

CSEC 730 - Advanced Computer Forensics

Homework 1 – Disk Imaging and Linux Memory Dump

Please submit your answers **in PDF format** to the assignment submission folder on *myCourses* > *Assignments* by the deadline.

This homework comprises three (3) parts.

Part 1. Disk imaging using *FTK Imager*

Software

Download the latest version of the free Windows software **FTK Imager** and **FTK Imager USER GUIDE** on your own Windows system from: <https://www.exterro.com/> (choose FTK Imager). Read the USER GUIDE before starting part 1.

USB Drive

For this activity, you will use a USB drive with at least three files of any format on it. You can use any size drive, but **using one that is 2 GB or less will keep the imaging process from being too long**. To make this activity realistic, do not use a brand-new drive, instead, use one that you have loaded and deleted files from over time.

Goal: We have learned that bitstream copies make a bit-for-bit copy of all sectors on a drive. In this activity, you will use a well-known forensics imaging tool, FTK Imager, to create a bitstream image of your USB drive and examine the results.

NOTE: FTK Imager does not guarantee that the drive is not modified during imaging. For this reason, investigators will use a write blocker when using FTK Imager in a real case. In this exercise, you will use FTK Imager ONLY for learning FTK Imager's features without a USB write blocker.

A. Create the USB image

Insert your USB into your Windows system and launch the FTK imager.

Following the following steps, create an image of your USB drive in Raw (dd) format and save the copy to your desktop. FTK imager will also generate hashes of the image.

- Select File -> Create Disk Image...
- Choose Physical Drive
- Choose your USB Device
- Press Finish.
- Add the image destination.
- Select Raw (dd) as format.
- Provide destination folder and image filename information.
- Press Start

Task 1. Complete the following multiple-choice questions by highlighting the best answers. (6 points)

- After the imaging process was complete, what files did FTK Imager create? (Select all that apply.)
 - The image file with an extension of .001**
 - A text file for image summary**
 - An image file with an extension of .Ex01
 - No files
- During the imaging process, you should have noticed that "Verify images after they are created" is checked by default. What is the result of having this option checked? (choose a single best answer)
 - FTK imager will compute the hash value of the image
 - FTK imager will compute the hash value of the USB drive
 - FTK imager will compute the MD5 and SHA1 hashes of the USB drive and the MD5 and SHA1 hashes of the image, and verify the hashes match.**
 - FTK imager will compute the MD5 hash of the USB drive and the MD5 hash of the image, and verify the hashes match
- How many hash algorithms did the FTK imager use to verify the image has not been altered? (choose a single best answer)
 - One hash algorithm
 - Two hash algorithms**
 - Three hash algorithms
 - Four hash algorithms

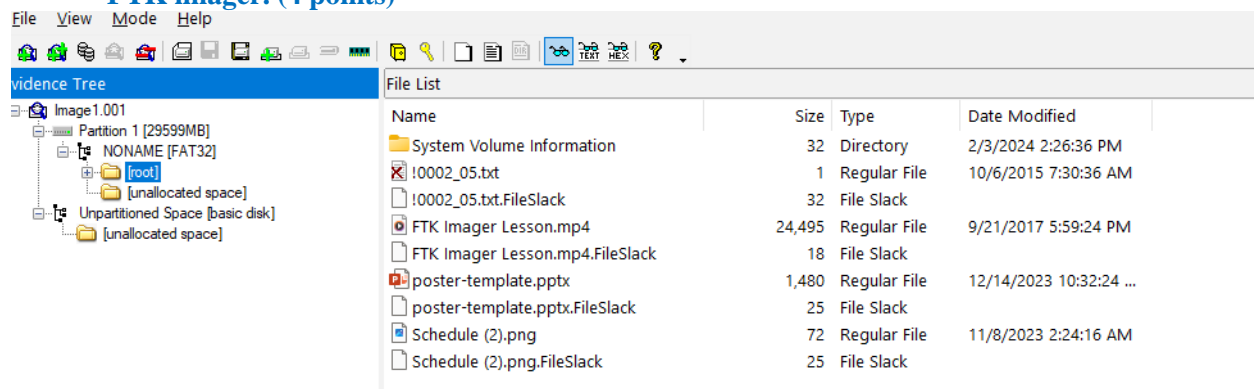
B. View the acquired USB image

In this exercise, you will load your USB image to FTK imager, and examine the content.

