Australian Government
Department of Defence
Science and Technology

My approach to solving Flare-on 2018 Challenge 3 – “Fleggo”

Kris Jorgensen

/usr/bin/whoami

- Kris Jorgensen
  - DST Group
  - CEWD - Cyber & Electronic Warfare Division
  - CWO - Cyberwarfare Operations
  - CCT - Counter Cyber Threats
- About me
  - Working at DST Group approximately 8 years
  - Previously Saab, BAE
  - Previous Projects include
    - Jindalee, Wedgetail, P3 Upgrade, Anzac Frigates

What is The Flare On Challenge and why do it?

- Made by FireEye Labs Advanced Reverse Engineering Team - http://flare-on.com
- Reverse engineering challenge once a year
- Includes challenges of increasing difficulty
- Why would we do this?
  - Interesting and challenging problems
  - Introduce various reverse engineering tasks you may encounter in real malware
  - Practice
  - Fun

Today: Challenge 3 – FLEGGO Walkthrough

- Fleggo arrives as a 7z zipped file with the password "infected"
- This unzips to a zip file which again unzips to 48 executable files all having the same size.
  - ls -l *.exe | wc –l
    48
- Running linux /usr/bin/file on random members reveals windows pe32 executables
  - file u8mbI3GZ8WtwruEiFkIl0UKxJS917407.exe
    u8mbI3GZ8WtwruEiFkIl0UKxJS917407.exe: PE32 executable (console) Intel 80386, for MS Windows
  - Time to dust off the Windows

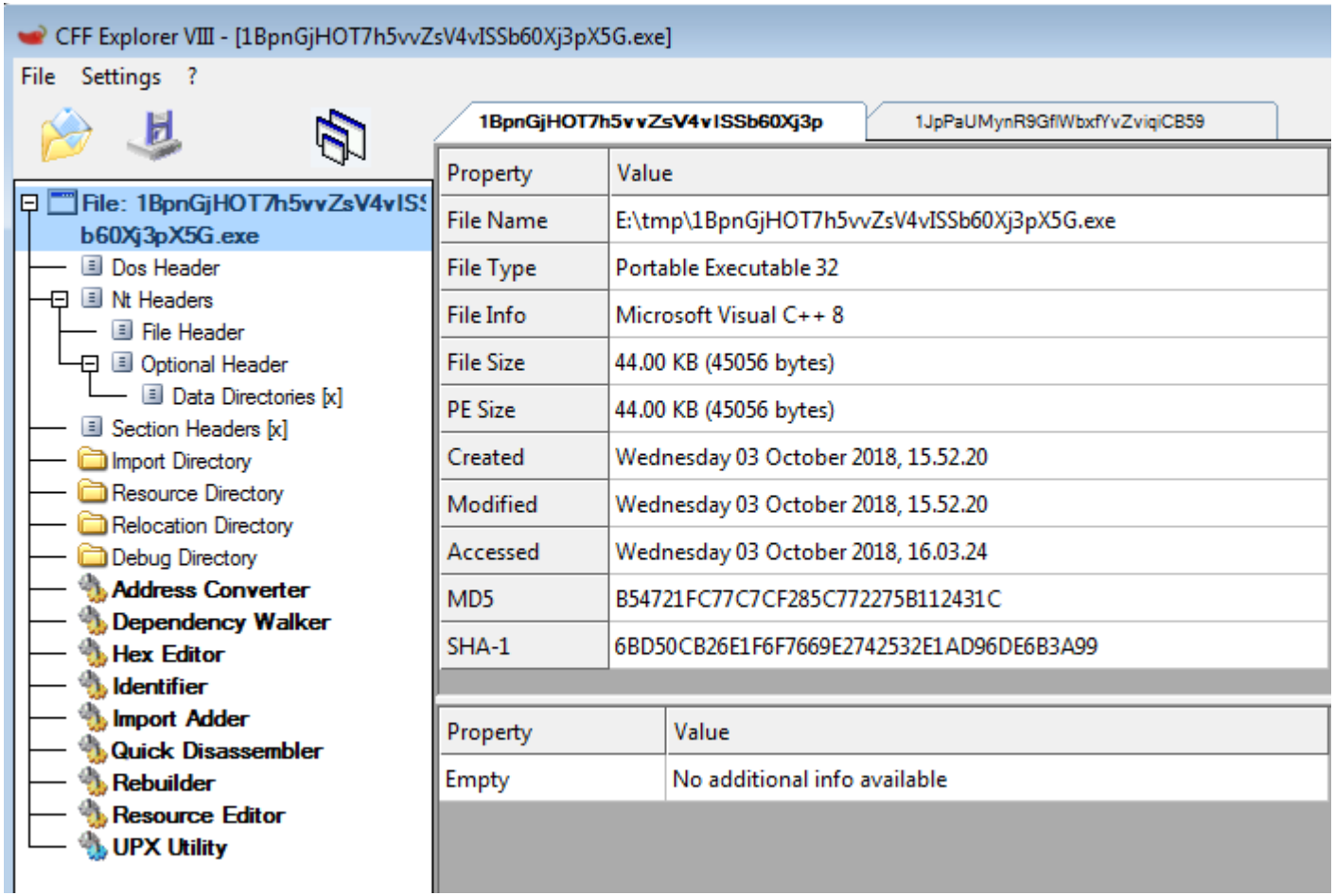
General Approach

- Look at the windows executable PE Header
- In this case diff the files due to the file sizes
- Disassemble the file
- Dynamic analysis required?
- Execute the file or script the bits out we need
- Script a solution
  - Includes scripting a debugger if required

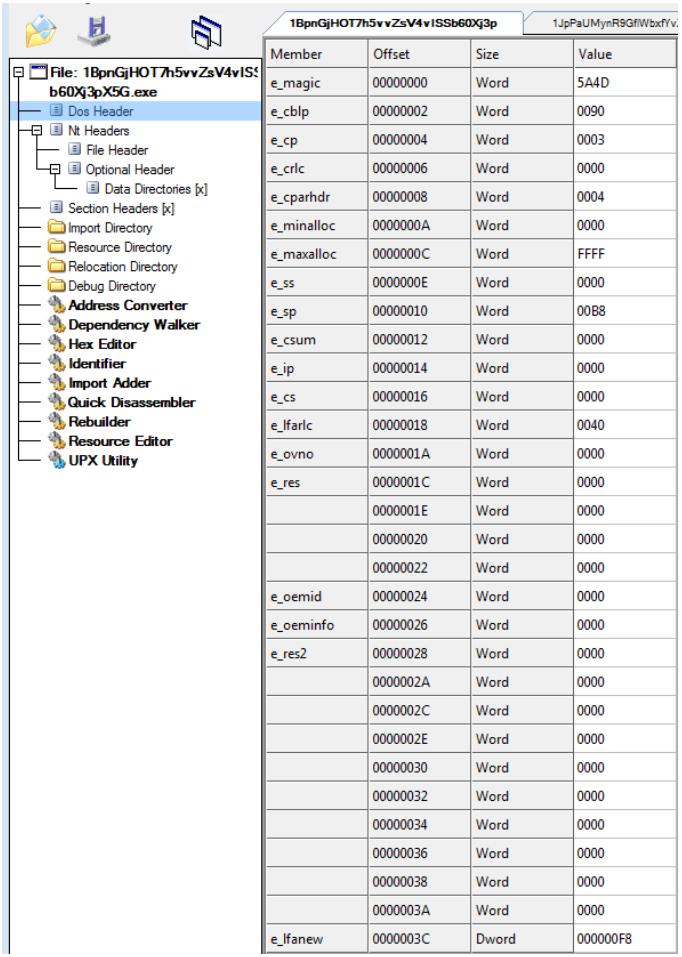
Tools used

- CFF Explorer VIII
  - PE file analysis
- IDA Pro (Free version is up to version 7 now)
  - Or disassembler of choice
- 010 Editor
  - Hex editor including binary diff functionality
- Python or C or your favourite something
  - Speed things up and act as challenge documentation
- Windows Explorer
  - Turns out to be handy to view pictures quickly in the end

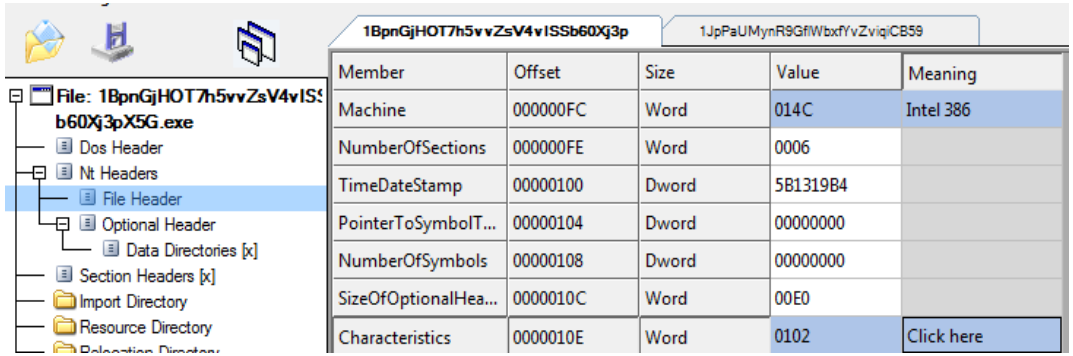
First up: CFF Explorer



CFF Explorer continued

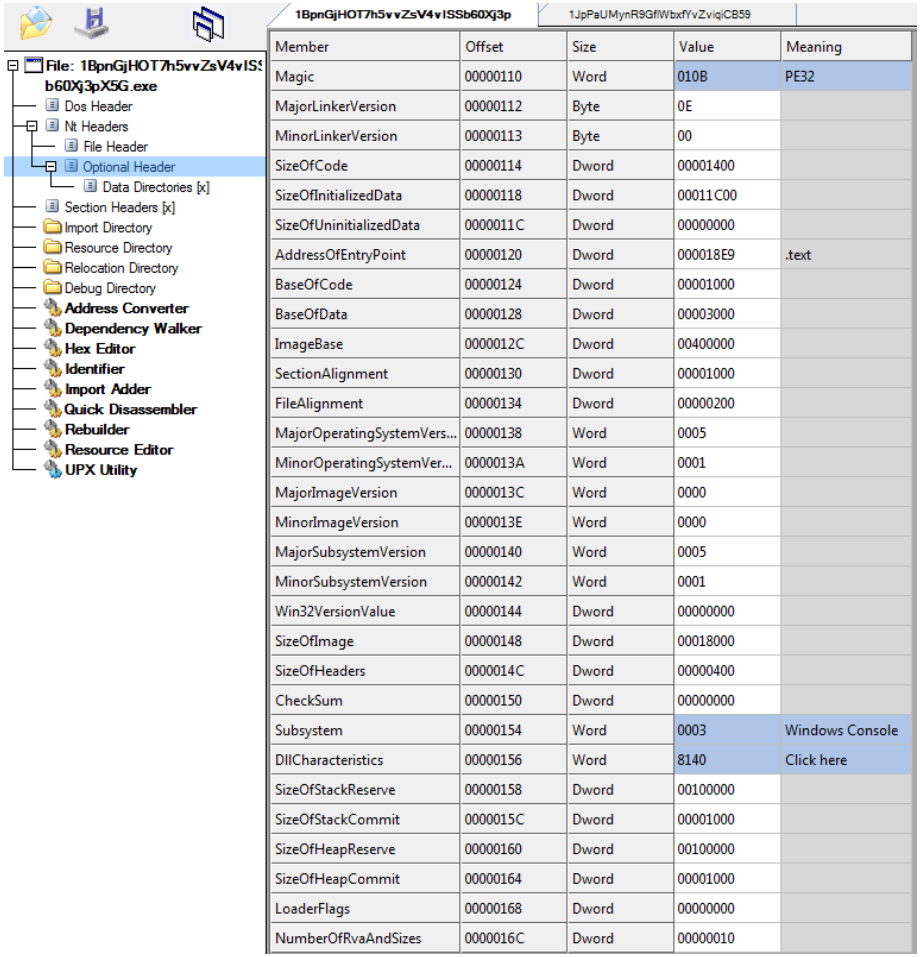


Member	Offset	Size	Value
e_magic	00000000	Word	5A4D
e_cblp	00000002	Word	0090
e_cp	00000004	Word	0003
e_crlc	00000006	Word	0000
e_cparhdr	00000008	Word	0004
e_minalloc	0000000A	Word	0000
e_maxalloc	0000000C	Word	FFFF
e_ss	0000000E	Word	0000
e_sp	00000010	Word	00B8
e_csum	00000012	Word	0000
e_ip	00000014	Word	0000
e_cs	00000016	Word	0000
e_lfarlc	00000018	Word	0040
e_ovno	0000001A	Word	0000
e_res	0000001C	Word	0000
	0000001E	Word	0000
	00000020	Word	0000
	00000022	Word	0000
e_oemid	00000024	Word	0000
e_oeminfo	00000026	Word	0000
e_res2	00000028	Word	0000
	0000002A	Word	0000
	0000002C	Word	0000
	0000002E	Word	0000
	00000030	Word	0000
	00000032	Word	0000
	00000034	Word	0000
	00000036	Word	0000
	00000038	Word	0000
	0000003A	Word	0000
e_lfanew	0000003C	Dword	000000F8

