Digital Forensics

Report- s5372043- Gaurav Sood

Evidence A - A disk image of a desktop computer found in Alex's dorm room.

Evidence A had 11 disk files in a zip file, which was extracted into the case folder to investigate. These files were then converted into a single dd file, and an MD5 sum hash was obtained before the investigation to ensure its integrity.

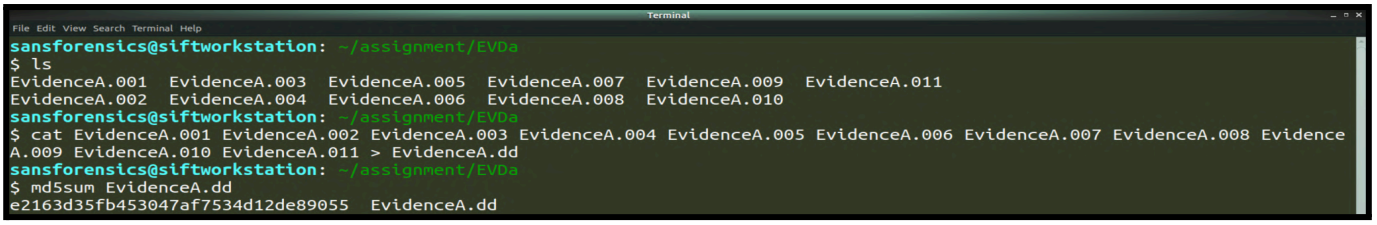


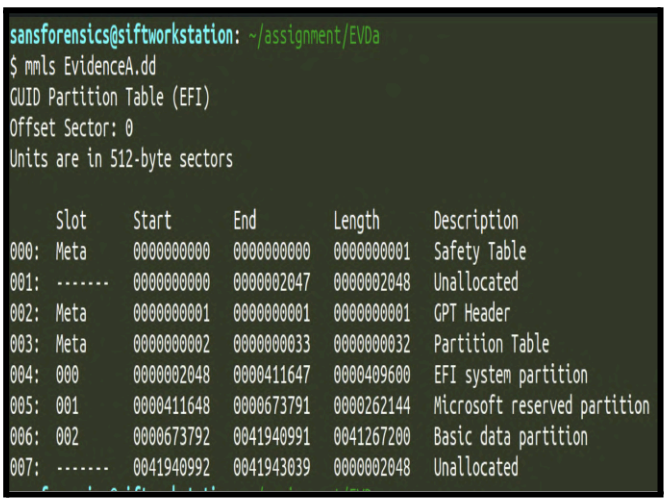
Figure 1. Getting the EvidenceA.dd file and its hash value

The EvidenceA.dd file needed to be mounted to start investigating Alex's mysterious death.

To begin with my investigation, I used **img_stat** to get information about the image type. This command provided me with information about image size and sector size(**512**) which will be required later during the mounting process.

Figure 2. Getting the Information on sector size

Next, I used the **'mmls'** command to display the layout of the partitions within the disk image. This command provided me with information about disk structure and partition layout as well as identifying areas where a large amount of data resides. From mmls we get to know there is a large amount of data stored in partition 006. My next task was to get information about the partition 006 file format. I used the **fsstat** command to get the details of the file partition structure. The partition is **NTFS** which hints that it is a **Windows** file and has a large space allocated to it.



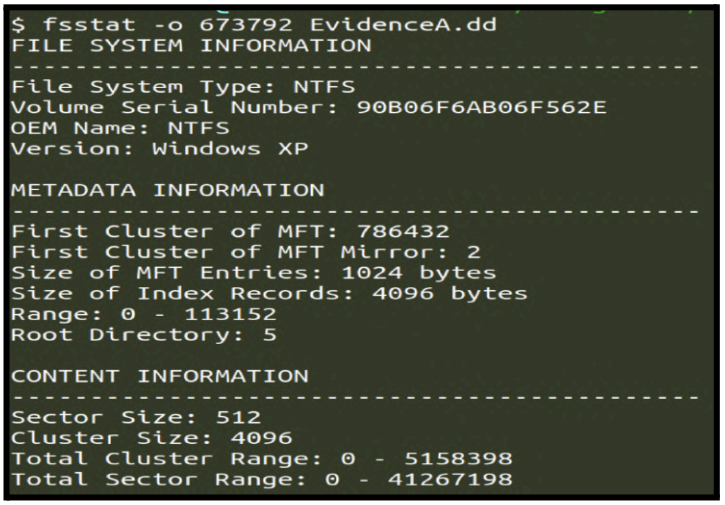


Figure 3. Getting the Partition layout for the file ***Figure 4. Getting the details for the disk partition***

After getting the information, my next task was to mount the file in mnt/windows_mount, which I did by calculating the offset. The offset here is calculated by the sector size(512) multiplied by the starting offset value of partition 006 i.e.(673792), which gives the final calculation as **344,981,504**.

Figure 5. Mounting the Partition

The EvidenceA.dd file was mounted with **read-only** settings to avoid data alteration.

1. Who is the owner of the computer?

We can find the owner of the laptop by going through the registry hives using **RegRipper**. RegRipper helps us extract and analyse data present in the Windows Registry. Here I used samparse plugins to extract the username information from the SAM database.

```
sansforensics@siftworkstation: /mnt/windows_mount
$ rip.pl -r /mnt/windows_mount/Windows/System32/config/SAM -p samparse | grep Username
Launching samparse v.20200825
Username        : Administrator [500]
Username        : Guest [501]
Username        : DefaultAccount [503]
Username        : WDAGUtilityAccount [504]
Username        : Alex Marshall [1000]
```

Figure 6. Finding the Username of the laptop

From the above image, I learned that the laptop's owner is **Alex.** To confirm my assumption, I used the **profilelist** plugin to identify active or previously active user accounts on the systems and see when they were last modified.

```
$ rip.pl -r /mnt/windows_mount/Windows/System32/config/SOFTWARE -p profilelist
Launching profilelist v.20200518
profilelist v.20200518
(Software) Get content of ProfileList key

Microsoft\Windows NT\CurrentVersion\ProfileList

Path      : %systemroot%\system32\config\systemprofile
SID       : S-1-5-18
LastWrite : 2019-12-07 09:16:08Z

Path      : %systemroot%\ServiceProfiles\LocalService
SID       : S-1-5-19
LastWrite : 2019-12-07 09:16:08Z

Path      : %systemroot%\ServiceProfiles\NetworkService
SID       : S-1-5-20
LastWrite : 2019-12-07 09:16:08Z

Path      : C:\Users\Alex Marshall
SID       : S-1-5-21-212117580-4225460857-3930097821-1000
LastWrite : 2024-08-27 13:19:41Z

Domain Accounts
```

Figure 7. Users ProfileList on the laptop

From the above two images, we can confirm that the recent login and owner of the laptop is **Alex.**

2. What programs have been installed on the computer? What recent programs have been run?

I was able to get programs that were installed on the computer by using the RegRipper plugin **Uninstall** which is responsible for tracking all installed software that appears in the control panel of a Windows system. Some of the programs that I found are **'7-zip'**, '**Microsoft Edge**',' **VLC Media**', and' **Mozilla Firefox**'.

```
$ rip.pl -r /mnt/windows_mount/Windows/System32/config/SOFTWARE -p uninstall
Launching uninstall v.20200525
uninstall v.20200525
(Software, NTUSER.DAT) Gets contents of Uninstall keys from Software, NTUSER.DAT hives

Uninstall
Microsoft\Windows\CurrentVersion\Uninstall

2024-08-27 13:12:17Z
  7-Zip 24.08 (x64) v.24.08

2024-08-27 13:07:35Z
  Mozilla Maintenance Service v.129.0.2

2024-08-27 13:05:11Z
  VMware Tools v.11.1.5.16724464

2024-08-27 13:04:44Z
  Microsoft Visual C++ 2019 X64 Additional Runtime - 14.24.28127 v.14.24.28127

2024-08-27 13:04:43Z
  Microsoft Visual C++ 2019 X64 Minimum Runtime - 14.24.28127 v.14.24.28127

2019-12-07 09:51:08Z
  WIC

2019-12-07 09:16:09Z
  AddressBook
  Connection Manager
  DirectDrawEx
  Fontcore
  IE40
  IE4Data
  IE5BAKEX
  IEData
  MobileOptionPack
  SchedulingAgent
```

```
  SchedulingAgent

Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall

2024-08-28 06:01:06Z
  Microsoft Edge v.92.0.902.67

2024-08-27 13:16:25Z
  VLC media player v.3.0.21

2024-08-27 13:07:35Z
  Mozilla Firefox (x86 en-US) v.129.0.2

2024-08-27 13:04:44Z
  Microsoft Visual C++ 2015-2019 Redistributable (x64) - 14.24.28127 v.14.24.28127.4

2024-08-27 13:04:41Z
  Microsoft Visual C++ 2019 X86 Minimum Runtime - 14.24.28127 v.14.24.28127
  Microsoft Visual C++ 2015-2019 Redistributable (x86) - 14.24.28127 v.14.24.28127.4
  Microsoft Visual C++ 2019 X86 Additional Runtime - 14.24.28127 v.14.24.28127

2023-05-05 12:32:49Z
  Microsoft Edge Update v.1.3.147.37
```

Figure 8. List of programs installed in the Computer

I was able to find the most recent run program by using another RegRipper plugin named **runmru**, which keeps track of the commands in the order they were run. The order is stored in **edcba** format, therefore the most recently run program is **'firefox'** followed by **'mspaint'**,**'notepad'**,**'calc'** and **'cmd'**

```
sansforensics@siftworkstation: /mnt/windows_mount
$ rip.pl -r /mnt/windows_mount/Users/Alex\ Marshall/NTUSER.DAT -p runmru
Launching runmru v.20200525
runmru v.20200525
(NTUSER.DAT) Gets contents of user's RunMRU key

RunMru
Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU
LastWrite Time 2024-08-27 13:13:05Z
MRUList = edcba
a    cmd\1
b    calc\1
c    notepad\1
d    mspaint\1
e    firefox\1
```

Figure 9. Most Recent Run Programs

3. Recover the content of any files in the recycle bin.

The recycle bin contains two deleted files which I copied to my case directory for further investigation.

```
sansforensics@siftworkstation: /mnt/windows_mount/$Recycle.Bin/S-1-5-21-212117580-4225460857-3930097821-1000
$ ls
'$IAUO6YK.txt'  '$ILY0J7N.zip'  '$RAUO6YK.txt'  '$RLY0J7N.zip'  desktop.ini
sansforensics@siftworkstation: /mnt/windows_mount/$Recycle.Bin/S-1-5-21-212117580-4225460857-3930097821-1000
$ cp '$RAUO6YK.txt' /home/sansforensics/assignment/EVDa/
sansforensics@siftworkstation: /mnt/windows_mount/$Recycle.Bin/S-1-5-21-212117580-4225460857-3930097821-1000
$ cp '$RLY0J7N.zip' /home/sansforensics/assignment/EVDa/
```

Figure 10. Copying of the files

In my current directory, I tried opening the zip file using the command unzip, but it was protected with a password. I had to then crack the password with **fcrackzip** using a **rockyou.txt** dictionary. The password was found as **'football'** which I later used to unzip the file to find a PDF named **UniversityWarning_Letter.pdf**

The pdf states Alex is on probation due to his poor performance and past misconduct.

```
sansforensics@siftworkstation: ~/assignment/EVDa
$ unzip '$RLY0J7N.zip'
Archive:  $RLY0J7N.zip
[$RLY0J7N.zip] UniversityWarning_Letter.pdf password:
   skipping: UniversityWarning_Letter.pdf  incorrect password
sansforensics@siftworkstation: ~/assignment/EVDa
$ fcrackzip -u -v -D -p rockyou.txt '$RLY0J7N.zip'
found file 'UniversityWarning_Letter.pdf', (size cp/uc  63178/ 66769, flags 9, chk 7688)


PASSWORD FOUND!!!!: pw == football
sansforensics@siftworkstation: ~/assignment/EVDa
$ unzip '$RLY0J7N.zip'
Archive:  $RLY0J7N.zip
[$RLY0J7N.zip] UniversityWarning_Letter.pdf password:
  inflating: UniversityWarning_Letter.pdf
```

Figure 11. Unzipping '$RLY0J7N.zip' File

Dear Mr. Marshall,

We are writing to inform you that your recent academic performance has raised significant concerns. Our records indicate that you have failed two courses in the previous semester and have missed several key assignments and exams in the current semester.

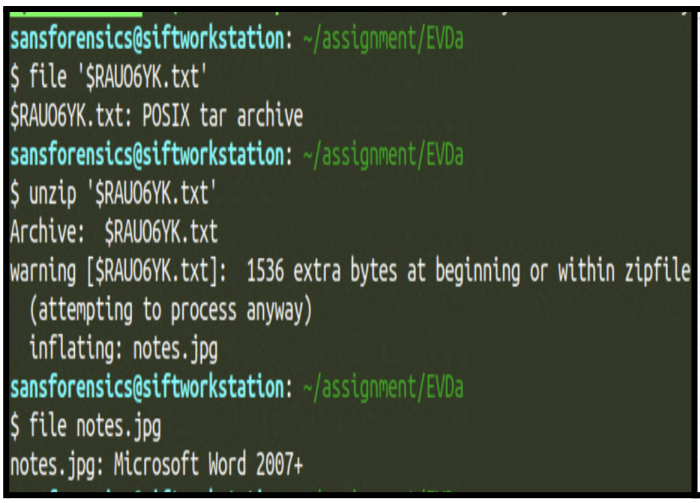
Please be advised that you are now on academic probation. If you do not demonstrate substantial improvement in your academic performance by the end of this term, you may face suspension from the university, and your scholarship may be revoked.

Additionally, we are aware of an incident last semester where you were suspected of unauthorized access to university systems. Although this case was closed due to insufficient evidence, any further infractions will be met with severe disciplinary action.

We strongly encourage you to seek academic counseling and make use of the resources available to help you succeed.

Figure 12. UniversityWarning_Letter.pdf

Next, I tried to determine the file type for '**$RAU06YK.txt'** using the file command, I got the output as **POSIX tar** archive. The tar file is a type of zip file, I then unzipped the '$RAU06YK.txt' using the unzip command and found a document named **notes.jpg**. I again used the file command on the notes.jpg file and got the output as **Microsoft Word 2007+**. After determining the file type, the notes.jpg was opened directly using **LibreOffice**. The notes.jpg contains the '**last will**' of Alex where he apologises to Lily and leaves his belongings to his family.




This is my last will and testament.

If you're reading this, it means something has gone terribly wrong. I'm sorry for everything that's happened, for the pain I've caused. I want my parents to know that I love them, even though we didn't always see eye to eye. I wish I could have been a better son.

To Lily, I'm sorry for all the hurt I caused. You deserve so much better. I hope you can find it someday.

