Descripción

Nombre: 0gDrU_J4h4D (https://twitter.com/Hackers4F/status/1109010112172175360)
Related: Hellboy (https://es.wikipedia.org/wiki/Hellboy_(pel%C3%ADcula))
Fecha de liberación: 22 de marzo de 2019
Autor: 1v4n

Un agente del #BPRD se enfrenta a nueva amenaza del #Infierno 🔥 y nada amigable 😈. Han detectado una intrusión y les han dejado una foto a modo de aviso 😮 👉 Nos ayudas a investigarlo

Objetivo
Formato de flag: H4F{string}

Herramientas utilizadas
Firefox V. 60.6.1 https://www.mozilla.org/en-US/firefox/60.6.1/releasenotes/
gdown 3.7.4 https://pypi.org/project/gdown/
7-Zip [64] 16.02 https://www.7-zip.org/download.html
Exiftool 11.16 https://www.sno.phy.queensu.ca/~phil/exiftool/
zsteg https://github.com/zed-0xff/zsteg
EasyBMP http://easybmp.sourceforge.net/steganography.html
Steganography https://github.com/GarrettBeatty/Steganography // http://encode.gbt.codes/
Malbolgue Interpreter https://zb3.me/malbolge-tools/

Resumen:

Visitamos el tuit publicado y descargamos el archivo comprimido *R3t0_20_H4F_0gDrU_J4h4D.7z* con hash MD5 *b7c83477eff82c974b4fbe60997b04df* que contendrá el archivo de la "intrusión"


```
root@1v4n:~/CTF/Hackers4Fun/Reto20/Recursos2# gdown
https://drive.google.com/uc?id=1QeCF5lnu5MZ8e-jevis8MDhw6W0H08vT
Downloading...
From: https://drive.google.com/uc?id=1QeCF5lnu5MZ8e-jevis8MDhw6W0H08vT
To: /root/CTF/Hackers4Fun/Reto20/Recursos2/R3t0_20_H4F_0gDrU_J4h4D.7z
100%|████████████████████████████████████████| 1.84M/1.84M [00:00<00:00, 4.80MB/s]
```

Al descomprimirlo obtenemos un archivo de imagen *R3t0_20_H4F_0gDrU_J4h4D.bmp* y con hash MD5 *3f3df69ea75e556b525dbc797cf1b78b*. Desconociendo lo que esconde por el momento no debemos descartar realizar un análisis forense con *exiftool* sin arrojar nada significativo.

```
root@1v4n:~/CTF/Hackers4Fun/Reto20# exiftool R3t0_20_H4F_0gDrU_J4h4D.bmp
ExifTool Version Number         : 11.16
File Name                       : R3t0_20_H4F_0gDrU_J4h4D.bmp
Directory                       : .
File Size                       : 3.7 MB
File Modification Date/Time     : 2019:03:21 19:14:47-04:00
File Access Date/Time           :
File Inode Change Date/Time     :
File Permissions                : rw-r--r--
File Type                       : BMP
File Type Extension             : bmp
MIME Type                       : image/bmp
BMP Version                     : Windows V3
Image Width                     : 1200
Image Height                    : 800
Planes                          : 1
Bit Depth                       : 32
Compression                     : None
Image Length                    : 3840000
Pixels Per Meter X              : 0
Pixels Per Meter Y              : 0
Num Colors                      : Use BitDepth
Num Important Colors            : All
Image Size                      : 1200x800
Megapixels                      : 0.960
```

