Evolution of Rootkits

Samuel Chevet

samuel@lse.epita.fr
http://www.lse.epita.fr

20 July 2012

Why this talk ?

- VxStuff
- Fun
- Windows internal
- Share RE stuff

Malware

- Malicious software
- Disrupt computer operation
- Gather sensitive information
- Gain access

Packer

- Pack or Compress
- Avoid AV Detection
- Not like ZIP, or anything else
- Own decryption / loading stub
- Physical Size is usually smaller
- Resistant to casual JNZ -> JMP changes
- ...
- UPX
- FSG
- ...

Exploit pack

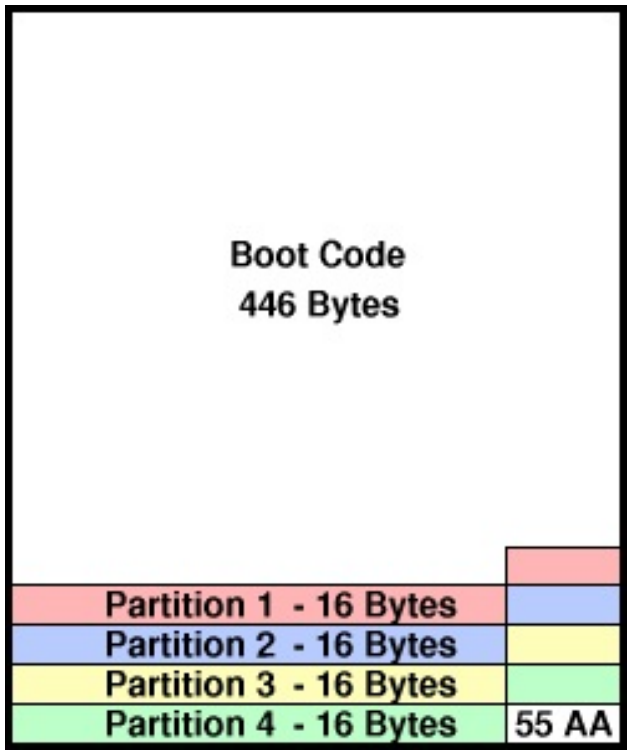
- Software toolkit
- Injected into hacked or malicious sites
- Test full of exploits on visitors
- Outdated software

Monitoring

- Irc bot
- Hash every hour
- Malware dropped
- Stat : each 5 / 6 hours repack malware
- Avoid AV detection

Master Boot Record

- First Sector hard drive
- 512 bytes
- 0x7C00



Volume Boot Record

- NTFS Boot Sector
- 512 bytes
- Ability to read FAT32 / NTFS
- Loading OS boot components
- Bootmgr, BCD, . . .

Moar

BootMGR

- 16bits stub
- Multiple checksum
- 32bits PE
- No External dependencies

BCD

- Boot Configuration Data
- Old boot.ini (NTDLR)
- bcdedit.exe
- Same format Windows Registry
- Menu entries

Rootkit

- Type of malicious software
- "root"
- "kit"
- Full control
- Kernel-mode
- Adding or replacing portions of OS
- Hide attacker's presence

Rootkit on 64 bits systems ?

- Super market or whatever
- Kernel protection
- Driver signing
- When you can't buy or steal certificate :)

Introduction

- 2005, 64 bits, XP / 2003 SP1
- Driver -> DPL 0
- Driver not design to modify kernel structure
- Some game protection abuse of it (Starforce v3, . . .)
- Malware, AntiVirus

