

작성자	김서영
분석 일자	2024.05.09.
작성 일자	2024.05.09~2024.05.11.
분석 대상	Enc, flag.jpg
문서 버전	2.0
작성자 E-mail	<a href="mailto:sykim1802@naver.com">sykim1802@naver.com</a>

0. 목차

1. 문제 .....3

2. 분석 도구 .....3

3. 환경 .....3

4. Write-Up.....4



5. Flag..... 10

6. 별도 첨부 ..... 11

7. Reference ..... 12



## 1. 문제

URL	<a href="#">Enc-JPG   워게임   Dreamhack</a>
문제 내용	<p>드림이가 JPG 파일 만드는 중에 번조가 되었어요!!</p> <p>JPG 파일이 단단히 이상하네요...??</p> <p>문제 파일이 하나 잘못생성 되었습니다. π-π</p> <p>마지막 사진 플래그 글자수는 4 개입니다.</p> <p>DH{JPG_TXT_PNG}</p>
문제 파일	<div>  Enc            flag.jpg         </div>
문제 유형	Disk forensics
난이도	2 / 5

## 2. 분석 도구

도구명	다운로드 링크	Version
HxD	<a href="#">HxD - Freeware Hex Editor and Disk Editor   mh-nexus</a>	2.5.0.0

## 3. 환경

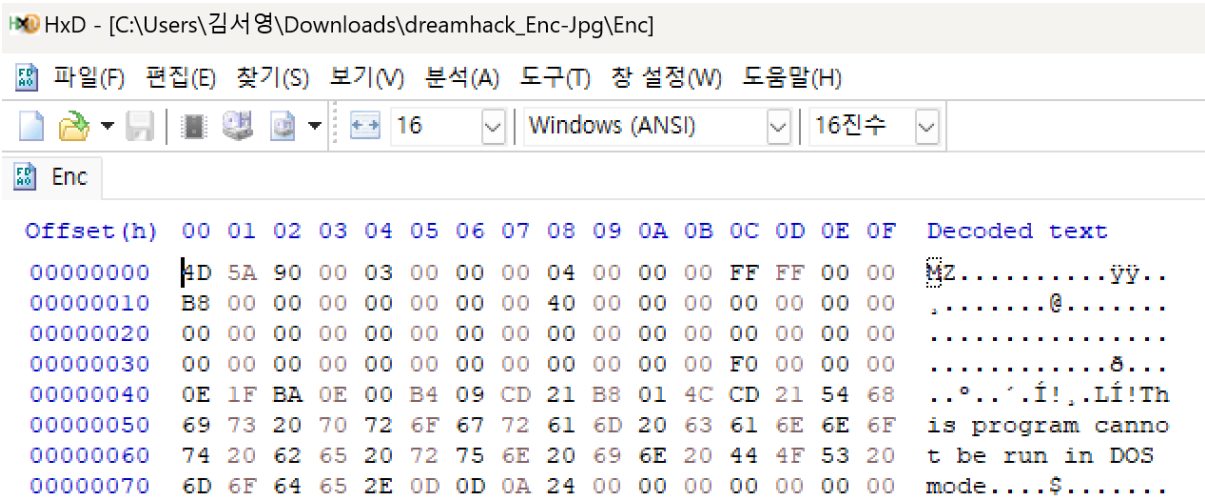
OS
Windows11 x64

## 4. Write-Up

파일명	Enc
용량	12kb
SHA256	8c580ed164ff882562525259b1c8a7f8621198fb17129a5172e587acad599521
Timestamp	2023-01-07 09:37:02

파일명	flag.jpg
용량	8kb
SHA256	d49f71f8216af57d6d297b97e25404abcb4bcb6bfc1553e79fa5f39d3bb0a0af
Timestamp	2023-01-07 09:37:02

1. Enc 파일의 확장자명이 없다는 점을 수상히 여기며, 우선 Hex Editor 를 사용하여 Enc 파일을 열어보았다.



[사진 1] Enc 파일을 HxD 로 본 결과

헤더 부분을 보니 파일 시그니처가 4D 5A (MZ)인 것을 보고 파일 시그니처 사이트에서 검색했다.

4D 5A	MZ
COM, DLL, DRV, EXE, PIF, QTS, QTX, SYS	Windows/DOS executable file (See <a href="#">The MZ EXE File Format</a> page for the structure of an EXE file, with coverage of NE, TLINK, PE, self-extracting archives, and more.) <b>Note:</b> MZ are the initials of Mark Zbikowski, designer of the DOS executable file format.

