ENJOY SAFER TECHNOLOGY™

Malware Buried Deep Down the SPI Flash: Sednit's First UEFI Rootkit Found in the Wild

Jean-Ian Boutin | Senior Malware Researcher

Frédéric Vachon | Malware Researcher

Jean-Ian Boutin

Senior Malware Researcher

@jiboutin

Frédéric Vachon

Malware Researcher

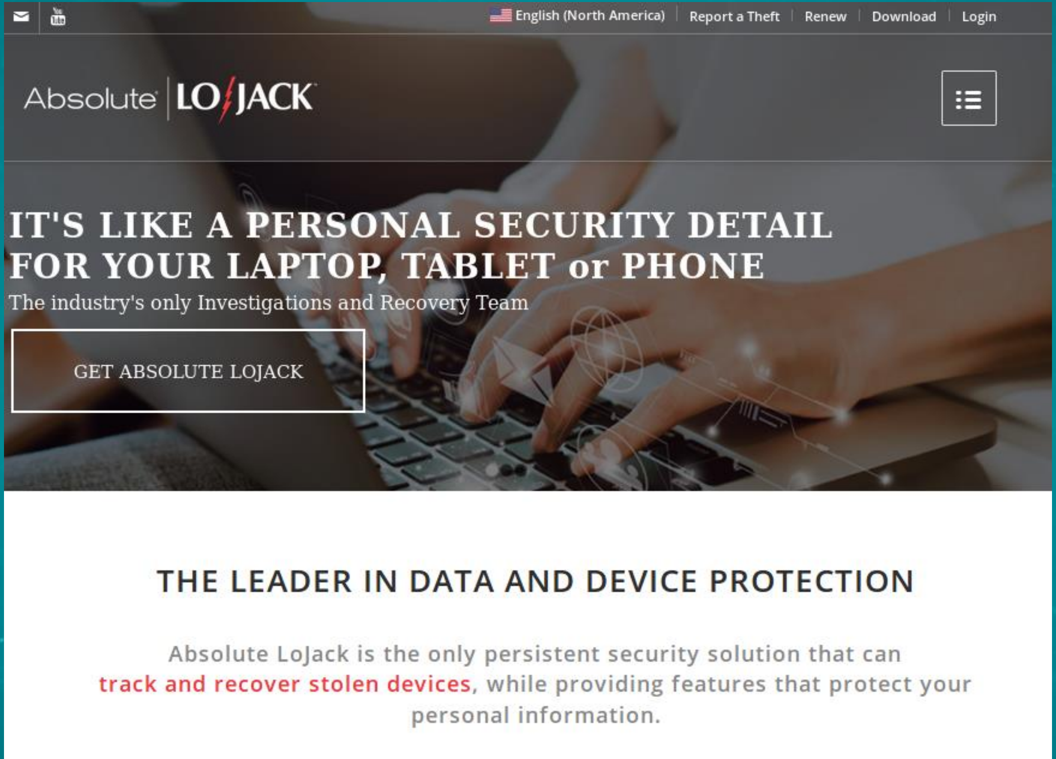
@Freddrickk_

Agenda

- What is LoJack?
- Past research
- Digging in
- Descending through the rings

Computrace/LoJack

Absolute Software



English (North America) | Report a Theft | Renew | Download | Login

Absolute | LOJACK

IT'S LIKE A PERSONAL SECURITY DETAIL FOR YOUR LAPTOP, TABLET or PHONE

The industry's only Investigations and Recovery Team

GET ABSOLUTE LOJACK

THE LEADER IN DATA AND DEVICE PROTECTION

Absolute LoJack is the only persistent security solution that can track and recover stolen devices, while providing features that protect your personal information.

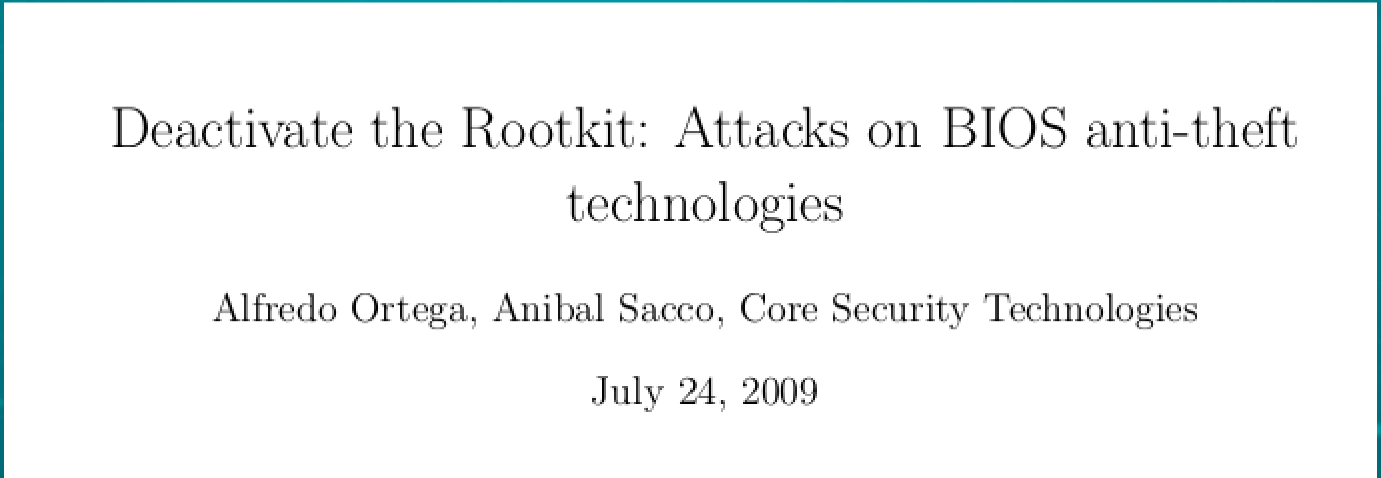
LoJack capabilities in a nutshell

- Locate
- Lock
- Delete
- Recover

Past Research

Black Hat USA 2009

- Exposed design vulnerabilities in agent

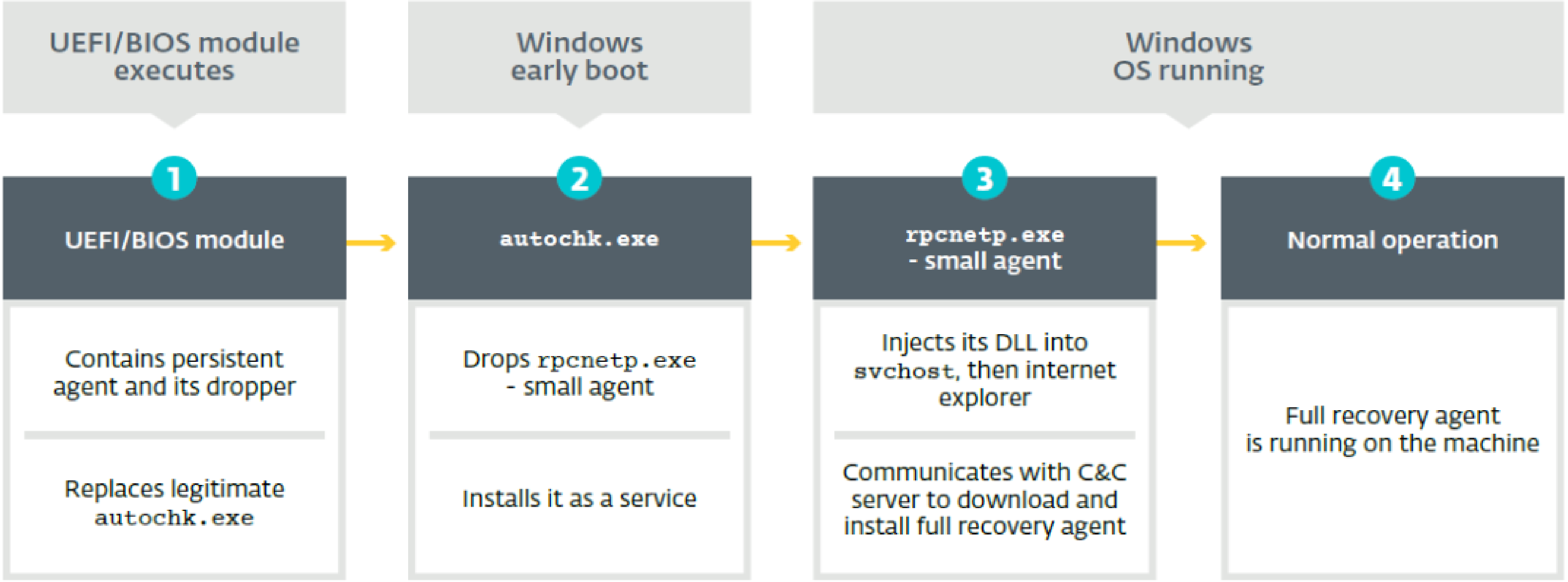


Deactivate the Rootkit: Attacks on BIOS anti-theft technologies

Alfredo Ortega, Anibal Sacco, Core Security Technologies

July 24, 2009

LoJack Architecture back then



UEFI/BIOS module executes

Windows early boot

Windows OS running

1

UEFI/BIOS module

Contains persistent agent and its dropper

Replaces legitimate autochk.exe

2

autochk.exe

Drops rpcnetp.exe - small agent

Installs it as a service

3

rpcnetp.exe - small agent

Injects its DLL into svchost, then internet explorer

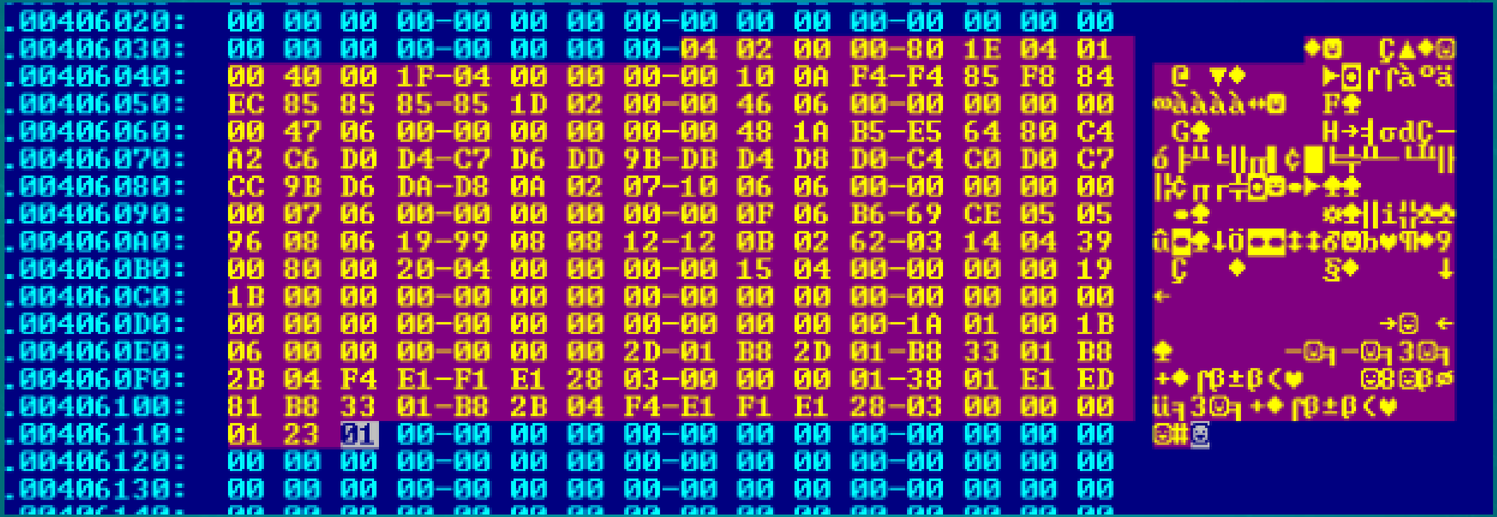
Communicates with C&C server to download and install full recovery agent

4

Normal operation

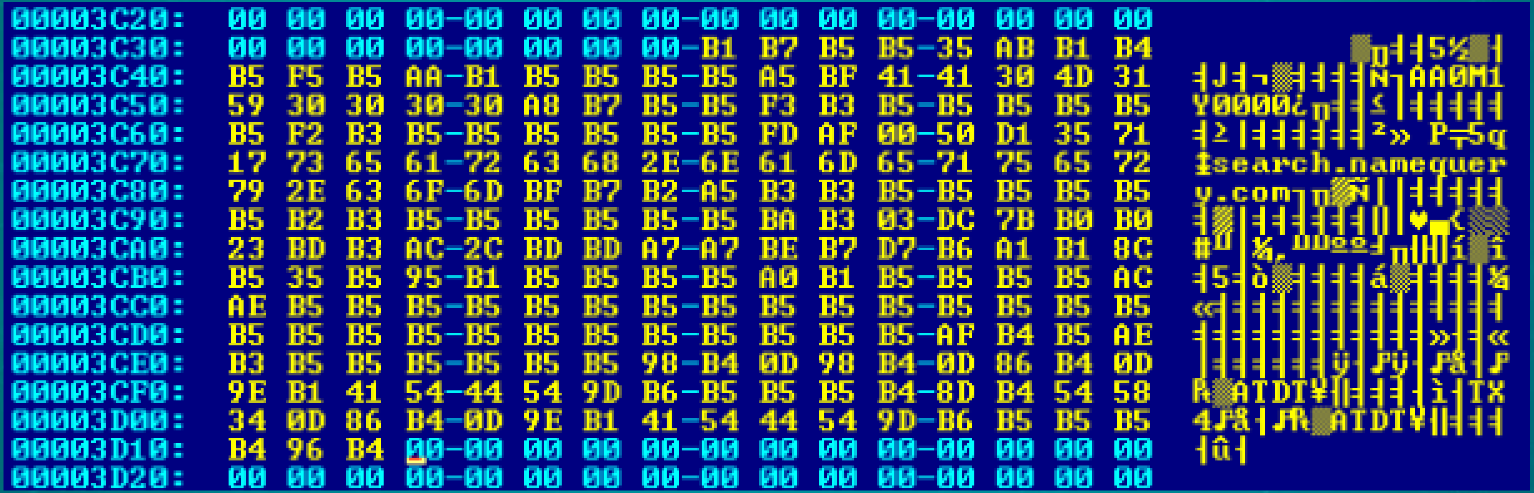
Full recovery agent is running on the machine

Configuration file vulnerability



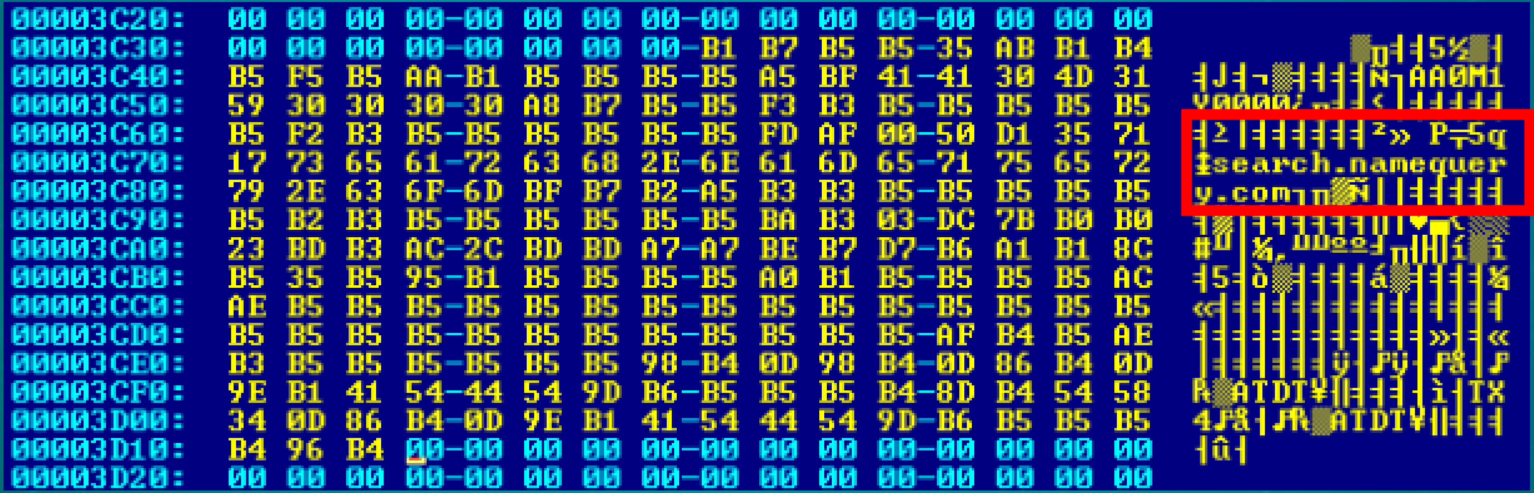
```
.00406020:   00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
.00406030:   00 00 00 00-00 00 00 00-04 02 00 00-80 1E 04 01
.00406040:   00 40 00 1F-04 00 00 00-00 10 0A F4-F4 85 F8 84
.00406050:   EC 85 85 85-85 1D 02 00-00 46 06 00-00 00 00 00
.00406060:   00 47 06 00-00 00 00 00-00 48 1A B5-E5 64 80 C4
.00406070:   A2 C6 D0 D4-C7 D6 DD 9B-DB D4 D8 D0-C4 C0 D0 C7
.00406080:   CC 9B D6 DA-D8 0A 02 07-10 06 06 00-00 00 00 00
.00406090:   00 07 06 00-00 00 00 00-00 0F 06 B6-69 CE 05 05
.004060A0:   96 08 06 19-99 08 08 12-12 0B 02 62-03 14 04 39
.004060B0:   00 80 00 20-04 00 00 00-00 15 04 00-00 00 00 19
.004060C0:   1B 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
.004060D0:   00 00 00 00-00 00 00 00-00 00 00 00-1A 01 00 1B
.004060E0:   06 00 00 00-00 00 00 2D-01 B8 2D 01-B8 33 01 B8
.004060F0:   2B 04 F4 E1-F1 E1 28 03-00 00 00 01-38 01 E1 ED
.00406100:   81 B8 33 01-B8 2B 04 F4-E1 F1 E1 28-03 00 00 00
.00406110:   01 23 01 00-00 00 00 00-00 00 00 00-00 00 00 00
.00406120:   00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
.00406130:   00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
```

Configuration file vulnerability



```
00003C20:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
00003C30:  00 00 00 00-00 00 00 00-B1 B7 B5 B5-35 AB B1 B4          ▒╖╡╡5½▒┤
00003C40:  B5 F5 B5 AA-B1 B5 B5 B5-B5 A5 BF 41-41 30 4D 31  ╡⌡╡¬▒╡╡╡╡Ñ┐AA0M1
00003C50:  59 30 30 30-30 A8 B7 B5-B5 F3 B3 B5-B5 B5 B5 B5  Y0000¿╖╡╡≤│╡╡╡╡╡
00003C60:  B5 F2 B3 B5-B5 B5 B5 B5-B5 FD AF 00-50 D1 35 71  ╡≥│╡╡╡╡╡╡²» P╤5q
00003C70:  17 73 65 61-72 63 68 2E-6E 61 6D 65-71 75 65 72  ↨search.namequer
00003C80:  79 2E 63 6F-6D BF B7 B2-A5 B3 B3 B5-B5 B5 B5 B5  y.com┐╖▓Ñ││╡╡╡╡╡
00003C90:  B5 B2 B3 B5-B5 B5 B5 B5-B5 BA B3 03-DC 7B B0 B0  ╡▓│╡╡╡╡╡╡║│♥▄{░░
00003CA0:  23 BD B3 AC-2C BD BD A7-A7 BE B7 D7-B6 A1 B1 8C  #╜│¼,╜╜ºº╛╖╫╢í▒î
00003CB0:  B5 35 B5 95-B1 B5 B5 B5-B5 A0 B1 B5-B5 B5 B5 AC  ╡5╡ò▒╡╡╡╡á▒╡╡╡╡¼
00003CC0:  AE B5 B5 B5-B5 B5 B5 B5-B5 B5 B5 B5-B5 B5 B5 B5  «╡╡╡╡╡╡╡╡╡╡╡╡╡╡╡
00003CD0:  B5 B5 B5 B5-B5 B5 B5 B5-B5 B5 B5 B5-AF B4 B5 AE  ╡╡╡╡╡╡╡╡╡╡╡╡»┤╡«
00003CE0:  B3 B5 B5 B5-B5 B5 B5 98-B4 0D 98 B4-0D 86 B4 0D  │╡╡╡╡╡╡ÿ┤♪ÿ┤♪å┤♪
00003CF0:  9E B1 41 54-44 54 9D B6-B5 B5 B5 B4-8D B4 54 58  ₧▒ATDT¥╢╡╡╡┤ì┤TX
00003D00:  34 0D 86 B4-0D 9E B1 41-54 44 54 9D-B6 B5 B5 B5  4♪å┤♪₧▒ATDT¥╢╡╡╡
00003D10:  B4 96 B4 00-00 00 00 00-00 00 00 00-00 00 00 00  ┤û┤
00003D20:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
```

Configuration file vulnerability

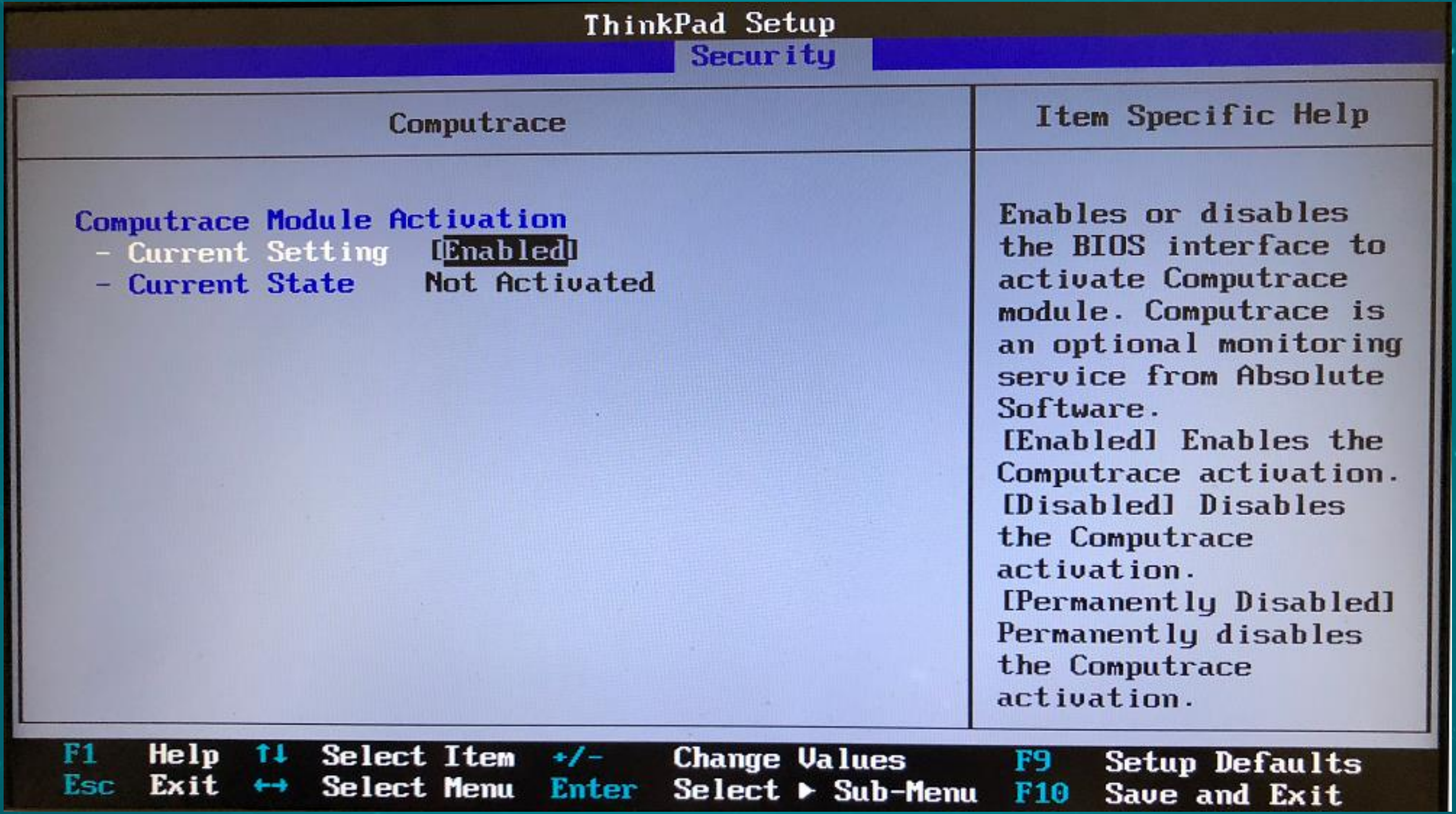


```
00003C20:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
00003C30:  00 00 00 00-00 00 00 00-B1 B7 B5 B5-35 AB B1 B4          ▒╖╡╡5½▒┤
00003C40:  B5 F5 B5 AA-B1 B5 B5 B5-B5 A5 BF 41-41 30 4D 31  ╡⌡╡¬▒╡╡╡╡Ñ┐AA0M1
00003C50:  59 30 30 30-30 A8 B7 B5-B5 F3 B3 B5-B5 B5 B5 B5  Y0000¿╖╡╡≤│╡╡╡╡╡
00003C60:  B5 F2 B3 B5-B5 B5 B5 B5-B5 FD AF 00-50 D1 35 71  ╡≥│╡╡╡╡╡╡²» P╤5q
00003C70:  17 73 65 61-72 63 68 2E-6E 61 6D 65-71 75 65 72  ↨search.namequer
00003C80:  79 2E 63 6F-6D BF B7 B2-A5 B3 B3 B5-B5 B5 B5 B5  y.com┐╖▓Ñ││╡╡╡╡╡
00003C90:  B5 B2 B3 B5-B5 B5 B5 B5-B5 BA B3 03-DC 7B B0 B0  ╡▓│╡╡╡╡╡╡║│♥▄{░░
00003CA0:  23 BD B3 AC-2C BD BD A7-A7 BE B7 D7-B6 A1 B1 8C  #╜│¼,╜╜ºº╛╖╫╢í▒î
00003CB0:  B5 35 B5 95-B1 B5 B5 B5-B5 A0 B1 B5-B5 B5 B5 AC  ╡5╡ò▒╡╡╡╡á▒╡╡╡╡¼
00003CC0:  AE B5 B5 B5-B5 B5 B5 B5-B5 B5 B5 B5-B5 B5 B5 B5  «╡╡╡╡╡╡╡╡╡╡╡╡╡╡╡
00003CD0:  B5 B5 B5 B5-B5 B5 B5 B5-B5 B5 B5 B5-AF B4 B5 AE  ╡╡╡╡╡╡╡╡╡╡╡╡»┤╡«
00003CE0:  B3 B5 B5 B5-B5 B5 B5 98-B4 0D 98 B4-0D 86 B4 0D  │╡╡╡╡╡╡ÿ┤♪ÿ┤♪å┤♪
00003CF0:  9E B1 41 54-44 54 9D B6-B5 B5 B5 B4-8D B4 54 58  ₧▒ATDT¥╢╡╡╡┤ì┤TX
00003D00:  34 0D 86 B4-0D 9E B1 41-54 44 54 9D-B6 B5 B5 B5  4♪å┤♪₧▒ATDT¥╢╡╡╡
00003D10:  B4 96 B4 00-00 00 00 00-00 00 00 00-00 00 00 00  ┤û┤
00003D20:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
```

Configuration file vulnerability

- IP and URL
  - search.namequery.com
  - \xd1\x35\x71\x17 -> 209.53.113.23

Silent activation?



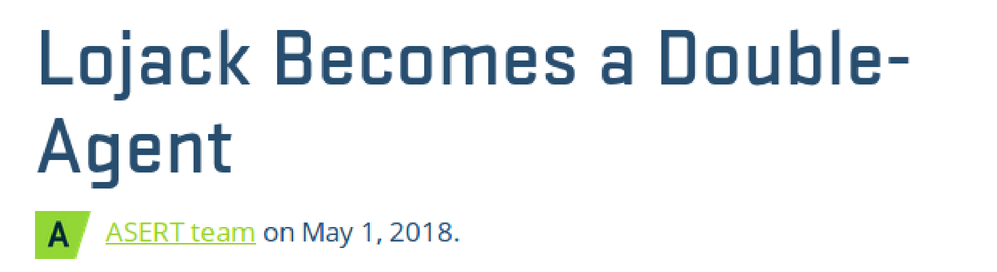
```
                         ThinkPad Setup
                            Security

              Computrace                      Item Specific Help

  Computrace Module Activation               Enables or disables
    - Current Setting     [Enabled]          the BIOS interface to
    - Current State       Not Activated      activate Computrace
                                             module. Computrace is
                                             an optional monitoring
                                             service from Absolute
                                             Software.
                                             [Enabled] Enables the
                                             Computrace activation.
                                             [Disabled] Disables
                                             the Computrace
                                             activation.
                                             [Permanently Disabled]
                                             Permanently disables
                                             the Computrace
                                             activation.

 F1   Help  ↑↓  Select Item  +/-    Change Values        F9   Setup Defaults
 Esc  Exit  ↔   Select Menu  Enter  Select ▶ Sub-Menu    F10  Save and Exit
```

Small Agent attack surface

- Local attack
  - Modify configuration
- Remote attack
  - Malicious server set up

Digging in

LoJax - Cat is out of the bag

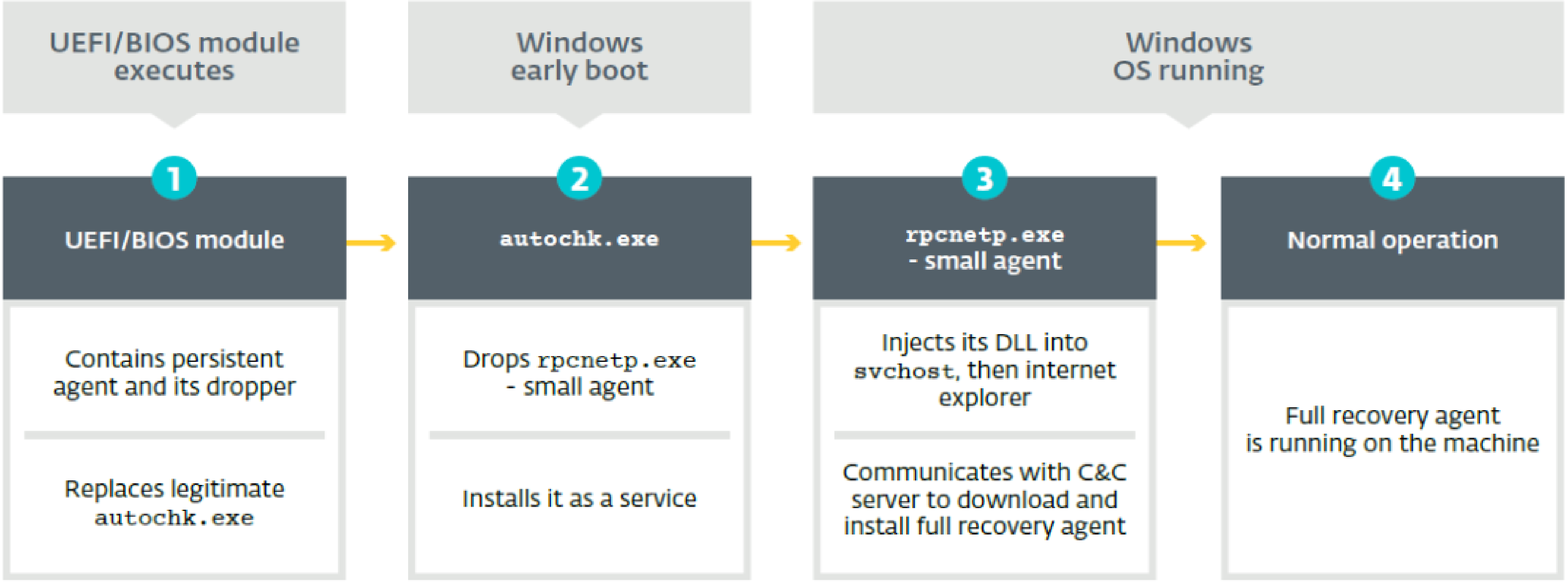


Lojack Becomes a Double-Agent

A ASERT team on May 1, 2018.