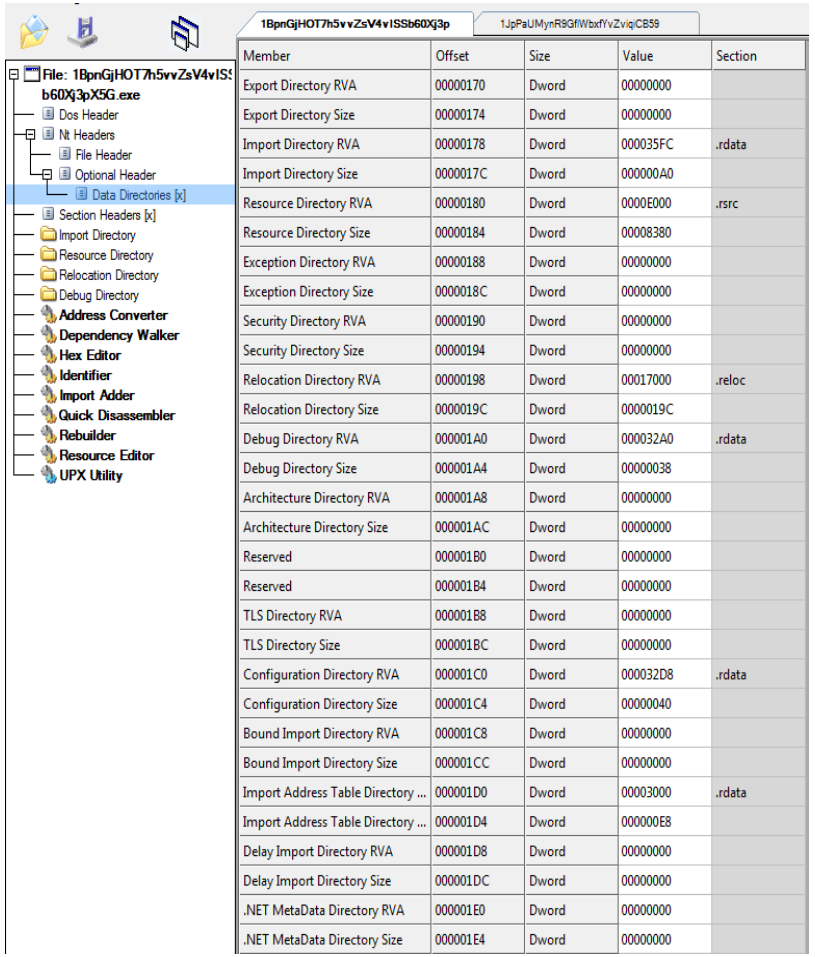


Member	Offset	Size	Value	Meaning
Machine	000000FC	Word	014C	Intel 386
NumberOfSections	000000FE	Word	0006	
TimeDateStamp	00000100	Dword	5B1319B4	
PointerToSymbolT...	00000104	Dword	00000000	
NumberOfSymbols	00000108	Dword	00000000	
SizeOfOptionalHea...	0000010C	Word	00E0	
Characteristics	0000010E	Word	0102	Click here

CFF Explorer continued

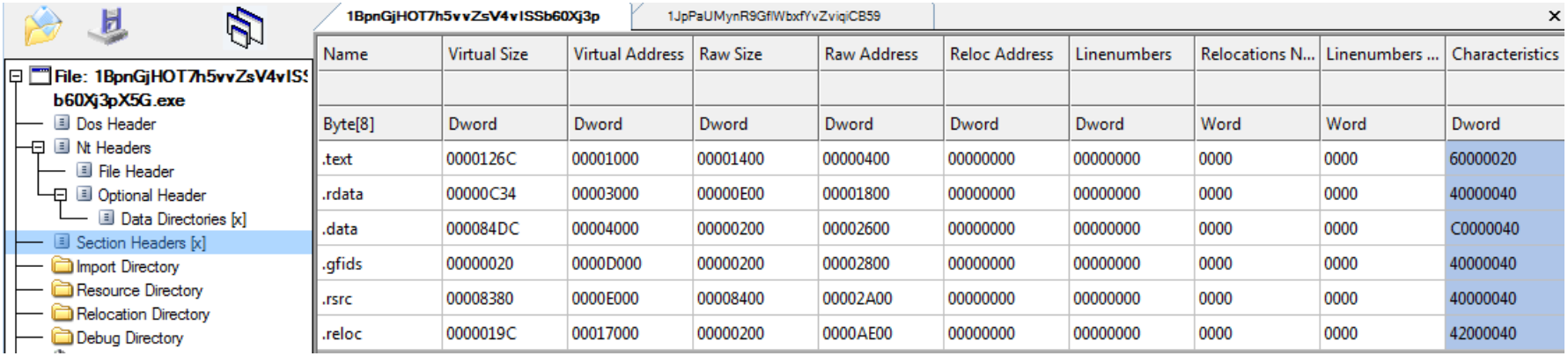


Member	Offset	Size	Value	Meaning
Magic	00000110	Word	010B	PE32
MajorLinkerVersion	00000112	Byte	0E	
MinorLinkerVersion	00000113	Byte	00	
SizeOfCode	00000114	Dword	00001400	
SizeOfInitializedData	00000118	Dword	00011C00	
SizeOfUninitializedData	0000011C	Dword	00000000	
AddressOfEntryPoint	00000120	Dword	000018E9	.text
BaseOfCode	00000124	Dword	00001000	
BaseOfData	00000128	Dword	00003000	
ImageBase	0000012C	Dword	00400000	
SectionAlignment	00000130	Dword	00001000	
FileAlignment	00000134	Dword	00000200	
MajorOperatingSystemVers...	00000138	Word	0005	
MinorOperatingSystemVer...	0000013A	Word	0001	
MajorImageVersion	0000013C	Word	0000	
MinorImageVersion	0000013E	Word	0000	
MajorSubsystemVersion	00000140	Word	0005	
MinorSubsystemVersion	00000142	Word	0001	
Win32VersionValue	00000144	Dword	00000000	
SizeOfImage	00000148	Dword	00018000	
SizeOfHeaders	0000014C	Dword	00000400	
CheckSum	00000150	Dword	00000000	
Subsystem	00000154	Word	0003	Windows Console
DllCharacteristics	00000156	Word	8140	Click here
SizeOfStackReserve	00000158	Dword	00100000	
SizeOfStackCommit	0000015C	Dword	00001000	
SizeOfHeapReserve	00000160	Dword	00100000	
SizeOfHeapCommit	00000164	Dword	00001000	
LoaderFlags	00000168	Dword	00000000	
NumberOfRvaAndSizes	0000016C	Dword	00000010	



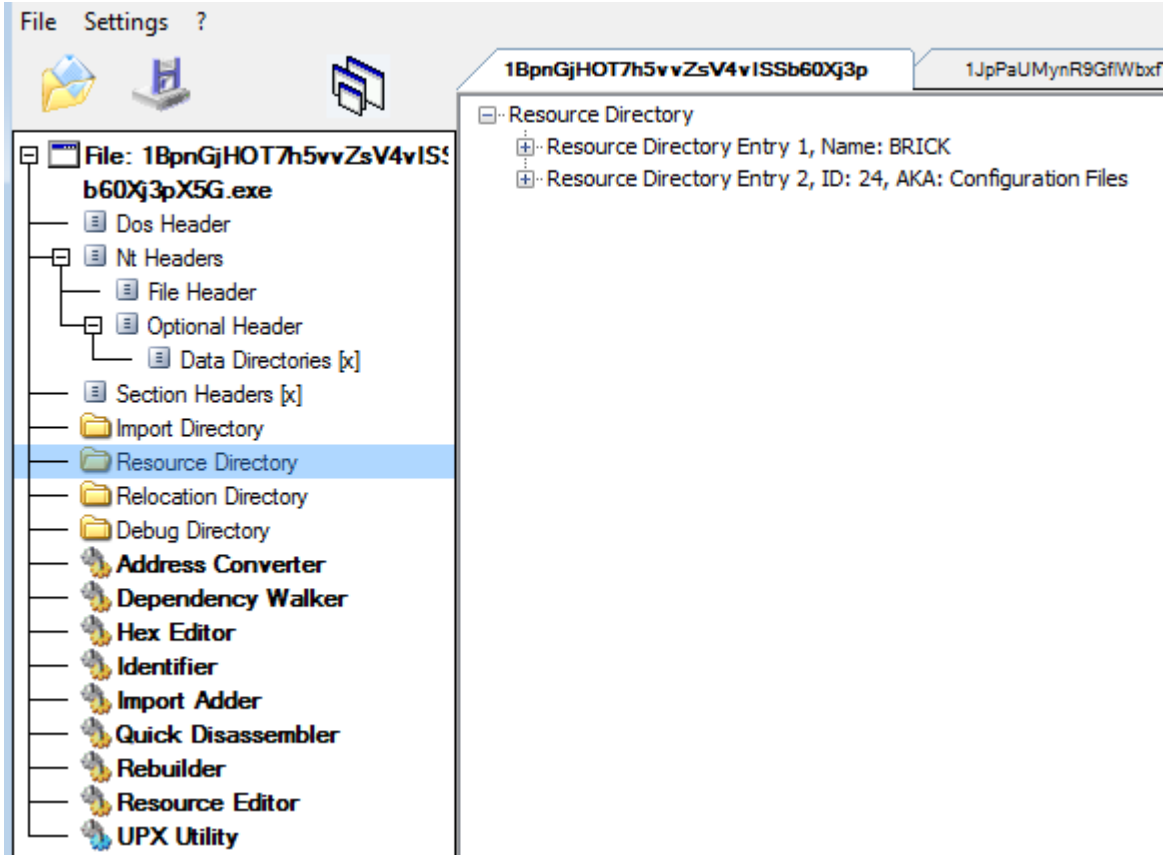
Member	Offset	Size	Value	Section
Export Directory RVA	00000170	Dword	00000000	
Export Directory Size	00000174	Dword	00000000	
Import Directory RVA	00000178	Dword	000035FC	.rdata
Import Directory Size	0000017C	Dword	000000A0	
Resource Directory RVA	00000180	Dword	0000E000	.rsrc
Resource Directory Size	00000184	Dword	00008380	
Exception Directory RVA	00000188	Dword	00000000	
Exception Directory Size	0000018C	Dword	00000000	
Security Directory RVA	00000190	Dword	00000000	
Security Directory Size	00000194	Dword	00000000	
Relocation Directory RVA	00000198	Dword	00017000	.reloc
Relocation Directory Size	0000019C	Dword	0000019C	
Debug Directory RVA	000001A0	Dword	000032A0	.rdata
Debug Directory Size	000001A4	Dword	00000038	
Architecture Directory RVA	000001A8	Dword	00000000	
Architecture Directory Size	000001AC	Dword	00000000	
Reserved	000001B0	Dword	00000000	
Reserved	000001B4	Dword	00000000	
TLS Directory RVA	000001B8	Dword	00000000	
TLS Directory Size	000001BC	Dword	00000000	
Configuration Directory RVA	000001C0	Dword	000032D8	.rdata
Configuration Directory Size	000001C4	Dword	00000040	
Bound Import Directory RVA	000001C8	Dword	00000000	
Bound Import Directory Size	000001CC	Dword	00000000	
Import Address Table Directory ...	000001D0	Dword	00003000	.rdata
Import Address Table Directory ...	000001D4	Dword	000000E8	
Delay Import Directory RVA	000001D8	Dword	00000000	
Delay Import Directory Size	000001DC	Dword	00000000	
.NET MetaData Directory RVA	000001E0	Dword	00000000	
.NET MetaData Directory Size	000001E4	Dword	00000000	

CFF Explorer continued

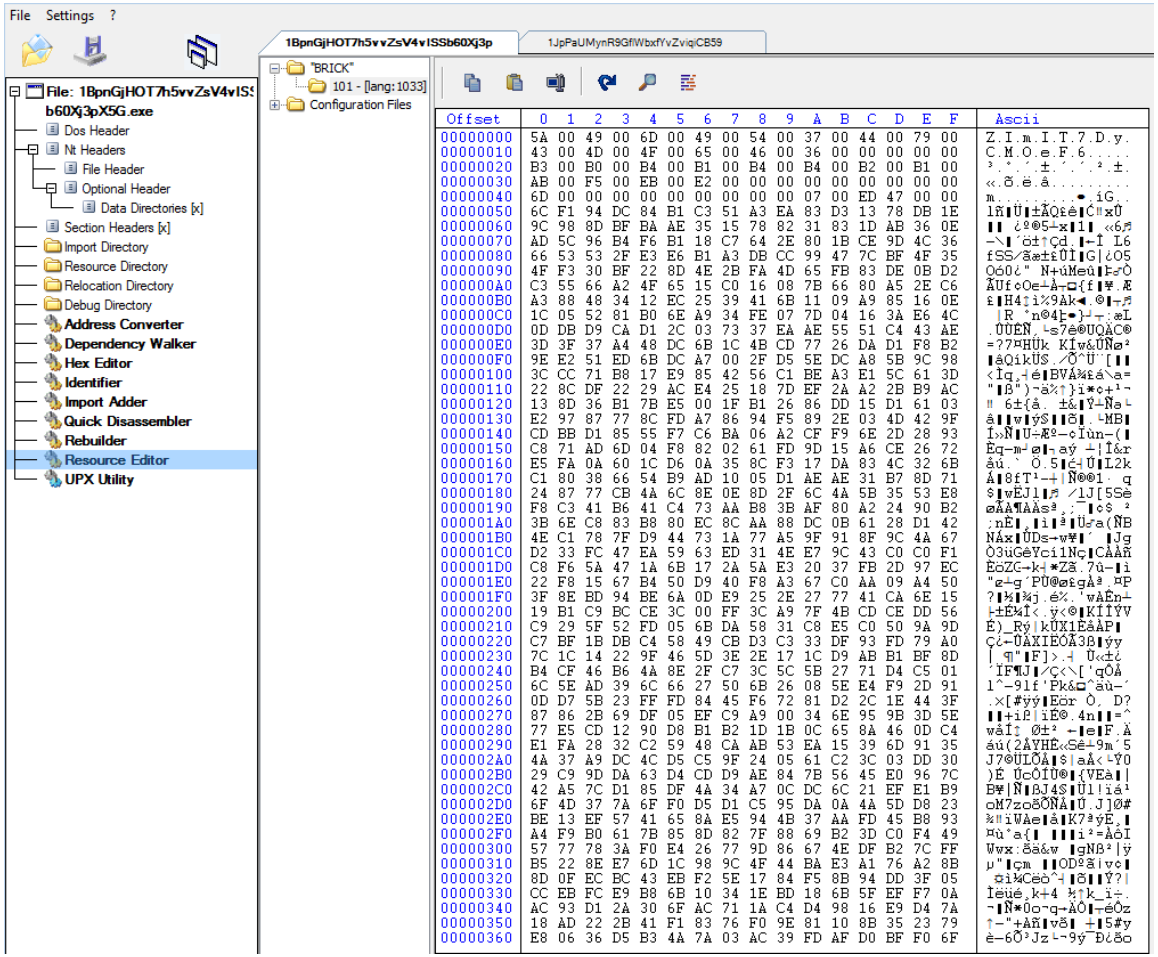


Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	0000126C	00001000	00001400	00000400	00000000	00000000	0000	0000	60000020
.rdata	00000C34	00003000	00000E00	00001800	00000000	00000000	0000	0000	40000040
.data	000084DC	00004000	00000200	00002600	00000000	00000000	0000	0000	C0000040
.gfids	00000020	0000D000	00000200	00002800	00000000	00000000	0000	0000	40000040
.rsrc	00008380	0000E000	00008400	00002A00	00000000	00000000	0000	0000	40000040
.reloc	0000019C	00017000	00000200	0000AE00	00000000	00000000	0000	0000	42000040

CFF Explorer continued



CFF Explorer continued



CFF Explorer continued

File Settings ?

