Deactivate the Rootkit

Anibal Sacco

Alfredo A. Ortega

CORE SECURITY TECHNOLOGIES

History:

2004: The BIOS size of 60% of all notebooks suffered an increase of 25Kb

- Fast forward 5 years, 2009:
  - We were trying to install our own BIOS rootkit (Persistent BIOS Infection Talk, CanSecWest / Syscan)
  - We found that there was something already there!

What is the rootkit?

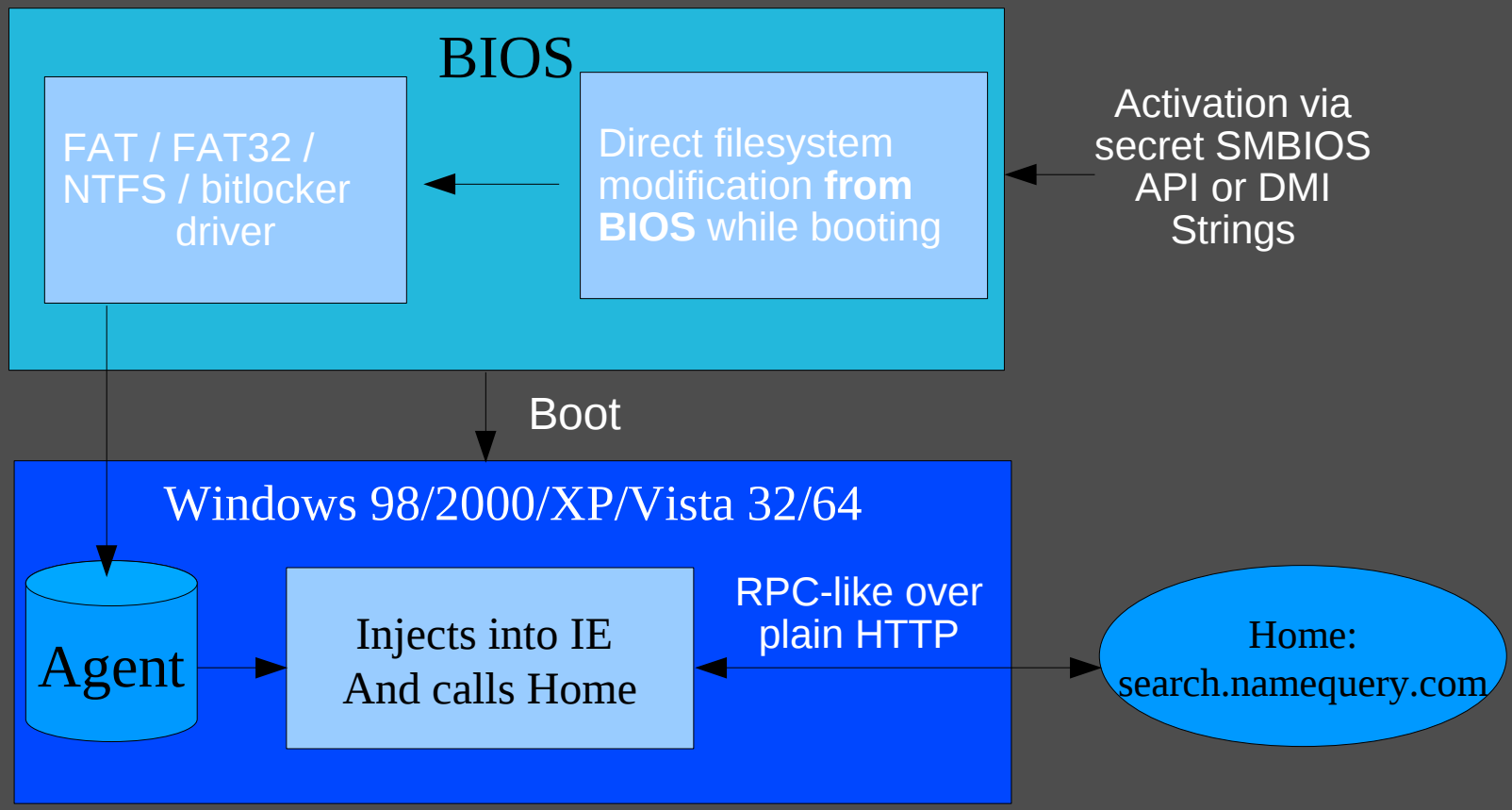
- Absolute Corp. Computrace, Anti-theft agent
- Option ROM Embedded in Phoenix BIOS
- Agreements with law enforcement agencies.
- Inside notebooks from HP, Dell, Lenovo, Toshiba, Gateway, Asus, Panasonic, and more.