File -> Add Evidence Item...

Under *Select the Source Evidence Type*, choose Image File (since we created an image in part A).

Task 2. Provide a screenshot of the FTK imager after you have loaded your USB image into the FTK imager. (4 points)



Task 3. Explore the features supported in FTK Imager, discuss one FTK Imager feature, and include a screenshot. (10 points)

The "Export Directory Listing" feature in FTK Imager allows users to create a comprehensive and detailed list of all files and directories contained within a forensic image or a specific device. This feature generates a report which includes critical information about each file and directory, such as their names, paths, sizes, and timestamps (creation, modification, and last accessed dates). This directory listing can be exported into a format that is easy to read and analyze, such as a CSV (Comma Separated Values) file.

A	B	C	D	E	F	G
Filename	Full Path	Size (bytes)	Created	Modified	Accessed	Is Deleted
[root]	Partition 1\NONAME	32768				no
VBR	Partition 1\NONAME	512				no
reserved sectors	Partition 1\NONAME	811520				no
[unallocated space]	Partition 1\NONAME	0				no
FAT1	Partition 1\NONAME	3788288				no
FAT2	Partition 1\NONAME	3788288				no
System Volume Inform	Partition 1\NONAME	32768	2024-Feb-	2024-Feb-03 14:26:36		no
FTK Imager Lesson.mp4	Partition 1\NONAME	25082569	2024-Feb-	2017-Sep-21 17:59:24		no
Schedule (2).png	Partition 1\NONAME	73084	2024-Feb-	2023-Nov-08 02:24:10		no
poster-template.pptx	Partition 1\NONAME	1514882	2024-Feb-	2023-Dec-14 10:32:24		no
!0002_05.txt	Partition 1\NONAME	53	2024-Feb-	2015-Oct-06 07:30:36		yes
WPSettings.dat	Partition 1\NONAME	12	2024-Feb-	2024-Feb-03 14:26:36		no
IndexerVolumeGuid	Partition 1\NONAME	76	2024-Feb-	2024-Feb-03 14:26:38		no
822	Partition 1\NONAME	104857600				no
4022	Partition 1\NONAME	104857600				no

Part 2. Imaging with dd and netcat (nc)

A. SANS SIFT Virtual Workstation

The SANS Investigative Forensic Toolkit (SIFT) Workstation (virtual machine) is an Ubuntu system that has a group of free open-source incident response and forensic tools pre-installed to allow you to perform detailed digital forensic examinations in a variety of settings. Throughout this course, we will use SIFT Workstation for several Linux-based homework and labs.

Step 1. Download and install VMware Workstation Player (for Windows and Linux users) or Fusion (for Mac OSX) from RIT through “ITAcademy.” A free version of VMware Workstation Player can be downloaded at <https://www.vmware.com/products/player/playerpro-evaluation.html>.

Step 2. Create a SANS account online and download the **SANS SIFT Workstation** at <https://www.sans.org/tools/sift-workstation/> (Choose Option 1). **You have to create an account** to download the free SANS SIFT Workstation file *SIFT-Workstation.ova*.

Start the VMware Workstation Player, and select “Player>File>Open”. Navigate to the SIFT-Workstation.ova file and click “Open”. Import the SIFT Virtual machine to your desired location by clicking “Import”.

Step 3. Start the SIFT VM and install VMware Tools. After a successful import, you can start the SIFT Workstation. You will be prompted for a username and password for SIFT:

- Default username: *sansforensics*
- Default password: *forensics*

A note for M1 or M2 MAC users:

M1 or M2 Mac cannot run the SIFT VMware. Please contact Dr. Pan.

B. Using *dd* and *nc*

Sometimes, investigators will capture data from a suspect machine and send data to another networked computer (a forensic machine). *dd* and netcat (*nc*) can be used for imaging over the network. In this activity, you will **mimic** this process by sending a full image of your own USB **from one terminal to another terminal on the same machine** using *dd* and *nc*.

