Week 05 Metadata Report

Name: Huynh Lam **Student ID: 13264763** **Date: 05/09/2021**
Activity No.: Cmp1/03

Due Date: Three days after the lab.

Setup

You need to download this week's samples onto a Linux Box.

Q1) Hex Viewer – cmd line

We can use xxd -l 256 to see the file signatures.

Note your response in the table

Can you determine the imposters file type?

- Flowers.txt should be exe
- cars.txt should be exe

Exe files have a PE jump between 80 and FF see slides. Add to the table.(exe only)

Why does the address of the PE marker vary?

- These files are referred to as Portable Executable PE. The name Portable Executable refers to the fact that the format is not architecture specific which means the address of PE being in a different spot is normal

File	Hex Signature	exe PE Jump address
Trade_secrets.txt	Ascii text	
logo.gif	GIF89a at 00	
MS Office Meta Data.jpg	(JFIF) FFD8 FF	
IMAG1672a.jpg	(JFIF) FFD8 FF	
Cygwin1.dll	(MZ) 4d5a	
Strings.exe	(MZ) 4d5a	D8
Sample.docs	(PK) 504B 0304	
Sample.pdf	(%PDF) 2550 4446	
Flowers.txt	(MZ) 4d5a	80
Cars.txt	(MZ) 4d5a	D8

Q2) Magic Files

We use Linux command file on your downloaded files.

Open a Linux shell window and cd to your Forensics folder.

Use the command file to check the file extensions.

```
file /mnt/c/Forensics_yourname  * | cut -c 1-120
```

Add the responses in the table

File	Magic Signature
Trade_secrets.txt	iso-8859 (simple) text, CRLF
logo.gif	version 89a, 220 x 50 pixels
MS Office Meta Data.jpg	JPEG image data, JFIF standard 1.01, resolution (DPI)
IMAG1672a.jpg	JPEG image data, JFIF standard 1.01, resolution (DPI)
cygwin1.dll	PE32 executable (DLL) (console) Intel 80386, for MS Windows
strings.exe	PE32 executable (console) Intel 80386, for MS Windows
Sample.docs	Microsoft Word 2007+
Sample.pdf	PDF document, version 1.5
Flowers.txt	PE32 executable (DLL) (console) Intel 80386, for MS Windows
cars.txt	PE32 executable (console) Intel 80386, for MS Windows

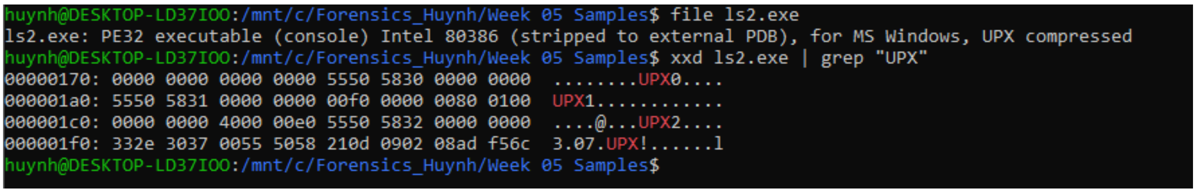
Are any files imposters as seen by file?

Imposters: Cars and flowers text files are exe files

Run file again on ls2.exe without using cut.

ls2.exe is a compressed form of ls.exe using UPX packing to avoid detection in an IDS.

You can also use xxd on ls2.exe and grep for UPX. Take a screenshot of this cmd and the result.



```
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh/Week 05 Samples$ file ls2.exe
ls2.exe: PE32 executable (console) Intel 80386 (stripped to external PDB), for MS Windows, UPX compressed
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh/Week 05 Samples$ xxd ls2.exe | grep "UPX"
00000170: 0000 0000 0000 0000 5550 5830 0000 0000  ........UPX0....
000001a0: 5550 5831 0000 0000 00f0 0000 0080 0100  UPX1............
000001c0: 0000 0000 4000 00e0 5550 5832 0000 0000  ....@...UPX2....
000001f0: 332e 3037 0055 5058 210d 0902 08ad f56c  3.07.UPX!......l
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh/Week 05 Samples$
```

What is UPX packing?

- UPX (Ultimate Packer for Executables) is an open-source executable packer supporting a number of file formats from different operating systems.
- By packing malware binary files, the data stored within the file becomes unreadable and thus will need to be unpacked in order to become readable again. These require the malware to be unpacked manually

Q3) Editing a File Header

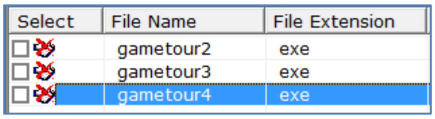
Confirm you have the Image File C08InChp.dd in your Forensics folder.

Run ProDiscover.

Add Image File C08InChp.dd.

