Case Name: Introduction to Single Purpose Forensic Tools

Couse Name: IST402

Instructor: Robert Price

Date: 03/17/2025

Examiner Name: Jaspreet Singh

Table of Contents

List of Illustrative Materials

EXECUTIVE SUMMARY

Background

The examiner will demonstrate an understanding of forensic methodology, key forensic concepts, identifying types of evidence on current Windows operating systems and be familiar with the structure and composition of modern Windows file systems. The examiner will demonstrate an understanding of the methodologies and tools used to collect and process digital forensic evidence.

Hashing is the process of taking in a stream of plain text and transforming the data into a hashed text using a hashing algorithm. You can use the hash to make sure that a message was not modified during transmission. That hash can make sure that the disk image was not tampered with. Hashed images are used in forensics investigations. Hashing is also used on files, passwords, and other pieces of data. In this lab, the examiner is going to image a disk and create a hash of that disk, verify integrity using file hashing tools, use Foremost to carve and recover deleted files from a disk, and use a hex editor to review files. Foremost is a file carving tool used to recover files based on known headers and file signatures, even after they have been deleted from the file system. A hex editor is a tool that allows the examiner to inspect the raw hexadecimal content of files, which is helpful in verifying file types and identifying data fragments.

Evidence

Description	Hash Algorithm	Hash Value	Examiner
Evidence	MD5	6958437CFB625D29A17121893E07402C	Jaspreet Singh
Evidence	SHA1	FEE3A78ADF5DD06D048BC90345CA7C546CF38D09	Jaspreet Singh

COLLECTION AND ANALYSIS

Collection

The examiner began by verifying the integrity of the disk image called image.dd by using and utilizing the md5 and sha1 hash values to configure that they match the original FTK generated hash values. Which were verified though, .txt files which contained the original hash values hence, once checked to ensure an exact match for the hash values the examiner would know that the image's integrity is maintained, and the image has not been altered.

The examiner then mounted the disk image using the mount command in the command prompt hence allowing the examiner to see once mounted that there are no visible files present. Therefore, suggesting that there has been deleted data, and it was not overwritten. The examiner utilized the following command "ls -la" to check for hidden files and verify that the disk image was mounted in read-only mode so that the integrity of the image is preserved through this process.

To then recover this deleted data the examiner uses a forensic tool called Foremost which is a file carving tool to extract specific types of files from the disk image. In this case the examiner did separate command prompt output directories using Foremost for the following file types: JPG, GIF, AVI, PNG, EXE, DOC, and PDF. After each file type was carved, the examiner was able to review each directory to verify that the forensic tool was able to successfully recover files based on the file headers and signatures. Then the examiner was able to see audit.txt in the command prompt which showed the examiner a log for each file carving and showed the number and types of files recovered from each type.

Then the examiner went into the directory and containing the carved JPG files and opened the files using a hex editor. By examining the raw hexadecimal content, the examiner was able to confirm the file signature of "JFIF" which indicates a JPEG image. From that the examiner was able to see that raw hex analysis can also be used to verify the integrity of the recovered files from the system.

Analysis

Forensic File Viewing

The examiner was able to use the command prompt to navigate to the forensics directory and viewed its contents using the ls command. Then the examiner uses an application called leafpad to open the sampleflag.txt file hence showing that the flag value is 999818.