Instructions

1. Launch SIFT Workstation.
2. Unplug the USB drive from your Windows system, and insert the USB drive you used in Part 1 into your SIFT Workstation (in a real case, a write blocker should be used). The USB drive should auto-mount. (**Note:** check the VM USB settings for the “USB Controller” if you have any issues with mounting your USB drive. If the "USB Controller" is not listed in settings simply go down to the "Add..." and then add a "USB Controller" with the setting for USB compatibility set to USB 3.1 or whatever version of USB is the latest. If "USB Controller" is already listed, then make sure the latest USB version is being used. Restart the SWIFT VM and physically disconnect the USB.)
3. Open two terminals on SIFT Workstation. One terminal represents a forensic machine; the other represents the suspect machine.
4. On the forensic machine terminal, use *nc -l* to listen on port 8888 for the incoming data. Save the received data as *usb.dd*.
5. On the suspect machine terminal
 - First, generate both MD5 (*md5sum*) and SHA1 (*shasum*) hashes of your USB device
 - Second, use *dd* to make a full image of your USB and pipe (|) to netcat (*nc*), sending the USB image to the forensic machine terminal.

Hint: When generating hashes and/or using *dd* command to make a full image of your USB, you need to know the USB device file name as an input. You can run the command *mount* to learn the device file name assigned to your USB device.

For example, my USB's device file name is */dev/sdc1*

```
/dev/sdc1 on /media/sansforensics/B86D-C764 type vfat (rw,nosuid,nodev,uid=1000,gid=1000,shortname=mixed,dmask=0077,utf8=1,showexec,flush,uhelper=udisks2)
/dev/sdb1
```

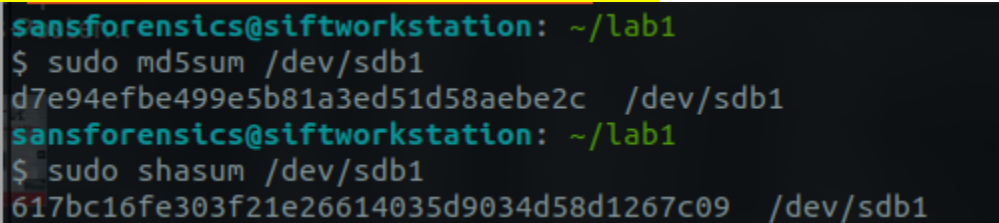
6. On the receiving terminal, once you have received the whole USB image, generate both MD5 (*md5sum*) and SHA1 (*shasum*) hashes of your USB image, *usb.dd*. (Note: if your USB size is large, it will take a while to compute the hash value.)

7. Make sure that:
 The md5 hash of the USB flash matches the md5 hash of the USB image
 The sha1 hash of the USB flash matches the sha1 hash of the USB image.

Task 4: Answer the following questions. Please enter your answers carefully, including all spaces and pipes as you would when entering commands into a system. (50 points – 10 points for each question below)

- What *nc* command did you use on the forensic machine to receive data on port 8888 and save the received data as *usb.dd*?
nc -l 8888 > usb.dd
- What command did you use on the suspect machine to use *dd* to make a full image of your USB and use netcat (*nc*) to send the USB image to the forensic machine terminal?
Sudo dd if=/dev/sdb1 | nc localhost 8888
- What command did you use to generate both MD5 (*md5sum*) and SHA1 (*shasum*) hashes of your USB device in step 5? Include a screenshot.

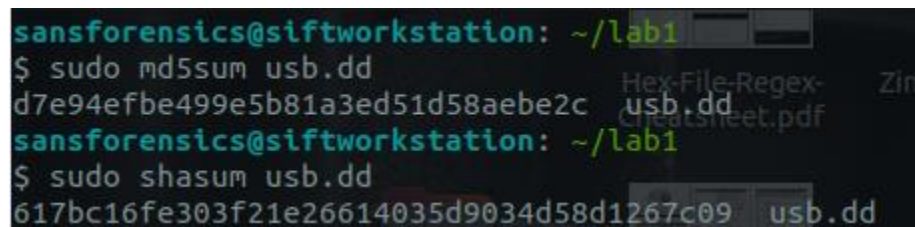
Sudo md5sum /dev/sdb1 and sudo shasum /dev/sdb1



```
sansforensics@siftworkstation: ~/lab1
$ sudo md5sum /dev/sdb1
d7e94efbe499e5b81a3ed51d58aeb2c /dev/sdb1
sansforensics@siftworkstation: ~/lab1
$ sudo shasum /dev/sdb1
617bc16fe303f21e26614035d9034d58d1267c09 /dev/sdb1
```

- What command did you use to generate both MD5 (*md5sum*) and SHA1 (*shasum*) hashes of your USB image in step 6? Include a screenshot.