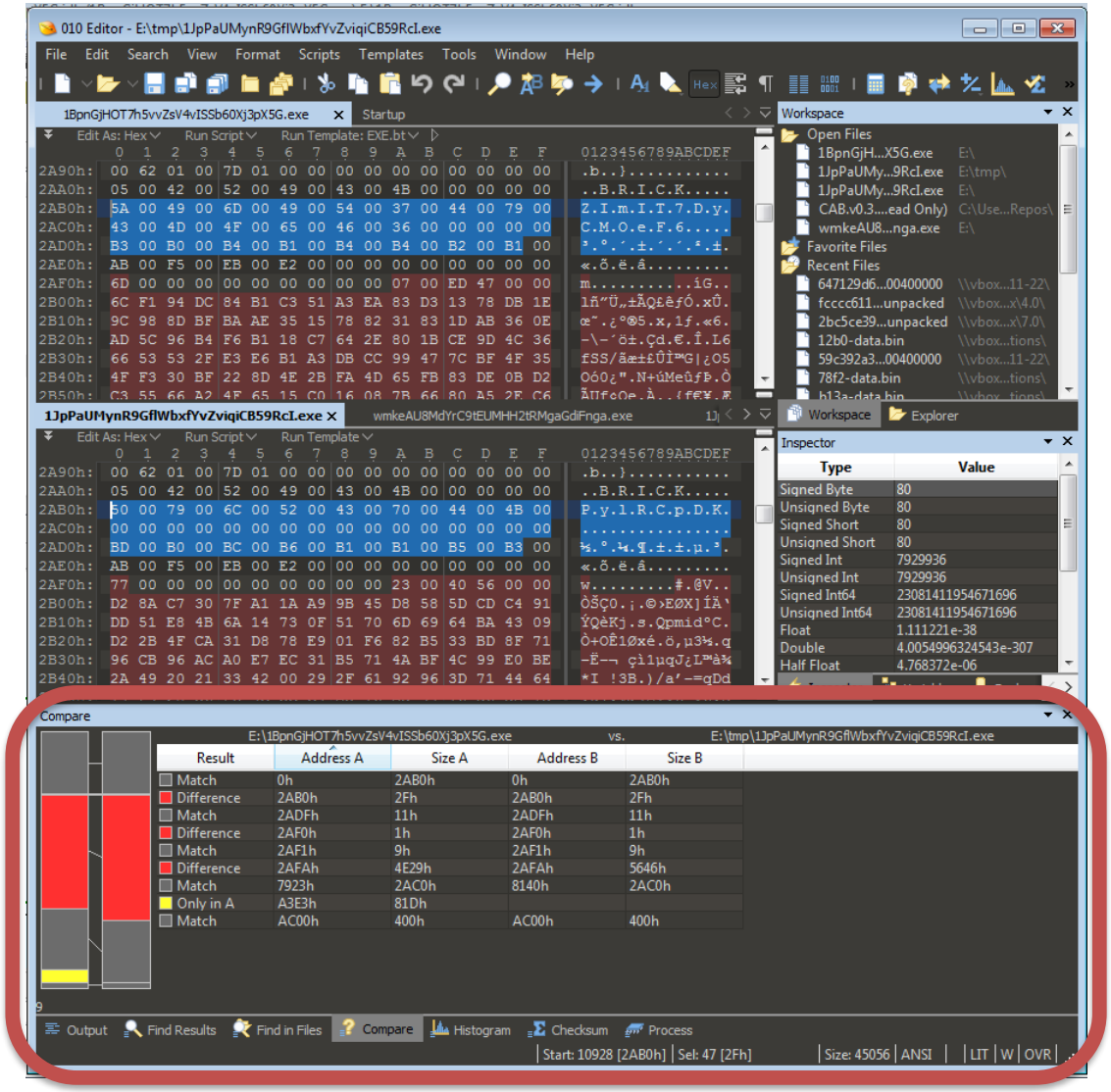
1BpnGjHOT7h5vvZsV4vISSb60Xj3p | 1JpPaUMynR9GfWbxfYvZviqiCB59

- File: 1BpnGjHOT7h5vvZsV4vISSb60Xj3pX5G.exe
  - Dos Header
  - Nt Headers
    - File Header
    - Optional Header
      - Data Directories [x]
  - Section Headers [x]
  - Import Directory
  - Resource Directory
  - Relocation Directory
  - Debug Directory
  - Address Converter
  - Dependency Walker
  - Hex Editor
  - Identifier
  - Import Adder
  - Quick Disassembler
  - Rebuilder
  - Resource Editor
  - UPX Utility

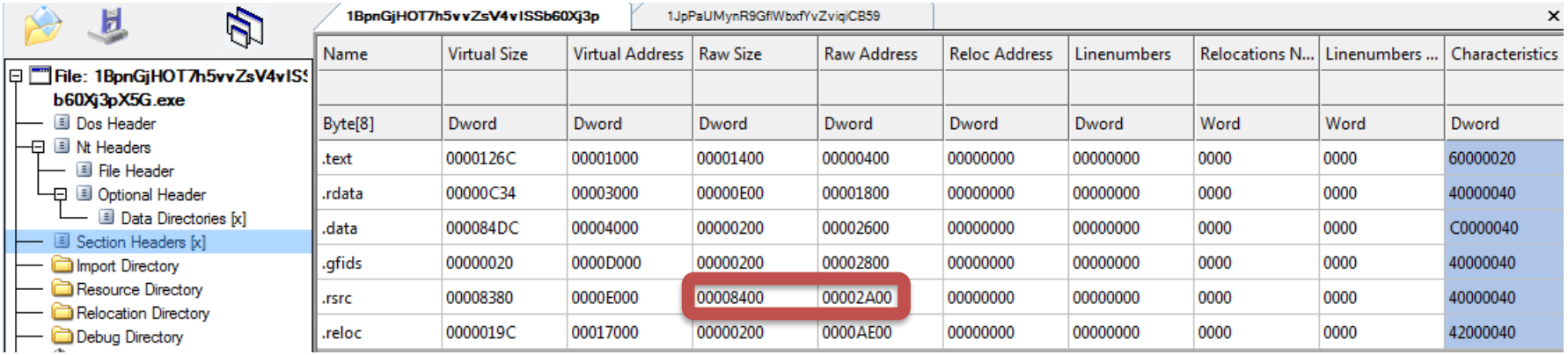
Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
00002004	N/A	00001DFC	00001E00	00001E04	00001E08	00001E0C
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
KERNEL32.dll	20	0000369C	00000000	00000000	00003804	00003000
VCRUNTIME140.dll	3	000036F0	00000000	00000000	00003836	00003054
api-ms-win-crt-stdi...	6	00003768	00000000	00000000	00003A6A	000030CC
api-ms-win-crt-run...	19	00003718	00000000	00000000	00003A8A	0000307C
api-ms-win-crt-mat...	1	00003710	00000000	00000000	00003AAC	00003074
api-ms-win-crt-loc...	1	00003708	00000000	00000000	00003ACC	0000306C
api-ms-win-crt-hea...	1	00003700	00000000	00000000	00003AEE	00003064

OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
00003784	00003784	014E	FindResourceW
00003794	00003794	0341	LoadResource
000037A4	000037A4	0354	LockResource
000037B4	000037B4	04B1	SizeofResource
000037C6	000037C6	0214	GetModuleFileNameW
000037DC	000037DC	008F	CreateFileW
000037EA	000037EA	0525	WriteFile
000037F6	000037F6	0052	CloseHandle
00003B2A	00003B2A	04A5	SetUnhandledExceptionFilter
00003B48	00003B48	01C0	GetCurrentProcess
00003B5C	00003B5C	04C0	TerminateProcess
00003B70	00003B70	0304	IsProcessorFeaturePresent

010 Editor Binary Comparison



CFF Explorer continued



Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	0000126C	00001000	00001400	00000400	00000000	00000000	0000	0000	60000020
.rdata	00000C34	00003000	00000E00	00001800	00000000	00000000	0000	0000	40000040
.data	000084DC	00004000	00000200	00002600	00000000	00000000	0000	0000	C0000040
.gfids	00000020	0000D000	00000200	00002800	00000000	00000000	0000	0000	40000040
.rsrc	00008380	0000E000	00008400	00002A00	00000000	00000000	0000	0000	40000040
.reloc	0000019C	00017000	00000200	0000AE00	00000000	00000000	0000	0000	42000040

IDA – Main Function … Some handy strings

```
argc= dword ptr  8
argv= dword ptr  0Ch
envp= dword ptr  10h

push    ebp
mov     ebp, esp
sub     esp, 24h
mov     eax, ___security_cookie
xor     eax, ebp
mov     [ebp+var_4], eax
xorps   xmm0, xmm0
movups  [ebp+var_24], xmm0
movups  [ebp+var_14], xmm0
call    sub_4012D0
call    sub_401050
test    eax, eax
jnz     short loc_401347
```

```
push    offset aISuperHateYouR ; "I super hate you right now.\n"
call    sub_401510
add     esp, 4
or      eax, 0FFFFFFFFh
mov     ecx, [ebp+var_4]
xor     ecx, ebp
call    @__security_check_cookie@4 ; __security_check_cookie(x)
mov     esp, ebp
pop     ebp
retn
```

```
loc_401347:
push    offset aWhatIsThePassw ; "What is the password?\n"
call    sub_401510
push    10h
lea     eax, [ebp+var_24]
push    eax
push    offset a15ls    ; "%15ls"
call    sub_4014C0
add     esp, 10h
lea     ecx, [ebp+var_24]
call    sub_401240
test    eax, eax
jnz     short loc_40138E
```

```
push    offset aGoStepOnABrick ; "Go step on a brick!\n"
call    sub_401510
add     esp, 4
or      eax, 0FFFFFFFFh
mov     ecx, [ebp+var_4]
xor     ecx, ebp
call    @__security_check_cookie@4 ; __security_check_cookie(x)
mov     esp, ebp
pop     ebp
retn
```

```
loc_40138E:
lea     ecx, [ebp+var_24]
call    sub_401080
call    sub_401100
test    eax, eax
jnz     short loc_4013BD
```

```
push    offset aOhLookARainbow ; "Oh look a rainbow.\n"
call    sub_401510
add     esp, 4
or      eax, 0FFFFFFFFh
mov     ecx, [ebp+var_4]
xor     ecx, ebp
call    @__security_check_cookie@4 ; __security_check_cookie(x)
mov     esp, ebp
pop     ebp
retn
```

```
loc_4013BD:
push    offset aEverythingIsAw ; "Everything is awesome!\n"
call    sub_401510
push    offset unk_4043C0
push    offset unk_4043A0
push    offset aSS_0    ; "%s => %s\n"
call    sub_401510
mov     ecx, [ebp+var_4]
add     esp, 10h
movzx   eax, word_4043CA
xor     ecx, ebp
call    @__security_check_cookie@4 ; __security_check_cookie(x)
mov     esp, ebp
pop     ebp
retn
```

```
,18) 00000700 00401300: _main (Synchronized with Hex View-1)
```

IDA – Where are the strings??