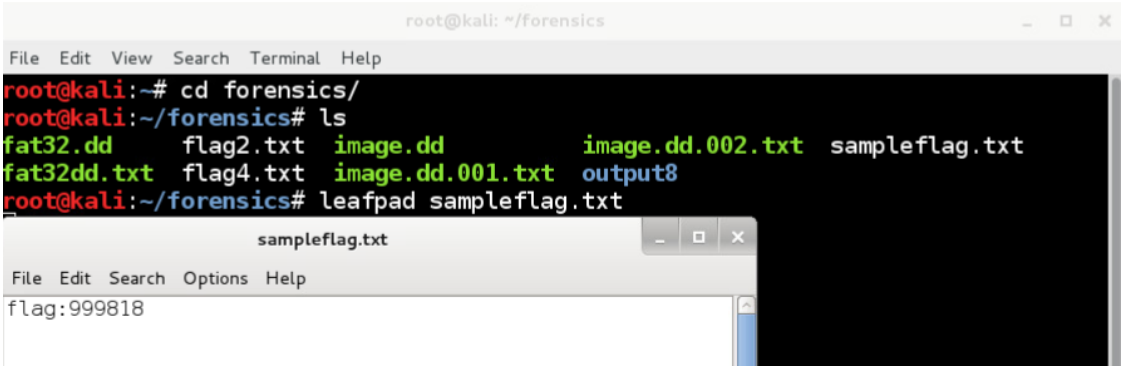

Figure 1: Access Forensic Files & Viewing Flag

FTK Imager Hash Verification

The examiner was able to open a .txt file by using the ls command "ls image.dd.001.txt" in the prompt hence allowing this file to open and show the examiner data about the disk image as well as the hash values. The examiner can see that both the MD5 and SHA1 checksums are both marked as verified, confirming that the image's integrity is maintained.

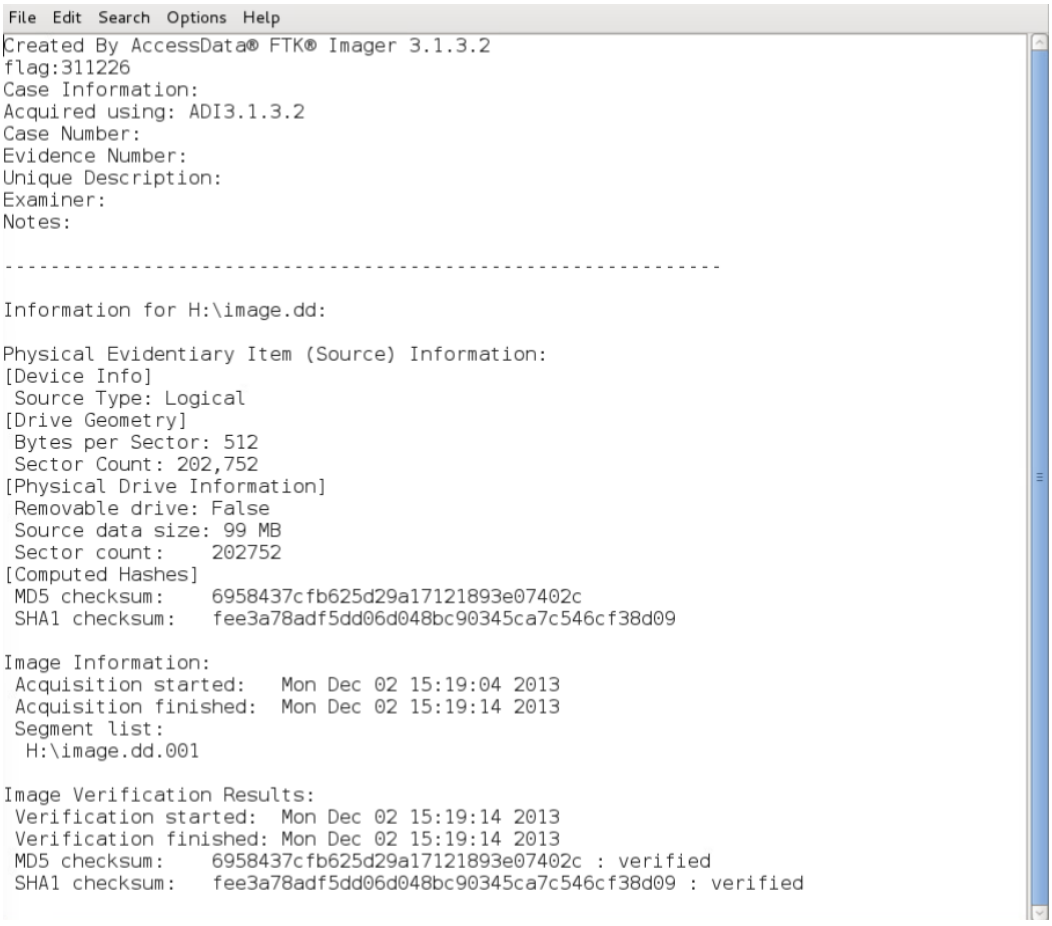

Figure 2: Viewing FTK Imager Hash File

Cat Command Hash Verification

The examiner can use the "cat" command which is used to display the contents of a file directly in the terminal to verify that both the MD5 and SHA1 hash values match and are verified.

