% 19/12/2024 11:07:38 %

- The line manager gave me the image copy of the USB key in a zip file, which they received from a client, X2024, for digital investigation.

- The zip file also contained an MD5 hash value of the image file for an integrity check; it was taken when they found the evidence.

Exhibit	**Description**
#DFT-USB-A-315	Forensic image of a USB stick [X2024.dd]
#DFT-MD5-A-315	Exhibit containing MD5 hash values related to USB stick. [X2024.dd.md5]

Case reference number: 6893549
Client: X2024

The line manager has the following questions for you:

i. Is there evidence on the USB stick (#DFT-USB-A-315) that suggests it contains or previously contained bamboo images? If so, how many unique images?

ii. Is there evidence showing how any images that are present were obtained?

iii. Are there indicators suggesting whether the suspect acted alone or as part of an organized group?

iv. As bamboo images are not illegal outside of England, is there evidence of potential illegal imports of such images? If evidence exists, for how long have such imports occurred?

v. Is there evidence of another party having accessed the USB stick?

Initial examination:

% 19/12/2024 12:10:15 %

- Using Tsurugi VM (2024.1) and a terminal (Kernal Version 6.9.3) with the bash shell (GNU bash version 5.1.16).

- First, I opened the default Firefox browser in Tsurugi Linux and uploaded the USB image file to the VirusTotal website to check if it contains any malware or virus that will affect my workstation.

  `https://www.virustotal.com/gui/home/upload`

- To do the integrity check, I used Jupyter Lab (4.2.5) and the Python (Python 3.13.1) code given in Lab 7 for further analysis.

- First I opened the terminal and navigated to the USB image file directory to run the `Jupyter Lab` command. Using the program, I extracted the file metadata.

```
import os
import time

def file_metadata(file_path):
    file_info = os.stat(file_path)
    return {
        'File Size': file_info.st_size,
        'Creation Time': time.ctime(file_info.st_ctime),
```

```
        'Modification Time': time.ctime(file_info.st_mtime)
    }

metadata = file_metadata('X2024.dd')
print("Metadata:", metadata)
```

- Using this python program I extracted the file metadata of image file.

```
Metadata: {'File Size': 131072000, 'Creation Time': 'Sun Dec 15 12:57:14 2024', 'Modification Time': 'Sat Aug 27 05:08:13 2022'}
```

% 19/12/2024 12:25:56 %

- I compared both the MD5 hashes that I extracted from the X2024.dd image using a Python program and the hash I received from the line manager in zip file.

```
import hashlib

# Function to generate hash of a file
def hash_file(file_path, hash_algorithm='md5'):
    hash_obj = hashlib.new(hash_algorithm)
    with open(file_path, 'rb') as file:
        while chunk := file.read(4096):
            hash_obj.update(chunk)
    return hash_obj.hexdigest()

# Calculate and display MD5 hash of a file
file_hash = hash_file('X2024.dd')
print("File MD5 Hash:", file_hash)
```

```
File MD5 Hash: 11499b323e0e49db5b0c7a9537bc81ae   11499b323e0e49db5b0c7a9537bc81ae  X2024.dd
```

- Also extracted the SHA256 hash for future investigation by using same programm above and bye changing this line `hash_algorithm='sha256'`

```
File SHA-256 Hash: 8056e2ea51d0505d87b9e3a033cba6f43b28bae71de31c541947614245ece558
```

Primary investigation:

% 19/12/2024 13:52:44 %

- For a primary investigation of the image file and its directory tree, I mounted the X2024.dd file in the terminal of Tsurugi Linux (Superuser `sudo su`) using the following commands:

```
fdisk -l X2024.dd                        # list the partition in disk
sudo mkdir /mnt/diskimage    # Create the mount directory if it doesn't
sudo mount -o ro X2024.dd /mnt/diskimage  #Mount in read-only mode
ls /mnt/diskimage            # check directory
sudo umount /mnt/diskimage               # unmount the disk image
```

- Now I changed my directory to /mnt/diskimages and checked the content of the disk X2024.dd (using command “`pcmanfm`” command in terminal).

- disk file is having 2 folders DCP and uskz4iry. default, each contained 19 (images) and 52 (files and folders).

```
root@tsurugi: /mnt/diskimage (as superuser)
root@tsurugi: /mnt/diskimage 79x27
root@tsurugi:/mnt/diskimage# ls -al
total 26
drwxr-xr-x  4 root root 16384 Jan  1  1970 .
drwxr-xr-x  1 root root    60 Dec 17 02:29 ..
drwxr-xr-x  2 root root  4096 Jun  1  2012 DCP
drwxr-xr-x 15 root root  6144 Jul 22  2022 uskz4iry.default
root@tsurugi:/mnt/diskimage#
```