To Sophia... I don't know what to say. I never meant for things to end this way. I just wanted to find a way to be happy, but I made too many mistakes. I hope you can find peace.

All my belongings should go to my family, to help them with anything they need. I don't have much, but it's all I can offer.

Goodbye, everyone.

Alex Marshall

Figure 13. Opening process for notes.jpg file ***Figure 14. Alex Will***

4. Is there evidence that gives an indication of the state of mind of the owner of the computer?

There are multiple pieces of evidence that indicate that Alex was both mentally and psychologically stressed. Some of the findings:

- **Alex's will and testament (Figure 14)**: This evidence suggests that Alex feels hopeless and may consider suicide. His final words are for his family and Lily and he expresses regret for how he had everyone around him.
- **UniversityWarning_Letter.pdf (Figure 12)**: This evidence indicates Alex's poor academic performance. Alex is on academic probation and he might be at risk of suspension with his scholarship being revoked. This letter contributes to his failure, thereby adding more stress and a sense of failure.
- **Dad_Email_March.pdf (Figure 15)**: This evidence indicates that Alex was having family pressure, his dad was disappointed in Alex's recent performance and threatened to withdraw financial support.
- J**ournal1.txt and journal2.txt (Figure 17,18)**: Both of the journals indicate that Alex was facing financial issues and regrets selling exam answers. He also expresses fear about getting caught and losing everything. Alex also talks about his growing depth and how he owes money to Lily and not being able to manage his credit card debt, this shows a sign of entrapment.
- **Debt_Tracker.csv (Figure 16)**: This evidence shows that Alex is in serious debt, with the deadline on his door. He is struggling to pay the money and has no way out of the debt trap.
- **InternshipOffer_Reynolds.pdf (Figure 19)**: This evidence shows that Alex has been offered an Internship and it highlights the importance of him needing to be focused on his academics, which Alex may struggle with given his current mental state.
- **Mum_Email_April.jpg (Figure 20)**: This evidence shows that Alex's mother was providing him with emotional support and mentions that his family cared for Alex. This may contribute to Alex's current mental state, as he might now meet their expectations.

All the above evidence shows that Alex was overwhelmed with family expectations, poor academic performance and financial debts which contributed to his mental and emotional stress.

From: Oliver Marshall <oliver.marshall@email.com>

To: Alex Marshall <alex.marshall@university.edu>

Date: March 10, 2024

Subject: Your Future

Alex,

I've just received your latest report card, and I'm deeply disappointed. This is not the level of performance I expect from my son. You have all the potential in the world, but you're throwing it away by not applying yourself.

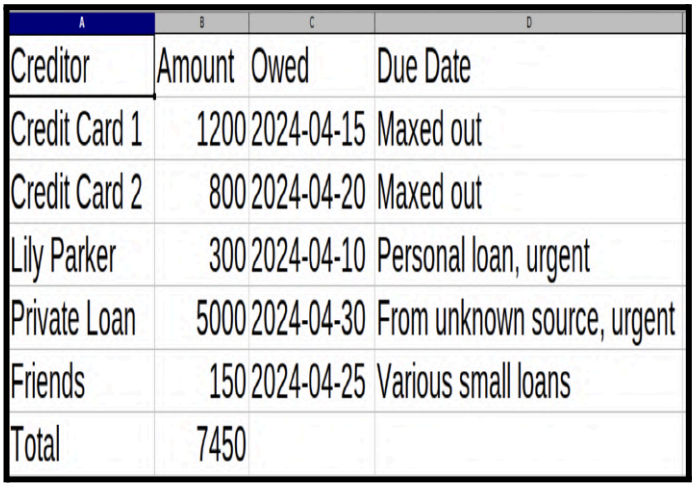
I've warned you before that you're running out of chances. If you don't get your act together, I will have no choice but to cut off your financial support. You need to understand that the Marshall family has a reputation to uphold, and you're not living up to it.

You need to focus on your studies and start making decisions that reflect the kind of man you want to become. I didn't work my entire life for you to squander these opportunities.

Consider this your final warning.

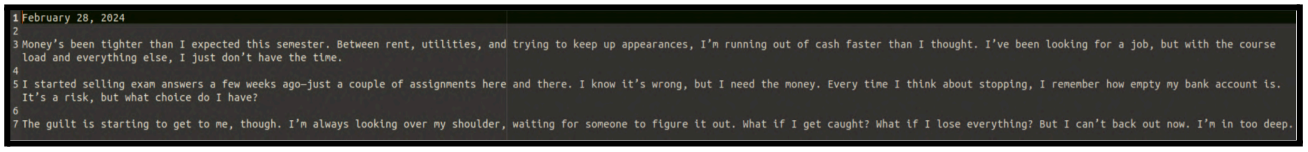
Dad

Figure 15. Dad_Email_March.pdf



Creditor	Amount	Owed	Due Date
Credit Card 1	1200	2024-04-15	Maxed out
Credit Card 2	800	2024-04-20	Maxed out
Lily Parker	300	2024-04-10	Personal loan, urgent
Private Loan	5000	2024-04-30	From unknown source, urgent
Friends	150	2024-04-25	Various small loans
Total	7450		

Figure 16. Debt_Tracker.csv



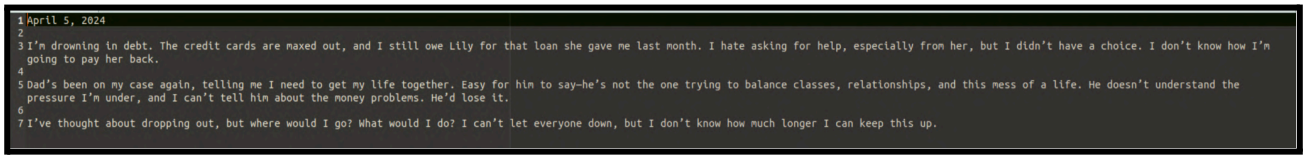
February 28, 2024

Money's been tighter than I expected this semester. Between rent, utilities, and trying to keep up appearances, I'm running out of cash faster than I thought. I've been looking for a job, but with the course load and everything else, I just don't have the time.

I started selling exam answers a few weeks ago—just a couple of assignments here and there. I know it's wrong, but I need the money. Every time I think about stopping, I remember how empty my bank account is. It's a risk, but what choice do I have?

The guilt is starting to get to me, though. I'm always looking over my shoulder, waiting for someone to figure it out. What if I get caught? What if I lose everything? But I can't back out now. I'm in too deep.

Figure 17 journal1.txt



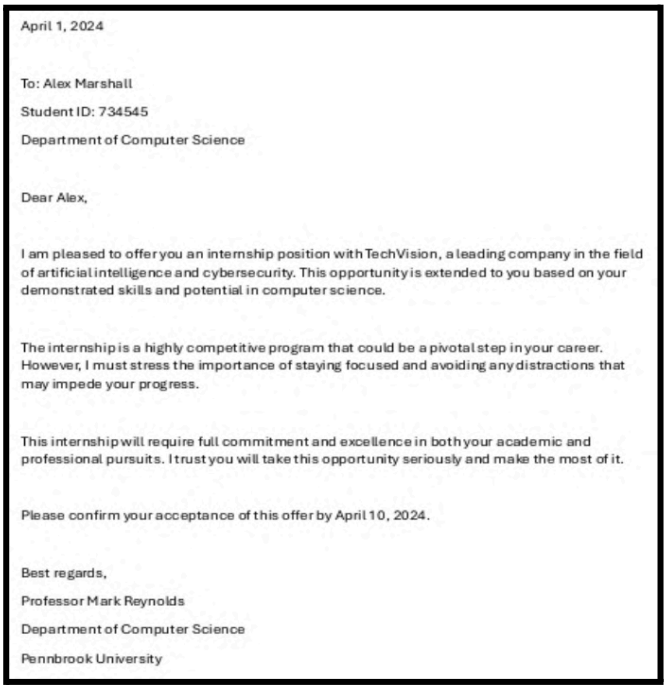
April 5, 2024

I'm drowning in debt. The credit cards are maxed out, and I still owe Lily for that loan she gave me last month. I hate asking for help, especially from her, but I didn't have a choice. I don't know how I'm going to pay her back.

Dad's been on my case again, telling me I need to get my life together. Easy for him to say—he's not the one trying to balance classes, relationships, and this mess of a life. He doesn't understand the pressure I'm under, and I can't tell him about the money problems. He'd lose it.

I've thought about dropping out, but where would I go? What would I do? I can't let everyone down, but I don't know how much longer I can keep this up.

Figure 18 journal2.txt



April 1, 2024

To: Alex Marshall

Student ID: 734545

Department of Computer Science

Dear Alex,

I am pleased to offer you an internship position with TechVision, a leading company in the field of artificial intelligence and cybersecurity. This opportunity is extended to you based on your demonstrated skills and potential in computer science.

The internship is a highly competitive program that could be a pivotal step in your career. However, I must stress the importance of staying focused and avoiding any distractions that may impede your progress.

This internship will require full commitment and excellence in both your academic and professional pursuits. I trust you will take this opportunity seriously and make the most of it.

Please confirm your acceptance of this offer by April 10, 2024.

Best regards,

Professor Mark Reynolds

Department of Computer Science

Pennbrook University

Figure 19 IntenshipOffer_Reynolds.pdf

April 5, 2024

Dear Alex,

I hope you're doing well. I've been thinking about you a lot lately and wanted to write you a quick note to check in.

I know things have been tough with school, and your father can be hard on you. He just wants what's best for you, but I understand it can be overwhelming. I want you to know that I'm here for you, no matter what.

If you ever feel like you need someone to talk to, please don't hesitate to reach out. I'm worried about you, and I hope you're taking care of yourself. Life can be challenging, but you're strong, and I know you'll find your way.

Remember that we love you very much, and we're always here for you.

Love,
Mum

Figure 20Mum_Email_April.jpg

Evidence B – Network capture of the dormitory network.
Evidence B was provided with a pcap file which I examined using **Wireshark** and **NetworkMiner** Tool.

5. Who are the people communicating in the transmission? When does the first transmission begin and the last transmission finish?
To get the people communicating in the transmission I used Wireshark, where I followed the TCP Stream to find the plain text that was not encrypted.

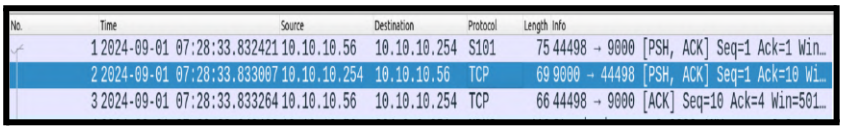


Figure 21. TCP network Capture

Upon following the TCP stream I found various conversations in TCP stream 0,3 and 371 respectively

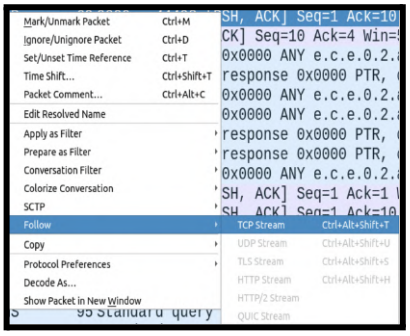


Figure 23. TCP Stream

The following names were found in the conversations:
AlexM21, DormKing, BookWorm, PartDude, LilyLaw, ArtLover99
In the TCP Stream 0, AlexM21 and ArtLover99 are having a conversation. In TCP Stream 3, AlexM21, DormKing, BookWorm and PartDude are having a conversation. Finally, In the TCP stream, 371 AlexM21 and LilyLaw are having a conversation.

Figure 23. TCP Stream 0 conversations

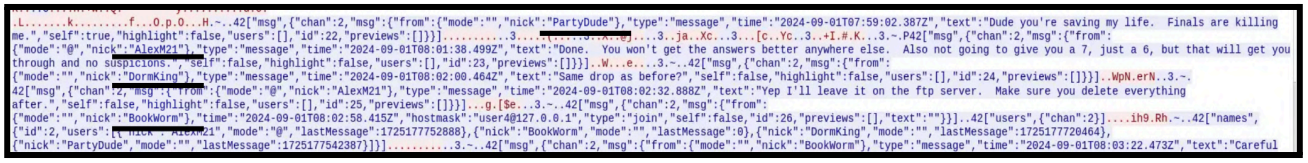


Figure 24. TCP Stream 3 conversations

Figure 25. TCP Stream 371 conversations

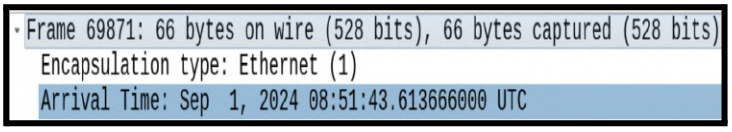


Figure 26. Frame 69871

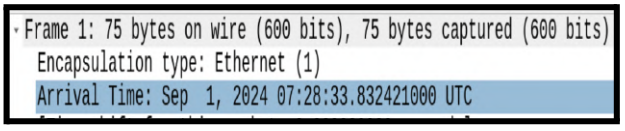


Figure 27. Frame 1

I was able to get transmission time by going through the first and the last frame from the Wireshark network capture. The first transmission was on Sep 1, 2024, 07:28:33 UTC and the last transmission was on Sep 1, 2024, 08:51:43 UTC. For the conversation, the time ranges from 07:57:11 to 08:46:20**(Fig 88)**

6. What browsers, operating systems, and IP addresses are used by the communication endpoints?

I was able to take out the list of IP addresses by using NetworkMiner. NetworkMiner is a tool for analysing traffic and extracting valuable information such as IP addresses. The IP addresses included in the transmission

10.10.10.1 (Firefox/129.0)	10.10.10.22(Firefox/129.0)
10.10.10.33(Firefox/129.0)	10.10.10.44(Firefox/15.0.1)
10.10.10.56(Firefox/129.0)	10.10.10.254

To find the browser and operating systems, I copied the user-agent string that I found on TCP Stream 4 and searched it on Safari to get the details of the browser and operating system used by the communication endpoint.User-Agent.net suggests the Operating system used is "**Ubuntu**" and the Browser used is "**Firefox**" (*User Agents*, 2024).

```
GET /canonical.html HTTP/1.1
Host: detectportal.firefox.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:129.0) Gecko/20100101 Firefox/129.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cache-Control: no-cache
Pragma: no-cache
Connection: keep-alive
```

Figure 28 User-Agent String Found in TCP Stream 4

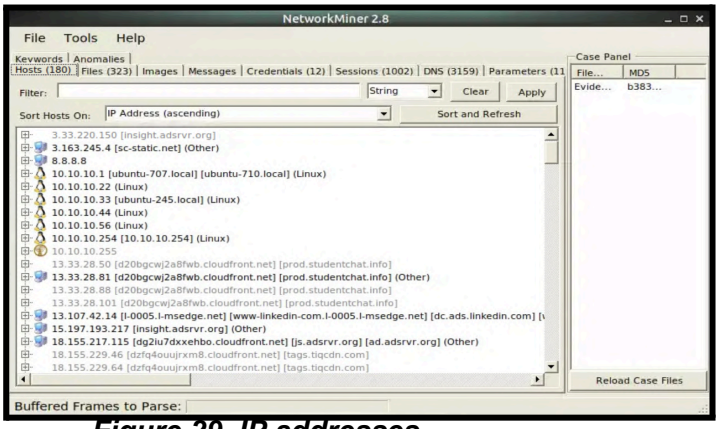


Figure 29. IP addresses

Figure 30. User_agent.net

7. What files were transmitted on the local network?

In **TCP stream 3** there is a conversation of AlexM21 mentioning uploading a file on a FTP server. I then filtered network packets using **ftp** and there I found that an **SSLKey** had been transferred using FTP. I filtered network packets again using **ftp-data** and recovered the SSLKey by saving it in raw format. The SSLKey then was added to the Wireshark by going through the preferences and adding the raw SSLKey to the **TLS protocol**. This allowed me to filter network packets using **http2**.