.rdata:004033B0	00000009	C	.CRT$XCZ
.rdata:004033C4	00000009	C	.CRT$XIA
.rdata:004033D8	0000000A	C	.CRT$XIAA
.rdata:004033EC	0000000A	C	.CRT$XIAC
.rdata:00403400	00000009	C	.CRT$XIZ
.rdata:00403414	00000009	C	.CRT$XPA
.rdata:00403428	00000009	C	.CRT$XPZ
.rdata:0040343C	00000009	C	.CRT$XTA
.rdata:00403450	00000009	C	.CRT$XTZ
.rdata:00403464	00000007	C	.rdata
.rdata:00403474	0000000E	C	.rdata$sxdata
.rdata:0040348C	0000000E	C	.rdata$zzzdbg
.rdata:004034A4	00000009	C	.rtc$IAA
.rdata:004034B8	00000009	C	.rtc$IZZ
.rdata:004034CC	00000009	C	.rtc$TAA
.rdata:004034E0	00000009	C	.rtc$TZZ
.rdata:004034F4	00000009	C	.xdata$x
.rdata:00403508	00000009	C	.idata$2
.rdata:0040351C	00000009	C	.idata$3
.rdata:00403530	00000009	C	.idata$4
.rdata:00403544	00000009	C	.idata$6
.rdata:00403558	00000006	C	.data
.rdata:00403568	00000005	C	.bss
.rdata:00403578	00000009	C	.gfids$y
.rdata:0040358C	00000009	C	.rsrc$01
.rdata:004035A0	00000009	C	.rsrc$02
.rdata:00403804	0000000D	C	KERNEL32.dll
.rdata:00403836	00000011	C	VCRUNTIME140.dll
.rdata:00403A6A	00000020	C	api-ms-win-crt-stdio-l1-1-0.dll
.rdata:00403A8A	00000022	C	api-ms-win-crt-runtime-l1-1-0.dll
.rdata:00403AAC	0000001F	C	api-ms-win-crt-math-l1-1-0.dll
.rdata:00403ACC	00000021	C	api-ms-win-crt-locale-l1-1-0.dll
.rdata:00403AEE	0000001F	C	api-ms-win-crt-heap-l1-1-0.dll

IDA – They exist in the data section?

```
.rdata:0040311F                 db    0
.rdata:00403120 ; struct _EXCEPTION_POINTERS ExceptionInfo
.rdata:00403120 ExceptionInfo   _EXCEPTION_POINTERS <offset dword_404018, offset dword_404068>
.rdata:00403120                                         ; DATA XREF: ___report_gsfailure+ED↑o
.rdata:00403128 ; const WCHAR Type
.rdata:00403128 Type:                                   ; DATA XREF: sub_401000+1↑o
.rdata:00403128                 text "UTF-16LE", 'BRICK',0
.rdata:00403134 aSS:                                    ; DATA XREF: sub_401100+AA↑o
.rdata:00403134                 text "UTF-16LE", '%s\%s',0
.rdata:00403140 aIronmansucks:                          ; DATA XREF: sub_401240↑o
.rdata:00403140                 text "UTF-16LE", 'IronManSucks',0
.rdata:0040315A                 align 4
.rdata:0040315C aOhHelloBatman:                         ; DATA XREF: sub_401240+38↑o
.rdata:0040315C                 text "UTF-16LE", 'Oh, hello Batman...',0Ah,0
.rdata:00403186                 align 4
.rdata:00403188 aISuperHateYouR:                        ; DATA XREF: _main+29↑o
.rdata:00403188                 text "UTF-16LE", 'I super hate you right now.',0Ah,0
.rdata:004031C2                 align 4
.rdata:004031C4 aWhatIsThePassw:                        ; DATA XREF: _main:loc_401347↑o
.rdata:004031C4                 text "UTF-16LE", 'What is the password?',0Ah,0
.rdata:004031F2                 align 4
.rdata:004031F4 a15ls:                                  ; DATA XREF: _main+57↑o
.rdata:004031F4                 text "UTF-16LE", '%15ls',0
.rdata:00403200 aGoStepOnABrick:                        ; DATA XREF: _main+70↑o
.rdata:00403200                 text "UTF-16LE", 'Go step on a brick!',0Ah,0
.rdata:0040322A                 align 4
.rdata:0040322C aOhLookARainbow:                        ; DATA XREF: _main+9F↑o
.rdata:0040322C                 text "UTF-16LE", 'Oh look a rainbow.',0Ah,0
.rdata:00403254 aEverythingIsAw:                        ; DATA XREF: _main:loc_4013BD↑o
.rdata:00403254                 text "UTF-16LE", 'Everything is awesome!',0Ah,0
.rdata:00403284 aSS_0:                                  ; DATA XREF: _main+D1↑o
.rdata:00403284                 text "UTF-16LE", '%s => %s',0Ah,0
```

IDA – Locate the interesting imports

IDA View-A | Imports | Strings window | Hex View-

Address	Ordinal	Name	Library
00403000		FindResourceW	KERNEL32
00403004		LoadResource	KERNEL32
00403008		LockResource	KERNEL32
0040300C		SizeofResource	KERNEL32
00403010		GetModuleFileNameW	KERNEL32
00403014		CreateFileW	KERNEL32
00403018		WriteFile	KERNEL32
0040301C		CloseHandle	KERNEL32
00403020		SetUnhandledExceptionFilter	KERNEL32
00403024		GetCurrentProcess	KERNEL32
00403028		TerminateProcess	KERNEL32
0040302C		IsProcessorFeaturePresent	KERNEL32
00403030		GetModuleHandleW	KERNEL32
00403034		IsDebuggerPresent	KERNEL32
00403038		InitializeSListHead	KERNEL32
0040303C		GetSystemTimeAsFileTime	KERNEL32
00403040		GetCurrentThreadId	KERNEL32
00403044		GetCurrentProcessId	KERNEL32
00403048		QueryPerformanceCounter	KERNEL32
0040304C		UnhandledExceptionFilter	KERNEL32
00403054		memset	VCRUNTIME140

IDA –Brick looking important

```
00401000
00401000                        sub_401000 proc near
00401000 57                     push    edi
00401001 68 28 31 40 00         push    offset Type     ; "BRICK"
00401006 6A 65                  push    65h ; 'e'       ; lpName
00401008 6A 00                  push    0               ; hModule
0040100A FF 15 00 30 40 00      call    ds:FindResourceW
00401010 8B F8                  mov     edi, eax
00401012 85 FF                  test    edi, edi
00401014 75 04                  jnz     short loc_40101A
```

```
0040101A
0040101A                        loc_40101A:             ; hResInfo
0040101A 57                     push    edi
0040101B 6A 00                  push    0               ; hModule
0040101D FF 15 04 30 40 00      call    ds:LoadResource
00401023 85 C0                  test    eax, eax
00401025 74 EF                  jz      short loc_401016
```

```
00401016
00401016                        loc_401016:
00401016 33 C0                  xor     eax, eax
00401018 5F                     pop     edi
00401019 C3                     retn
```

```
00401027 56                     push    esi
00401028 50                     push    eax             ; hResData
00401029 FF 15 08 30 40 00      call    ds:LockResource
0040102F 57                     push    edi             ; hResInfo
00401030 6A 00                  push    0               ; hModule
00401032 8B F0                  mov     esi, eax
00401034 FF 15 0C 30 40 00      call    ds:SizeofResource
0040103A 33 C9                  xor     ecx, ecx
0040103C 3D 50 81 00 00         cmp     eax, 8150h
00401041 0F 45 F1               cmovnz  esi, ecx
00401044 8B C6                  mov     eax, esi
00401046 5E                     pop     esi
00401047 5F                     pop     edi
00401048 C3                     retn
00401048                        sub_401000 endp
00401048
```

IDA – Stepping out - Brick copied local

```
00401050
00401050
00401050
00401050                    sub_401050 proc near
00401050 E8 AB FF FF FF     call    sub_401000
00401055 85 C0              test    eax, eax
00401057 75 01              jnz     short loc_40105A
```

```
00401059 C3                 retn
```

```
0040105A                    loc_40105A:             ; Size
0040105A                    push    8150h
0040105A 68 50 81 00 00     push    eax             ; Src
0040105F 50                 push    offset unk_404380 ; Dst
00401060 68 80 43 40 00     call    memcpy
00401065 E8 FC 11 00 00     add     esp, 0Ch
0040106A 83 C4 0C           mov     eax, 1
0040106D B8 01 00 00 00     retn
00401072 C3                 sub_401050 endp
00401072
00401072
```

IDA – Resource loading is at the start of Main

```
00401300                        ; int __cdecl main(int argc, const char **argv, const char **envp)
00401300                        _main proc near
00401300
00401300                        var_24= xmmword ptr -24h
00401300                        var_14= xmmword ptr -14h
00401300                        var_4= dword ptr -4
00401300                        argc= dword ptr  8
00401300                        argv= dword ptr  0Ch
00401300                        envp= dword ptr  10h
00401300
00401300 55                     push    ebp
00401301 8B EC                  mov     ebp, esp
00401303 83 EC 24               sub     esp, 24h
00401306 A1 04 40 40 00         mov     eax, ___security_cookie
0040130B 33 C5                  xor     eax, ebp
0040130D 89 45 FC               mov     [ebp+var_4], eax
00401310 0F 57 C0               xorps   xmm0, xmm0
00401313 0F 11 45 DC            movups  [ebp+var_24], xmm0
00401317 0F 11 45 EC            movups  [ebp+var_14], xmm0
0040131B E8 B0 FF FF FF         call    sub_4012D0
00401320 E8 2B FD FF FF         call    sub_401050
00401325 85 C0                  test    eax, eax
00401327 75 1E                  jnz     short loc_401347
```

```
fset aISuperHateYouR ; "I super hate you right now.\n"
b_401510
p, 4
x, 0FFFFFFFFh
x, [ebp+var_4]
x, ebp
_security_check_cookie@4 ; __security_check_cookie(x)
p, ebp
p
```

```
00401347
00401347                        loc_401347:
00401347 68 C4 31 40 00         push    offset aWhatIsThePassw ; "What is the password?\n"
0040134C E8 BF 01 00 00         call    sub_401510
00401351 6A 10                  push    10h
00401353 8D 45 DC               lea     eax, [ebp+var_24]
00401356 50                     push    eax
00401357 68 F4 31 40 00         push    offset a15ls    ; "%15ls"
0040135C E8 5F 01 00 00         call    sub_4014C0
00401361 83 C4 10               add     esp, 10h
00401364 8D 4D DC               lea     ecx, [ebp+var_24]
00401367 E8 D4 FE FF FF         call    sub_401240
0040136C 85 C0                  test    eax, eax
0040136E 75 1E                  jnz     short loc_40138E
```

IDA – Main Continued

```
00401347
00401347
00401347 68 C4 31 40 00          push    offset aWhatIsThePassw ; "What is the password?\n"
0040134C E8 BF 01 00 00          call    sub_401510
00401351 6A 10                   push    10h
00401353 8D 45 DC                lea     eax, [ebp+var_24]
00401356 50                      push    eax
00401357 68 F4 31 40 00          push    offset a15ls    ; "%15ls"
0040135C E8 5F 01 00 00          call    sub_4014C0
00401361 83 C4 10                add     esp, 10h
00401364 8D 4D DC                lea     ecx, [ebp+var_24]
00401367 E8 D4 FE FF FF          call    sub_401240
0040136C 85 C0                   test    eax, eax
0040136E 75 1E                   jnz     short loc_40138E
```

```
right now.\n"
eck_cookie(x)
```

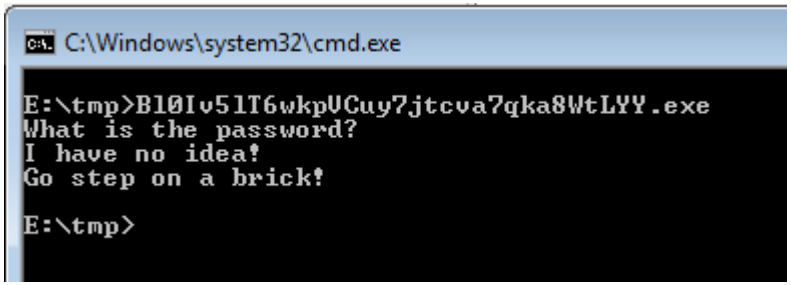
```
2 40 00          push    offset aGoStepOnABrick ; "Go step on a brick!\n"
1 00 00          call    sub_401510
4                add     esp, 4
:                or      eax, 0FFFFFFFFh
:                mov     ecx, [ebp+var_4]
:                xor     ecx, ebp
3 00 00          call    @__security_check_cookie@4 ; __security_check_cookie(x)
                 mov     esp, ebp
                 pop     ebp
                 retn
```

```
0040138E
0040138E                         loc_40138E:
0040138E 8D 4D DC                lea     ecx, [ebp+var_24]
00401391 E8 1A FD FF FF          call    sub_401080
00401396 E8 65 FD FF FF          call    sub_401100
0040139B 85 C0                   test    eax, eax
0040139D 75 1E                   jnz     short loc_4013BD
```

```
0040139F 68 2C 32 40 00          push    offset aOhLookARainbow ; "Oh look a rainbow.\n"
004013A4 E8 67 01 00 00          call    sub_401510
004013A9 83 C4 04                add     esp, 4
004013AC 83 C8 FF                or      eax, 0FFFFFFFFh
004013AF 8B 4D FC                mov     ecx, [ebp+var_4]
004013B2 33 CD                   xor     ecx, ebp
004013B4 E8 F2 02 00 00          call    @__security_check_cookie@4 ; __security_check_cookie(x)
004013B9 8B E5                   mov     esp, ebp
004013BB 5D                      pop     ebp
004013BC C3                      retn
```

```
004013BD
004013BD                         loc_4013BD:
004013BD 68 54 32 40 00          push    offset aEverythingIsAw ; "Everything is awesome!\n"
004013C2 E8 49 01 00 00          call    sub_401510
004013C7 68 C0 43 40 00          push    offset unk_4043C0
004013CC 68 A0 43 40 00          push    offset unk_4043A0
004013D1 68 84 32 40 00          push    offset aSS_0    ; "%s => %s\n"
004013D6 E8 35 01 00 00          call    sub_401510
004013DB 8B 4D FC                mov     ecx, [ebp+var_4]
004013DE 83 C4 10                add     esp, 10h
004013E1 0F B7 05 CA 43 40 00    movzx   eax, word_4043CA
004013E8 33 CD                   xor     ecx, ebp
004013EA E8 BC 02 00 00          call    @__security_check_cookie@4 ; __security_check_cookie(x)
004013EF 8B E5                   mov     esp, ebp
004013F1 5D                      pop     ebp
004013F2 C3                      retn
004013F2                         _main endp
004013F2
```

Ok – Why not run one?