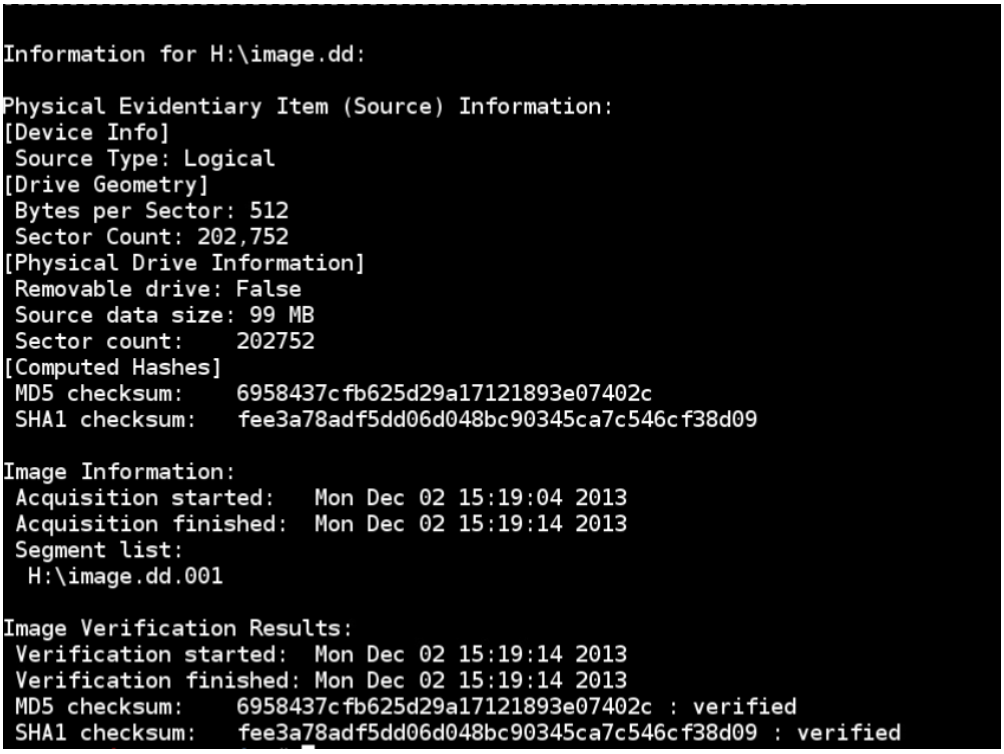

Figure 3: Cat Command Hash Verification

Cat Command MD5

The examiner uses the cat command in the prompt to extract the MD5 has value from the image.dd.001.txt file and the checksum value to verify integrity. The examiner was able to use "grep MD5" in the prompt to extract only information on the MD5 hash values.

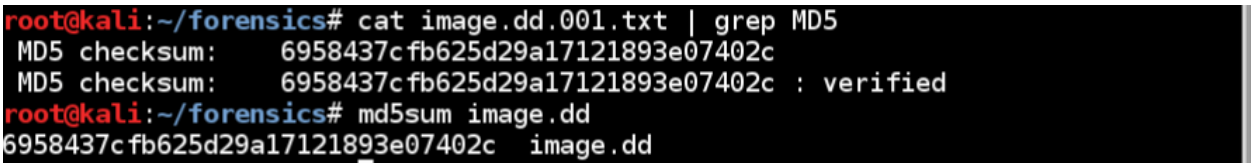

Figure 4: Cat MD5 Checksum

Cat Command SHA1

The examiner uses the cat command in the prompt to extract the SHA1 has value from the image.dd.001.txt file and the checksum value to verify integrity. The

examiner was able to use “grep SHA1” in the prompt to extract only information on the SHA1 hash values.

Figure 5: Cat SHA1 Checksum

Files and Directory before Mounting

The examiner can use the following command “ls -l” to view information about the forensic folder and the details of the files within the folder.