- Periodically checksum
- CRITICAL_STRUCTURE_CORRUPTION
- 0x109

System Service Descriptor Table

- System Call
- Dispatch to appropriate function

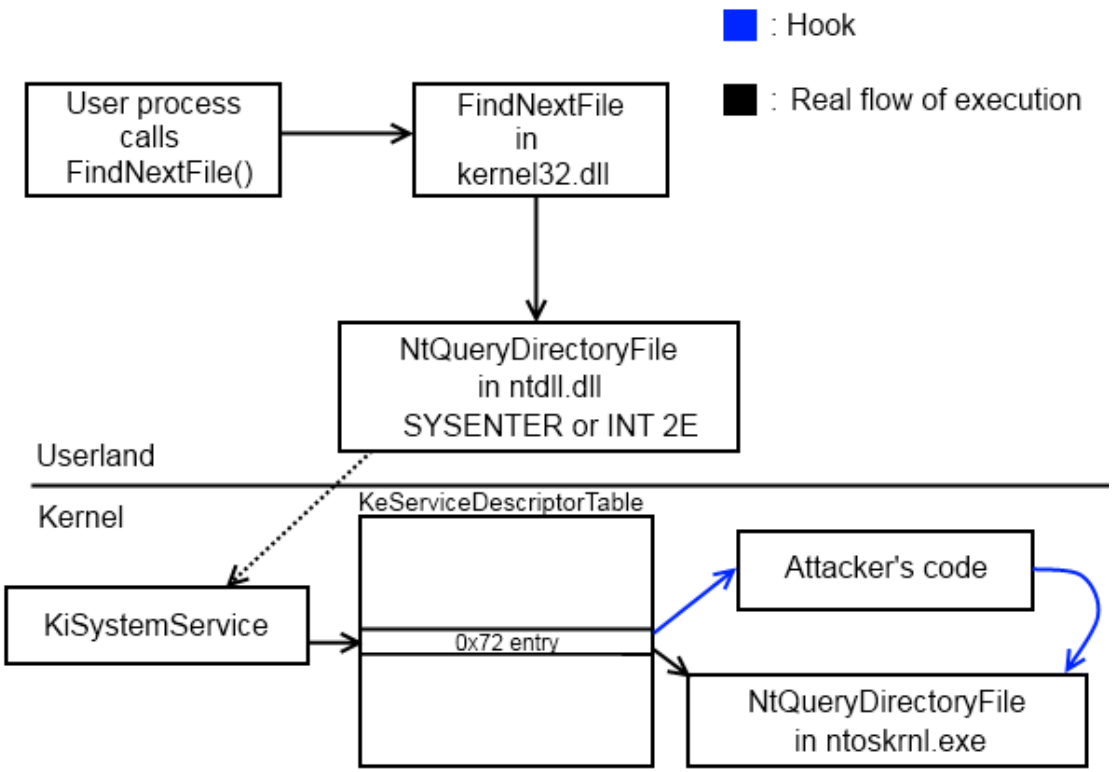
```
.text:7C91D090 ; Exported entry 123.  NtCreateFile
.text:7C91D090 ; Exported entry 933.  ZwCreateFile
.text:7C91D090
.text:7C91D090 ; =============== S U B R O U T I N E =======================================
.text:7C91D090
.text:7C91D090
.text:7C91D090 ; __stdcall NtCreateFile(x, x, x, x, x, x, x, x, x, x, x)
.text:7C91D090                 public _NtCreateFile@44
.text:7C91D090 _NtCreateFile@44 proc near
.text:7C91D090
.text:7C91D090                 mov     eax, 25h        ; NtCreateFile
.text:7C91D095                 mov     edx, 7FFE0300h
.text:7C91D09A                 call    dword ptr [edx]
.text:7C91D09C                 retn    2Ch
.text:7C91D09C _NtCreateFile@44 endp
.text:7C91D09C
```