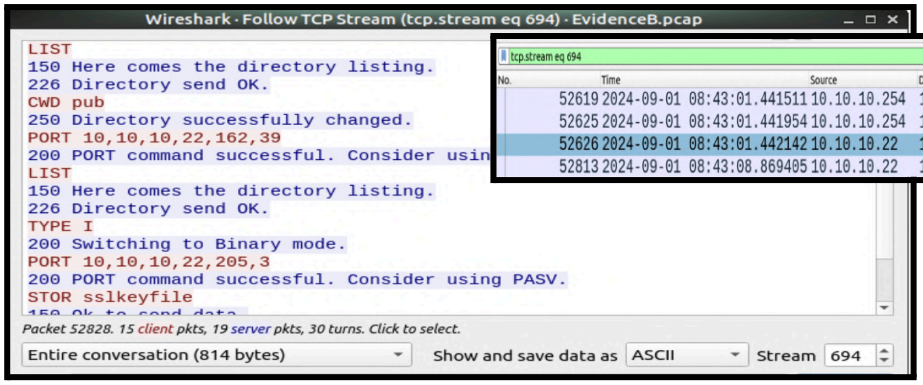


Figure 31. Storing of SSL key files in FTP TCP Stream 694

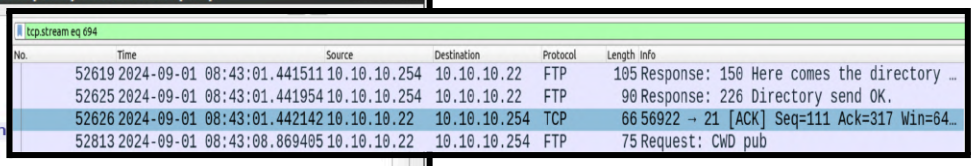


Figure 32. Packet filter using ftp

Figure 33. SSL key found in ftp-data (Tcp stream 701)

By going through the http2 packets I found a packet named **ExamSample.png**. I then tried to investigate further by scrolling through the packets and found a hhtp2 packet as **DATA[25] (PNG)** Figure 35. I then tried to look through the stream data and found Body Fragments with 120094 bytes. The stream data was then investigated by going through the option '**Show packets bytes**'. Here I found **ExamSample.png** which had **BIO101 Final Term Questions** and a **secret.key**. I was able to obtain the secret key by following the http2 stream. The secret key was present at **substream 23**

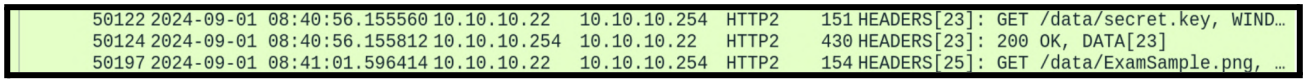


Figure 34. Analysing HTTP2 packets

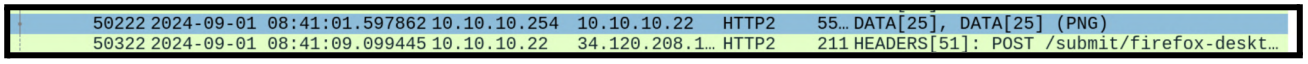


Figure 35. Discovered PNG File

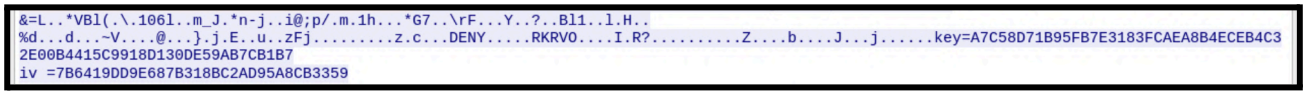


Figure 36. Secret.key And iv (substream 23)

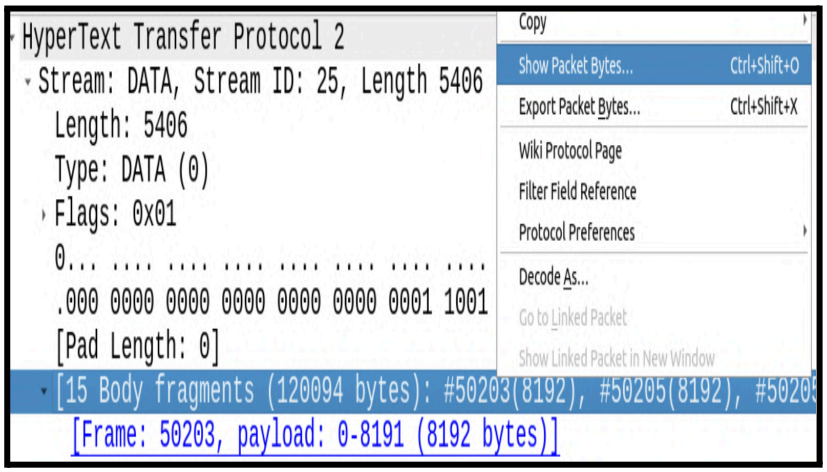


Figure 37. Stream data found in Data[25] PNG

Figure 38. ExamSample.png

On investigating ftp network packets, I found two files that were shown in a public directory and these files **were uploaded previously** (as shown in the conversation between Alexm21, DormKing and BookWorm - Figure 24). The files are **test.txt** and **upload.txt** which can be found in the **tcp stream 699**

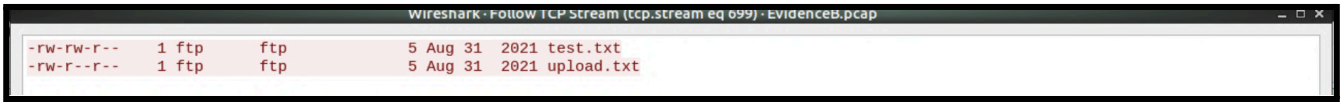


Figure 39. test.txt and upload.txt file in a pub directory (tcp stream 699)

The files that I was able to recover are **SSLKey**, **secret.key** and **ExamSample.png**.

8. What is the relationship of the people communicating in the network capture? How are they related to the victim?

There are 6 people in the conversation as found in Figures 24, 25 and 26 respectively. All the evidence indicated that the people in the conversation were friends and they studied at the same university.
Relationship with the victim(**AlexM21**):

- **LilyLaw (Figure 25):** Lily seems to be in a romantic relationship with Alex and she is frustrated about Alex being with someone else
- **ArtLover(Sophia, Figure 23):** Sophia is also in a romantic relationship with Alex She too feels frustrated about the unclear relationship between them
- **DormKing (Figure 24):** DormKing is involved in the academic cheating scheme with Alex. DormKing tries to manipulate Alex into not paying 200$ for the Bio exam answers and instructs PartyDude to exploit Alex by grabbing the session key from Alex's PC.
- **PartyDude (Figure 24):** PartDude is too involved in the academic cheating scheme with Alex but has less knowledge of the situation as compared to DormKing. PartDude is more dependent on Alex's help for the answer and expresses gratitude but follows DormKing instructions to exploit Alex and not pay 200$ for the exam answers.
- **BookWorm (Figure 24):** BookWorm did not get involved in the cheating scheme because he feared getting caught by the professors. He even warns them(AlexM21, PartDude, DormKing) about faculty cracking down on these activities. BookWorm does not have a close relationship with Alex and acts as an observer in the situation.

All six people had different relationships with Alex. ArtLover and LilyLaw were involved in a romantic relationship with Alex, whereas DormKing and PartyDude were primarily interested in exploiting Alex for academic gains. BookWorm was on the neutral side and showed concern about academic cheating.

Evidence C – A memory dump of a personal laptop found in Alex's bedroom.

Evidence C was provided as EvidenceC.vnem file, which is a memory dump of a ram found in one of the personal laptops in Alex's bedroom. To investigate the file I used a tool called **Volatility**. Volatility helps in digital forensics to examine system memory (RAM) and allows us to extract valuable information from a memory dump. To begin the investigation, I used the plugin **imageinfo** to get the list of **system profiles** found in the memory dump. The correct system profile will help us to perform a proper analysis. The imageinfo suggests that the memory dump is taken from a **Win7SP1x64** which is a Windows 7 machine with 64bit architecture.

```
sansforensics@siftworkstation: ~/assignment/evdc
$ vol.py -f EvidenceC.vmem imageinfo
Volatility Foundation Volatility Framework 2.6.1
INFO    : volatility.debug    : Determining profile based on KDBG search...
          Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_24000, Win2008R2SP1x64_23418, Win2008R2SP1x64, Win7SP1x64_24000, Win7SP1x64_23418
                     AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
                     AS Layer2 : FileAddressSpace (/home/sansforensics/assignment/evdc/EvidenceC.vmem)
                      PAE type : No PAE
                           DTB : 0x187000L
                          KDBG : 0xf800029f40a0L
          Number of Processors : 1
     Image Type (Service Pack) : 1
                KPCR for CPU 0 : 0xfffff800029f5d00L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2024-09-08 12:26:34 UTC+0000
     Image local date and time : 2024-09-08 22:26:34 +1000
```

Figure 40. System Profile

9. What applications are running on the memory dump computer?

To find the application running on the memory dump, I used volatility plugins **pstree**. Pstree helped me to find all the running processes at the time of the memory dump and arrange the running process in a tree-like structure. On going through the list of applications, I found some of the user applications running on the memory dump: **thunderbird.exe**, **firefox.exe**, **explorer.exe**, and **mynotepad++.exe**. On examining the applications, I found the start time of the user application as **2024-09-8**:**12:02:41**, **08:00:31**, **07:08:29**, and **12:15:06 UTC** respectively. Other applications and processes running on the memory dump can be found in the images provided below.

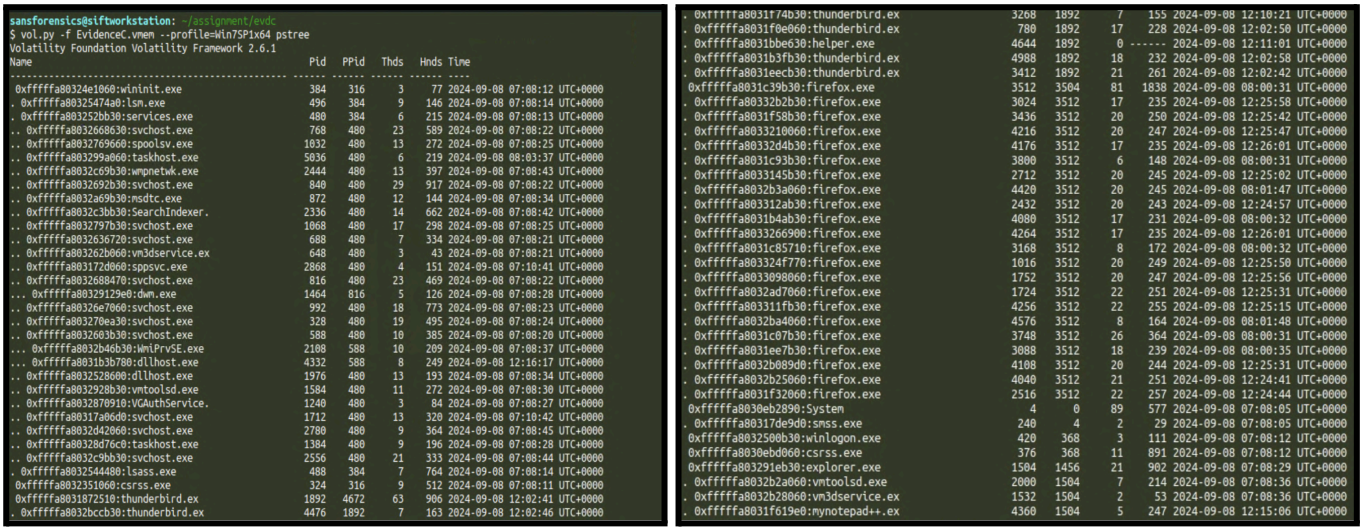


Figure 41. Applications running on the memory dump

10. What web pages has the memory dump computer visited recently?

Initially, to get the web pages visited by the memory dump, I used the following plugins: **firefoxhistory**, **chromehistory**, and **iehistory** to extract browsing history from the memory dump. The firefoxhistory and chromehistory did not yield any results. However, iehistory provided the result of a URI file from the Windows file system.

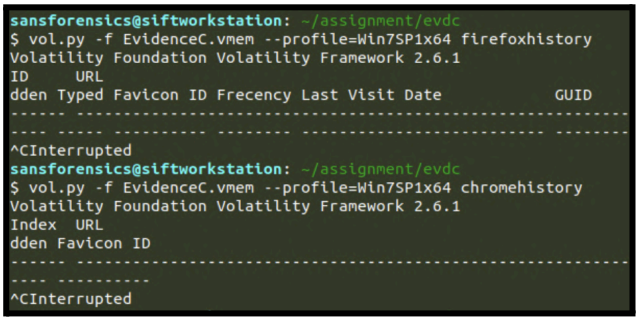


Figure 42. firefoxhistory and chromehistory

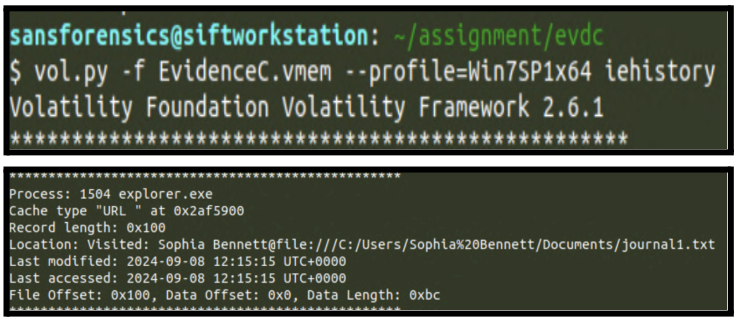


Figure 43. iehistory

To further investigate the recently searched web pages, I used **dumpfile** plugin to extract the files present in the process id **3512** of firefox.exe(figure 41) and saved the dumpfiles in firefoxFiles directory.