IDA – Main Continued – return 0 steps on bricks

```
00401347
00401347
00401347 68 C4 31 40 00        push    offset aWhatIsThePassw ; "What is the password?\n"
0040134C E8 BF 01 00 00        call    sub_401510
00401351 6A 10                 push    10h
00401353 8D 45 DC              lea     eax, [ebp+var_24]
00401356 50                    push    eax
00401357 68 F4 31 40 00        push    offset a15ls    ; "%15ls"
0040135C E8 5F 01 00 00        call    sub_4014C0
00401361 83 C4 10              add     esp, 10h
00401364 8D 4D DC              lea     ecx, [ebp+var_24]
00401367 E8 D4 FE FF FF        call    sub_401240
0040136C 85 C0                 test    eax, eax
0040136E 75 1E                 jnz     short loc_40138E
```

```
00 40 00                       push    offset aGoStepOnABrick ; "Go step on a brick!\n"
00 00                          call    sub_401510
                               add     esp, 4
                               or      eax, 0FFFFFFFFh
                               mov     ecx, [ebp+var_4]
                               xor     ecx, ebp
00 00                          call    @__security_check_cookie@4 ; __security_check_cookie(x)
                               mov     esp, ebp
                               pop     ebp
                               retn
```

```
0040138E
0040138E                       loc_40138E:
0040138E 8D 4D DC              lea     ecx, [ebp+var_24]
00401391 E8 1A FD FF FF        call    sub_401080
00401396 E8 65 FD FF FF        call    sub_401100
0040139B 85 C0                 test    eax, eax
0040139D 75 1E                 jnz     short loc_4013BD
```

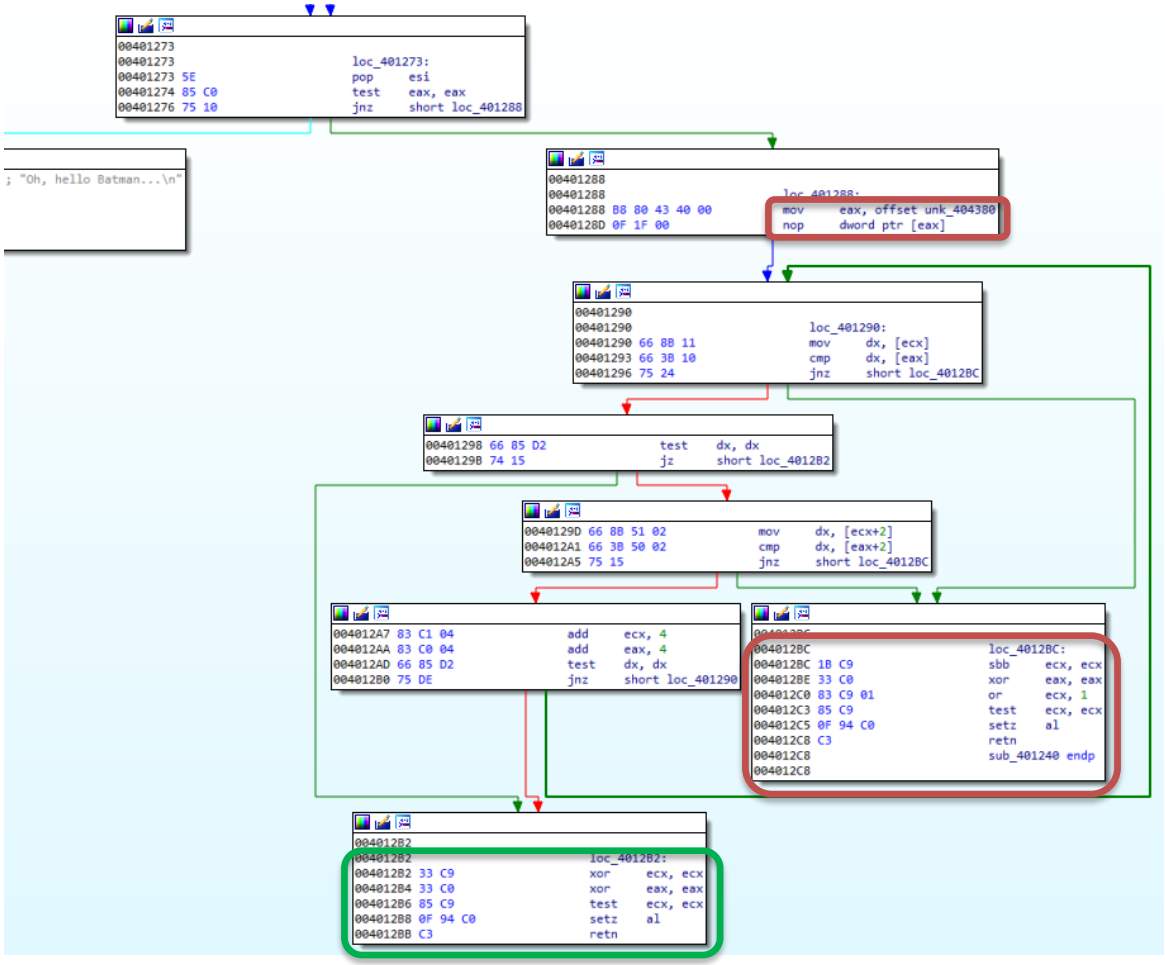
```
0040139F 68 2C 32 40 00        push    offset aOhLookARainbow ; "Oh look a rainbow.\n"
004013A4 E8 67 01 00 00        call    sub_401510
004013A9 83 C4 04              add     esp, 4
004013AC 83 C8 FF              or      eax, 0FFFFFFFFh
004013AF 8B 4D FC              mov     ecx, [ebp+var_4]
004013B2 33 CD                 xor     ecx, ebp
004013B4 E8 F2 02 00 00        call    @__security_check_cookie@4 ; __security_check_cookie(x)
004013B9 8B E5                 mov     esp, ebp
004013BB 5D                    pop     ebp
004013BC C3                    retn
```

```
004013BD
004013BD                       loc_4013BD:
004013BD 68 54 32 40 00        push    offset aEverythingIsAw ; "Everything is awesome!\n"
004013C2 E8 49 01 00 00        call    sub_401510
004013C7 68 C0 43 40 00        push    offset unk_4043C0
004013CC 68 A0 43 40 00        push    offset unk_4043A0
004013D1 68 84 32 40 00        push    offset aSS_0    ; "%s => %s\n"
004013D6 E8 35 01 00 00        call    sub_401510
004013DB 8B 4D FC              mov     ecx, [ebp+var_4]
004013DE 83 C4 10              add     esp, 10h
004013E1 0F B7 05 CA 43 40 00  movzx   eax, word_4043CA
004013E8 33 CD                 xor     ecx, ebp
004013EA E8 BC 02 00 00        call    @__security_check_cookie@4 ; __security_check_cookie(x)
004013EF 8B E5                 mov     esp, ebp
004013F1 5D                    pop     ebp
004013F2 C3                    retn
004013F2                       _main endp
004013F2
```

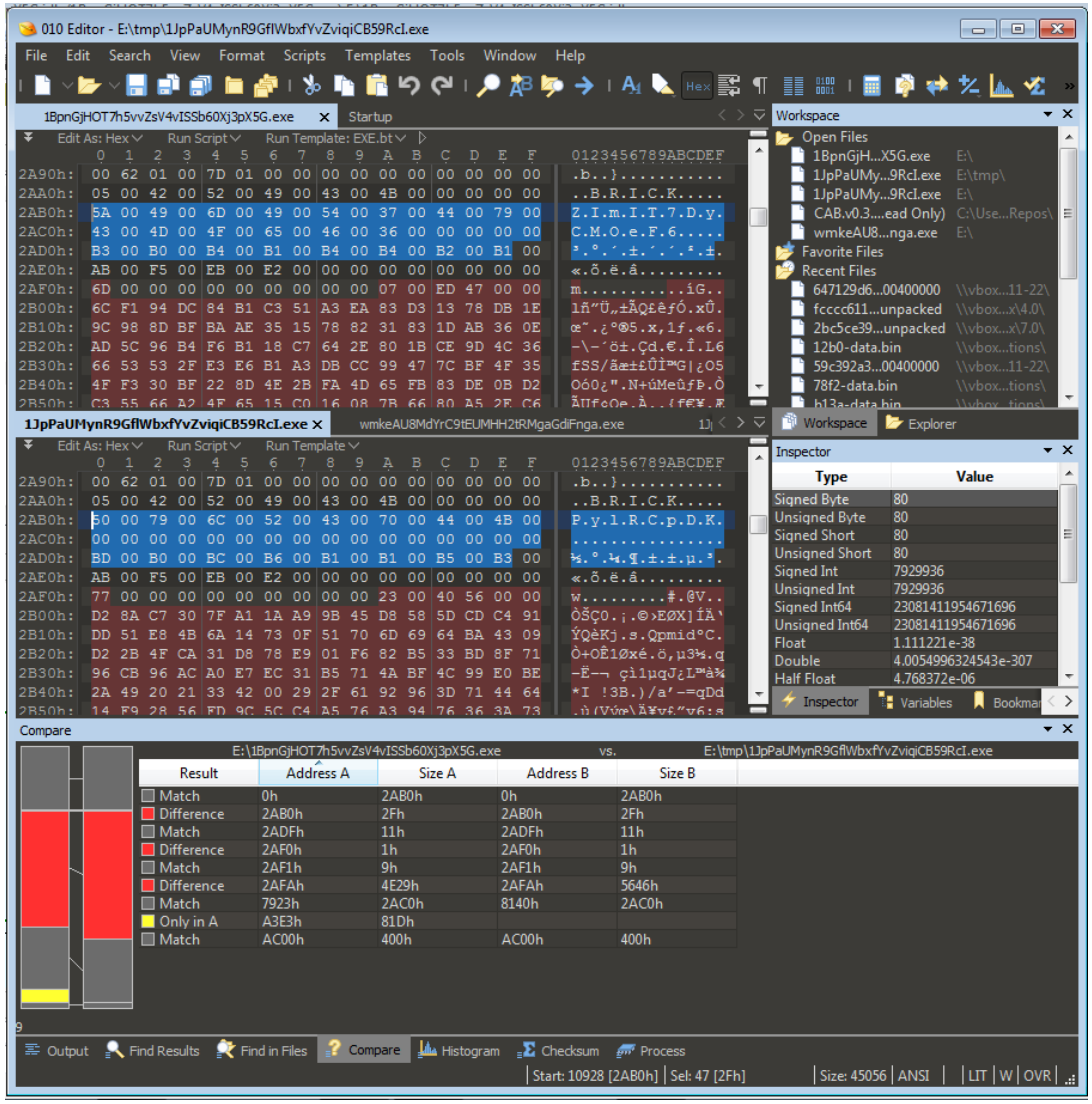
IDA – Check the users input – False Lead

```
00401240
00401240
00401240
00401240
00401240 BA 40 31 40 00        mov     edx, offset aIronmansucks ; "IronManSucks"
00401245 8B C1                 mov     eax, ecx
00401247 56

00401248
00401248                       loc_401248:
00401248 66 8B 30              mov     si, [eax]
0040124B 66 3B 32              cmp     si, [edx]
0040124E 75 1E                 jnz     short loc_40126E

00401250 66 85 F6              test    si, si
00401253 74 15                 jz      short loc_40126A

00401255 66 8B 70 02           mov     si, [eax+2]
00401259 66 3B 72 02           cmp     si, [edx+2]
0040125D 75 0F                 jnz     short loc_40126E

0040125F 83 C0 04              add     eax, 4
00401262 83 C2 04              add     edx, 4
00401265 66 85 F6              test    si, si
00401268 75 DE                 jnz     short loc_401248

0040126A
0040126A                       loc_40126A:
0040126A 33 C0                 xor     eax, eax
0040126C EB 05                 jmp     short loc_401273

0040126E
0040126E                       loc_40126E:
0040126E 1B C0                 sbb     eax, eax
00401270 83 C8 01              or      eax, 1

00401273
00401273                       loc_401273:
00401273 5E                    pop     esi
00401274 85 C0                 test    eax, eax
00401276 75 10                 jnz     short loc_401288

push    offset aOhHelloBatman ; "Oh, hello Batman...\n"
call    sub_401510
add     esp, 4
xor     eax, eax
retn

00401288
00401288                       loc_401288:
00401288 B8 80 43 40 00        mov     eax, offset unk_404380
0040128D 0F 1F 00              nop     dword ptr [eax]
```

