

## [Suninatas 28] Write-Up

작성자	김서영
분석 일자	2024.05.10.
작성 일자	2024.05.11.
분석 대상	So_simple.zip
문서 버전	1
작성자 E-mail	sykim1802@naver.com

0. 목차

1. 문제 .....3

2. 분석 도구 .....3

3. 환경 .....3


4. Write-Up.....4

5. Flag.....7

6. 별도 첨부 .....8

7. Reference .....8

### 1. 문제

URL	<a href="#">Game 28 (suninatas.com)</a>
문제 내용	<p>암호를 풀어야 뭘 볼수 있지! 암호가 있거나 한건가!</p> <p>이 문제를 낸 사람은 너무 쉬워서 사람들이 빨리 풀지 않을까 걱정하다가 시름시름 앓고 있다는 전설이 있다.</p>
문제 파일	<div>  </div> <p>So_Simple.zip</p>
문제 유형	forensics
난이도	2 / 5

### 2. 분석 도구

도구명	다운로드 링크	Version
HxD	<a href="#">Downloads   mh-nexus</a>	2.5.0.0

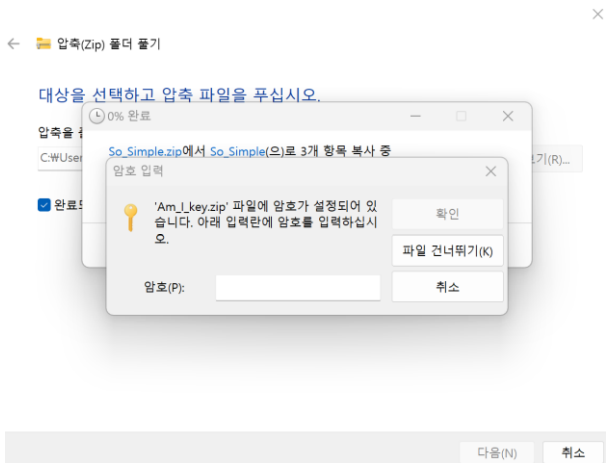
### 3. 환경

OS
Windows11 x64

4. Write-Up

파일명	So_simple.zip
용량	1kb
SHA256	071d27158aefeff97f1c3b3b1644071c0e3acb56a136e2e39bb8835c7413c720
Timestamp	2024.05.04. 23:29:38

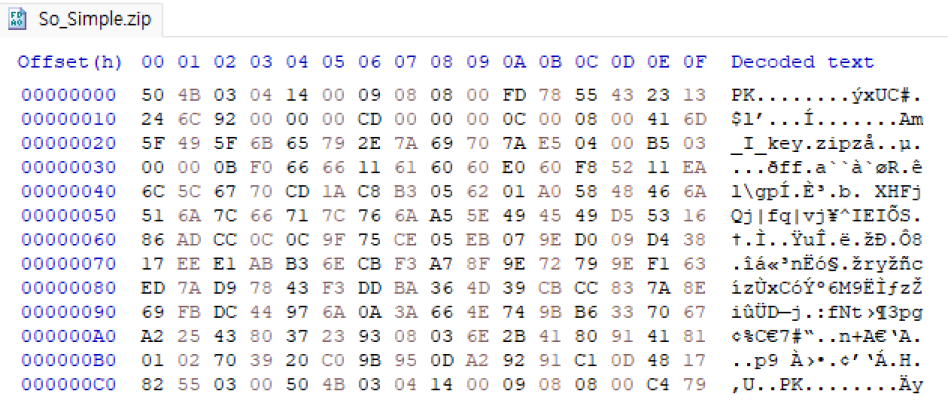
1. 문제를 다운로드 받은 후, 열어보았다. 암호가 설정되어 있다.



[사진 1] 문제 파일 압축 풀기 시 암호설정

하지만 문제 내용에서 “암호가 있거나 한건가!” 가 명시해주듯이 암호가 없다면, zip구조를 분석하여 암호화된 방법을 파악해야 할 것으로 방향을 잡았다.