[사진 2] 파일 헤더 시그니처 값 4D 5A 검색결과([File Signatures \(garykessler.net\)](#))

윈도우 실행파일의 헤더 시그니처를 사용하는 것을 확인했다.

## [WHS-2] .iso

2. Flag.jpg 파일이 열리지 않는 것을 수상히 여기며, Hex Editor 를 통해 헤더를 살펴보았다.

Offset (h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	1D	DA	DB	B9	6D	D3	E2	34	61	3A	11	CF	C8	00	F6	FA	.ÚÛ²mÓâ4a:.ÏÈ.ôú
00000010	F2	A2	0E	05	AD	E0	5A	C9	97	98	30	C0	A2	C1	5D	37	òç...àZÉ~0ÀçÁ]7
00000020	AA	8A	E5	E4	FE	10	18	0A	85	1E	31	7C	86	70	36	01	*Šâäp.....l tp6.
00000030	1B	2C	DB	28	01	62	FA	36	38	E9	CA	FC	7D	6B	4D	CD	.,Û(.bú68éÊû}kMÍ
00000040	C2	E7	FC	EB	61	3F	9E	71	6A	3F	D2	49	17	E5	C6	5F	Âçüëa?žqj?ÒI.âÆ_
00000050	FE	8B	12	91	F8	2E	3C	FE	A4	6F	B4	BD	FC	91	D3	E0	p<.'ø.<pøø'ü'Óà
00000060	EF	4C	5A	E3	DB	52	1B	71	13	64	8E	5A	7C	BB	88	2B	iLZãÛR.q.dŽZ »^+
00000070	C7	79	C8	9C	6A	80	BD	1D	B6	52	0B	7B	7C	67	58	29	ÇyÊœj€¾.ŸR.{ gX)
00000080	19	FF	3E	C8	B5	93	04	2F	7F	54	8B	FF	52	34	61	8A	.ÿ>Êµ"/.T<ÿR4aŠ
00000090	CC	FA	9A	65	F0	1C	83	30	CF	42	C0	85	B5	42	5B	48	İúšeð.f0İBÀ...µB[H
000000A0	98	DD	58	34	B6	3A	C9	D3	A7	5F	94	E0	EF	23	DF	5B	~ýX4Ÿ:ÉÓ\$_"âi#B[

[사진 3] flag.jpg파일을 HxD로 본 결과

헤더를 보니 파일 시그니처 값이 1D DA 인데, 사이트 검색 결과 .jpg 파일의 헤더 시그니처 값은 FF D8 이다.

FF D8

ÿø

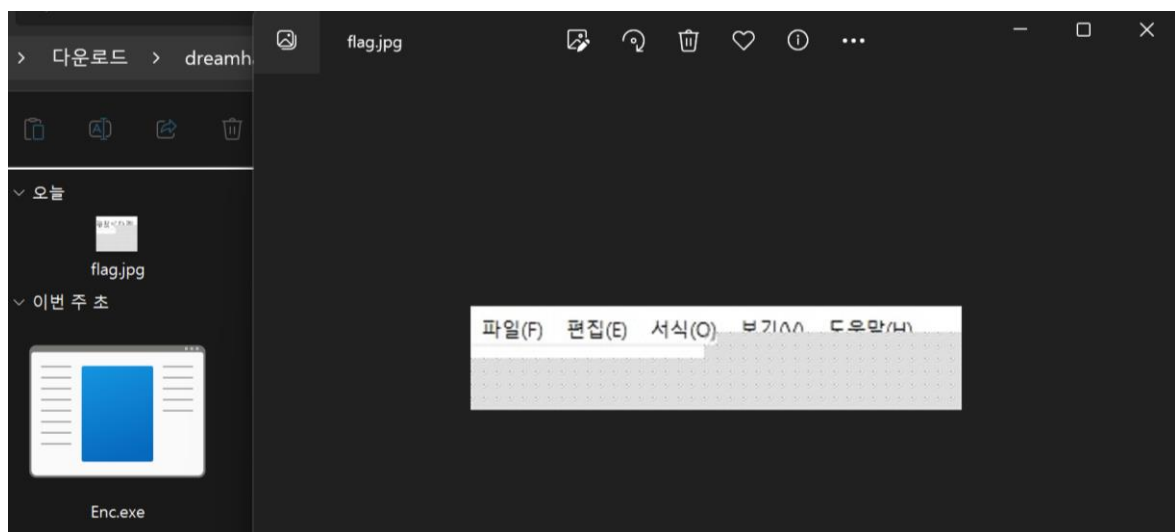
JPE, JPEG, JPG Generic JPEG Image file

Trailer: FF D9 (ÿÙ)