Option ROM header:

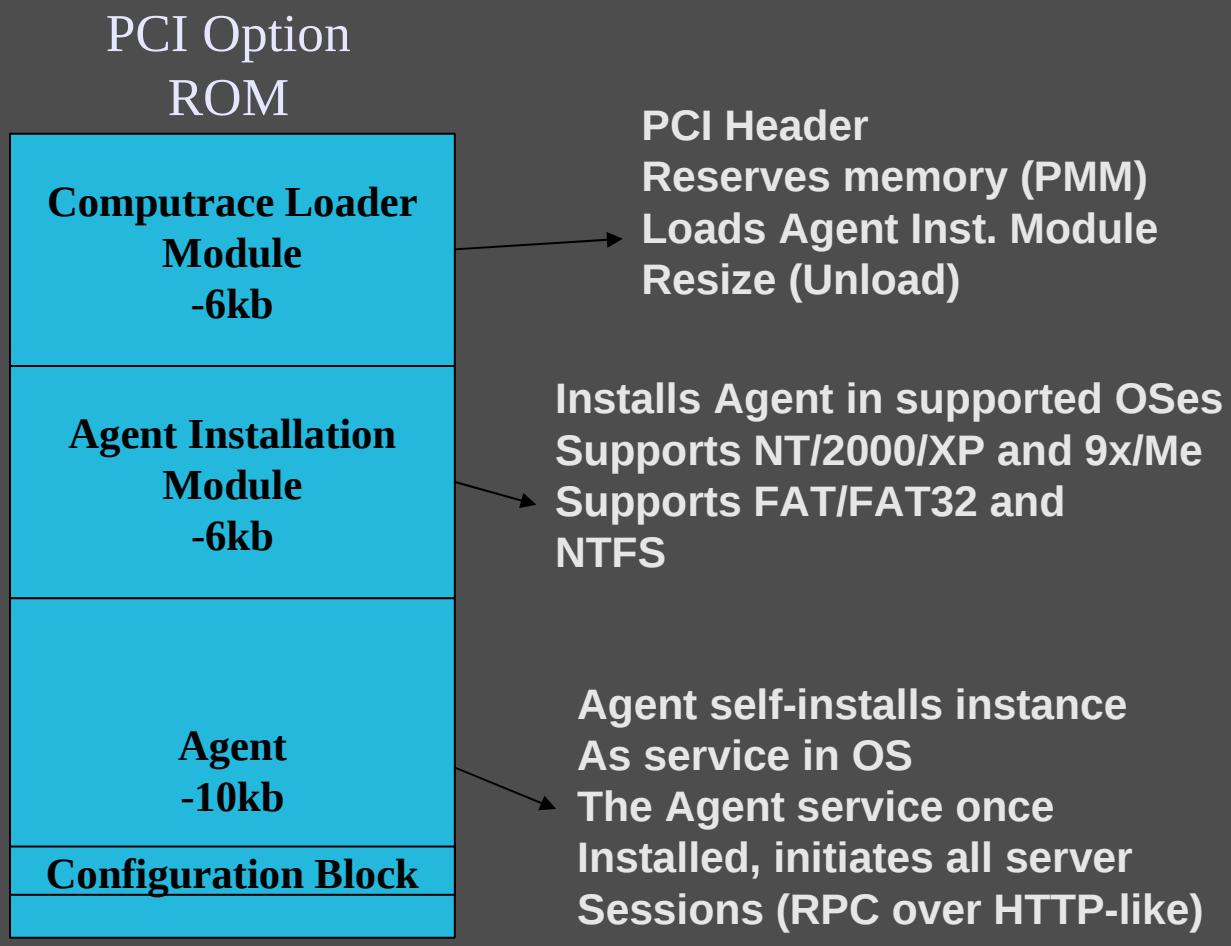
```
00000000  55 aa 2a eb 15 43 6f 6d  70 75 54 72 61 63 65 20  |U.*..CompuTrace |
00000010  56 38 30 2e 38 36 36 78  1d 00 e9 5c 01 50 43 49  |V80.866x...\.PCI|
00000020  52 17 19 34 12 00 00 18  00 00 06 00 00 2a 00 00  |R..4.........*..|
```