- In this part, I am showing the files and subfolders of the both disk image folders that were shown when I mounted the disk image.

```
root@tsurugi: /mnt/diskimage/DCP (as superuser)
root@tsurugi: /mnt/diskimage/DCP 79x27
root@tsurugi:/mnt/diskimage/DCP# ls -al
total 8640
drwxr-xr-x 2 root root    4096 Jun  1  2012 .
drwxr-xr-x 4 root root   16384 Jan  1  1970 ..
-rwxr-xr-x 1 root root   61038 Jul 13  2016 02010026.jpg
-rwxr-xr-x 1 root root   60716 Jul 13  2016 09260002.jpg
-rwxr-xr-x 1 root root 1210966 Jul 13  2016 100_0001.JPG
-rwxr-xr-x 1 root root 1121471 Jul 13  2016 100_0393.JPG
-rwxr-xr-x 1 root root  635577 May 12  2019 adapt.png
-rwxr-xr-x 1 root root  252545 May 12  2019 alter.png
-rwxr-xr-x 1 root root  775342 May 12  2019 bench.png
-rwxr-xr-x 1 root root  659016 May 12  2019 dozen.png
-rwxr-xr-x 1 root root  436038 May 12  2019 force.png
-rwxr-xr-x 1 root root  384799 May 12  2019 fresh.png
-rwxr-xr-x 1 root root  232799 May 12  2019 glass.png
-rwxr-xr-x 1 root root  424374 May 12  2019 large.png
-rwxr-xr-x 1 root root  633346 May 12  2019 least.png
-rwxr-xr-x 1 root root  492380 May 12  2019 newly.png
-rwxr-xr-x 1 root root  481187 May 12  2019 print.png
-rwxr-xr-x 1 root root  118479 May 12  2019 think.png
-rwxr-xr-x 1 root root  430188 May 12  2019 vital.png
-rwxr-xr-x 1 root root  237528 May 12  2019 watch.png
-rwxr-xr-x 1 root root  163203 May 12  2019 zeros.png
root@tsurugi:/mnt/diskimage/DCP#
```

```
root@tsurugi: /mnt/diskimage/uskz4iry.default (as superuser)
root@tsurugi: /mnt/diskimage/uskz4iry.default 86x35
root@tsurugi:/mnt/diskimage/uskz4iry.default# ls -al
total 12680
drwxr-xr-x 15 root root    6144 Jul 22  2022 .
drwxr-xr-x  4 root root   16384 Jan  1  1970 ..
-rwxr-xr-x  1 root root    6148 Jul 22  2022 .DS_Store
-rwxr-xr-x  1 root root       0 Jul 22  2022 .parentlock
-rwxr-xr-x  1 root root  109310 Jul 22  2022 AlternateServices.bin
-rwxr-xr-x  1 root root   30405 Jul 22  2022 ExperimentStoreData.json
-rwxr-xr-x  1 root root    7150 Jul 22  2022 SiteSecurityServiceState.bin
-rwxr-xr-x  1 root root    4510 Jul 22  2022 addonStartup.json.lz4
-rwxr-xr-x  1 root root      24 Jul 22  2022 addons.json
-rwxr-xr-x  1 root root       0 Jul 22  2022 autofill-profiles.json
drwxr-xr-x  2 root root    2048 Jul 22  2022 bookmarkbackups
-rwxr-xr-x  1 root root   98304 Jul 22  2022 bounce-tracking-protection.sqlite
-rwxr-xr-x  1 root root     221 Jul 22  2022 broadcast-listeners.json
-rwxr-xr-x  1 root root  229376 Jul 22  2022 cert9.db
-rwxr-xr-x  1 root root     225 Jul 22  2022 cert_override.txt
-rwxr-xr-x  1 root root     221 Jul 22  2022 compatibility.ini
-rwxr-xr-x  1 root root     688 Jul 22  2022 containers.json
-rwxr-xr-x  1 root root  262144 Jul 22  2022 content-prefs.sqlite
-rwxr-xr-x  1 root root  524288 Jul 22  2022 cookies.sqlite
drwxr-xr-x  3 root root    2048 Jul 22  2022 crashes
drwxr-xr-x  4 root root    2048 Jul 22  2022 datareporting
-rwxr-xr-x  1 root root   98304 Jul 22  2022 domain_to_categories.sqlite
-rwxr-xr-x  1 root root       2 Jul 22  2022 enumerate_devices.txt
-rwxr-xr-x  1 root root     799 Jul 22  2022 extension-preferences.json
drwxr-xr-x  2 root root    2048 Jul 22  2022 extension-store
drwxr-xr-x  2 root root    2048 Jul 22  2022 extension-store-menus
-rwxr-xr-x  1 root root   22993 Jul 22  2022 extensions.json
-rwxr-xr-x  1 root root 5242880 Jul 22  2022 favicons.sqlite
-rwxr-xr-x  1 root root  262144 Jul 22  2022 formhistory.sqlite
drwxr-xr-x  3 root root    2048 Jul 22  2022 gmp-gmpopenh264
drwxr-xr-x  3 root root    2048 Jul 22  2022 gmp-widevinecdm
```

```
drwxr-xr-x  3 root root    2048 Jul 22  2022 gmp-gmpopenh264
drwxr-xr-x  3 root root    2048 Jul 22  2022 gmp-widevinecdm
-rwxr-xr-x  1 root root     380 Jul 22  2022 handlers.json
-rwxr-xr-x  1 root root  294912 Jul 22  2022 key4.db
-rwxr-xr-x  1 root root    2701 Jul 22  2022 logins-backup.json
-rwxr-xr-x  1 root root    3388 Jul 22  2022 logins.json
drwxr-xr-x  2 root root    2048 Jul 22  2022 minidumps
-rwxr-xr-x  1 root root   98304 Jul 22  2022 permissions.sqlite
-rwxr-xr-x  1 root root     491 Jul 22  2022 pkcs11.txt
-rwxr-xr-x  1 root root 5242880 Jul 22  2022 places.sqlite
-rwxr-xr-x  1 root root   22950 Jul 22  2022 prefs.js
-rwxr-xr-x  1 root root   65536 Jul 22  2022 protections.sqlite
drwxr-xr-x  2 root root    2048 Jul 22  2022 saved-telemetry-pings
-rwxr-xr-x  1 root root     325 Jul 22  2022 search.json.mozlz4
drwxr-xr-x  2 root root    2048 Jul 22  2022 security_state
-rwxr-xr-x  1 root root     915 Jul 22  2022 serviceworker.txt
-rwxr-xr-x  1 root root     288 Jul 22  2022 sessionCheckpoints.json
drwxr-xr-x  2 root root    2048 Jul 22  2022 sessionstore-backups
-rwxr-xr-x  1 root root  149606 Jul 22  2022 sessionstore.jsonlz4
drwxr-xr-x  2 root root    2048 Jul 22  2022 settings
-rwxr-xr-x  1 root root      18 Jul 22  2022 shield-preference-experiments.json
drwxr-xr-x  5 root root    2048 Jul 22  2022 storage
-rwxr-xr-x  1 root root    7168 Jul 22  2022 storage.sqlite
-rwxr-xr-x  1 root root    5737 Jul 22  2022 targeting.snapshot.json
-rwxr-xr-x  1 root root      72 Jul 22  2022 times.json
-rwxr-xr-x  1 root root   98304 Jul 22  2022 webappsstore.sqlite
-rwxr-xr-x  1 root root     485 Jul 22  2022 xulstore.json
root@tsurugi:/mnt/diskimage/uskz4iry.default#
```

% 19/12/2024 15:51:56 %

- In this part, I place a copy of the dd disk image and its hash in a separate folder and name it the same as the USB recovered.