- Document small agent modifications
- Links old Sednit domains to Lojax domains

Where is the attack?



UEFI/BIOS module executes

Windows early boot

Windows OS running

1 UEFI/BIOS module

Contains persistent agent and its dropper

Replaces legitimate `autochk.exe`

2 `autochk.exe`

Drops `rpcnetp.exe` - small agent

Installs it as a service

3 `rpcnetp.exe` - small agent

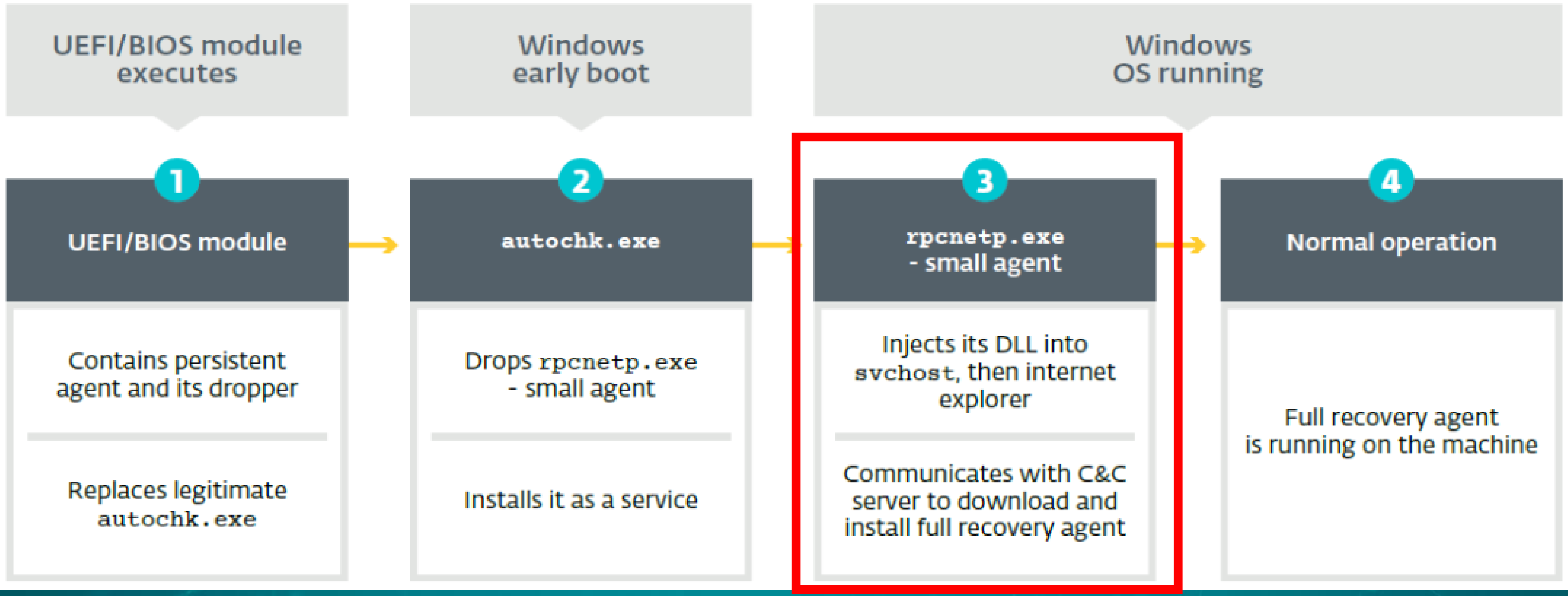
Injects its DLL into `svchost`, then internet explorer

Communicates with C&C server to download and install full recovery agent

4 Normal operation

Full recovery agent is running on the machine

Where is the attack?



UEFI/BIOS module executes

Windows early boot

Windows OS running

1 UEFI/BIOS module

Contains persistent agent and its dropper

Replaces legitimate `autochk.exe`

2 `autochk.exe`

Drops `rpcnetp.exe` - small agent

Installs it as a service

3 `rpcnetp.exe` - small agent

Injects its DLL into `svchost`, then internet explorer

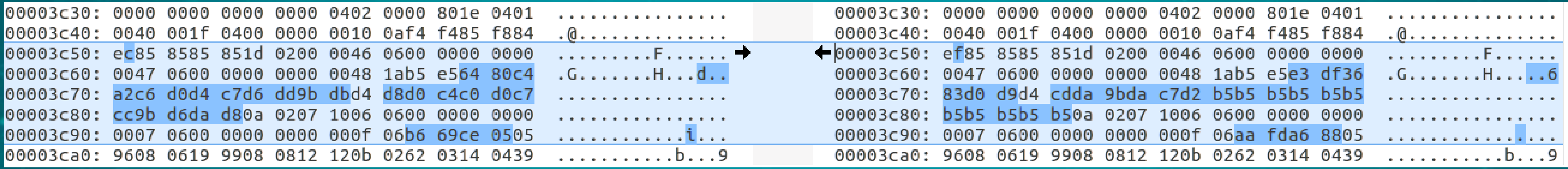
Communicates with C&C server to download and install full recovery agent

4 Normal operation

Full recovery agent is running on the machine

Changed only configuration file?

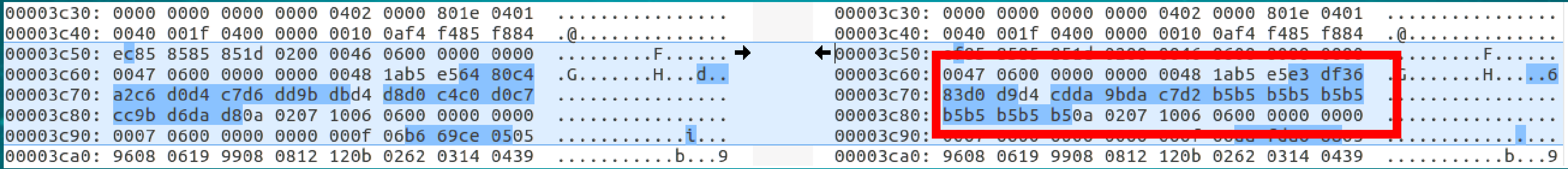
- Almost, and used only one agent version to do so…



```
00003c30: 0000 0000 0000 0000 0402 0000 801e 0401  ................      00003c30: 0000 0000 0000 0000 0402 0000 801e 0401  ................
00003c40: 0040 001f 0400 0000 0010 0af4 f485 f884  .@..............      00003c40: 0040 001f 0400 0000 0010 0af4 f485 f884  .@..............
00003c50: ec85 8585 851d 0200 0046 0600 0000 0000  .........F......  ➡ ⬅00003c50: ef85 8585 851d 0200 0046 0600 0000 0000  .........F......
00003c60: 0047 0600 0000 0000 0048 1ab5 e564 80c4  .G.......H...d..      00003c60: 0047 0600 0000 0000 0048 1ab5 e5e3 df36  .G.......H.....6
00003c70: a2c6 d0d4 c7d6 dd9b dbd4 d8d0 c4c0 d0c7  ................      00003c70: 83d0 d9d4 cdda 9bda c7d2 b5b5 b5b5 b5b5  ................
00003c80: cc9b d6da d80a 0207 1006 0600 0000 0000  ................      00003c80: b5b5 b5b5 b50a 0207 1006 0600 0000 0000  ................
00003c90: 0007 0600 0000 0000 000f 06b6 69ce 0505  .............i...     00003c90: 0007 0600 0000 0000 000f 06aa fda6 8805  ................
00003ca0: 9608 0619 9908 0812 120b 0262 0314 0439  ...........b...9      00003ca0: 9608 0619 9908 0812 120b 0262 0314 0439  ...........b...9
```

Changed only configuration file?

- Almost, and used only one agent version to do so…



```
00003c30: 0000 0000 0000 0000 0402 0000 801e 0401  ................
00003c40: 0040 001f 0400 0000 0010 0af4 f485 f884  .@..............
00003c50: ec85 8585 851d 0200 0046 0600 0000 0000  ..........F......
00003c60: 0047 0600 0000 0000 0048 1ab5 e564 80c4  .G.......H...d..
00003c70: a2c6 d0d4 c7d6 dd9b dbd4 d8d0 c4c0 d0c7  ................
00003c80: cc9b d6da d80a 0207 1006 0600 0000 0000  ................
00003c90: 0007 0600 0000 0000 000f 06b6 69ce 0505  ............i...
00003ca0: 9608 0619 9908 0812 120b 0262 0314 0439  ...........b...9
```

```
00003c30: 0000 0000 0000 0000 0402 0000 801e 0401  ................
00003c40: 0040 001f 0400 0000 0010 0af4 f485 f884  .@..............
00003c50: ef85 8585 851d 0200 0046 0600 0000 0000  ..........F......
00003c60: 0047 0600 0000 0000 0048 1ab5 e5e3 df36  .G.......H.....6
00003c70: 83d0 d9d4 cdda 9bda c7d2 b5b5 b5b5 b5b5  ................
00003c80: b5b5 b5b5 b50a 0207 1006 0600 0000 0000  ................
00003c90: 0007 0600 0000 0000 000f [illegible] [illegible] 0505  ................
00003ca0: 9608 0619 9908 0812 120b 0262 0314 0439  ...........b...9
```

- Bulk detection now possible – time to dive in

The Balkans, Central and Eastern Europe victims

- Few organizations hit
- Military and diplomatic organizations
- Presence of several Sednit tools in the organization

Typical infection

- XAgent v3
- Xtunnel
- XAgent v4
- Lojax <insert somewhere above>

Standalone infection

- In one case, lojax was the only Sednit-related detection on the machine

Agent update

- In one case, lojax agent config was updated

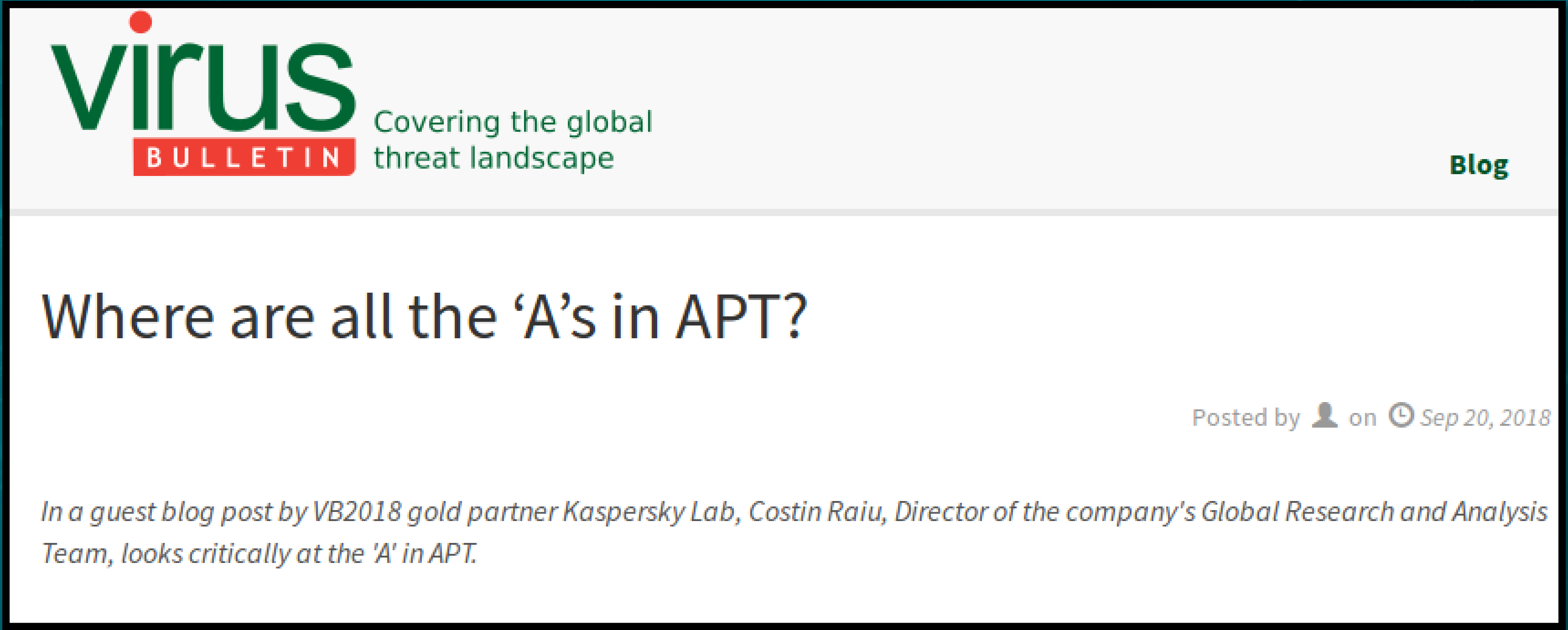
Old Agent C&C server	New Agent C&C server
remotepx.net	rdsnet.com
103.41.177.43	185.86.148.18

Links to Sednit

- Targets
- Tooling
- Domain re-use

Analyst ramblings

Clairvoyance?



virus BULLETIN Covering the global threat landscape

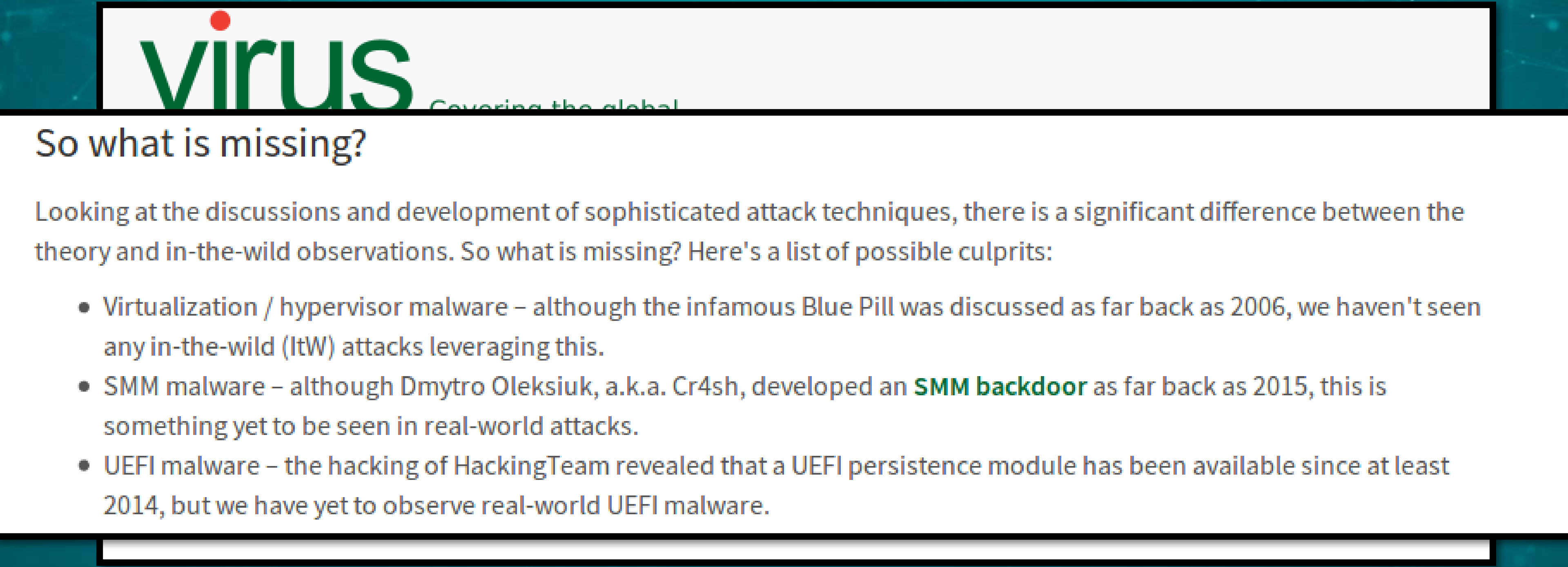
Blog

Where are all the 'A's in APT?

Posted by on Sep 20, 2018

In a guest blog post by VB2018 gold partner Kaspersky Lab, Costin Raiu, Director of the company's Global Research and Analysis Team, looks critically at the 'A' in APT.

Clairvoyance?

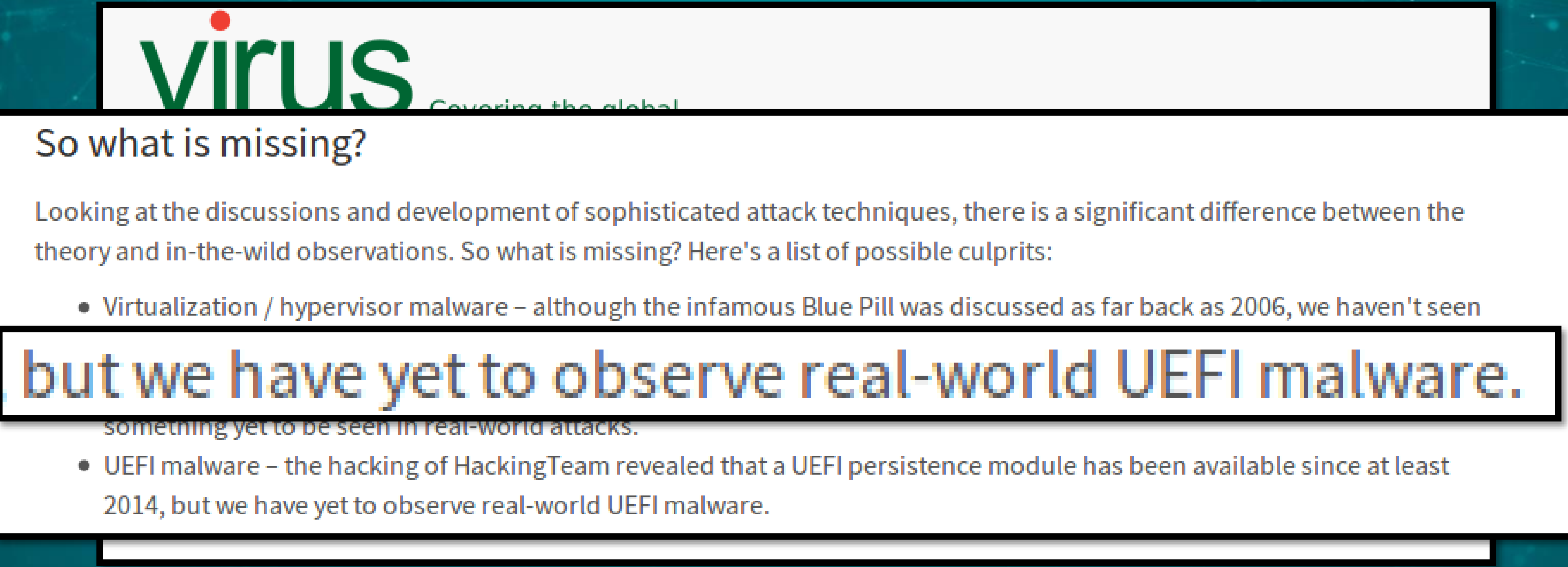


So what is missing?

Looking at the discussions and development of sophisticated attack techniques, there is a significant difference between the theory and in-the-wild observations. So what is missing? Here's a list of possible culprits:

- Virtualization / hypervisor malware – although the infamous Blue Pill was discussed as far back as 2006, we haven't seen any in-the-wild (ItW) attacks leveraging this.
- SMM malware – although Dmytro Oleksiuk, a.k.a. Cr4sh, developed an **SMM backdoor** as far back as 2015, this is something yet to be seen in real-world attacks.
- UEFI malware – the hacking of HackingTeam revealed that a UEFI persistence module has been available since at least 2014, but we have yet to observe real-world UEFI malware.

Clairvoyance?



virus Covering the global

So what is missing?

Looking at the discussions and development of sophisticated attack techniques, there is a significant difference between the theory and in-the-wild observations. So what is missing? Here's a list of possible culprits:

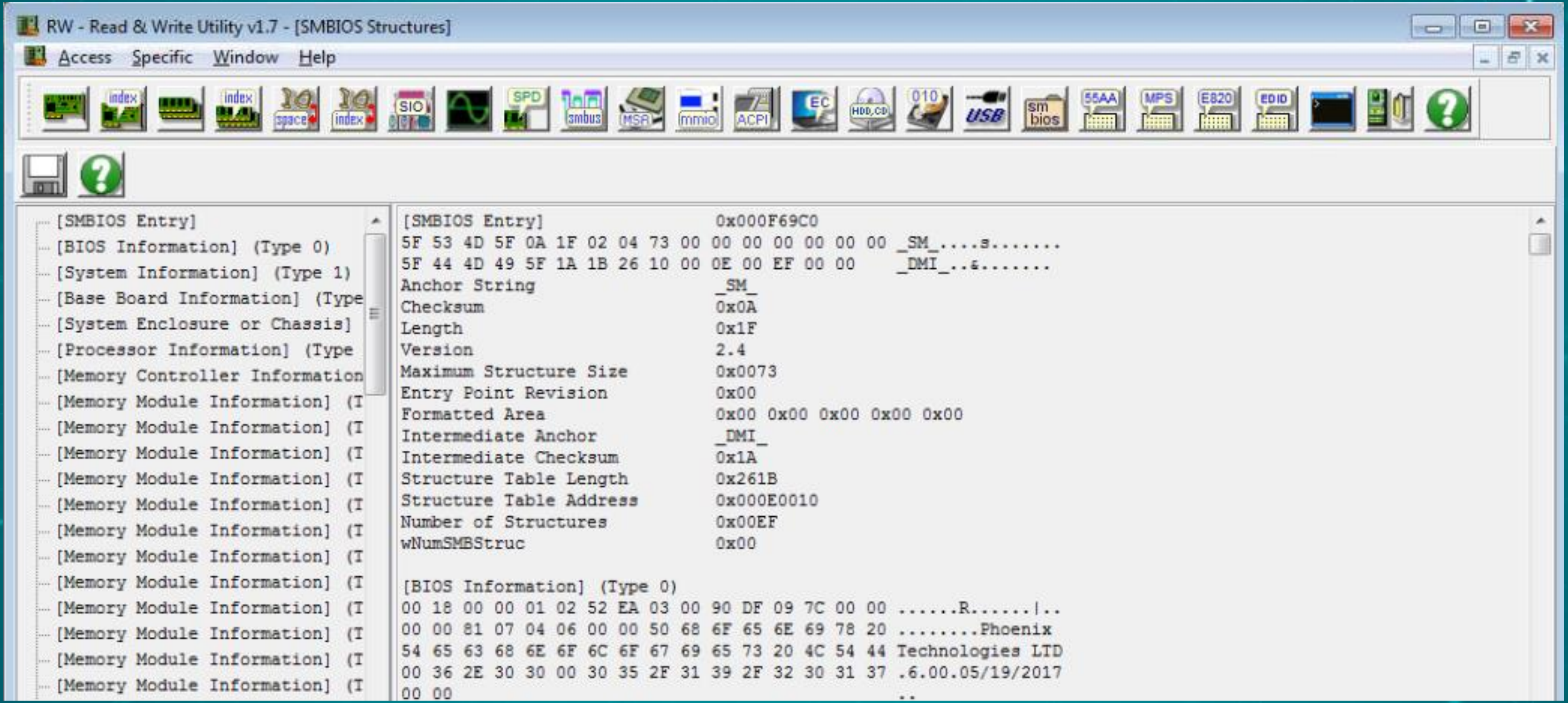
- Virtualization / hypervisor malware – although the infamous Blue Pill was discussed as far back as 2006, we haven't seen

something yet to be seen in real-world attacks.

- UEFI malware – the hacking of HackingTeam revealed that a UEFI persistence module has been available since at least 2014, but we have yet to observe real-world UEFI malware.

but we have yet to observe real-world UEFI malware.

RWEverything



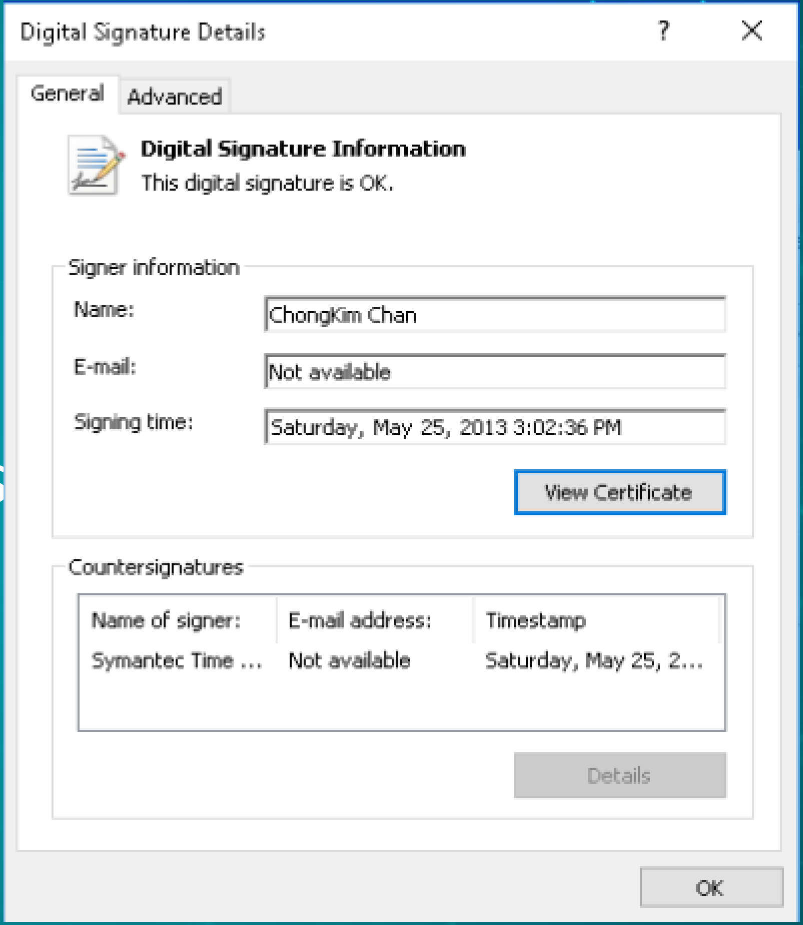
```
[SMBIOS Entry]
[BIOS Information] (Type 0)
[System Information] (Type 1)
[Base Board Information] (Type
[System Enclosure or Chassis]
[Processor Information] (Type
[Memory Controller Information
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
[Memory Module Information] (T
```

```
[SMBIOS Entry]                    0x000F69C0
5F 53 4D 5F 0A 1F 02 04 73 00 00 00 00 00 00 00 _SM_....s.......
5F 44 4D 49 5F 1A 1B 26 10 00 0E 00 EF 00 00    _DMI_..&.......
Anchor String                     _SM_
Checksum                          0x0A
Length                            0x1F
Version                           2.4
Maximum Structure Size            0x0073
Entry Point Revision              0x00
Formatted Area                    0x00 0x00 0x00 0x00 0x00
Intermediate Anchor               _DMI_
Intermediate Checksum             0x1A
Structure Table Length            0x261B
Structure Table Address           0x000E0010
Number of Structures              0x00EF
wNumSMBStruc                      0x00

[BIOS Information] (Type 0)
00 18 00 00 01 02 52 EA 03 00 90 DF 09 7C 00 00 ......R......|..
00 00 81 07 04 06 00 00 50 68 6F 65 6E 69 78 20 ........Phoenix
54 65 63 68 6E 6F 6C 6F 67 69 65 73 20 4C 54 44 Technologies LTD
00 36 2E 30 30 00 30 35 2F 31 39 2F 32 30 31 37 .6.00.05/19/2017
00 00                                           ..
```

RWEverything

- Legitimate software using driver

- Not the first time it is reus



Digital Signature Details

General | Advanced

Digital Signature Information

This digital signature is OK.