IDA – Now check against the loaded resource



Refresh –Password seems to be at resource start



Refresh –Password seems to be at resource start

1BpnGjHOT7h5vvZsV4vISSb60Xj3pX5G.exe

Edit As: Hex Run Script Run Template: EXE.bt

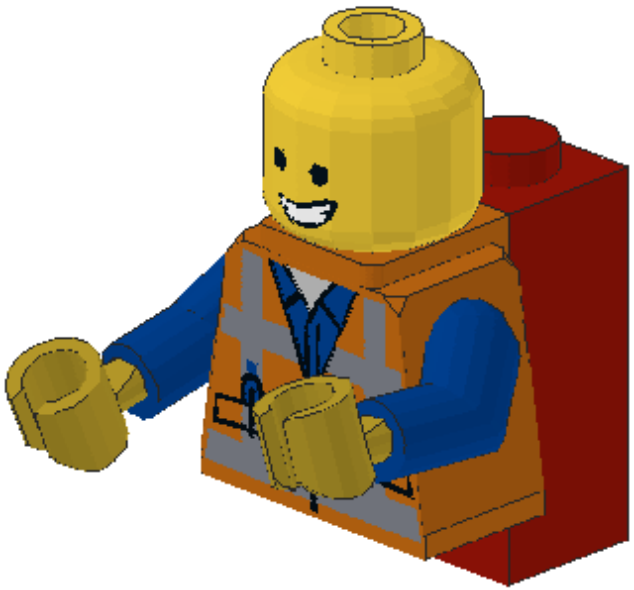
```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0123456789ABCDEF
2A90h: 00 62 01 00 7D 01 00 00 00 00 00 00 00 00 00 00    .b..}...........
2AA0h: 05 00 42 00 52 00 49 00 43 00 4B 00 00 00 00 00    ..B.R.I.C.K.....
2AB0h: 5A 00 49 00 6D 00 49 00 54 00 37 00 44 00 79 00    Z.I.m.I.T.7.D.y.
2AC0h: 43 00 4D 00 4F 00 65 00 46 00 36 00 00 00 00 00    C.M.O.e.F.6.....
2AD0h: B3 00 B0 00 B4 00 B1 00 B4 00 B4 00 B2 00 B1 00    ³.°.´.±.´.´.².±.
2AE0h: AB 00 F5 00 EB 00 E2 00 00 00 00 00 00 00 00 00    «.õ.ë.â.........
2AF0h: 6D 00 00 00 00 00 00 00 00 00 07 00 ED 47 00 00    m...........íG..
2B00h: 6C F1 94 DC 84 B1 C3 51 A3 EA 83 D3 13 78 DB 1E    lñ"Ü„±ÃQ£êƒÓ.xÛ.
2B10h: 9C 98 8D BF BA AE 35 15 78 82 31 83 1D AB 36 0E    œ˜.¿º®5.x‚1ƒ.«6.
```

Try it out

Select C:\Windows\system32\cmd.exe

```
E:\tmp>1BpnGjHOT7h5vvZsV4vISSb60Xj3pX5G.exe
What is the password?
ZImIT7DyCMOeF6
Everything is awesome!
65141174.png => w

E:\tmp>dir *.PNG
 Volume in drive E is VBOX_share
 Volume Serial Number is 0000-0802

 Directory of E:\tmp

10/04/2018  03:19 PM            18,413 65141174.png
               1 File(s)         18,413 bytes
               0 Dir(s)  103,112,105,984 bytes free

E:\tmp>
```

65141174.png



IDA – Everything is awesome! – We have a file?

```
00401347
00401347                        loc_401347:
00401347 68 C4 31 40 00         push    offset aWhatIsThePassw ; "What is the password?\n"
0040134C E8 BF 01 00 00         call    sub_401510
00401351 6A 10                  push    10h
00401353 8D 45 DC               lea     eax, [ebp+var_24]
00401356 50                     push    eax
00401357 68 F4 31 40 00         push    offset a15ls    ; "%15ls"
0040135C E8 5F 01 00 00         call    sub_4014C0
00401361 83 C4 10               add     esp, 10h
00401364 8D 4D DC               lea     ecx, [ebp+var_24]
00401367 E8 D4 FE FF FF         call    sub_401240
0040136C 85 C0                  test    eax, eax
0040136E 75 1E                  jnz     short loc_40138E
```

```
0040138E
0040138E                        loc_40138E:
0040138E 8D 4D DC               lea     ecx, [ebp+var_24]
00401391 E8 1A FD FF FF         call    sub_401080
00401396 E8 65 FD FF FF         call    sub_401100
0040139B 85 C0                  test    eax, eax
0040139D 75 1E                  jnz     short loc_4013BD
```

```
0040139F 68 2C 32 40 00         push    offset aOhLookARainbow ; "Oh look a rainbow.\n"
004013A4 E8 67 01 00 00         call    sub_401510
004013A9 83 C4 04               add     esp, 4
004013AC 83 C8 FF               or      eax, 0FFFFFFFFh
004013AF 8B 4D FC               mov     ecx, [ebp+var_4]
004013B2 33 CD                  xor     ecx, ebp
004013B4 E8 F2 02 00 00         call    @__security_check_cookie@4 ; __security_check_cookie(x)
004013B9 8B E5                  mov     esp, ebp
004013BB 5D                     pop     ebp
004013BC C3                     retn
```

```
004013BD
004013BD                        loc_4013BD:
004013BD 68 54 32 40 00         push    offset aEverythingIsAw ; "Everything is awesome!\n"
004013C2 E8 49 01 00 00         call    sub_401510
004013C7 68 C0 43 40 00         push    offset unk_4043C0
004013CC 68 A0 43 40 00         push    offset unk_4043A0
004013D1 68 84 32 40 00         push    offset aSS_0    ; "%s => %s\n"
004013D6 E8 35 01 00 00         call    sub_401510
004013DB 8B 4D FC               mov     ecx, [ebp+var_4]
004013DE 83 C4 10               add     esp, 10h
004013E1 0F B7 05 CA 43 40 00   movzx   eax, word_4043CA
004013E8 33 CD                  xor     ecx, ebp
004013EA E8 BC 02 00 00         call    @__security_check_cookie@4 ; __security_check_cookie(x)
004013EF 8B E5                  mov     esp, ebp
004013F1 5D                     pop     ebp
004013F2 C3                     retn
004013F2                        _main endp
004013F2
```

IDA – Lets decrypt stuff – 0x404380 is resource

```
004010B0
004010B0
004010B0
004010B0                        sub_4010B0 proc near
004010B0 56                     push    esi
004010B1 8B F1                  mov     esi, ecx
004010B3 BA 20 00 00 00         mov     edx, 20h ; ' '
004010B8 68 85 00 00 00         push    85h ; '…'
004010BD B9 A0 43 40 00         mov     ecx, offset unk_4043A0
004010C2 E8 B9 FF FF FF         call    sub_401080
004010C7 6A 1A                  push    1Ah
004010C9 BA 0A 00 00 00         mov     edx, 0Ah
004010CE B9 C0 43 40 00         mov     ecx, offset unk_4043C0
004010D3 E8 A8 FF FF FF         call    sub_401080
004010D8 68 D0 43 40 00         push    offset unk_4043D0
004010DD FF 35 CC 43 40 00      push    Dst
004010E3 BA D0 43 40 00         mov     edx, offset unk_4043D0
004010E8 8B CE                  mov     ecx, esi
004010EA E8 71 05 00 00         call    sub_401660
004010EF 83 C4 10               add     esp, 10h
004010F2 5E                     pop     esi
004010F3 C3                     retn
004010F3                        sub_4010B0 endp
004010F3
```

This address is 32 bytes from the start of the resource

This address is 64 bytes from the start of the resource

This address is 80 bytes from the start of the resource

IDA – 0x401080 - Simple XOR with a loop

```
00401080
00401080
00401080                        ; Attributes: bp-based frame
00401080
00401080                        sub_401080 proc near
00401080
00401080                        arg_0= byte ptr  8
00401080
00401080 55                     push    ebp
00401081 8B EC                  mov     ebp, esp
00401083 33 C0                  xor     eax, eax
00401085 56                     push    esi
00401086 8B F1                  mov     esi, ecx
00401088 85 D2                  test    edx, edx
0040108A 74 16                  jz      short loc_4010A2
```

```
0040108C 53                     push    ebx
0040108D 8A 5D 08               mov     bl, [ebp+arg_0]
```

```
00401090
00401090                        loc_401090:
00401090 8A 0C 30               mov     cl, [eax+esi]
00401093 84 C9                  test    cl, cl
00401095 74 05                  jz      short loc_40109C
```

```
00401097 32 CB                  xor     cl, bl
00401099 88 0C 30               mov     [eax+esi], cl
```

```
0040109C
0040109C                        loc_40109C:
0040109C 40                     inc     eax
0040109D 3B C2                  cmp     eax, edx
0040109F 72 EF                  jb      short loc_401090
```

```
004010A1 5B                     pop     ebx
```

```
004010A2
004010A2                        loc_4010A2:
004010A2 B8 01 00 00 00         mov     eax, 1
004010A7 5E                     pop     esi
004010A8 5D                     pop     ebp
004010A9 C3                     retn
004010A9                        sub_401080 endp
004010A9
```

Decrypt in place and using arg_0

Refresh – This gives up the filename and letter

1BpnGjHOT7h5vvZsV4vISSb60Xj3pX5G.exe

Edit As: Hex | Run Script | Run Template: EXE.bt

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
2A90h:	00	62	01	00	7D	01	00	00	00	00	00	00	00	00	00	00	.b..}.....
2AA0h:	05	00	42	00	52	00	49	00	43	00	4B	00	00	00	00	00	..B.R.I.C.K.....
2AB0h:	5A	00	49	00	6D	00	49	00	54	00	37	00	44	00	79	00	Z.I.m.I.T.7.D.y.
2AC0h:	43	00	4D	00	4F	00	65	00	46	00	36	00	00	00	00	00	C.M.O.e.F.6.....
2AD0h:	B3	00	B0	00	B4	00	B1	00	B4	00	B4	00	B2	00	B1	00	³.°.´.±.´.´.².±.
2AE0h:	AB	00	F5	00	EB	00	E2	00	00	00	00	00	00	00	00	00	«.õ.ë.â.....
2AF0h:	6D	00	00	00	00	00	00	00	00	00	07	00	ED	47	00	00	m.....íG..
2B00h:	6C	F1	94	DC	84	B1	C3	51	A3	EA	83	D3	13	78	DB	1E	lñ"Ü„±ÃQ£êƒÓ.xÛ.
2B10h:	9C	98	8D	BF	BA	AE	35	15	78	82	31	83	1D	AB	36	0E	œ˜.¿°®5.x‚1ƒ.«6.

Pass		XOR		ASCII	Char
B3	xor	85	=	36	6
B0	xor	85	=	35	5
B4	xor	85	=	31	1
B1	xor	85	=	34	4
B4	xor	85	=	31	1
B4	xor	85	=	31	1
B2	xor	85	=	37	7
B1	xor	85	=	34	4
AB	xor	85	=	2E	.
F5	xor	85	=	70	p
EB	xor	85	=	6E	n
E2	xor	85	=	67	g
6D	xor	1A	=	77	w

Select C:\Windows\system32\cmd.exe

```
E:\tmp>1BpnGjHOT7h5vvZsV4vISSb60Xj3pX5G.exe
What is the password?
ZImIT7DyCMOeF6
Everything is awesome!
65141174.png => w

E:\tmp>dir *.PNG
 Volume in drive E is VBOX_share
 Volume Serial Number is 0000-0802

 Directory of E:\tmp

10/04/2018  03:19 PM            18,413 65141174.png
               1 File(s)         18,413 bytes
               0 Dir(s)  103,112,105,984 bytes free

E:\tmp>
```

IDA – Look mum, no hands and no extra input

```
004010B0
004010B0
004010B0
004010B0                        sub_4010B0 proc near
004010B0 56                     push    esi
004010B1 8B F1                  mov     esi, ecx
004010B3 BA 20 00 00 00         mov     edx, 20h ; ' '
004010B8 68 85 00 00 00         push    85h ; '…'
004010BD B9 A0 43 40 00         mov     ecx, offset unk_4043A0
004010C2 E8 B9 FF FF FF         call    sub_401080
004010C7 6A 1A                  push    1Ah
004010C9 BA 0A 00 00 00         mov     edx, 0Ah
004010CE B9 C0 43 40 00         mov     ecx, offset unk_4043C0
004010D3 E8 A8 FF FF FF         call    sub_401080
004010D8 68 D0 43 40 00         push    offset unk_4043D0
004010DD FF 35 CC 43 40 00      push    Dst
004010E3 BA D0 43 40 00         mov     edx, offset unk_4043D0
004010E8 8B CE                  mov     ecx, esi
004010EA E8 71 05 00 00         call    sub_401660
004010EF 83 C4 10               add     esp, 10h
004010F2 5E                     pop     esi
004010F3 C3                     retn
004010F3                        sub_4010B0 endp
004010F3
```

This address is 32 bytes from the start of the resource

This address is 64 bytes from the start of the resource

This address is 80 bytes from the start of the resource

Dst is actually address 0x4043CC Unhelpfully mislabeled by IDA and4 bytes before 0x4043D0

Refresh – This gives up the filename and letter

1BpnGjHOT7h5vvZsV4vISSb60Xj3pX5G.exe

Edit As: Hex | Run Script | Run Template: EXE.bt

```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0123456789ABCDEF
2A90h: 00 62 01 00 7D 01 00 00 00 00 00 00 00 00 00 00    .b..}...........
2AA0h: 05 00 42 00 52 00 49 00 43 00 4B 00 00 00 00 00    ..B.R.I.C.K.....
2AB0h: 5A 00 49 00 6D 00 49 00 54 00 37 00 44 00 79 00    Z.I.m.I.T.7.D.y.
2AC0h: 43 00 4D 00 4F 00 65 00 46 00 36 00 00 00 00 00    C.M.O.e.F.6.....
2AD0h: B3 00 B0 00 B4 00 B1 00 B4 00 B4 00 B2 00 B1 00    ³.°.´.±.´.´.².±.
2AE0h: AB 00 F5 00 EB 00 E2 00 00 00 00 00 00 00 00 00    «.õ.ë.â.........
2AF0h: 6D 00 00 00 00 00 00 00 00 00 07 00 ED 47 00 00    m...........íG..
2B00h: 6C F1 94 DC 84 B1 C3 51 A3 EA 83 D3 13 78 DB 1E    lñ"Ü„±ÃQ£êƒÓ.xÛ.
2B10h: 9C 98 8D BF BA AE 35 15 78 82 31 83 1D AB 36 0E    œ˜.¿º®5.x‚1ƒ.«6.
```