Sudo md5sum usb.dd and sudo shasum usb.dd



```
sansforensics@siftworkstation: ~/lab1
$ sudo md5sum usb.dd
d7e94efbe499e5b81a3ed51d58aeb2c usb.dd
sansforensics@siftworkstation: ~/lab1
$ sudo shasum usb.dd
617bc16fe303f21e26614035d9034d58d1267c09 usb.dd
```

- Compare the hash values of *usb.dd* with the hash values of the raw image created by the FTK imager in Part 1, are the hash values the same or different? Provide a screenshot of the hash values from the FTK imager to support your answer. How do you explain your answer?

NO, the difference in hash values between images created by dd and FTK Imager can be attributed to two main factors: File System Interpretation and Different Imaging Processes. FTK Imager interprets and processes the file system, selectively including data based on its structure, while dd creates a bit-by-bit copy without such interpretation. Additionally, the imaging processes differ: FTK Imager might handle metadata, read errors, and data copying mechanisms in a more sophisticated manner than the straightforward approach of dd. These variations in approach lead to discrepancies in the final images, resulting in different hash values.

```
sansforensics@siftworkstation: ~/lab1
$ sudo md5sum usb.dd
d7e94efbe499e5b81a3ed51d58aeb2c  usb.dd
$ sudo shasum usb.dd
617bc16fe303f21e26614035d9034d58d1267c09  usb.dd
sansforensics@siftworkstation: ~/lab1
```

Drive Serial Number: 1107050340000259
Drive Interface Type: USB
Removable drive: True
Source data size: 29600 MB
Sector count: 60620800
[Computed Hashes]
MD5 checksum: f0fb0957259ad0d5a89418e124f535b9
SHA1 checksum: 772089d0233d1bf607234cbdf22beadc768a287e
Image Information:

Part 3. Linux memory acquisition using LiME

This part uses the SIFT Workstation.

Software and environment setting

On the SIFT Workstation, download LiME-master.zip from <https://github.com/504ensicslabs/lime>, and extract the zip file.

Goal

Using LiME to dump out the live memory, and try to extract information from the dump.

A. Using LiME to dump SIFT Workstation's memory

cd to the LiME *src* directory and compile LiME source code using *make*.

Now you should have the kernel module, *lime-VERSION-generic.ko*. Let's load the kernel module and dump out the memory called *yourusername_memory_dump.bin* to your desktop:

```
sudo insmod lime-VERSION-generic.ko  
"path=/home/sansforensics/Desktop/yourusername_memory_dump.bin format=lime"
```

After the command is executed, you should see the file, *yourusername_memory_dump.bin* is saved on the desktop.

To check the inserted lime kernel module: *lsmod | grep lime*. You should see "lime" has been inserted.

Task 5: Show me a screenshot of the *lsmod* result. (10 points)

```
sansforensics@siftworkstation: ~/lab1/lime/src
$ sudo insmod lime-5.15.0-70-generic.ko "path=/home/sansforensics/Desktop/nma4209_memory_dump.bin format=lime"
sansforensics@siftworkstation: ~/lab1/lime/src
$ lsmod | grep lime
lime                16384  0
sansforensics@siftworkstation: ~/lab1/lime/src
$
```

Clean-up: To remove the lime kernel module: *sudo rmmod lime*.

B. Extracting information from your memory dump

Examine the *yourusername_memory_dump.bin* file to find some printable information, such as your login password. For example, you may use *strings* to display printable information (*man strings* to learn how to use this command) and pipe the strings result to *grep* to filter your desired result. In addition, you use data carving tools such as *foremost* (installed in SIFT) to extract files based on file headers.