Search for FIF case sensitive

Click Show File. You should see a deleted file called gametour4.exe.



Select	File Name	File Extension
□	gametour2	exe
□	gametour3	exe
□	gametour4	exe

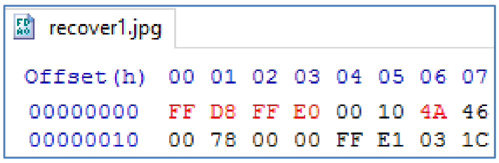
Right click, select copy file, save as Recover1.jpg in your Forensics folder. Exit ProDiscover.

Run HxD.exe in your Forensics folder. (Download and Install HXD as required,)

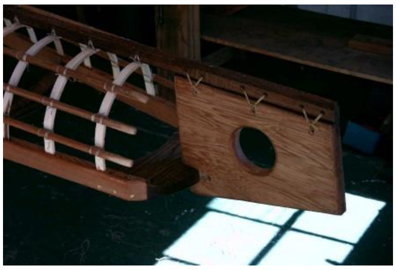
Open Recover1.jpg

Change the header from 7A 7A 7A 7A to FF D8 FF E0 (see lecture slides).

Change the 7A at address 06 to 4A.



```
Offset(h) 00 01 02 03 04 05 06 07
00000000  FF D8 FF E0 00 10 4A 46
00000010  00 78 00 00 FF E1 03 1C
```



Does the image support the allegation?

- If the image is part of the bike next model that has not been released by the company, then the image does support the allegation otherwise there would be more evidence needed

Q4) File Metadata

Confirm you have downloaded the sample files as in Q1.

Q4 A) docx files

Use the Windows 10 File Explorer.

Right click **Sample.docx** and select properties. Select the details tab.

Note the metadata.

What is the Document Title? Forensic Sample

What is the Document Subject? Forensic

What are the Document Tags? Forensics, Metadata

Who is the Author? G G Lee

When was it last printed? 9/29/2013 4:36 PM

Q4 B) pdf files

Open Sample.pdf with a pdf reader such as pdf-XChange or Chrome. Select the File Tab.

https://www.tracker-software.com/product/pdf-xchange-editor

Check the Document Properties.

Note here data of forensic interest:

List the tag:value pairs of forensic interest and explain what they mean.

The Title Forensic Sample (Title of the PDF Document)

The Keywords Forensics, Metadata (Keywords to specify what the topic is)

The Author G G Lee (The owner or author of this document)

The program that generated the pdf. Microsoft Word 2021 (This is the original program before it was a PDF)

The date last modified 29/09/2013, 16:36:31 (file was last 'Saved' by an application (whether or not its contents were actually modified/changed)

(This is usually the same as the file modify date.)

The PDF Version 1.5 (Version of the PDF)

Q4 C) Changing file dates

You need the Linux Box you used in Q1 again.

Open your Terminal shell and cd to the folder containing your sample files.

Let us now **stat** Sample.pdf

stat Sample.pdf

Note here the file modify date Modify: 2021-08-25 17:38:01.418284500 +1000

Does it match the Meta data date above? No

Now **touch** the modify date.

touch –m Sample.pdf

Use stat to check the modify date is now today Modify: 2021-08-25 18:43:11.042835400 +1000

Does the File modify date still match the pdf metadata date as seen by pdfinfo?

__ The file modify date has changed in Linux WSL, but the pdf properties have stayed the same

D) Exif files

Use the Windows 10 File Explorer to select IMAG1762a.jpg.

Look at the image file properties.

Explain the several dates.

- Date taken is when the photo was taken
- Date created should be the date that a particular physical instance of a file was written to disk
- Date modified should only ever be the date that the file was last 'Saved' by an application (whether or not its contents were actually modified/changed)

What is the camera? HTC Sensation Z710a

Where was the photo taken? Balmoral, Mosman NSW 2088

Explain how you decoded the GPS coordinates.

- Using this link I was able to find the location
- [https://support.google.com/maps/answer/18539?hl=en&co=GENIE.Platform%3DDesktop]
- GPS Latitude: 33°49' 37.86 S
- GPS Longitude: 151°15' 7.42 E