Pass		XOR		ASCII	Char
B3	xor	85	=	36	6
B0	xor	85	=	35	5
B4	xor	85	=	31	1
B1	xor	85	=	34	4
B4	xor	85	=	31	1
B4	xor	85	=	31	1
B2	xor	85	=	37	7
B1	xor	85	=	34	4
AB	xor	85	=	2E	.
F5	xor	85	=	70	p
EB	xor	85	=	6E	n
E2	xor	85	=	67	g
6D	xor	1A	=	77	w

Select C:\Windows\system32\cmd.exe

```
E:\tmp>1BpnGjHOT7h5vvZsV4vISSb60Xj3pX5G.exe
What is the password?
ZImIT7DyCMOeF6
Everything is awesome!
65141174.png => w

E:\tmp>dir *.PNG
 Volume in drive E is VBOX_share
 Volume Serial Number is 0000-0802

 Directory of E:\tmp

10/04/2018  03:19 PM            18,413 65141174.png
               1 File(s)         18,413 bytes
               0 Dir(s)  103,112,105,984 bytes free

E:\tmp>
```

Automate for understanding and reproduction

```
import sys
import io
import os
from subprocess import Popen, PIPE, STDOUT

allFiles = os.listdir(".")
cwd = os.getcwd()
picArray = []

for currentFile in allFiles:
    if currentFile.endswith("exe"):
        # Open the current file
        with open(currentFile, mode='rb') as file:
            fileContent = file.read()
            index = 0x2ab0

            #
            # Collect the embedded password stopping at a null
            #
            password = ''
            while ord(fileContent[index]) > 0:
                password += fileContent[index]
                index += 2

            p = Popen(['%s/%s' % (cwd, currentFile)], stdout=PIPE, stdin=PIPE, stderr=PIPE)
            stdout_data = p.communicate(input=password)[0]

            #
            # Add the filename and letter to its own array and print
            #
            picArray.append(stdout_data.splitlines()[2])
            print '%s (%s) => %s' % (currentFile, password, stdout_data.splitlines()[2])

#
# Print just the file and corresponding letter sorted to make it easier
#
print "\n"
picArray.sort()
for entry in picArray:
    print entry
```

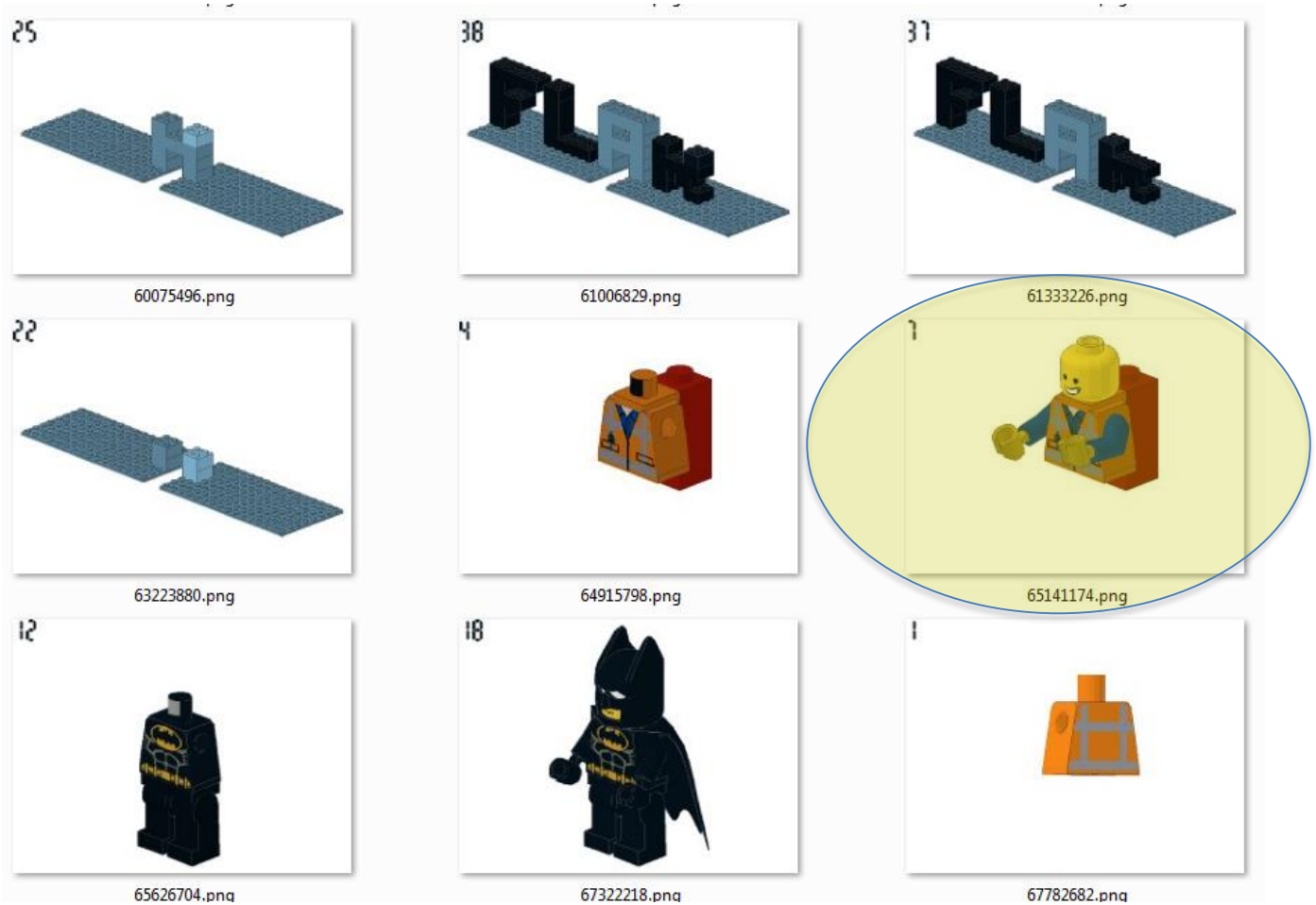
1. Iterate through directory
2. Offset to resource data
3. Read password from resource
4. Execute passing password
5. Collect filename and letter
6. Print sorted filenames

Script output complete with pictures (Of Text)

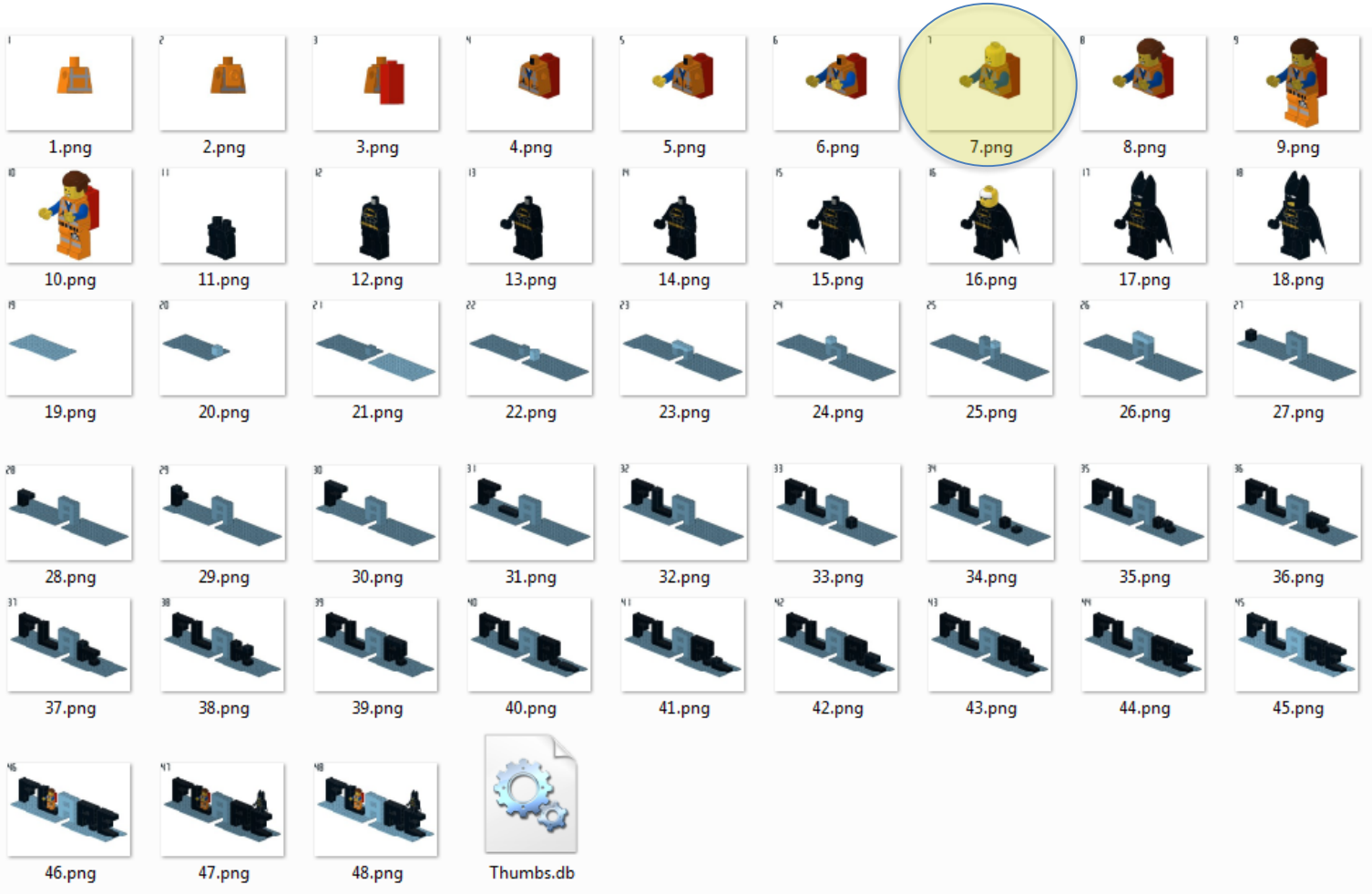
```
Select C:\Windows\system32\cmd.exe
E:\tmp>python test.py
B10Iv51T6wkpVCuy7jtcva7qka8WtLYY.exe (uLKEIRAEn) => 36870498.png => m
Ew93SSPDbgiQYo4E4035A16MJUxXegDW.exe (eoneTNuryZ3eF) => 42255131.png => t
y77GmQGdwVL7Fc9mMdiLJMgFQ8rgeSrl.exe (8Etmc0DAF8Qv) => 12268605.png => s
AEVYfSTJwubrlJKgxV8RA10AdZJ5vhhy.exe (UkuAJxmt8) => 58770751.png => o
xyjJcvGAgswB7Yno5e9qLF4i13L1iGoI.exe (gNbeYAjn) => 19343964.png => o
SeDdxvPJFHCr7uoQMjwmdRBAYEelHBZB.exe (ohj5W6Goli) => 65626704.png => 3
hajfdokqjogmoWfpyp4w0feoeyhs1QLo.exe (hqpNm7UJL) => 61333226.png => f
IXITujCLucnD4P3YrXOud5gC7Bwcw6mr.exe (aGUwVeVZ2c19mgE) => 80333569.png => o
NaobGsJ2w6qqblcIsj4QYNIBQhg3gmTR.exe (C446Zdun) => 47893007.png => _
wmkeAU8MdYrC9tEUMHH2tRMgaGdiFnga.exe (0rhvT5GX) => 51227743.png => a
u8mbI3GZ8WtwruEiFkI10UKxJS917407.exe (r6ZWNWeFadW) => 36494753.png => 0
eEJhUoNbuc40kLHRo8GB7bwFPkuhgaVN.exe (J1kj42jZsC9) => 71290032.png => a
dT4Xze8paLOG7srCdGLsbLE1s6m3EsfX.exe (dRnTVwZPjf0U) => 82100368.png => m
PvlqINbYjAY1E4WPfc2N6rZ2nKVhNZTP.exe (0d7qdvEhYGc) => 89295012.png => 0
3Jh0ELkck1MuRvzr8PLIpBNUGlspmGnu.exe (uVmH96JGdPkEBfd) => 75072258.png => r
dnAciAGVdlovQFSJmNiPOdHjkM3Ji18o.exe (ZYNGeumv6QuI7) => 15566524.png => e
xatgyd15cadiWFY4EXMRuoQr22ZIRC1Y.exe (8V9AzigUcb2J) => 52817899.png => n
x4neMBrqkYIQxDuXpwJNQZOlfyfA0eXs.exe (Fs3Ogu6W3qk59kZ) => 73903128.png => u
JXADoHafRHDyHmcTUjEBOvqq95spU7sj.exe (jZRmFmeIchneGS) => 30171375.png => s
eovBHrlDb809jf08yaAcSzcX4T37F1NI.exe (rXZE7pDx3) => 33718379.png => .
cWvFLblUfJl7KFDUYF1ABBFYFb6FJMz.exe (yu7hNshnpM4Vy) => 70037217.png => m
7mCysSKfiHJ4WqH2T8ERLE33Wrbp6Mqe.exe (Q9WdIAGjUKdNxr6) => 67322218.png => _
Bp7836noYu71VAWc27sUdfaGwieALfc2.exe (NcMkqwelbRu) => 47619326.png => p
SDIADRKhATsagJ3K8WwaNcQ52708TyRo.exe (5O2godXTZePdWZd) => 87730986.png => 0
azcyERV8HUbXmqPTEq5JFt7Ax1W5K4wl.exe (qNb6tr7n) => 88763595.png => e
lk0SOpnVIzTcCIDcou9R7prKAC3laX0k.exe (9eDMpbMSEeZ) => 33098947.png => _
gFZw71PU1bOXBvHRc31HJI5PKwy745Wv.exe (jZAorSlICuQa0g8) => 16295588.png => a
iJO15JsCa1bU5anXnZ9dTC9iWbEDmdtf.exe (2LUmPSYdxDcil) => 16785906.png => 4
MrA1JmEDfPhnTi5MNMhqVS8aaTKdxbMe.exe (auDB6HtMv) => 33662866.png => r
bmYBZTBJ1aFNbbwpiOiiQVdzimx8QVTI.exe (7kcuVMWeIBFGWfJ) => 72501159.png => c
aSfSVMn7B8eRtxgJgwPP5Y5HiDEidvKg.exe (b1VRfMTNPu) => 64915798.png => 3
zRx3bsMfOwG8IaayOeS8rHSSpiRfc9IB.exe (XgkvZJKe) => 47202222.png => n
JIdE7SESzC1aS58Wwe5j3i6XbpkCa3S6.exe (goTZP4go) => 37723511.png => n
u3PL12jk5jCZKiUm0omvh46yK7NDfZLT.exe (4z0gAyKdk) => 72562746.png => -
v6RkHsLya4wTAh71C65hMXBsTc1ZhGZT.exe (dEDDxJaxc1R) => 79545849.png => s
jJHgJjbyeWTTyQqISuJMpEGgE1aFs5ZB.exe (9aIZjTerf0) => 44958449.png => _
2AljFfLleprkThTHuUvg63I7OgjG2LQT.exe (UvCG4jaaIc4315) => 67782682.png => m
w3Y5YeglxqIWstp1PLbFoHvrQ9rN3F3x.exe (HQG0By9q) => 63223880.png => a
HDHugJBqTJqKKVtqi3sfR4BTq6P5XLZY.exe (45psrewIRS) => 13147895.png => w
BG3IDbHOUt9yHumPceLTVbObBHFneYEu.exe (KSL8EAnlIZin1gG) => 18376743.png => _
d4NlRo5umkvWhZ2FmEG32rXBNeSSLt2Q.exe (5xj9HmHyhF) => 18309310.png => @
E36RGTbCE4LDtyLi97191SFoR7xVMKGN.exe (dPVLAQ8LwmhH) => 60075496.png => s
kGQY35HJ7gvXzDJLWe8mabs3oKpwCo6L.exe (14bm9pHvbufOA) => 82236857.png => e
1JpPaUMynR9Gf1WbxfYvZviqiCB59RcI.exe (PylRCpDK) => 85934406.png => m
1BpnGjHOT7h5vvZsU4vISSb60Xj3pX5G.exe (ZImIT7DyCMOeF6) => 65141174.png => w
4ihY3RWK4WYqI4XOXLtAH6XU5lkoIdgv.exe (3nEiXqMnXG) => 16544936.png => e
P2PxxSJpnquBQ3xCvLoYj4pD3iyQcaKj.exe (nLSGJ2BdwC) => 61006829.png => l
K7HjR3Hf10SGG7rgke9WrRfxqhaGixS0.exe (Z8VCO7XbKUk) => 72263993.png => h
```

```
12268605.png => s
13147895.png => w
15566524.png => e
16295588.png => a
16544936.png => e
16785906.png => 4
18309310.png => @
18376743.png => _
19343964.png => o
30171375.png => s
33098947.png => _
33662866.png => r
33718379.png => .
36494753.png => 0
36870498.png => m
37723511.png => n
42255131.png => t
44958449.png => _
47202222.png => n
47619326.png => p
47893007.png => _
51227743.png => a
52817899.png => n
58770751.png => o
60075496.png => s
61006829.png => l
61333226.png => f
63223880.png => a
64915798.png => 3
65141174.png => w
65626704.png => 3
67322218.png => _
67782682.png => m
70037217.png => m
71290032.png => a
72263993.png => h
72501159.png => c
72562746.png => -
73903128.png => u
75072258.png => r
79545849.png => s
80333569.png => o
82100368.png => m
82236857.png => e
85934406.png => m
87730986.png => 0
88763595.png => e
89295012.png => 0
```