Figure 44. Dumping firefox files

On examining the dumped files, I found a file named **file.3512.0xfffffa8031aac110.formhistory.sqlite.dat.** This file appeared to be a sqlite database, so to further investigate I opened this file using sqlite and found the searchbar history dumped by the memory of the laptop.

Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: moz_formhistory — New Record | Delete Record

	id	fieldname	value	timesUsed	firstUsed	lastUsed	guid
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	searchbar-history	download thunderbird 115	1	1725782462716000	1725782462716000	NorhaZ4LRoGMGdBB
2	2	searchbar-history	How to stop someone from leaving you	1	1725798280204000	1725798280204000	8xMxmGvJS4inUXsN
3	3	searchbar-history	Signs someone is cheating	1	1725798292305000	1725798292305000	8JjrLyERRQ69WUMn
4	4	searchbar-history	How to get away with self-defense	1	1725798299826000	1725798299826000	uk5POvm7SNSE2LX5
5	5	searchbar-history	Emergency therapy services	1	1725798313777000	1725798313777000	CmT5RtqRTHG1w2gY

Figure 45. search bar-history

To confirm the recently searched web pages, I analysed the file **file.3512.0xfffffa8031c7fb30.places.sqlite.dat,** as firefox stores the recent browsing data such as browsing history, bookmarks and favicons in places.sqlite file. I used the strings command on the places.sqlite.dat files and grep to extract all the https information present in the file.

```
sansforensics@siftworkstation: ~/assignment/evdc/firefoxFiles
$ strings file.3512.0xfffffa8031c7fb30.places.sqlite.dat | grep "https"
        https://www.qld.gov.aud%
        https://www.verywellmind.comd
        https://www.quora.comd%
        https://www.thunderbird.net
        https://download-installer.cdn.mozilla.net%
        https://download.mozilla.org
        https://ucarecdn.com
        https://www.google.com
        https://support.mozilla.org
        https://www.mozilla.org
```

```
3       https://support.mozilla.org
        https://www.google.com/search?client=firefox-b-e&q=Emergency+therapy+servicesEmergency therapy services - Google Searchmoc.elgoog.www.d
        https://www.verywellmind.com/warning-signs-your-spouse-is-cheating-230065211 Signs of Cheatingmoc.dnimllewyrev.www.d
XVWhat are the signs of a cheating husband or wife? Here are eleven things that could mean your spouse is cheating, such as noticing a change in attitude or appearance.https://www.verywellmind.com/thmb/Iwa6ttLPM
tAgUUclTJITpvbvlbU=/1500x0/filters:no_upscale():max_bytes(150000):strip_icc()/Warning-signs-your-spouse-is-cheating-2300652-V1-6f25e933a9204c04af16c84ce5dec864.png
        https://www.google.com/search?client=firefox-b-e&q=How+to+get+away+with+self-defenseHow to get away with self-defense - Google Searchmoc.elgoog.www.d
        https://www.qld.gov.au/health/mental-health-and-wellbeing/how-to-get-helpua.vog.dlq.www.d
        https://www.google.com/search?client=firefox-b-e&q=Signs+someone+is+cheatingSigns someone is cheating - Google Searchmoc.elgoog.www.d
        https://www.quora.com/How-can-I-stop-people-leaving-meHow to stop people leaving me - Quoramoc.aroug.www.d
 * You are ridiculhttps://qph.cf2.quoracdn.net/main-custom-t-6783-600x315-fwiilmlownyfbruycjcdlavtlxrmdyga.jpeg
        https://www.google.com/search?client=firefox-b-e&q=How+to+stop+someone+from+leaving+youHow to stop someone from leaving you - Google Searchmoc.elgoog.www.d
        https://ucarecdn.com/381c0d21-61f7-46fc-940a-e8e639fb41fd/-/resize/516x/-/format/auto/https___ucarecdn.com_381c0d21-61f7-46fc-940a-e8e639fb41fd_-_resize_516x_-_format_auto_moc.ndceracu.
        https://support.mozilla.org/products/firefoxgro.alllzom.troppus.
        https://www.thunderbird.net/en-US/thank-you/?form=support&utm_content=post_download&utm_source=thunderbird.net&utm_medium=fru&utm_campaign=donation_2023&downloaded=True&download_channel=esrThank You!
s loaded with great features!https://www.thunderbird.net/media/img/thunderbird/thunderbird-256.png
        https://www.thunderbird.net/en-US/thank-you?form=support&utm_content=post_download&utm_source=thunderbird.net&utm_medium=fru&utm_campaign=donation_2023&downloaded=True&download_channel=esrten.dribrednuht
.www.
        https://download-installer.cdn.mozilla.net/pub/thunderbird/releases/115.15.0/win64/en-US/Thunderbird%20Setup%20115.15.0.exeThunderbird Setup 115.15.0.exeten.alllzom.ndc.rellatsni-daolnwod.
```

Figure 46. Webpages found in the memory dump

Thus we confirm the following web pages were searched:

- https://www.quora.com/How-can-I-stop-people-leaving-me
- https://www.verywellmind.com/warning-signs-your-spouse-is-cheating-2300652
- https://www.google.com/search?client=firefox-b-e&q=How+to+get+away+with+self-defense
- https://www.google.com/search?client=firefox-b-e&q=Emergency+therapy+services
- https://download-installer.cdn.mozilla.net/pub/thunderbird/releases/115.15.0/win64/en-US/Thunderbird%20Setup%20115.15.0.exe

11. What is email address of the owner of the memory dump computer and how are they connected to the case?

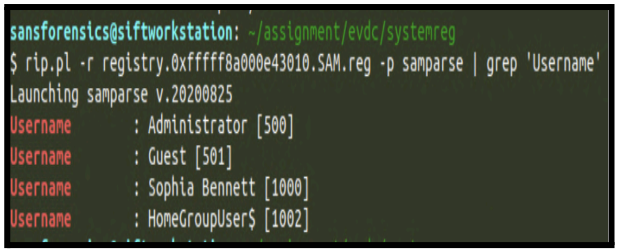
To begin with the investigation, I dumped all the registry files present in the memory dump using the plugin **dumpregistry** to identify the owner of the laptop.

```
sansforensics@siftworkstation: ~/assignment/evdc
$ vol.py -f EvidenceC.vmem --profile=Win7SP1x64 dumpregistry -D systemreg/
Volatility Foundation Volatility Framework 2.6.1
**************************************************
Writing out registry: registry.0xfffff8a000b80410.BCD.reg

**************************************************
**************************************************
Writing out registry: registry.0xfffff8a000024010.SYSTEM.reg
```

Figure 47. Dumping System registry

I then used RegRipper with **samparse** plugin to extract the user account details. SAM stores user account and security credential information on a Windows system. I again used RegRipper with **lastloggedon** plugin to determine the last logged-in user.



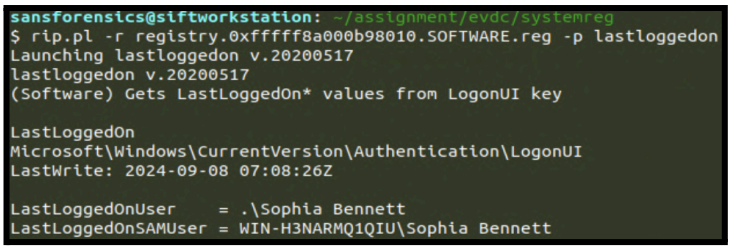


Figure 48. Username Sophia Bennett ***Figure 49. LastLoggenOnUser Sophia Bennett***

The above two images confirm that the owner of the laptop is **Sophia Bennett**.

Now to search for the email address of the owner, I used the strings command on the EvidenceC.vnem file and extracted the “gmail.com” keyword using the grep command.

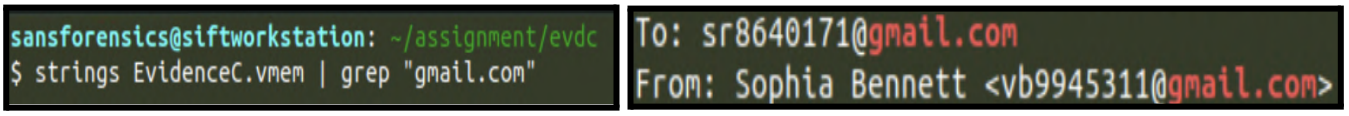


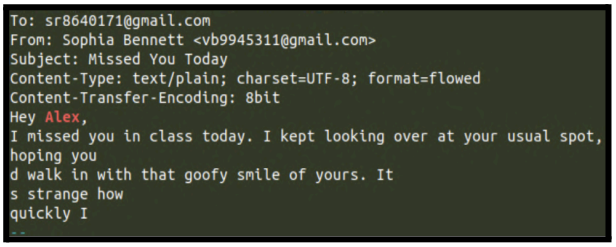
Figure 50. Finding the owner's email address(vb9945311@gmail.com)

The above image confirms the email address of the owner(Sophia Bennett):**vb9945311@gmail.com**.

To get the email address for Alex, I again used the strings command on the EvidenceC.vnem file and this time I extracted information for the keyword “Alex”.

Figure 51. Finding the victim's email address (sr864017@gmail.com)

I was able to find Alex's email address (**sr864017@gmail.com**). The owner of the laptop i.e. Sophia was related in the case as the victim (Alex) had a close/romantic relationship with Sophia. There are many of emotional content in the emails with the Subject stating as **Missed You Today**, **You and Me**, **Feeling Overwhelmed** with the victim Alex



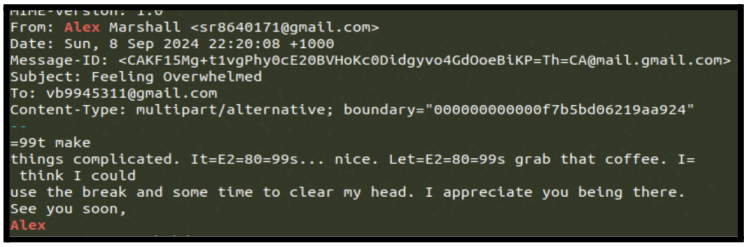


Figure 52. Subject: Missed You Today ***Figure 53. Subject: Feeling Overwhelmed***

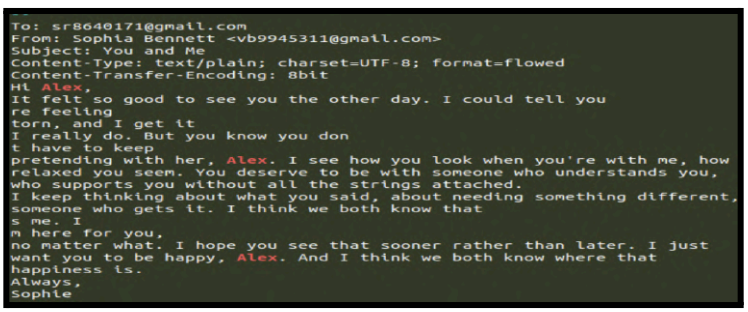


Figure 54: Subject You and Me

All these emails indicate a strong close/romantic relationship between the owner of the laptop and the victim.

12. What is password of the memory dump computer?

I was able to find the password of the memory dump computer by using a plugin **lsadump** with volatility. The lsadump plugins help to extract sensitive information about the user authentication and security policies. The password was found to be '**Alexisbeautiful**'.

```
sansforensics@siftworkstation: ~/assignment/evdc
$ vol.py -f EvidenceC.vmem --profile=Win7SP1x64 lsadump
Volatility Foundation Volatility Framework 2.6.1
DefaultPassword
0x00000000  1e 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0x00000010  41 00 6c 00 65 00 78 00 69 00 73 00 62 00 65 00   A.l.e.x.i.s.b.e.
0x00000020  61 00 75 00 74 00 69 00 66 00 75 00 6c 00 00 00   a.u.t.i.f.u.l...

DPAPI_SYSTEM
0x00000000  2c 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ,...............
0x00000010  01 00 00 00 f1 df 54 0f a8 3d 4d df e1 68 87 7b   ......T..=M..h.{
0x00000020  90 0a 0f 1b ee b0 ae 61 29 f9 b3 8b 25 49 e3 f9   .......a)...%I..
0x00000030  bd 9c 0d ec df c0 bc e0 c0 30 9c 8e 00 00 00 00   .........0......
```

Figure 55. Password of the memory dump computer

Evidence D – A disk image of a damaged mobile phone found in Alex's apartment apartment under some furniture.

Evidence D had 10 disk files in a zip file, which was extracted into the case folder to investigate. To identify the files with the largest amount of data, I used **ls -all** command which displayed the file sizes in bytes.

```
sansforensics@siftworkstation: ~/assignment/Evdd
$ ls -all
total 21199420
drwxr-xr-x 2 sansforensics sansforensics       4096 Oct 11 02:38 .
drwxr-xr-x 5 sansforensics sansforensics       4096 Oct 11 02:14 ..
-rw-r--r-- 1 sansforensics sansforensics 2641915904 Sep 14 13:13 dm-0
-rw-r--r-- 1 sansforensics sansforensics 6442450944 Sep 14 13:16 dm-1
-rw-r--r-- 1 sansforensics sansforensics 2686451712 Sep 14 13:04 vda
-rw-r--r-- 1 sansforensics sansforensics 2684354560 Sep 14 13:07 vda1
-rw-r--r-- 1 sansforensics sansforensics   69206016 Sep 14 13:07 vdb
-rw-r--r-- 1 sansforensics sansforensics 6442450944 Sep 14 13:10 vdc
-rw-r--r-- 1 sansforensics sansforensics    1048576 Sep 14 13:12 vdd
-rw-r--r-- 1 sansforensics sansforensics  102760448 Sep 14 13:12 vde
-rw-r--r-- 1 sansforensics sansforensics  100663296 Sep 14 13:12 vde1
-rw-r--r-- 1 sansforensics sansforensics  536870912 Sep 14 13:12 vdf
```

Figure. 56 Checking the file size

```
sansforensics@siftworkstation: ~/assignment/Evdd
$ mmls dm-1
Cannot determine partition type
sansforensics@siftworkstation: ~/assignment/Evdd
$ mmls vdc
Cannot determine partition type
```

Figure 57. Using mmls on the files

Upon examining the file size, I found that dm-1 and vdc have the most amount of data present in them. Next, I used the **'mmls'** command to display the layout of the partitions within the disk image. However, I was not able to determine the partition type.

