

Writeup du challenge bjICS Tragedy: Suspicious file 1 de la phase de qualification pour le Hackerlab 2024

Nom d'utilisateur : takeoff

E-mail : robertohoungbo@gmail.com

Affiliation : IFRI

Pays : Bénin

bjICS Tragedy: Suspicious file 1

300

OSINT


[FR]

Un membre du groupe ICS a été arrêté et avait en sa possession une clé USB contenant un fichier suspect. Nous pensons que ce fichier pourrait fournir des informations sur les divers systèmes que le groupe a pu compromettre.

[EN]

A member of the ICS group was apprehended, and in their possession was a USB containing a suspicious file. We believe this file could provide insights into the various systems the group may have compromised.

Author: **r3s0lv3r**

 suspicious.png

Flag

Submit

Le challenge est accompagné d'un fichier nommé suspicious.png. Comme le dit l'énoncé le fichier était présent sur une clé usb d'un membre du groupe ICS et pourrait contenir des informations sur les divers systèmes que ce groupe a pu compromettre.

Je télécharge alors le fichier et à première vue c'est un fichier png valide. Je fais ensuite la commande zsteg sur le fichier et je découvre qu'il contient un fichier zip.

```
roberto@roberto-HP-Pavilion-x360-Convertible:~/Téléchargements$ zsteg suspicious.png
[?] 95368 bytes of extra data after image end (IEND), offset = 0xdb9fe
extradata:0 .. file: Zip archive data, at least v1.0 to extract, compression method=store
00000000: 50 4b 03 04 0a 00 00 00 00 00 af 29 bd 58 00 00 |PK.....).X..|
00000010: 00 00 00 00 00 00 00 00 00 00 06 00 1c 00 63 68 |.....ch|
00000020: 65 63 6b 2f 55 54 09 00 03 69 ab 56 66 6c ab 56 |eck/UT...i.VfL.V|
00000030: 66 75 78 0b 00 01 04 e8 03 00 00 04 e8 03 00 00 |fux.....|
00000040: 50 4b 03 04 14 00 09 00 08 00 d3 26 bd 58 b0 a7 |PK.....&.X..|
00000050: b9 d1 67 00 00 00 60 00 00 00 13 00 1c 00 63 68 |.g...`.....ch|
00000060: 65 63 6b 2f 70 69 65 63 65 5f 32 30 34 2e 70 6e |eck/piece_204.pn|
00000070: 67 55 54 09 00 03 fe a6 56 66 da aa 56 66 75 78 |gUT....Vf..Vfux|
00000080: 0b 00 01 04 e8 03 00 00 04 e8 03 00 00 ed bd 1c |.....|
00000090: 39 02 50 d1 f1 e9 6f 77 69 f0 c3 a4 7e b8 98 52 |9.P...owi...~.R|
000000a0: a6 5c d6 63 73 18 46 86 28 41 91 84 66 d2 bf 06 |.\.cs.F.(A..f...|
000000b0: a2 71 94 c6 3b ad 65 35 0a 22 0a 5b 85 7d 8d fa |.q...;e5."[.].|
000000c0: 02 05 78 79 36 5c ce 3d 40 2e 60 73 b3 4f 48 3b |..xy6\.=@.`s.OH;|
000000d0: 7c fa b2 41 ce 07 af 6c 8d d5 b8 42 5e 23 b8 49 ||.A...l...B^#.I|
000000e0: 2d 24 5f 25 42 98 74 6b b0 ab 49 62 d7 f9 46 9e |-$_%B.tk..Ib..F.|
000000f0: db 6d c5 78 50 4b 07 08 b0 a7 b9 d1 67 00 00 00 |.m.xPK.....g...|
meta_data:create .. text: "2024-05-29T03:48:08+00:00"
meta_data:modify .. text: "2024-05-29T03:48:07+00:00"
meta_data:timestamp .. text: [same as "meta_data:create"]
unspdata .. file: PDP-11 UNIX/RT ldp
01.r.lab.xy .. file: VISX image file
01.g.lab.xy .. text: ["0" repeated 10 times]
02.g.lab.xy .. text: ["0" repeated 12 times]
01.B.pdb.xy .. text: ["0" repeated 8 times]
02.B.pdb.xy .. text: ["0" repeated 8 times]
```

J'extrais ensuite le fichier zip du png avec binwalk comme suit :

```
roberto@roberto-HP-Pavilion-x360-Convertible:~/Téléchargements$ binwalk --dd='.*' suspicious.png
```

DECIMAL	HEXADECIMAL	DESCRIPTION
0	0x0	PNG image, 1715 x 980, 8-bit/color RGB, non-interlaced
287	0x11F	TIFF image data, big-endian, offset of first image directory: 8
389	0x185	Zlib compressed data, best compression
899582	0xDB9FE	Zip archive data, at least v1.0 to extract, name: check/
899646	0xDBA3E	Zip archive data, encrypted at least v2.0 to extract, compressed size: 103, uncompressed size: 96,
check/piece_204.png		
899842	0xDBB02	Zip archive data, encrypted at least v2.0 to extract, compressed size: 98, uncompressed size: 92, r
heck/piece_52.png		
900032	0xDBBC0	Zip archive data, encrypted at least v2.0 to extract, compressed size: 101, uncompressed size: 95,
check/piece_295.png		
900226	0xDBC82	Zip archive data, encrypted at least v2.0 to extract, compressed size: 99, uncompressed size: 95, r
heck/piece_314.png		
900418	0xDBD42	Zip archive data, encrypted at least v2.0 to extract, compressed size: 100, uncompressed size: 94,
check/piece_325.png		
900611	0xDBE03	Zip archive data, encrypted at least v2.0 to extract, compressed size: 101, uncompressed size: 94,
check/piece_245.png		
900805	0xDBEC5	Zip archive data, encrypted at least v2.0 to extract, compressed size: 96, uncompressed size: 90, r
heck/piece_55.png		
900993	0xDBF81	Zip archive data, encrypted at least v2.0 to extract, compressed size: 100, uncompressed size: 93,
check/piece_94.png		
901185	0xDC041	Zip archive data, encrypted at least v2.0 to extract, compressed size: 103, uncompressed size: 96,
check/piece_207.png		
901381	0xDC105	Zip archive data, encrypted at least v2.0 to extract, compressed size: 101, uncompressed size: 94,
check/piece_66.png		