  `/home/user/#DFT-USB-A-315`

- After that I use foremost to check if it ables to carve files from the disk using command given below in Terminal:

  `foremost -i X2024.dd -o output_imge`

- Total 64 files carved from the disk using foremost commands. Out of which there are 8 jpg, 1 htm, 3 zip, 48 png, and 4 pdf. The number is the same when checking the audit file within the output_image folder.

% 19/12/2024 19:05:42 %

- Using Autopsy version 4.21.0 in Tsurugi Linux.

- Opening a new case file with **Case Name**: 6893549-X2024, **Case Number**: 6893549 (here X2024 is the Client name) and examiner details.

- Choosing the image file (X2024.dd) from path given above and adding the hash value form file (X2024.dd.md5).

- In the Ingust modules, I deselected all the 8 option below starting form Android Analyzer (aLEAPP) and Extension Mismatch detection to check all file type from global setting in right hand side.

Examination of #DFT-USB-A-315

% 19/12/2024 21:15:43 %

- Exploring image files by their extension.
- I selected the file type option from the left sidebar in Autopsy and further sorted by image extension.

  `File Views > File Types > By Extension > Images >`

- There are 97 images shown by the extension; most of them are corrupted or deleted; out of that, only 34 are displayed when right click on them.
- During the initial investigation, using Formost (8 jpg, 45 png, and 1 html) and mounting an image disk (only 19 both jpg and png), on the other hand, using Autopsy (searched by extension), I am able to extract hidden images.
- Extracting the image files from Autopsy to a local directory for future analysis by right-clicking on all of them and choosing the Extract files option.

  `/home/user/#DFT-USB-A-315/6893549-X024/Export/By_Extension/Images`

% 20/12/2024 13:55:29 %

- For some reason, I am not able to get both the hash values (MD5 and SHA256) in Autopsy, so I am again starting the case with the same details as above.
- This time, I extracted the whole CSV table list of image files (by extension category in Autosy) to the following directory. It can be done by clicking the save table option in the top-right side of Autosy.

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension/Images 20241220024249.csv`

- I also extracted the CSV table for images that are visible on the Thumbnails menu (top in Autopsy and next to the table) to the following location.

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension/Results 20241220030644.csv`

- While doing further investigation, I found 5 bamboo images, which are important for this investigation, so I marked them as noticeable items by `right-clicking them > Add file type > Noticeable items`
- Furthermore, I also extracted their CSV in the following path.

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension/Results 20241220034849.csv`

% 20/12/2024 18:23:08 %

S.N	Name	Modified Time	Access Time	Created Time	Location	MD5 Hash
1.	_00_0013.JPG	2019-04-27 14:44:32 BST	2019-04-27 01:00:00 BST	2019-04-27 14:44:32 BST	/img_X2024.dd/_ILES/_00_0013.JPG	c200635e6d08605c98feb53d42ef5376
2.	_00_0015.JPG	2019-04-27 14:45:00 BST	2019-04-27 01:00:00 BST	2019-04-27 14:45:00 BST	/img_X2024.dd/_ILES/_00_0015.JPG	2c55c84dec572e38d02e6cbbc4c66d32
3.	_00_0022.JPG	2019-04-27 14:45:20 BST	2019-04-27 01:00:00 BST	2019-04-27 14:45:20 BST	/img_X2024.dd/_ILES/_00_0022.JPG	7f0af17343351db61eb015578c753690
4.	_00_0025.JPG	2019-04-27 14:45:54 BST	2019-04-27 01:00:00 BST	2019-04-27 14:45:54 BST	/img_X2024.dd/_ILES/_00_0025.JPG	d6409696fb766d52a8f75d90b18a3d08
5.	_00_0026.JPG	2019-04-27 14:46:14 BST	2019-04-27 01:00:00 BST	2019-04-27 14:46:14 BST	/img_X2024.dd/_ILES/_00_0026.JPG	daa6f39572643c45c2b0d2e3c6f5bfed

% 20/12/2024 21:56:39 %

- Exploring zip files of #DFT-USB-A-315 with Autopsy.
- Moving towards the next file types, there are three encrypted zip files in Archives containing some content. The zip files include _00.zip, _01.zip, and _02.zip.

  `File Views > File Types > By Extension > Archives >`

- Extracted all zip files from Autopsy to the local directory (using the export file option) and compared their hash value with Autopsy zip files for data integrity using the Tsurugi Terniml command.

  ```
  /home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension/Archives

  md5sum zip_file_name.jpg > hashlist     # for data integrity
  ```

- Furthermore, I also added these zip files to Notable items for future reference by `right-clicking on them > Add File Tags > Noticeable items`
- This is the path where I extracted the CSV table of all three zip files. It contained most of the metadata for further analysis. The last part of the path is the filename.

  ```
  /home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension/Archives
  20241220075056.csv
  ```

- I did try to unlock the zip file, but it is encrypted with some kind of password.