Basic Inner workings:

- See patent application US 2006/0272020 A1



Basic Inner workings:



Problems found:

- Huge privacy risk (bad/no authentication)
- Anyone could activate it with enough privileges
- Anyone can change the configuration
- Anyone can de-activate it (at least in certain known cases)
- Whitelisted by AV (potentally indetectable)

More problems found:

- Use of URL instead of IP (hosts redirection)
- Configuration block modification:

Demo if there is time...

```
Configuration block XOR 0xB5:
00000000  b1 b7 b5 b5 35 ab b1 b4  b5 f5 b4 aa b1 b5 b5 b5  |....5...........|
00000010  b5 a5 bf 41 41 30 49 4e  30 30 30 30 30 95 b1 1f  |...AA0IN00000...|
00000020  ee 30 86 a0 b1 8b b5 35  b5 ac ae 4a 4a 4a 4a 4a  |.0.....5...JJJJJ|
00000030  4a 4a 4a 4a 4a 4a 4a 4a  4a 4a 4a 4a 4a 4a 4a 4a  |JJJJJJJJJJJJJJJJ|
00000040  4a 4a 4a 4a 4a 4a af b4  35 ae b3 b5 b5 b5 b5 b5  |JJJJJJ..5.......|
00000050  b5 a8 b7 b5 b5 f3 b3 b5  b5 b5 b5 b5 b5 f2 b3 b5  |................|
00000060  b5 b5 b5 b5 b5 fd af 00  50 d1 35 71 17 73 65 61  |........P.5q.sea|
00000070  72 63 68 2e 6e 61 6d 65  71 75 65 72 79 2e 63 6f  |rch.namequery.co|
00000080  6d bf b7 b2 a5 b3 b3 ac  35 b4 b4 b5 b5 b2 b3 b5  |m.......5.......|
00000090  b5 b5 b5 b5 4a 98 b4 0d  98 b4 0d 9e b1 41 54 44  |....J........ATD|
000000a0  54 81 b7 38 2c 80 b7 39  2c 82 b2 39 2c 39 31 38  |T..8,..9,..9,918|
```