To dig out more details from these files, I tried to mount the vdc file without any offset but it resulted in an error.

```
sansforensics@siftworkstation: ~/assignment/Evdd
$ sudo mount -o ro,loop,noexec,noload  vdc /mnt/windows_mount
mount: /mnt/windows_mount: wrong fs type, bad option, bad superblock on /dev/loop10, missing codepage or helper program, or other error.
```

Figure 58. Error in mounting vdc file

After getting the information, my next task was to mount the dm-1 file in mnt/windows_mount. The file was mounted successfully without **offset** and with **read-only** settings to avoid data alteration.

```
sansforensics@siftworkstation: ~/assignment/Evdd
$ sudo mount -o ro,loop,noexec,noload  dm-1 /mnt/windows_mount
```

Figure 59. Mounting dm-1 file

13. What are the non-stock applications installed on the phone?

To find the non-stock application, I examined the **app** directory and found 5 non-stock applications that were installed on the phone. The Android packages installed on the phone use reverse domain name notation that starts with '**.com'** notation to uniquely identify the applications.

Figure 60. Non-Stock Applications installed on the phone

Here, I was able to identify 4 apps that are: **'facebook'**, **'bubbleshooter'**, **'twitter'** and **'angrybirds'.** To identify the 5th application, I searched the package name (**com.tencent.mm)** on the safari which revealed it is a **'WeChat'** application

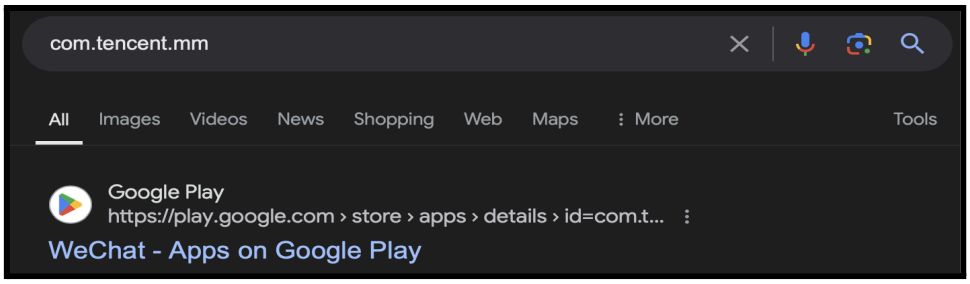


Figure 61. Websurfing for WeChat package name

14. Who is in the contacts list? What messages and calls have been sent and received by the phone?

To find the contact list, messages and call logs on the phone I used the **find** command to list all the file names that end with "**.db**". On examining the list of files, I found two files named "**contact2.db**" and "**calllogs.db**". To determine the database where messages are stored in an Android phone, I did a Google search which revealed that messages on the Android phone are saved in "**mmssms.db**". Next, I did a specific search for mmssms.db using the find command, I managed to find the mmssms.db which holds the database for the messages in the Android phone. All of the database i.e. contact2.db, callogs.db, and mmssms.db were copied and saved in the Evidence directory for further investigation.

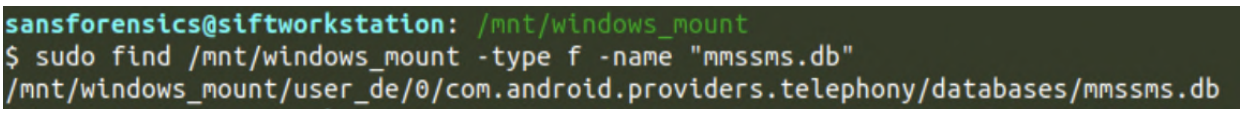


Figure 62. Finding mmssms.db file

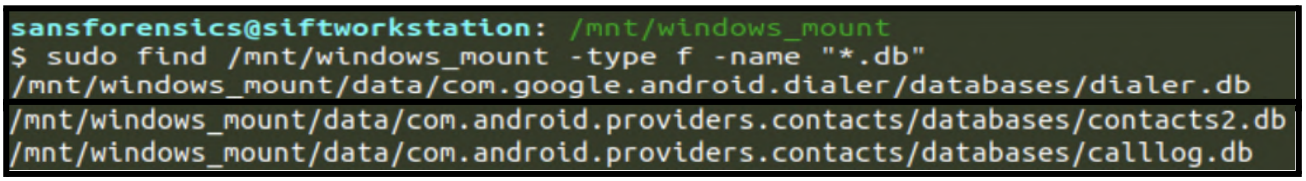


Figure 63. Finding all ".db" files present in the file dm-1

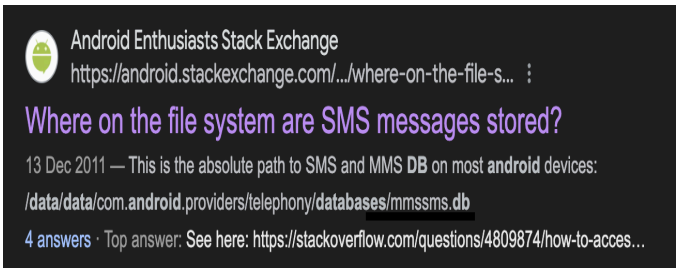


Figure 64. Web Surfing for the sms database

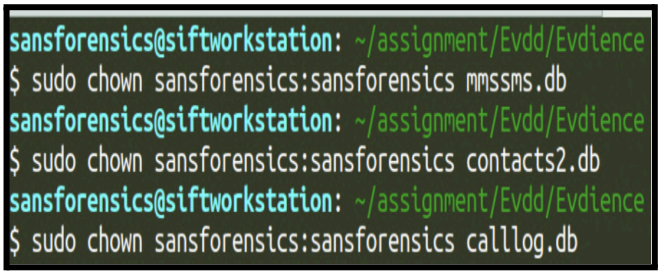


Figure 65. Taking ownership of the files

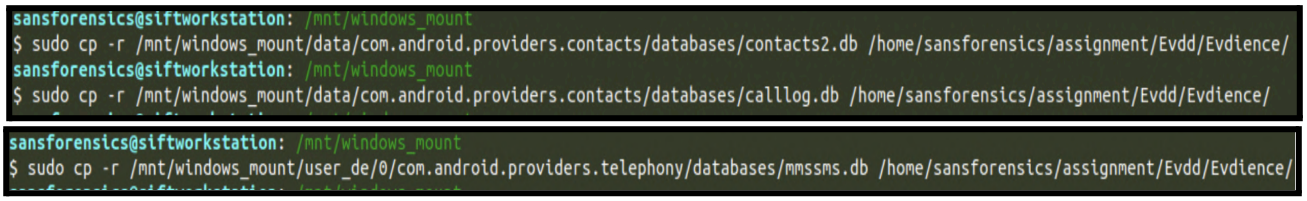


Figure 66. Copying the database file in the Evidence directory

Before I examine the database, I have to change the ownership of the file to the current user (sansforensics) because when we extract the files from a disk image the files may have ownership by other users. I can change the ownership of the extracted files by using the chown(Figure 65) command on each file. All of the files that I extracted are databases, so I open each file using sqlite.
On opening contacts.db and browsing through the data table I found 4 contacts with the following phone numbers:
Lily Parker : +61 449 857 236 , Sophie Bennett: +61 417 692 485, Mum: +61 438 179 564, Dad:+61 467 315 782

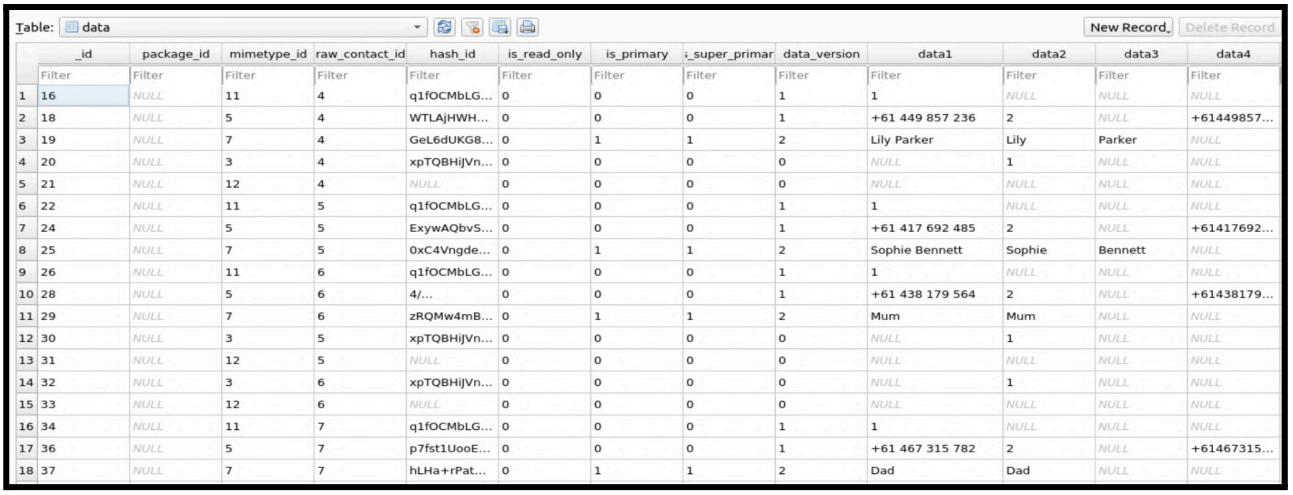


Table: data

	_id	package_id	mimetype_id	raw_contact_id	hash_id	is_read_only	is_primary	is_super_primar	data_version	data1	data2	data3	data4
1	16	NULL	11	4	q1fOCMbLG...	0	0	0	1	1	NULL	NULL	NULL
2	18	NULL	5	4	WTLAjHWH...	0	0	0	1	+61 449 857 236	2	NULL	+61449857...
3	19	NULL	7	4	GeL6dUKG8...	0	1	1	2	Lily Parker	Lily	Parker	NULL
4	20	NULL	3	4	xpTQBHijVn...	0	0	0	0	NULL	1	NULL	NULL
5	21	NULL	12	4	NULL	0	0	0	0	NULL	NULL	NULL	NULL
6	22	NULL	11	5	q1fOCMbLG...	0	0	0	1	1	NULL	NULL	NULL
7	24	NULL	5	5	ExywAQbvS...	0	0	0	1	+61 417 692 485	2	NULL	+61417692...
8	25	NULL	7	5	0xC4Vngde...	0	1	1	2	Sophie Bennett	Sophie	Bennett	NULL
9	26	NULL	11	6	q1fOCMbLG...	0	0	0	1	1	NULL	NULL	NULL
10	28	NULL	5	6	4/...	0	0	0	1	+61 438 179 564	2	NULL	+61438179...
11	29	NULL	7	6	zRQMw4mB...	0	1	1	2	Mum	Mum	NULL	NULL
12	30	NULL	3	5	xpTQBHijVn...	0	0	0	0	NULL	1	NULL	NULL
13	31	NULL	12	5	NULL	0	0	0	0	NULL	NULL	NULL	NULL
14	32	NULL	3	6	xpTQBHijVn...	0	0	0	0	NULL	1	NULL	NULL
15	33	NULL	12	6	NULL	0	0	0	0	NULL	NULL	NULL	NULL
16	34	NULL	11	7	q1fOCMbLG...	0	0	0	1	1	NULL	NULL	NULL
17	36	NULL	5	7	p7fst1UooE...	0	0	0	1	+61 467 315 782	2	NULL	+61467315...
18	37	NULL	7	7	hLHa+rPat...	0	1	1	2	Dad	Dad	NULL	NULL

Figure 67. Contacts lists

Next, I opened callogs.db using sqlite and found the following calls that have been sent and received by the phone:

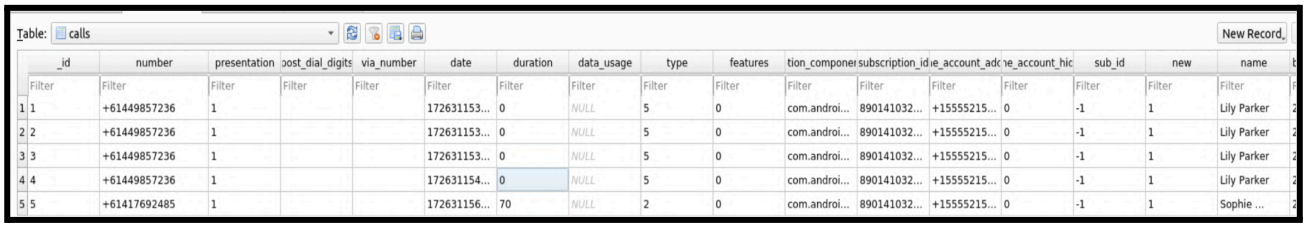


Table: calls

	_id	number	presentation	post_dial_digits	via_number	date	duration	data_usage	type	features	tion_componer	subscription_id	e_account_adc	e_account_hic	sub_id	new	name
1	1	+61449857236	1			172631153...	0	NULL	5	0	com.androi...	890141032...	+15555215...	0	-1	1	Lily Parker
2	2	+61449857236	1			172631153...	0	NULL	5	0	com.androi...	890141032...	+15555215...	0	-1	1	Lily Parker
3	3	+61449857236	1			172631153...	0	NULL	5	0	com.androi...	890141032...	+15555215...	0	-1	1	Lily Parker
4	4	+61449857236	1			172631154...	0	NULL	5	0	com.androi...	890141032...	+15555215...	0	-1	1	Lily Parker
5	5	+61417692485	1			172631156...	70	NULL	2	0	com.androi...	890141032...	+15555215...	0	-1	1	Sophie ...

Figure 68. Call logs

From the above figure, we get to know that the phone received calls from two contacts **Lily Parker** and **Sophie Bennett** (in total **5 calls**). From the duration column, we can determine that the owner of the

phone picked up Sophie Bennet's call and spoke with her for **70 seconds**, whereas declined 4 calls from Lily Parker.
Finally, I examined the mmssms.db file using the same process and found the following messages that were exchanged on the phone.

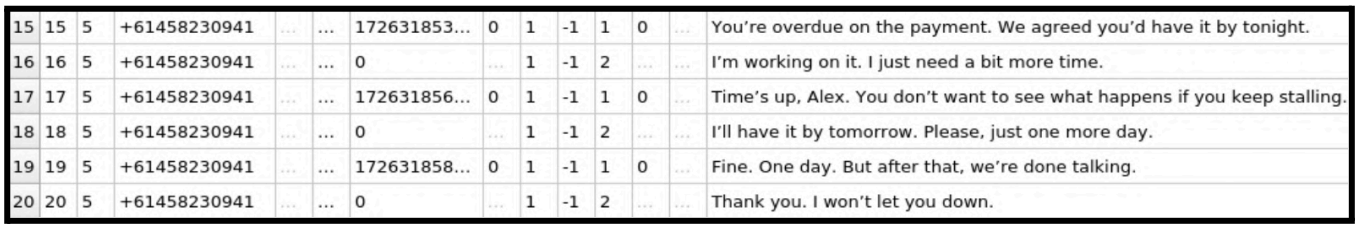
	_id	ead	address	ersc	date	date_sent	otoc	eac	tatu	ype	ath_	ubje	body
	F...	F...	Filter	F...	F...	Filter	F...	F...	F...	F...	F...	F...	Filter
1	1	3	+61449857236	4	...	172631829...	0	1	-1	1	0	...	Hey, can we talk tonight? We really need to sort this out.
2	2	3	+61449857236	...	...	0	...	1	-1	2	...	...	Not tonight, Lily. I'm dealing with a lot right now.
3	3	3	+61449857236	4	...	172631832...	0	1	-1	1	0	...	Alex, this is important. I can't keep going on like this, wondering where we stand.
4	4	3	+61449857236	...	...	0	...	1	-1	2	...	...	I know, I know. But I just can't tonight. Tomorrow?
5	5	3	+61449857236	4	...	172631835...	0	1	-1	1	0	...	Tomorrow might be too late, Alex. If you don't end this with her, I will.
6	6	3	+61449857236	...	...	0	...	1	-1	2	...	...	Lily, please don't do anything rash. I'll figure it out, I promise.
7	7	3	+61449857236	4	...	172631838...	0	1	-1	1	0	...	You better. I'm done waiting.