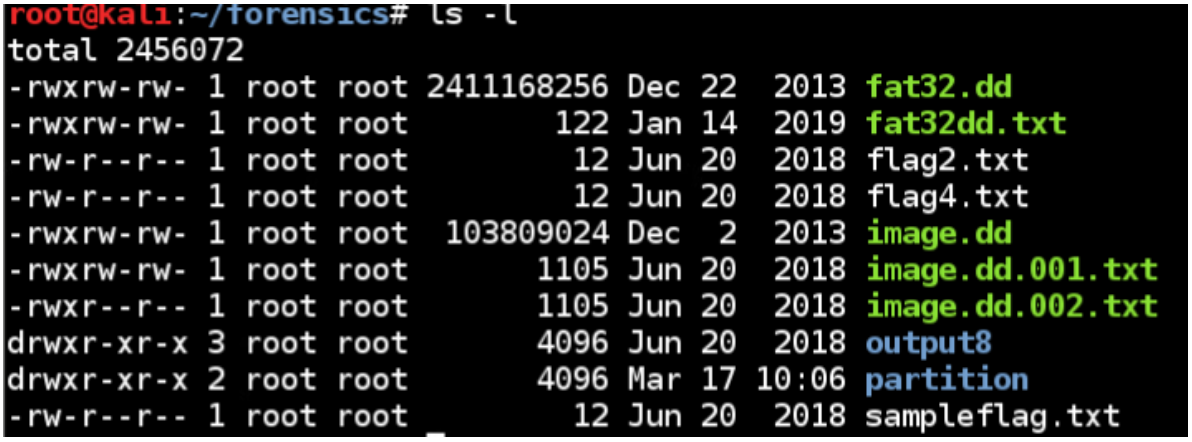

Figure 6: Viewing Forensics Folder and Files

Checking Contents of Mount

The examiner runs the following command “ls -la partition” to check for hidden files and confirm the partition was empty prior to mounting.

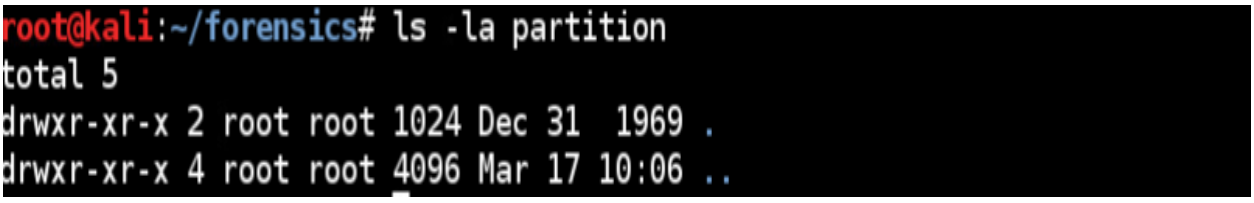

Figure 7: Mount Contents

Mount the Disk Image

The examiner uses the following command to mount the disk image “mount | grep vfat” this then confirms that image.dd was mounted to the directory using the VFAT file system. Then after the examiner made a mistake by misspelling the umount

partition command by saying “unmount” but after executing the command correctly with umount the examiner safely unmounts the image.

```
root@kali:~/forensics# mount | grep vfat
/root/forensics/image.dd on /root/forensics/partition type vfat (rw,relatime,fma
sk=0022,dmask=0022,codepage=cp437,iocharset=utf8,shortname=mixed,errors=remount-
ro)
root@kali:~/forensics# unmount partition
bash: unmount: command not found
root@kali:~/forensics# umount partition
```

Figure 8: Mounting

Viewing Foremost Help Menu/Options

The examiner uses the “foremost -h” command to display available options and instructions for the Foremost forensic file carving tool. This menu provides a reference for the examiner for specifying file types, input files, and output directories during the file recovery process.

```
root@kali:~/forensics# foremost -h
foremost version 1.5.7 by Jesse Kornblum, Kris Kendall, and Nick Mikus.
$ foremost [-v|-V|-h|-T|-Q|-q|-a|-w-d] [-t <type>] [-s <blocks>] [-k <size>]
        [-b <size>] [-c <file>] [-o <dir>] [-i <file]

-V  - display copyright information and exit
-t  - specify file type.  (-t jpeg,pdf ...)
-d  - turn on indirect block detection (for UNIX file-systems)
-i  - specify input file (default is stdin)
-a  - Write all headers, perform no error detection (corrupted files)
-w  - Only write the audit file, do not write any detected files to the disk
-o  - set output directory (defaults to output)
-c  - set configuration file to use (defaults to foremost.conf)
-q  - enables quick mode. Search are performed on 512 byte boundaries.
-Q  - enables quiet mode. Suppress output messages.
-v  - verbose mode. Logs all messages to screen
```

Figure 9: Foremost Menu

Viewing Foremost Configuration File

The examiner used the following command “head -n 20 /etc/foremost.conf” to preview the Foremost configuration file. Within the file defines how Foremost identifies file types based on headers and footers and allows customization for supported file formats.