Task 6: Show me at least two command(s) and options you used to extract useful information using *strings*. What interesting data did you find in your memory? Provide screenshots of the commands and results. (10 points)

strings nma4209_memory_dump.bin | grep -Eo '([0-9]{1,3}[\.]){3}[0-9]{1,3}'. Using this command I could find some Ips which will sure help in the DFIR investigation.

```
sansforensics@siftworkstation: ~/Desktop
$ strings nma4209_memory_dump.bin | grep -Eo '([0-9]{1,3}[\.]){3}[0-9]{1,3}'
192.168.12.1
192.168.12.1
1.3.5.7
2.4.6.8
1.3.5.7
2.4.6.8
1.3.5.7
1.3.5.7
2.4.6.8
1.3.5.7
2.4.6.8
1.3.5.7
0.8.18.1
127.0.0.0
0.8.18.1
47.36.36.36
```

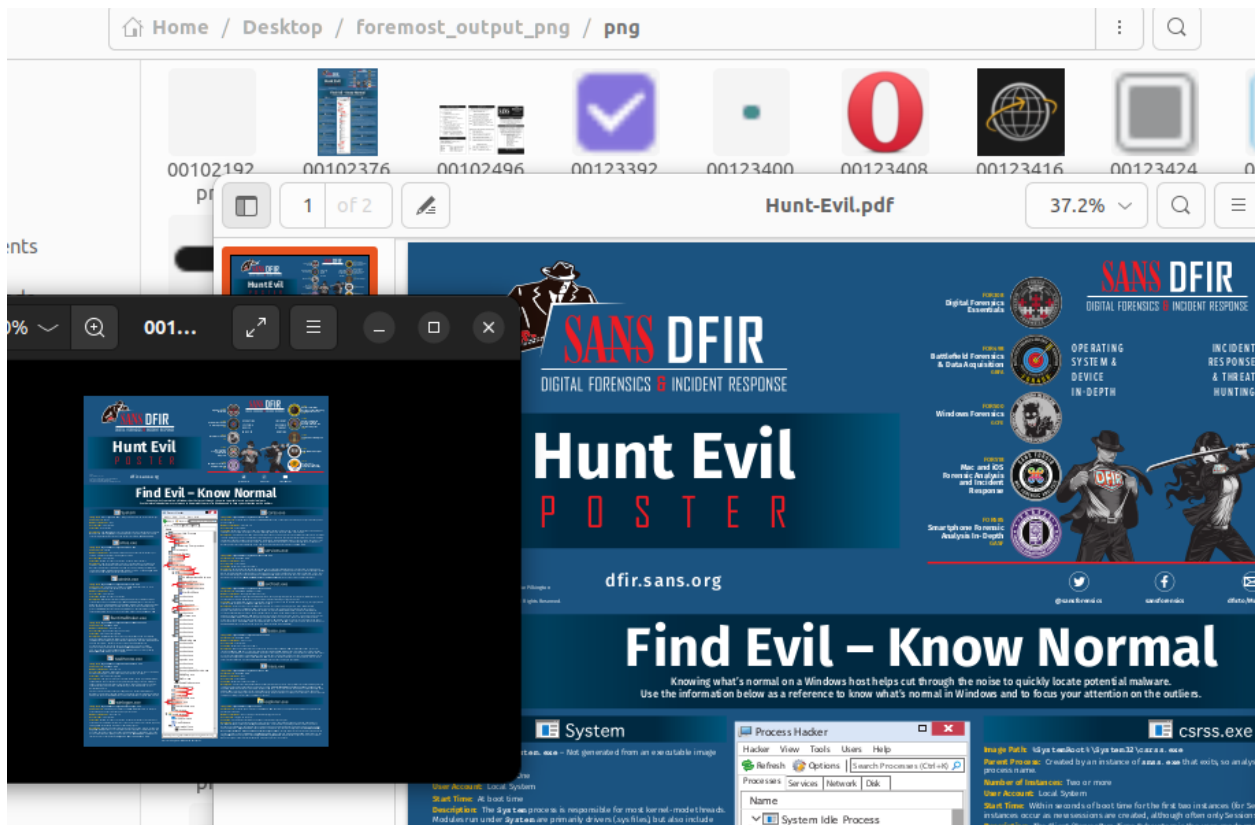
strings nma4209_memory_dump.bin | grep -i 'sudo'. Using this command I could find some Sudo commands which will for sure be helpful in the DFIR investigation.

```
sansforensics@siftworkstation: ~/Desktop
$ strings nma4209_memory_dump.bin | grep -i 'sudo'
$ sudo md5sum /dev/sdb1
$ sudo shasum /dev/sdb1
$ sudo dd if=/dev/sdb1 | nc localhost 8888
$ sudo md5sum usb.dd
$ sudo shasum usb.dd
/usr/share//bash-completion/completions/sudo
sudo
sudo
re//bash-completion/completions/sudo
sudo md5sum usb.dd
visudo
```