S.N	Name	Modified Time	Access Time	Created Time	Location	MD5 Hash
1.	_00.zip	2020-01-13 10:18:32 GMT	2020-01-13 00:00:00 GMT	2020-01-13 10:18:32 GMT	/img_X2024.dd/_ILES/_00.zip	b6ae6c367ca9b100035529ce854bf05f
2.	_01.zip	2020-02-11 14:23:04 GMT	2020-02-11 00:00:00 GMT	2020-02-11 14:23:04 GMT	/img_X2024.dd/_ILES/_01.zip	5eccfdbcc84e00e25a4ebc247c6060a9
3.	_02.zip	2020-03-12 15:58:26 GMT	2020-03-12 00:00:00 GMT	2020-03-12 15:58:26 GMT	/img_X2024.dd/_ILES/_02.zip	6153001f00a8ff05d4a712e25958f235

% 22/12/2024 22:24:06 %

- Exploring databases by their extension.

  `File Views > File Types > By Extension > Databaseses >`

- There are 44 database files carved by the extension type; most of them belong to the SQLite database. Only 2 files have a .db extension, and out of those, one is empty and the other one is not showing data (BLOB data not shown).

- Most of the SQLite files are either empty or contain some kind of web search history, such as Google, Ubuntu, Wikipedia, Surrey, Favicon, and Domino, etc. (normal). Also, few database tables contained Chrome data.

- I selected 4 files from the database that contained web browser search history and exported them to a local directory (using Export files(s)) for further investigation. Furthermore, I also extracted their CSV table by right-clicking and selecting Export CSV.

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension/Databases`

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension/Databases 20241223120842.csv`

- During investigation, I notice a database file (places.sqlite) contained a link to a bamboo website. So I marked it as a notable item by right-clicking on the file and selecting Add tag.

  `https:// www.bamboogrove.com`

S.N	Name	Modified Time	Access Time	Created Time	Location	MD5 Hash
1.	cookies.sqlite	2022-07-22 22:08:24 BST	2022-07-22 01:00:00 BST	2022-07-22 22:08:25 BST	/img_X2024.dd/uskz4iry.default/cookies.sqlite	c4f2d505f5301289264 5df15c2814413
2.	permissions.sqlite	2022-07-22 22:08:24 BST	2022-07-22 01:00:00 BST	2022-07-22 22:08:25 BST	/img_X2024.dd/uskz4iry.default/permissions.sqlite	1c1eb7fbcc590e4e633a185b9dcb6114
3.	favicons.sqlite	2022-07-22	2022-07-	2022-07-	/img_X2024.dd/	5d1bf4c653ae376f7bb

		22:08:24 BST	22 01:00:00 BST	22 22:08:25 BST	uskz4iry.default/ favicons.sqlite	afb4e9c424917
4.	places.sqlite	2022-07-22 22:08:24 BST	2022-07-22 01:00:00 BST	2022-07-22 22:08:25 BST	/img_X2024.dd/ uskz4iry.default/ places.sqlite	40fc7f96a62de5379477d817014f6295

% **23/12/2024 01:06:09** %

- Exploring the Documents by their extension

  `File Views > File Types > By Extension > Documents > PDF >`

- Documents contains 8 files, 4 pdf and 4 text files, Iam unable to understand the text contained in text file.

- On the other hand, pdf contained some content in it but unable to load when clicked (Empty pages), So i exported all the 4 files to local directory using extract option (right-click).

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension/Documents`

- Furthermore, i also extracted their CSV by clicking on save table as CSV on top-right conner of Autosy.

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension/PDF 20241223015527.csv`

S. N	Name	Modified Time	Access Time	Created Time	Location	MD5 Hash
1.	Crime.pdf	2019-12-27 10:58:26 GMT	2019-12-27 00:00:00 GMT	2019-12-27 10:58:26 GMT	/ img_X2024.dd /_ILES/ Crime.pdf	375c3827b3bb15d2886c5daddd2a5b16
2.	Lupin.pdf	2019-12-27 10:58:26 GMT	2019-12-27 00:00:00 GMT	2019-12-27 10:58:26 GMT	/ img_X2024.dd /_ILES/ Lupin.pdf	cb669d48e5e216be31ef3c0efdc4fd78
3.	MurderGunroom.pdf	2019-12-27 10:58:26 GMT	2019-12-27 00:00:00 GMT	2019-12-27 10:58:26 GMT	/ img_X2024.dd /_ILES/ MurderGunroom.pdf	33d8f36346bfc9a1727246eaa8d40be2
4.	ThiefNight.pdf	2019-12-27 10:58:26 GMT	2019-12-27 00:00:00 GMT	2019-12-27 10:58:26 GMT	/ img_X2024.dd /_ILES/ ThiefNight.pdf	4dbfdd4fa734d57b69605e2afbe80cd0

% 25/12/2024 19:14:34 %

- All of the PDFs do not have their few front and back pages that may contain some information about the PDFs (opened using the Linux file browser using the command pcmanfm in the terminal).

- To get the information about the PDFs, I searched the first few lines of the text using the Firefox browser v128.0 (64-bit). I have also given the online links to these books pdf books for further investigation.

S. N	Name	Book_name	Author_name	Links
1.	Crime.pdf	CRIME AND PUNISHMENT	Fyodor Dostoevsky	https://www.gutenberg.org/cache/epub/2554/pg2554-images.html#link2H_EPIL
2.	Lupin.pdf	ARSÈNE LUPIN	Maurice Leblanc	https://www.gutenberg.org/files/4014/4014-h/4014-h.htm
3.	MurderGunroom.pdf	Murder in the gunroom	H. Beam Piper	https://gutenberg.org/files/17866/17866-h/17866-h.htm#CHAPTER_21
4.	ThiefNight.pdf	A Thief in the Night	E. W. Hornung	The Project Gutenberg eBook of A Thief in the Night, by E. W. Hornung

% 25/12/2024 20:03:33 %

- Exploring the deleted files from the Data Source

  `File Views > Deleted Files > All >`

- The Autopsy carved a total of 95 files that were deleted by the suspect; most of the files are corrupted, and some of them are the same as the abovementioned files filtered by their extension.