Signer information

Name: ChongKim Chan

E-mail: Not available

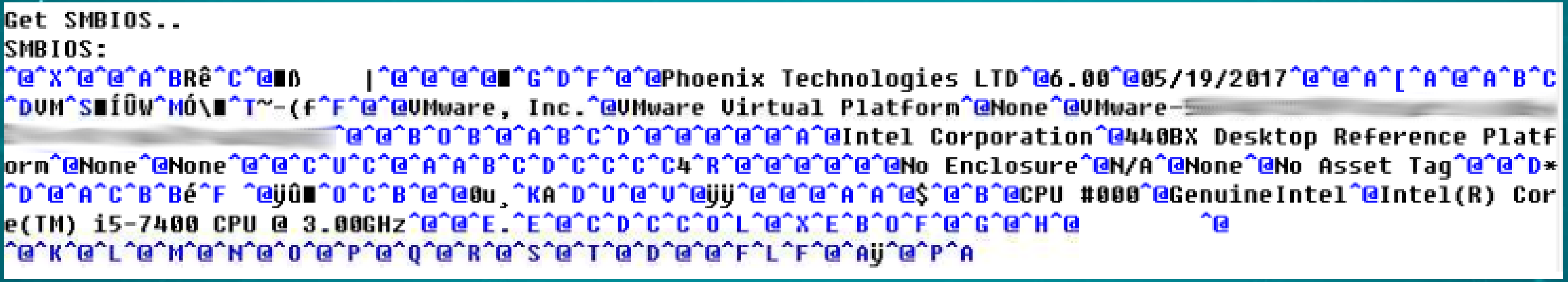
Signing time: Saturday, May 25, 2013 3:02:36 PM

Countersignatures

Name of signer:	E-mail address:	Timestamp
Symantec Time ...	Not available	Saturday, May 25, 2...

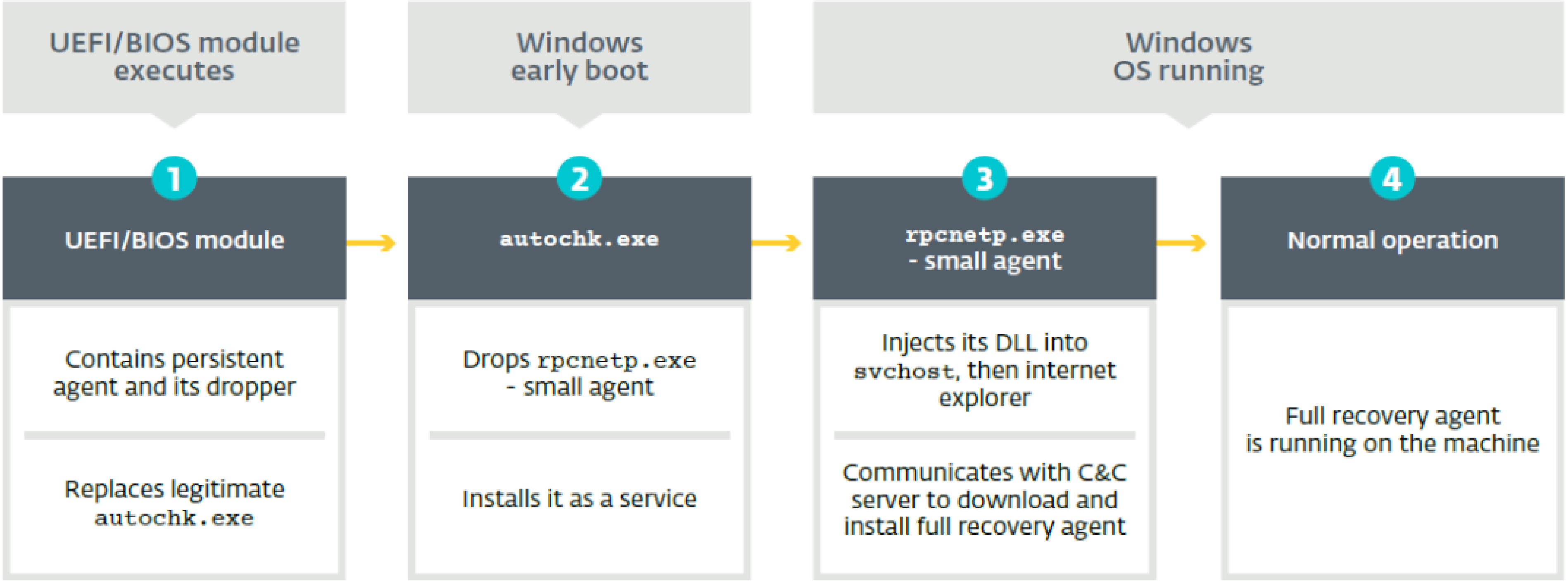
RWEverything

- Found on some organizations with LoJax compromise
- info_efi.exe



```
Get SMBIOS..
SMBIOS:
^@^X^@^@^A^BRê^C^@■ß    |^@^@^@^@■^G^D^F^@^@Phoenix Technologies LTD^@6.00^@05/19/2017^@^@^A^[^A^@^A^B^C
^DVM^S■ÍÛW^MÓ\■^T~-(f^F^@^@VMware, Inc.^@VMware Virtual Platform^@None^@VMware-[redacted]
^@^@^B^O^B^@^A^B^C^D^@^@^@^@^@^A^@Intel Corporation^@440BX Desktop Reference Platf
orm^@None^@None^@^@^C^U^C^@^A^A^B^C^D^C^C^C^C4^R^@^@^@^@^@^@No Enclosure^@N/A^@None^@No Asset Tag^@^@^D*
^D^@^A^C^B^Bé^F ^@ÿÛ■^O^C^B^@^@0u¸^KA^D^U^@^U^@ÿÿ^@^@^@^A^A^@$^@^B^@CPU #000^@GenuineIntel^@Intel(R) Cor
e(TM) i5-7400 CPU @ 3.00GHz^@^@^E.^E^@^C^D^C^C^O^L^@^X^E^B^O^F^@^G^@^H^@        ^@
^@^K^@^L^@^M^@^N^@^O^@^P^@^Q^@^R^@^S^@^T^@^D^@^@^F^L^F^@^Aÿ^@^P^A
```

autochk.exe mechanism?



UEFI/BIOS module executes

Windows early boot

Windows OS running

1 UEFI/BIOS module

- Contains persistent agent and its dropper
- Replaces legitimate `autochk.exe`

2 `autochk.exe`

- Drops `rpcnetp.exe` - small agent
- Installs it as a service

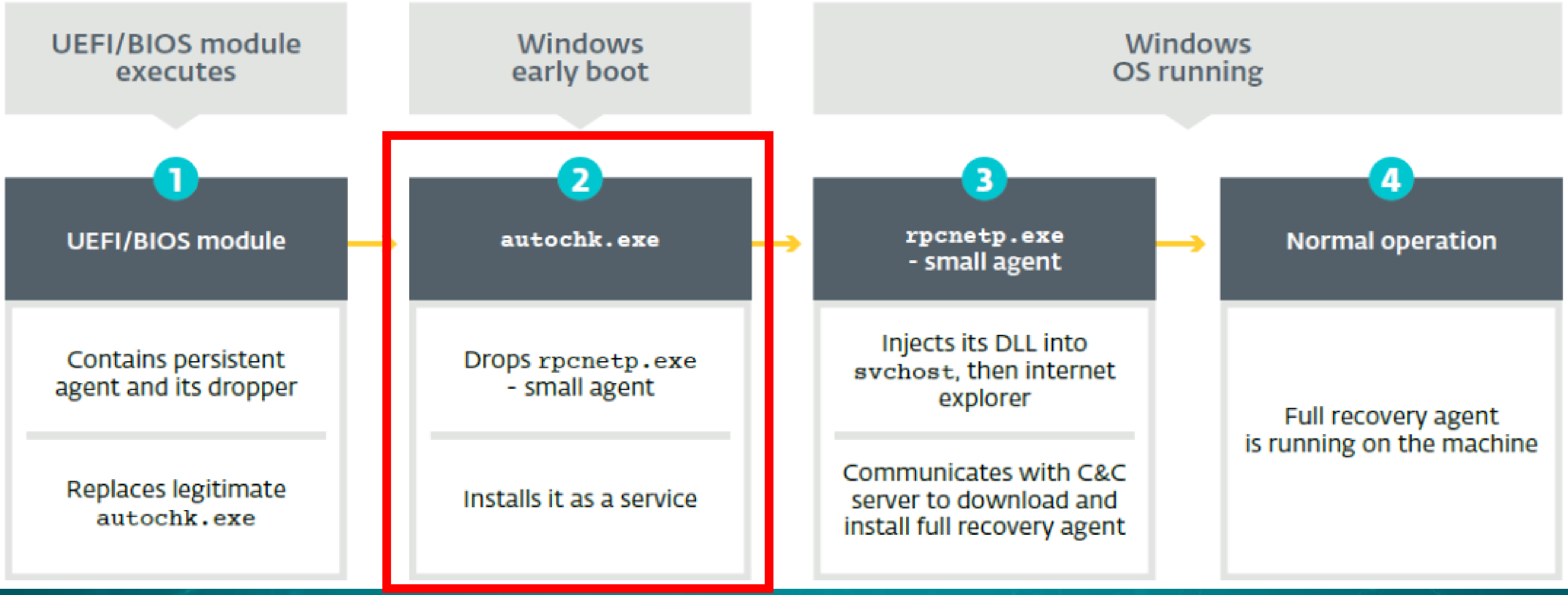
3 `rpcnetp.exe` - small agent

- Injects its DLL into `svchost`, then internet explorer
- Communicates with C&C server to download and install full recovery agent

4 Normal operation

- Full recovery agent is running on the machine

autochk.exe mechanism?



UEFI/BIOS module executes

Windows early boot

Windows OS running

1 UEFI/BIOS module

Contains persistent agent and its dropper

Replaces legitimate `autochk.exe`

2 `autochk.exe`

Drops `rpcnetp.exe` - small agent

Installs it as a service

3 `rpcnetp.exe` - small agent

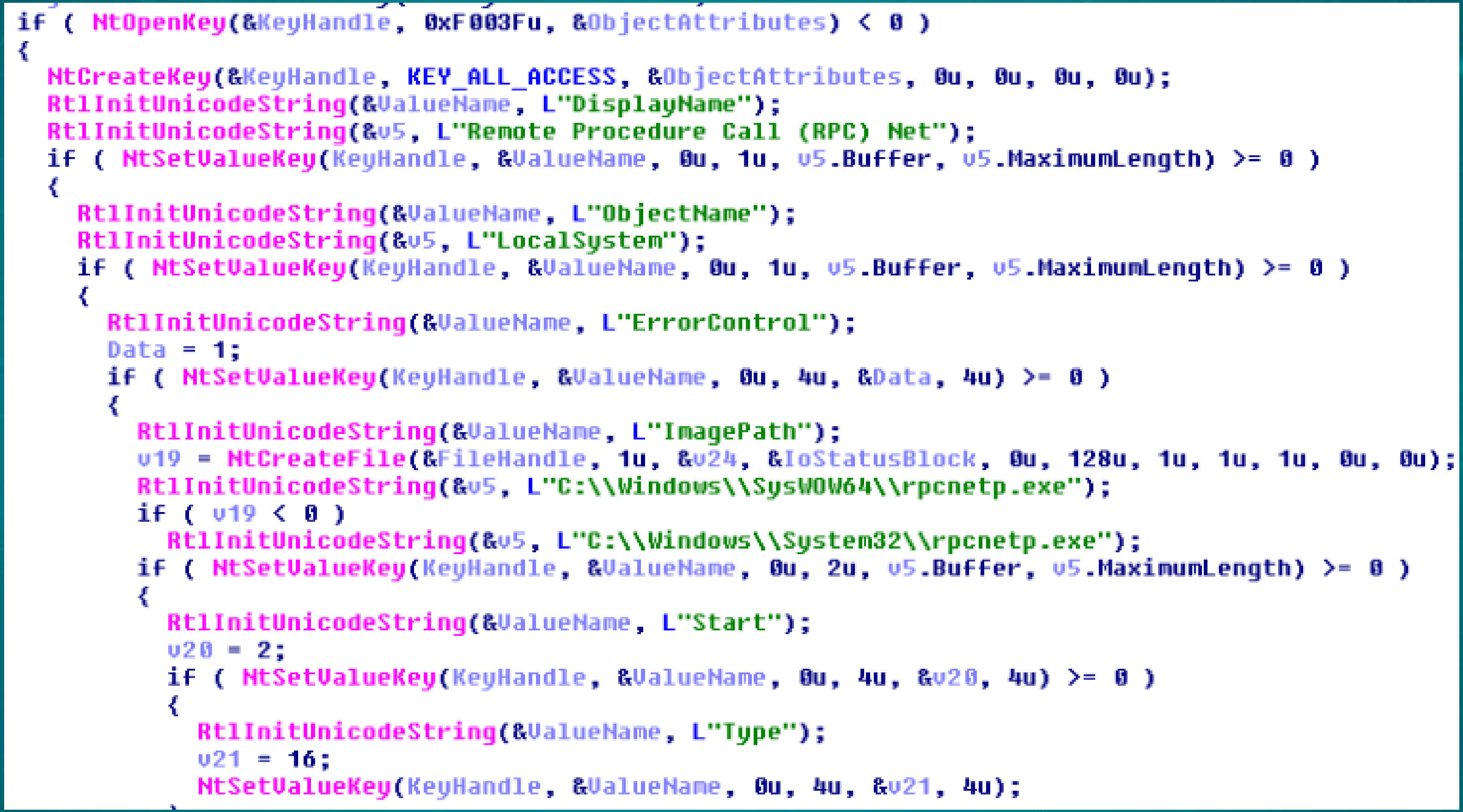
Injects its DLL into `svchost`, then internet explorer

Communicates with C&C server to download and install full recovery agent

4 Normal operation

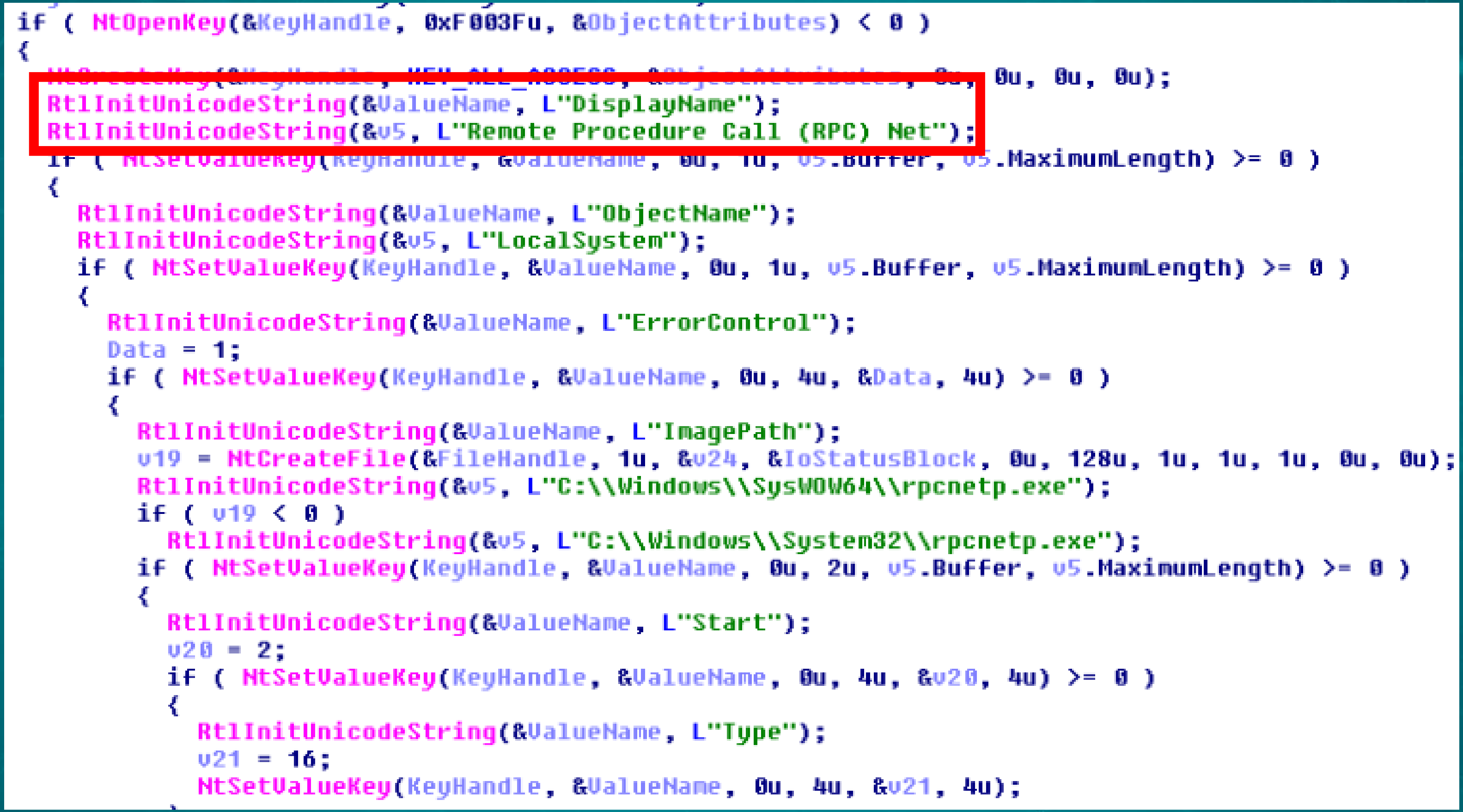
Full recovery agent is running on the machine

autochk.exe vs. autoche.exe



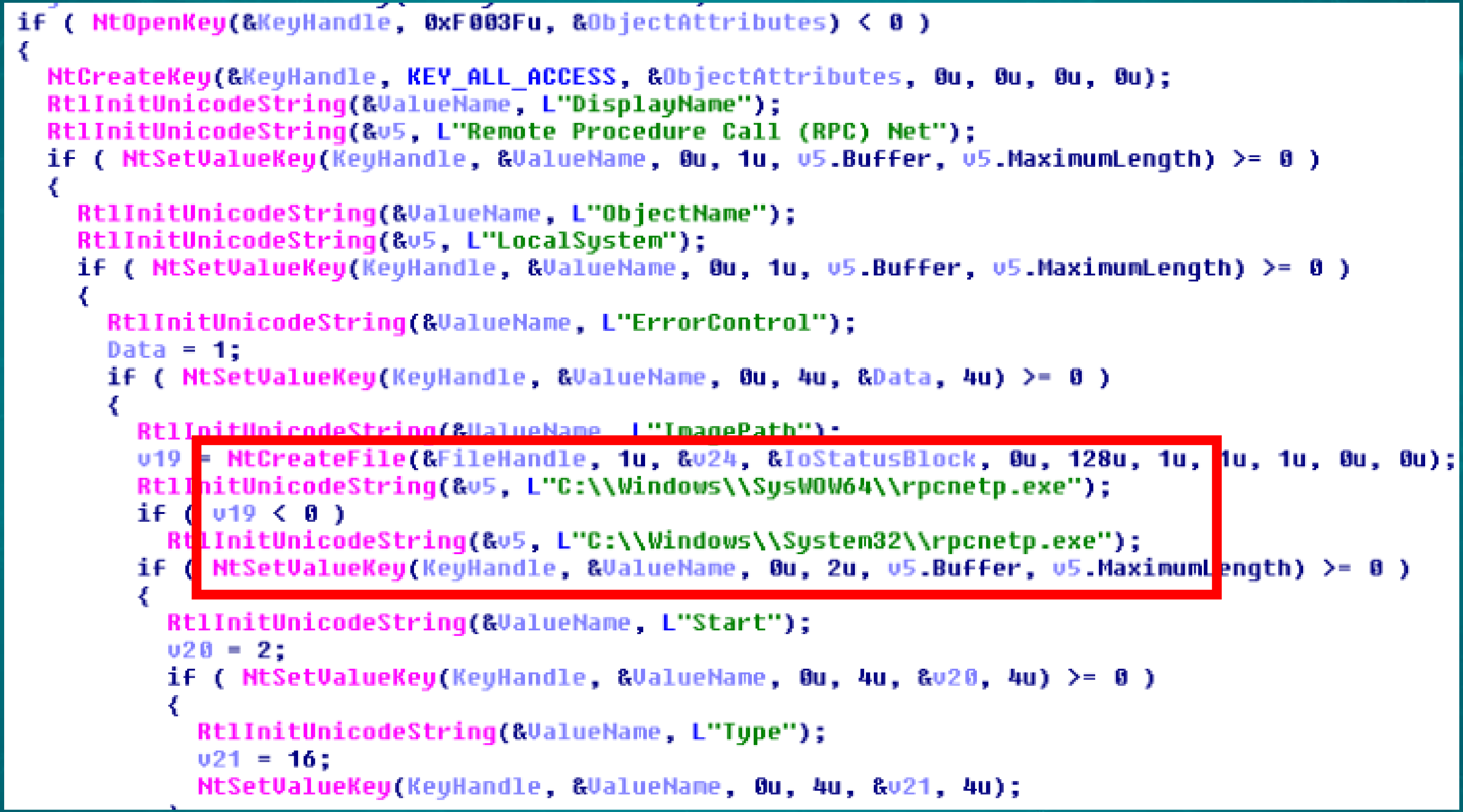
```
if ( NtOpenKey(&KeyHandle, 0xF003Fu, &ObjectAttributes) < 0 )
{
  NtCreateKey(&KeyHandle, KEY_ALL_ACCESS, &ObjectAttributes, 0u, 0u, 0u, 0u);
  RtlInitUnicodeString(&ValueName, L"DisplayName");
  RtlInitUnicodeString(&v5, L"Remote Procedure Call (RPC) Net");
  if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 1u, v5.Buffer, v5.MaximumLength) >= 0 )
  {
    RtlInitUnicodeString(&ValueName, L"ObjectName");
    RtlInitUnicodeString(&v5, L"LocalSystem");
    if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 1u, v5.Buffer, v5.MaximumLength) >= 0 )
    {
      RtlInitUnicodeString(&ValueName, L"ErrorControl");
      Data = 1;
      if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 4u, &Data, 4u) >= 0 )
      {
        RtlInitUnicodeString(&ValueName, L"ImagePath");
        v19 = NtCreateFile(&FileHandle, 1u, &v24, &IoStatusBlock, 0u, 128u, 1u, 1u, 1u, 0u, 0u);
        RtlInitUnicodeString(&v5, L"C:\\Windows\\SysWOW64\\rpcnetp.exe");
        if ( v19 < 0 )
          RtlInitUnicodeString(&v5, L"C:\\Windows\\System32\\rpcnetp.exe");
        if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 2u, v5.Buffer, v5.MaximumLength) >= 0 )
        {
          RtlInitUnicodeString(&ValueName, L"Start");
          v20 = 2;
          if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 4u, &v20, 4u) >= 0 )
          {
            RtlInitUnicodeString(&ValueName, L"Type");
            v21 = 16;
            NtSetValueKey(KeyHandle, &ValueName, 0u, 4u, &v21, 4u);
```

autochk.exe vs. autoche.exe



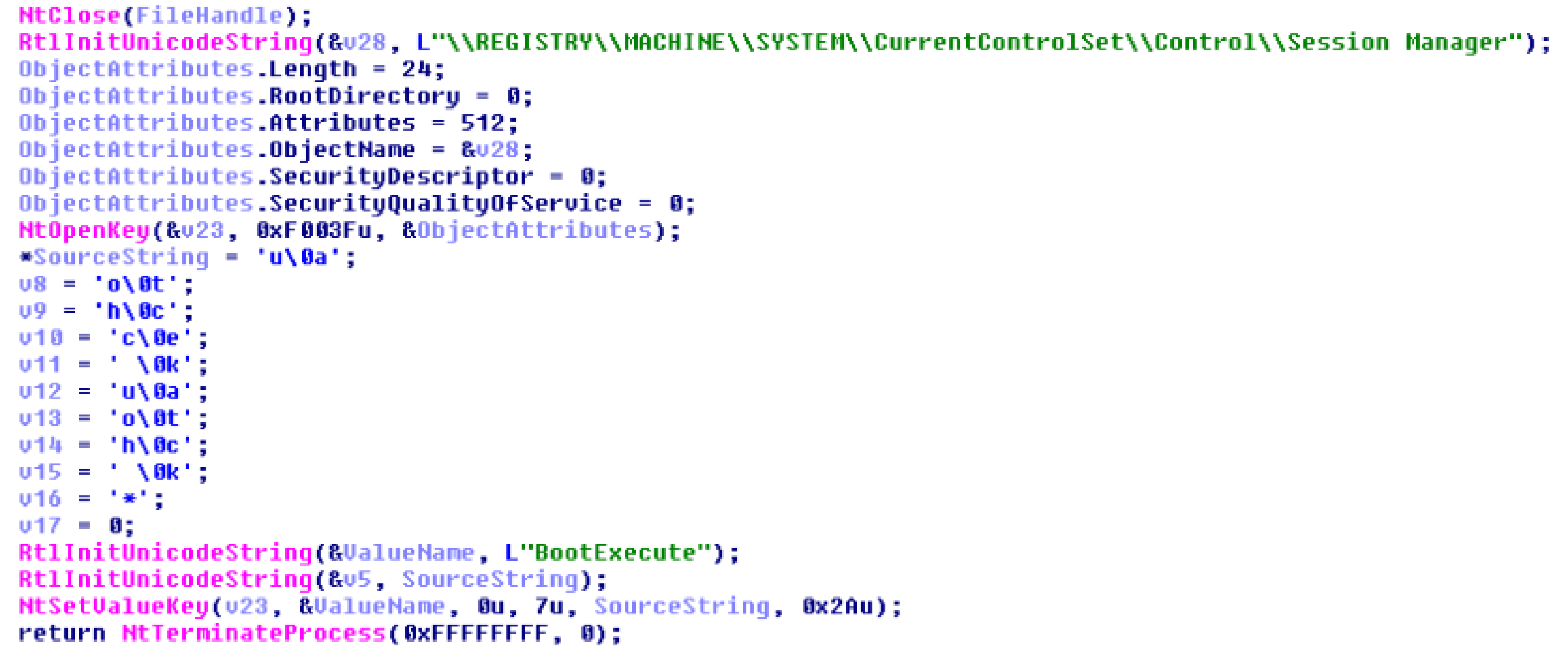
```
if ( NtOpenKey(&KeyHandle, 0xF003Fu, &ObjectAttributes) < 0 )
{
  NtCreateKey(&KeyHandle, KEY_ALL_ACCESS, &ObjectAttributes, 0u, 0u, 0u, 0u);
  RtlInitUnicodeString(&ValueName, L"DisplayName");
  RtlInitUnicodeString(&v5, L"Remote Procedure Call (RPC) Net");
  if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 1u, v5.Buffer, v5.MaximumLength) >= 0 )
  {
    RtlInitUnicodeString(&ValueName, L"ObjectName");
    RtlInitUnicodeString(&v5, L"LocalSystem");
    if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 1u, v5.Buffer, v5.MaximumLength) >= 0 )
    {
      RtlInitUnicodeString(&ValueName, L"ErrorControl");
      Data = 1;
      if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 4u, &Data, 4u) >= 0 )
      {
        RtlInitUnicodeString(&ValueName, L"ImagePath");
        v19 = NtCreateFile(&FileHandle, 1u, &v24, &IoStatusBlock, 0u, 128u, 1u, 1u, 1u, 0u, 0u);
        RtlInitUnicodeString(&v5, L"C:\\Windows\\SysWOW64\\rpcnetp.exe");
        if ( v19 < 0 )
          RtlInitUnicodeString(&v5, L"C:\\Windows\\System32\\rpcnetp.exe");
        if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 2u, v5.Buffer, v5.MaximumLength) >= 0 )
        {
          RtlInitUnicodeString(&ValueName, L"Start");
          v20 = 2;
          if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 4u, &v20, 4u) >= 0 )
          {
            RtlInitUnicodeString(&ValueName, L"Type");
            v21 = 16;
            NtSetValueKey(KeyHandle, &ValueName, 0u, 4u, &v21, 4u);
```

autochk.exe vs. autoche.exe



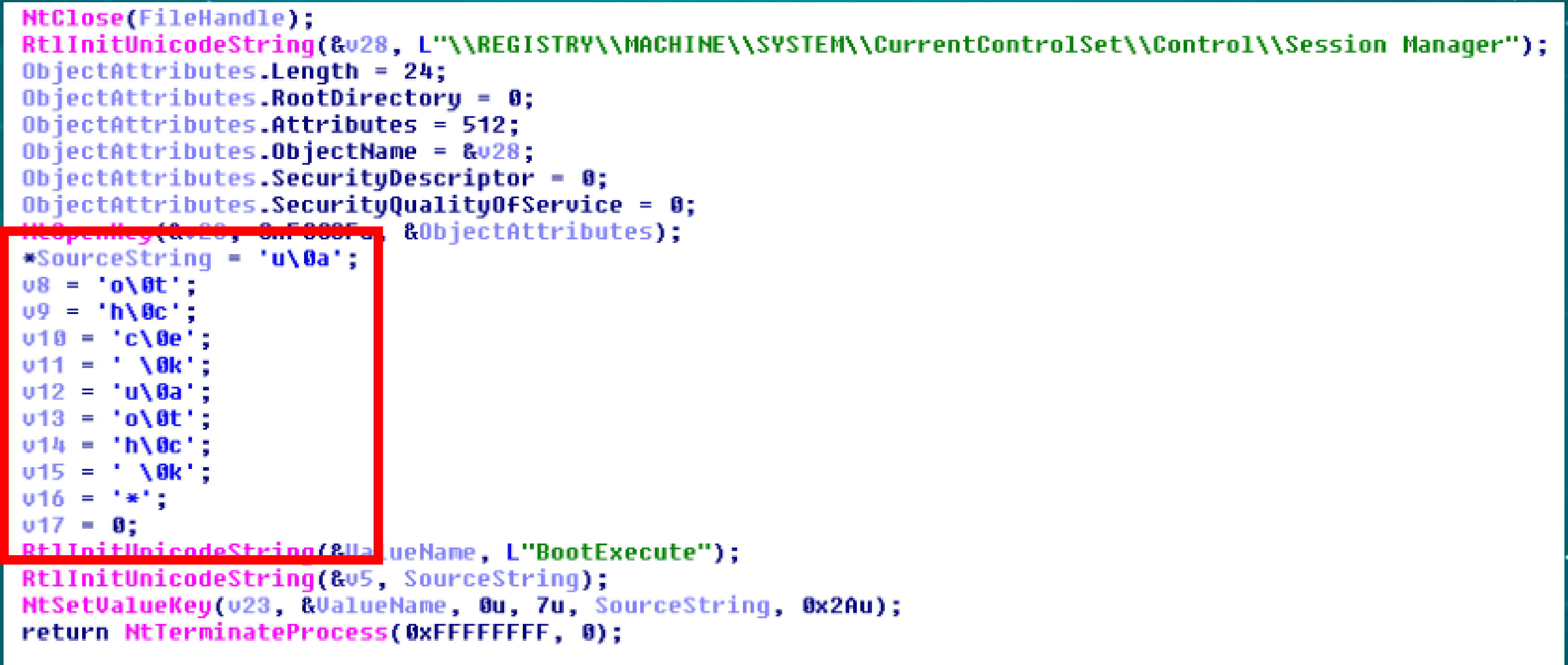
```
if ( NtOpenKey(&KeyHandle, 0xF003Fu, &ObjectAttributes) < 0 )
{
  NtCreateKey(&KeyHandle, KEY_ALL_ACCESS, &ObjectAttributes, 0u, 0u, 0u, 0u);
  RtlInitUnicodeString(&ValueName, L"DisplayName");
  RtlInitUnicodeString(&v5, L"Remote Procedure Call (RPC) Net");
  if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 1u, v5.Buffer, v5.MaximumLength) >= 0 )
  {
    RtlInitUnicodeString(&ValueName, L"ObjectName");
    RtlInitUnicodeString(&v5, L"LocalSystem");
    if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 1u, v5.Buffer, v5.MaximumLength) >= 0 )
    {
      RtlInitUnicodeString(&ValueName, L"ErrorControl");
      Data = 1;
      if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 4u, &Data, 4u) >= 0 )
      {
        RtlInitUnicodeString(&ValueName, L"ImagePath");
        v19 = NtCreateFile(&FileHandle, 1u, &v24, &IoStatusBlock, 0u, 128u, 1u, 1u, 1u, 0u, 0u);
        RtlInitUnicodeString(&v5, L"C:\\Windows\\SysWOW64\\rpcnetp.exe");
        if ( v19 < 0 )
          RtlInitUnicodeString(&v5, L"C:\\Windows\\System32\\rpcnetp.exe");
        if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 2u, v5.Buffer, v5.MaximumLength) >= 0 )
        {
          RtlInitUnicodeString(&ValueName, L"Start");
          v20 = 2;
          if ( NtSetValueKey(KeyHandle, &ValueName, 0u, 4u, &v20, 4u) >= 0 )
          {
            RtlInitUnicodeString(&ValueName, L"Type");
            v21 = 16;
            NtSetValueKey(KeyHandle, &ValueName, 0u, 4u, &v21, 4u);
```