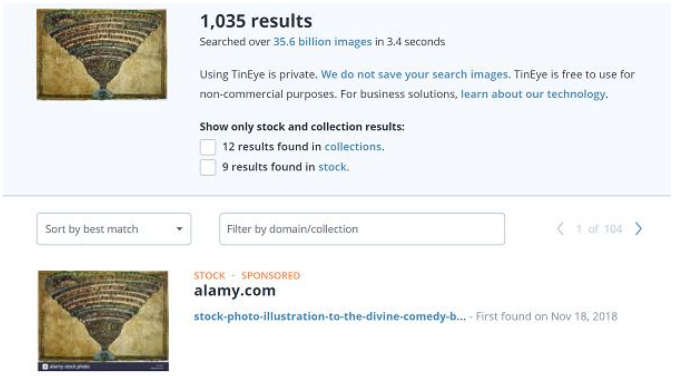
Sin descartar métodos de ocultación realizamos un estegoanálisis con *zsteg -a R3t0_20_H4F_0gDrU_J4h4D.bmp* donde podemos detectar una posible esteganografía.

```
root@1v4n:~/CTF/Hackers4Fun/Reto20# zsteg -a R3t0_20_H4F_0gDrU_J4h4D.bmp
b1,msb,bY           .. text: "w4F3dUpAG3A\"%"
b4,lsb,bY           .. text: "`#`Vpf`T0C"
b1,rgb,msb,xY       .. text: "b[TQUu}G"
b1,rgba,msb,xY      .. text: "w`3G0Gr@cP"
b1,abgr,lsb,xY      .. text: "e!fbBPRRVVv"
b1,abgr,msb,xY      .. text: "fFB\nJJjjnH"
b2,b,lsb,xY         .. text: ")0,lh$OA"
b2,rgb,msb,xY       .. text: "sawV#srv."
b2,abgr,lsb,xY      .. text: "\"-8\r&;;;;&"
b4,b,lsb,xY         .. text: "\n^cFwweB"
b4,rgb,lsb,xY       .. file: PGP\011Secret Sub-key -
b4,rgb,msb,xY       .. text: "N&hLnjf,\n,O\nljn"
b4,bgr,lsb,xY       .. text: "b6VvfT40R"
b4,bgr,msb,xY       .. text: "(Fljnf*,"
b4,rgba,lsb,xY      .. file: PGP\011Secret Sub-key -
b5,b,msb,xY         .. text: "\tA1\"R+A)"
b8,r,lsb,xY         .. text: "+>R[XTQNMMPRSSRRSVZ]"
b8,g,lsb,xY         .. text: "'&$$&),--,)(+/5743664323650'"
...
```

Liberado en el hilo del reto el tuit con un Hint >_ https://twitter.com/Hackers4F/status/1109119294112235520 nos desvela la categoría de Esteganografía y un posible método de ocultación en el archivos BMP. Pasamos a detectar y extraer la información obteniendo *Inferno_stego.jpg* posible imagen JPG con hash MD5 a5e585b9a651855ca6e506e95901c600


```
root@1v4n:~/CTF/Hackers4Fun/Reto20# wine ~/Stego/EasyBMP/Stego.exe -d
R3t0_20_H4F_0gDrU_J4h4D.bmp
...
Hidden data detected! Outputting to file Inferno_stego.7z ...
root@1v4n:~/CTF/Hackers4Fun/Reto20# 7z x Inferno_stego.7z
...
root@1v4n:~/CTF/Hackers4Fun/Reto20# file Inferno_stego.jpg
Inferno_stego.jpg: PNG image data, 160 x 112, 8-bit/color RGB, non-interlaced
root@1v4n:~/CTF/Hackers4Fun/Reto20/Recursos2# md5sum Inferno_stego.jpg
a5e585b9a651855ca6e506e95901c600  Inferno_stego.jpg
```

Investigando con herramientas OSINT hacemos reversing de la imagen y vemos que está relacionada con una ilustración del “Mapa del Infierno” del artista Botticelli.



Pasamos un análisis forense a la imagen y detectamos que es una imagen con formato PNG. Realizaremos un estegoanálisis con zsteg que será positivo.