- I also extracted all the deleted files and their CSV to a local directory, using the same option mentioned above.

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/Deleted_Files`

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/Deleted_Files/All 20241225085105.csv`

- Furthermore, I noticed a few (7 total) email files containing some kind of conversation between 2 people. So, I marked it as a notable item using the Add File Tags option and extracted both the email and their CSV table to a separate folder.

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/Emails`

- One thing I notice is that emails 5 and 6 are having the same MD5 hash but different file name extensions. To confirm, it is the same. I renamed and checked (Re_Re_ Forgot.eml).

S.N	Name	Modified Time	Access Time	Created Time	Location	MD5 Hash
1.	Check Out My Latest Photo!.eml	2019-01-15 12:11:08 GMT	2019-01-15 00:00:00 GMT	2019-01-15 12:11:08 GMT	/img_X2024.dd/_ILES/Check Out My Latest Photo!.eml	de72fd746882000a75934fc73f26fa0c
2.	Re_ Check Out My Latest Photo!.eml	2019-01-16 13:13:10 GMT	2019-01-16 00:00:00 GMT	2019-01-16 13:13:10 GMT	/img_X2024.dd/_ILES/Re_ Check Out My Latest Photo!.eml	68fa87934f8e1a441bad63cf9c963f86
3.	Forgot.eml	2019-01-17 18:13:44 GMT	2019-01-17 00:00:00 GMT	2019-01-17 18:13:44 GMT	/img_X2024.dd/_ILES/Forgot.eml	9b8c886c626052ead70cd19e0539d5e9
4.	Re_ Forgot.eml	2019-01-20 18:13:44 GMT	2019-01-20 00:00:00 GMT	2019-01-20 18:13:44 GMT	/img_X2024.dd/_ILES/Re_ Forgot.eml	2c9eedb4538ff0b90bede1703696df87
5.	Re_Re_ Forgot.eml	2019-01-21 16:08:28 GMT	2019-01-21 00:00:00 GMT	2019-01-21 16:08:28 GMT	/img_X2024.dd/_ILES/Re_Re_ Forgot.eml	4a136daf760597f26229152728ecb7d9
7.	Re_Re_Re_ Forgot.eml	2019-01-22 16:18:32 GMT	2019-01-22 00:00:00 GMT	2019-01-22 16:18:32 GMT	/img_X2024.dd/_ILES/Re_Re_Re_ Forgot.eml	89091d1c5ef6e6b27d20672c167d6d86

% 29/12/2024 12:59:52 %

- Conversation in the email
  - These emails contain records of some conversations between two people. (Jamie and Alex). Both of them are using the following emails for the conversation, and the date matches with the metadata.

    ```
    Jamie            <aoijhegoijfeim.com>
    Alex        <alex@345897349578fgjoeqijfgo.com>
    ```

  - In the first mail (Check Out My Latest Photo! eml), Jamie sends Alex an email with a bamboo image attached to it (open in local directory in Tsurugi) and asks for its suggestions.
  - In the next few emails, they are talking about the password to a zip file that Alex downloaded from Jamie's website. He also confirms that the zip file contains pictures of some kind.

    ```
    https://www.bamboogrove.com
    ```

  - In the last mail (Re_Re_Re_ Forgot.eml), Jamie talks about a group and gives a clue how to decrypt the zip files. Furthermore, there is a mention of another person named Rob.

- Below are the images that possibly contain passwords and passphrases, sorted by date (befor suspect wrote emails) and the path where I put it.

  `/home/user/#DFT-USB-A-315/Images`

Name	Modified Time	Access Time	Created Time	Location	MD5 Hash
02010026.jpg	2016-07-13 11:04:22 BST	2016-07-13 01:00:00 BST	2016-07-13 11:04:22 BST	/ img_X2024.dd/ DCP/ 02010026.jpg	20f34a3f571d394ab9342ac21588e96e
09260002.jpg	2016-07-13 11:04:46 BST	2016-07-13 01:00:00 BST	2016-07-13 11:04:46 BST	/ img_X2024.dd/ DCP/ 09260002.jpg	bb1e236cf935815e5bb714760b23f549
100_0001.JPG	2016-07-13 11:05:32 BST	2016-07-13 01:00:00 BST	2016-07-13 11:05:32 BST	/ img_X2024.dd/ DCP/ 100_0001.JPG	49d2f6611ce8be710f98f7f348c2334a
100_0393.JPG	2016-07-13 11:06:20 BST	2016-07-13 01:00:00 BST	2016-07-13 11:06:20 BST	/ img_X2024.dd/ DCP/ 100_0393.JPG	417948ff026e25b29ca8194348b1c1bf
_otato.png	2019-01-22 17:33:08 GMT	2019-01-22 00:00:00 GMT	2019-01-22 17:33:08 GMT	/ img_X2024.dd/ _ILES/ _otato.png	6f1f0fbe61a8c35a46273ab5c4d5f72e
_ulp.png	2019-01-22 17:33:08 GMT	2019-01-22 00:00:00 GMT	2019-01-22 17:33:08 GMT	/ img_X2024.dd/ _ILES/_ulp.png	c5ea34850f8327cc7bff5bf89c06edb3

****
****

% 07/01/2025 18:06:51 %

- I had made a mistake while selecting the Ingest module (Autopsy); I selected the Key Search option without giving any, which resulted in an error message and incomplete analysis. Though I still get most of the files, but did not get all. So, this time I again created a new case with the same credentials, and in the below part I am giving the files that were not included.
- Furthermore, I also change the directory name from `#DFT-USB-A-315` to `case_files`

  `/home/user/#DFT-USB-A-315/6893549-X2024/Export/By_Extension`

  `/home/user/case_files/6893549-X2024/Export/By_Extension`

- % 19/12/2024 21:15:43 %, There are total 113 (35 PNG, 17 JPEG) images instead of 97, and 50 out of 113, which are not corrupted (15 JPG, 35 PNG), were extracted. Some of these files (13 images) are having the same hash value as other.

