Digital Forensics T1 (2023) 1–10

Digital Forensic Analysis - Assignment 2

Olivia Fletcher
ID: 300534281, Email: fletcholiv@myvuw.ac.nz

School of Engineering and Computer Science, Victoria University of Wellington, P.O. Box 600, Wellington 6140, New Zealand.

Abstract

With the continuous advancement of technology, new IoT devices arise that introduce new communication methods making the Digital Forensics world increasingly more complicated. In this report, we will explore the significance of ten distinct subcategories within Digital Forensics that are still utilized to this day. The categories will be supplied by NDG's Netlab+[3]. We will be referring to the knowledge from the literature "Digital Forensics"[1] to assist us in this assignment. The following categories are as follows:

1. Lab Work

We will be utilizing Netlabs NDG to complete the above labs.

1.1. Live Forensics

Live Forensics refers to the procedures used when collecting and analyzing data from a live system while also ensuring the preservation of evidence and data integrity. In this lab, we utilized various tools for analyzing and gathering important system information while also ensuring full documentation, as per the Chain of Custody principle.

The first step of this lab involves utilizing Forensic Toolkits such as 'Digital Evidence and Forensic Toolkit (DEFT 8)' and 'Digital Advance Recovery Toolkit (DART)'. Implementing these tools will allow us to review live data while also ensuring no data alteration can take place. Within the DART software we are able to access a specialised tool known as 'DriveMan'. DriveMan provides detailed driver information such as showcasing disk info, serial numbers, volume of serials, size of drive, allocated/unallocated clusters, name details and whether or not it is removeable or fixed to the system.

Next step is to implement FTK Imager and TreeSizeFree tools through DART to further analyze the data contained within the drive. By default sensitive system files/folders are hidden from the user; however, FTK Imager and TreeSizeFree can help us identify and explore these hidden files/directories while also providing a GUI screen where we can review their contents.

This section of the lab taught us how to navigate through system directories by examining the sizes of files and folders, allowing us to narrow down potential locations of interest. This method proved useful in identifying downloaded files and caches of (potentially) illicit data.

The last stage of the lab involves implementing WinAudit and BrowsingHistoryView, where we gathered vasts amount of user data for analysis such as reviewing user downloaded software and user browsing history.

In conclusion, this lab taught us how sufficiently navigate through live data while also maintaining data integrity by utilizing specialised tools.

1.2. Steganography and Alternative Data Streams

Steganography and Alternative Data Streams are forms of a data obfuscation technique where files are hidden inside seemingly innocent/innocuous data. In this lab, we will utilize two forms of data obfuscation methods to understand the process involved and be able to recognise if there may potentially be hidden files present within data.

The first step of this lab is to utilize HxD, a Hex Editor, to review the hex values of a JPG image in search of potentially hidden files within. Since the base file is a JPG image, there should only be one relevant hex header value present - "89-50-4E-47- 0D-0A-1A-0A". However, by searching for the hexadecimal representation of other file header values, we have uncovered an additional file within.

By implementing the search function we uncovered a "PNG" (89-50-4E-47-0D-0A-1A-0A), indicating the presence of an unrelated file. After noting the offset location of this header, we searched for the corresponding PNG footer value "49 45 4E 44 AE 42 60 82" and extracted this block. Opening the extracted file, we revealed the following secret message:

- "THIS IS MY SECRET MESSAGE. THEY WON'T EVER FIND THIS."

Another method of data obfuscation involves Alternate Data Streams. In section, we utilized an ADS technique to hide a text file within a seemingly legitimate executable. By using Command Prompt (CMD) with administrative privileges, we added added a text file within an executable via the command:

- type Legitimate_Program.exe > Legitimate_Program.exe:secret.txt

To then access and edit the file, we used the following:

- notepad Legitimate_Program.exe:secret.txt

This technique prevents viewing through regular file browsing. Review requires specialised tools such as FTK Image Viewer or by the "dir /r" command, which recursively checks files.

In conclusion, this lab taught us how to uncover hidden files through Steganography means but to also hide secret messages within legitimate executables (ADS).

1.3. Picture File Analysis

Picture File Analysis involves the process of extracting additional, potentially valuable information from smartphone images. By analyzing embedded image data, such as EXIF data, we will be able to uncover details such as creation time, file modification history, file permissions and even comprehensive GPS information.

The first step of this lab is to review and compare image creation dates to their corresponding GPS data to see if there are any discrepancies contained, which could indicate file manipulation has taken place at some point. For this, we will utilize Exiftool which will provide this detailed output. We can extract and review an images EXIF information by the following commands:

- exiftool IMG_001.jpg -w txt
- gedit IMG_001.txt

Taking note of each image files entries, we are comparing the "Date/Time Original" against the "GPS Date/Time". We observed the following from the provided four images:

- File IMG_001: Creation data - 2020:08:01 14:21, GPS - 2020:08:01 21:21
- File IMG_002: Creation data - 2018:10:06 13:40, GPS - 2018:10:06 20:40
- File IMG_003: Creation data - 2020:08:01 14:32, GPS - 2020:08:01 21:28
- File IMG_004: Creation data - 2020:08:01 17:43, GPS - 2020:08:02 00:43

Comparing file-system information with EXIF metadata helps verify the correct date/time of an image. The only discrepancy we found was with File IMG_004, where the GPS date was a day earlier than the set creation time.