A ma grande surprise, le fichier zip contient un grand nombre de fichiers png. Après l'extraction, j'essaye la commande unzip sur le fichier zip en question mais je me rends compte qu'il est protégé par un mot de passe. Je fais alors appel à mon outil de prédilection pour le crack de fichiers zip protégés par mot de passe, en l'occurrence john the ripper. Et bim j'ai pu avoir le mot de passe qui était souljaboytellem (de la liste rockyou.txt).

J'ai ensuite dézippé le fichier et j'ai eu comme résultat un répertoire check qui contenant les fichiers png vus précédemment.

Après inspection je me rend compte que ce sont des parties d'un code qr qui a été divisé en plusieurs petits morceaux.



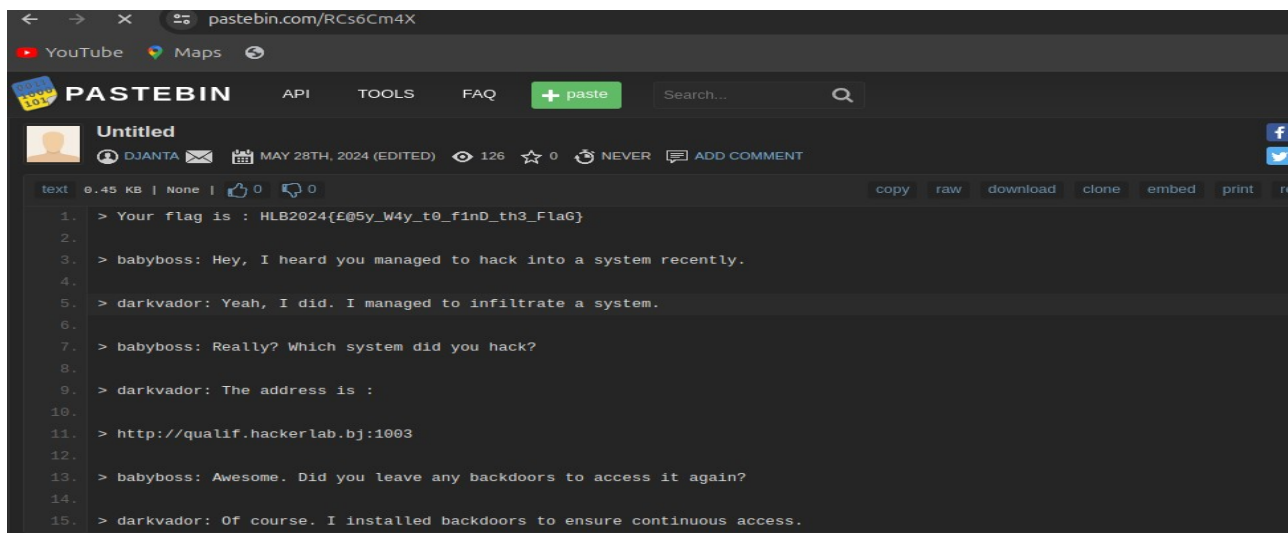
L'objectif maintenant est donc de rassembler ces petits morceaux dans le bon ordre afin de retrouver le code qr d'origine. Pour cela j'ai d'abord essayé de déterminer l'intervalle de découpage qui correspond à une ligne du code qr et j'ai vite remarqué que une ligne du code était divisé en 20 morceaux, par exemple pour la première ligne du code qr, pour le rassembler il faut concatener

les fichiers de piece_0 à piece_19.

J'ai alors utilisé la bibliothèque python cv2 pour écrire le code de récupération suivant :



Voilà le qr code que j'obtiens, après scannage je suis redirigé vers un compte pastebin où le flag est écrit en clair.

A screenshot of a web browser displaying a Pastebin page. The browser's address bar shows 'pastebin.com/RCS6Cm4X'. The page header includes the Pastebin logo, navigation links (API, TOOLS, FAQ), a '+ paste' button, and a search bar. The post is titled 'Untitled' and is by user 'DJANTA', dated 'MAY 28TH, 2024 (EDITED)'. It has 126 views and 0 stars. The content is a text-based conversation between two users, 'babyboss' and 'darkvador'. The conversation starts with 'darkvador' revealing a flag, followed by 'babyboss' asking for details, and 'darkvador' providing a URL and confirming the presence of backdoors. The text is displayed in a dark-themed code editor with line numbers from 1 to 15.

```
1. > Your flag is : HLB2024{£@5y_W4y_t0_f1nD_th3_FlaG}
2.
3. > babyboss: Hey, I heard you managed to hack into a system recently.
4.
5. > darkvador: Yeah, I did. I managed to infiltrate a system.
6.
7. > babyboss: Really? Which system did you hack?
8.
9. > darkvador: The address is :
10.
11. > http://qualif.hackerlab.bj:1003
12.
13. > babyboss: Awesome. Did you leave any backdoors to access it again?
14.
15. > darkvador: Of course. I installed backdoors to ensure continuous access.
```

FLAG: HLB2024{£@5y_W4y_t0_f1nD_th3_FlaG}

Merci !