  `/home/user/case_files/6893549-X2024/Export/By_Extension/Images`

- % 20/12/2024 21:56:39 % There are 6 zip files; 3 are having the same hash value as others with different names.

- % 23/12/2024 01:06:09 % There is an HTML file in the Documents section, having the same content as the email (Check out my latest photo!), using the text option in the bottom corner of Autopsy.

  `File Views > File Types > By Extension > Documents > HTML >`

- % 23/12/2024 01:06:09 % Other files Documents contain 8 PDFs and 14 text files. Out of 8 PDFs only 4 are having the same content as the others.

- 25/12/2024 19:14:34 % Another mistake I made with the book name of ThiefNight.pdf titled **"The Amateur Cracksman"** by **E. W. Hornung**

- % 25/12/2024 20:03:33 % There There are a total of 129 deleted files and 95 file system files in deleted. section of autopsy.

- I also add the updated CVS table based upon my findings on the same directory, for example.

% 08/01/2025 14:49:04 %

- 29/12/2024 12:59:52 I did mention the potential images that contain hidden passwords and passphrases, but the last 2 (otato.png and ulp.png) have another copy with different names and hash values. So, I included both in the same image folder.

f0010220.png	/img_X2024.dd/$CarvedFiles/1/f0010220.png	26ffc68309f99c972d2b3f5db8dd0c88
f0011736.png	/img_X2024.dd/$CarvedFiles/1/f0011736.png	0ef9a3a98d23ddf2f1ce4f6d8f6e3d71

- I did Try to compare both these images to otato.png. and ulp.png using a Matlab (R2024b) program and find the difference on a single image. But, both are the same images of its counterpart. Furthermore, add the matlab file to case_files with name `same_img.m`

```
img_JPG = imread('Autopsy\195-_otato.png');
img_PNG = imread('Autopsy\4261-f0010220.png')

subplot(2,1,1),imshow(img_PNG),title('95-_otato.png');
subplot(2,1,2),imshow(img_JPG),title('4261-f0010220.png');

img_diff = abs(img_JPG-img_PNG);
figure,imshow(img_diff),title('|JPG Image - PNG Image|');

img_diff = abs(double(img_JPG)-double(img_PNG))/255;
img_diff = img_diff/max(img_diff(:));
figure,imshow(img_diff),title('|JPG Image - PNG Image|');
```

- % 19/12/2024 12:10:15 % I forgot to include the Python code file (Lab7_Autopsy.ipynb) in the case_files folder, which is used to get files metadata, including images.

****
****

% 09/01/2025 02:20:29 %

- Password Creaking, To crack the passwords, I use John the Ripper tools (v1.9.0).
- First, extract the hash value from three main zip files using JTR commands given below (`219-_00.zip, 221-_01.zip,and 223-_02.zip`), and then place their hashes on the seperate folder (`/home/user/case_files/zip_hash`).

```
zip2john 219-_00.zip > 219_hash.txt     # convert zip into hash value
zip2john 221-_01.zip > 221_hash.txt    # for this case its it pkzip
zip2john 223-_02.zip > 223_hash.txt
```

- Second, I created a blank document, name “wordlist,”

  - This document contains all the different possible passwords that could decrypt the hash value. This wordlist has the PDF book names and author names.
  - Furthermore, I also included a bash file (`binbash.sh`) and a wordlist file (`wordlist.txt`) in the case file, which is used to create different possible combinations of the generated wordlist (7.2 GB), and it took around 10-20 sec.
  - To run the bash script, I used these commands on Terminal, which will generate a new wordlist (gen_wordlist.txt) having multiple combinations.

```
chmod +x binbash.sh              # To run the bash Script
./binbash.sh
```

  - Then I did try this gen_wordlist to crack the zip file hashes using these commands and by default setting of JTR in the terminal. (no result).

```
john --format=pkzip --wordlist=gen_wordlist.txt password.hash
john --show password.hash
```

  - Moreover, I also tried putting possible hash values (both MD5 and SHA256) in the wordlist but was still not able to crack it.

% 09/01/2025 19:51:44 %

- After doing more investigation, I found the two images (`195-_otato.png, 197-_ulp.png`) are having the passphrases that will decrypt the zip file's hash. I used the LSB_steg_extract function in MATLAB (R2024b) to extract the hidden data from both images.

```
img = imread('File_name');      % Load image
p = [1, 100];                      % Range of characters to extract
p_class = 'char';                         % Data type of output
plaintext = LSB_steg_extract(img, [], p, p_class);   % Extract hidden text
disp(plaintext);                       % Display output
```

- I created a separate text file with the name `wordlist1.txt` and added the passphrases into it. Furthermore, I extracted all the images from the autopsy using the extract option and put them into two different folders as given below.

```
File Views > File Types > image > png >        #location in autopsy
File Views > File Types > image > jpg >

/home/user/case_files/png                # location where i extracted
/home/user/case_files/jpeg
```

- I also modified the above MATLAB program that can extract all the hidden passphrases from images within the same directory. So, I used that to extract the hidden text and added it to the `wordlist2.txt file`.

- Moreover, I added both the files (`utility_functions, sol.m`) in the case file and ran the code using MATLAB. Also, I added these 10 images out of 35 PNGs that contained passphrases to the case_file folder.