The next stage of this lab involves using HxD to carve image files from a thumbnail database file. By searching for PDF header values "ff-d8-ff-e0", we have successfully located all other files present within. By saving each block with their corresponding footer value "ff-d9" we were able to extract the thumbnail images and save them with a JPG extension for viewing. Through this process, we discovered the following information for each image file:

- IMG_001.jpg: Start offset - A18, End offset - 6E38, Scenic image of a river
- IMG_002.jpg: Start offset - 7018, End offset - D7BF, Scenic image of a natural bridge
- IMG_003.jpg: Start offset - DA18, End offset - 11892, Scenic image of a lake, forest, and boats/kayaks
- IMG_004.jpg: Start offset - 11C18, End offset - 162B3, Scenic image of a lake behind a large rock formation

In conclusion, this lab taught us the technique of reviewing an images EXIF data and extract multiple thumbnail images from database file by it's hex data alone.

1.4. Email Analysis

Email Analysis involves the forensic procedures of reviewing and analyzing email header metadata to potentially uncover additional meaningful information. In this lab, we will learn what an email header is, what type of data is contained within and how to utilize an 'Email Header Analyzer' to potentially decipher any meaningful information.

The first step of this lab is employing a tool known as 'Thunderbird Mail', which offers email header analyzer abilities by allowing us to access an emails metadata directly, we will start by creating a forensic copy and exploring the metadata of a specified email. After extracting the email 'The Deadline' from Thunderbird Mail we then review its contents from the bottom-up. We start from the bottom when reviewing email header data because the entries are stacked on top of each other, with the most recent entries being at the top.

At the bottom, we start with the base email data such as, sender, receiver, and timestamp information. Followed after is the Message-ID which is a unique code assigned to this email which is used as a form of identification. The next section contains useful data such as the sender's IP address, mail server used, and SPF security check results. The SPF, Sender Policy Framework, is a security feature used by mail servers to prevent spoofing and other malicious activity. Overall, reviewing email header metadata can be helpful due to showcasing meaningful data such as, authentication measures, sender/receiver details, security checks, and aid in tracing the email's source.

The last step of this lab is to utilize an browser-enabled 'Email Header Analyzer' where we can paste the email metadata to. After compiling, the output data is then organised on the GUI screen in a neat and readable format.

In conclusion, we learnt the importance of email header metadata and the types of data we can extract.

1.5. IoT Forensics

IoT Forensics refer to the procedures used when reviewing data from different type of (Internet of Things) devices. This can range from smartphones, to computer drives, to smartwatches and various other devices that have the ability to connect to the internet. In this lab we will learn how to interpret data from different devices in order to extract potentially meaningful data.

The first step of this lab is to review an android phones contents through FTK Image Viewer. Navigating through the directories by expanding the tree view we can see some familiar folders such as "media" which is commonly found in Android Devices and contains subfolders containing different types of media files accessed by the device such as, images/videos to music and downloads.

Moving our attention to the Images subfolder we can see a list of 4 images which is viewable within the View Pane when selected. Reviewing files through FTK Imager can provide valuable insights as it also showcases hidden and in some-cases, deleted data.

Moving onto the '.thumb' directory we can see a list of thumbnails derived from synchronized pictures and videos. By navigating through the '.thumb' directory and its subfolders, we discover thumbnails corresponding to various file types such as images (.jpg), videos (.mp4), and even embedded album cover images for MP3 files.

In conclusion, this lab introduced us to the world of IoT devices and the significance of interpreting the different data types contained within. Through the use of tools like FTK Image Viewer and DB Browser (SQLite), we gained insights into the captured information from these devices. By loading the 'Forensic Evidence File' (FEF) we were able to explore the file system, and review some familiar folders and access different types of media files associated with the device.

1.6. Timeline Analysis

Timeline Analysis refers to the process involved in analyzing sets of data by their corresponding dates and times as a means of gathering potentially insightful user data. In this lab, we will utilise Autopsy to establish a timeline of events that occurred on a suspects machine by reviewing the systems drive contents.

The first step of this lab is to examine a drives contents in search for anything noteworthy through the Autopsy Timeline function provided on the top menu bar. By navigating to the My Pictures directory location under /vol2/Documents and Settings/Fred/My Documents we can see a list of 5 GIF files. Adding a bookmark and taking note of the timestamps of these files we will then analyze how they got there in the first place by reviewing other processes that took place (on the timeline feature) during their timestamp.

Reviewing the other processes showed that a folder called 'Temporary Internet Files' that stores cached browser usage was used, indicating that an internet connection was active during this specified timeline. Reviewing the browsing history for this day shows that the first gif file, m001.gif was saved to the system 11 seconds after it was accessed through Internet Explorer. Reviewing these entries show that the user accessed these gif images from www.mickey-mouse.com before downloading them to the system.

In conclusion, we learnt how to utilize Autopsy's Timeline Analysis tool as a means of reconstructing a sequence of events behind a files creation (save) date. This helps us gain a deeper understanding into a users activity/actions during a timeline of interest in regards to evidence collection.