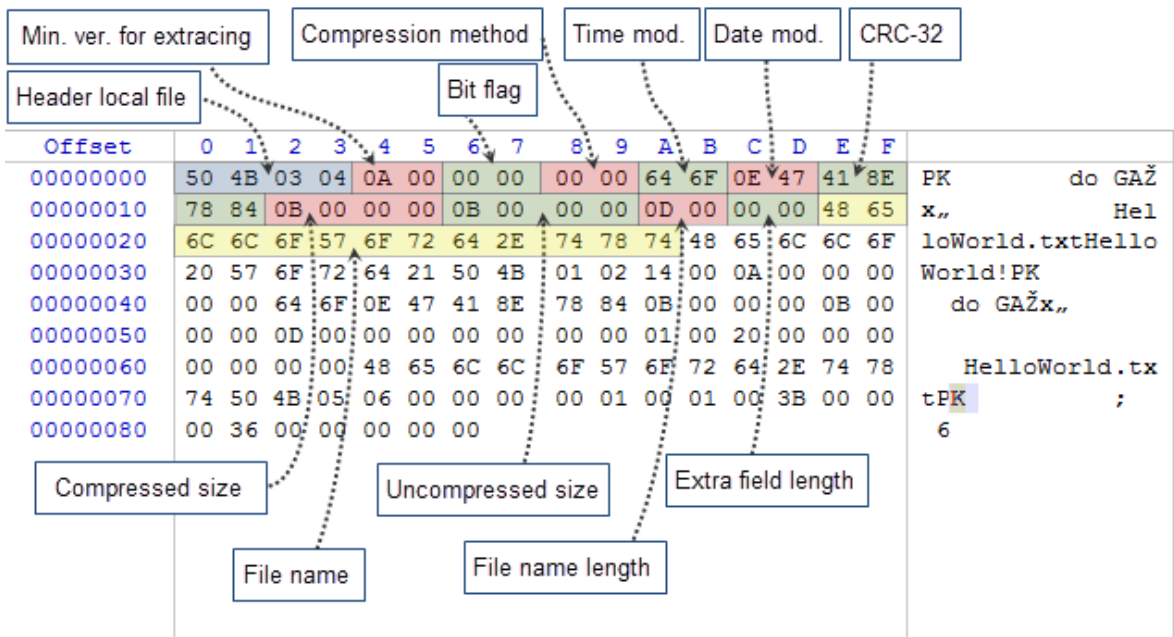
HxD로 zip파일을 열어보았다.



[사진 2] HxD로 So\_simple.zip을 열어본 결과

[WHS-2] .iso

2. So\_simple.zip파일의 헤더 구조를 더 자세히 살펴보기 위해, zip파일 구조에 대해 찾아보았다.



[사진 3] zip파일 헤더 구조([Handling ZIP Archives in Pure MQL5 - MQL5 Articles](#))

[사진 3]을 바탕으로 So\_simple.zip의 Bit Flag값은 09 08이다.

이를 bit값으로 보면 00001001 00001000이다.

Hex value:

0x14000908

1	4	0	0	0	9	0	8
0 0 0 1	0 1 0 0	0 0 0 0	0 0 0 0	0 0 0 0	1 0 0 1	0 0 0 0	1 0 0 0
0 00101000				00000000000100100001000			

[사진 4] Float to hex convertor 사용하여 계산 결과

이를 리틀엔디안 방식에 따라 00001000 00001001로 보았을 때,  
오른쪽부터 0번째, 3번째, 11번째 비트에 값이 존재한다.