autochk.exe vs. autoche.exe



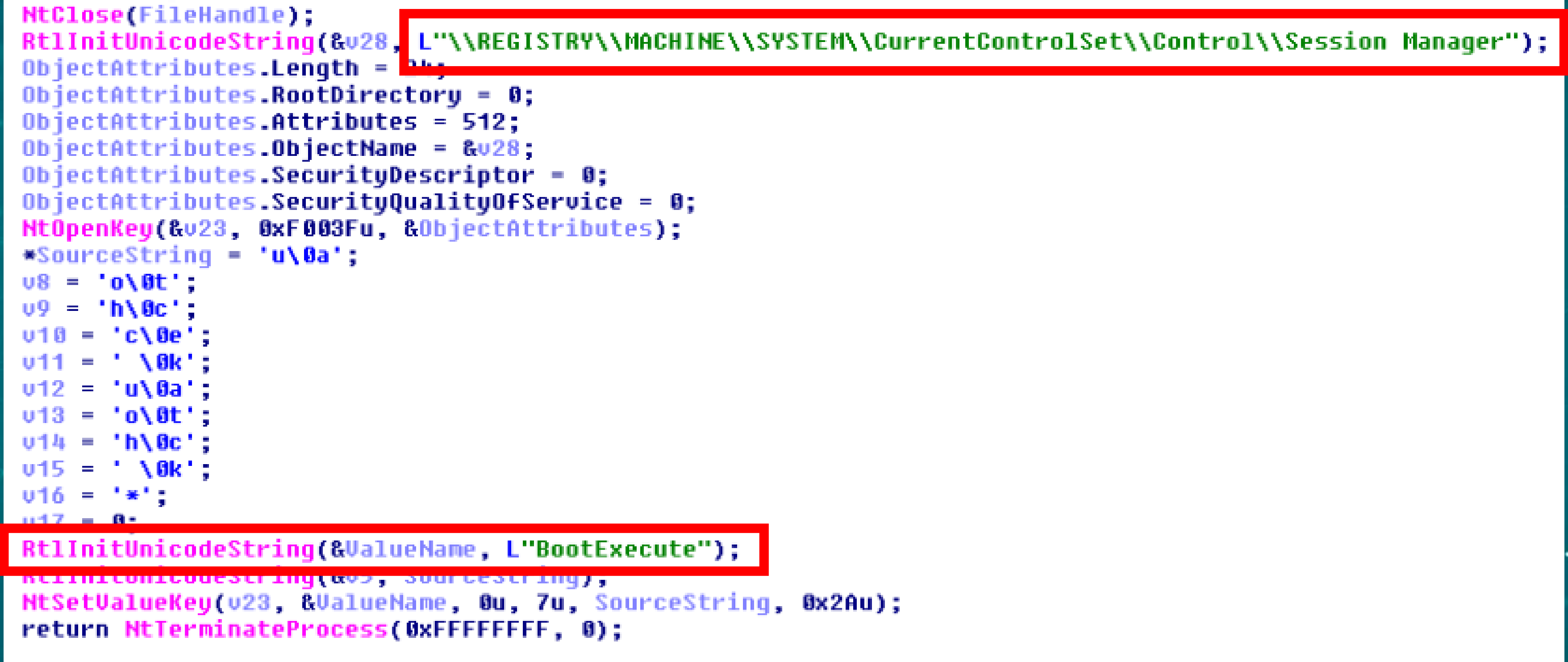
```
NtClose(FileHandle);
RtlInitUnicodeString(&v28, L"\\REGISTRY\\MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Session Manager");
ObjectAttributes.Length = 24;
ObjectAttributes.RootDirectory = 0;
ObjectAttributes.Attributes = 512;
ObjectAttributes.ObjectName = &v28;
ObjectAttributes.SecurityDescriptor = 0;
ObjectAttributes.SecurityQualityOfService = 0;
NtOpenKey(&v23, 0xF003Fu, &ObjectAttributes);
*SourceString = 'u\0a';
v8 = 'o\0t';
v9 = 'h\0c';
v10 = 'c\0e';
v11 = ' \0k';
v12 = 'u\0a';
v13 = 'o\0t';
v14 = 'h\0c';
v15 = ' \0k';
v16 = '*';
v17 = 0;
RtlInitUnicodeString(&ValueName, L"BootExecute");
RtlInitUnicodeString(&v5, SourceString);
NtSetValueKey(v23, &ValueName, 0u, 7u, SourceString, 0x2Au);
return NtTerminateProcess(0xFFFFFFFF, 0);
```

autochk.exe vs. autoche.exe



```
NtClose(FileHandle);
RtlInitUnicodeString(&v28, L"\\REGISTRY\\MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Session Manager");
ObjectAttributes.Length = 24;
ObjectAttributes.RootDirectory = 0;
ObjectAttributes.Attributes = 512;
ObjectAttributes.ObjectName = &v28;
ObjectAttributes.SecurityDescriptor = 0;
ObjectAttributes.SecurityQualityOfService = 0;
NtOpenKey(&v23, 0xF003Fu, &ObjectAttributes);
*SourceString = 'u\0a';
v8 = 'o\0t';
v9 = 'h\0c';
v10 = 'c\0e';
v11 = ' \0k';
v12 = 'u\0a';
v13 = 'o\0t';
v14 = 'h\0c';
v15 = ' \0k';
v16 = '*';
v17 = 0;
RtlInitUnicodeString(&ValueName, L"BootExecute");
RtlInitUnicodeString(&v5, SourceString);
NtSetValueKey(v23, &ValueName, 0u, 7u, SourceString, 0x2Au);
return NtTerminateProcess(0xFFFFFFFF, 0);
```

autochk.exe vs. autoche.exe



```
NtClose(FileHandle);
RtlInitUnicodeString(&v28, L"\\REGISTRY\\MACHINE\\SYSTEM\\CurrentControlSet\\Control\\Session Manager");
ObjectAttributes.Length = 24;
ObjectAttributes.RootDirectory = 0;
ObjectAttributes.Attributes = 512;
ObjectAttributes.ObjectName = &v28;
ObjectAttributes.SecurityDescriptor = 0;
ObjectAttributes.SecurityQualityOfService = 0;
NtOpenKey(&v23, 0xF003Fu, &ObjectAttributes);
*SourceString = 'u\0a';
v8 = 'o\0t';
v9 = 'h\0c';
v10 = 'c\0e';
v11 = ' \0k';
v12 = 'u\0a';
v13 = 'o\0t';
v14 = 'h\0c';
v15 = ' \0k';
v16 = '*';
v17 = 0;
RtlInitUnicodeString(&ValueName, L"BootExecute");
RtlInitUnicodeString(&v3, SourceString);
NtSetValueKey(v23, &ValueName, 0u, 7u, SourceString, 0x2Au);
return NtTerminateProcess(0xFFFFFFFF, 0);
```

Down the rings we go

ReWriter_read.exe

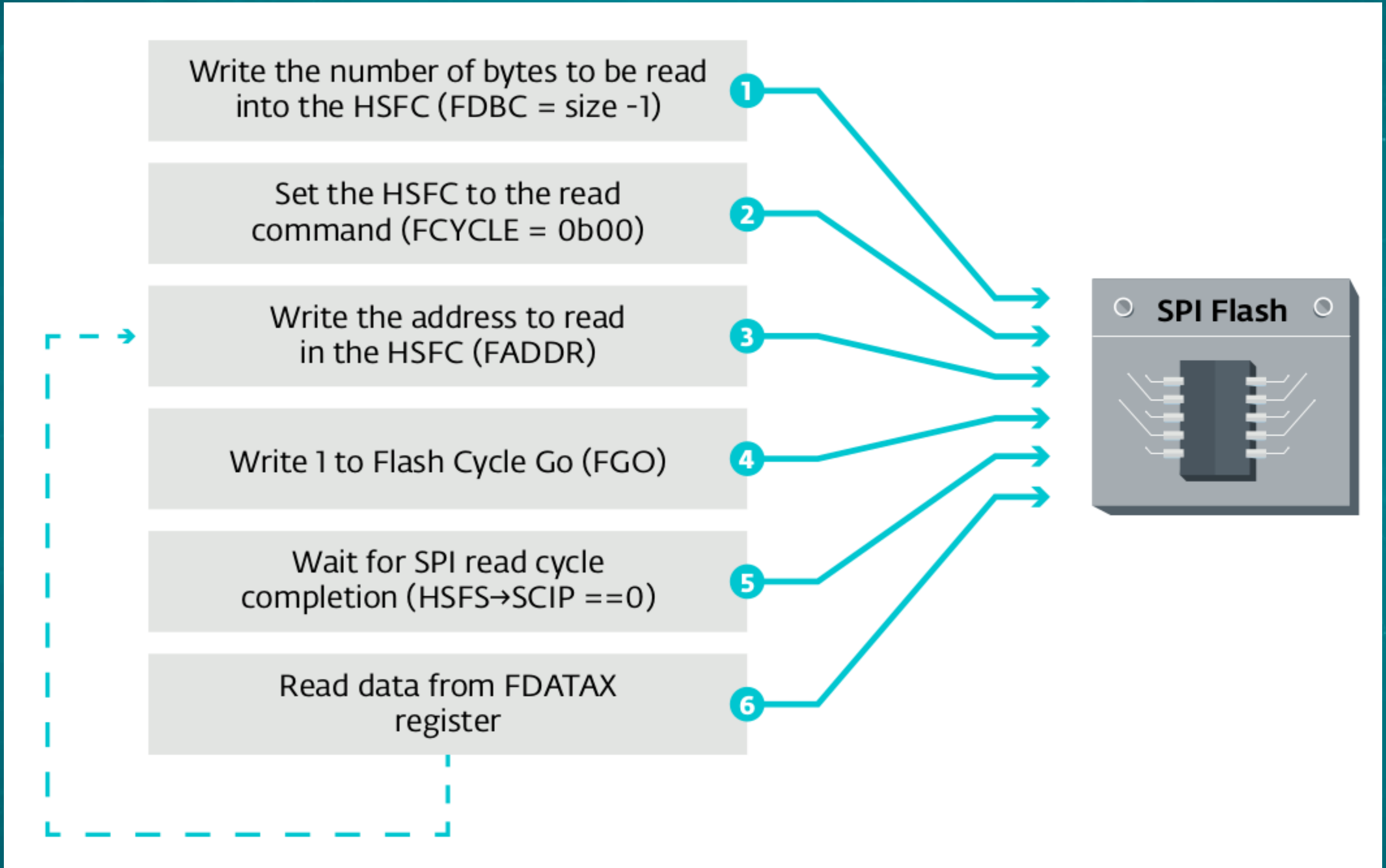
- Tool to dump SPI flash memory content found alongside LoJax sample

IOCTL code	Description
0x22280c	Writes to memory mapped I/O space
0x222808	Reads from memory mapped I/O space
0x222840	Reads a dword from given PCI Configuration Register
0x222834	Writes a byte to given PCI Configuration Register

ReWriter_read.exe

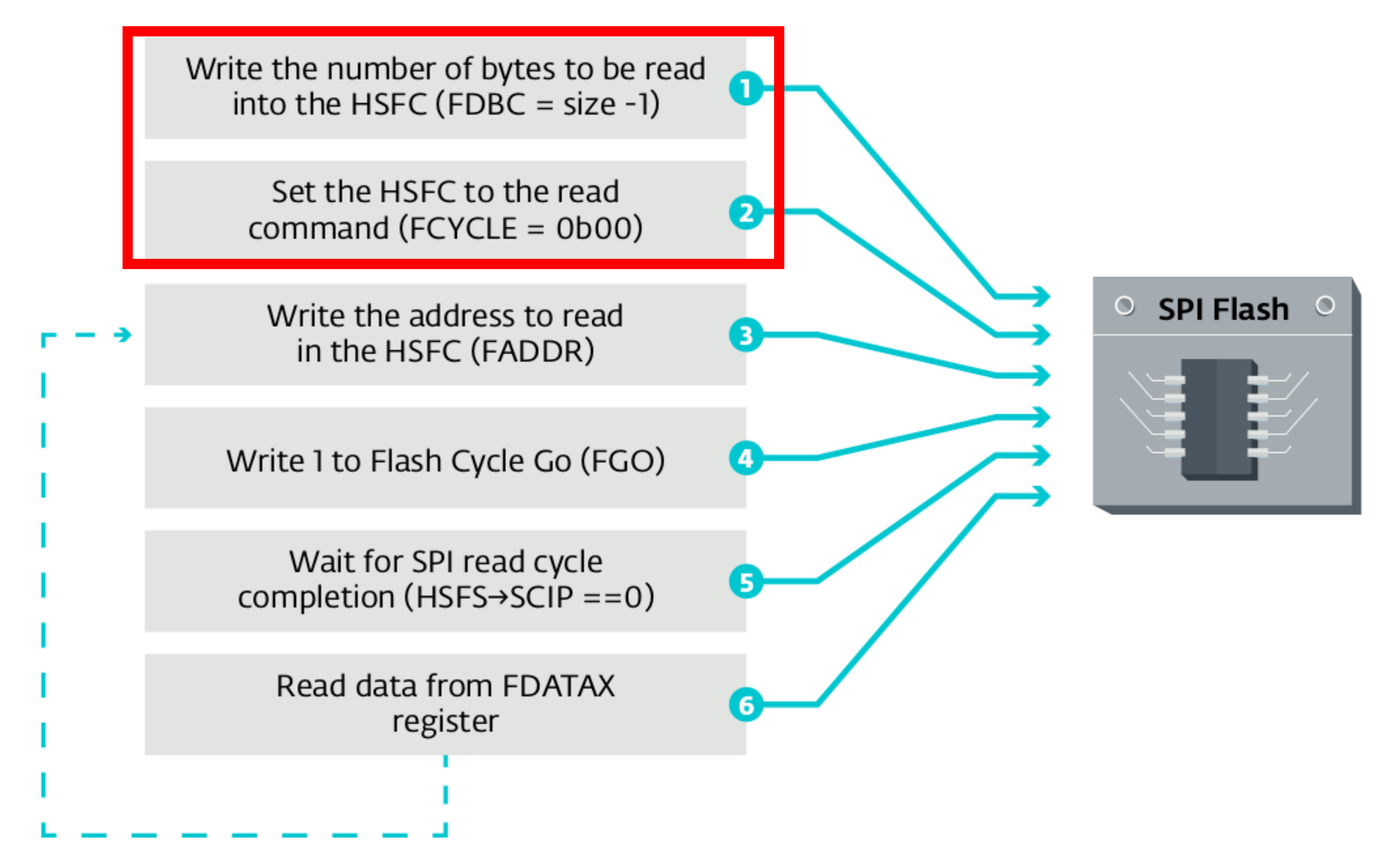
- Contains *lots* of debug strings
- Consists of the following operations
  - Log information on BIOS_CNTL register
  - Locate BIOS region base address
  - Read UEFI firmware content and dump it to a file

Reading from the SPI Flash Memory



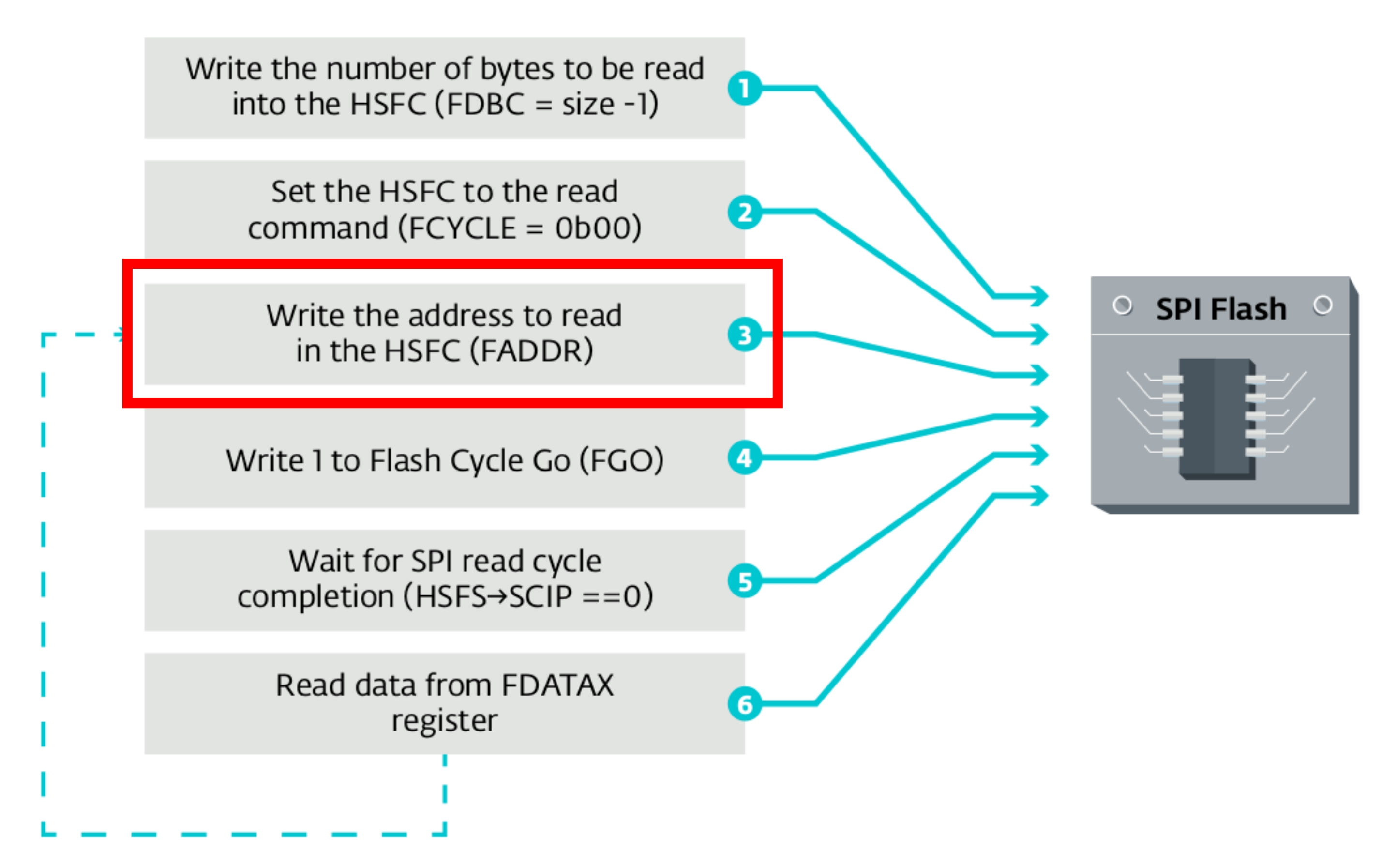
1. Write the number of bytes to be read into the HSFC (FDBC = size -1)
2. Set the HSFC to the read command (FCYCLE = 0b00)
3. Write the address to read in the HSFC (FADDR)
4. Write 1 to Flash Cycle Go (FGO)
5. Wait for SPI read cycle completion (HSFS→SCIP ==0)
6. Read data from FDATAX register

Reading from the SPI Flash Memory



1. Write the number of bytes to be read into the HSFC (FDBC = size -1)
2. Set the HSFC to the read command (FCYCLE = 0b00)
3. Write the address to read in the HSFC (FADDR)
4. Write 1 to Flash Cycle Go (FGO)
5. Wait for SPI read cycle completion (HSFS→SCIP ==0)
6. Read data from FDATAX register

Reading from the SPI Flash Memory



1 Write the number of bytes to be read into the HSFC (FDBC = size -1)

2 Set the HSFC to the read command (FCYCLE = 0b00)

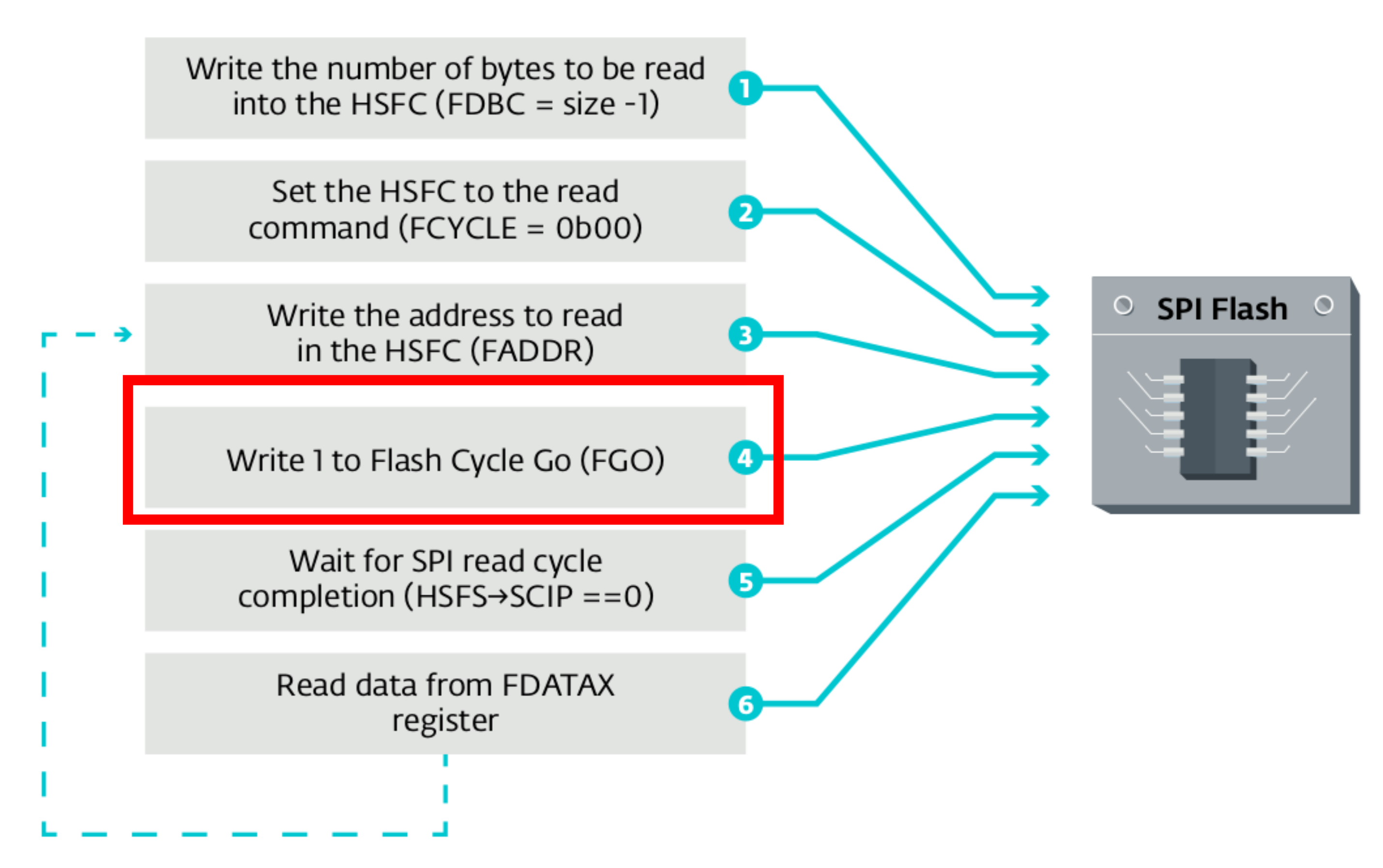
3 Write the address to read in the HSFC (FADDR)

4 Write 1 to Flash Cycle Go (FGO)

5 Wait for SPI read cycle completion (HSFS→SCIP ==0)

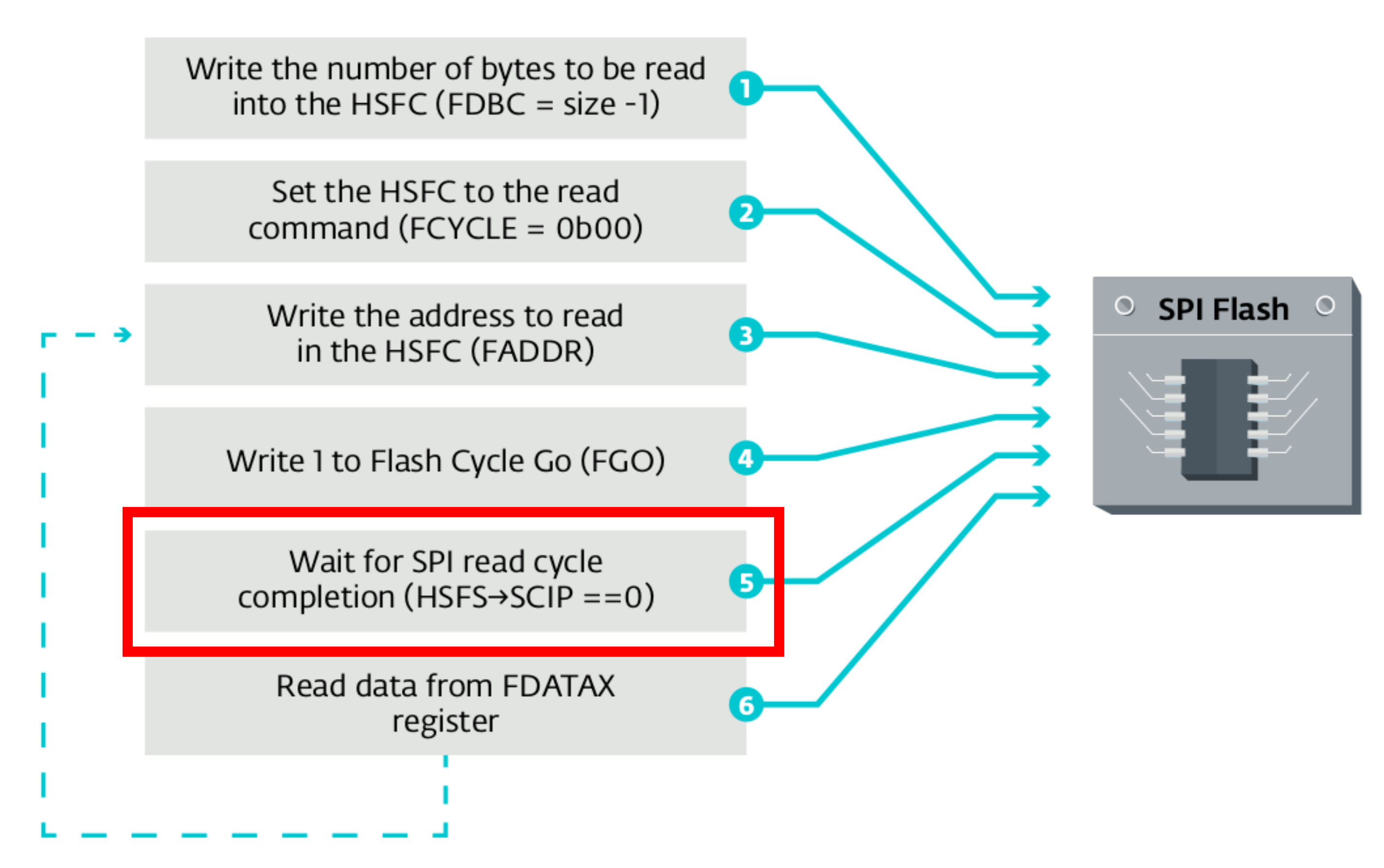
6 Read data from FDATAX register

Reading from the SPI Flash Memory



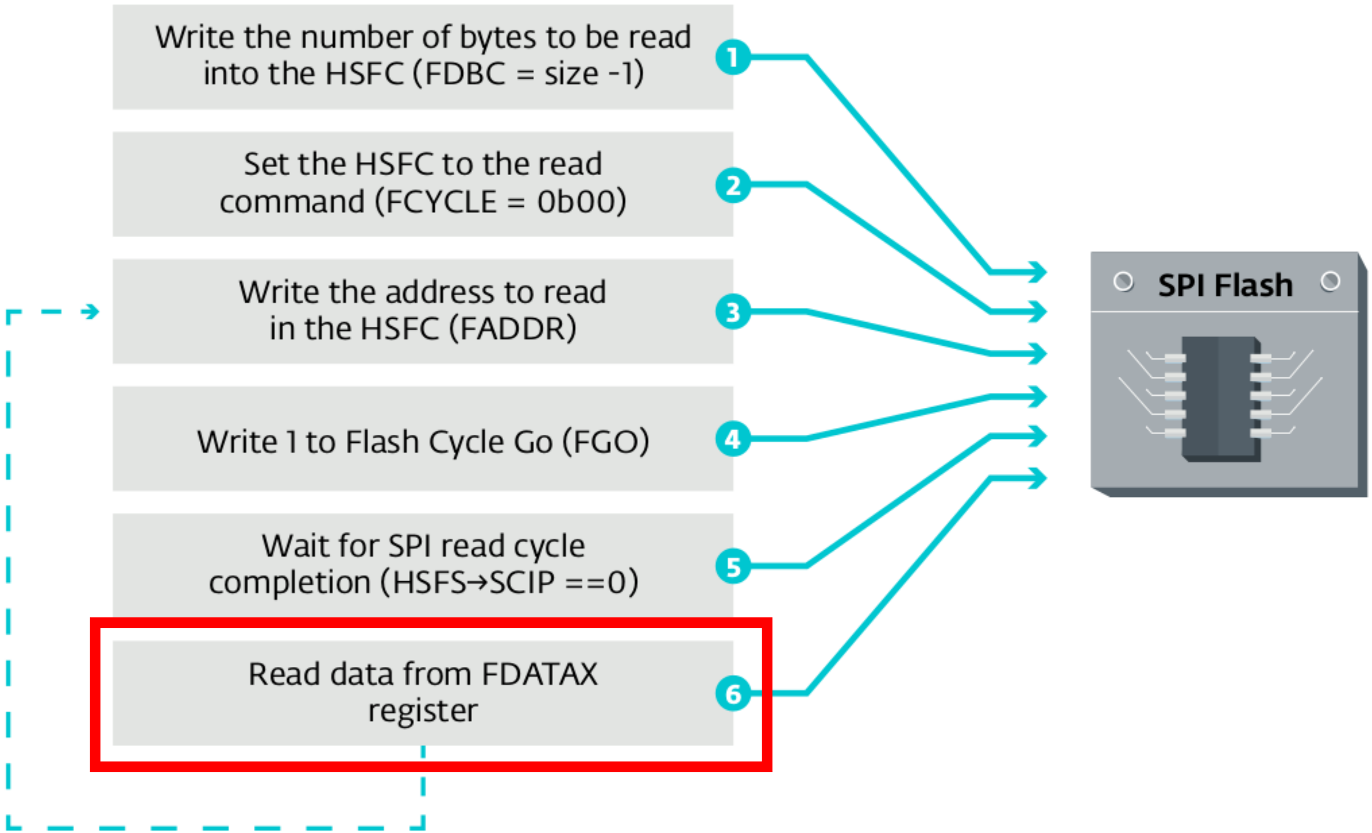
Write the number of bytes to be read into the HSFC (FDBC = size -1)
1
Set the HSFC to the read command (FCYCLE = 0b00)
2
Write the address to read in the HSFC (FADDR)
3
Write 1 to Flash Cycle Go (FGO)
4
Wait for SPI read cycle completion (HSFS→SCIP ==0)
5
Read data from FDATAX register
6
SPI Flash