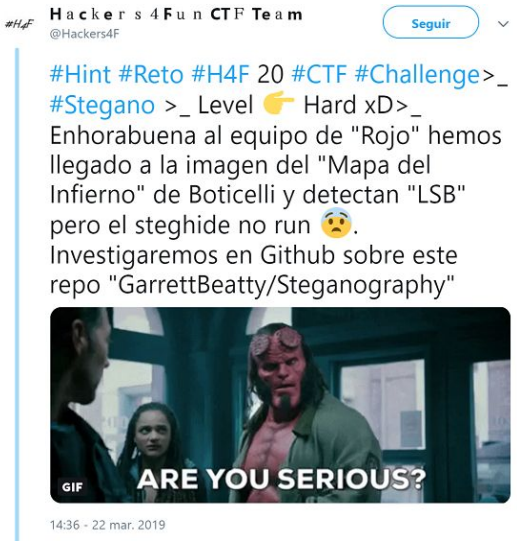
```
root@1v4n:~/CTF/Hackers4Fun/Reto20# binwalk -v Inferno_stego.jpg

Scan Time:
Target File:   /root/CTF/Hackers4Fun/Reto20/Recursos2/Inferno_stego.jpg
MD5 Checksum:  a5e585b9a651855ca6e506e95901c600
Signatures:    386

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
0             0x0             PNG image, 160 x 112, 8-bit/color RGB, non-interlaced
41            0x29            Zlib compressed data, default compression
root@1v4n:~/CTF/Hackers4Fun/Reto20/Recursos2# mv Inferno_stego.jpg Inferno_stego.png
```

```
root@1v4n:~/CTF/Hackers4Fun/Reto20# zsteg -a Inferno_stego.png
imagedata           .. text: ",+)2&-#\t\n"
b1,bgr,lsb,xy       .. text: "R7Rgkm~j"
b1,bgr,msb,xy       .. file: PDP-11 UNIX/RT ldp
b2,r,lsb,xy         .. file: PGP symmetric key encrypted data -
b2,bgr,lsb,xy       .. text: "Txl2Ni0s"
b3,b,lsb,xy         .. text: "4nt\ra93#"
b3,b,msb,xy         .. text: "! >%GpZq"
b4,rgb,msb,xy       .. text: "U8B,$dfPspt"
b4,bgr,msb,xy       .. text: "XE2$l$`vSt"
b5,rgb,msb,xy       .. text: "RCX?DY%M"
b5,bgr,lsb,xy       .. text: "G-d\"(``H"
b6,r,msb,xy         .. file: PGP\011Secret Sub-key -
...
```

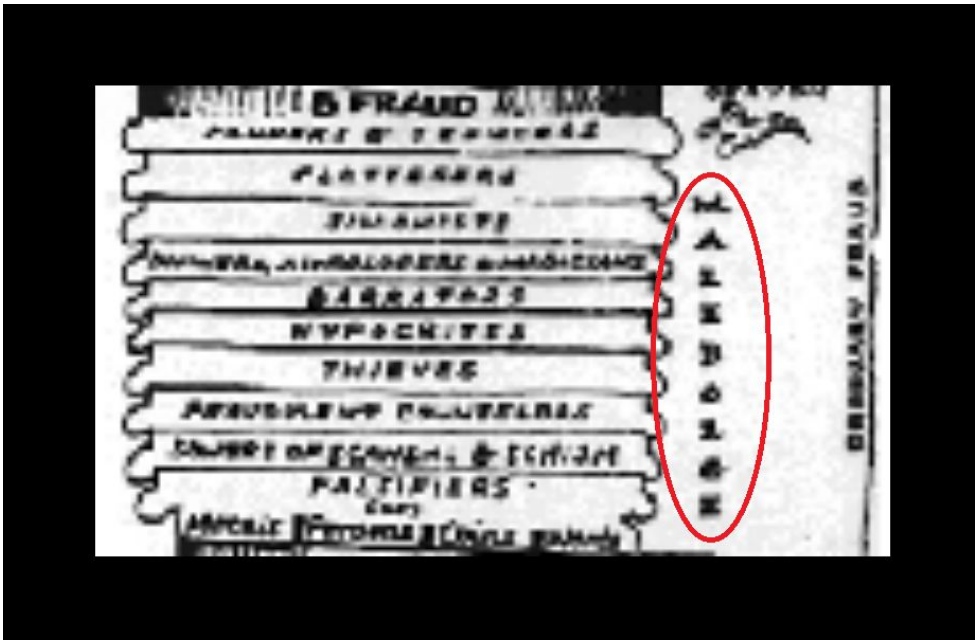
De nuevo en el hilo del reto un tuit con un Hint >_ https://twitter.com/Hackers4F/status/1109207120069775360 nos desvelan otro posible método estaganográfico en el repositorio de Github > GarrettBeatty/Steganography.