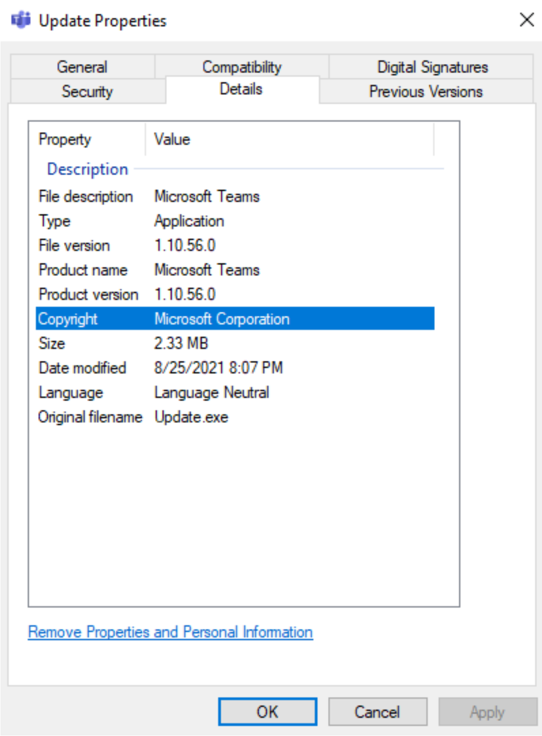
E) Other Metadata Samples

Find more files with interesting metadata. (Email? Photos?)

Describe the file and it's metadata of forensic interest.

Chosen to do Microsoft Teams

- Microsoft teams exe file is an automatic update file when the shortcut is ran
- This has the product detail such as name and version
- Company details which are Microsoft Corporation
- Date modified would be the date that it was installed



Property	Value
Description	
File description	Microsoft Teams
Type	Application
File version	1.10.56.0
Product name	Microsoft Teams
Product version	1.10.56.0
Copyright	Microsoft Corporation
Size	2.33 MB
Date modified	8/25/2021 8:07 PM
Language	Language Neutral
Original filename	Update.exe

Q5) Hashing.

Run Ubuntu on Windows 10. Then cd to /mnt/c/Forensics_yourname.

Create a text file called test with contents "I will pay you \$1000" using echo. Note the back slash. Check with cat test.

Create a second file called test1 with contents "I will pay you \$9000" using echo.

Compare the two file lengths with ls -al

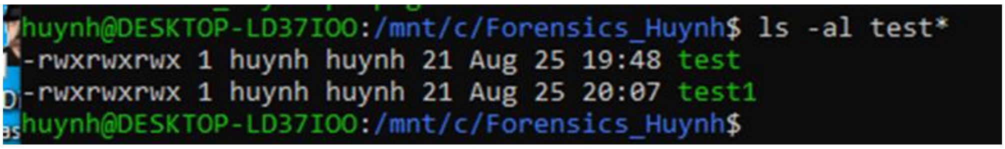
Hash both files with md5 and then sha 256.

Compare the two md5 hash lengths and compare the two sha 256 hash lengths.

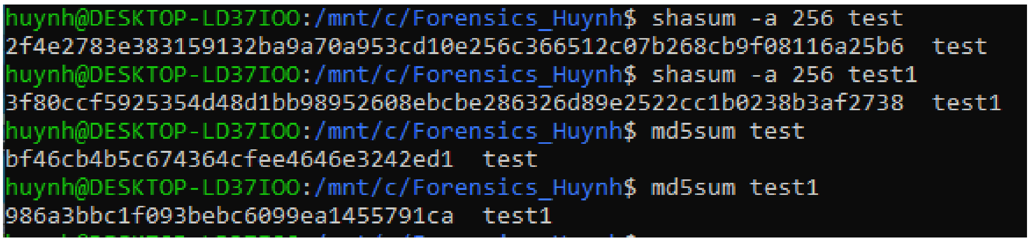
- md5:
  - Test: bf46cb4b5c674364cfee4646e3242ed1
  - Test1: 986a3bbc1f093bebc6099ea1455791ca
- SHA256:
  - Test: 2f4e2783e383159132ba9a70a953cd10e256c366512c07b268cb9f08116a25b6
  - Test2: 3f80ccf5925354d48d1bb98952608ebcbe286326d89e2522cc1b0238b3af2738

Compare the hashes of the two files. The hash length for both hashes is respectively the same number of characters. However, MD5 results in an output of 128 bits whereas SHA256 results output of 256 bits.

Take a screen shot showing the two file lengths, the two MD5 hashes and the two SHA 256 hashes.



```
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh$ ls -al test*
-rwxrwxrwx 1 huynh huynh 21 Aug 25 19:48 test
-rwxrwxrwx 1 huynh huynh 21 Aug 25 20:07 test1
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh$
```



```
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh$ shasum -a 256 test
2f4e2783e383159132ba9a70a953cd10e256c366512c07b268cb9f08116a25b6  test
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh$ shasum -a 256 test1
3f80ccf5925354d48d1bb98952608ebcbe286326d89e2522cc1b0238b3af2738  test1
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh$ md5sum test
bf46cb4b5c674364cfee4646e3242ed1  test
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh$ md5sum test1
986a3bbc1f093bebc6099ea1455791ca  test1
```

Comment on the results as follows:.