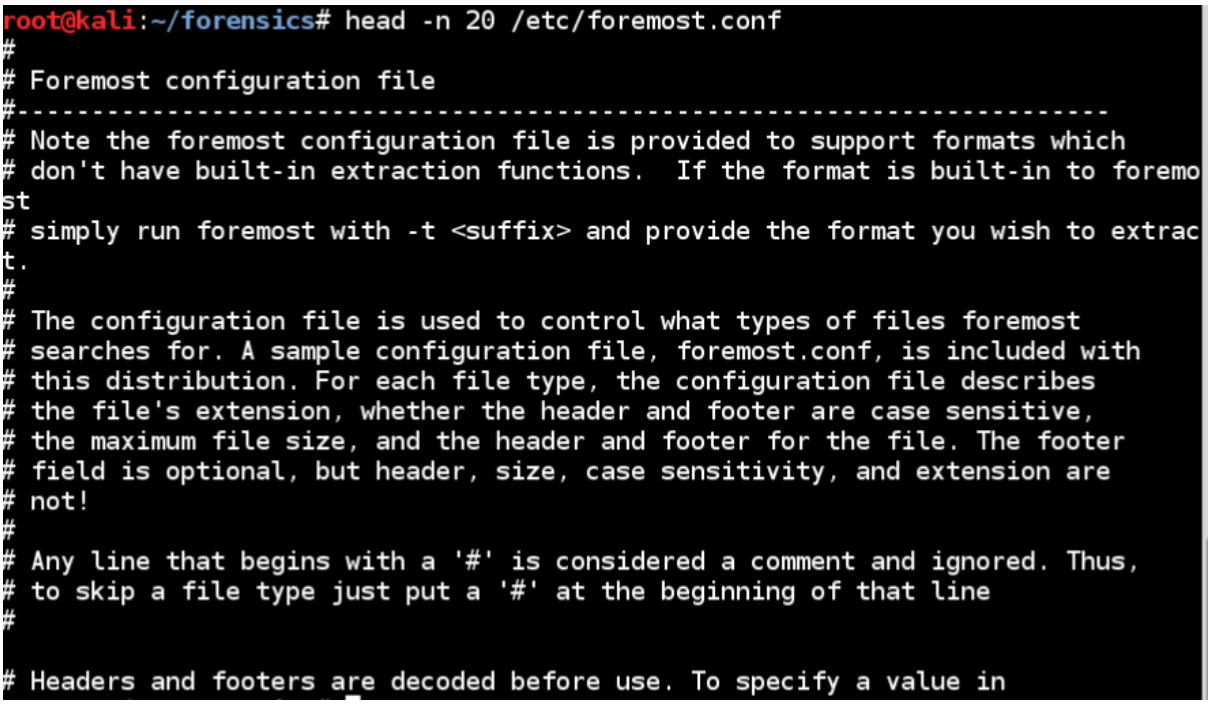

Figure 10: Foremost Configuration File

JPG Recovery

The examiner was able to execute the following command “foremost -i image.dd -t jpg -o output1” to recover the deleted JPG files from the disk image. In this the command the “-i” is specifying the input file which is image.dd. Then the “-t” tells the type of file to search and look for which is JPG. Finally, the “-o” tells the output directory will be stored in output 1. The examiner then executes the command “cat output1/audit.txt” to then find that 83 JPG images were recovered and the details for the JPG images as well.

Figure 11: Foremost JPG Recovery

GIF Recovery

The examiner executes the same foremost command as done before for JPG, “foremost -i image.dd -t gif -o output2” but this time for GIF’s then saving it as output2. The examiner then executes the command “cat output2/audit.txt” to then find that 10 GIFs were recovered and the details for the GIFS as well.

```
root@kali:~/forensics# foremost -i image.dd -t gif -o output2
Processing: image.dd
|*|
root@kali:~/forensics# cat output2/audit.txt
```

Figure 12: Foremost GIF Recovery

AVI Recovery

The examiner executes the same foremost command but for avi as shown "foremost -i image.dd -t avi -o output3". The examiner then executes the command "cat output3/audit.txt" to then find that 5 AVIs were recovered and the details for the AVI's as well.

```
root@kali:~/forensics# foremost -i image.dd -t avi -o output3
Processing: image.dd
|*|
root@kali:~/forensics# cat output3/audit.txt
```

Figure 13: Foremost AVI Recovery

EXE Recovery

The examiner executes the same foremost command but for exe as shown "foremost -i image.dd -t exe -o output4". The examiner then executes the command "cat output4/audit.txt" to then find that 3 EXEs were recovered and the details for the EXE's as well.

```
root@kali:~/forensics# foremost -i image.dd -t exe -o output4
Processing: image.dd
|*|
root@kali:~/forensics# cat output4/audit.txt
```

Figure 14: Foremost EXE Recovery

PNG Recovery

The examiner executes the same foremost command but for png as shown "foremost -i image.dd -t png -o output5". The examiner then executes the command "cat output5/audit.txt" to then find that 230 PNGs were recovered and the details for the PNG's as well.

```
root@kali:~/forensics# foremost -i image.dd -t png -o output5
Processing: image.dd
|*|
root@kali:~/forensics# cat output5/audit.txt
```

Figure 15: Foremost PNG Recovery

DOC Recovery

The examiner executes the same foremost command but for png as shown "foremost -i image.dd -t doc -o output6". The examiner then executes the command "cat output6/audit.txt" to then find that 3 doc files were recovered and the details for the doc files as well.

```
root@kali:~/forensics# foremost -i image.dd -t doc -o output6
Processing: image.dd
|*|
root@kali:~/forensics# cat output6/audit.txt
```

Figure 16: Foremost DOC Recovery

PDF Recovery

The examiner executes the same foremost command but for png as shown "foremost -i image.dd -t pdf -o output7". The examiner then executes the command "cat output7/audit.txt" to then find that 3 pdf files were recovered and the details for the pdf files as well.

```
root@kali:~/forensics# foremost -i image.dd -t pdf -o output7
Processing: image.dd
|*|
root@kali:~/forensics# cat output7/audit.txt
```

Figure 17: Foremost PDF Recovery

Command Prompt Opening HEX Editor

The examiner executes the following cod below to navigate to the jpg folder within the output1 directory, which contains the recovered JPG files from the disk image. Then the examiner uses the hex editor command to then begin a hexadecimal analysis on the carved images files from Foremost.

```
root@kali:~/forensics# pwd
/root/forensics
root@kali:~/forensics# cd output1
root@kali:~/forensics/output1# cd jpg
root@kali:~/forensics/output1/jpg# hexeditor
```

Figure 18: Opening HEX Editor in Command Prompt

JPG HEX Editor

The examiner opens the recovered JPG file called 00043125.jpg which is a unique identifier assigned by Foremost in the hex editor application and confirmed the presence of the JFIF file signature which helps verify that the file is a valid JPEG file.

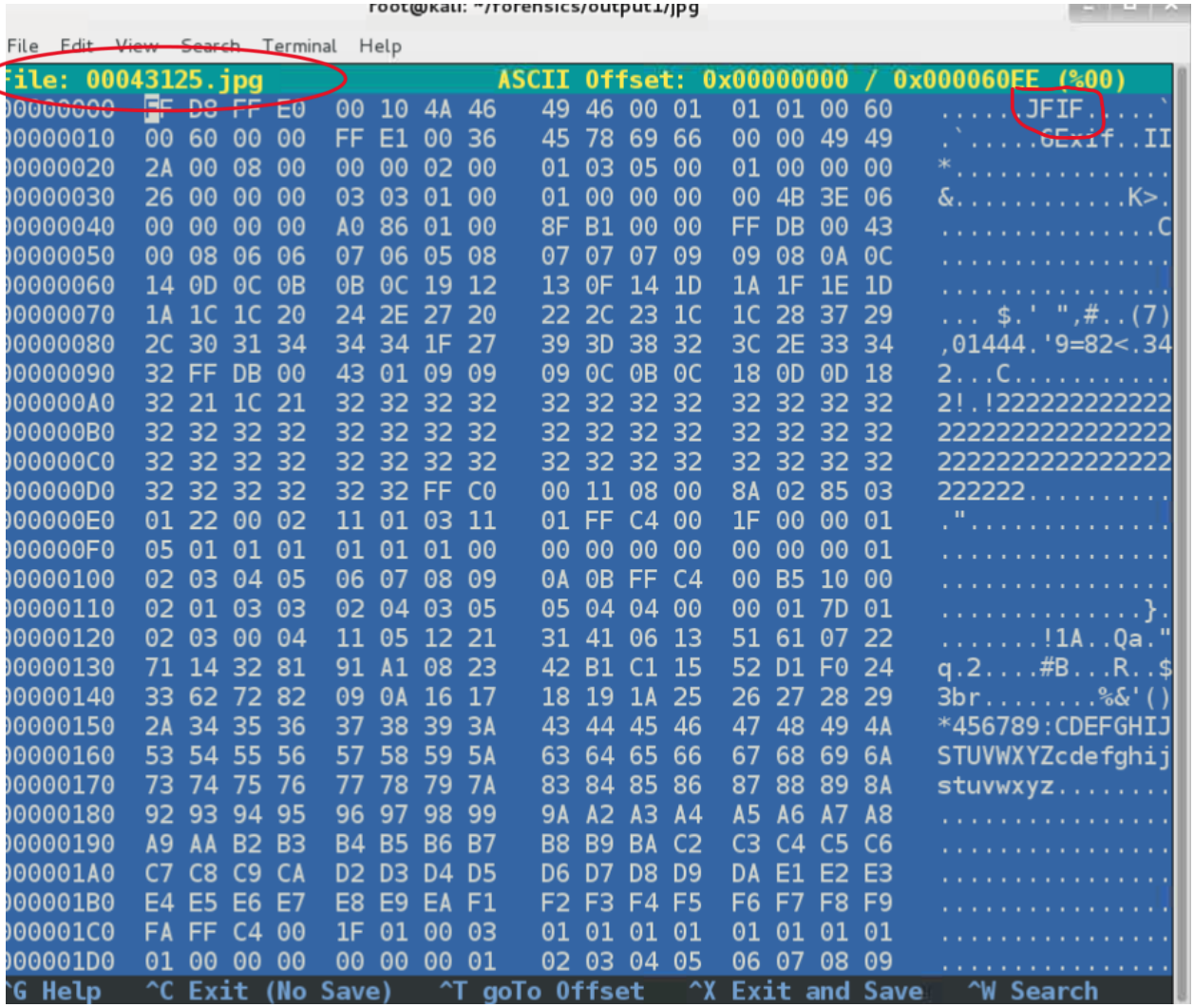

Figure 19: JPG HEX Editor

Conclusion

Overall, through this lab the examiner was able to gain hands-on experience using single-purpose forensic tools to verify data integrity, recover deleted files and analyze file content at a byte level. The examiner used the following forensic tools such as sha1sum, md5sum, mount, Foremost, and HEX editor to simulate and learn real-world forensic techniques and maintain evidence integrity throughout the entire investigation process.

Hashing

The examiner utilized the md5sum and sha1sum in to generate and compare hash values for the disk image image.dd file. Therefore, through these checksums of md5 and sha1 the examiner was able to confirm the disk images integrity against the

original FTK hash values. Hence ensuring that the disk image was reliable to use for forensic analysis and the examiner can continue with the forensic investigation.

Mounting

The examiner was able to create a mount in the command prompt then mounted the disk image in read-only mode using the mount command. From there the examiner was able to see that no visible or hidden files were present. Hence indicating to the examiner that there is deleted data that needs to be recovered using additional forensic tools.

Foremost

Through this lab the examiner utilized a forensic tool called Foremost which is used to help carve out deleted file types such as JPG, GIF, AVI, PNG, EXE, DOC, and PDF from the disk image. Then the examiner was able to output each file type into their own output directories and audit log each file so that the examiner can confirm the successful completion and recovery of files.

HEX Editor

Finally, the examiner uses HEX editor to examine and recovered JPF files from output 1 and confirmed that the JPEG pictures are verified due to the JFIF signature. This usage of the HEX editor application highlights the importance of byte level analysis when detecting a files authenticity and trying to find hidden data.