SSDT 64bits

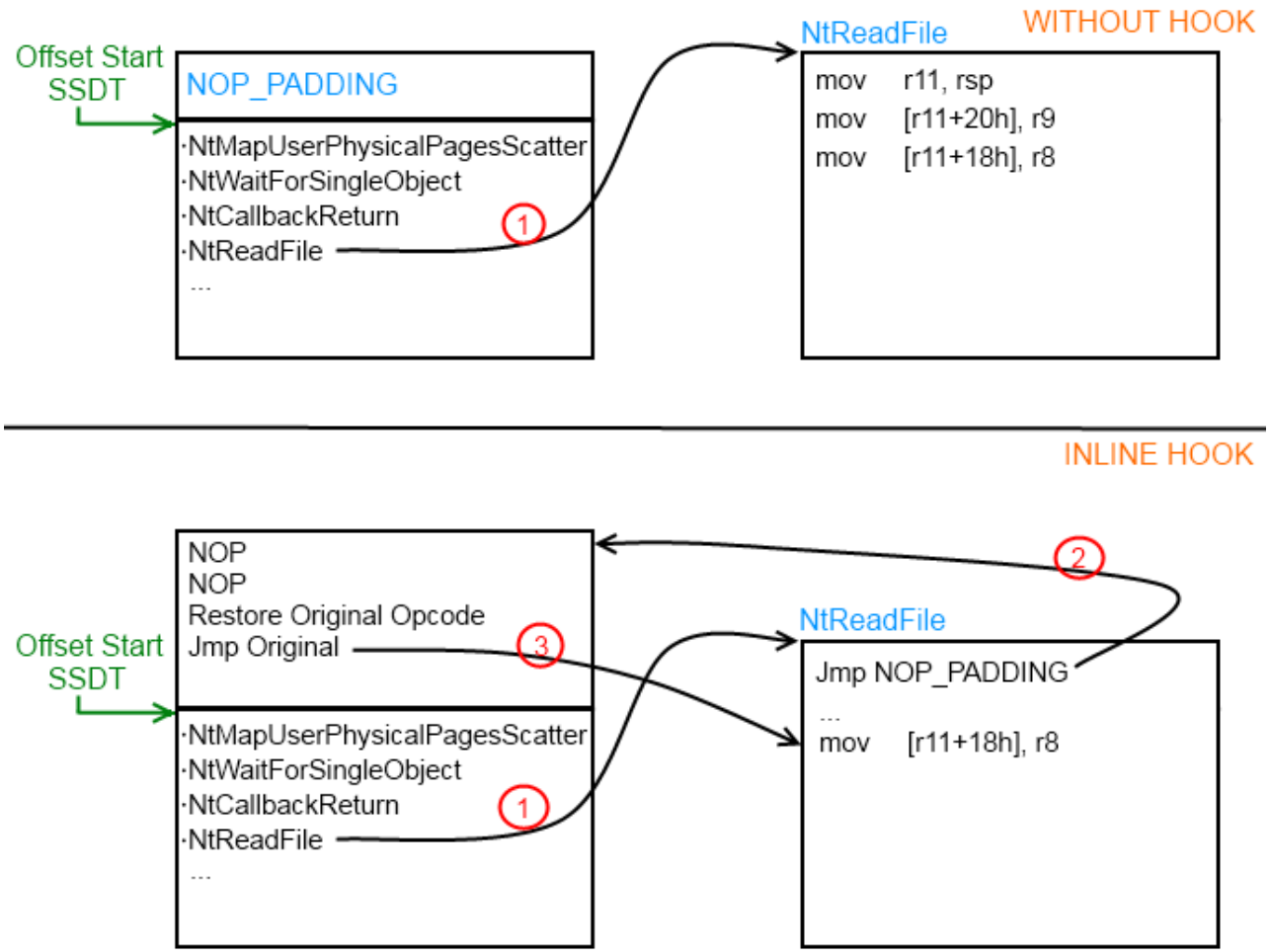
- No more exported
- But ...

```
.text:0000000140070FDE KiSystemServiceStart proc near              ; DATA XREF: KiServiceInternal+5A↑o
.text:0000000140070FDE                                             ; .data:00000001401E6618↓o
.text:0000000140070FDE                 mov     [rbx+1D8h] rsp
.text:0000000140070FE5                 mov     edi, eax
.text:0000000140070FE7                 shr     edi, 7
.text:0000000140070FEA                 and     edi, 20h
.text:0000000140070FED                 and     eax, 0FFFh
.text:0000000140070FF2                 lea     r10, KeServiceDescriptorTable
.text:0000000140070FF9                 lea     r11, KeServiceDescriptorTableShadow
```

Hook


: Hook
: Real flow of execution
User process calls FindNextFile()
FindNextFile in kernel32.dll
NtQueryDirectoryFile in ntdll.dll SYSENTER or INT 2E
Userland
Kernel
KeServiceDescriptorTable
KiSystemService
0x72 entry
Attacker's code
NtQueryDirectoryFile in ntoskrnl.exe

Inline Hook



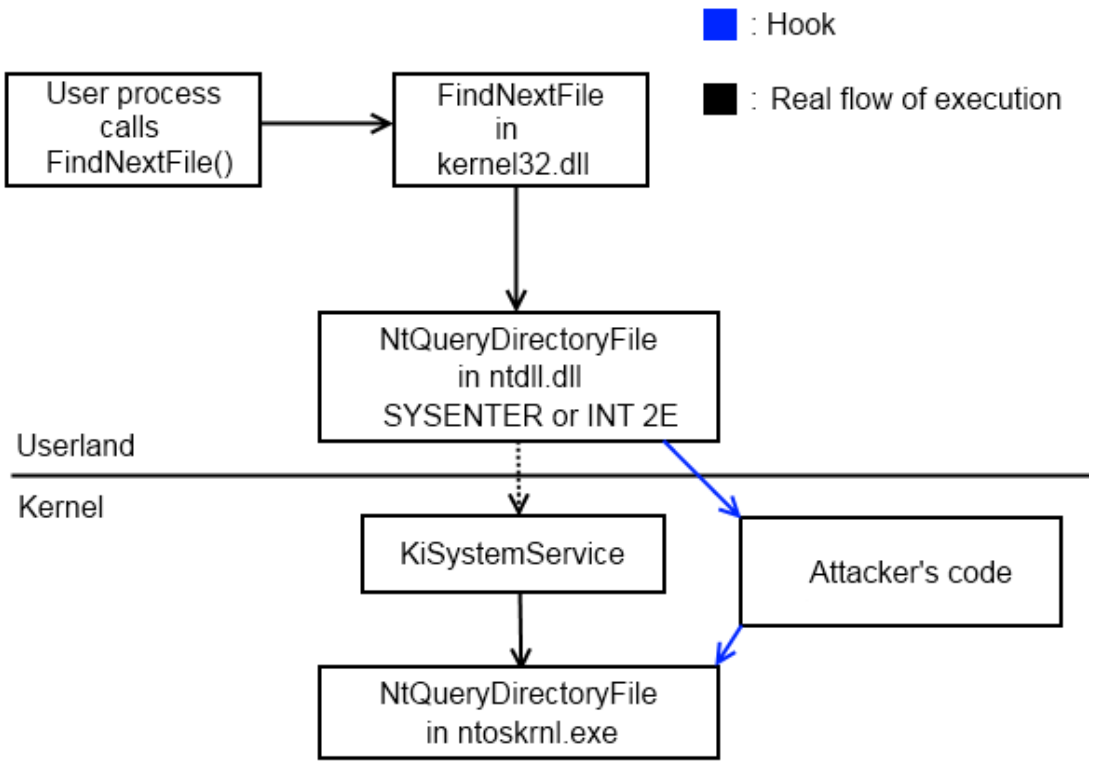
Global Descriptor Table

- Data structure
- Characteristics of various memory areas
- Base Adress
- Size
- Privilegies
- No more CallGate

Interrupt Descriptor Table

- Data structure
- 256 entry
- Manage interruption (Hardware or Software)
- 0x2E / SYSENTER

Interrupt Descriptor Table hook



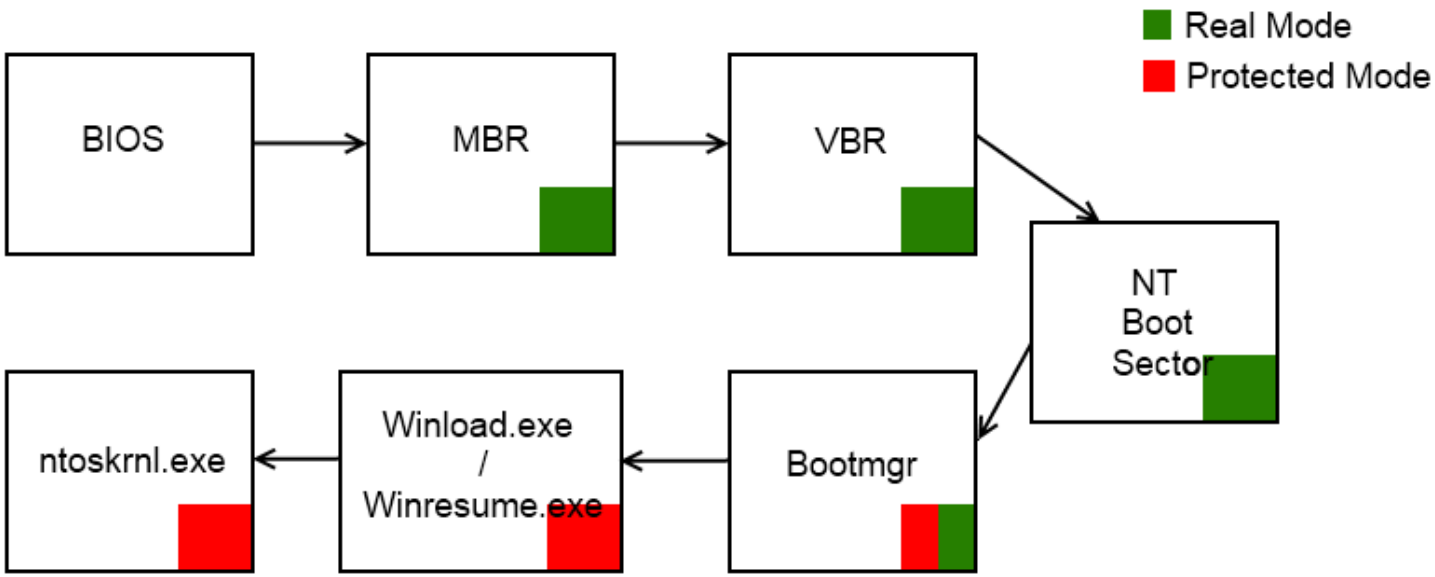
System Image

- ntoskrnl.exe
- Hardware Abstraction Layout
- ...

What is it ?

- Rootkit variant
- Replace legitime boot loader
- Loader persistant during transition to protected mode
- Kernel mode driver signing
- Hidden Storage

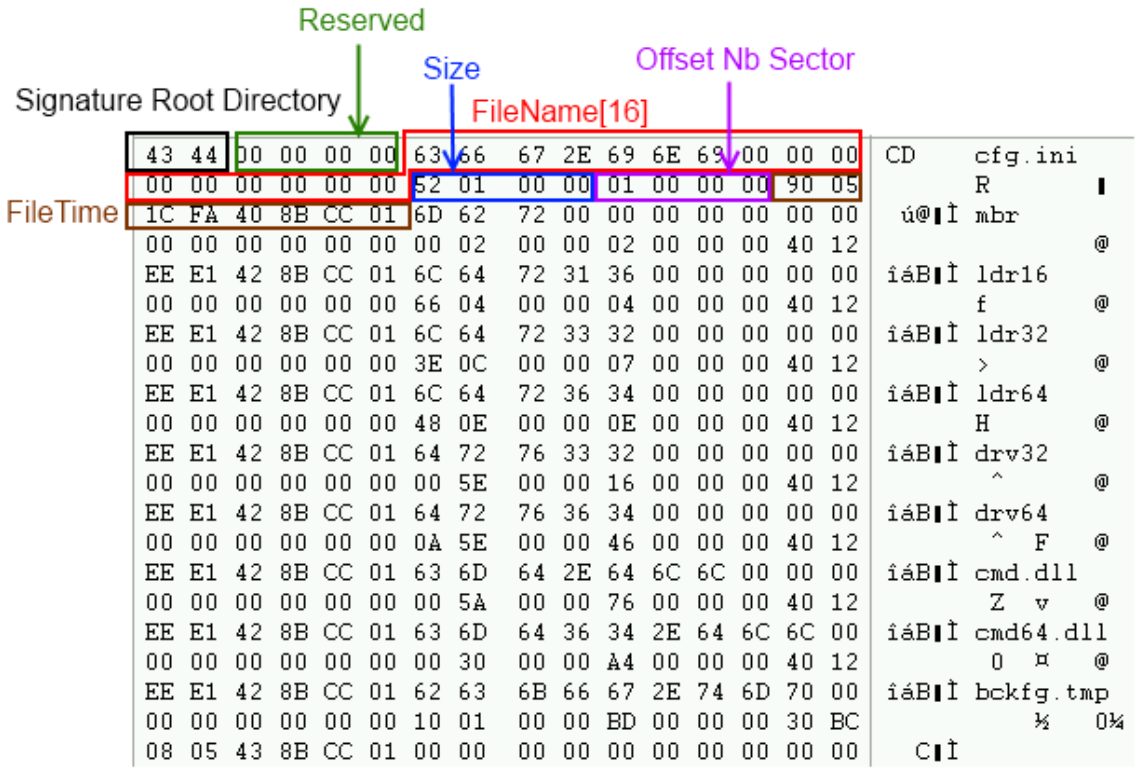
Boot Process



"It's not a bug it's a feature"

- Attack directly hard drive
- Hidden as possible
- Create own File System
- Usually encrypted (RC4, . . .)

TDL4 FS




```
43 44 00 00 00 00 63 66 67 2E 69 6E 69 00 00 00   CD    cfg.ini
00 00 00 00 00 00 52 01 00 00 01 00 00 00 90 05         R       ▌
1C FA 40 8B CC 01 6D 62 72 00 00 00 00 00 00 00    ú@▌Ì mbr
00 00 00 00 00 00 00 02 00 00 02 00 00 00 40 12                 @
EE E1 42 8B CC 01 6C 64 72 31 36 00 00 00 00 00   îáB▌Ì ldr16
00 00 00 00 00 00 66 04 00 00 04 00 00 00 40 12         f       @
EE E1 42 8B CC 01 6C 64 72 33 32 00 00 00 00 00   îáB▌Ì ldr32
00 00 00 00 00 00 3E 0C 00 00 07 00 00 00 40 12         >       @
EE E1 42 8B CC 01 6C 64 72 36 34 00 00 00 00 00   îáB▌Ì ldr64
00 00 00 00 00 00 48 0E 00 00 0E 00 00 00 40 12         H       @
EE E1 42 8B CC 01 64 72 76 33 32 00 00 00 00 00   îáB▌Ì drv32
00 00 00 00 00 00 00 5E 00 00 16 00 00 00 40 12          ^      @
EE E1 42 8B CC 01 64 72 76 36 34 00 00 00 00 00   îáB▌Ì drv64
00 00 00 00 00 00 0A 5E 00 00 46 00 00 00 40 12          ^  F   @
EE E1 42 8B CC 01 63 6D 64 2E 64 6C 6C 00 00 00   îáB▌Ì cmd.dll
00 00 00 00 00 00 00 5A 00 00 76 00 00 00 40 12          Z  v   @
EE E1 42 8B CC 01 63 6D 64 36 34 2E 64 6C 6C 00   îáB▌Ì cmd64.dll
00 00 00 00 00 00 00 30 00 00 A4 00 00 00 40 12          0  ¤   @
EE E1 42 8B CC 01 62 63 6B 66 67 2E 74 6D 70 00   îáB▌Ì bckfg.tmp
00 00 00 00 00 00 10 01 00 00 BD 00 00 00 30 BC              ½   0¼
08 05 43 8B CC 01 00 00 00 00 00 00 00 00 00 00     C▌Ì
```

Be my friend INT 0x13

- Real mode
- Hard disk, floppy disk
- Disk read, write services
- AH, Function table
- DL, Drive
- . . .
- Just setup an Hook

Required toolz

- Bochs / Qemu (Installation)
- IDA (+ plugins)
- Python
- Brain

Demo

This is demo time !

Another method (Cidox case)

- Not only MBR
- VBR

```
.text:00401D37                push    0Ah             ; lpType
.text:00401D39                push    offset Name     ; "VBR"
.text:00401D3E                xor     ebx, ebx
.text:00401D40                push    ebx             ; hModule
.text:00401D41                call    ds:FindResourceW
```

Another method (Cidox case)

```
.text:00402043
.text:00402043 check_mbr:
.text:00402043                 cmp     word ptr [ebx+1FEh], 0AA55h
.text:0040204C                 jnz     exit_func
.text:00402052                 lea     ecx, [ebx+1BEh]
.text:00402058                 xor     eax, eax
.text:0040205A                 mov     dl, 7
.text:0040205C                 lea     esp, [esp+0]
.text:00402060
.text:00402060 next_partition:                         ; CODE XREF: sub_401FD0+A3↑j
.text:00402060                 test    byte ptr [ecx], 80h
.text:00402063                 jz      short not_bootable
.text:00402065                 cmp     [ecx+4], dl
.text:00402068                 jz      short is_NTFS
.text:0040206A
.text:0040206A not_bootable:                           ; CODE XREF: sub_401FD0+93↓j
.text:0040206A                 add     eax, 1
.text:0040206D                 add     ecx, 10h
.text:00402070                 cmp     eax, 4
.text:00402073                 jb      short next_partition
```

Another Method (Cidox case)

```
.text:004020B3                 mov     ecx, offset aNtfs ; "NTFS "
.text:004020B8                 lea     edx, [ebx+3]
.text:004020BB                 jmp     short loc_4020C0
.text:004020BD ; -------------------------
.text:004020BD                 lea     ecx, [ecx+0]
.text:004020C0
.text:004020C0 loc_4020C0:                             ; CODE XREF: sub_401FD0+EB↓j
.text:004020C0                                         ; sub_401FD0+102↑j
.text:004020C0                 mov     esi, [edx]
.text:004020C2                 cmp     esi, [ecx]
```

Another Method (Cidox case)

control to it. At the same time the original contents of Extended NTFS IPL are encrypted saved and added to the end of the malicious code.

WRONG !

- aPLib
- LDE (Little disassembly engine)

That's it ?

- Disable / Bypass PatchGuard
- Hook INT13
- Scan for BOOTMGR signature

That's it ?

- Last 16bits stuff
- Before BmMain()

```
seg000:0A5A sub_A5A proc near               ; CODE XREF: sub_56C+1C4↓p
seg000:0A5A
seg000:0A5A OEP = dword ptr 4
seg000:0A5A arg_4 = dword ptr 8
seg000:0A5A
seg000:0A5A                 enter   0, 0
seg000:0A5E                 mov     ebx, [bp+OEP]
seg000:0A62                 mov     edx, [bp+arg_4]
seg000:0A66                 xor     ecx, ecx
seg000:0A69                 mov     cx, ds:word_1536
seg000:0A6D                 shl     ecx, 4
seg000:0A71                 mov     ebp, ecx
seg000:0A74                 xor     ecx, ecx
seg000:0A77                 mov     cx,0A9Ah
seg000:0A7A                 add     ebp, ecx
seg000:0A7D                 mov     cx,30h; '0'
seg000:0A80                 mov     ss, cx
seg000:0A82                 assume ss:nothing
seg000:0A82                 mov     ds, cx
seg000:0A84                 assume ds:nothing
seg000:0A84                 mov     es, cx
seg000:0A86                 assume es:nothing
seg000:0A86                 mov     esp, 61FFCh
seg000:0A8C                 push    edx
seg000:0A8E                 push    ebp
seg000:0A90                 xor     ebp, ebp
seg000:0A93                 push    large 20h; ' '
seg000:0A96                 push    ebx
seg000:0A98                 retfd
seg000:0A98 sub_A5A endp
```

That's it ?

- No self verification
- Extracted binary is not modified at the moment
- BmLaunchBootEntry()
- Too much difference

That's it ?

md5sum

d6ae2d5521dd93aebc90d411d099fa36	Professional_SP0/bootmgr
9e722b768e33d26ad8fa7d642e707443	Professional_SP0/ntoskrnl.exe
87b2086d7382a42935d55ec69e5e71ab	Professional_SP0/winload.exe
259525cfb422e6ac8e87bc9777b1df73	Professional_SP1/bootmgr
c6cec3e6cc9842b73501c70aa64c00fe	Professional_SP1/ntoskrnl.exe
e2f68dc7fbd6e0bf031ca3809a739346	Professional_SP1/winload.exe

That's it ?

- Last Step, take hand before KiSystemStartup()
- OslArchTransferToKernel()
- Enjoy :)
- Do whatever you want PatchGuard will not check you !

- Modify ntoskrnl header
- ProcessNotifyCallback
- Block AV process
- Inject payload in browsers
- aPLIB too !

Plite Bootkit

- Loader 16, 32
- Parse FAT & NTFS
- Replace explorer.exe
- Restore it

And more and more !

Self - Defense

- Protect MBR, VBR, Hidden File System, . . .
- No SSDT's Hook
- No DKOM
- IRP (I/O Request Packet)
- \Device\Harddisk
- IRP_MJ_INTERNAL_DEVICE_CONTROL

Security Features

- UEFI Secure Boot
- TPM
- Anti - malware launch module

Conclusion

- Load unsigned driver
- Target especially MBR and VBR
- Windows 8 introduced new security features
- Sophisticated ?
- Not really used by Cybercriminal

Questions ?

Thank you for your attention

- @w4kfu
- blog.w4kfu.com
- samuel@lse.epita.fr