[사진 4] Jpg파일의 헤더 시그니처, 푸터 시그니처 값([File Signatures \(garykessler.net\)](http://File Signatures (garykessler.net)))

이를 통해 flag.jpg 파일이 손상되었음을 알 수 있다.

3. 우선 Enc 파일에 윈도우 실행파일의 확장자명인 .exe 를 붙여 다시 실행해보았다.



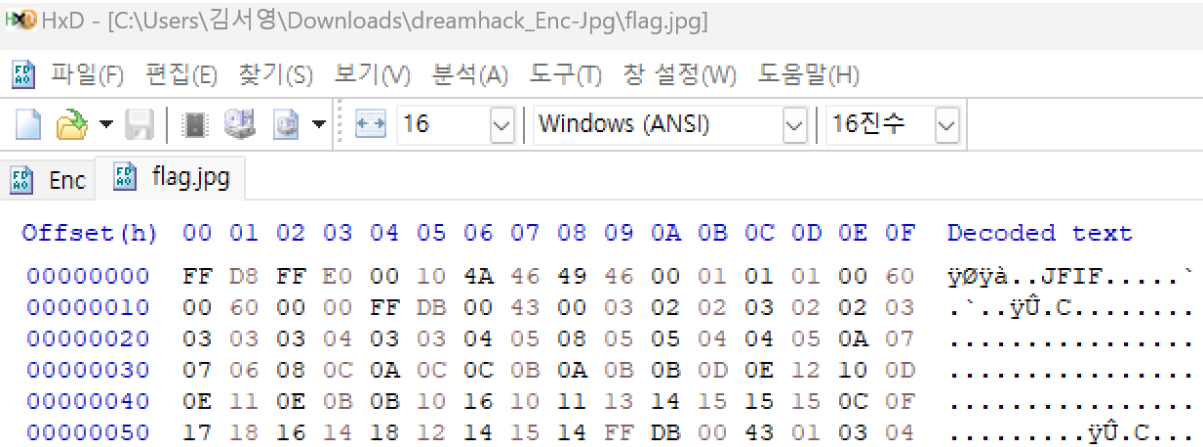
[사진 5] 복구된 flag.jpg

**[WHS-2] .iso**

Enc.exe 파일을 실행하자 잠시 cmd 창이 켜졌다가 꺼진 후, 기존에 열리지 않았던 Flag.jpg 가 복구되어 열렸다.

하지만, 회색 부분에 의해 Flag 값이 가려져 있는 것으로 추측된다.

4. 회색 부분을 제거할 수 있는 방법을 모색하기 위해 우선 복구된 flag.jpg 파일을 다시 Hex Editor 로 열어보았다.

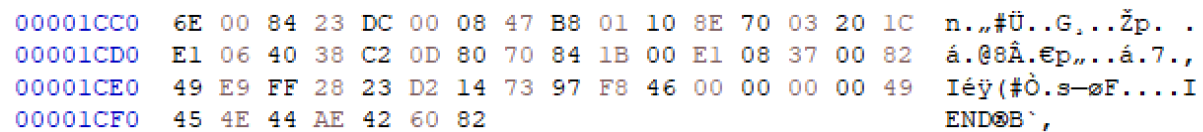


**[사진 6] 복구된 flag.jpg파일을 HxD로 본 결과**

헤더 시그니처가 정상적으로 FF D8 로 시작되는 것을 확인할 수 있다.

문제 내용에서 알 수 있듯이 플래그 값의 첫 부분은 JPG 와 관련되어 있기에, Jpg 파일 관련해서 더 살펴볼 것이다.

헤더 시그니처가 정상이니, 다음으로 푸터 시그니처를 살펴보았다.



**[사진 7] 복구된 flag.jpg 파일의 푸터 시그니처 값**

[사진 4]에서 확인했듯이 jpg 파일의 푸터 시그니처는 FF D9 인데, 일치하지 않는 것을 확인했다.