1.7. Mobile Forensic Analysis

Mobile Forensic Analysis refers to procedures taken when reviewing smartphone data. Smartphones contain vast amounts of potentially useful information that can assist in digital forensics investigations. In this lab, we will be reviewing phone config logs, such as internet connectivity, connection history, Bluetooth information, and phone hardware details (IMEI) and phone type/version to potentially uncover any useful information.

To start the lab, we will utilize Autopsy to analyze data extracted from a mobile phone. Navigating through the system logs in the tree-pane we can gather valuable insights into user activities, including location history, app usage, and communication history through calls, texts, and emails. The following system files contain some key information, as seen below:

- android.providers.settings/databases/settings.db -
  Showcases system settings such as bluetooth, data roaming, airplane mode and GPS preferences.
- google.android.gms/shared_prefs/checkin.xml -
  Contains phone IMEI number and user email address asssociated with the device.
- google.android.gms/databases -
  The entry for the email address mailstore.cfttmobile1@gmail.com showcases email exchange history, which includes sent/received emails, attachments, and other relevant details.

Reviewing these entries can be incredibly useful in a case due to the vast amount of data available we will be able to potentially track down the route that evidence took on a users device.

The next stage of the lab we will turn our attention to the extracted call logs tab which contains outgoing/incoming call entries with their associated metadata.

In conclusion, this lab taught us the importance of reviewing phone system logs and how they can potentially assist in a real world investigation. Being able to sufficiently interpret and analyze various types of file system metadata is valuable skill to have.

1.8. Log Capturing and Interpretation

Log Capturing and Interpretation refers to procedures taken when extracting and analyzing system event log data. Event log analysis can be crucial to a Digital Forensics case due to the vast amount of system activity covered. In this lab, we will be utilizing Autopsy as a means to navigate USNJournel entries and system event logs to decipher any meaningful data.

USN Journal entries focus on file modification and deletion history, providing essential details about changes made. System Event Logs cover a wide range of system processes, including user logins, remote connections, system updates, and internet activity. Analyzing these logs can reveal valuable clues and help reconstruct events.

To begin we will analyze the USNJournel entries via Autopsy by first navigating to the following directories

- >$Extend>$UsnJrnlJ & >$UsnJrnlMax - Contains history of modifications and size of journal.
- > Windows > System32 > WinEVT > Logs & TraceFormat - Contains application, system and security event history logs.

Reviewing these logs and being able to sufficiently interpret its results is an incredibly useful skill. The next section of this lab is to take a closer look at individual event logs by their associated ID to uncover more potentially useful data. By navigating to the extracted Windows Event Logs tab we can see a comprehensive history of every system event that has taken place. The followings logs of interest for this lab and their corresponding information are as follows:

- Event ID 4724 - Contains logon and logoff events.
- Event ID 44 - Contains various logs for Security Intelligence, Windows version control and update information.
- Event ID 43 - Contains logs on Windows updates.
- Event ID 19 - Contains logs on whether or not a specified install/update was succesful or not.

The last section of this lab continues reviewing USNJounral data, by combining findings from event logs to USNJournel entries we can further our understanding on evidences route on a system and associated processes used.

In conclusion, we learnt how to sufficiently be able to interpret system event logs and how to correlate entries with Journal data to potentially uncover insightful information.

1.9. Pagefile Analysis

Pagefile Analysis refers to the process of extracting ‘lost’ data that has been stored within system memory and how to interpret its output. In this lab we will be delving into the concept of pagefiles, their location in memory, and the extraction of potentially useful information.

Digital Forensics investigators commonly utilize PageFile Analysis due to its ability to showcase data stored in recent system memory, even after the computer is turned off. This information can provide useful insights to an investigator such as what a user was viewing, doing and any background processes during a moment of time the device was used.

The start off this lab, we will utilize a drive analysis tool such as Autopsy to review the contents of the drives memory. Within the drives root directory we can see hidden system files, such as the pagefile.sys. The pagefile size is determined by the requirements to sufficiently store the RAM memory, the larger the pagefile is the more potential for meaningful data to be extracted.

Autopsy then enables us to extract the pagefile itself where we can then further analyze it’s data through an specialised extracting tool, in this lab, we will be using ’Bulk Extractor’. The Bulk Extractor scan will deliver us a report that has categorized the pagefile by data types and information contained. By reviewing each entry we can see some valuable user data, such as email addresses, messages, visited URLs and even images. However, it’s important to note that since the pagefile serves as a backup memory and not a direct data backup, the contained data may be fragmented and incomplete.

In conclusion, this lab taught us the importance of temporary system memory, and how to utilize software to extract meaningful data that could potentially assist in a real-world scenario.

1.10. Password Cracking

Password Cracking is a crucial aspect of Digital Forensics investigations, as investigators frequently encounter encrypted or password-protected files that often pose a significant challenge. The ability to identify password-protected files and employ password cracking techniques is a crucial skill for digital forensics professionals.