Windows Explorer – Icons reveal number in all



And the (F)Lego renamed tells a story – Order!



Refresh – The one part of the resource not used

1BpnGjHOT7h5vvZsV4vISSb60Xj3pX5G.exe

Edit As: Hex | Run Script | Run Template: EXE.bt

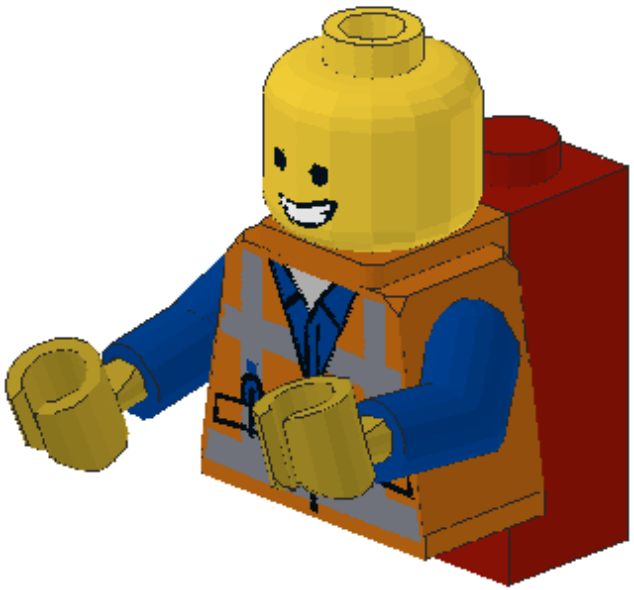
```
        0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F    0123456789ABCDEF
2A90h: 00 62 01 00 7D 01 00 00 00 00 00 00 00 00 00 00    .b..}...........
2AA0h: 05 00 42 00 52 00 49 00 43 00 4B 00 00 00 00 00    ..B.R.I.C.K.....
2AB0h: 5A 00 49 00 6D 00 49 00 54 00 37 00 44 00 79 00    Z.I.m.I.T.7.D.y.
2AC0h: 43 00 4D 00 4F 00 65 00 46 00 36 00 00 00 00 00    C.M.O.e.F.6.....
2AD0h: B3 00 B0 00 B4 00 B1 00 B4 00 B4 00 B2 00 B1 00    ³.°.´.±.´.´.².±.
2AE0h: AB 00 F5 00 EB 00 E2 00 00 00 00 00 00 00 00 00    «.õ.ë.â.........
2AF0h: 6D 00 00 00 00 00 00 00 00 00 07 00 ED 47 00 00    m...........íG..
2B00h: 6C F1 94 DC 84 B1 C3 51 A3 EA 83 D3 13 78 DB 1E    lñ"Ü„±ÃQ£êƒÓ.xÛ.
2B10h: 9C 98 8D BF BA AE 35 15 78 82 31 83 1D AB 36 0E    œ˜.¿°®5.x‚1ƒ.«6.
```

Pass		XOR		ASCII	Char
B3	xor	85	=	36	6
B0	xor	85	=	35	5
B4	xor	85	=	31	1
B1	xor	85	=	34	4
B4	xor	85	=	31	1
B4	xor	85	=	31	1
B2	xor	85	=	37	7
B1	xor	85	=	34	4
AB	xor	85	=	2E	.
F5	xor	85	=	70	p
EB	xor	85	=	6E	n
E2	xor	85	=	67	g
6D	xor	1A	=	77	w

Select C:\Windows\system32\cmd.exe

```
E:\tmp>1BpnGjHOT7h5vvZsV4vISSb60Xj3pX5G.exe
What is the password?
ZImIT7DyCMOeF6
Everything is awesome!
65141174.png => w

E:\tmp>dir *.PNG
 Volume in drive E is VBOX_share
 Volume Serial Number is 0000-0802

 Directory of E:\tmp

10/04/2018  03:19 PM            18,413 65141174.png
               1 File(s)         18,413 bytes
               0 Dir(s)  103,112,105,984 bytes free

E:\tmp>
```

65141174.png



A quick improvement to the script

```
            #
            index = 0x2ab0
            orderIndex = index + 0x4A

            #
            # Collect the embedded password stopping at a null
            #
            password = ''
            while ord(fileContent[index]) > 0:
                password += fileContent[index]
                index += 2

            #
            # Run the subprocess, stdin the password and let it unencrypt the picture
            #
            p = Popen(['%s/%s' % (cwd, currentFile)], stdout=PIPE, stdin=PIPE, stderr=PIPE)
            stdout_data = p.communicate(input=password)[0]

            #
            # Add the filename and letter to its own array
            #
            picArray.append((stdout_data.splitlines()[2], ord(fileContent[orderIndex])))

            #
            # Print our progress with the extracted passwords
            #
            print '%s (%s) => %s' % (currentFile, password, stdout_data.splitlines()[2])

#
# Print just the file and corresponding letter sorted to make it easier
# And gather the corect order to sort
#
print "\n"

flagDict = {}
picArray.sort()
for entry in picArray:
    print '%s at position %d' % (entry[0], entry[1])
    flagDict[entry[1]] = entry[0][-1]

#
# Now sort and print the flag
#
flag = ''
orderedList = flagDict.keys()
orderedList.sort()

for entry in orderedList:
    flag += flagDict[entry]

print "\n%s" % flag
```

1. Iterate through directory
2. Offset to resource data
3. Read password from resource
4. Read order index from resource
5. Execute passing password
   - Not needed anymore
6. Collect name, letter and order
7. Print name, password and letter
8. Print sorted names at positions
9. Work out the flag
10. Reveal the flag

Keep calm and submit it

```
58770751.png => o at position 47
60075496.png => s at position 25
61006829.png => l at position 38
61333226.png => f at position 37
63223880.png => a at position 22
64915798.png => 3 at position 4
65141174.png => w at position 7
65626704.png => 3 at position 12
67322218.png => _ at position 18
67782682.png => m at position 1
70037217.png => m at position 11
71290032.png => a at position 19
72263993.png => h at position 15
72501159.png => c at position 46
72562746.png => - at position 42
73903128.png => u at position 34
75072258.png => r at position 3
79545849.png => s at position 32
80333569.png => o at position 2
82100368.png => m at position 27
82236857.png => e at position 8
85934406.png => m at position 35
87730986.png => 0 at position 26
88763595.png => e at position 28
89295012.png => 0 at position 31

mor3_awes0m3_th4n_an_awes0me_p0ssum@flare-on.com

E:\tmp>
```

IDA – Functions renamed to wrap up

```
call    zeroMemoryInDataSection
call    loadResourceAndCopyToDataSection
test    eax, eax
jnz     short loc_401347
```

```
er hate you right now.\n"

security_check_cookie(x)
```

```
00401347
00401347                        loc_401347:
00401347 68 C4 31 40 00         push    offset aWhatIsThePassw ; "What is the password?\n"
0040134C E8 BF 01 00 00         call    writeToConsole
00401351 6A 10                  push    10h
00401353 8D 45 DC               lea     eax, [ebp+var_24]
00401356 50                     push    eax
00401357 68 F4 31 40 00         push    offset a15ls    ; "%15ls"
0040135C E8 5F 01 00 00         call    getUserinput
00401361 83 C4 10               add     esp, 10h
00401364 8D 4D DC               lea     ecx, [ebp+var_24]
00401367 E8 D4 FE FF FF         call    parseUserInput
0040136C 85 C0                  test    eax, eax
0040136E 75 1E                  jnz     short loc_40138E
```

```
                        failMatch:
30 32 40 00             push    offset aGoStepOnABrick ; "Go step on a brick!\n"
96 01 00 00             call    writeToConsole
C4 04                   add     esp, 4
C8 FF                   or      eax, 0FFFFFFFFh
4D FC                   mov     ecx, [ebp+var_4]
CD                      xor     ecx, ebp
21 03 00 00             call    @__security_check_cookie@4 ; __security_check_cookie(x)
E5                      mov     esp, ebp
                        pop     ebp
                        retn
```

```
0040138E
0040138E                        loc_40138E:
0040138E 8D 4D DC               lea     ecx, [ebp+var_24]
00401391 E8 1A FD FF FF         call    decryptPictureFilenameAndLetter
00401396 E8 65 FD FF FF         call    writePictureToDisk
0040139B 85 C0                  test    eax, eax
0040139D 75 1E                  jnz     short loc_4013BD
```

```
0040139F 68 2C 32 40 00         push    offset aOhLookARainbow ; "Oh look a rainbow.\n"
004013A4 E8 67 01 00 00         call    writeToConsole
004013A9 83 C4 04               add     esp, 4
004013AC 83 C8 FF               or      eax, 0FFFFFFFFh
004013AF 8B 4D FC               mov     ecx, [ebp+var_4]
004013B2 33 CD                  xor     ecx, ebp
004013B4 E8 F2 02 00 00         call    @__security_check_cookie@4 ; __security_check_cookie(x)
004013B9 8B E5                  mov     esp, ebp
```

```
004013BD
004013BD                        loc_4013BD:
004013BD 68 54 32 40 00         push    offset aEverythingIsAw ; "Everything
004013C2 E8 49 01 00 00         call    writeToConsole
004013C7 68 C0 43 40 00         push    offset unk_4043C0
004013CC 68 A0 43 40 00         push    offset unk_4043A0
004013D1 68 84 32 40 00         push    offset aSS_0    ; "%s => %s\n"
004013D6 E8 35 01 00 00         call    writeToConsole
```

Questions

Questions