Task 7: Show me the command(s) with options you used to extract useful information using *foremost*. What interesting data did you find in your memory? Provide screenshots of the *foremost* command and results showing at least two types of interesting data. (10 points)

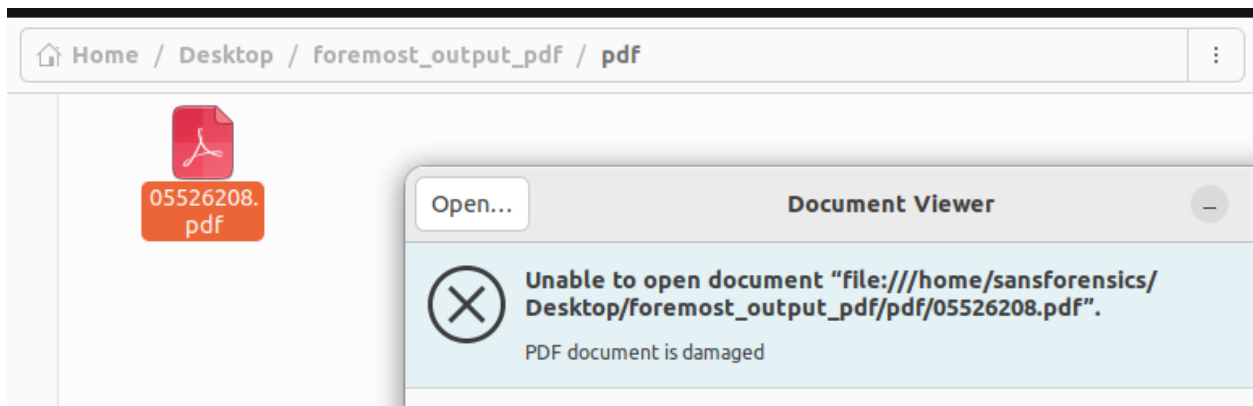
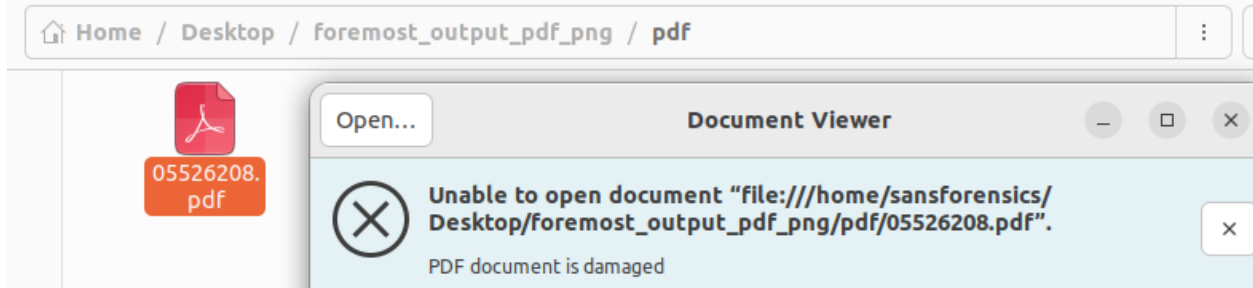
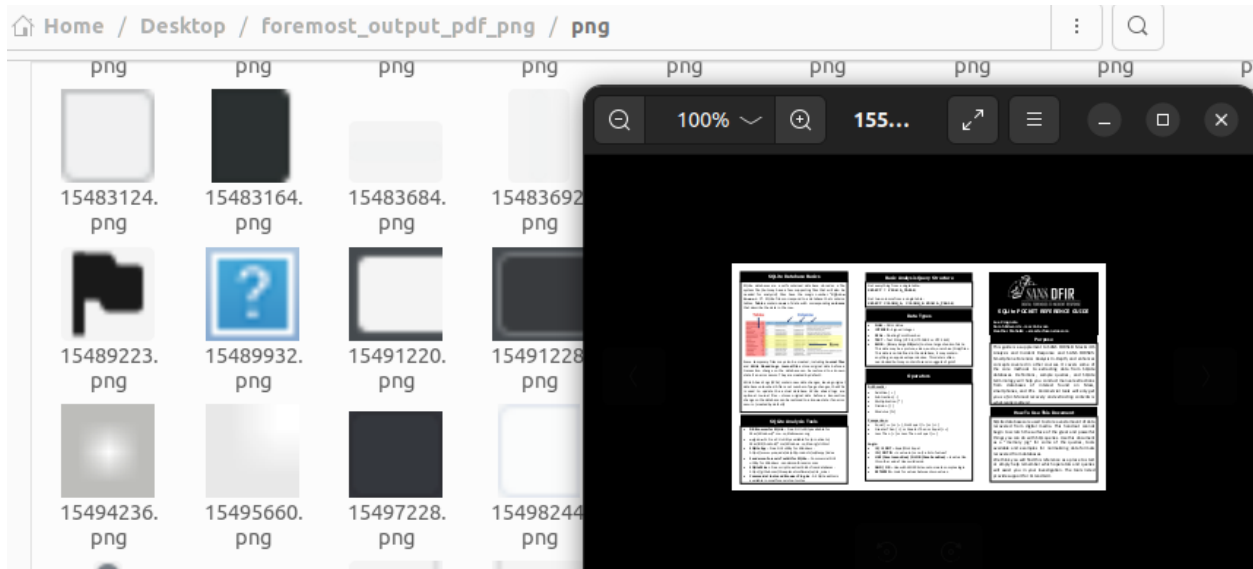
- 1- Using this command (`foremost -v -t png -i /home/sansforensics/Desktop/nma4209_memory_dump.bin -o /home/sansforensics/Desktop/foremost_output_png`), I found a png picture that actual is a pdf file which is very unusual for me to find data that has been recovered with different format.

```
sansforensics@siftworkstation: ~/lab1/time/src
$ foremost -v -t png -i /home/sansforensics/Desktop/nma4209_memory_dump.bin -o /home/sansforensics/Desktop/foremost_output_png
```