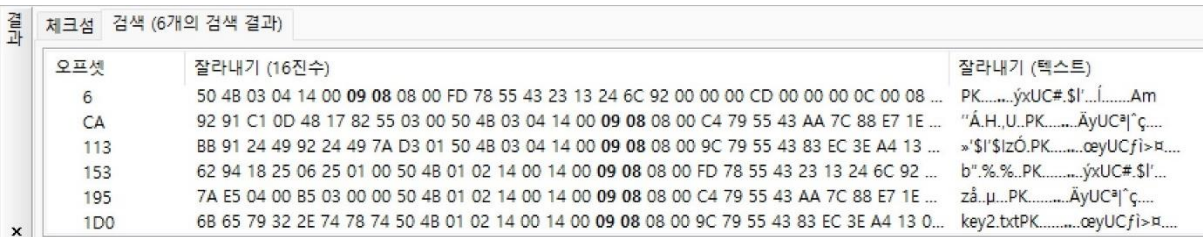
[WHS-2] .iso

Flags	General purpose bit flag:
	Bit 00: encrypted file
	Bit 01: compression option
	Bit 02: compression option
	Bit 03: data descriptor
	Bit 04: enhanced deflation
	Bit 05: compressed patched data
	Bit 06: strong encryption
	Bit 07-10: unused
	Bit 11: language encoding
	Bit 12: reserved
	Bit 13: mask header values
	Bit 14-15: reserved

[사진 5] bit값 별 특성([The structure of a PKZip file \(jmu.edu\)](#))

0번째 비트가 암호화된 파일을 뜻하기 때문에 이 값을 0으로 변경해주면 암호화가 해결될 것으로 예상한다. 그렇다면 비트는 00001000 00001000이 되고 이는 16진수값으로 08 08이 된다.

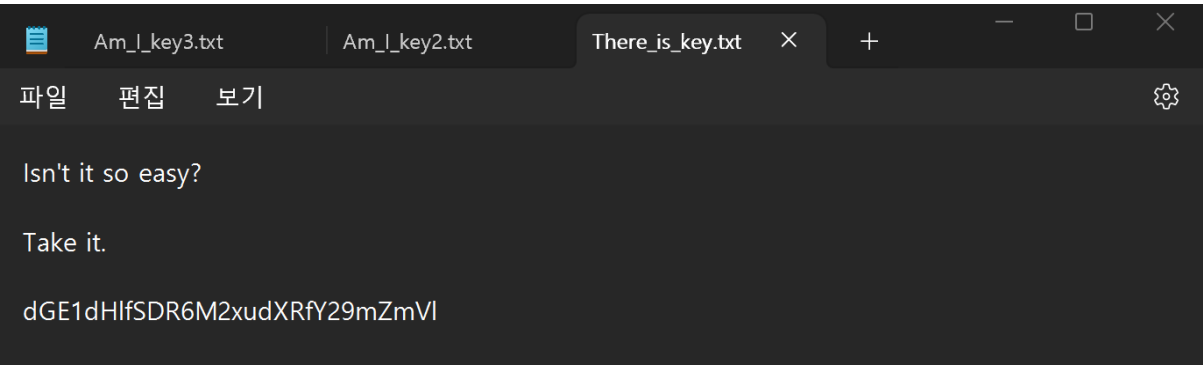
3. 이제 모든 파일의 로컬 파일 헤더에서 flag 필드를 09 08에서 08 08로 바꿔주면 된다.



오프셋	잘라내기 (16진수)	잘라내기 (텍스트)
6	50 4B 03 04 14 00 09 08 08 00 FD 78 55 43 23 13 24 6C 92 00 00 00 CD 00 00 00 0C 00 08 ...	PK.....ýxUC#.\$!'.í.....Am
CA	92 91 C1 0D 48 17 82 55 03 00 50 4B 03 04 14 00 09 08 08 00 C4 79 55 43 AA 7C 88 E7 1E ...	"Á.H.,U..PK.....ÄyUC# "ç....
113	BB 91 24 49 92 24 49 7A D3 01 50 4B 03 04 14 00 09 08 08 00 9C 79 55 43 83 EC 3E A4 13 ...	»'\$!'\$izÖ.PK.....øeyUCfi>¤....
153	62 94 18 25 06 25 01 00 50 4B 01 02 14 00 14 00 09 08 08 00 FD 78 55 43 23 13 24 6C 92 ...	b".%%.PK.....ýxUC#.\$!'....
195	7A E5 04 00 85 03 00 00 50 4B 01 02 14 00 14 00 09 08 08 00 C4 79 55 43 AA 7C 88 E7 1E ...	zâ..µ...PK.....ÄyUC# "ç....
1D0	6B 65 79 32 2E 74 78 74 50 4B 01 02 14 00 14 00 09 08 08 00 9C 79 55 43 83 EC 3E A4 13 0...	key2.txtPK.....øeyUCfi>¤....