그래서 '찾기' 기능을 사용하여 FF D9 을 검색해보았다.

[WHS-2] .iso

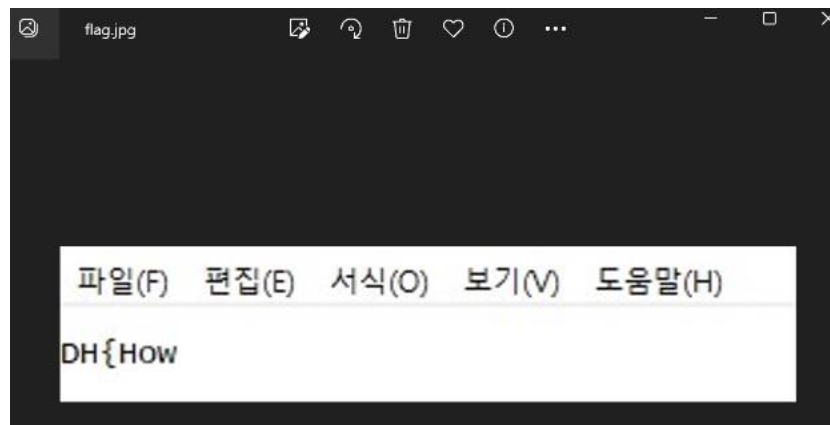
오프셋	잘라내기 (16진수)	잘라내기 (텍스트)
CC7	E9 56 FA AB EA B0 5C EB 30 EB 5E 20 D6 F5 8B A5 FF D9 D9 6D 42 FE 58 1D 25 84 6D 8E 35 8F ...	èVú«è««èèè^ Ôö«ÿÿÜmBpX.%mZ5.ä...
12C6	28 A2 8A 00 28 A2 8A 00 28 A2 8A 00 28 A2 8A 00 FF D9 46 69 6E 64 00 74 68 65 00 53 74 ...	(«.(«.(«.(«.«ÿÿÜFind.the.Strin

**[사진 8] 푸터 시그니처 FF D9 검색 결과(1)**

총 2 개의 검색 결과가 나왔다.

푸터 시그니처가 두 번인 것을 수상히 여기며 첫 번째 결과의 FF D9 을 삭제해보았다.

저장 후 다시 flag.jpg 파일을 열어보니, 기존에 회색 부분이 사라지고 flag 값 일부가 보였다.



**[사진 9] Flag값 첫 번째 부분**

Flag 값 첫 번째 부분을 찾았다!

5. 푸터 시그니처 검색의 두 번째 결과를 살펴보았다.

```
000012C0  28 A2 8A 00 FF D9 46 69 6E 64 00 74 68 65 00 53  (<Š.ÿFind.the.S
000012D0  74 72 69 6E 67 00 22 5F 45 4E 63 5F 45 43 72 79  tring."_ENC_ECry
000012E0  70 74 22 00 00 00 00 00 00 00 00 89 50 4E 47 0D  pt".....%PNG.
000012F0  0A 1A 0A 00 00 00 0D 49 48 44 52 00 00 01 B7 00  .....IHDR....
```

[사진 10] 푸터 시그니처 FF D9 검색 결과(2)

Flag 값의 중간 부분은 TXT 와 관련 있다고 하였는데, FF D9 이후의 텍스트가 눈에 띈다.

Find the String 이후의 "\_ENC\_ECrypt"가 Flag 값의 중간 부분이다.

[WHS-2] .iso

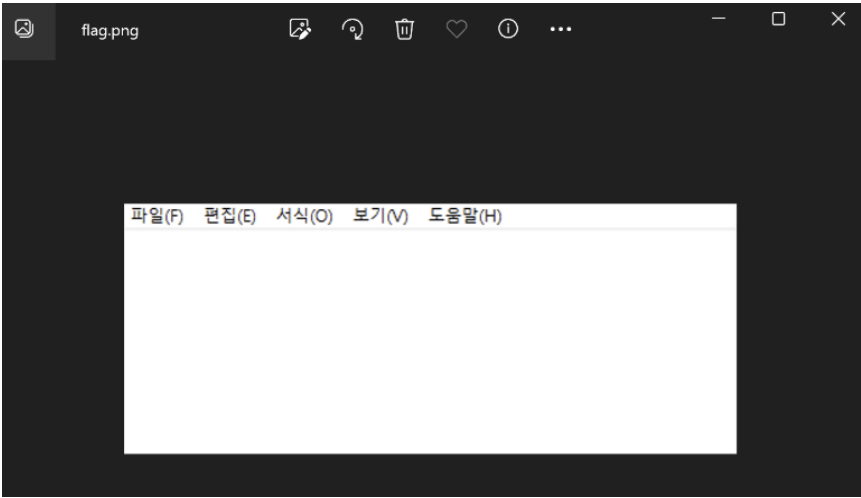
6. Flag 값 마지막은 PNG 와 연관되어 있다.

```
89 50 4E 47 0D 0A 1A 0A          %PNG....  
PNG Portable Network Graphics file  
Trailer: 49 45 4E 44 AE 42 60 82 (IENDⓈB`,...)
```