S.N.	File_name	Passphrases
1.	185-_ward.png	19:00, Monday, Pantomime, Cat Cafe, Hyderabad,India
2.	187-_arry.png	20:00, Saturday, David Copperfield, Lighthouse, Oeiras,Portugal
3.	189-earth-fruit.png	3pm, Saturday, Tap Dancing, Community Centre, Bern,Switzerland
4.	191-_njoy.png	6pm, Monday, 1980s Acapella, Cafe 123, Yangon,Myanmar
5.	193-_rain.png	18:30, Saturday, Pantomime, Area 51, Bangkok,Thailand
6.	195-_otato.png	6pm, Thursday, Southern Appalachian Step Dancing, Area 51, Atlanta,United State
7.	197-_ulp.png	21:00, Wednesday, Open Mic Night, Podium, Istanbul,Turkey
8.	199-_auce.png	19:00, Wednesday, The Beatless, Motel 24, Thessaloniki,Greece
9.	201-_tory.png	19:30, Tuesday, Karate Championship, Pizzeria Cosa Nostra, Kilkenny,Ireland
10.	203-_rban.png	18:00, Monday, Open Mic Night, The Upside-Down, Pune,India

- For jpg images, I am not able to extract the passphrases. The program will not work except for 2 images, 10-09260002.jpg and 12-100_0001.JPG, but it is unable to read text.

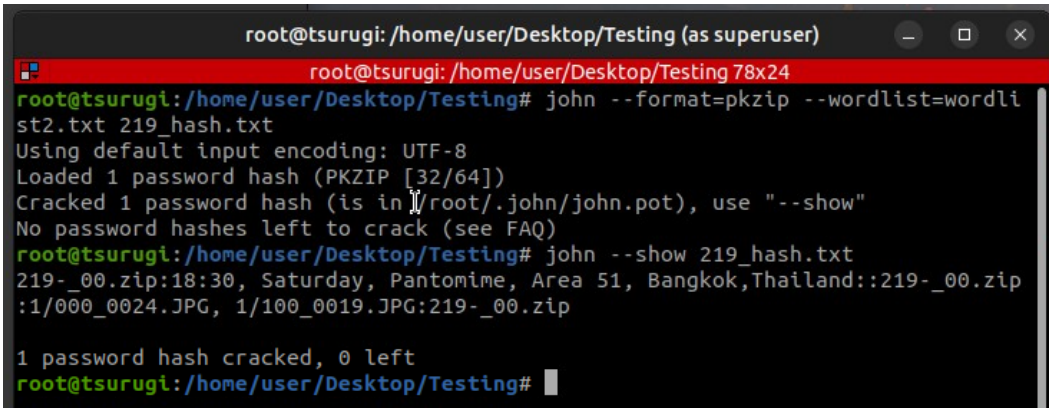
% 10/01/2025 01:32:57 %

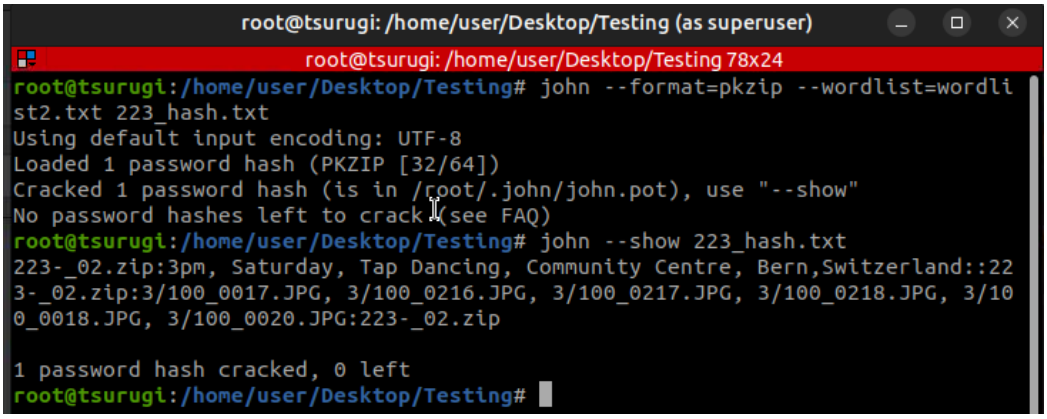
- Again, I used the JTR to crack the zip file hashes using the wordlist (`wordlist2.txt`) created out of passphrases.

  `john --format=pkzip --wordlist=wordlist2.txt 219_hash.txt`

  `john --format=pkzip --wordlist=wordlist2.txt 223_hash.txt`

  `john --show 223_hash.txt`





S.N.	zip_file_name	password
1.	219-_00.zip	`18:30, Saturday, Pantomime, Area 51, Bangkok,Thailand`
2.	221-_01.zip	
3.	223-_02.zip	`3pm, Saturday, Tap Dancing, Community Centre, Bern,Switzerland`

% 10/01/2025 02:53:38 %

- Two zip files contain a total of 8 bamboo images. I used the same Python program in Jupyter to extract their hash value. (`Lab7_Autopsy.ipynb`)

S.N.	219-_00.zip	md5_hash
1.	000_0024.JPG	a0f370019a38ae509368aa225179d53d
2.	100_0019.JPG	3c33bc0a98ed26750111d58a3707bafc

S.N.	223-_02.zip	md5_hash
1.	100_0017.JPG	4e8d55fec1d40cc70d4cf4abb902acba
2.	100_0018.JPG	ed128d9487c16f2396bb55ff2f4cd64b
3.	100_0020.JPG	c4293660cd9482b70875fd8c4bf0c296
4.	100_0216.JPG	31c64a735135098d7fd3ac08635c12a4
5.	100_0217.JPG	bdc1381322ba67f15d9f3577e44cb333
6.	100_0218.JPG	2987e1bd7a187d0cb8dae9333aed7408

****
****

%09/01/2025 19:51:44 % i did not add the hash value of all the main 10 png images having a passphrases