Figure 69. Conversation with Lily Parker

8	8	4	+61417692485	5	...	172631841...	0	1	-1	1	0	...	Alex, I'm outside your dorm. Can we talk?
9	9	4	+61417692485	...	...	0	...	1	-1	2	...	...	Sophia, now's not a good time. I'm swamped.
10	10	4	+61417692485	5	...	172631844...	0	1	-1	1	0	...	It'll only take a minute. Please, I really need to see you.
11	11	4	+61417692485	...	...	0	...	1	-1	2	...	...	Fine, come up. But I don't have long.
12	12	4	+61417692485	5	...	172631847...	0	1	-1	1	0	...	Thank you. I just... I need to know where we stand. I can't keep feeling like this.
13	13	4	+61417692485	...	...	0	...	1	-1	2	...	...	We'll talk when you get here.
14	14	4	+61417692485	5	...	172631850...	0	1	-1	1	0	...	Okay, I'm on my way.

21	21	4	+61417692485	5	...	172631868...	0	1	-1	1	0	...	I'm just outside. I can't take this anymore, Alex. You promised you'd choose me.
22	22	4	+61417692485	...	...	0	...	1	-1	2	...	...	Sophia, please, I need time. We'll talk, but not like this.
23	23	4	+61417692485	5	...	172631871...	0	1	-1	1	0	...	No, I need to know now. It's either me or her.
24	24	4	+61417692485	...	...	0	...	1	-1	2	...	...	Come up, we'll talk. But you need to calm down.
25	25	4	+61417692485	5	...	172631874...	0	1	-1	1	0	...	I'm calm, but I don't think you understand what this means for me. You can't just toss me aside.
26	26	4	+61417692485	...	...	0	...	1	-1	2	...	...	I'm not tossing you aside. Let's talk when you get here.

Figure 70. Conversation With Sophie Bennett



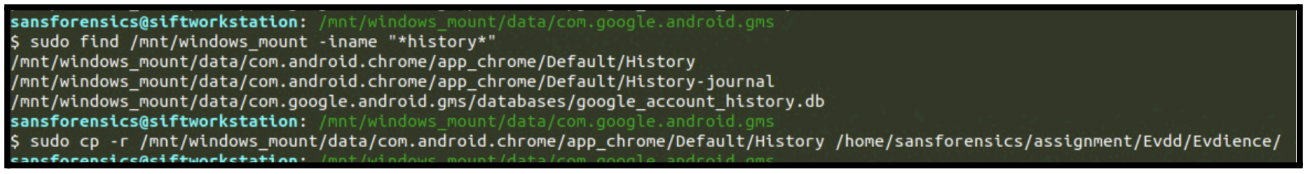
15	15	5	+61458230941	...	...	172631853...	0	1	-1	1	0	...	You're overdue on the payment. We agreed you'd have it by tonight.
16	16	5	+61458230941	...	...	0	...	1	-1	2	...	...	I'm working on it. I just need a bit more time.
17	17	5	+61458230941	...	...	172631856...	0	1	-1	1	0	...	Time's up, Alex. You don't want to see what happens if you keep stalling.
18	18	5	+61458230941	...	...	0	...	1	-1	2	...	...	I'll have it by tomorrow. Please, just one more day.
19	19	5	+61458230941	...	...	172631858...	0	1	-1	1	0	...	Fine. One day. But after that, we're done talking.
20	20	5	+61458230941	...	...	0	...	1	-1	2	...	...	Thank you. I won't let you down.

Figure 71. Conversation with the Unknown Number(+61458230941)

3 contacts were having a conversation with Alex, these are "**Sophie Bennett**", "**Lily Parker**", and **"Unknown contact Number(+61458230941)".** Out of all conversations, the Unknown Contact threatens Alex with an overdue payment and warns him about their losing patience.

15. What Internet searches has the owner of the phone made?

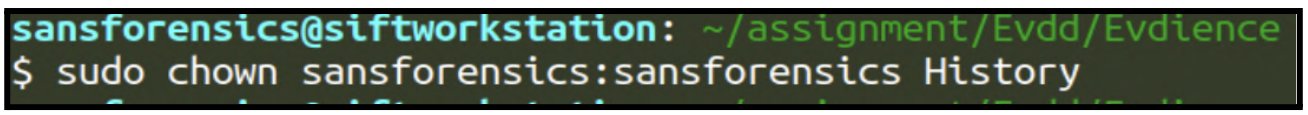
I was able to find internet searches done by the owner of the phone by using the **find** command and specifically searching for the "**history**" keyword throughout the dm-1 disk. I was able to find a history database stored in the Google Chrome directory. I then copied the file and saved it in my Evidence folder.



```
sansforensics@siftworkstation: /mnt/windows_mount/data/com.google.android.gms
$ sudo find /mnt/windows_mount -iname "*history*"
/mnt/windows_mount/data/com.android.chrome/app_chrome/Default/History
/mnt/windows_mount/data/com.android.chrome/app_chrome/Default/History-journal
/mnt/windows_mount/data/com.google.android.gms/databases/google_account_history.db
sansforensics@siftworkstation: /mnt/windows_mount/data/com.google.android.gms
$ sudo cp -r /mnt/windows_mount/data/com.android.chrome/app_chrome/Default/History /home/sansforensics/assignment/Evdd/Evdience/
```

Figure 72. Searching for the Google Chrome history

Before I start examining the history file, I have to change the ownership to sansforensics using the chown command.



```
sansforensics@siftworkstation: ~/assignment/Evdd/Evdience
$ sudo chown sansforensics:sansforensics History
```

Figure 73. Changing the ownership

I opened the history file using sqlite and found the following web searches done by the owner of the phone:

Table: urls

	id	url	title	visit_count	typed_count	last_visit_time	hidden
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	https://...	cheap burner phones near me - Google Search	1	0	133707886...	0
2	2	https://...	Prepaid Phones in Phones With Plans - Walmart.com	1	0	133707886...	0
3	3	https://...	Prepaid Phones in Phones With Plans - Walmart.com	2	0	133707886...	0
4	4	https://...	emergency loan options for students - Google Search	1	0	133707886...	0
5	5	https://...	403 Forbidden	1	0	133707886...	1
6	6	https://...	403 Forbidden	1	0	133707886...	1
7	7	https://...	how to block calls from a specific number - Google Search	1	0	133707886...	0
8	8	https://...	Block or unblock a phone number - Phone app Help	1	0	133707886...	0
9	9	https://...	Block or unblock a phone number - Phone app Help	1	0	133707886...	0
10	10	https://...	how to delete messages permanently on Android - Google Search	1	0	133707886...	0
11	11	https://...	Archive or delete messages, calls, or voicemails - Android - Google Voice Help	1	0	133707886...	0
12	12	https://...	Archive or delete messages, calls, or voicemails - Android - Google Voice Help	1	0	133707886...	0
13	13	https://...	nearest campus security office - Google Search	1	0	133707887...	0
14	14	https://...	Clery Act Policy | California Community Colleges Chancellor's Office	1	0	133707887...	0
15	15	https://...	Clery Act Policy | California Community Colleges Chancellor's Office	1	0	133707887...	0
16	16	https://...	google - Google Search	1	0	133707887...	0

Figure 74. Web searches

Thus on reviewing the above image, we get to know the following Google searches were made:

- cheap burner phones near me
- emergency loan options for students
- how to block calls from a specific number
- how to delete messages permanently on Android
- nearest campus security office

16. Is there other evidence on the phone that might indicate the role of the owner in Alex's death?

I found a few pieces of evidence that indicate a major role in Alex's mysterious death. Initially, I found 3 deleted contacts in the **contacts2.db (Figure 67)** that were present in the deleted_contact table, which was very unusual. Then I tried to dig more and found another database (using the method in **Figure 63**) named **plus_contacts.db,** which had 3 names: **Mike**, **Gus Fring,** and **Lalo Salamanca** present in the **ac_container** table. I tried to look for these names in the dm-1 using strings and grep "Lalo Salamanca" to extract all the key terms present in the dm-1 and found 3 related phone numbers. I assume these people have a major role in the death of Alex. Alex had a debt of 5000 dollars(**Figure 16**) which he was not able to pay in time, due to which he met with his tragic death.

I also found an encrypted file named **recording.aes** in the media directory, which suggests someone tried to encrypt the file to prevent accessing it. This could be a major clue related to Alex's death.

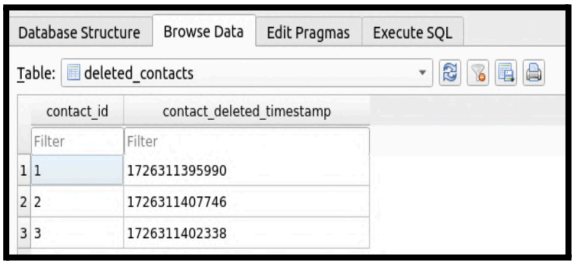


Figure 75. 3 Deleted Contacts

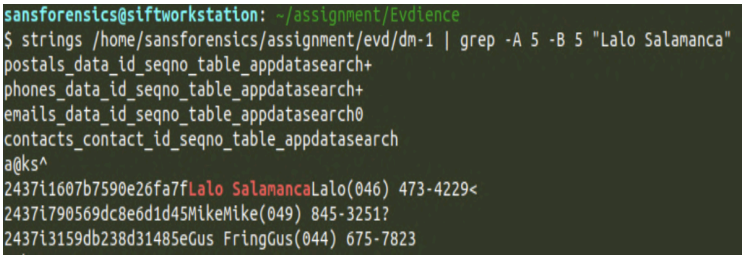


Figure 76. 3 different numbers discovered

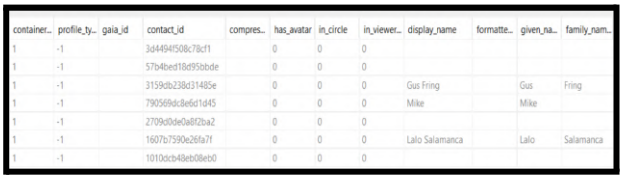


Figure 77. Names Discovered

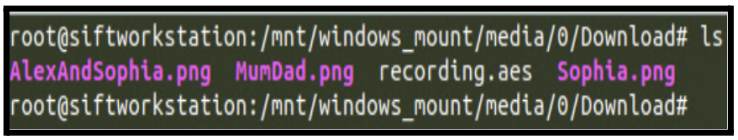


Figure 78. Discovered recording.aes

17. Conduct a timeline analysis of the pieces of evidence.

For Evidence A timeline analysis, I used **fls**(from The Sleuth Kit), a file analysis tool to extract the file paths, types and timestamps. The output from the fls is stored as **evidencebodyfile.txt** which is later again used by another tool **mactime**(from The Sleuth Kit) to read the metadata from the evidencebodyfile.txt and generate a timeline in a readable format. The timeline shows the files that were modified, accessed and changed, which helps me to analyse the sequences of events.

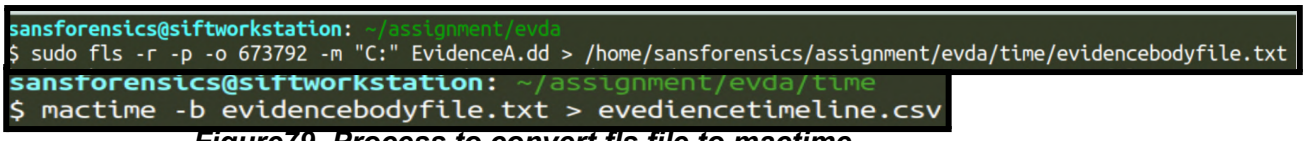


Figure79. Process to convert fls file to mactime

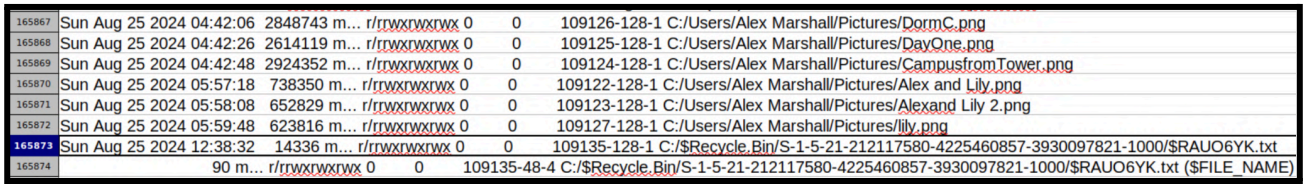


Figure 80. Timeline for the files found in Evidence A

The above image provides us with the timeline for the file/evidence that I found in Evidence A

Next, I did the timeline analysis for Evidence B, where I looked through all the conversations(**Figures 23, 24 and 25**) and noted down the time when the main events happened. I even looked for the time, for the files**(Figures 31,32,33,34, 37,38 and 39)**, I found in Evidence B.

The timeline analysis for Evidence C was done using volatility with the help of a plugin **timeliner,** which extracts the timeline information for the processes in the system, the output from the timeliner is saved as **Evidence-timeliner.body**. I again used volatility with the plugin **mftparser** to parse the master file table from the memory dump. The output from the mftparser is saved as **Evidencemftparser.body**.Next, I combined both of the outputs into one file (**Evidence-bodyfile**) using **cat** command and used mactime to generate a timeline of all processes in a readable format.