[사진 11] Png파일의 헤더 시그니처, 푸터 시그니처 값([File Signatures \(garykessler.net\)](#))

PNG 파일의 시그니처 값을 보니, flag.jpg 파일의 푸터 시그니처가 png 파일의 푸터 시그니처였다.  
또한, 두 번째 플래그 값을 찾으려 검색했던 jpg 파일의 푸터 시그니처 이후로 PNG 파일의 헤더 시그니처가 존재한다는 것을 알 수 있다. Flag.jpg 파일은 jpg+png 가 결합된 파일이었던 것이다.

마지막 flag 값을 찾기 위해 PNG 헤더 부분부터 푸터 시그니처까지 선택하여 flag.png 로 새로 저장했다.  
Flag.png 파일을 열어보았지만, 아무것도 적혀 있지 않았다.

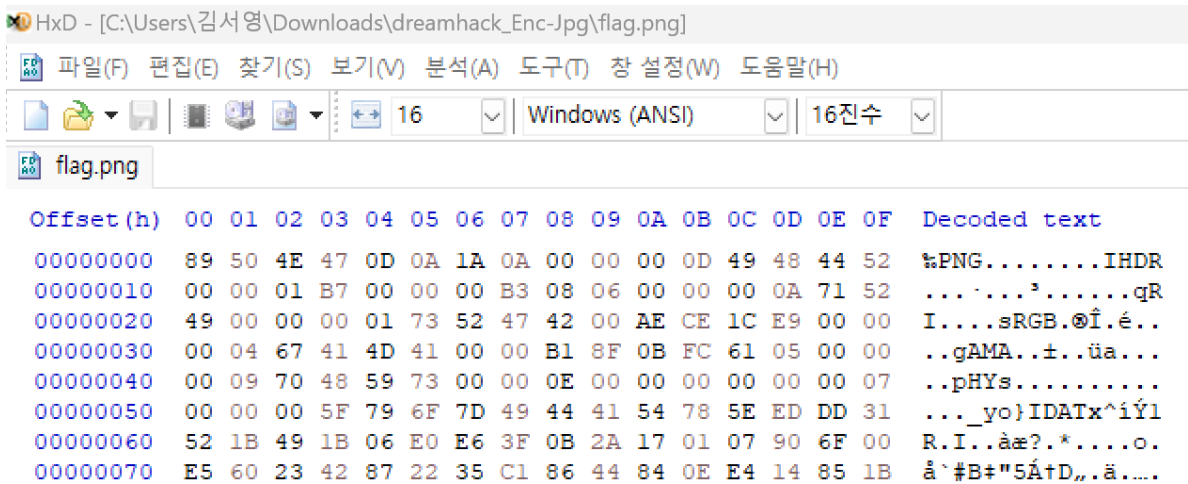


[사진 12] flag.png 파일

Hex Editor 를 통해 다시 한 번 flag.png 파일을 살펴보았다.



[WHS-2] .iso



[사진 13] flag.png파일을 HxD로 본 결과

천천히 살펴보던 중 IDAT 앞에 4 바이트가 수상하다는 것을 발견했다.

IDAT 는 PNG 파일의 이미지 데이터가 들어가는 부분으로, 앞 4 바이트는 IDAT 의 크기를 뜻하는데, 문자열이 있다.

형태 또한 flag 값의 마지막 부분과 일치하다. Flag 값의 마지막은 \_yo}이다.

# 5. Flag

Flag = DH{How\_ENc\_ECrypt\_ yo}



## 6. 별도 첨부

## 7. Reference

- [\[Dreamhack\] Forensics - Enc-JPG \(tistory.com\)](#)