185-_ward.png	a8aae24cd3c9eaff444961c2b289363e
187-_arry.png	2b58a64787b378475b5bd910994d1899
189-earth-fruit.png	a67fa0c8b5fb4bd1cb29d897382750b2
191-_njoy.png	a540d613a3c2c5c0fbafc3f0f77aa6c1
193-_rain.png	d6b6270f2a2f30f13a0fcdafaf0ca053
195-_otato.png	6f1f0fbe61a8c35a46273ab5c4d5f72e
197-_ulp.png	c5ea34850f8327cc7bff5bf89c06edb3
199-_auce.png	a05be6281929bf01608a0d4caf97d49c
201-_tory.png	dde733995a1234009800ca69e13628d9
203-_rban.png	6e3483b79eecc171a7a14ad079a2590f

%10/01/2025 11:58:11 % doing some modification

- I included all the 13 bamboo images that I was able to extract into a single folder. (`case_files\bamboo_images`).
- I submitted the case in Autopsy by selecting the button and created an HTML page in this location.

`"case_files\6893549-X2024\Reports\6893549-X2024 HTML Report 01-10-2025-06-00-42"`

****
****

The line manager has the following questions for you:

i. The forensic investigation of the USB stick (#DFT-USB-A-315) uncovered evidence of 13 unique bamboo images. Initially, five images, 00_0013.JPG, 00_0015.JPG, 00_0022.JPG, 00_0025.JPG, and 00_0026.JPG, were recovered by mounting the disk and exploring its contents using Autopsy. Further analysis revealed the presence of three encrypted zip files (00.zip, 01.zip, and 02.zip), which were suspected to contain additional images. Using the LSB_steg_extract.m function, hidden passphrases were extracted from 10 stego images using MATLAB. This allowed decryption of the two zip files, resulting in the successful extraction of eight additional images. 000_0024.JPG, 100_0019.JPG, 100_0017.JPG, 100_0018.JPG, 100_0020.JPG, 100_0216.JPG, 100_0217.JPG, and 100_0218.JPG. However, three images, 00_0219.JPG, 100_0220.JPG, and 100_0221.JP, could not be decrypted using the same wordlist.txt file.

ii. Yes, the evidence indicates how the images were obtained through an email exchange between Jamie and Alex. Alex discovered encrypted zip files containing images on Jamie's webpage but lacked the password to access them. In a reply *Re: Check Out My Latest Photo!*, Jamie explained that the passphrases required for decryption were embedded in photos they had taken. He also mentioned Rob, who was familiar with the process, and advised Alex to use a decryption tool (John the Ripper). This correspondence shows that the images were deliberately concealed using encryption and steganography, suggesting a methodical approach to secure their distribution.

iii. The investigation suggests that the suspect acted as part of an organized group rather than alone. Email exchanges reveal collaboration between Jamie, Alex, and a third individual named Rob, who was aware of the encrypted zip files. Jamie's instructions to use specific decryption tools and his reference to frequent group interactions imply a coordinated effort. Additionally, the emphasis on deleting emails highlights the group's intent to conceal their activities, indicating an organized and deliberate approach to sharing encrypted content.

iv. There are two key pieces of evidence indicating potential illegal imports of bamboo images. The first is a link to the website (https://www.bamboogrove.com), which suggests that Alex may have obtained encrypted zip files and images from this source. The second involves an IT administrator at the University of Surrey, who detected illegal bamboo-related traffic through a deep packet inspection system. Furthermore, the extraction of hidden passphrases from stego images revealed details such as, 19:00, Monday, Pantomime, Cat Cafe, Hyderabad, India, indicating that these images may have been imported from various cities between 2016 and 2019.

v. During my investigation, I am not able to find any definitive clue directly indicating who accessed the USB stick. However, based on the available evidence, only three individuals Jamie, Alex, and Rob are linked to the encrypted zip files and bamboo images stored on the device.

3. Reflection:

- Alice's setup provides stronger integrity guarantees for contemporaneous notes compared to Bob's because it uses an automated, periodic timestamping system that operates independently of her actions. This ensures any modifications to her notes disrupt the sequence of timestamps, making tampering detectable. In contrast, Bob's setup, which only generates timestamps when new entries are made, creates gaps in the record, leaving room for alterations between entries. While Bob's method links timestamps directly to specific actions, it depends heavily on his self-discipline, reducing the system's overall reliability. Alice's setup, driven by external policy, minimizes human

influence and provides a more robust audit trail, better ensuring the integrity of the notes in forensic contexts.

- Bob can cheat in ways that Alice cannot due to the voluntary nature of his timestamping system. Since Bob's setup generates timestamps only when he actively makes new entries, he can modify older entries without creating a new timestamp or leaving evidence of tampering. This gap between timestamps allows for undetected alterations, as no mechanism records periods of inactivity changes. Alice, on the other hand, cannot exploit such gaps because her system generates timestamps continuously, irrespective of her activity. Any alteration to her notes would disrupt the sequence of timestamps, making tampering evident. Thus, Bob’s setup provides opportunities for undetected manipulation that Alice’s system.
  On the other hand, Alice’s system automatically generates timestamps at regular intervals, even when no changes are made. This continuous timestamping creates a verifiable log, making it much harder for her to alter past entries without detection. Any modification would disrupt the regular timestamp pattern, revealing tampering.

- Alice's TSA-generated timestamps, even when her contemporaneous notes remain unchanged, are not redundant. These regular timestamps ensure a continuous audit trail, vital for detecting any tampering or alterations. By providing verifiable evidence that no modifications occurred, they strengthen the integrity of her records. While they use some storage space, the benefits of maintaining accountability and security in forensic investigations outweigh this minimal cost. In an academic or legal context, such tamper-evident mechanisms are crucial for preserving the reliability of contemporaneous documentation.