Decodificamos la imagen portadora de la esteganografía y obtenemos de nuevo otra imagen con hash MD5 *e050ab08b9745465db0e070daab417b5*

```
root@1v4n:~/CTF/Hackers4Fun/Reto20# python3 ~/Stego/Steganography3/cli.py --source
Inferno_stego.png --decode --output out --source-type image
File saved to out
root@1v4n:~/CTF/Hackers4Fun/Reto20/Recursos2# file out
out: PNG image data, 150 x 89, 8-bit grayscale, non-interlaced
root@1v4n:~/CTF/Hackers4Fun/Reto20/Recursos2# md5sum out
e050ab08b9745465db0e070daab417b5  out
root@1v4n:~/CTF/Hackers4Fun/Reto20/Recursos2# zsteg -a out
imagedata           .. text: ",277B>?6,*"
b1,r,msb,xy         .. text: "tu6gdTeP"
b2,r,msb,xy         .. text: "lIoncmblDa"
b4,rgb,lsb,xy       .. text: "U\"#3ws3fl"
b4,rgb,msb,xy       .. file: dBase IV DBT of 3\273\03.DBF, block length 8704, next free
block index 4286019447, next free block 454229251, next used block 4259904017
b6,r,lsb,xy         .. text: "u]4QL*M>"
b8,r,lsb,xy         .. text: "=8889;:9e"
b8,r,msb,xy         .. file: RDI Acoustic Doppler Current Profiler (ADCP)
b8,rgb,lsb,xy       .. text: "222ddd000XXX"
...
```

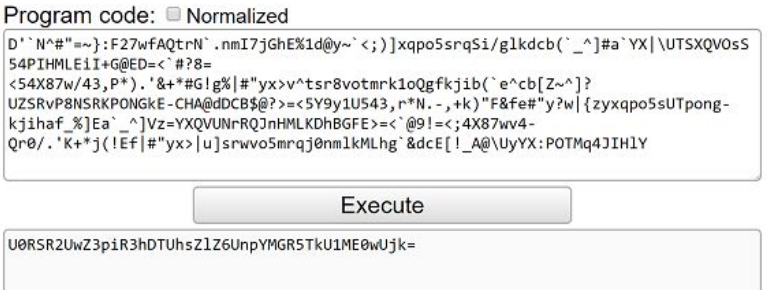
En la primera vuelta conseguimos la pista del lenguaje de programación “esotérico” llamado *MALEBOLGUE*



Se libera el último Hint >_ https://twitter.com/Hackers4F/status/1109220570493845504 y nos dan pistas para llegar a la cadena codificada en MALBOLGE.


```
root@1v4n:~/CTF/Hackers4Fun/Reto20# mv out out.png
root@1v4n:~/CTF/Hackers4Fun/Reto20# python3 ~/Stego/Steganography3/cli.py --source
out.png --decode --output out2 --source-type image
File saved to out2
root@1v4n:~/CTF/Hackers4Fun/Reto20# file out2
out2: ASCII text, with very long lines
root@1v4n:~/CTF/Hackers4Fun/Reto20# cat out2
D'`N^#"=~}:F27wfAQtrN`.nmI7jGhE%1d@y~`<;)]xqpo5srqSi/glkdcb(`_^]#a`YX|\UTSXQVOsS54PIHML
EiI+G@ED=<`#?8=<54X87w/43,P*).'&+*#G!g%|#"yx>v^tsr8votmrk1oQgfkjib(`e^cb[Z~^]?UZSRvP8NS
RKPONGkE-CHA@dDCB$@?>=<5Y9y1U543,r*N.-,+k)"F&fe#"y?w|{zyxqpo5sUTpong-kjihaf_%]Ea`_^]Vz=
YXQVUNrRQJnHMLKDhBGFE>=<`@9!=<;4X87wv4-Qr0/.'K+*j(!Ef|#"yx>|u]srwvo5mrqj0nmlkMLhg`&dcE[
!_A@\UyYX:POTMq4JIHlY
```

Utilizamos un Intérprete de Malbolge online obteniendo una cadena que la decodificamos y obtenemos la Flag.




```
root@1v4n:~/CTF/Hackers4Fun/Reto20# printf
'U0RSR2UwZ3piR3hDTUhsZlZ6UnpYMGR5TkU1ME0wUjk=' | base64 -d
SDRGe0gzbGxCMHlfVzRzX0dyNE50M0R9root@1v4n:~/CTF/Hackers4Fun/Reto20# printf
'U0RSR2UwZ3piR3hDTUhsZlZ6UnpYMGR5TkU1ME0wUjk=' | base64 -d | base64 -d
H4F{H3llB0y_W4s_Gr4Nt3D}
```

Autor: 1v4n a.k.a. @1r0Dm48O
Twitter: https://twitter.com/1r0Dm448O