Reading from the SPI Flash Memory



1. Write the number of bytes to be read into the HSFC (FDBC = size -1)
2. Set the HSFC to the read command (FCYCLE = 0b00)
3. Write the address to read in the HSFC (FADDR)
4. Write 1 to Flash Cycle Go (FGO)
5. Wait for SPI read cycle completion (HSFS→SCIP ==0)
6. Read data from FDATAX register

Reading from the SPI Flash Memory



1. Write the number of bytes to be read into the HSFC (FDBC = size -1)
2. Set the HSFC to the read command (FCYCLE = 0b00)
3. Write the address to read in the HSFC (FADDR)
4. Write 1 to Flash Cycle Go (FGO)
5. Wait for SPI read cycle completion (HSFS→SCIP ==0)
6. Read data from FDATAX register

ReWriter_binary.exe

- Contains *lots* of debug strings
- Uses RWEverything’s driver
- Consists of the following operations
  - Add the rootkit to the firmware
  - Write it back to the SPI flash memory

Patching the UEFI firmware

Unified Extensible Firmware Interface (UEFI)

- Replacement for the legacy BIOS
- New standard for firmware development
- Provides a set of services to UEFI applications
  - Boot services
  - Runtime services
- No more MBR/VBR

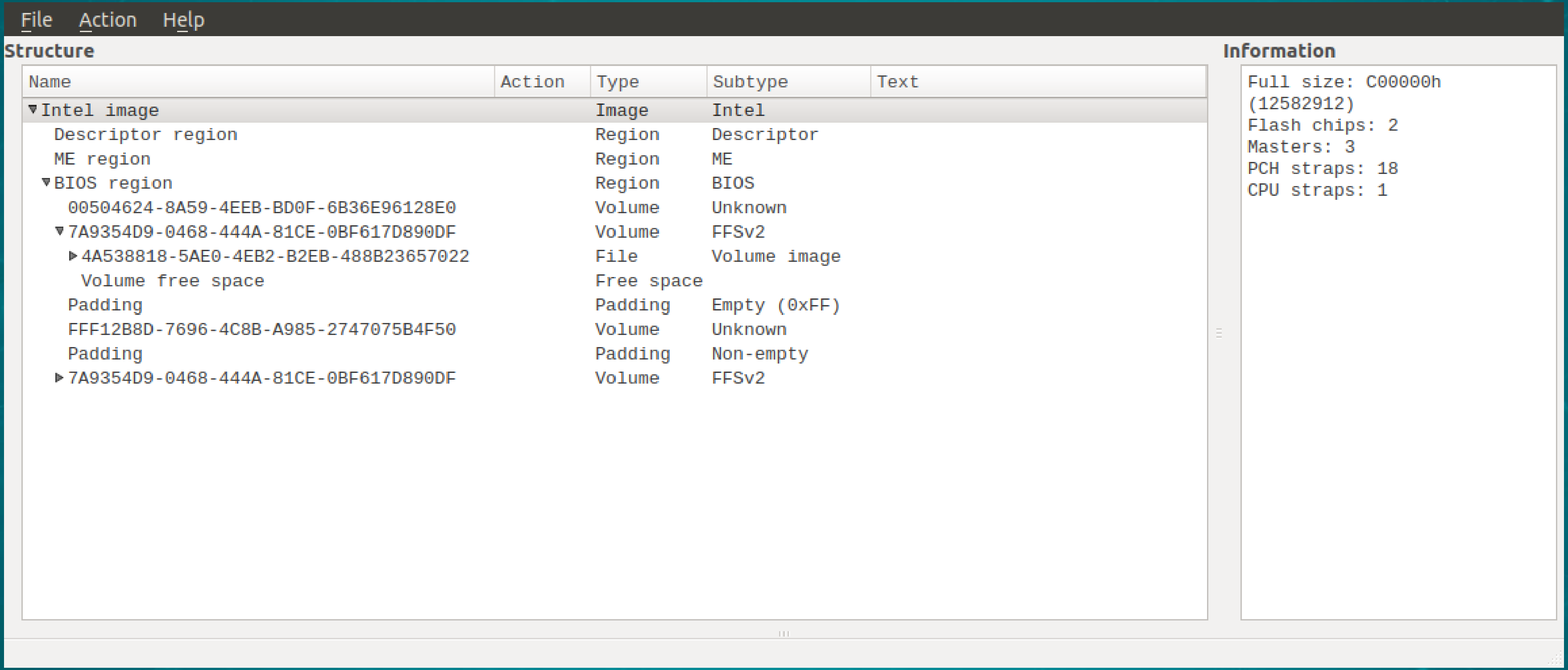
Driver Execution Environment (DXE) Drivers

- PE/COFF images
- Abstract the hardware
- Produce UEFI standard interface
- Register new services (protocols)
- Loaded during the DXE phase of the Platform initialization
- Loaded by the DXE dispatcher (DXE Core)

UEFI firmware layout

- Located in the BIOS region of the SPI flash memory
- Contains multiple volumes
  - Volumes contain files identified by GUIDs
  - File contain sections
  - One of these sections is the actual UEFI image
  - It’s more complex than that but it suffices for our purpose

SPI flash memory layout



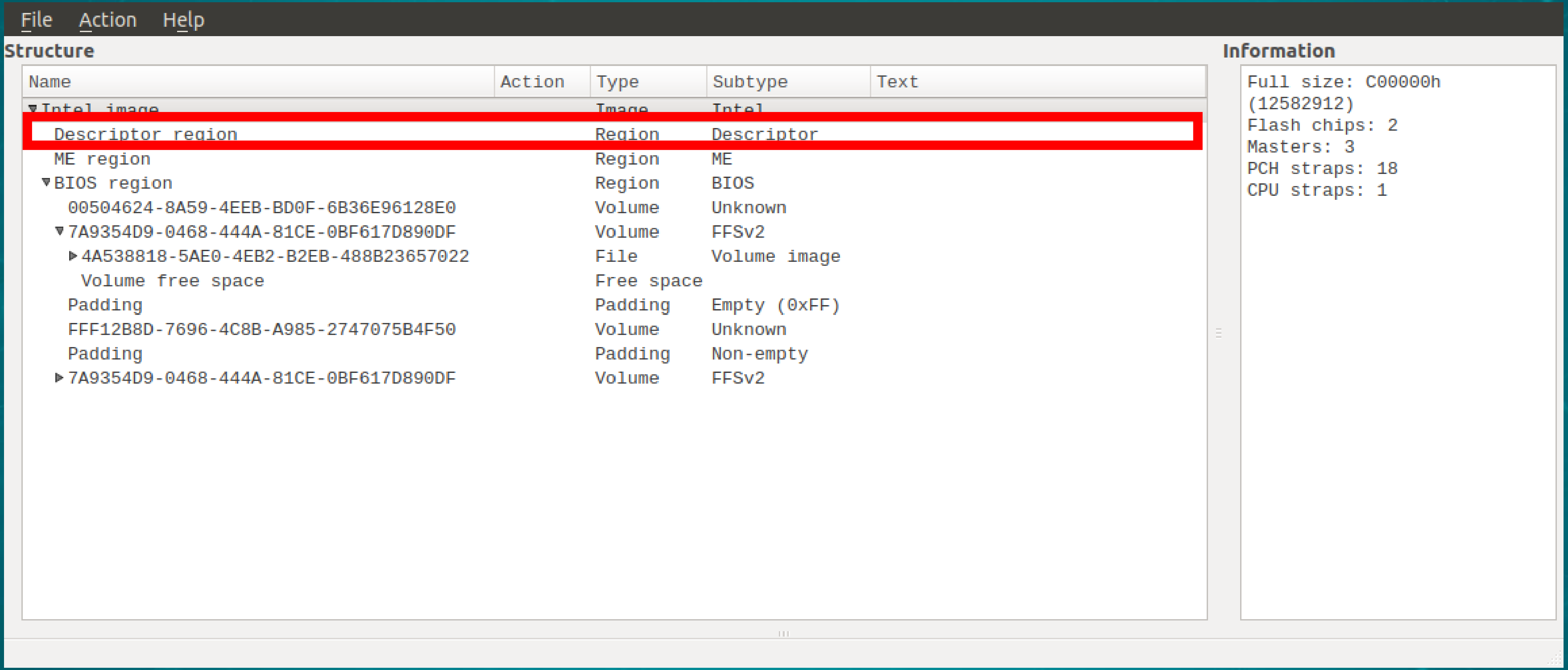
Structure

```
Name                                      Action   Type         Subtype          Text
▼Intel image                                       Image        Intel
  Descriptor region                                Region       Descriptor
  ME region                                        Region       ME
 ▼BIOS region                                      Region       BIOS
    00504624-8A59-4EEB-BD0F-6B36E96128E0           Volume       Unknown
   ▼7A9354D9-0468-444A-81CE-0BF617D890DF           Volume       FFSv2
     ▶4A538818-5AE0-4EB2-B2EB-488B23657022         File         Volume image
      Volume free space                            Free space
    Padding                                        Padding      Empty (0xFF)
    FFF12B8D-7696-4C8B-A985-2747075B4F50           Volume       Unknown
    Padding                                        Padding      Non-empty
   ▶7A9354D9-0468-444A-81CE-0BF617D890DF           Volume       FFSv2
```

Information

```
Full size: C00000h
(12582912)
Flash chips: 2
Masters: 3
PCH straps: 18
CPU straps: 1
```

SPI flash memory layout



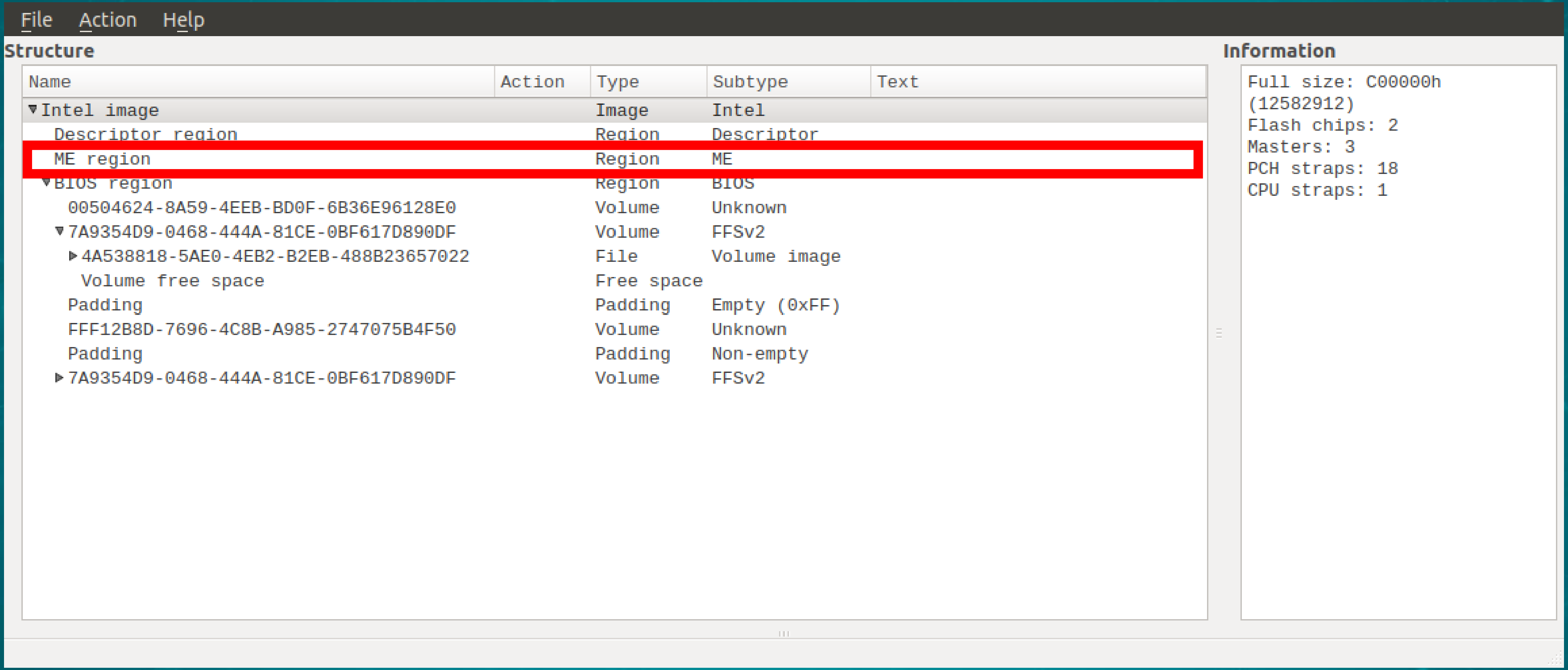
Structure

```
Name                                       Action   Type        Subtype         Text
▼Intel image                                        Image       Intel
  Descriptor region                                 Region      Descriptor
  ME region                                         Region      ME
 ▼BIOS region                                       Region      BIOS
    00504624-8A59-4EEB-BD0F-6B36E96128E0            Volume      Unknown
   ▼7A9354D9-0468-444A-81CE-0BF617D890DF            Volume      FFSv2
     ▶4A538818-5AE0-4EB2-B2EB-488B23657022          File        Volume image
      Volume free space                             Free space
    Padding                                         Padding     Empty (0xFF)
    FFF12B8D-7696-4C8B-A985-2747075B4F50            Volume      Unknown
    Padding                                         Padding     Non-empty
   ▶7A9354D9-0468-444A-81CE-0BF617D890DF            Volume      FFSv2
```

Information

```
Full size: C00000h
(12582912)
Flash chips: 2
Masters: 3
PCH straps: 18
CPU straps: 1
```

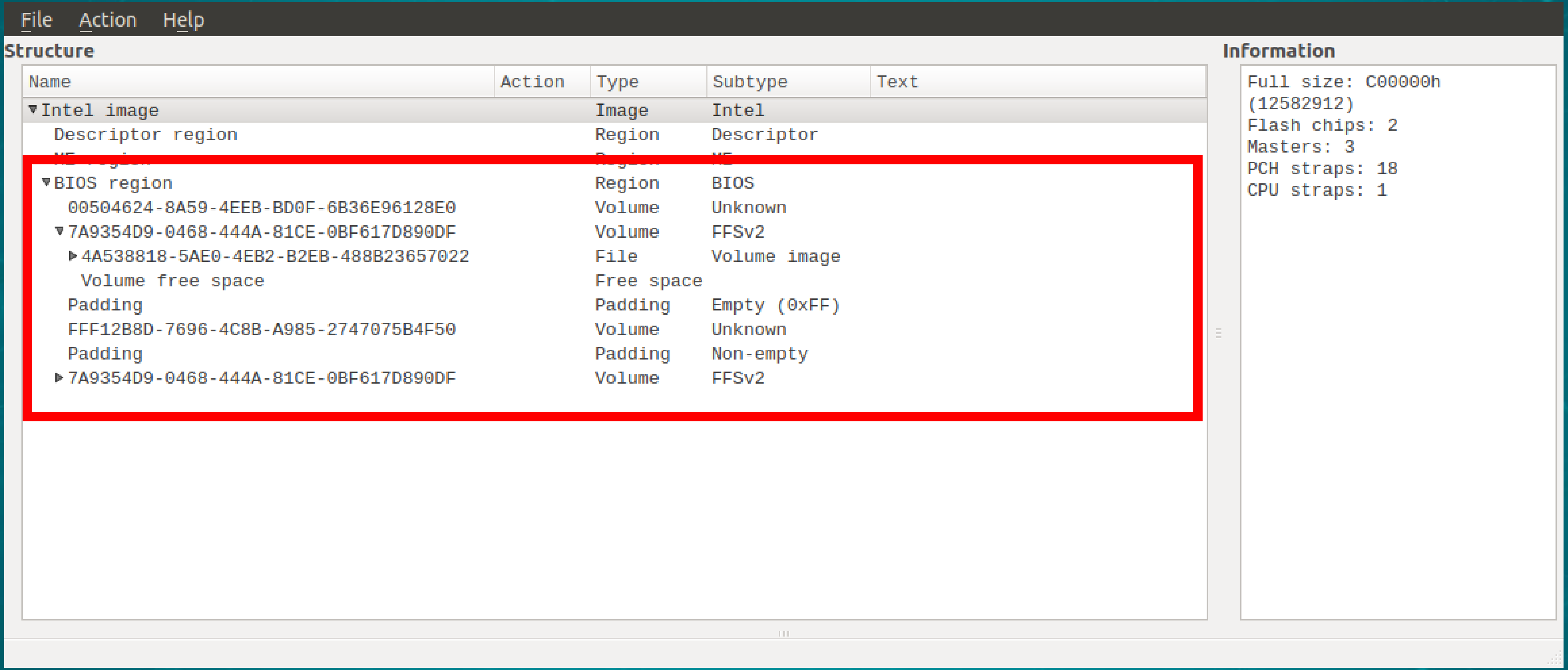
SPI flash memory layout



Name	Action	Type	Subtype	Text
▼Intel image		Image	Intel	
Descriptor region		Region	Descriptor	
ME region		Region	ME	
▼BIOS region		Region	BIOS	
00504624-8A59-4EEB-BD0F-6B36E96128E0		Volume	Unknown	
▼7A9354D9-0468-444A-81CE-0BF617D890DF		Volume	FFSv2	
▶4A538818-5AE0-4EB2-B2EB-488B23657022		File	Volume image	
Volume free space		Free space		
Padding		Padding	Empty (0xFF)	
FFF12B8D-7696-4C8B-A985-2747075B4F50		Volume	Unknown	
Padding		Padding	Non-empty	
▶7A9354D9-0468-444A-81CE-0BF617D890DF		Volume	FFSv2	

```
Full size: C00000h
(12582912)
Flash chips: 2
Masters: 3
PCH straps: 18
CPU straps: 1
```

SPI flash memory layout



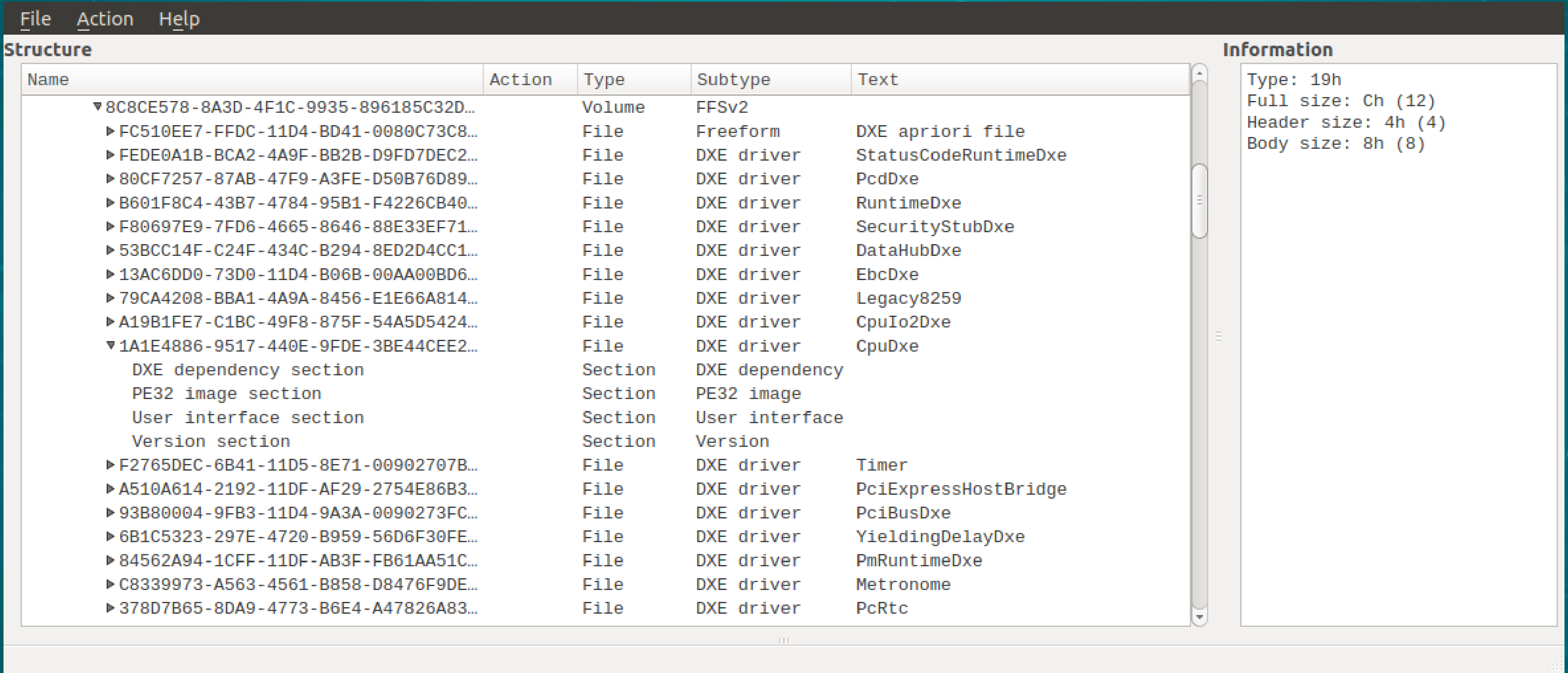
Structure

Name	Action	Type	Subtype	Text
▼Intel image		Image	Intel	
Descriptor region		Region	Descriptor	
ME region		Region	ME	
▼BIOS region		Region	BIOS	
00504624-8A59-4EEB-BD0F-6B36E96128E0		Volume	Unknown	
▼7A9354D9-0468-444A-81CE-0BF617D890DF		Volume	FFSv2	
▶4A538818-5AE0-4EB2-B2EB-488B23657022		File	Volume image	
Volume free space		Free space		
Padding		Padding	Empty (0xFF)	
FFF12B8D-7696-4C8B-A985-2747075B4F50		Volume	Unknown	
Padding		Padding	Non-empty	
▶7A9354D9-0468-444A-81CE-0BF617D890DF		Volume	FFSv2	

Information

```
Full size: C00000h
(12582912)
Flash chips: 2
Masters: 3
PCH straps: 18
CPU straps: 1
```

BIOS region layout



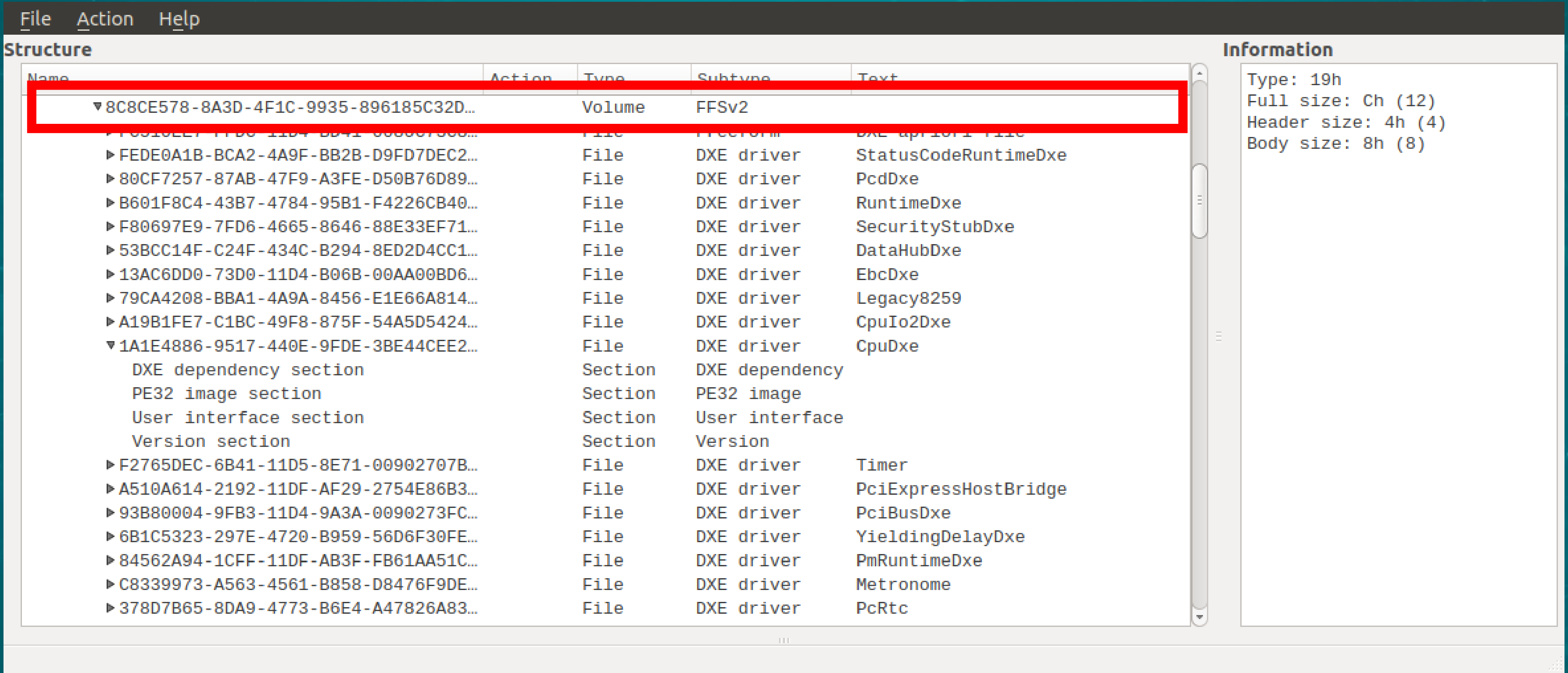
Structure

Name	Action	Type	Subtype	Text
▼8C8CE578-8A3D-4F1C-9935-896185C32D…		Volume	FFSv2	
▶FC510EE7-FFDC-11D4-BD41-0080C73C8…		File	Freeform	DXE apriori file
▶FEDE0A1B-BCA2-4A9F-BB2B-D9FD7DEC2…		File	DXE driver	StatusCodeRuntimeDxe
▶80CF7257-87AB-47F9-A3FE-D50B76D89…		File	DXE driver	PcdDxe
▶B601F8C4-43B7-4784-95B1-F4226CB40…		File	DXE driver	RuntimeDxe
▶F80697E9-7FD6-4665-8646-88E33EF71…		File	DXE driver	SecurityStubDxe
▶53BCC14F-C24F-434C-B294-8ED2D4CC1…		File	DXE driver	DataHubDxe
▶13AC6DD0-73D0-11D4-B06B-00AA00BD6…		File	DXE driver	EbcDxe
▶79CA4208-BBA1-4A9A-8456-E1E66A814…		File	DXE driver	Legacy8259
▶A19B1FE7-C1BC-49F8-875F-54A5D5424…		File	DXE driver	CpuIo2Dxe
▼1A1E4886-9517-440E-9FDE-3BE44CEE2…		File	DXE driver	CpuDxe
DXE dependency section		Section	DXE dependency	
PE32 image section		Section	PE32 image	
User interface section		Section	User interface	
Version section		Section	Version	
▶F2765DEC-6B41-11D5-8E71-00902707B…		File	DXE driver	Timer
▶A510A614-2192-11DF-AF29-2754E86B3…		File	DXE driver	PciExpressHostBridge
▶93B80004-9FB3-11D4-9A3A-0090273FC…		File	DXE driver	PciBusDxe
▶6B1C5323-297E-4720-B959-56D6F30FE…		File	DXE driver	YieldingDelayDxe
▶84562A94-1CFF-11DF-AB3F-FB61AA51C…		File	DXE driver	PmRuntimeDxe
▶C8339973-A563-4561-B858-D8476F9DE…		File	DXE driver	Metronome
▶378D7B65-8DA9-4773-B6E4-A47826A83…		File	DXE driver	PcRtc