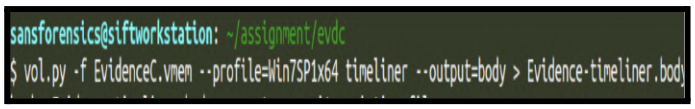


Figure 81. Using Plugin Timeliner

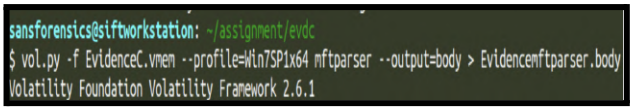


Figure 82. Using mf

Figure 83. Using mactime

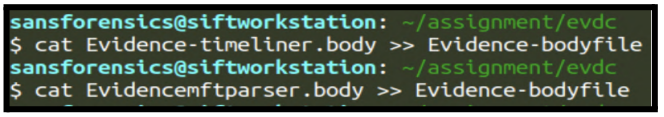


Figure 84. Combining both files into one bodyfile

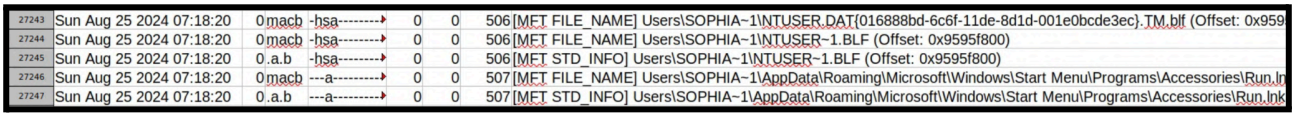


Figure 85. Timeline analysis for the processes in Evidence C

The above image provides us with the timeline analysis for the processes used by Sophia. I even gathered the start time for all the processes using plugin **pstree** from Volatility (**Figure 41**).

Finally, For Evidence D, I again used fls without an offset because mmls was unable to determine the partition type for dm-1 file(**Figure 57**). The output for the fls was saved as **fls_body.txt**, which was again used by mactime to generate the timeline in readable format.

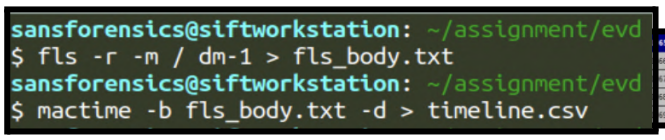


Figure 86. Generating Timeline for Evidence D

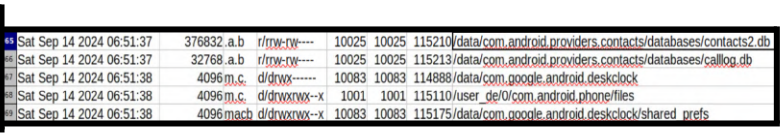


Figure 87. Evidence D timeline

After collecting all the evidence of the timeline, I created a **CSV file** and provided a detailed format of the main events I found by going through all four Evidence in ascending order TimeLineAnalysis .

Date	Time	Action
8/12/2024	11:00	C:/Program Files/7-Zip/History.txt
8/17/2024	2:36	C:/Users/Alex Marshall/Documents/journal2.txt
8/17/2024	2:38	C:/Users/Alex Marshall/Documents/Expenses_March.csv
8/17/2024	2:39	C:/Users/Alex Marshall/Documents/Debt_Tracker.csv
8/17/2024	2:31	C:/Users/Alex Marshall/Documents/journal1.txt
8/17/2024	11:54	C:/Users/Alex Marshall/Documents/Mum_Email_April.jpg
8/18/2024	3:48	C:/Users/Alex Marshall/Documents/Dad_Email_March.pdf
8/18/2024	4:48	C:/Users/Alex Marshall/Documents/InternshipOffer_Reynolds.pdf
8/18/2024	6:43	C:/$Recycle.Bin/S-1-5-21-212117580-4225460857-3930097821-1000/$RLY0J7N.zip
8/25/2024	4:42	C:/Users/Alex Marshall/Pictures/DayOne.png
8/25/2024	12:38	C:/$Recycle.Bin/S-1-5-21-212117580-4225460857-3930097821-1000/$RAUO6GYK.txt
8/25/2024	4:42	C:/Users/Alex Marshall/Pictures/DormC.png
8/25/2024	4:42	C:/Users/Alex Marshall/Pictures/CampusfromTower.png
8/25/2024	5:57	C:/Users/Alex Marshall/Pictures/Alex and Lily.png
8/25/2024	5:58	C:/Users/Alex Marshall/Pictures/Alexand Lily 2.png
8/25/2024	5:59	C:/Users/Alex Marshall/Pictures/lily.png
9/1/2024	8:41:01	Bio101 file found in network packets
9/1/2024	08:09:28 - 08:12	Chatting between Alex and Sophia
9/1/2024	07:57:11 - 08:46	Conversation between AlexM21,DormKing,PartDude,BookWarm
9/1/2024	07:57:11 - 08:02	Deal for Bio101 answers
9/1/2024	08:16:09 - 08:18	DormKing Manipulates PartyDude
9/1/2024	08:44:49 - 08:46	DormKing accuses PartDude
9/1/2024	8:43:17	SSL/TLS found in the network packets
9/1/2024	08:19:53 - 08:23	Conversation between LillyLaw and Alex
9/8/2024	07:08:41 - 12:11	Thunderbird session
9/8/2024	08:00:31 - 12:26	Firefox session
9/8/2024	12:15:06 - 12:15	Notepad session
9/8/2024	8:01:02	download thunderbird 115 - Sophia web search
9/8/2024	12:23:22	How to stop someone from leaving you - Sophia web search
9/8/2024	12:24:52	Signs someone is cheating - Sophia web search
9/8/2024	12:24:59	How to get away with self-defense - Sophia web search
9/8/2024	12:25:13	Emergency therapy services - Sophia web search

9/14/2024	12:58:49	Come up, we'll talk. But you need to calm down. (From: +61417692485)
9/14/2024	10:58:56	Call Log: Lily Parker (+61449857236)
9/14/2024	10:58:53	Call Log: Lily Parker (+61449857236)
9/14/2024	10:59:22	Call Log: Sophie Bennett (+61417692485)
9/14/2024	12:51:36	Hey, can we talk tonight? We really need to sort this out. (From: +61449857236)
9/14/2024	12:52:09	Alex, this is important. I can't keep going on like this, wondering where we stand. (From: +61449857236)
9/14/2024	12:52:25	I know, I know. But I just can't tonight. Tomorrow? (From: +61449857236)
9/14/2024	12:52:38	Tomorrow might be too late, Alex. If you don't end this with her, I will. (From: +61449857236)
9/14/2024	12:52:55	Lily, please don't do anything rash. I'll figure it out, I promise. (From: +61449857236)
9/14/2024	12:53:03	You better. I'm done waiting. (From: +61449857236)
9/14/2024	12:53:34	Alex, I'm outside your dorm. Can we talk? (From: +61417692485)
9/14/2024	12:54:01	Sophia, now's not a good time. I'm swamped. (From: +61417692485)
9/14/2024	12:54:08	It'll only take a minute. Please, I really need to see you. (From: +61417692485)
9/14/2024	12:54:22	Fine, come up. But I don't have long. (From: +61417692485)
9/14/2024	12:54:36	Thank you. I just… I need to know where we stand. I can't keep feeling like this. (From: +61417692485)
9/14/2024	12:54:55	We'll talk when you get here. (From: +61417692485)
9/14/2024	12:55:07	Okay, I'm on my way. (From: +61417692485)
9/14/2024	12:55:51	I'm working on it. I just need a bit more time. (From: +61458230941)
9/14/2024	12:56:03	Time's up, Alex. You don't want to see what happens if you keep stalling. (From: +61458230941)
9/14/2024	12:56:14	I'll have it by tomorrow. Please, just one more day. (From: +61458230941)
9/14/2024	12:56:28	Fine. One day. But after that, we're done talking. (From: +61458230941)
9/14/2024	12:56:42	Thank you. I won't let you down. (From: +61458230941)
9/14/2024	12:58:08	I'm just outside. I can't take this anymore, Alex. You promised you'd choose me. (From: +61417692485)
9/14/2024	12:58:25	Sophia, please, I need time. We'll talk, but not like this. (From: +61417692485)
9/14/2024	12:58:36	No, I need to know now. It's either me or her. (From: +61417692485)
9/14/2024	10:58:59	Call Log: Lily Parker (+61449857236)
9/14/2024	12:59:00	I'm calm, but I don't think you understand what this means for me. You can't just toss me aside. (From: +61417692485)
9/14/2024	10:59:02	Call Log: Lily Parker (+61449857236)
	11:23:00	Not tonight, Lily. I'm dealing with a lot right now. (From: +61449857236)
	11:57:02	You're overdue on the payment. We agreed you'd have it by tonight. (From: +61458230941)

Figure 88. Timeline Analysis of all the Main Events.

18. Provide a brief final analysis of the evidence and your conclusions.

Evidence A: The investigation of Evidence A shows that Alex was struggling with financial, academic and emotional stress. From the debt_tracker we get to know that he owes a total of $7450 and he is required to pay urgently a sum of $5000(taken on 2024-04-30) to an **Unknown source**. With a large loan amount and his credit card being maxed out, Alex struggles to find money to get out of his debt. An email from his father(Oliver Marshall) on March 10, 2024, adds more pressure on Alex by warning him that if he does not improve his grades, then his financial support will be cut off. This led Alex to sell exam answers to earn money which also made him feel guilty and scared of getting caught. Alex in his 'will' admits that he is in big trouble with no way out, suggesting someone is targeting him. At the end of Evidence A, we get to see Alex is stuck in a difficult situation where he tries to earn money and meet expectations while finding ways to escape his growing debt.

Evidence B: The investigation of Evidence B(The entire Evidence B took place on 1 September 2024)shows that Alex was selling Bio101 exam answers for $200 to cover up his debt (as shown in Evidence A). Here, Alex is making a deal with DormKing and PartyDude(Time:07:57:11 - 08:02:32). However, DormKing(Time:08:16:09 - 08:18:25) tries to manipulate Alex into not paying $200 for the Bio exam answers and instructing PartyDude to exploit Alex by grabbing the session key from Alex's PC. This shows that Alex's friends were primarily interested in exploiting Alex for academic gains. In the later TCP stream, I get to see Alex facing relationship problems with LilyLaw and ArtLover99(Sophia). Alex has a conversation With Sophia (Time:08:09:28 - 08:12:26) about her **meeting with Alex in his room**(Will find in Evidence D) to find out about their unclear relationship. Alex had another conversation with LilyLaw(Time: 08:19:53 - 08:23:51), where she confronted him about seeing someone else. The entire Evidence B indicates that Alex was struggling financially and facing complicated relationships. This pressure led him to make poor decisions which pushed him into further trouble.

Evidence C: The investigation of Evidence C is the memory dump of Sophia's laptop which shows the complex relationship between Sophia and Alex along with some suspicious behaviour that links to Sophia. The emails between Sophia and Alex show a close emotional/obsessive connection with the victim Alex. The content of emails with the Subject stating **Missed You Today**, **You and Me**, **Feeling Overwhelmed(Figures 52, 53 and 54)** with the victim Alex, reflects her strong feelings and frustration for Alex. Additionally, there was one email with the sender as "**Victor Ivanov**" which seems suspicious as it appeared as if Sophia might have spoofed her email and pretended to be someone else to give a "**friendly warning**" to **gG Professor Kane** in the possibility of manipulating him**.** Sophia's web searches also raised some concerns as she looked for some topics such as "**How to stop someone from leaving you**" and "**Sign someone is cheating**"(Figure 45). These searches indicate that she was insecure and was possibly planning something harmful due to her emotional state. In the process of looking through the dumped memory, I was able to find two journals named "**journal1** and **journal2**" which highlight her obsession with Alex. In this journal (**Figure 90**), Sophia describes her jealousy of Alex's relationship with Lily and reveals her fear of losing Alex. However, we can see a strong attachment from Sophia towards Alex as her laptop password is "**Alexisbeautiful**". The entire Evidence C indicates that Sophia was deeply connected to Alex but may have been struggling with jealousy and insecurity.

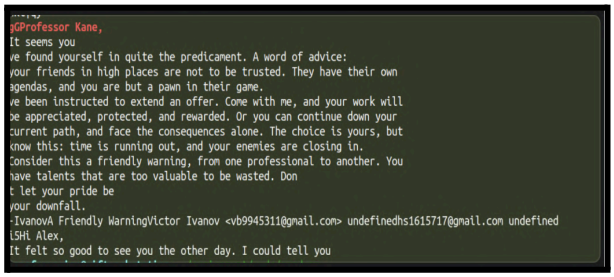


Figure 89. Spoofed email

Figure 90. Journal Entry

Evidence D: The investigation of Evidence D is the disk image of a damaged phone that was found in Alex's apartment. There were 10 disk files out of all I was able to mount dm-1 and examine it for evidence. From this disk file, I was able to extract important data which includes contacts, call-logs, messages and web searches. In contact2.db file(Figure 67) I found four contact numbers. However, I found **3 deleted contacts** that were present in the deleted_contact(Figure 75) table, which was very unusual. This indicates that Alex was trying to **cut ties** with some individuals before he met with his tragic death. These deleted contacts raise suspicion about the people Alex was trying to distance himself from. The calllogs.db file(Figure 68) shows Alex received calls from Sophia and Lilly, out of all he spoke with Sophie for 70 seconds. In the mmssms.db file (Figures 69,70 and 71), I found a conversation of Alex with Lily and Sophia. Here, Lily was pressuring Alex to decide between her and Sophie, while Sophie wanted to speak with Alex and was **waiting outside Alex's dorm.** This confirms the reason we found **Sophia's laptop in Alex's bedroom**. Most importantly, Alex had a conversation with an **Unknown contact** number that threatens Alex with an overdue payment and warns him about their losing patience. Next, Alex's web searches raised concerns as he searched for the topics "Cheap Burner phone", "Quick Student loan", "Campus Security Office number' and "How to block specific phone numbers". These searches show that Alex was afraid, trying to hide his identity, find money and possibly seek protection from the campus security. Additionally, in the plus_contacts.db(Figure 77) I found 3 names: **Mike**, **Gus Fring,** and **Lalo Salamanca** which likely related to the three numbers that Alex had deleted. This suggests that these individuals were connected to the **$5000 debt Alex owned**(Figure 16) and it appears that Alex was attempting to remove their details to protect himself. Lastly, I found an encrypted file named recording.aes which indicates that Alex was trying

to hide or protect something crucial linked to these threats. The entire Evidence D indicates that Alex was under pressure from dangerous individuals and he was trying to protect himself. Alex's attempts to delete contacts, encrypt files and search for ways to protect himself indicate that he is fearful of what might happen to him, which ultimately led to his traffic death.

Conclusion: The investigation reveals that Alex was overwhelmed by financial, academic and emotional stress. **Evidence A** shows that Alex had owed **$5000** and was desperately trying to earn money. In **Evidence B,** Alex attempted to earn money by selling exam answers to pay off his debt but his plan was manipulated by DormKing which made it harder for him to succeed. **Evidence C**, confirms that Sophia was present in Alex's room (from messages) before the incident. Finally, **Evidence D** indicates that dangerous individuals: **Mike**, **Gus Fring,** and **Lalo Salamanca** came to collect the unpaid debt. Since Sophia was in Alex's dorm room and the room was likely open, this allowed the three individuals to **enter the room without force.** With **no money to offer**, Alex met with his **tragic death.**

19. Provide advice to your forensics team about the identification and collection methods for each specific evidence item.