Stub agent: Unauthenticated BIOS code execution

Second Stage (AIM) loader, Stub Agent (DELL Vostro 1510 Computrace V 70.785)

```
seg000:01CF sub_1CF         proc near               ; CODE XREF: sub_27F+20↓p
seg000:01CF                 push    cx
seg000:01D0                 pop     es
seg000:01D1                 assume es:nothing
seg000:01D1                 mov     si, 0BFh ; '+'
seg000:01D4                 mov     [si+6], cx
seg000:01D7                 mov     dl, 80h ; 'Ç'
seg000:01D9                 mov     ah, 42h ; 'B'
seg000:01DB                 int     13h             ; DISK -
seg000:01DD                 push    es
seg000:01DE                 pop     ds
seg000:01DF                 jnb     short loc_1E2
seg000:01E1
seg000:01E1 locret_1E1:                             ; CODE XREF: sub_1CF+1B↓j
seg000:01E1                                         ; sub_1CF+72↓j
seg000:01E1                 retn
seg000:01E2 ; ---------------------------------------------------------------------------
seg000:01E2
seg000:01E2 loc_1E2:                                ; CODE XREF: sub_1CF+10↑j
seg000:01E2                 xor     ecx, ecx
seg000:01E5
seg000:01E5 loc_1E5:                                ; CODE XREF: sub_1CF+2D↓j
seg000:01E5                                         ; sub_1CF+33↓j ...
seg000:01E5                 inc     cl
seg000:01E7                 cmp     cl, 3Eh ; '>'
seg000:01EA                 ja      short locret_1E1
seg000:01EC                 mov     ebx, ecx
seg000:01EF                 shl     bx, 9
seg000:01F2                 lea     bx, [bx+7E00h]
seg000:01F6                 movzx   eax, byte ptr [bx]
seg000:01FA                 cmp     al, 3Eh ; '>'
seg000:01FC                 ja      short loc_1E5
seg000:01FE
seg000:01FE loc_1FE:                                ; CODE XREF: sub_27F+33↓J
seg000:01FE                                         ; DATA XREF: sub_27F+30↓o
seg000:01FE                 cmp     eax, [bx+4]
seg000:0202                 jbe     short loc_1E5
seg000:0204                 cmp     ecx, [ebx+eax*4]
seg000:0209                 jnz     short loc_1E5
seg000:020B                 cmp     eax, [ebx+eax*4+4]
seg000:0211                 jnz     short loc_1E5
seg000:0213                 mov     dx, [bx+2]
seg000:0216                 movzx   ebp, byte ptr [bx+1]
seg000:021B                 mov     si, bp
seg000:021D                 lea     bp, [ebx+ebp*4+4]
seg000:0222                 lea     bx, [ebx+eax*4-4]
```

```
seg000:0227                 mov     di, bx
seg000:0229                 sub     di, bp
seg000:022B                 shr     di, 2
seg000:022E                 add     di, si
seg000:0230                 inc     di
seg000:0231                 inc     di
seg000:0232                 cmp     di, ax
seg000:0234                 jnz     short loc_1E5
seg000:0236                 shl     edx, 10h
seg000:023A
seg000:023A loc_23A:                                ; CODE XREF: sub_1CF+A6↓j
seg000:023A                 mov     esi, [bx]
seg000:023D                 cmp     esi, 3Eh ; '>'
seg000:0241                 ja      short locret_1E1
seg000:0243                 shl     si, 9
seg000:0246                 lea     si, [si+7E00h]
seg000:024A                 mov     di, bx
seg000:024C                 sub     di, bp
seg000:024E                 shr     di, 2
seg000:0251                 dec     di
seg000:0252                 shl     di, 9
seg000:0255                 lea     di, [di+100h]
seg000:0259                 mov     cx, 200h
seg000:025C
seg000:025C loc_25C:                                ; CODE XREF: sub_1CF+9F↓j
seg000:025C                 lodsb
seg000:025D                 xor     dh, al
seg000:025F                 mov     ah, 8
seg000:0261
seg000:0261 loc_261:                                ; CODE XREF: sub_1CF+9C↓j
seg000:0261                 shl     dx, 1
seg000:0263                 jnb     short loc_269
seg000:0265                 xor     dx, 1021h
seg000:0269
seg000:0269 loc_269:                                ; CODE XREF: sub_1CF+94↑j
seg000:0269                 dec     ah
seg000:026B                 jnz     short loc_261
seg000:026D                 stosb
seg000:026E                 loop    loc_25C
seg000:0270                 sub     bx, 4
seg000:0273                 cmp     bx, bp
seg000:0275                 jnz     short loc_23A
seg000:0277                 shld    eax, edx, 10h
seg000:027C                 sub     ax, dx
seg000:027E                 retn
seg000:027E sub_1CF         endp
```