Information

```
Type: 19h
Full size: Ch (12)
Header size: 4h (4)
Body size: 8h (8)
```

BIOS region layout



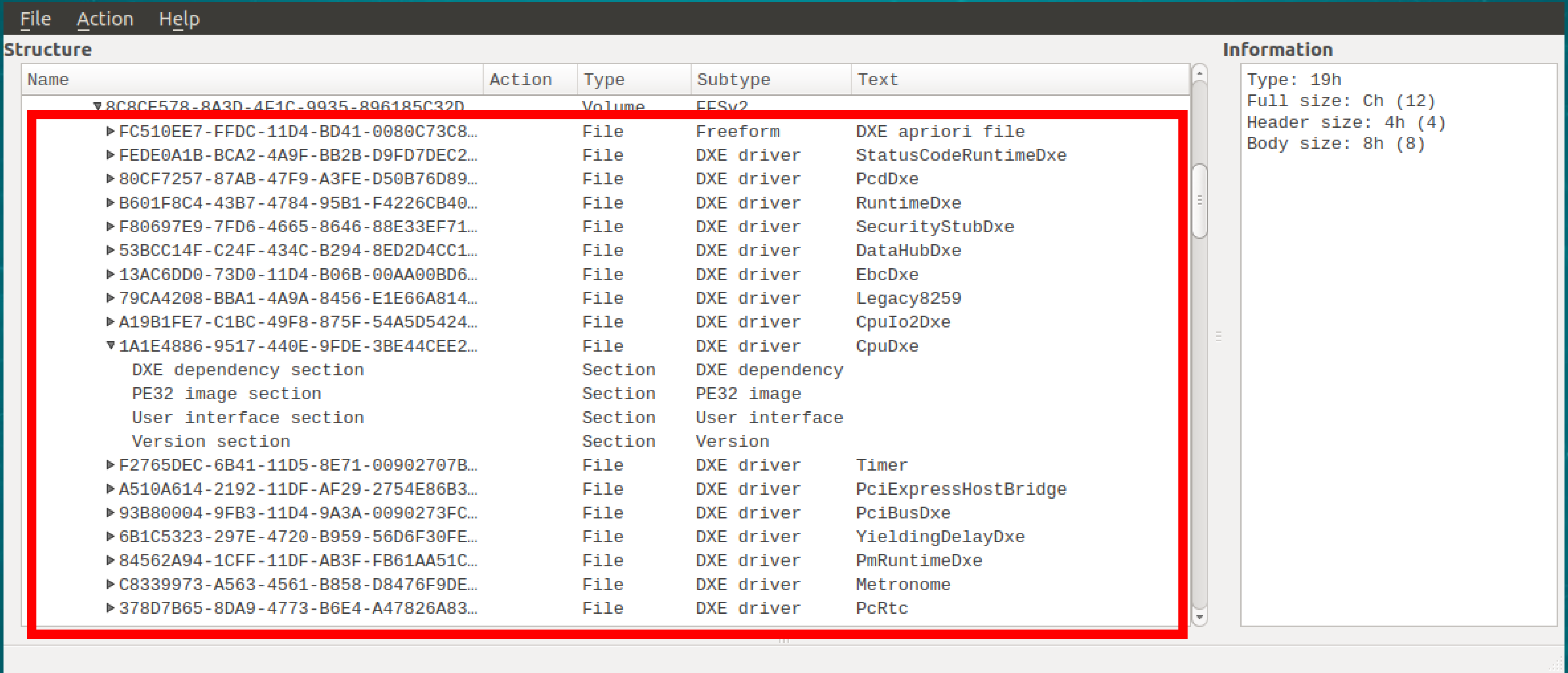
Structure

Name	Action	Type	Subtype	Text
▾8C8CE578-8A3D-4F1C-9935-896185C32D…		Volume	FFSv2	
▸FC510EE7-FFDC-11D4-BD41-0080C73C88…		File	Freeform	DXE apriori file
▸FEDE0A1B-BCA2-4A9F-BB2B-D9FD7DEC2…		File	DXE driver	StatusCodeRuntimeDxe
▸80CF7257-87AB-47F9-A3FE-D50B76D89…		File	DXE driver	PcdDxe
▸B601F8C4-43B7-4784-95B1-F4226CB40…		File	DXE driver	RuntimeDxe
▸F80697E9-7FD6-4665-8646-88E33EF71…		File	DXE driver	SecurityStubDxe
▸53BCC14F-C24F-434C-B294-8ED2D4CC1…		File	DXE driver	DataHubDxe
▸13AC6DD0-73D0-11D4-B06B-00AA00BD6…		File	DXE driver	EbcDxe
▸79CA4208-BBA1-4A9A-8456-E1E66A814…		File	DXE driver	Legacy8259
▸A19B1FE7-C1BC-49F8-875F-54A5D5424…		File	DXE driver	CpuIo2Dxe
▾1A1E4886-9517-440E-9FDE-3BE44CEE2…		File	DXE driver	CpuDxe
DXE dependency section		Section	DXE dependency	
PE32 image section		Section	PE32 image	
User interface section		Section	User interface	
Version section		Section	Version	
▸F2765DEC-6B41-11D5-8E71-00902707B…		File	DXE driver	Timer
▸A510A614-2192-11DF-AF29-2754E86B3…		File	DXE driver	PciExpressHostBridge
▸93B80004-9FB3-11D4-9A3A-0090273FC…		File	DXE driver	PciBusDxe
▸6B1C5323-297E-4720-B959-56D6F30FE…		File	DXE driver	YieldingDelayDxe
▸84562A94-1CFF-11DF-AB3F-FB61AA51C…		File	DXE driver	PmRuntimeDxe
▸C8339973-A563-4561-B858-D8476F9DE…		File	DXE driver	Metronome
▸378D7B65-8DA9-4773-B6E4-A47826A83…		File	DXE driver	PcRtc

Information

```
Type: 19h
Full size: Ch (12)
Header size: 4h (4)
Body size: 8h (8)
```

BIOS region layout



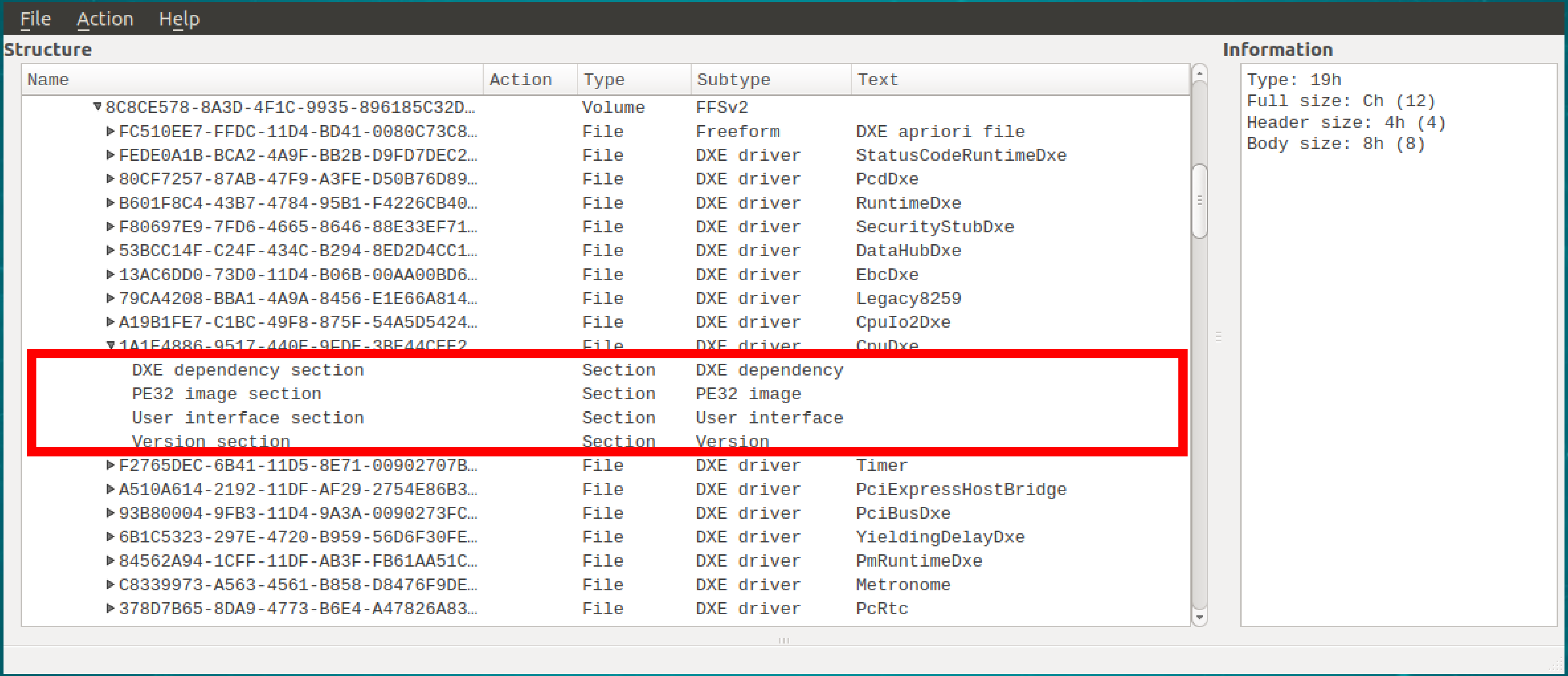
Structure

Name	Action	Type	Subtype	Text
▼8C8CE578-8A3D-4F1C-9935-896185C32D...		Volume	FFSv2	
▶FC510EE7-FFDC-11D4-BD41-0080C73C8...		File	Freeform	DXE apriori file
▶FEDE0A1B-BCA2-4A9F-BB2B-D9FD7DEC2...		File	DXE driver	StatusCodeRuntimeDxe
▶80CF7257-87AB-47F9-A3FE-D50B76D89...		File	DXE driver	PcdDxe
▶B601F8C4-43B7-4784-95B1-F4226CB40...		File	DXE driver	RuntimeDxe
▶F80697E9-7FD6-4665-8646-88E33EF71...		File	DXE driver	SecurityStubDxe
▶53BCC14F-C24F-434C-B294-8ED2D4CC1...		File	DXE driver	DataHubDxe
▶13AC6DD0-73D0-11D4-B06B-00AA00BD6...		File	DXE driver	EbcDxe
▶79CA4208-BBA1-4A9A-8456-E1E66A814...		File	DXE driver	Legacy8259
▶A19B1FE7-C1BC-49F8-875F-54A5D5424...		File	DXE driver	CpuIo2Dxe
▼1A1E4886-9517-440E-9FDE-3BE44CEE2...		File	DXE driver	CpuDxe
DXE dependency section		Section	DXE dependency	
PE32 image section		Section	PE32 image	
User interface section		Section	User interface	
Version section		Section	Version	
▶F2765DEC-6B41-11D5-8E71-00902707B...		File	DXE driver	Timer
▶A510A614-2192-11DF-AF29-2754E86B3...		File	DXE driver	PciExpressHostBridge
▶93B80004-9FB3-11D4-9A3A-0090273FC...		File	DXE driver	PciBusDxe
▶6B1C5323-297E-4720-B959-56D6F30FE...		File	DXE driver	YieldingDelayDxe
▶84562A94-1CFF-11DF-AB3F-FB61AA51C...		File	DXE driver	PmRuntimeDxe
▶C8339973-A563-4561-B858-D8476F9DE...		File	DXE driver	Metronome
▶378D7B65-8DA9-4773-B6E4-A47826A83...		File	DXE driver	PcRtc

Information

```
Type: 19h
Full size: Ch (12)
Header size: 4h (4)
Body size: 8h (8)
```

BIOS region layout



Name	Action	Type	Subtype	Text
▼8C8CE578-8A3D-4F1C-9935-896185C32D…		Volume	FFSv2	
▶FC510EE7-FFDC-11D4-BD41-0080C73C8…		File	Freeform	DXE apriori file
▶FEDE0A1B-BCA2-4A9F-BB2B-D9FD7DEC2…		File	DXE driver	StatusCodeRuntimeDxe
▶80CF7257-87AB-47F9-A3FE-D50B76D89…		File	DXE driver	PcdDxe
▶B601F8C4-43B7-4784-95B1-F4226CB40…		File	DXE driver	RuntimeDxe
▶F80697E9-7FD6-4665-8646-88E33EF71…		File	DXE driver	SecurityStubDxe
▶53BCC14F-C24F-434C-B294-8ED2D4CC1…		File	DXE driver	DataHubDxe
▶13AC6DD0-73D0-11D4-B06B-00AA00BD6…		File	DXE driver	EbcDxe
▶79CA4208-BBA1-4A9A-8456-E1E66A814…		File	DXE driver	Legacy8259
▶A19B1FE7-C1BC-49F8-875F-54A5D5424…		File	DXE driver	CpuIo2Dxe
▼1A1E4886-9517-440E-9FDE-3BE44CEE2…		File	DXE driver	CpuDxe
DXE dependency section		Section	DXE dependency	
PE32 image section		Section	PE32 image	
User interface section		Section	User interface	
Version section		Section	Version	
▶F2765DEC-6B41-11D5-8E71-00902707B…		File	DXE driver	Timer
▶A510A614-2192-11DF-AF29-2754E86B3…		File	DXE driver	PciExpressHostBridge
▶93B80004-9FB3-11D4-9A3A-0090273FC…		File	DXE driver	PciBusDxe
▶6B1C5323-297E-4720-B959-56D6F30FE…		File	DXE driver	YieldingDelayDxe
▶84562A94-1CFF-11DF-AB3F-FB61AA51C…		File	DXE driver	PmRuntimeDxe
▶C8339973-A563-4561-B858-D8476F9DE…		File	DXE driver	Metronome
▶378D7B65-8DA9-4773-B6E4-A47826A83…		File	DXE driver	PcRtc

```
Type: 19h
Full size: Ch (12)
Header size: 4h (4)
Body size: 8h (8)
```

Parsing the firmware volumes

- Parses all the firmware volumes of the UEFI firmware
- Looks for 4 specific files
  - Ip4Dxe (8f92960f-2880-4659-b857-915a8901bdc8)
  - NtfsDxe (768bedfd-7b4b-4c9f-b2ff-6377e3387243)
  - SmiFlash (bc327dbd-b982-4f55-9f79-056ad7e987c5)
  - DXE Core

Ip4Dxe and DXE Core

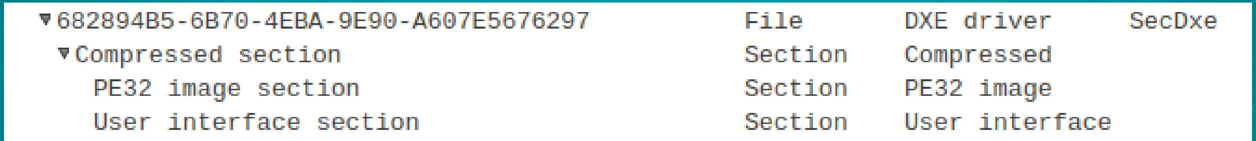
- Used to find the firmware volume to install the rootkit
- All DXE drivers are usually in the same volume
- DXE Core may be in a different volume
- The chosen volume will be the one with enough free space available

NtfsDxe and SmiFlash

- NtfsDxe the AMI NTFS driver
- Will be removed if found
- SmiFlash metadata are not used
- SmiFlash is a known-vulnerable DXE driver

Adding the rootkit

- Creates a FFS file header (EFI_FFS_FILE_HEADER)
- Append the Rootkit file



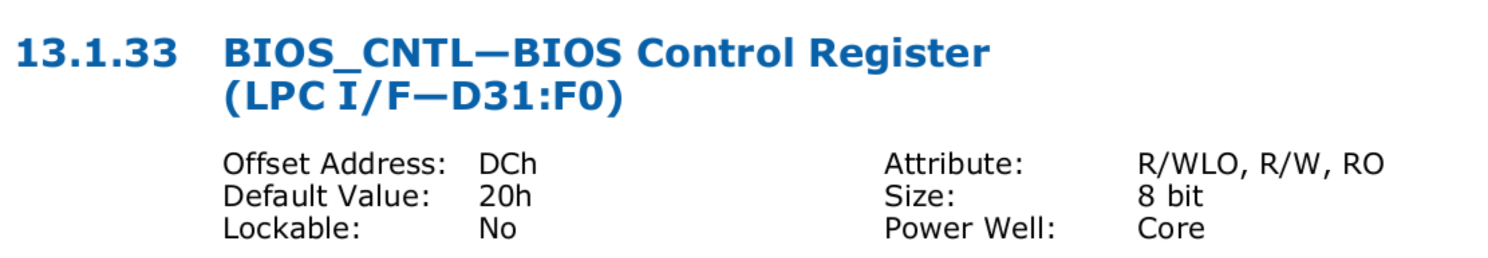
```
▼682894B5-6B70-4EBA-9E90-A607E5676297         File      DXE driver     SecDxe
 ▼Compressed section                          Section   Compressed
   PE32 image section                         Section   PE32 image
   User interface section                     Section   User interface
```

- Write it at the end of the DXE drivers volume or the DXE Core volume
  - Checks if there’s enough free space available

Write the compromised firmware to the SPI Flash memory

BIOS Write Protection Mechanisms

- Platform exposes write protection mechanisms
- Need to be properly configured by the firmware
- We’ll only cover relevant protections to our research
    - Won’t cover Protected Range Registers
- Exposed via the BIOS Control Register (BIOS_CNTL)

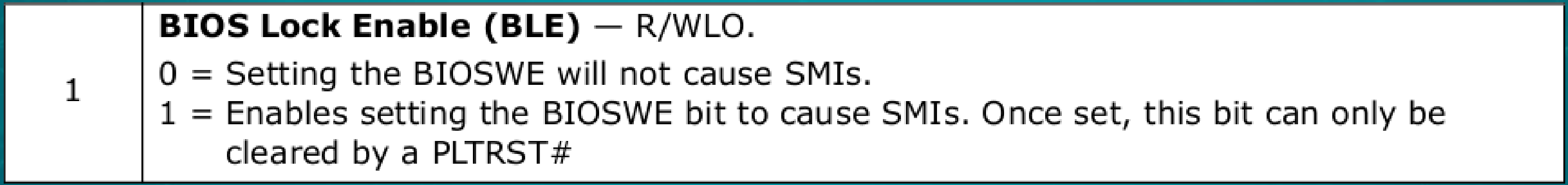


13.1.33 BIOS_CNTL—BIOS Control Register (LPC I/F—D31:F0)

Offset Address:	DCh	Attribute:	R/WLO, R/W, RO
Default Value:	20h	Size:	8 bit
Lockable:	No	Power Well:	Core

BIOS Write Protection Mechanisms

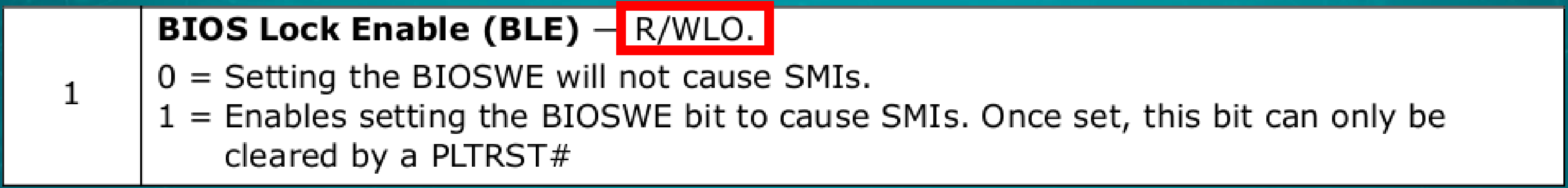
- To write to the BIOS region BIOS Write Enable (BIOSWE) must be set to 1
- BIOS Lock Enable (BLE) allows to lock BIOSWE to 0



1	**BIOS Lock Enable (BLE)** — R/WLO. 0 = Setting the BIOSWE will not cause SMIs. 1 = Enables setting the BIOSWE bit to cause SMIs. Once set, this bit can only be cleared by a PLTRST#

BIOS Write Protection Mechanisms

- To write to the BIOS region BIOS Write Enable (BIOSWE) must be set to 1
- BIOS Lock Enable (BLE) allows to lock BIOSWE to 0



1	**BIOS Lock Enable (BLE)** — R/WLO. 0 = Setting the BIOSWE will not cause SMIs. 1 = Enables setting the BIOSWE bit to cause SMIs. Once set, this bit can only be cleared by a PLTRST#

BIOS Write Protection Mechanisms

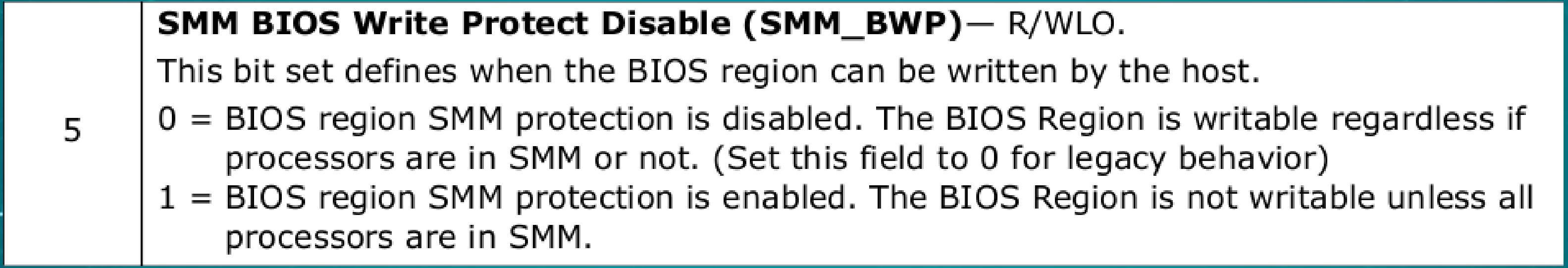
- The implementation of BLE is vulnerable
- When BIOSWE is set to 1, its value change in BIOS_CNTL
- A System Management Interrupt (SMI) is triggered
- The SMI handler sets BIOSWE back to 0
  - The SMI handler must be implemented by the firmware

BIOS Write Protection Mechanisms

- What if we write to the SPI flash memory before the SMI handler sets BIOSWE to 0?
- Race condition vulnerability (Speed racer)
  - A thread continuously set BIOSWE to 1
  - Another thread tries to write data
- Works on multicore processors and single core processors with hyper-threading enabled

BIOS Write Protection Mechanisms

- Platform Controller Hub family of Intel chipsets introduces a fix for this issue

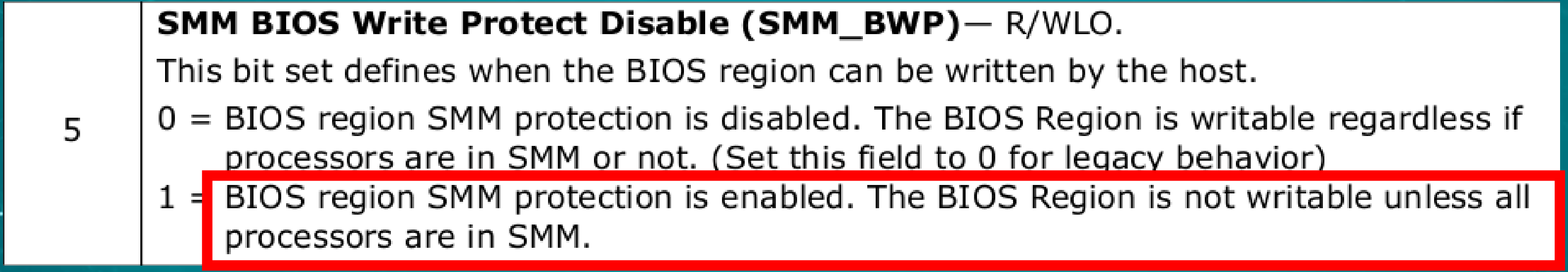


5	**SMM BIOS Write Protect Disable (SMM_BWP)**— R/WLO. This bit set defines when the BIOS region can be written by the host. 0 = BIOS region SMM protection is disabled. The BIOS Region is writable regardless if processors are in SMM or not. (Set this field to 0 for legacy behavior) 1 = BIOS region SMM protection is enabled. The BIOS Region is not writable unless all processors are in SMM.

- The firmware must set this bit

BIOS Write Protection Mechanisms

- Platform Controller Hub family of Intel chipsets introduces a fix for this issue



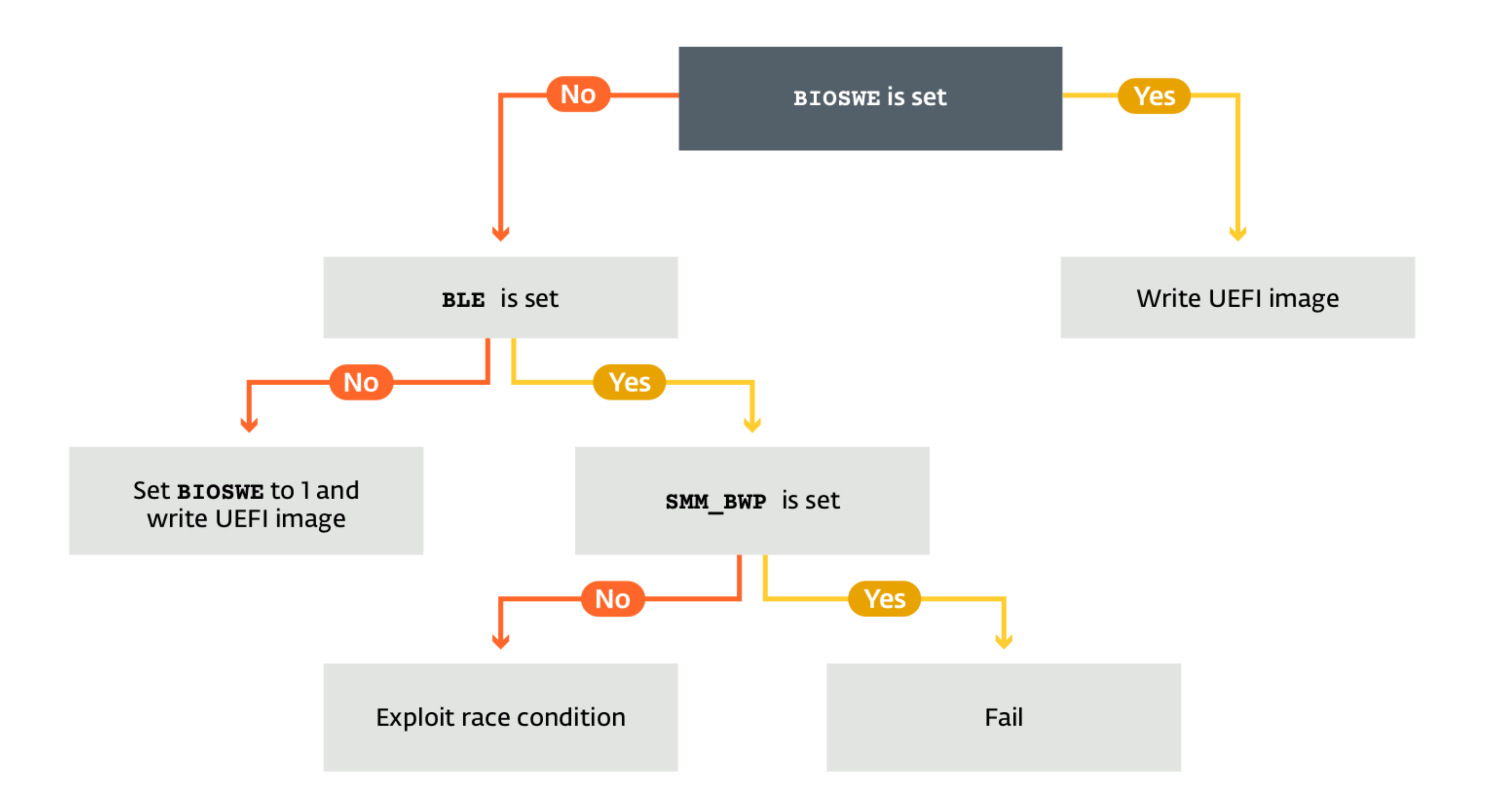
5	**SMM BIOS Write Protect Disable (SMM_BWP)**— R/WLO. This bit set defines when the BIOS region can be written by the host. 0 = BIOS region SMM protection is disabled. The BIOS Region is writable regardless if processors are in SMM or not. (Set this field to 0 for legacy behavior) 1 = BIOS region SMM protection is enabled. The BIOS Region is not writable unless all processors are in SMM.