There are 5 critical steps in digital forensics: Identification, Collection, Examination, Analysis and Presentation. Before the start of the collection phase, the best practice is to ensure the integrity of the evidence by write-protecting it. The copy of the **write-protected** data is then used for analysis instead of the original data so that the original data stays safe. The final step in the collection process is to create a **chain of custody** to keep the record of who had access to the evidence and ensure its integrity during a court hearing.

Evidence A – A disk image of a desktop computer found in Alex's dorm room.

Identification Process:

- The disk image found on the desktop computer in Alex's room needs to be carefully examined as it may contain a large amount of data files, logs, and application records. The forensic team can use tools like FTK imager to create a bit-by-bit image of the hard drive.
- After the disk file has been secured, the forensic team is required to convert all the disk files into a single dd file for thorough analysis.

Collection Process:

- During the collection process, the forensic team needs to ensure the evidence is write-protected to prevent data alteration during the analysis. Next, we need to use a cryptographic hash such as MD5/SHA256 to verify the integrity of the evidence before the acquisition. Forensic teams can use tools like RegRipper to extract valuable information from the dd file

Evidence B – Network capture of the dormitory network.

Identification process:

- The forensic team is required to identify the incidents based on the network indicator. A network capture contains records of all the network communication. The forensic team can use tools like Wireshark to extract IP addresses, protocols and data transferred during the communication. During the process, the forensic team must look for the pattern related to the case.

Collection Process:

- The network data must be saved in .pcap format and ensure that it is securely stored. It is important to guarantee that the capturing of the data is conducted legally and with proper authorization.

Evidence C – A memory dump of a personal laptop found in Alex's bedroom.

Identification process:

- The first step for the forensic team is to identify the state of the system and to identify whether the memory is dead or alive. The memory acquisition must be done with the system still running to capture the volatile data such as running processes and open files. Once the forensic team identifies the system as being active, they can use Volatilty with the plugins pstree and pslist to identify the running process.

Collection Process:

- FTK can be used to collect volatile memory. These tools create bit stream images of the memory, which ensures all the data is copied completely and accurately. During this process, the investigating team may use a **write blocker,** which is a tool designed specifically to protect the suspected driver from accidental tampering.

Evidence D – A disk image of a damaged mobile phone found in Alex's apartment apartment under some furniture.

Identification process:

- Mobile phones contain a vast amount of information such as contacts, messages and non-stock application data. The forensic team must identify the data using forensic tools such as Cellebrite to extract information from the damaged device.

Collection process:

- The mobile phone should be handled with care due to its fragile state. The forensic team can use a write blocker to avoid any accidental tampering. The disk image should be made before any repair attempts or power-on to avoid any potential data loss.
- If the mobile phone uses a SIM card, data from the SIM can be retrieved and analyzed, such as phone numbers, IMSI, and text messages.

In the end, the forensic team must create a timeline analysis for all the artifact collected and reconstruct the event timeline for a better understanding of the order of the events.

Task2 Memory Forensics

Introduction - Overview of Memory Forensics

Memory Forensics is important in the field of Digital Forensics which deals with the analysis of memory to retrieve critical information for criminal investigation. Memory forensics focuses on the actual process/programs that were running on a device at the time the memory dump was captured, unlike hard disk where the disk can be cloned and every file on the disk can be restored and analysed. Memory forensics mostly focuses on analysing volatile memory(RAM), where an investigator needs to be careful in handling the volatile memory as a running device will lose its memory when turned off. The volatile data is not written on the hard drive and is constantly changing in memory, thus turning off the device destroys the valuable evidence (Fox, 2021).

Memory forensics is done by taking a snapshot of the computer memory data from a specific instant of time with the help of specialised forensic tools. This snapshot is called a memory dump, a memory dump can contain valuable insights about the system's state before an incident, such as a crash or security breach. Memory dumps contain RAM data that can be used to determine the cause of an incident as well as other critical information. The insights from a memory dump can include system activity, open network connection and last modified processes. In many cases, an investigator can find chat messages and internet search history from a memory dump (Lord, 2020). For example in the case of Alex's death, the data retrieved from Sophia’s laptop memory dump includes her email conversations with Alex and internet searches. Memory forensics is useful in those events where a suspect deletes files from the hard disk but the residues of these may still be present on the RAM. The investigator then can use specialised tools to create timelines for the process found in the memory to reconstruct the events.

These days cyber-attacks have become more advanced and complex which makes them hard to detect with the existing system. Security attacks like scripts can be written directly inside of physical memory, thus making memory forensics important in the field of digital investigations.

Evidence Acquisition

In memory forensics acquiring evidence is the most critical part which must be handled with care to maintain the integrity of the volatile data. Volatile data like random access memory are very sensitive because once the system's power is turned off, the data in the memory can be lost or altered. For example in Alex’s death, the investigating team was required to capture volatile data from Sophia’s laptop, the process to capture the memory dump needed to be done carefully so that the team could retrieve crucial evidence like email conversations and internet searches. Before the start of any investigation, the best practice is to ensure the integrity of the evidence by write-protecting it. The copy of the **write-protected** data is then used for analysis instead of the original data so that the original data stays safe.

The first step for any memory acquisition involves **determining the state of the system**, whether the memory is dead or alive. The dead acquisition is the collection of data when the system has been shut down. The dead acquisition is only helpful when the investigation involves a sensitive system that must not be disturbed. Whereas, in the live acquisition, the data is collected during the running phase of the memory. This method of acquisition can provide information on the running

process, network connections and current state of the system (*View of Volatile Data Acquisition and Analysis by Using Memory Forensics Techniques*, 2024).

After determining the state of the system, memory acquisition is required to **copy the content of a volatile memory to another device** to maintain its integrity. Various tools could be used to create copies of this volatile memory such as FTK, and SMART. These tools create bit stream images of the memory which ensures all the data is copied completely and accurately. If the transfer of volatile memory is not possible during the process, then the bitstream disk-to-disk file is used with the help of SafeBack and Norton Ghost tools. (EC-Council, 2022). During this process, the investigating team may use a **write blocker** which is a hardware or software tool that is designed specifically to protect the suspected drive from any accidental tampering. Write blockers are very important for maintaining the data integrity and during the chain of custody to prove that data was not tampered with during the examination of the drive (RedTeam Cybersecurity Labs, 2023).

The next step for memory acquisition is the use of **cryptographic hashing**. Algorithms such as MD5 or SHA1, ensure the signed data remains exactly the same. Most modern forensic tools automatically sign the data using one of the algorithms mentioned. During the court hearing, the integrity of these files is matched with the original file before and after analysis to guarantee that the data was not tampered with during the court hearing.

The final step in evidence acquisition is to create a **chain of custody** to keep the record of who had access to the evidence and ensure its integrity during a court hearing. Any individual not involved in the case should not have access to the evidence as this would contaminate the chain of custody of evidence. Therefore chain of custody involves comprehensive documentation for the entire acquisition process that includes time, date, tools used, and individuals involved, to maintain the legitimacy of the evidence.

By following all the steps mentioned in the evidence acquisition phase, the forensic investigator can ensure the integrity and legitimacy of the evidence.

Forensic Analysis

Memory forensics analysis involves examination of the volatile data to extract valuable information about the system activity, running process, and network connection. The forensic analysis follows through evidence acquisition and focuses on reconstructing the event timeline to identify security breaches and crucial data from the volatile memory. The main goal of forensic analysis is to extract accurate results by utilising specialised tools like volatility

Some of the algorithms, techniques and methods used by forensic investigators for memory analysis (Batur, 2024):

Process analysis: In memory forensics, the digital investigators extract information about the running process to determine the state of a system. For example, In the case of Alex’s death, a forensic investigator will check for the running process on Sophia's laptop to construct a timeline of events. These processes can help us to identify suspicious activity that was running during the time of the incident. Investigators can use plugins like pslist, pstree, psxview and psscan with volatility to detect any hidden process. Pstree helps to visualise and find all the running processes at the time of the memory dump and arrange the running process in a tree-like structure.

VAD Tree analysis: The Windows memory manager uses the Virtual Address Descriptor tree to describe memory ranges which is used by the process when they are allocated. The VAD tree is then created by the memory manager when a memory uses a VirtualAlloc to allocate the memory. The analysis of the VAD tree is useful for finding hidden code or evidence that has been bypassed by the traditional process listing (Process analysis). Investigators can use tools like vadinfo or vadtree to reveal any suspicious process that might occur in the system such as hidden communication channels.

Memory Dumping: Memory dump of volatile data is processed by creating complete snapshots of the system's RAM at a specific point in time for offline analysis. For example, by dumping the memory from Sophia's laptop, we were able to analyze the running process and extract in-memory data like email conversations and browsing history. Investigators can use plugins such as memdump with volatility to analyze the structure of the memory. After the memory has been dumped, the case investigator can use the Bulk extractor tool, to scan the memory dump for strings, keywords or artifacts like credit card numbers. Additionally, to extract the information contained in the system's registry, the case investigator can use a plugin dumpregisty with volatility to dump all the hive keys captured by the system's registry. These hive keys can be analyzed by the tool RegRipper to extract the system configuration and user activity during the time of the incident.
Some of the challenges faced during the forensic analysis:

Volatile Nature of the Ram: The volatile nature of the Ram makes memory forensics difficult. The data in memory is retained only when the power is switched on and it is constantly changing. Extracting a snapshot of the memory requires precise timing because running processes and system load can impact the result.

Memory Dump Size: Dumping the large amount of memory present in the most modern system with a large capacity of RAM is another challenge faced by digital investigators. Analysing this large amount of data for files, user activity is resource intensive. Investigators can use tools like Bulk Extractor and RegRipper to parse the data but these tools require a significant amount of computational resources to analyse the dumped memory.

In memory forensics, investigators can use tools like volatility and reg-ripper to maintain accuracy while performing data collection and minimise data loss. Investigators should focus on memory analysis by prioritising the integrity of the volatile data and using advanced tools to examine the hidden process and memory structures.

Forensic Tools

Memory forensics is an art where an investigator uses specialised tools to analyse a computer's memory dump to extract valuable information that can be used in court hearings. Memory forensics has multiple include diagnosing malware infections, figuring out the primary cause of system crashes, and retrieving lost data.
In the world of memory forensics, some of the tools used by investigators are :

Volatility: Volatility is an open-source memory forensic framework that is widely used by forensics investigators and incident responders. Volatility is capable of extracting digital artifacts from volatile memory samples and provides insights into running processes, network activity and malware artifacts. Volatility framework offers a wide range of plugins which are run through a command line

interface. Volatility can be used in either Windows, Linux and MAC. Some of the plugins that I used in volatility are:

- **pstree**: Pstree helped me to find all the running processes at the time of the memory dump and arranges the running process in a tree-like structure. Example (vol.py -f EvidenceC.vmem --profile=Win7SP1x64 pstree)
- **imageinfo**: imageinfo helped me to extract the list of **system profiles** found in the memory dump. Example (vol.py -f EvidenceC.vmem imageinfo)
- **lsadump**: The lsadump plugins helped me to extract sensitive information about the user authentication and security policies. Example (vol.py -f EvidenceC.vmem --profile=Win7SP1x64 lsadump)

The main advantage of using volatility is that it supports analysis capabilities in multiple operating systems. However, the disadvantage is the long learning process and volatility lack of user graphical interface (Xffccdd, 2022).

Rekall: Rekall is an open-source memory forensic framework that is similar to volatility and it can be used with plugins and scripts. Rekall was developed by the Google Security team and it is compatible with various memory capture formats which allowed it to be used in different forensic cases. The advantage of using Rekall is that it is scalable and provides a smooth installation process with more in-built interfaces. However, Rekall does not offer the same plugin's extendibility as offered by Volatility for specific forensics tasks (Xffccdd, 2022).

FTK Imager: The FTK imager is a forensic analysis tool used in the process of memory acquisition. FTK imager helps to acquire forensic images from hard drives, storage media and volatile memory. Most digital investigators use FTK imager to extract data from the memory dump. The main advantage of using FTK is its simple graphical user interface which makes it user-friendly and widely adopted in the field of digital forensics. However, the FTK imager lacks advanced memory analysis features and it is primarily used for memory acquisition rather than in-depth forensic analysis (Xffccdd, 2022).

In memory forensics, combining different tools is the most effective strategy, but Volatility remains a leader due to its flexibility and powerful plugins, while the Rekall provides an easier interface for those who are new in the field of digital forensics. By using all the tools mentioned, the case investigator can ensure an accurate investigation.

Conclusion

In conclusion, Memory forensics is one of the most important fields in digital forensics, where case investigator officers need to use a wide variety of tools to capture the data and perform an in-depth analysis of the volatile memory. In the case of Alex’s mysterious death, we use tools like Volatility to analyse Sophia’s laptop memory dump which provides us with useful information that includes running processes, email conversations, and web searches. While there are many challenges in the field of memory forensics, it will remain an essential method for solving cases and ensuring the security of digital systems in the future.

References

- Fox, N. (2021, July 26). *Memory Forensics for Incident Response*. Www.varonis.com. https://www.varonis.com/blog/memory-forensics
- Lord, N. (2020, September 29). *What Are Memory Forensics? A Definition of Memory Forensics*. Digital Guardian. https://www.digitalguardian.com/blog/what-are-memory-forensics-definition-memory-forensics
- *View of Volatile Data Acquisition and Analysis by Using Memory Forensics Techniques*. (2024). Lgu.edu.pk. http://ijeci.lgu.edu.pk/index.php/ijeci/article/view/169/129
- EC-Council. (2022, March 11). *How to Handle Data Acquisition in Digital Forensics*. Cybersecurity Exchange. https://www.eccouncil.org/cybersecurity-exchange/computer-forensics/data-acquisition-digital-forensics/
- *Shibboleth Authentication Request*. (2024). Griffith.edu.au. https://ieeexplore-ieee-org.libraryproxy.griffith.edu.au/stamp/stamp.jsp?tp=&arnumber=9597136
- RedTeam Cybersecurity Labs. (2023, September 23). Understanding write blockers: Safeguarding digital evidence in Forensic. RedTeam Cybersecurity Labs. https://theredteamlabs.com/understanding-write-blockers/
- Batur, A. (2024, October 9). Investigating memory forensic -Processes, DLLs, consoles, process memory and networking. *Medium*. https://alpbatursahin.medium.com/investigating-memory-forensic-processes-dlls-consoles-process-memory-and-networking-7277689a09b7
- Xffccdd. (2022, December 15). Memory Forensics Tools - 0XFFCCDD - medium. *Medium*. https://medium.com/@cloud_tips/memory-forensics-tools-123e32387adb
- *User agents*. (2024). https://user-agents.net/