[사진 6] HxD에서 Flag 필드 검색 결과

모두 08 08로 바꿔주면, 암호화가 사라져 zip파일 압축을 풀 수 있다.

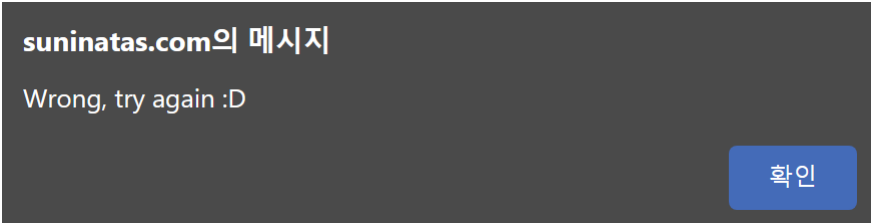


[사진 7] zip파일 내부 txt파일들

텍스트 파일들도 모두 암호화 해제되었고, key가 담겨 있는 파일을 볼 수 있다.

[WHS-2] .iso

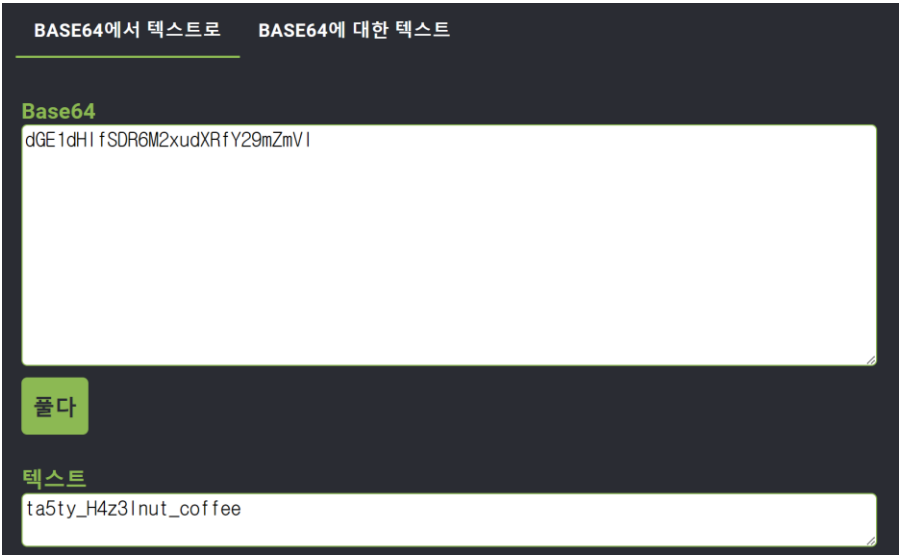
4. key값을 다 찾았다 생각하여 써니나타스 사이트에 입력했다.



[사진 8] key 찾기 실패 창

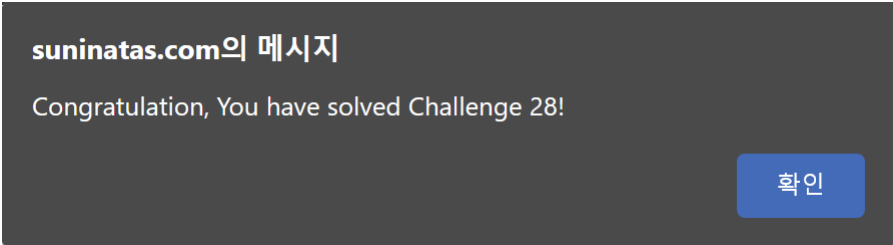
아직 한 단계가 더 남았다는 것을 알았다.

5. key값을 base64로 decode하자 "ta5ty\_H4z3lnut\_coffee"값이 나왔다.



[사진 9] Base64 decoder

이를 다시 입력해서 문제를 풀어냈다.



[사진 10] key 찾기 성공 창

## 5. Flag

Flag = ta5ty\_H4z3lnut\_coffee

## 6. 별도 첨부

## 7. Reference

- [Handling ZIP Archives in Pure MQL5 - MQL5 Articles](#)
- [The structure of a PKZip file \(jmu.edu\)](#)
- [Base64 디코드 \(magictool.ai\)](#)