The objective of this lab is to analyze a disk image in search of encrypted or password-protected files that may contain relevant data. In certain cases, individuals attempting to conceal malicious activities may provide hints or passwords within other files as a means of communication or guidance for their group members. To examine the disk image, we will utilize Autopsy to load the FEF (image) and a PDF cracker to understand the procedures involved in brute-forcing a password, both with and without hints such as password length. Using Autopsy, we extracted the encrypted or password-protected files and saved them separately for further analysis.

To accomplish this, we employed the password cracking tool ’pdfcrack’ with the following command to search for the presence of a password within a file:

- pdfcrack –wordlist=/usr/share/john/password.lst Got it.pdf

Upon finding the password, we used the following command to open the file:

- xdg-open Got it.pdf

After retrieving the password from the PDF, we employed the following command with 'fcrackzip' to crack the ZIP file:

- fcrackzip -D -p /usr/share/john/password.lst Zip File.zip

Alternatively, to perform a brute-force password check, the following command from 'fcrackzip' can be implemented:

- fcrackzip -b Zip File.zip

However, it's important to note that this method can be time-consuming, which is why learning how to extract passwords using decryption tools like Autopsy is extremely valuable.

In conclusion, this lab provided us with valuable practical experience in utilizing password cracking tools through various methods of a forensic disk image.

Part 2 - Lab Questions

Question 1

What does "deleting" a file really mean in terms of file system and storage drive (e.g. magnetic drive) operations?

Magnetic drive refers to the internal (hardware) operation process that HDD (Hard Drive Disks) have. HDD - Magnetic drives store data via magnetized particles on a spinning disk/s. In-terms of the process that a magnetic drive takes to delete a file is multifaceted, when a file is first deleted, the system marks the entry as deleted within its stored metadata structure (updates system metadata to indicate that the specified file is no longer accessible). The file system manages the allocation of storage space on the drive and keeps track of the file's physical location. When a file is deleted, the file system removes the references to the file's data clusters, which are the portions of the drive allocated to store the file's content. The space occupied by the file is marked as available for reuse. The above is what happens in terms of system memory - system metadata gets updated to indicate the file is gone and - the drive frees up the physical space it once occupied and states it is able to be reused. However, in terms of magnetic disks the hardware component slightly changes this. With magnetic drives, the file can persist even after deletion due to the files system, reference being removed, not the physical entry itself. The deleted data can/will still survive on the drive until the space that was allocated to that file has been overwritten. This means that even if a file has been deleted and the space has not been overwritten with other data the file is still retrievable. However, when using a drive the chances with time the more the ability to retrieve deleted files is possible, since the space is free, it can very quickly become used again.

Question 2

What happens when you move a file to Recycle Bin on a file system level and does it affect the acquisition and forensic process?

The Recycle Bin is just a special folder which serves as a temporary storage location for "deleted" files, this means the full files integrity is still in place and can be fully 100% recoverable. Recycle Bin files still hold their designated spot in memory. This means its space in memory cannot be overwritten and does not risk being overwritten accidentally. However, moving this file does change its base accessibility, relocating it to a recycle bin essentially marks the file as "hidden" from its original location. This means it is currently removed from regular explorer browsing. In terms of affecting the acquisition and forensic process, moving files to the recycle bin does not affect the integrity of data acquisition, however, the process of restoring files from the recycle bin does require in-depth documentation as per the Chain of Custody principle. However, this answer changes if the user did fully delete a file "emptying" the recycle bin of its contents. The above, question 1, refers to this process.

Question 3

Explain how you can identify the format and fix a graphics file which the forensics tool you are using is not able to recognize and open.

Using a hex editor such as HxD we are able to manually check for files by their corresponding hex header and footer values. You can identify certain files, as mentioned above, by their header value. For example the following files with their corresponding hex header value;

- JPG; FF D8 FF E0
- PNG; 89 50 4E 47 0D 0A 1A 0A
- PDF; 25 50 44 46 2D
- GIF; 47 49 46 38 37 61
- TIFF; 49 49 2A 00
- ZIP; 50 4B 03 04
- RAR; 52 61 72 21 1A 07 00
- MP3; 49 44 33
- WAV; 52 49 46 46

In terms of fixing a graphics file, you are able to manually save the file by locating its corresponding footer value and saving the selection with its corresponding file extension type. However sometimes you can come into contact with a false-positive code block and you will need to check this off by reviewing the following block of hex data to see if the values continue or not, if they do continue you are not at the end of the block, for example, the end of an image block should look like; "00 00 00 00".

Question 4

Explain the two different types of steganography and how they are performed.

The first form of steganography involves implanting another file within another 'base' file, for example, during our lab 'Steganography and Alternate Streams'[1.1] we analyzed a JPG image by uploading it to HxD and using the search function checked the presence for any other file headers. In this case, we got a hit on '89 50 4E 47 0D 0A 1A 0A' which is a PNG header value, however, the base file being a JPG has the header value of 'FF D8 FF E0'. This means that the PNG has been embedded (hidden) within the base JPG image. This can be easily extracted by selecting the block by its header/footer offset locations and saving it with its corresponding file type. This is one form of Steganography, hiding images/files within a 'base', seemingly legitimate file.

The next form of hiding data is via Alternative Data Streams. We delved into this during the same lab mentioned above, 'Steganography and Alternate Streams'. Alternate Streams refers to the process of hiding a file within system memory, which in-turn is not available through base explorer searching. To do this, we used a terminal such as CMD with administrative privileges and can hide a text file within a base file, and in our case, an executable called 'Legitimate_Program.exe' by implementing the following command (to create and edit):

- type Legitimate_Program.exe > Legitimate_Program.exe:secret.txt
- notepad Legitimate_Program.exe:secret.txt

By viewing the files within the windows explorer window, this file will not be shown, when using the 'dir' command we are also not able to see the hidden text file. The only way to view this hidden.txt even existed is by using the 'dir /r' command which specifically used to check through all the files, and additional files/directories recursively.

Question 5

Explain how the future of IoT devices will impact digital forensics and an examiner's response to an event involving one or multiple connected devices

A couple reasons, one of which is that as technology advances, the variety of new IoT devices increase that have different means of storing, moving and saving data which means that Digital Forensics software needs to be constantly updating and adapting to be able to handle new/alternative means of data collection. The other reason is that data encryption and data obfuscation methods are constantly advancing and becoming increasingly more difficult to crack.

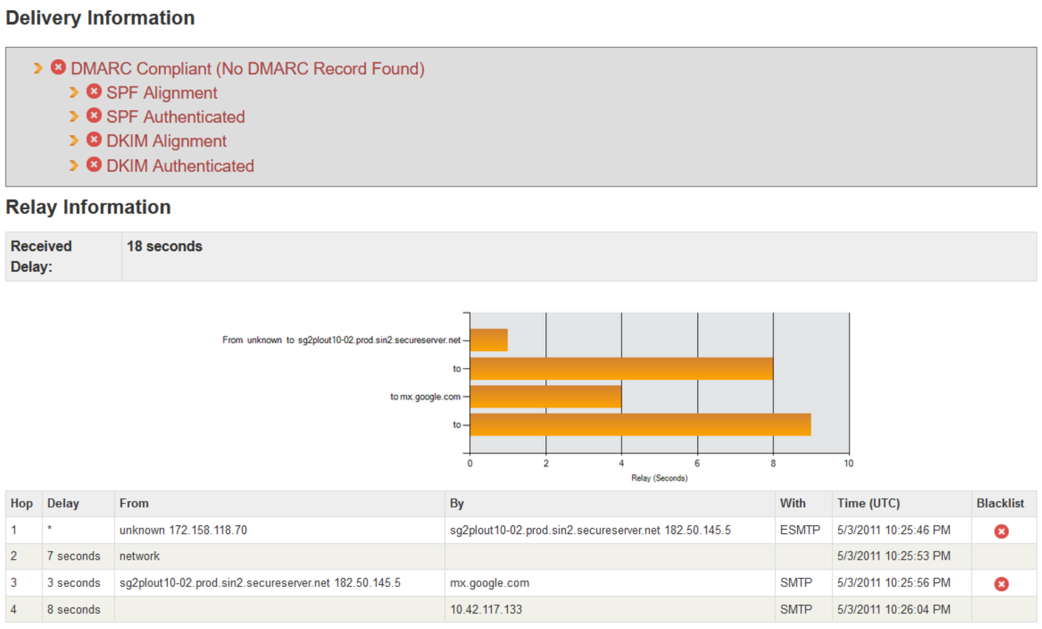
Part 3 - Tasks

Task 1

Analyse the attached email and extract all possible information which could be used in a forensic investigation.

Extracted Data		
Header Name	Header Value	Header Info
Message ID	BF7F22F37F776BA10DAA84290D4BBC 60FC4A03DE@p	Unique email identifier used for tracking message threads, avoiding duplicate messages and assists in message routing/delivery.
From	“Chris F” <user4@sportcarsvendor.info>	Chris Forbs. The message senders details, name and email address.
To	“dan.finnegan” <dan.finnegan@gmail.com>	Dan Finnegan. The message receivers details, name and email address.
Receiver Email Address	dan.finnegran@gmail.com	Receiver email address.
X-Mailer	Vodamail 10	Software or program used to send the email. Vodamail mail client application.
Subject	Re: With 128 GB SSD, Apple MacBook Air 13.3 - $350	Re: signifies this email is a reply email. Contains original mail title.
X-Priority	Normal	Senders set priority, in this case, normal is sufficient as the email is not of high priority.
Sender Policy Framework (SPF)	Error, DNS timeout, invalid email address; user1@sportcarsvendor.info	SPF error check.
Reply-To	chris$_f orb@live.com$	The respondents email address.
Content-Type	multipart/related	Signifies the message structure has been divided into multiple sections.
Date	Wed, 04 May 2011 01:06:15 +0300	Date the email was initially sent.
X-Unsent	1	The ‘1’ value signifies the email was either composed and not sent, not delivered successfully or was blocked.
Mime-Version	1.0	MIME - Multipurpose Internet Mail Extensions, this protocol allows for email content to contain non-ASCII characters. This email corresponds with specifications from the MIME 1.0 version.
Received	from unknown (172.158.118.70) by sg2plout10-02.prod.sin2.secureserver.net (182.50.145.5) with ESMTP; 03 May 2011 22:25:46 -0000	Represents the first/original server/network node that processed this message’s delivery. IP Address 172.158.118.70 points to London UK, could be the sender’s device or server IP address.
Received	Wed, 04 May 2011 01:06:15 +0300qmail 19185 invoked from network); 3 May 2011 22:25:53 -0000	Represents the second server the message travels through. Invoked by qmail which is a mail transfer agent.

Authentication-Results	mx.google.com; spf=temperror (google.com: error in processing during lookup of user1@sportcarsvendor.info: DNS timeout) smtp.mail=user1@sportcarsvendor.info	Authentication was not able to take place due to a DNS timeout when attempting to process the email.
Received-SPF	Error (google.com: error in processing during lookup of user1@sportcarsvendor.info: DNS timeout) client-ip=182.50.145.5;	SPF error, SPF represents the authentication of the email, which is used to prevent spoofing. This failed either due to the SPF record retrieval not responding in time or due to the SPF record not being reachable (entry could possibly not exist).
Return-Path	<user1@sportcarsvendor.info>	<user1@sportcarsvendor.info>
Received	by 10.42.117.133 with SMTP id t5mr561320icq.490.1304461564856; Tue, 3 May 2011 15:26:04 -0700 (PDT)	Represents the last server the message travels through. Servers private IP Address, unable to get more information, could be used as an internal address.
Received-To	by 10.52.187.136 with SMTP id fs8cs74847vdc; Tue, 3 May 2011 15:26:05 -0700 (PDT)	Messages final destination, IP address is private, could be used as an internal address.
Delivered-To	dan.finnegan@gmail.com	Receiver's mail address.


Delivery Information

- DMARC Compliant (No DMARC Record Found)
  - SPF Alignment
  - SPF Authenticated
  - DKIM Alignment
  - DKIM Authenticated

Relay Information

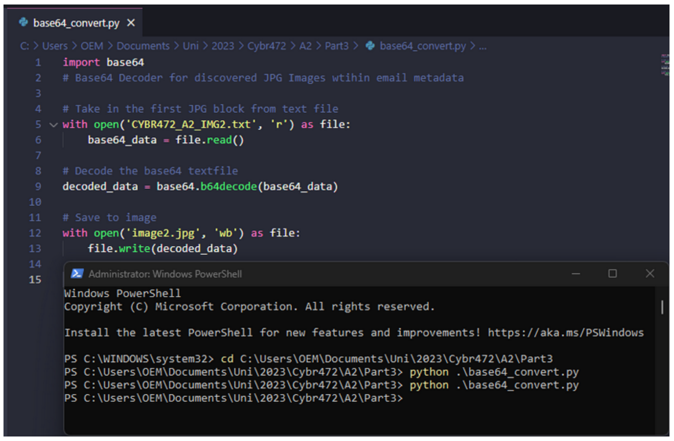
Received Delay:	18 seconds

Hop	Delay	From	By	With	Time (UTC)	Blacklist
1	*	unknown 172.158.118.70	sg2plout10-02.prod.sin2.secureserver.net 182.50.145.5	ESMTP	5/3/2011 10:25:46 PM	✖
2	7 seconds	network			5/3/2011 10:25:53 PM	
3	3 seconds	sg2plout10-02.prod.sin2.secureserver.net 182.50.145.5	mx.google.com	SMTP	5/3/2011 10:25:56 PM	✖
4	8 seconds		10.42.117.133	SMTP	5/3/2011 10:26:04 PM	

Figure 1:
Google Header Analyzer - The above figure is the output when uploading the email metadata to an 'Email Header Analyzer' [2] similar to the implementation used in lab 'Email Analysis' [1.3]

Cont.

Next step in my analysis is to review the attached JPG images contained within the email metadata. Due to the images being encoded with base64 I created a simple script to run them through to extract the images as seen below:


base64_convert.py
C: > Users > OEM > Documents > Uni > 2023 > Cybr472 > A2 > Part3 > base64_convert.py > ...
Administrator: Windows PowerShell

```python
import base64
# Base64 Decoder for discovered JPG Images wtihin email metadata

# Take in the first JPG block from text file
with open('CYBR472_A2_IMG2.txt', 'r') as file:
    base64_data = file.read()

# Decode the base64 textfile
decoded_data = base64.b64decode(base64_data)

# Save to image
with open('image2.jpg', 'wb') as file:
    file.write(decoded_data)
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\WINDOWS\system32> cd C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Part3
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Part3> python .\base64_convert.py
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Part3> python .\base64_convert.py
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Part3>
```

Figure 2: Decoding JPG Images

The following images are the JPGs that were attached to the email.

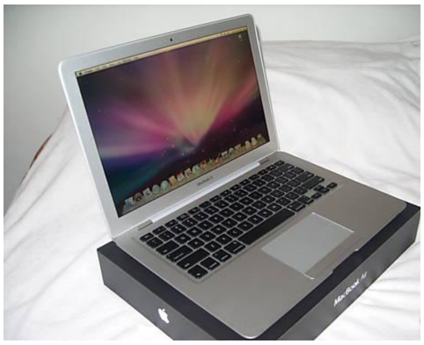


Figure 3: Decoded Image 1

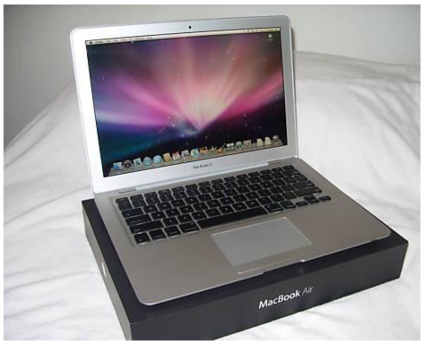
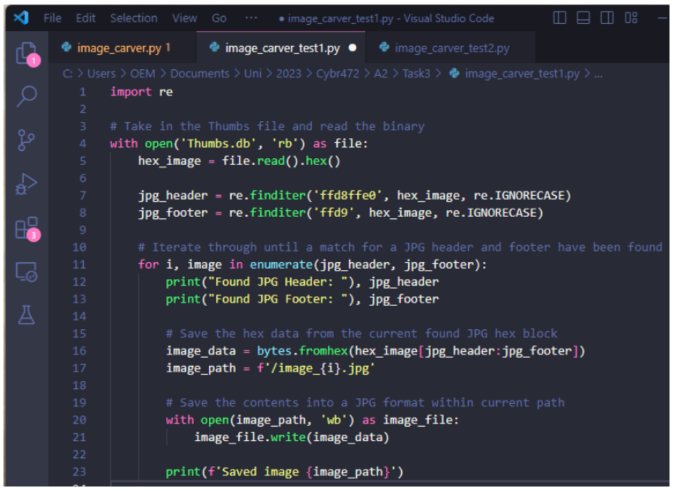


Figure 4: Decoded Image 2

Task 3

Write a script (Any programming language) to automate the process of extracting images from the attached Thumbs.zip file

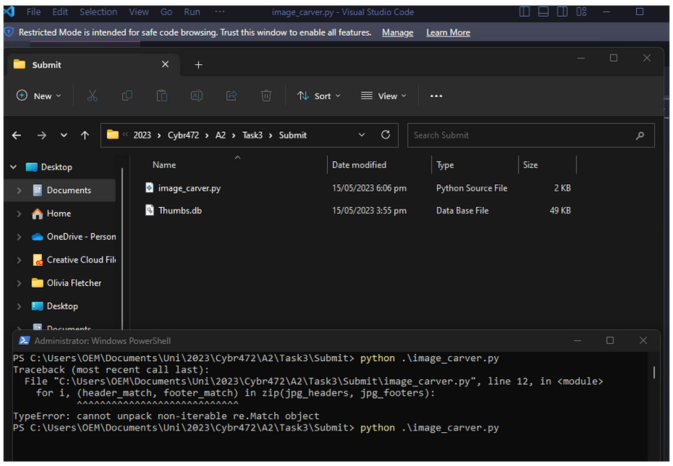
The below image showcases the script I created to complete this task. Utilizing the re library from Python I am able to quickly and efficiently search through the database file by checking for the presence of JPG header/footer hex values.


File Edit Selection View Go ···
● image_carver_test1.py - Visual Studio Code
image_carver.py 1
image_carver_test1.py
image_carver_test2.py
C: > Users > OEM > Documents > Uni > 2023 > Cybr472 > A2 > Task3 > image_carver_test1.py > ...

```python
import re

# Take in the Thumbs file and read the binary
with open('Thumbs.db', 'rb') as file:
    hex_image = file.read().hex()

    jpg_header = re.finditer('ffd8ffe0', hex_image, re.IGNORECASE)
    jpg_footer = re.finditer('ffd9', hex_image, re.IGNORECASE)

    # Iterate through until a match for a JPG header and footer have been found
    for i, image in enumerate(jpg_header, jpg_footer):
        print("Found JPG Header: "), jpg_header
        print("Found JPG Footer: "), jpg_footer

        # Save the hex data from the current found JPG hex block
        image_data = bytes.fromhex(hex_image[jpg_header:jpg_footer])
        image_path = f'/image_{i}.jpg'

        # Save the contents into a JPG format within current path
        with open(image_path, 'wb') as image_file:
            image_file.write(image_data)

        print(f'Saved image {image_path}')
```

Figure 5: Image Carver Script

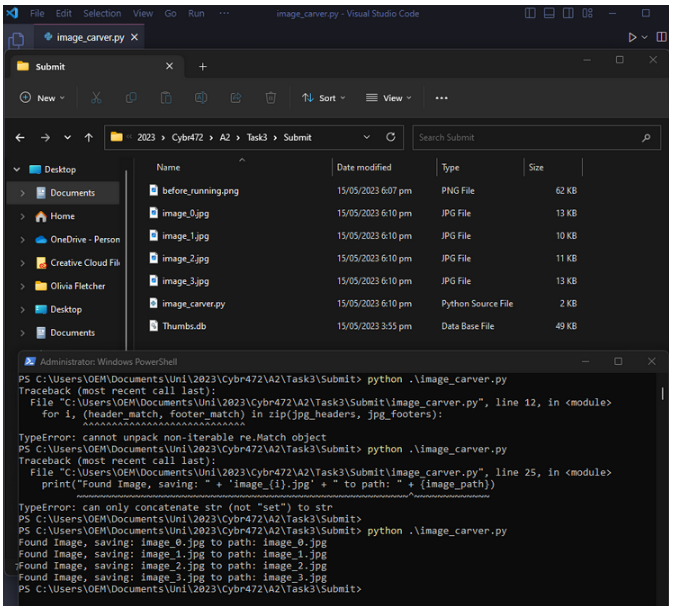
The below two figures showcase the successful execution of the above script, followed by the extracted images:


File Edit Selection View Go Run ···
image_carver.py - Visual Studio Code
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
Submit
New
Sort
View
2023 > Cybr472 > A2 > Task3 > Submit
Search Submit
Desktop
Documents
Home
OneDrive - Person
Creative Cloud Fil
Olivia Fletcher
Desktop
Documents
Administrator: Windows PowerShell

Name	Date modified	Type	Size
image_carver.py	15/05/2023 6:06 pm	Python Source File	2 KB
Thumbs.db	15/05/2023 3:55 pm	Data Base File	49 KB

```
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit> python .\image_carver.py
Traceback (most recent call last):
  File "C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit\image_carver.py", line 12, in <module>
    for i, (header_match, footer_match) in zip(jpg_headers, jpg_footers):
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^
TypeError: cannot unpack non-iterable re.Match object
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit> python .\image_carver.py
```

Figure 6: Before Script Execution



Name	Date modified	Type	Size
before_running.png	15/05/2023 6:07 pm	PNG File	62 KB
image_0.jpg	15/05/2023 6:10 pm	JPG File	13 KB
image_1.jpg	15/05/2023 6:10 pm	JPG File	10 KB
image_2.jpg	15/05/2023 6:10 pm	JPG File	11 KB
image_3.jpg	15/05/2023 6:10 pm	JPG File	13 KB
image_carver.py	15/05/2023 6:10 pm	Python Source File	2 KB
Thumbs.db	15/05/2023 3:55 pm	Data Base File	49 KB

```
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit> python .\image_carver.py
Traceback (most recent call last):
  File "C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit\image_carver.py", line 12, in <module>
    for i, (header_match, footer_match) in zip(jpg_headers, jpg_footers):
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^
TypeError: cannot unpack non-iterable re.Match object
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit> python .\image_carver.py
Traceback (most recent call last):
  File "C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit\image_carver.py", line 25, in <module>
    print("Found Image, saving: " + 'image_{i}.jpg' + " to path: " + {image_path})
          ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~^~~~~~~~~~~~~
TypeError: can only concatenate str (not "set") to str
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit>
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit> python .\image_carver.py
Found Image, saving: image_0.jpg to path: image_0.jpg
Found Image, saving: image_1.jpg to path: image_1.jpg
Found Image, saving: image_2.jpg to path: image_2.jpg
Found Image, saving: image_3.jpg to path: image_3.jpg
PS C:\Users\OEM\Documents\Uni\2023\Cybr472\A2\Task3\Submit>
```

Figure 7: After Script Execution

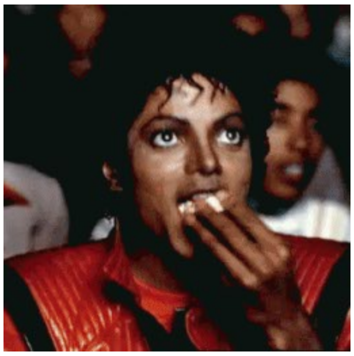


Figure 8: Found Thumbnail Image 1

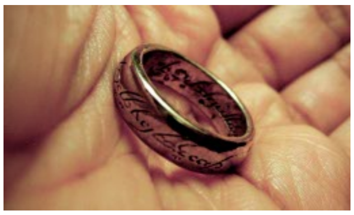


Figure 9: Found Thumbnail Image 2

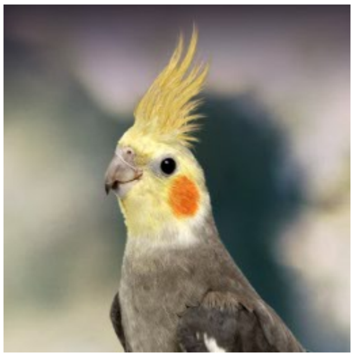


Figure 10: Found Thumbnail Image 3

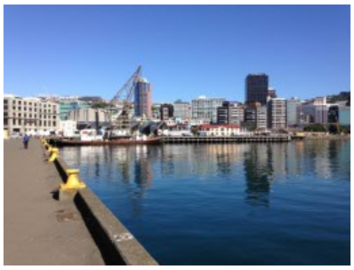


Figure 11: Found Thumbnail Image 4

References

[1] Anders, F., Inger, S., Ausra, D., Jeff, H., Jens, S., Petter, B., Katrin, F., Stefan, A., 23rd May 2017. Digital forensics. https://onlinelibrary.wiley.com/doi/book/10.1002/9781119262442, accessed: 01/05/2023.

[2] MxToolbox, G., 23rd May 2017. Email header analyzer. https://mxtoolbox.com/Public/Tools/EmailHeaders.aspx, accessed: 020/05/2023.

[3] Network Development Group, 1996. Ndg netlab+. https://netlab.ecs.vuw.ac.nz/, accessed: 01/05/2023.