- 2- So in this command I tried to delete a pdf file and then re-do the lime part to dump the memory again to check if I can recover the pdf file as pdf and the same time as png image. However, the result was surprising for me I did recover a pdf file but it was damaged and I couldn't open it, on the other hand I did see it as png picture. I don't know why but it was really surprising and I tried to do it separately but in both cases I couldn't open the pdf file.

```
sansforensics@siftworkstation: ~/lab1/time/src
$ foremost -v -t pdf,png -i /home/sansforensics/Desktop/nma4209_memory_dump.bin -o /home/sansforensics/Desktop/foremost_output_pdf_png
```

```
sansforensics@siftworkstation: ~/lab1/lime/src
$ foremost -v -T pdf -i /home/sansforensics/Desktop/nma4209_memory_dump.bin -o /home/sansforensics/Desktop/foremost_output_pdf
Foremost version 1.5.7 by Jesse Kornblum, Kris Kendall, and Nick Mikus
Audit File
  Other Locations
Foremost started at Sun Feb  4 05:22:35 2024
Invocation: foremost -v -T pdf -i /home/sansforensics/Desktop/nma4209_memory_dump.bin -o /home/sansforensics/Desktop/foremost_output_pdf
Output directory: /home/sansforensics/Desktop/foremost_output_pdf_Sun_Feb__4_05_22_35_2024
Configuration file: /etc/foremost.conf
Processing: /home/sansforensics/Desktop/nma4209_memory_dump.bin
```