchable Data From Unallocated Space:

To provide searchable data we can now extract all strings from our `unallocated.blkls` file. Note: we use the `-t d` option to extract the byte location of the string in decimal format. This will allow our unallocated space to be parsed by commands like `grep`.

```
# strings -t d output/unallocated.blkls > output/unallocated.str
```

Task 4.3: Find Byte Location For The First Instance of String "NASA" :

`grep` this unallocated.str file to locate the byte location of the first instance of "NASA".

```
# grep  -e "NASA" output/unallocated.str
```

4.3 Questions:

1. What is the byte location address for the first instance of "NASA"?

Task 4.4: Calculate Sector & Cluster/Data Unit Location :

With this byte location of our desired string we can calculate where in memory this location is by working our way up from byte to sector to cluster address. Task 3 provided us with information regarding the size of sectors on this file system, using that number we can perform the following calculation to retrieve the sector address:

```
BYTE_LOCATION / SECTOR_SIZE = SECTOR_LOCATION
```

The `blkcalc` TSK tool will allow us to convert this sector location into a cluster address:

```
# blkcalc -u [SECTOR_LOCATION] -o [file-system-starting-location] [path-to-image]
```

4.4 Questions:

1. What is the Sector Location?
2. What is the Data Unit/Cluster Location?

Task 4.5: Find A Meta Data Structure for this Data Unit (Cluster):

Now we have more address information about where on within the file system and physical memory this instance of the "NASA" string is located. We can look to see if there is any meta data structures for this section of memory. `ifind` with a `-a` flag will return all meta data structures for a given data unit. `-d` is where we enter the cluster/data unit location we calculated earlier.

```
# ifind -a -o [file-system-starting-location] -d [data-unit-location] [path-to-image]
```

4.5 Questions:

1. What is the meta data (inode) structure location?

Task 4.6: Extract Further Information About The Meta Data Structure

We can view more information about the files associated with this meta data structure such as file size, last written, read and modified times etc.

```
 # istat -o [file-system-starting-location] [path-to-image] [inode-location] > output/nasa.istat
```

Task 4.7: Confirm The File Name

For thoroughness confirm the file name from the file name structure.

```
# ffind -a -o [file-system-starting-location] [path-to-
image] [inode-location]
```

Task 4.8: Extract NASA file, check file type, open it.

Using the information we have gathered during this task we can use `icat` to extract the information based on the metadata (inode) address, `file` to confirm the actual file type and then open the document to confirm we have recovered the data successfully.

```
#icat -o [file-system-starting-location] [path-to-image] [inode-location]
> output/nasa.icat
# file output/nasa.icat
# libreoffice nasa.icat
```

Get Your Flag:

Optional: *Enter the data unit (cluster) address for the first "nasa" string into the* `findflag` *program to receive your flag.*

11 Task 5: Deleted File Recovery using Meta Data Pointers

Another member of your digital investigation team has mentioned there is something suspicious with a file called `progress/my_friends.svg` that was removed from the USB, they have asked you to to check if the deleted file is recoverable and verify the data is of format `.svg` then compare the recovered data to his SHA1 hash.

Co-Workers SHA1 hash:

```
95175c35be9d74ad3e271550e358e634154ba105 progress/my_friends.svg
```

Commands Used in This Task:

- **`fsstat`**:

  `fsstat` Displays the details associated with a file system. (requires partition starting point in sectors).

- **`fls`**:

  `fls` List file and directory names in a disk image, including its associated *inode* number.

- **istat**:

  `istat` displays the uid, gid, mode, size, link number, modified, accessed, changed times, and all the *disk units* a structure has allocated.
- **blkstat**: displays the allocation status of the given data unit.
- **blkls**:

  `blkls` opens the named image(s) and copies file system data units (blocks). By default, blkls copies the contents of unallocated data blocks.
- **sha1sum**: `sha1sum` - compute and check SHA1 message digest
- **icat**:

  icat Outputs the content of a file based on its *inode* number according to the details in its meta data pointer.
- **file**: file - determine file type

Task Walkthrough:

Task 5.1: List File Names and Meta Data (inode) Info of Deleted Files On The Image:

We are looking for a deleted file and have been given a name to find within this .dd image. We know that data in file systems have various pointers and structures for meta data, and file names. We can use TSK tools that begin with an `f` to do so. In this instance we want to see all files that were deleted on the data partition, `fls` will let us do this with the `-rd` flags to print recursively and only print deleted files.

```
# fls -rd -o [file-system-starting-location] [path-to-image]
```

5.1 Questions:

1. What is the *iNode Address* for this `my_friends.svg`?

Task 5.2: View Stats On the Meta Data (inode) Address For my_friends.svg:

Using this inode (meta data) pointer address, we can gather stats on the data units in question using `istat`.

```
# istat -o [file-system-starting-location] [path-to-image] [inode-address]
```

5.2 Questions:

1. What is the file size of the Image?
2. What sector does the data unit start at?
3. what sector does the data unit end at?

Task 5.3: Retrieve Data Units (Cluster) From Sector Addresses:

This will not only allow provide us more information about the files location within the memory and file system but it will allow us to verify that the memory locations have not been overwritten.

```
# blkstat -o [file-system-starting-location] [path-to-image] [start-sector
-address]
# blkstat -o [file-system-starting-location] [path-to-image] [end-sector
-address]
```

4.3 Questions:

1. What is difference between start and end clusters?

Task 5.4: Recover those sectors from unallocated space:

Given we know the start and end sectors and have verified that their clusters are not overwritten with other data, we can begin the recovery process.

`blkls` will extract all unallocated data units between a given start and end point and write it to a new file (if no points are given it will gather all unallocated space in the given partition).

```
# blkls -o [file-system-starting-location] [path-to-image] [start-stop]
> output/recovered-file.blkls
```

Task 5.4: Calculate hash for recoveredfile.blkls:

We have successfully recovered the requested file now verify that its hash matches the one our co-worker found as follows:

```
# sha1sum output/recovered-file.blkls
```

Task 5.5: Check the file type of the recovered data units:

Verify it is a `.svg` file with the file command:

```
# file output/recoved-file.blkls
```

5.5 Questions:

1. What is actual type of the file?
2. Do you know why the file extension contradicts the actual type?

Get Your Flag:

Optional: *Generate a* ***md5*** *hash of the* `output/formost/audit.txt` *file and enter it into the* `findflag` *program to receive your flag.*

12 Old Task 1: Create a Forensic Image

This is the old task 1 that walks you through createing a forensic image from a USB stick. Due to time constraints and execution time on university machines it is advised that you skip this task during the lab, but feel free to try it out in your own time.

There should be USB flash drives available for this lab with the removable media evidence on them. In this lab we will be working with .dd images also known as raw images. There are many tools available to create the a forensic image, Kali Linux comes installed with both GUI and command line tools. To get the ball rolling we will use a GUI based tool called `Guymager`.

NOTE: *Sadly we were not able to attain write blockers for this lab, please remember that write blockers are important for prevention of unwanted meta data changes to media by the investigators operating system. Write blockers are used in industry when creating forensic images of a system.*

Insert USB Stick into computer and confirm it has been recognised by the Kali Linux VM, you can check this in either the files (nautilus) program by checking a USB device is visible in the left hand pane. Or by running `df -h` in the command line and checking for a 4GB device located in `/media/root/`.

Open a terminal window (we will use this later anyway) and enter: `guymager &` the following GUI in Figure 1 will appear:

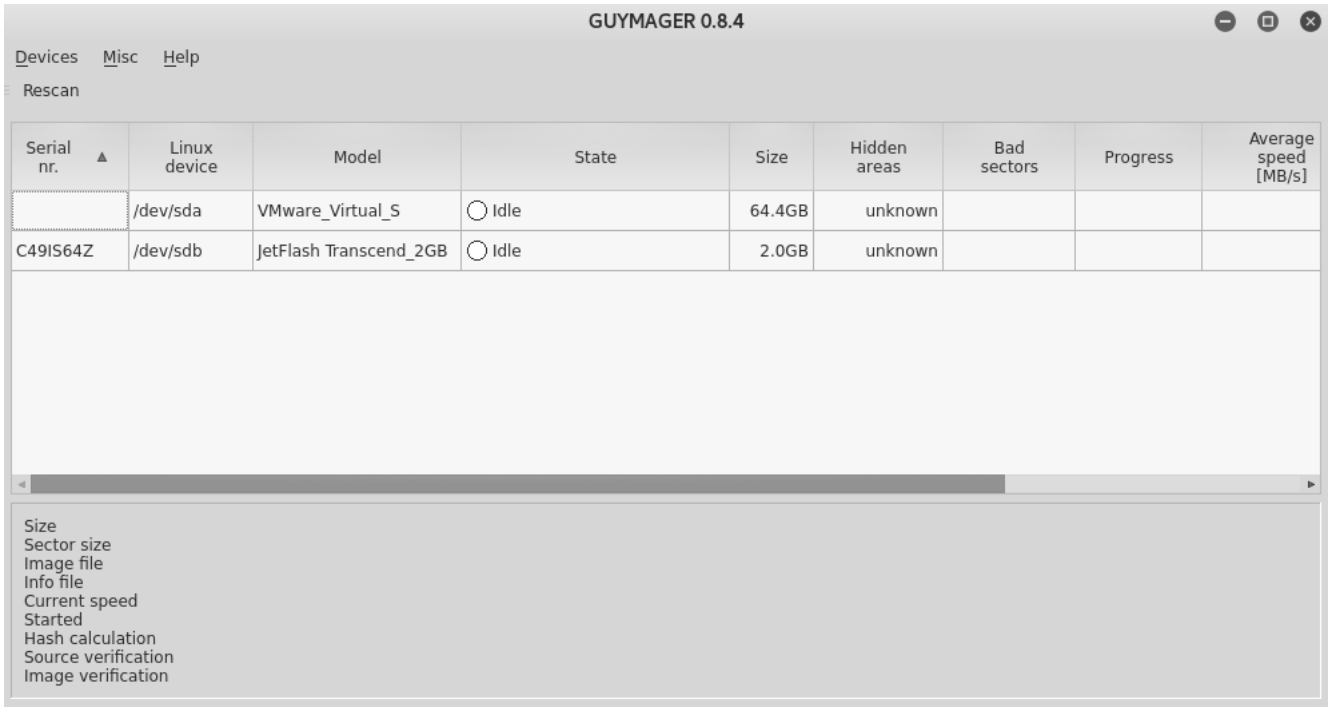

Figure 1: *guymager gui window example*

If the 4GB USB drive does not show up press `f5` to rescan devices. Select the drive you wish to create an image of in our case right click the 4GB drive and select the `acquire image` option. Fill out the displayed form making sure you have the following options selected as per the list below and/or Figure 2 then click start:

- Linux dd raw image
- Uncheck split image
- Destination is set to `/home/cases/001/image/`
- Image filename set to `revmed2`
- Info filename set to `revmed2`
- Check calculate md5 hash
- Check verify image after acquisition

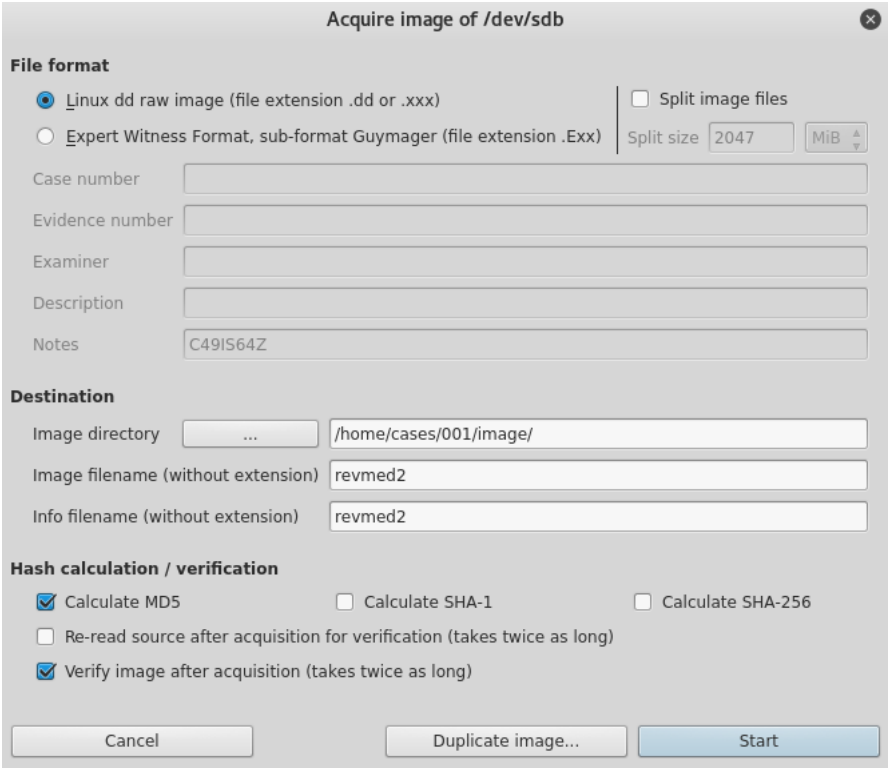

Figure 2: *guymager acquisition window example*

Once you have clicked start you will be returned to the main `guymager` window and a progress bar should be displayed. Once the progress bar states "Finished - Verified & ok" your forensic image have been created. To confirm list the contents of `/home/cases/001/image/` in a terminal window, you should find the following files:

- `revmed2.dd` : the image file we can now perform analysis on.
- `revmed2.info` : acquisition info such as cryptographic hash.

Verify The Media Hash

In order to confirm that our.dd image is the same as the contents of the USB, we must generate a hash of the .dd file and compare it to the hash listed in the .info file.
In a terminal window run the following:

```
# md5sum image/revmed2.dd
```

The output should match the hash listed in the .info file you just created.

Remove the USB from your machine and return it to the evidence box.

Get Your Flag:

Optional: *Enter the hash into the* `findflag` *program to receive your first flag.*