- What is the change in file length as seen by ls? There is no change in the file length
- How does MD5 output differ from SHA 256? MD5 results in an output of 128 bits whereas SHA256 results output of 256 bits.
- What is the change in the hash lengths of the two files?
  - SHA256 has 64 characters and is 256 bits
  - MD5 has and is 32 characters 128 bits
- What is the change in the hash values of the two files? Both hashes are alphanumeric characters and have their own respective order which are different to each other
- How good is hashing in protecting a file integrity? Hash values are also useful for verifying the integrity of data. Hashing is useful in verifying the actual integrity of a

file to prevent anybody from changing the content of a file or corrupting it and passing it off as the original file.

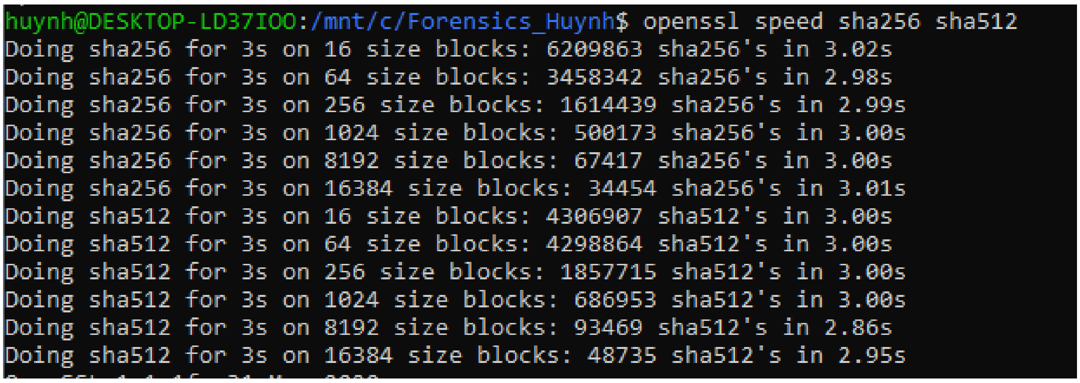
- How can you fool hashing? There is a chance that 2 files can output the same hash number. This is very unlikely, but it is called a hash collision. Since a hash function gets us a small number for a big key, there is the possibility that two keys result in the same value.

Explain the 512 and 224 options for SHA 2. These are different hash functions. SHA-512 novel hash functions computed with 64-bit words. SHA-224 are truncated versions of SHA-512. SHA-512/224 is a method for generating initial values for truncated versions of SHA-512. The 2 functions produce the digest of a message, respectively 512 and 224 bits long

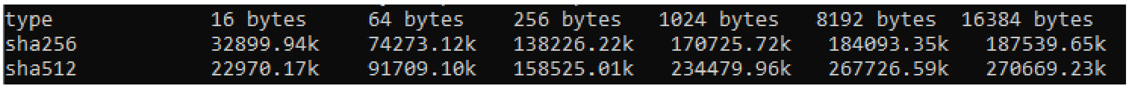
What was SHA 3 originally called? Keccak

Is SHA 3 better than SHA 2? In terms of collision potential SHA-3 is better and stronger than SHA-2 since SHA-3 uses more bits

Try this command openssl speed sha256 sha512 What does it show you?



```
huynh@DESKTOP-LD37IOO:/mnt/c/Forensics_Huynh$ openssl speed sha256 sha512
Doing sha256 for 3s on 16 size blocks: 6209863 sha256's in 3.02s
Doing sha256 for 3s on 64 size blocks: 3458342 sha256's in 2.98s
Doing sha256 for 3s on 256 size blocks: 1614439 sha256's in 2.99s
Doing sha256 for 3s on 1024 size blocks: 500173 sha256's in 3.00s
Doing sha256 for 3s on 8192 size blocks: 67417 sha256's in 3.00s
Doing sha256 for 3s on 16384 size blocks: 34454 sha256's in 3.01s
Doing sha512 for 3s on 16 size blocks: 4306907 sha512's in 3.00s
Doing sha512 for 3s on 64 size blocks: 4298864 sha512's in 3.00s
Doing sha512 for 3s on 256 size blocks: 1857715 sha512's in 3.00s
Doing sha512 for 3s on 1024 size blocks: 686953 sha512's in 3.00s
Doing sha512 for 3s on 8192 size blocks: 93469 sha512's in 2.86s
Doing sha512 for 3s on 16384 size blocks: 48735 sha512's in 2.95s
```



```
type             16 bytes     64 bytes    256 bytes   1024 bytes   8192 bytes  16384 bytes
sha256           32899.94k    74273.12k   138226.22k   170725.72k   184093.35k   187539.65k
sha512           22970.17k    91709.10k   158525.01k   234479.96k   267726.59k   270669.23k
```

The OpenSSL Speed command shows us the speed output for these algorithms. As you can see we move up the input block size the speed for SHA-3 processes much faster than SHA-2. With a smaller string sizes SHA-2 still performs faster.