- The firmware must set this bit

ReWriter_Binary.exe

- ReWriter_Binary.exe checks these settings
- Checks if the platform is properly configured
- Implements the exploit for the race condition

Writing process decision tree



BIOSWE is set

No

Yes

BLE is set

Write UEFI image

No

Yes

Set BIOSWE to 1 and write UEFI image

SMM_BWP is set

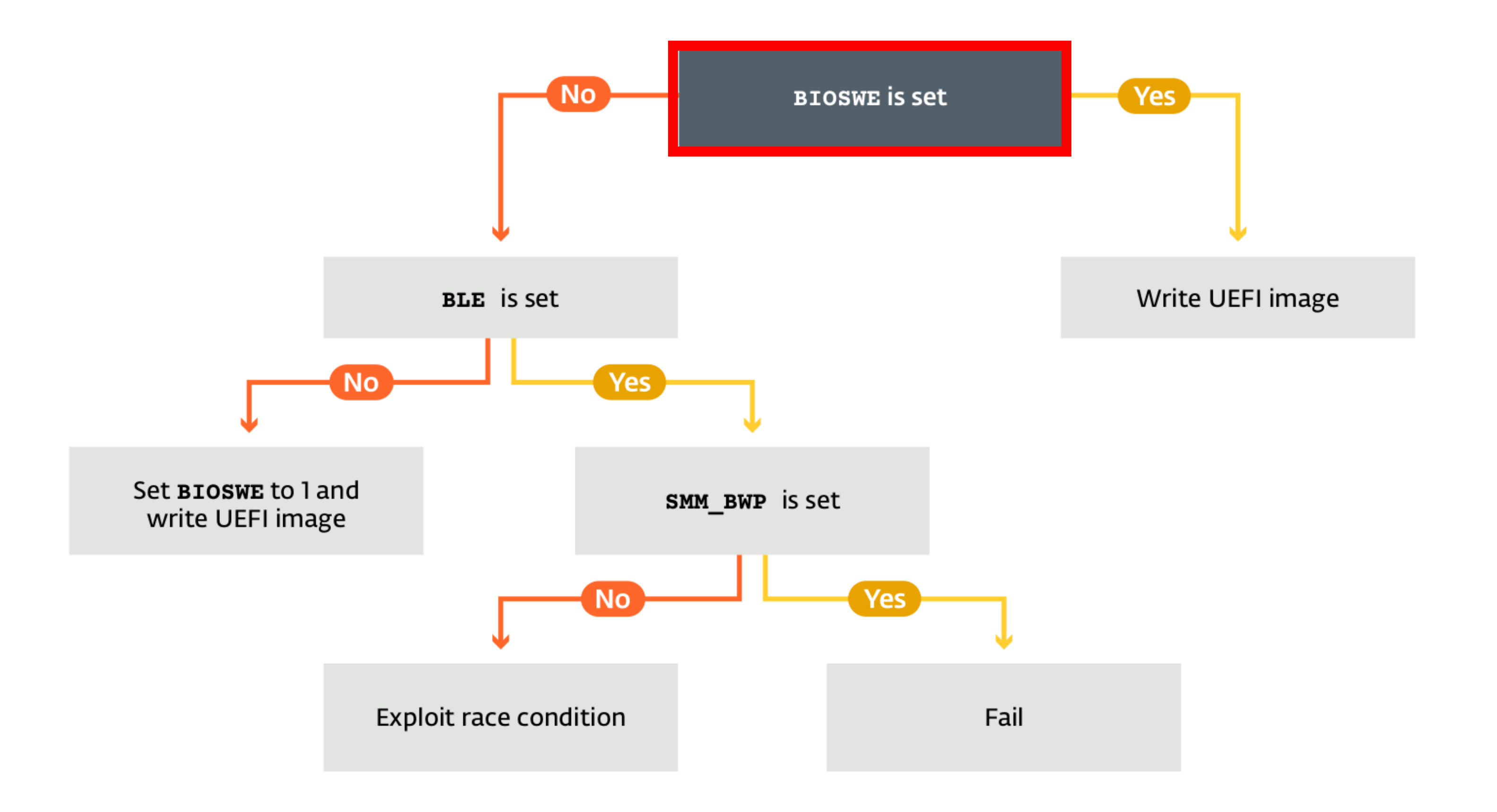
No

Yes

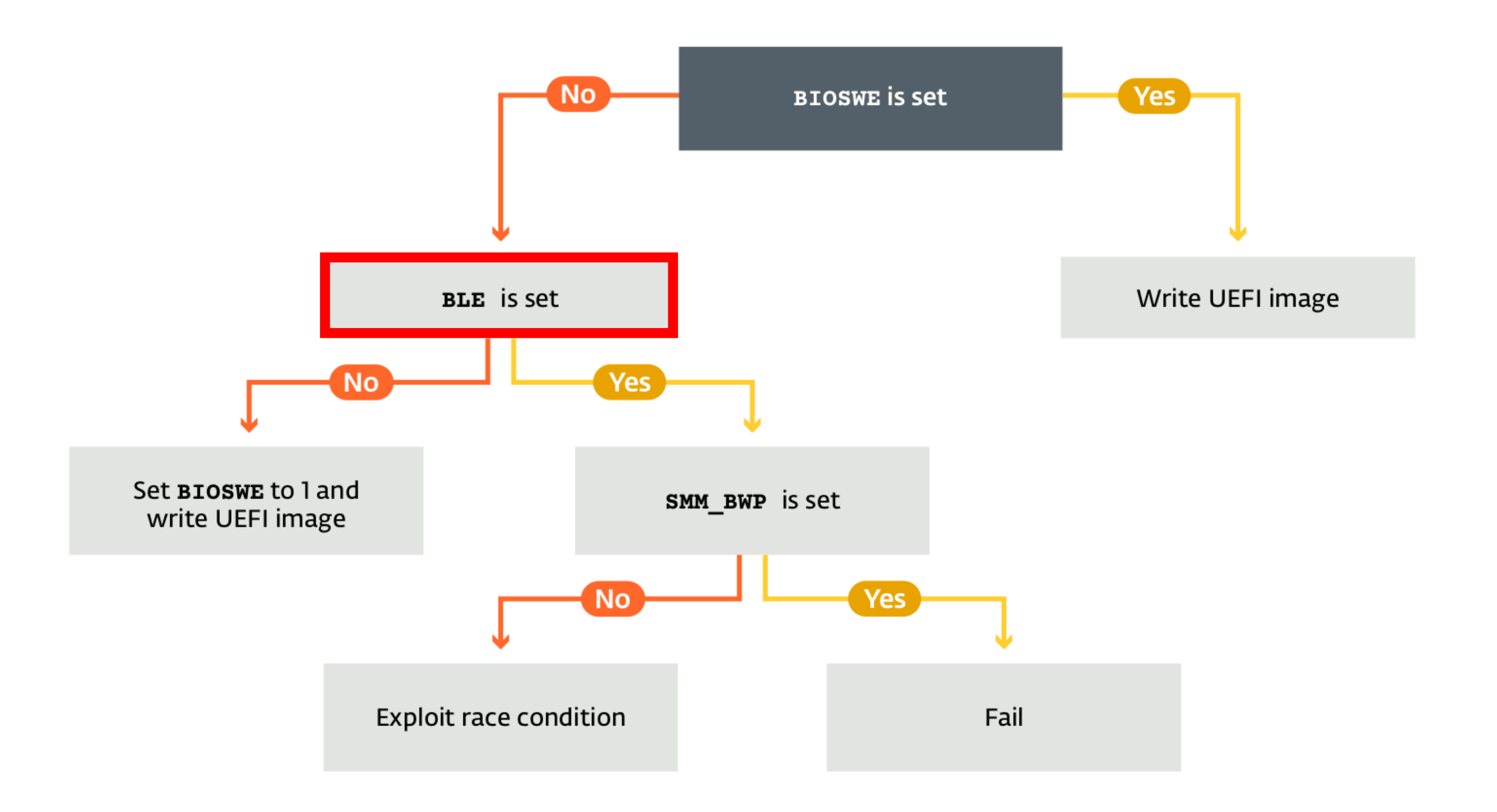
Exploit race condition

Fail

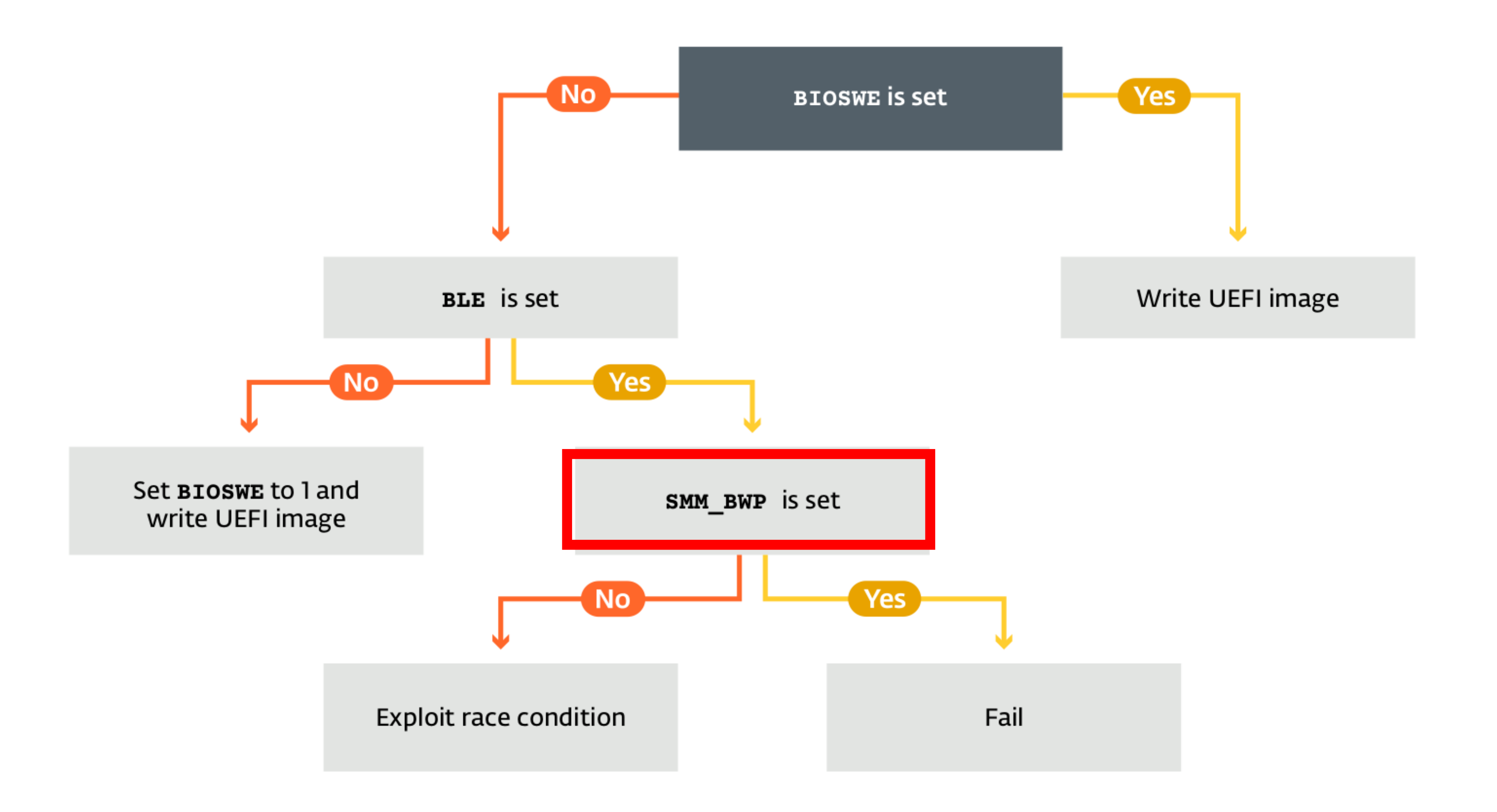
Writing process decision tree



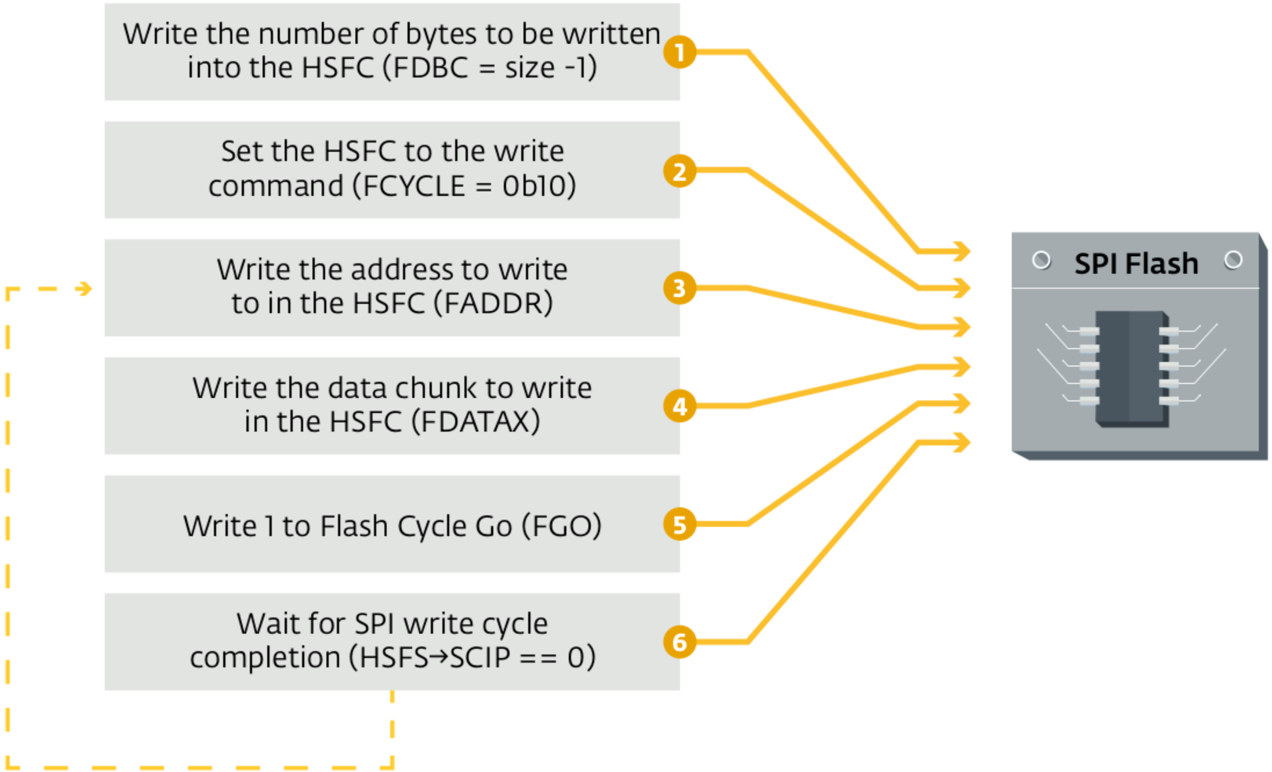
Writing process decision tree



Writing process decision tree

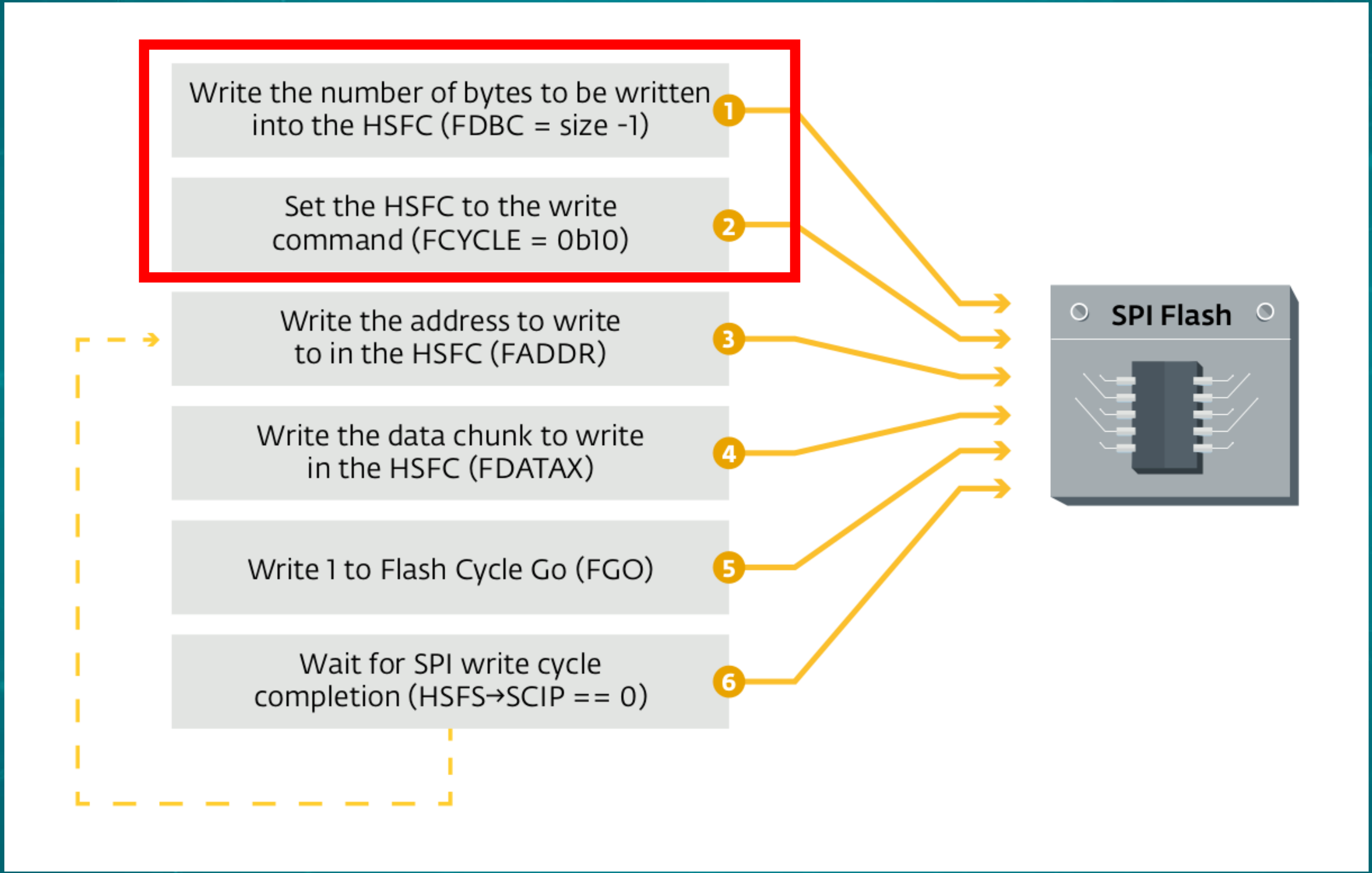


Writing to the SPI Flash Memory



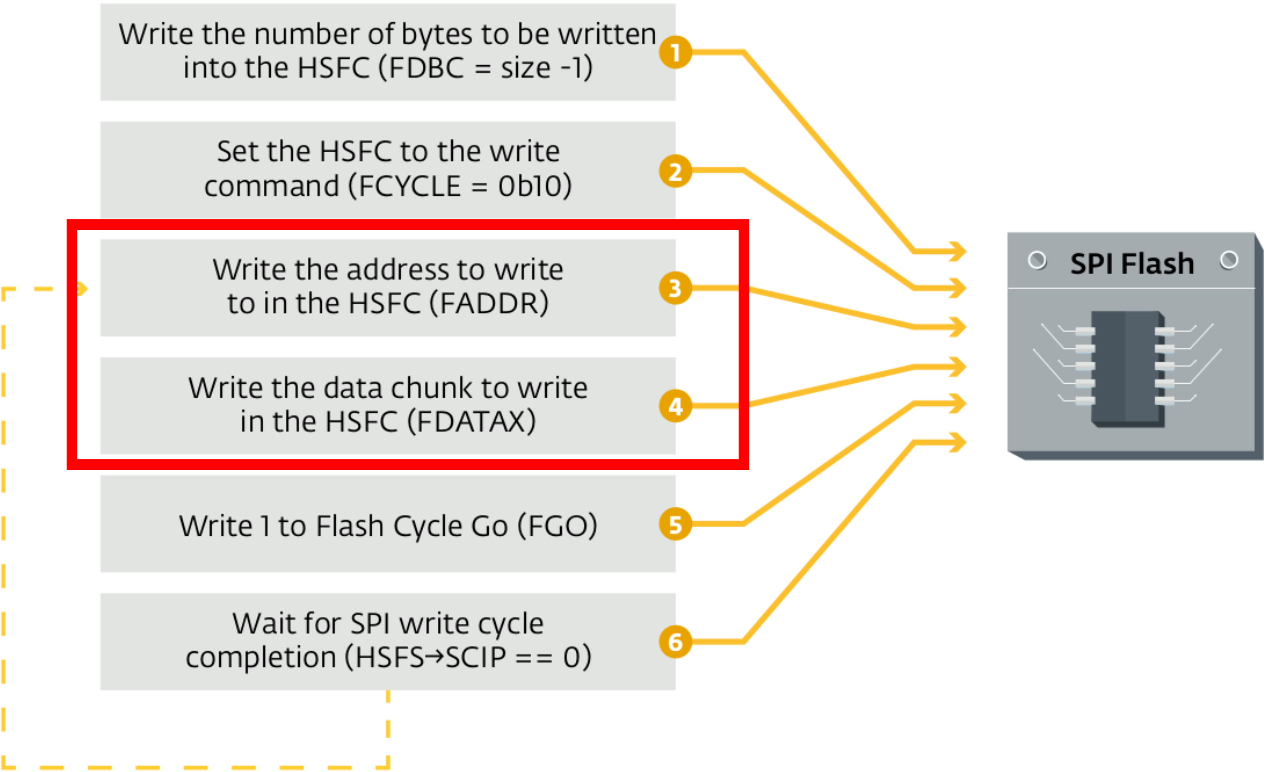
1. Write the number of bytes to be written into the HSFC (FDBC = size -1)
2. Set the HSFC to the write command (FCYCLE = 0b10)
3. Write the address to write to in the HSFC (FADDR)
4. Write the data chunk to write in the HSFC (FDATAX)
5. Write 1 to Flash Cycle Go (FGO)
6. Wait for SPI write cycle completion (HSFS→SCIP == 0)

Writing to the SPI Flash Memory



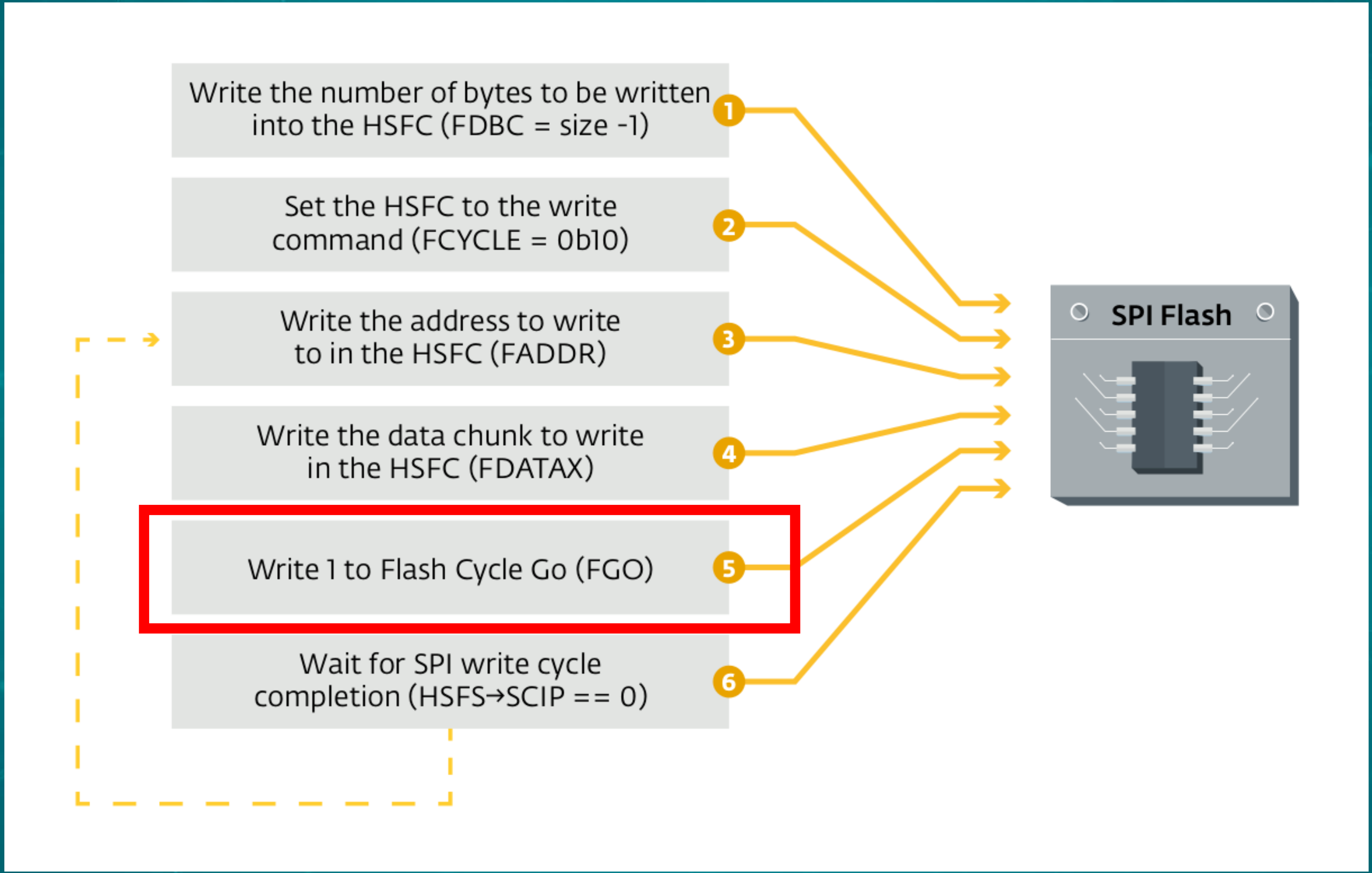
1. Write the number of bytes to be written into the HSFC (FDBC = size -1)
2. Set the HSFC to the write command (FCYCLE = 0b10)
3. Write the address to write to in the HSFC (FADDR)
4. Write the data chunk to write in the HSFC (FDATAX)
5. Write 1 to Flash Cycle Go (FGO)
6. Wait for SPI write cycle completion (HSFS→SCIP == 0)

Writing to the SPI Flash Memory



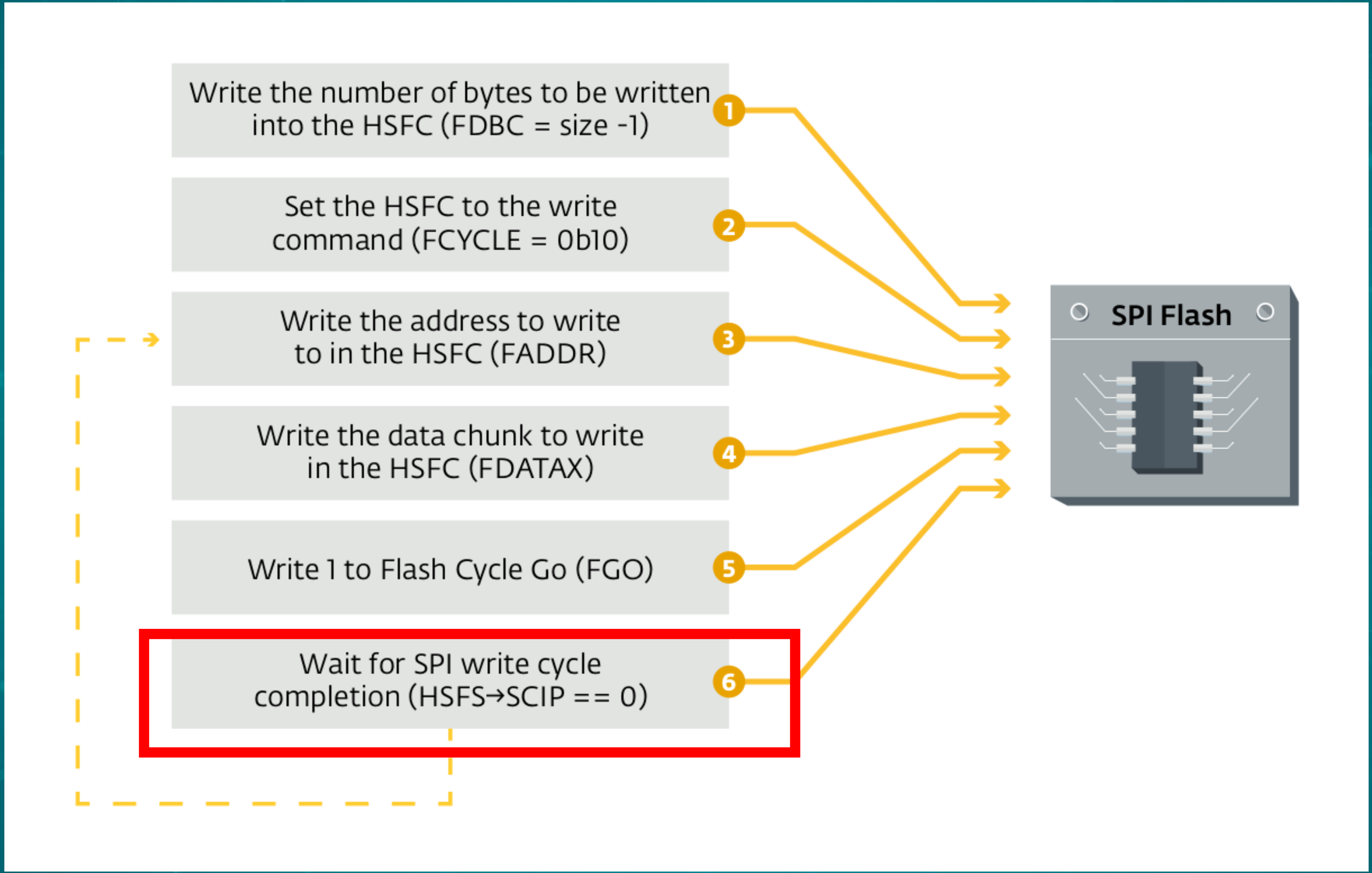
1. Write the number of bytes to be written into the HSFC (FDBC = size -1)
2. Set the HSFC to the write command (FCYCLE = 0b10)
3. Write the address to write to in the HSFC (FADDR)
4. Write the data chunk to write in the HSFC (FDATAX)
5. Write 1 to Flash Cycle Go (FGO)
6. Wait for SPI write cycle completion (HSFS→SCIP == 0)

Writing to the SPI Flash Memory



1. Write the number of bytes to be written into the HSFC (FDBC = size -1)
2. Set the HSFC to the write command (FCYCLE = 0b10)
3. Write the address to write to in the HSFC (FADDR)
4. Write the data chunk to write in the HSFC (FDATAX)
5. Write 1 to Flash Cycle Go (FGO)
6. Wait for SPI write cycle completion (HSFS→SCIP == 0)

Writing to the SPI Flash Memory



1. Write the number of bytes to be written into the HSFC (FDBC = size -1)
2. Set the HSFC to the write command (FCYCLE = 0b10)
3. Write the address to write to in the HSFC (FADDR)
4. Write the data chunk to write in the HSFC (FDATAX)
5. Write 1 to Flash Cycle Go (FGO)
6. Wait for SPI write cycle completion (HSFS→SCIP == 0)

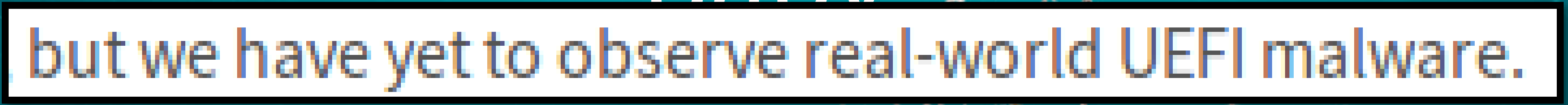
SPI Flash

Let’s take a step back

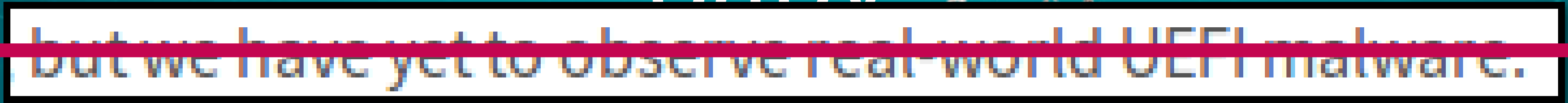
- Software implementation to flash firmware remotely
  - Hacking Team’s UEFI rootkit needed physical access
- We extracted the UEFI rootkit
- Looked at ESET’s UEFI scanner telemetry
- And…

We’re going to Black Hat Baby!

We’re going to Black Hat Baby!


but we have yet to observe real-world UEFI malware.

We’re going to Black Hat Baby!



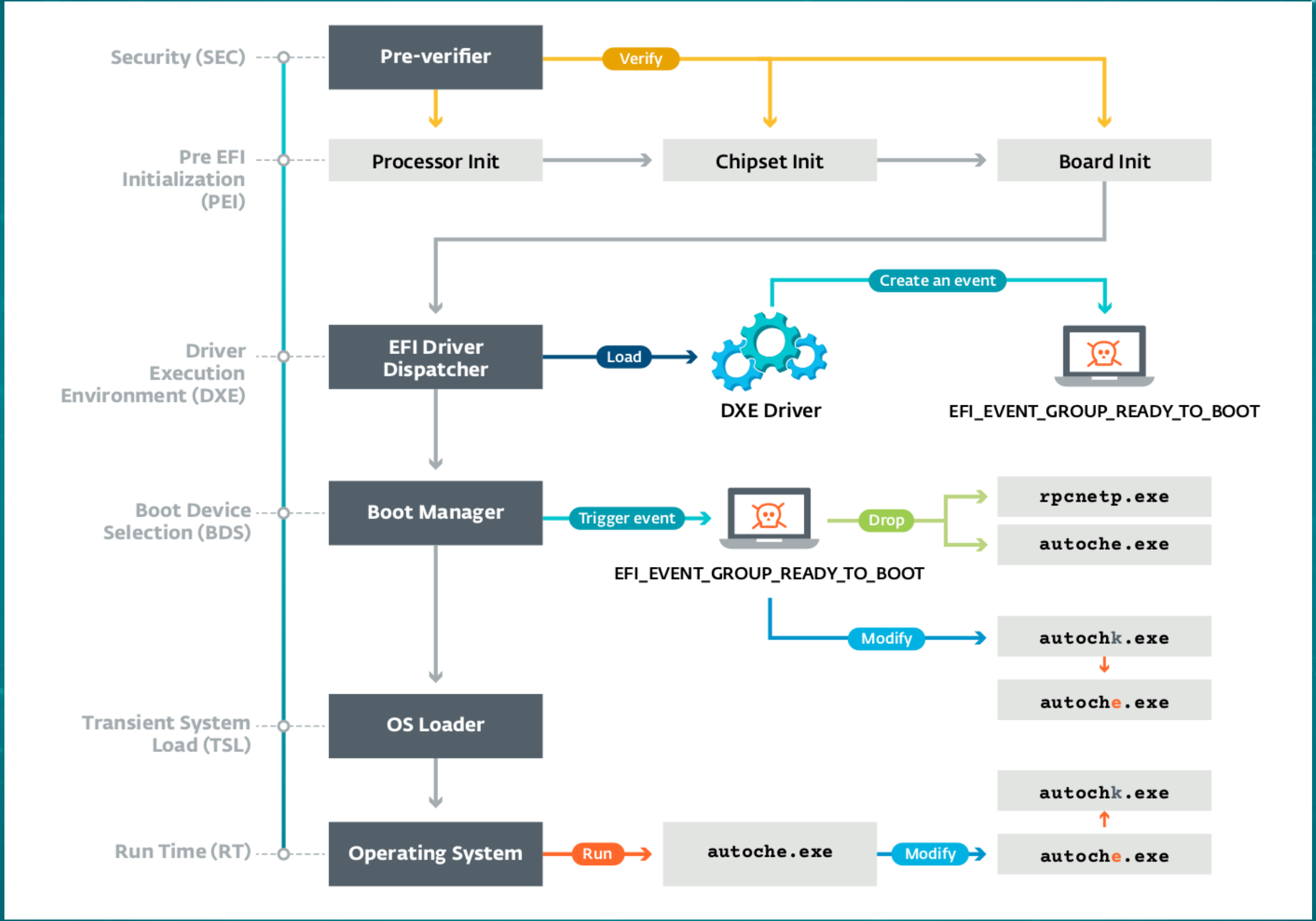
~~but we have yet to observe real-world UEFI malware.~~

UEFI Rootkit

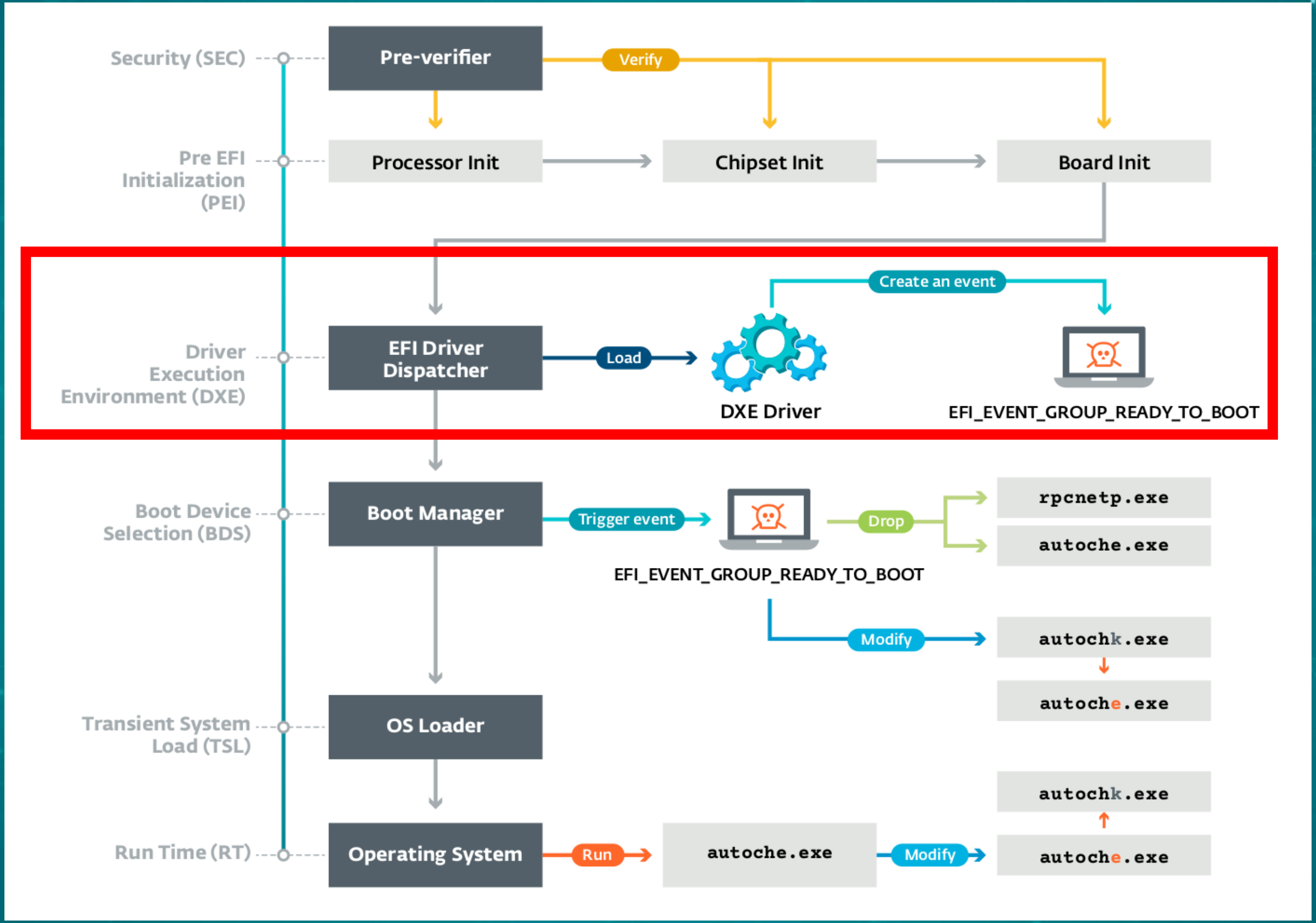
UEFI Rootkit: SecDxe

- DXE Driver loaded by the DXE Dispatcher
- Unsigned
- File GUID
    - 682894B5-6B70-4EBA-9E90-A607E5676297

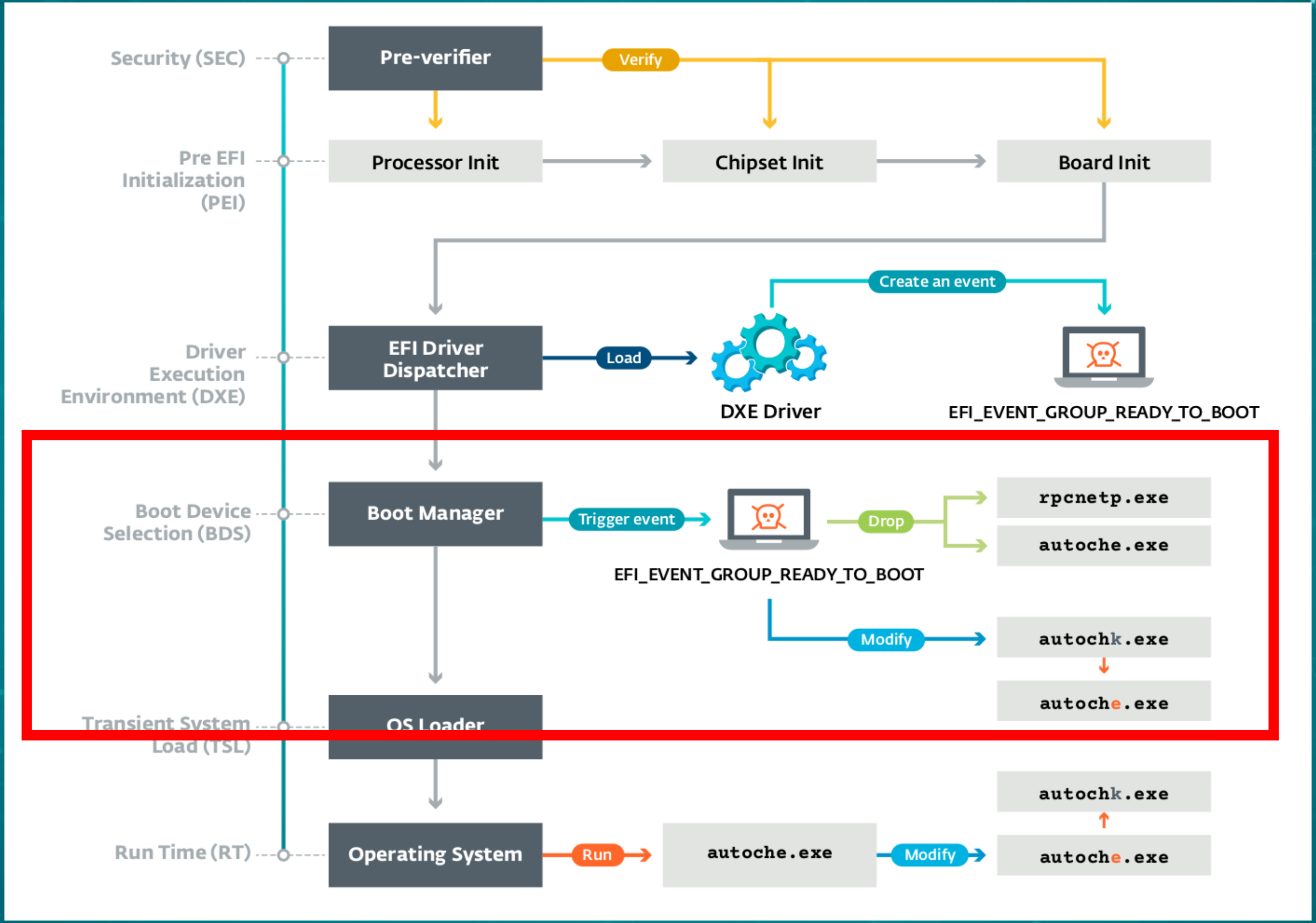
UEFI Rootkit Workflow



UEFI Rootkit Workflow



UEFI Rootkit Workflow



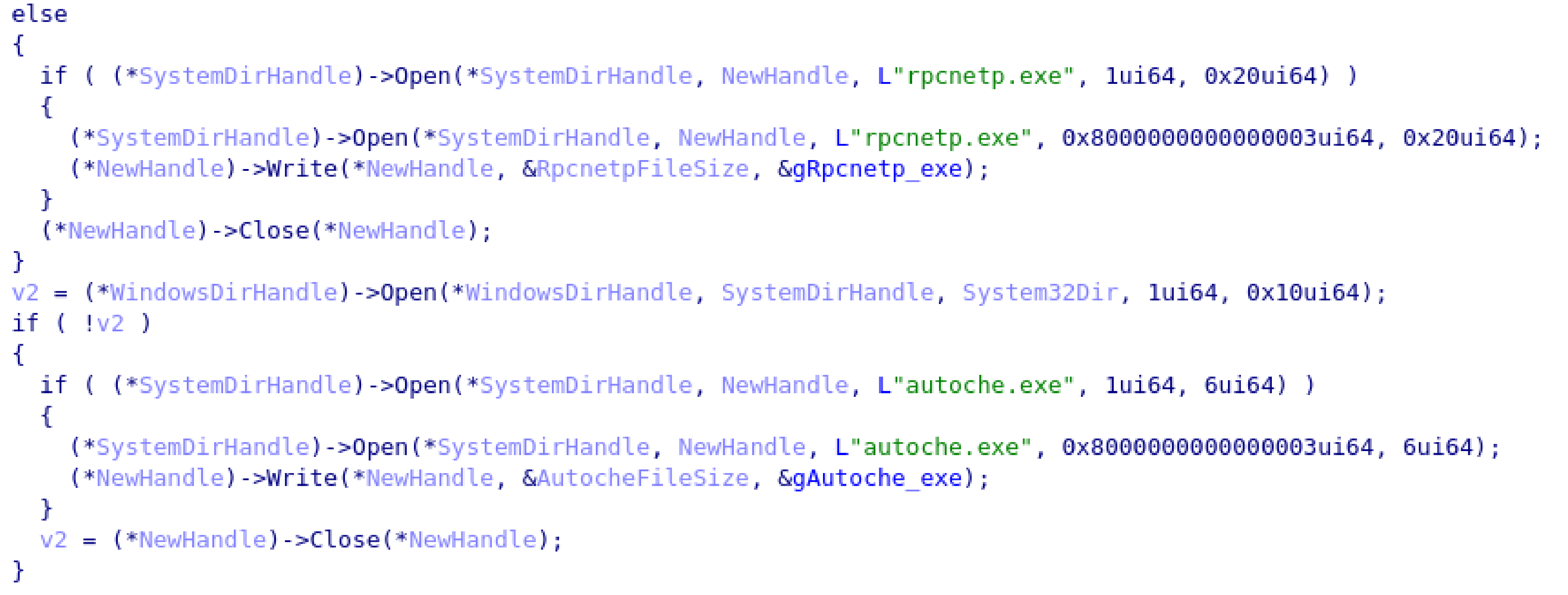
UEFI Rootkit: SecDxe

- Notify function
    - Installs NTFS driver
    - Drops autoche.exe and rpcnetp.exe
    - Patch a value in the Windows Registry

UEFI Rootkit: NTFS driver

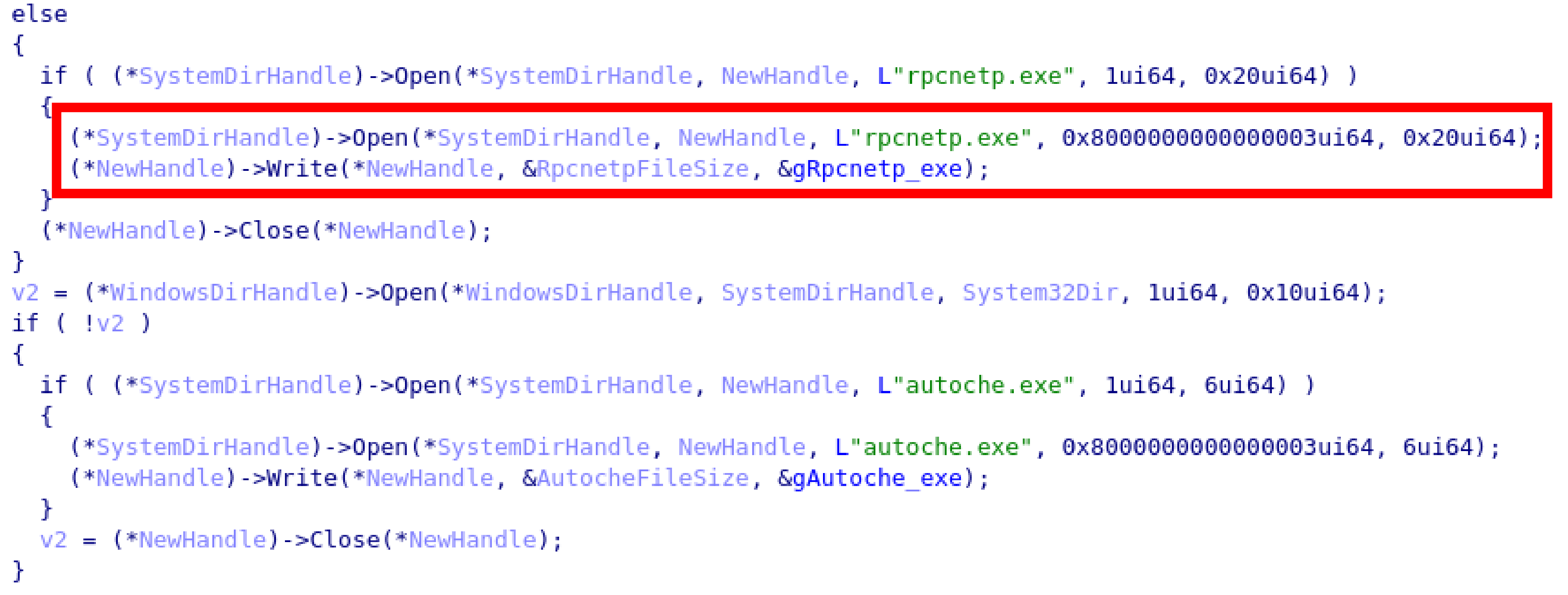
- NTFS driver needed to get file-based access to Windows’ partition
- Hacking Team’s NTFS driver from HT’s leak
    - NtfsDxe project from vector-edk

UEFI Rootkit: Dropping files



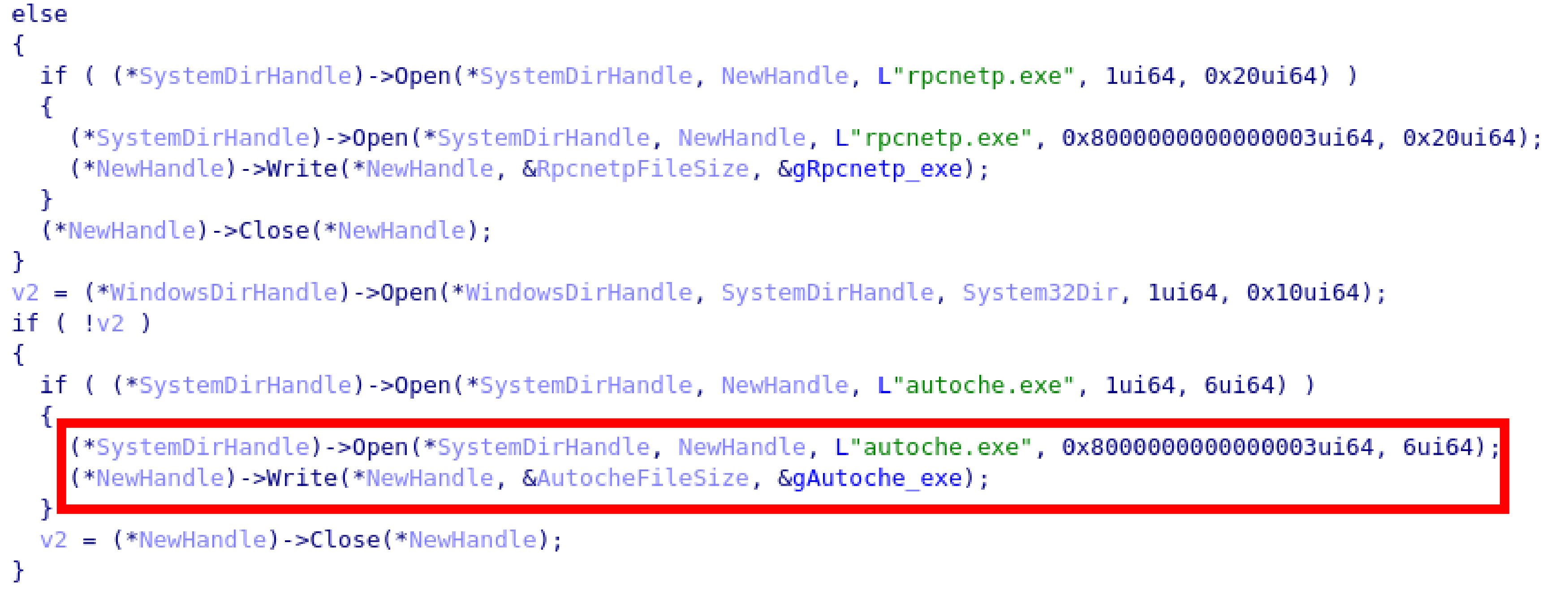
```
else
{
  if ( (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"rpcnetp.exe", 1ui64, 0x20ui64) )
  {
    (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"rpcnetp.exe", 0x8000000000000003ui64, 0x20ui64);
    (*NewHandle)->Write(*NewHandle, &RpcnetpFileSize, &gRpcnetp_exe);
  }
  (*NewHandle)->Close(*NewHandle);
}
v2 = (*WindowsDirHandle)->Open(*WindowsDirHandle, SystemDirHandle, System32Dir, 1ui64, 0x10ui64);
if ( !v2 )
{
  if ( (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"autoche.exe", 1ui64, 6ui64) )
  {
    (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"autoche.exe", 0x8000000000000003ui64, 6ui64);
    (*NewHandle)->Write(*NewHandle, &AutocheFileSize, &gAutoche_exe);
  }
  v2 = (*NewHandle)->Close(*NewHandle);
}
```

UEFI Rootkit: Dropping files



```
else
{
  if ( (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"rpcnetp.exe", 1ui64, 0x20ui64) )
  {
    (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"rpcnetp.exe", 0x8000000000000003ui64, 0x20ui64);
    (*NewHandle)->Write(*NewHandle, &RpcnetpFileSize, &gRpcnetp_exe);
  }
  (*NewHandle)->Close(*NewHandle);
}
v2 = (*WindowsDirHandle)->Open(*WindowsDirHandle, SystemDirHandle, System32Dir, 1ui64, 0x10ui64);
if ( !v2 )
{
  if ( (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"autoche.exe", 1ui64, 6ui64) )
  {
    (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"autoche.exe", 0x8000000000000003ui64, 6ui64);
    (*NewHandle)->Write(*NewHandle, &AutocheFileSize, &gAutoche_exe);
  }
  v2 = (*NewHandle)->Close(*NewHandle);
}
```

UEFI Rootkit: Dropping files



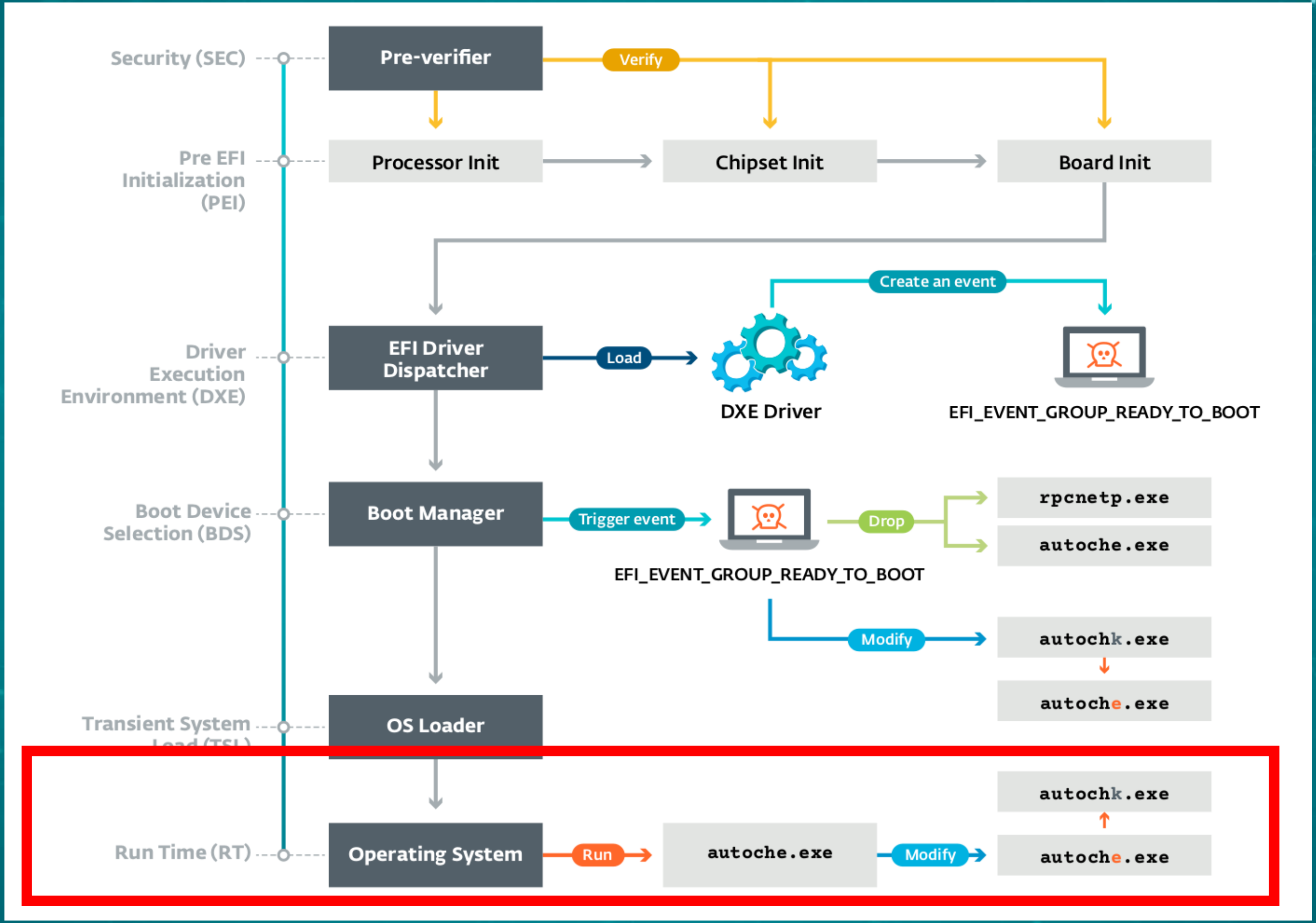
```
else
{
  if ( (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"rpcnetp.exe", 1ui64, 0x20ui64) )
  {
    (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"rpcnetp.exe", 0x8000000000000003ui64, 0x20ui64);
    (*NewHandle)->Write(*NewHandle, &RpcnetpFileSize, &gRpcnetp_exe);
  }
  (*NewHandle)->Close(*NewHandle);
}
v2 = (*WindowsDirHandle)->Open(*WindowsDirHandle, SystemDirHandle, System32Dir, 1ui64, 0x10ui64);
if ( !v2 )
{
  if ( (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"autoche.exe", 1ui64, 6ui64) )
  {
    (*SystemDirHandle)->Open(*SystemDirHandle, NewHandle, L"autoche.exe", 0x8000000000000003ui64, 6ui64);
    (*NewHandle)->Write(*NewHandle, &AutocheFileSize, &gAutoche_exe);
  }
  v2 = (*NewHandle)->Close(*NewHandle);
}
```

UEFI Rootkit: Patching Windows Registry Value

- Modifies Windows Registry via %WINDIR%\System32\config\SYSTEM
- Changes “autocheck autochk *” to “autocheck autoche *”
- HKLM\SYSTEM\CurrentControlSet\Control\

Session Manager\BootExecute

UEFI Rootkit Workflow



Security (SEC)

Pre EFI Initialization (PEI)

Driver Execution Environment (DXE)

Boot Device Selection (BDS)

Transient System Load (TSL)

Run Time (RT)

Pre-verifier

Verify

Processor Init

Chipset Init

Board Init

EFI Driver Dispatcher

Load

DXE Driver

Create an event

EFI_EVENT_GROUP_READY_TO_BOOT

Boot Manager

Trigger event

EFI_EVENT_GROUP_READY_TO_BOOT

Drop

rpcnetp.exe

autoche.exe

Modify

autochk.exe

autoche.exe

OS Loader

Operating System

Run

autoche.exe

Modify

autochk.exe

autoche.exe

Demo

Prevention and Remediation

Prevention

- Enable Secure Boot
- Hardware Root of Trust (ex. Intel BootGuard)
- Keep your UEFI firmware up-to-date
- Make sure you have modern chipsets (PCH)
- Hope that your firmware configures security mechanisms properly :-(
- Firmware security assessments can be done with CHIPSEC

Remediation

- You need to reflash your UEFI firmware
- If it’s not an option for you then…

Remediation

- You need to reflash your UEFI firmware
- If it’s not an option for you then…

Conclusion

ENJOY SAFER TECHNOLOGY™

Thanks!

Questions?

White paper available at welivesecurity.com

@jiboutin

@Freddrickk_