Detecting the Rootkit Agent

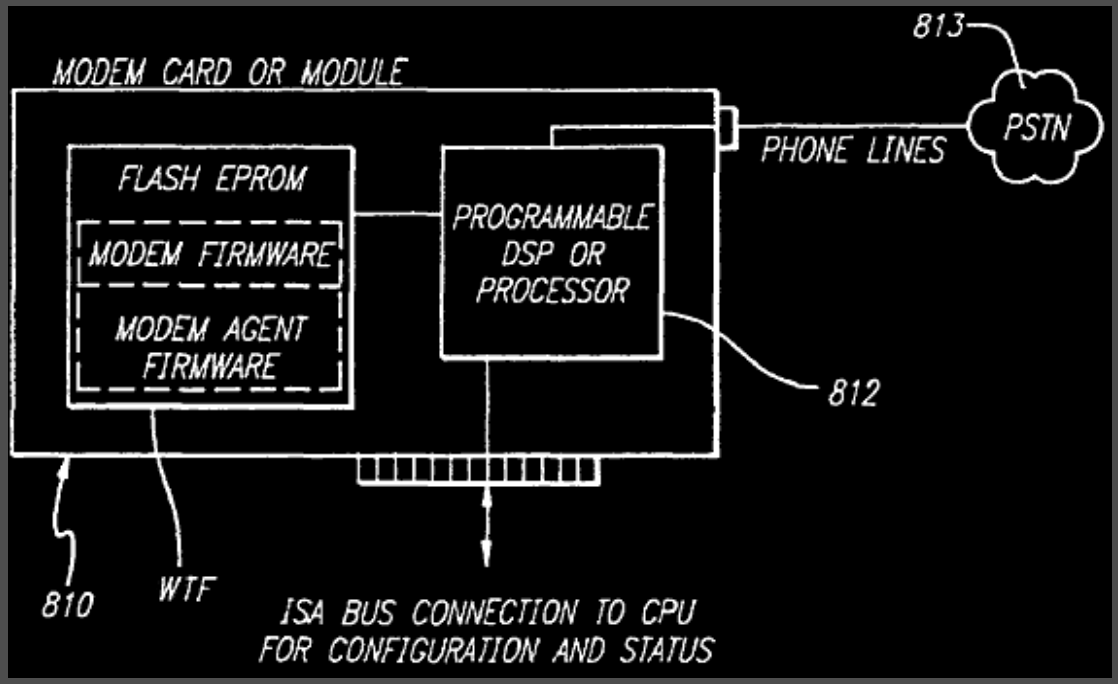
- A single file to look for:
  - system32\rpcnet.exe (Normal Agent)
  - system32\rpcnetp.exe (BIOS Persistent Agent)
- A service called ”Remote Procedure Call (RPC) Net” with no description
- Outgoing connections to search.namequery.com (209.53.113.223)
- Our Computrace Option Rom Dumper tool

Deactivating:

- Easiest way: hosts file redirection
- Modifying BIOS (only **unsigned BIOS!**)
- Modifying configuration block (Registry, hard-disk, etc.)
- Modifying nvram, then full HD Wipe.

The Past:

- US 6,300,863 B1 Pat. Figure 8A
- Filed Mar 24 **1998**, Absolute Corporation
- Agent inside modem Option ROM
- Support for DOS Backdooring



See “Implementing and Detecting a PCI Rootkit”, Heasman, BlackHat **2007**

The Future:

- Phoenix Failsafe:
  - Inside SMM, sounds familiar?
  - Always-on OS-independent, Wifi and GPS tracking
  - It has “safe” in the name instead of “trace”
- Intel Anti-theft technology:
  - vPro technology
  - Using AMT secondary processor
  - Works even with the notebook turned off!
- Other security aplications residing in BIOS

Strong authentication: *“Trust us, is for your own protection”***.**

This is only the begginning

- More research is needed in this area!
- CoreBoot (LinuxBIOS) project, is computrace-free
- Questions?
- Thanks! Now if you'll just look into the light: