

# Windows Rootkits and Bootkits Guide

A Guide to the Arsenal of Windows Kernel Tricks Employed by Kernel-Mode Malware

Glad to present the second version of the guide. This document is intended to be a comprehensive guide to Windows km malware, with some exceptions and remarks as noted below. The main feature of it is to provide direct document references to the researchers from which the information was taken. The document has the structure of a reference book, which allows you to easily navigate from a specific malware family to its rootkit TTPs (Windows kernel tricks).

The bootkit part of the guide is based on my <u>previous research</u> on bootkit families, but has been revised with a few corrections and the addition of new bootkit families. That blog post will no longer be updated and any possible fixes will be reflected in future versions of this document.

The following web resources made this document possible.

- Malpedia | https://malpedia.caad.fkie.fraunhofer.de/
- MITRE ATT&CK® | https://attack.mitre.org/
- KernelModeInfo forum | https://www.kernelmode.info/forum/
- rootkit\_com site mirror | https://github.com/claudiouzelac/rootkit.com/tree/master/
- Virus Bulletin | https://www.virusbulletin.com/virusbulletin/

<u>Wayback Machine</u> also contributed to this document by reviving old blog posts and malware research papers that are no longer available due to their age.

Also, the following studies are dedicated to the same purpose, i e summarizing information about Ring 0 malware

- An In-Depth Look at Windows Kernel Threats by Trend |
   <a href="https://documents.trendmicro.com/assets/white\_papers/wp-an-in-depth-look-at-windows-kernel-threats.pdf">https://documents.trendmicro.com/assets/white\_papers/wp-an-in-depth-look-at-windows-kernel-threats.pdf</a>
- «Nice Boots!» A Large-Scale Analysis of Bootkits and New Ways to Stop Them | <a href="https://publications.sba-research.org/publications/bootcamp\_dimva\_2015.pdf">https://publications.sba-research.org/publications/bootcamp\_dimva\_2015.pdf</a>
- Bootkit's development overview and trend | http://www.vxjump.net/files/seccon/bktrend.pdf
- Bootkits: Past, Present & Future | <a href="https://www.virusbulletin.com/uploads/pdf/conference/vb2014/VB2014-RodionovMatrosov.pdf">https://www.virusbulletin.com/uploads/pdf/conference/vb2014/VB2014-RodionovMatrosov.pdf</a>
- Positive Technologies | <u>Bootkits</u>: <u>evolution and detection methods</u>

In addition, these resources are very useful for finding information about the aforementioned low-level malware.

A complete list of sources is provided in a separate section below. Some articles are given without web links, because due to their age they are no longer accessible either by their original locations or by Wayback Machine, but I have my own repository with old rootkit articles, and it available for download from my site https://www.artemonsecurity.com/useful.html.

The document includes information only about Windows km (Ring 0) malware/concepts, so the following things are out of scope of it:

- SMM (Ring -2), hypervisor (Ring -1), and hw/fw related rootkit concepts (Blue Pill, SMM rootkit, Equation Group HDD fw implant, VGA rootkit), but with exception for IceLord and Mebromi; [25]
- Ring 3 (user mode) rootkits;
- Low-level crypto ransomware that encrypts MBR, VBR, MFT and ask for ransom at the earliest stage of system boot (MBRLock, Petya, Satana);
- BYOVD drivers and techniques of abusing legitimate Windows drivers;
- Drivers intended to help defeat Driver Signature Enforcement (DSE) or PatchGuard (except EfiGuard);
- Other malicious drivers that don't perform any modifications on the Windows kernel and its environment, for
  example, simple disk wipers, code injectors (with some exceptions), AV/EDR killers, backdoors allowing access to
  km address space, WFP and FS minifilter drivers that are only used to implement a private device stack w/o any
  OS modifications, etc (for example, Flame, Moriya, LuckyMouse, Exforel, ProjectSauron Remsec...); rootkits that
  use those techniques for self-defense are included.
- Windows kernel vulnerabilities

Nevertheless, there are also some exceptions for earlier Windows user mode rootkits performing kernel mode manipulations (DKOM) via \Device\PhysicalMemory. Also, rootkits that employ the concept of Virtual/Hidden/Encrypted File System are included [24].

Interestingly, that modern and well-known chatbots other than ChatGPT didn't prove useful and didn't provide relevant information answering almost any question without technical details. I didn't start by gathering info on the topic using chatbots but decided to check their output later as a kind of double-check. Compared to other chatbots, ChatGPT-40 provided a list of Windows km malware, all of which already were in my list. However, like other chatbots, it could provide inaccurate or even incorrect information. Therefore, asking for proof links and double-checking are not only necessary, but also mandatory.

Necessary reading skills for reading: Windows Internals, malware analysis, reverse engineering at medium+ level.



Contacts: Artem Baranov, an independent security researcher artembaranovex@gmail.com
<a href="https://www.linkedin.com/in/aibaranov/">https://www.linkedin.com/in/aibaranov/</a>

## Contents

Instead of introduction	4
Rootkit techniques	6
The table with rootkit families	
The table with bootkit families	16
Rootkit references	20
Bootkit references	27

«Adversaries may use rootkits to hide the presence of programs, files, network connections, services, drivers, and other system components.» MITRE ATT&CK  $\underline{T1014}$ 

## Instead of introduction

Diving into history A few words instead of an introduction to avoid bothering readers. The most sophisticated instances of the rootkits, with the exception for some notable x64 ones, supported only x86 systems and couldn't conquer Windows x64 with all its security limitations introduced by Microsoft. All discussed malware families (drivers, their modules and disk boot code) are statically detected by AV vendors or most of them (thanks to VirusTotal for its output). But unfortunately, not all security solutions can detect rootkit anomalies in the system and even fewer of them can cure the system of the malware. Of course, it depends on the rootkit strain and the km tricks it uses. To detect sophisticated rootkits and cure the system, some AV vendors have resorted to developing standalone tools (some of them could be called antirootkits) targeting a specific rootkit family. The idea behind this is that the modifications that rootkits make to the system vary from one family to another, making it difficult to create a single tool to remedy each of them. Thus, they have released different tools for different rootkit families (or even versions). Those tools were able to automatically scan the system, find rootkit anomalies and remove malware from the system (usually requiring reboot). In its turn, anti-rootkit is a tool for manually inspecting the system for rootkit anomalies, but it's intended for skilled IT pros with sufficient knowledge about the topic. An anti-rootkit tool is designed to gather all possible information about the km environment and highlight possible anomalies. The reason why AV vendors separate the tools for fighting sophisticated rootkits from their main antimalware solutions is the possible damage that AV anti-rootkit drivers may cause to the system by working with undocumented Windows kernel functions and structures to bypass rootkit hooks. In a nutshell, if your security company has hundreds of millions of customers, you won't be happy to receive feedback about BSoD on every system caused by a simple update of that anti-rootkit driver, sometimes even without the possibility of recovery.

Fighting for Windows x64 There are several methods that rootkit authors can use to make their Ring 0 malware more friendly to Windows x64:

- To sign it with a self-signed certificate that requires rebooting a target system in /TESTSIGNING mode (Necurs).
- To sign the driver with a stolen digital certificate (Derusbi, Purple Fox, Zacinlo, Autochk, DirtyMoe, etc).
- To use a bootkit to neutralize DSE (by patching the appropriate locations of the Windows boot manager, or loading
  the malware driver manually at an early stage of the system boot) and PatchGuard (by making changes to guarded
  memory locations before its initialization).
- BYOVD, i e abusing legitimate 3<sup>rd</sup> party drivers for indirect driver loading (most common way); we exclude here
  the option when 3<sup>rd</sup> party drivers are utilized to perform all the necessary km operations without any rootkit driver
  (Turla, Cahnadr, Demodex, FudModule, etc).
- Exploiting a Windows kernel vulnerability (Dugu 2.0).

The main advantage of bootkits over rootkits is that bootkits can bypass the key Windows kernel security features such as PatchGuard, DSE, and ELAM much easier. The malicious code can patch the necessary km code (Windows kernel and drivers) when it's loaded into memory before its execution and initialization of those features, thus making the changes invisible for them.

Clarification #1 Except for some notable malware families, each row in the table represents an entire malware family, including all its versions and highlights the rootkit techniques inherent in at least one of those versions. Following the assigned web links, you can learn more about specific versions.

Clarification #2 During system startup, bootkits typically patch ntoskrnl or Windows drivers in memory to continue executing code from the required location or to thwart initialization of Windows protection features, but later remove these hooks. Thus, due to this property and the fact that this tactic is common to all bootkits, these actions aren't reflected in the tables.

Clarification #3 The Year column in the tables that refer to the dates of malware appearances may not be entirely accurate. Many of them were taken from public reports, while others from the first submission field of the sample on VirtusTotal if the date could not be found in other sources. Also, this column may refer to the appearance of the malware family, rather than its km module, if it was added in a later version of the malware. The timestamps from the executable headers haven't been used.

The «Additional details» column in the tables provides web links to the corresponding entries in the Malpedia or MITRE ATT&CK catalogs if exist. There you can learn more details about the specific malware families and their TTPs.

## Rootkit techniques

[T3.i] ACPI Driver for NT – acpi.sys

```
[T1] Intercepting system services to control calls of basic Windows kernel functions (system services [3])
        [T1.a] Modifying SSDT (KiServiceTable) [1] [2]
                          MSR_SYSENTER (IA32_SYSENTER_EIP, CS) for sysenter on x86 [1] [2] [16]
        [T1.b]
                          KTHREAD.ServiceTable [4]
        [T1.c]
                          IDT[0x2E] system service interrupt [1] [2] [4]
        [T1.d]
        [T1.e] Inline patching of KiSystemService or KiFastCallEntry (x86) [1] [2]
        [T1.f]
                                  Nt* functions from SSDT
[T2] Direct Kernel Object Manipulation (DKOM) to manipulate Windows kernel structures [20]
        [T2.a] Unlinking drivers from PsLoadedModulesList (LDR DATA TABLE ENTRY) [1] [2]
                         processes from PsActiveProcessHead [1][2]
        [T2.b]
                         threads from KiWaitInListHead, KiWaitOutListHead, KiDispatcherReadyListHead [1] [2] [6] [7]
        [T2.c]*
        [T2.d] Modifying access token [1][2]
        [T2.e] Removing objects from Ob object directory
                          driver objects from the list of driver objects
        [T2.f]
                          device objects from the list of device objects
        [T2.g]
        [T2.h] Forging ETHREAD fields [1] [2]
                        EPROCESS fields [1] [2]
        [T2.i]
        [T2.j] Erasing items in PspCidTable [2]
                        handles in the process handle table
        [T2.k]
        [T2.1] Intercepting object type dispatch functions (procedures) [13]
        [T2.m] Forging DRIVER_OBJECT fields
        [T2.n] Hijacking or spoofing driver object [13]
        [T2.0] Hijacking or spoofing device object [13]
[T3] Inline patching kernel mode code (run-time patching, inline hooking, splicing) [1] [2]
        [T3.a] Ntoskrnl - * Nt*, IofCallDriver, IofCompleteRequest, IoCreateFile, etc. [7]
        [T3.b] FSD – Ntfs.sys, Fastfat.sys and attached filter, minifilter drivers (Filter Manager) [1] [2] [5]
        [T3.c] TCP/IP, NDIS - Tcpip.sys, Ndis.sys and its related internal structures [1] [2] [4]
        [T3.d] IP Filter Driver - Ipfilterdriver.sys
        [T3.e] SCSI Class System DII classpnp.sys
        [T3.f] Disk port drivers - atapi.sys, ataport.sys, storport.sys, scsiport.sys
        [T3.g] NULL Driver - Null.sys
```

[T3.j] MS QoS Packet Scheduler - psched.sys

[T4] Intercepting driver object major functions or DriverUnload

[T4.a] FSD – Fastfat.sys, Ntfs.sys to hide files [1] [2] [5]

[T4.b] TDI Tcpip.sys, Ndis.sys, also NDIS\_OPEN\_BLOCK and NDIS\_PROTOCOL\_BLOCK handlers to manipulate network communications [2] [17] [18]

[T4.c] Disk port/miniport drivers - atapi.sys, ataport.sys, storport.sys, scsiport.sys to hide disk sectors [5]

[T4.d] Fast I/O Dispatch Routine (FastIoDeviceControl) *AfdFastIoDeviceControl* of Afd.sys to manipulate network traffic

[T4.e] Network Store Interface (NSI) driver nsiproxy.sys to hide TCP ports [22]

[T4.f] NULL Driver - Null.sys to hide rootkit activity

[T4.g] LiveKd debugger driver

[T4.h] FS Filter Manager fltmgr.sys [4]

[T4.i] Disk driver disk.sys

[T4.j] SCSI CD-ROM driver cdrom.sys

[T5] Intercepting IDT/ISR (excluding the case with hooking int 13h, which is used by almost all bootkits) [1] [2] [5]

[T6] Setting up itself as a filter driver for (attaching to the corresponding device stack for self-defense) [1]

[T6.a] File System Driver (FSD), legacy or minifilter (fltmgr) [2]

[T6.b] Volume Manager (volmgr.sys, volmgrx.sys) [4]

[T6.c] TCP/IP stack, NDIS (tcpip.sys, ndis.sys) - \Device\Tcp, \Device\Ip, \Device\RawIp, \Device\Ndis, \Device\Udp  $^{[4][17]}$ 

[T6.d] NSI driver nsiproxy.sys

[T7] Using Windows kernel callbacks (in case of blocking access to system resources and self-protection) [4] [9]

[T7.a] CmRegistry, LoadImageNotify, ObRegisterCallbacks

[T8] Using and hiding NTFS Alternate Data Streams (ADS) [4]

[T9] Keylogger (attaching to \Device\KeyboardClass0) [1] [2]

[T10] Windows IP Filtering [1]

[T11] Disabling Windows kernel callbacks (create process, create thread, load image, registry ops, kernel object, object manager) to blind AV and EDR products [9]

[T11.a] LoadImageNotify, CreateThreadNotify, CreateProcessNotify, CmRegistry, ObRegisterCallback

[T12] Other tricks

[T12.a] ObMakeTemporaryObject to remove the driver object's name from the \Driver\ object manager directory

[T12.b] Disabling WFP callout drivers (via netio!gwfpGlobal) [10]

[T12.c] Disabling Event Tracing (ETW) (via nt!EtwpHostSiloState) [11]

[T12.d] Disabling System Loggers (via nt!EtwpActiveSystemLoggers)

- [T12.e] Disabling FS minifilter drivers (via unlinking the appropriate structures) [12]
- [T12.f] Disabling Image Verification Callbacks
- [T12.g] Hidden (Encrypted) File System (VFS) [5] [24]
- [T12.h] Hiding services by unlinking the corresponding SERVICE\_RECORD structure in the context of the Services.exe process [14]
- [T12.i] Preventing writing kernel memory dumps by registering its callback with KeRegisterBugCheckReasonCallback
- [T12.j] Replacing HHIVE.GetCellRoutine pointer to get control over system registry operations [19]
- [12.k] Disabling FS minifilter drivers via FltUnregisterFilter
- [12.I] Implements its own private TCP/IP stack
- [12.m] Disables or bypasses PatchGuard
- [12.n] Files signed by a stolen cert (x86) /for x64 refer to [T14.c]/
- [T13] The subject of bootkit infection (to survive potential OS reinstallation)
  - [T13.a] Master Boot Record (MBR)
  - [T13.b] Volume Boot Record (VBR)
  - [T13.c] UEFI, the EFI System Partition (ESP)
  - [T13.d] UEFI, SPI flash memory
  - [T13.e] Legacy BIOS flash (by inserting a malicious ISA module)
- [T14] Defeating DSE (for x64 only)
  - [T14.a] BYOVD to /covertly/ load the driver or patch the Windows kernel, DSE variables (g\_CiEnabled)
  - [T14.b] Signing the driver by a self-signed cert + reboot with /TESTSIGNING bootloader flag
  - [T14.c] The driver has a valid digital signature (signed by a stolen cert)
  - [T14.d] Exploitation of a Windows kernel vulnerability
  - [T14.e] The bootkit loads the driver manually
  - [T14.f] The bootkit patches the Windows kernel

**Note** that the references to the techniques in the tables below (the Feature column) are <u>clickable</u>. You can easily navigate to a specific technique to learn more about it, and that's what these tables are. References like <sup>[X]</sup> are also clickable and lead to the corresponding information source.

<sup>\*</sup>In fact, this technique is very rare, because if a thread is removed from one of those lists, it also becomes hidden to the thread scheduler and won't get any CPU time. The only rootkit using it is 90210's phide2 that runs its own thread scheduler after copying the necessary ntoskrnl functions (utilizing pullout engine) to a separate buffer to serve the hidden thread. In spite of this, it can be useful for anti-rootkits (for example, klister) that go through those lists, exposing hidden processes (unlinked from *PsLoadedModulesList*).

# The table with rootkit families

#### Has known authors

#### Utilize \Device\PhysicalMemory

Belong to the same malware family or TA (attribution according to the public information): **FU**, **Phide**, **Rustock**, **Tdss** (**TDL**, **Alureon**), **BlackEnergy**, **Attributed to the NSA** 

Name	Year	Platform	Features	Author/Detection	Additional details
NTRootkit <sup>[Family]</sup> Originally concept	1999	x86	[T1.a] [T1.d] [T9]	Greg Hoglund	Do not confuse this original rootkit concept and x86 SSDT hooking rootkits based on this widely known technique with the NtRootkit malware family of some AV companies that use it as a generic detection name for the rootkits that they have not classified. [26]
FU [Family]	2004	x86	[T2.a] [T2.d]	fuzen_op  WinNT/FURootkit.A	[27] [28]
Phide [Family] Concept	2004	x86	[T2.b] [T2.h]	90210	Kernel memory access via \Device\PhysicalMemory [29]
Phide2 [Family] Concept	2004	x86	[T2.b] [T2.c]	90210	Loads another copy of ntoskrnl and runs its own thread scheduler [6] [30]
FUTo [Family]	2005	x86	[T2.a] [T2.d] [T2.j] [T2.k]	Peter Silberman C.H.A.O.S Win32.Fuzen.a	[31]
Myfip [Family]	2005	x86	[T2.a]	W32/Myfip	Kernel memory access via \Device\PhysicalMemory [32]
Fanbot <sup>[Family]</sup>	2005	x86	[T2.a]	W32/Fanbot	Kernel memory access via \Device\PhysicalMemory There are many more rootkits utilizing this technique [32]
Shadow Walker [Family] Concept	2005	x86	<u>[T5]</u>	Greg Hoglund James Butler	Intercepts access to Translation Lookaside Buffer (TLB) [33] [34] [35]
Sony XCP rootkit [Family]	2005	x86	[T1.a]	SecurityRisk.First4DRM	[36] [37] [38] [39]
Apropos [Family]	2005	x86	<u>[T3.a]</u>	?	[40]

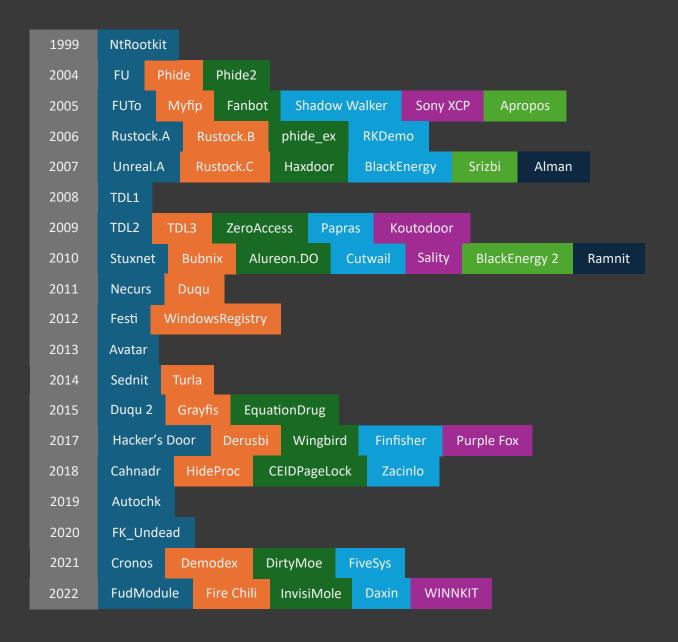
Rustock.A [Family]	2006	x86	[T1.b] [T1.c] [T2.a] [T3] [T4.a] [T8]	Backdoor.Rustock.A	Polymorphic packer <sup>[41]</sup> <u>Malpedia</u>
Rustock.B [Family]	2006	x86	[T1.b] [T1.c] [T2.a] [T3] [T4.a] [T4.b] [T8]	Backdoor.Rustock.B	Polymorphic packer <sup>[42]</sup> <sup>[43]</sup> <sup>[44]</sup> <u>Malpedia</u>
phide_ex <sup>[Family]</sup> Concept	2006	x86	[T1] [T2] [T4.a]	PE386/MS-REM WinNT/Rustock.C	[45] [46]
<b>RKDemo</b> Concept	2006	x86	?	MP-ART EP_X0FF	
Unreal.A <sup>[Family]</sup> Concept	2007	x86	[T2.a] [T2.e] [T2.f] [T2.g] [T2.h] [T6.a] [T8]	MP-ART EP_X0FF DSSDetection Hacktool.Unreal.A	[47]
Rustock.C [Family] originally named by mistake, doesn't belong to the Rustock family	2007	x86	[T1.e] [T2.l] [T3.b] [T3.c]	PE386/MS-REM WinNT/Rustock.D Win32.Ntldrbot Win32.Rustock.a	Heavy code obfuscation+ multiple encryption layers + hardware locking + anti-debugging tricks + anti-patching tricks [48] [49] [50] [51]  Malpedia
Haxdoor [Family]	2007	x86	[T1.a] [T2.a]	WinNT/Haxdoor	[52] [53] [54] [55] [56] [57]
BlackEnergy [Family]	2007	x86	[T1.a]	?	[57] [58] [59] <u>S0089</u> <u>Malpedia</u>
Srizbi <sup>[Family]</sup>	2007	x86	[T1.f] [T4.a] [T4.b] [12.l]	Trojan.Srizbi	The first itw malware that operates fully from Ring 0 + runs its own copy of ndis.sys
Alman <sup>[Family]</sup>	2007	x86	[T1.a]	Win32/Alman Virus:Win32/Almanahe Virus.Win32.Alman	[61] [62]
TDL1 Tdss Tidserv Alureon <sup>[Family]</sup>	2008	x86	[T1.f] [T2.a] [T6.b] [T6.c]	Rootkit.Win32.Clbd.a	[63] Malpedia

TDL2 Tdss Tidserv Alureon <sup>[Family]</sup>	2009	x86	[T1.f] [T2.a] [T3.a] [T6.b] [T6.c]	Win32/Alureon.BK Backdoor.Tidserv	[63] [64] [65] <u>Malpedia</u>
TDL3 Tdss Tidserv Alureon <sup>[Family]</sup>	2009	x86	[T4.c] early versions [T2.n] [T2.o] [T12.g]	Trojan:Win32/Alureon.CT Backdoor.Tidserv Win32/Olmarik.SC	Infects disk port drivers (atapi.sys, iastor.sys) [63] [66] [67] [68] [69] <u>Malpedia</u>
ZeroAccess ZAccess Sirefef [Family] shares similarities with TDL3	2009	x86	[T2.m] [T2.n] [T2.o] [T12.g]	Trojan:Win32/Sirefef Virus.Win32.ZAccess BackDoor.Maxplus	Early versions replaced system drivers + infects system drivers in TDL3 manner + creates \Device\max++ TDL3 removing routine + creates hidden encrypted volume [70] [71] [72] [73] [74] S0027 Malpedia
Papras Gozi Ursnif <sup>[Family]</sup>	2009	x86	[T1.a]	Win32/Ursnif.CU PSW.Papras.AO	[57] [75] <u>S0386</u> <u>Malpedia</u>
Koutodoor [Family]	2009	x86	[T2.l]	Win32/Koutodoor	Polymorphic code + After each reboot the rootkit moves the driver to another file <sup>[76]</sup>
Stuxnet [Family] attributed to the NSA	2010	x86	[T6.a]	Trojan:WinNT/Stuxnet Rootkit.Win32.Stuxnet	Signed by a stolen cert [77] [78]  Solution Signed by a stolen cert [77] [78]  Malpedia
Bubnix <sup>[Family]</sup>	2010	x86	[T2.l] [T4.a]	Win32/Bubnix WinNT/Bubnix	[79]
Alureon.DO [Family] TDL2 clone	2010	x86	[T1.f] [T2.a] [T3]	Win32/Alureon.DO Win32.TDSS.bwkw Win32/Olmarik.TL	[80] Malpedia
Cutwail [Family]	2010	x86	[T1.a] [T4.a]	WinNT/Cutwail	[81] [82] [83] <u>Malpedia</u>
Sality [Family]	2010	x86	[T10]	Trojan:WinNT/Sality Virus.Win32.Sality	Blocks network packets belonging to AV vendors <sup>[84]</sup> <u>Malpedia</u>
BlackEnergy v2	2010	x86	[T1.c] [T3.a]	Backdoor:WinNT/Phdet.A RTKT_RUSTOCK.SMB BackDoor.BlackEnergy	[ <u>85]</u> <u>S0089</u> <u>Malpedia</u>
Ramnit <sup>[Family]</sup>	2010	x86	[T1.a]	Worm:Win32/Ramnit.A Win32/Ramnit.A	Acts as AV killer + Restores SSDT pointers to disable AV hooks <sup>[86] [87] [88]</sup> <u>Malpedia</u>
Necurs [Family]	2011	x86 x64	[T1.a]  x86  [T6.a]  [T7.a]  [T14.b]  x64	Trojan:WinNT/Necurs Rootkit.Win32.Necurs	Creates \Device\NtSecureSys Starts with the highest priority as Boot Bus Extender [89] [90] [91] [92] [93] [94] Malpedia

			Tricky code		Unique file names [95] [96]
Duqu [Family]	2011	x86	injection	Trojan:WinNT/Dugu	S0038
attributed to the NSA			[12.n]	Trojan.Win32.Duqu	<u>Malpedia</u>
			[T1.a]	Win32/Rootki.Festi	
Festi [Family]	2012	x86	[T6.a]	Rootkit.Win32.Tent	<u>[5] [97] [98]</u>
MG: down Down Down II	2012	x86	[T3.c]	Cr4sh	Stored in the Windows registry +
WindowsRegistryRootkit [Family]	2012	XOU	[13.0]	Trojan:Win32/Leivion.I	Infects Windows drivers in
Concept				Trojan.Win32.Diple.fzxp	memory [99] [100]
					Capable of being loaded as
			[T2.m]	De estis Min 22 Anns en	fileless rootkit + Infects other system drivers +
Avatar [Family]	2013	x86	[T4.c]	Rootkit.Win32.Avatar BackDoor.Avatar	VM detection via
7104441	2013	λου	[T12.g]	Win32/Rootkit.Avatar	<i>MmMaploSpace</i> + Has its own SDK +
					Relocates disk port driver +
					Infects disk port driver in
		x86	[T1 a]	Dootlet Win22 Agent elele	memory [101] [102] [103]
Sednit [Family]	2014	x64	<u>[T1.a]</u> [T6.a]	Rootkit.Win32.Agent.ekik Win32/Rootkit.Agent	[104] [105] G0007
SVR-aligned TA			112121		<u>Malpedia</u>
					Creates its own (clean) copy of
					KiServiceTable + Bypasses PatchGuard by hooking
considered one of the most sophisticated rotkits			[T1.f]		KeBugCheckEx +
Turla (Snake)		x86	[T3.a]	Rootkit.Win64.Turla	Bypasses DSE by abusing Oracle VirtualBox driver VBoxDrv.sys +
Uroburos [Family]	2014	x64	[T4.b]	Backdoor:WinNT/Turla	Relocates itself into System
linked to Agent.BTZ state-sponsored			[T12.g]	Win64/Turla	process + Uses the interrupt 0xC3 for the
Russia-aligned			[12.m]	Rootkit:Uroburos	hooking engine +
			[T14.a]		Incorporates Udsi86 disasm <sup>[106]</sup> [107] [108] [109] [110] [111]
					<u>G0010</u>
		x86	[TG c]		Malpedia
Dugu 2.0 [Family]	2015	x86 x64	[ <u>T6.c]</u> [T14.d]		termport.sys <sup>[112]</sup> <u>S0038</u>
attributed to the NSA			<u> </u>		<u>Malpedia</u>
		x86	[T2.m]		
Grayfish [Family] Equation Group TA	2015	x64	[T12.a]	Trojan.Win32.GrayFish	\Driver\msvss [113] [114]
Equation Group IA					
EquationDrug		x86		Trojan:WinNT/Eqtonex.A	mstcp32.sys
[Family]	2015	x64	[T6.c]	Win32/Equdrug.A	\Device\Mstcp32 \frac{[115] [116]}{ \land \text{Device} \text{Mstcp32} \frac{115] [116]}{ Total Properties of the content of the conten
state-sponsored NSA-aligned				Trojan.Equdrug	<u>Malpedia</u>
Hooken's Been	2017	06	[T2 4]]	Win64/Hackdoor.A	kifesEn.sys + Instead of intercepting
Hacker's Door  [Family]	2017	x86 x64	[T3.d]	Backdoor:Win64/Hackdoor.A	functions and callbacks of Ndis,
					Tcpip drivers, patches
					IpFltDrv.sys <sup>[117]</sup>

			[T1.a] x86		
<b>Derusbi</b> <sup>[Family]</sup> state-sponsored	2017	x86 x64	[T4.a] [T4.b] [T4.e] [T14.c]	?	Anti-debug tricks <sup>[118] [119] [120]</sup> S0021 Malpedia
Wingbird [Family] state-sponsored	2017	x86	[T1.a] to remove AVers hooks	Backdoor:Win32/Wingbird	The code is heavy obfuscated + Self-modifying (mutable) code + Removes AVers SSDT hooks + Covert code injection [121] [122] S0176
Finfisher [Family] state-sponsored	2017	x86	Tricky code injection	Backdoor:Win32/Finfish Backdoor.Win32.FinFish Win32/FinSpy	The code is heavy obfuscated + Self-modifying (mutable) code [122] [123] [124] [125] S0182 Malpedia
Purple Fox [Family]	2017	x64	[T12.k] [T14.c]	Trojan.Win64.PURPLEFOX.YACAM PUA:Win32/Creprote	<u>[126]</u>
Cahnadr [Family] state-sponsored Slingshot APT	2018	x86 x64	[T1.c] x86 [T3.c] [T4.f] x64 [T4.g] [T12.i] [T14.a]	Backdoor.Win32.Slingshot Win64/Slingshot Trojan.Slingshot	Loaded covertly through  MSR_LSTAR handler syscall [16] [17]  +  Checks Ntoskrnl and Win32k for hooks +  Receives commands through its  CmRegisterCallback (x86) + Anti-debug tricks [127]
HideProc [Family]	2018	x86	[T2.b]	Win32/HideProc	[128] [129]
CEIDPageLock [Family]	2018	x86	[T4.d] [T12.n]	Backdoor.Graybird Trojan.NtRootKit.19661 PUA:Win32/Kuping	Packed with VMProtect [130]
Zacinlo [Family]	2018	x86 x64	[T6.a] [T7.a] [T14.c]	Troj/Zacinlo-A	radardt.sys \Device\DrvProtect <sup>[131]</sup> <u>Malpedia</u>
Autochk [Family] attributed to Emissary Panda	2019	x64	[T4.e] [T4.h] [T14.c]	Rootkit.Win64.ZXShell.e Trojan:Win32/Zxshell Win64/ZxShell.AH	Redirects requests to hidden files to legitimate ones [132]
FK_Undead <sup>[Family]</sup>	2020	x64	[T7.a]	Rootkit.Win32.Agent.elyj	Packed with VMProtect [133] [134]
Cronos [Family]	2021	x64	[T2.a] [T2.b] [T2.d]	Win64/Rootkit.Cronos Win64:CronosRootkit-A [Rtk]	[135]
Demodex [Family]	2021	x86 x64	[T4.e] [T3] <sub>Pci.sys</sub> [T6.a]		Loaded via abusing Cheat Engine driver dbk64.sys <sup>[136]</sup>

			[T7.a] [T12.h] [T14.a]		
DirtyMoe <sup>[Family]</sup>	2021	x86 x64	[T2.a] [T4.a] [T6.a] [T12.j] [T14.c]	Win32:DirtyMoe-C Win64/Rootkit.Agent.T Rootkit.Win64.Agent.bfo	The driver is written on disk every time the system starts and is deleted once it's loaded [137]  Malpedia
FiveSys [Family]	2021	x64	[T7.a] [T14.c]	Trojan.Win64.PACSYS.A Rootkit.Agent.AJIQ	[138] <u>S0618</u>
FudModule [Family] attributed to Lazarus	2022	x86 x64	[T2.h] [T3.b] [T11.a] [T12.b] [T12.c] [T12.d] [T12.e] [T12.f] [T14.a]	Acts from um abusing 3rd party drivers that enable it to operate on Ring 0 memory utilizing physical<->virtual address translation	Operates via BYOVD (ene.sys, appid.sys) + Uses physical memory to access km address space + Modifies KTHREAD.PreviousMode to allow um code accessing km address space via  ZwWriteVirtualMemory Removes AVers callbacks + Disables ETW + Unlinks FS minifilter drivers [139] [140] [141] [142] [143]  Malpedia
Fire Chili <sup>[Family]</sup> attributed to Deep Panda	2022	x86 x64	[T2.b] [T6.a] [T6.c] [T7.a] [T14.c]	Rootkit.Win64.Agent.bjm Hacktool.Rootkit	Drivers are signed by stolen certs + Loaded via BYOVD  [144]  Malpedia
InvisiMole [Family]	2022	x64	[T6.a] [T11.a] [T12.b] [T12.h] [T14.a]	Win64/InvisiMole	<u>[145]</u> <u>S0260</u> <u>Malpedia</u>
Daxin <sup>[Family]</sup>	2022	x86 x64	[T4.b] [12.l]	Trojan.Win64.Agentb.bwb Backdoor.Daxin	<u>[146]</u> <u>Malpedia</u>
WINNKIT [Family] state-sponsored, Winnti TA	2022	x86 x64	[T2.a] [T2.f] [T2.g] [T4.b] [T4.f] [12.l] [T14.c]	Backdoor.Multi.Winnti Win64:Winnti-L [Trj]	[147] [148] <u>Malpedia</u>



# The table with bootkit families

Name	Year	Platform	Features	Author/Detection	Additional details
eEye BootRoot [Family] Concept	2005	x86	[T4.i] [T13.a]	Derek Soeder Ryan Permeh  Trojan:DOS/Sinowal.A Backdoor.Win32.Sinowal	[149] [150]
IceLord <sup>[Family]</sup> Concept	2007	x86	[T13.e]		[151] [152] [153]
Vboot Kit [Family] Concept	2007	x86 x64 <sub>2009</sub>	[T13.a]	Nitin Kumar Vipin Kumar  Rootkit.Win32.VBoot.a	[154] [155] [156]
Mebroot Sinowal Maosboot <sup>[Family]</sup>	2007	x86	[T2.h] [T3.e] [T4.b] [T4.i] [T4.j] [12.l] [T13.a]	Backdoor.Win32.Sinowal PWS:Win32/Sinowal Trojan.Mebroot Trojan:DOS/Sinowal.O	Has a watchdog that reinfects disk and restores runtime kernel mode hooks +  Has code similar to Srizbi that loads the necessary code from clean copy of ndis.sys using 90210' «pullout engine» + reuses other code from phide2 [157] [158] [159] [160] [161] [162] [163] [164] Malpedia
Stoned [Family] Concept	2009	x86 x64	[T13.a]	Peter Kleissner	Utilizes Sinowal kernel driver [165] [166] [167]
Whistler [Family]	2010	x86 x64 ?	[T13.a]	Rootkit.MBR.Whistler.B Rootkit.Whistler Rootkit.Boot.Wistler.a	[168] [169] [170] [171]
Mebratix GhostShadow [Family]	2010	x86	[T13.a]	Trojan.Mebratix BackDoor.Nedoboot Backdoor.Win32.Phanta Trojan:Win32/Ghodow	Shares code similarities with Sinowal <sup>[172] [173]</sup>
TDL4 Olmarik <sup>[Family]</sup>	2010	x86 x64	[T2.0] [T12.g] [T13.a] [T14.e]	Rootkit.Win64.TDSS.a Trojan:DOS/Alureon.A Win64/Olmarik.AV Backdoor.Tidserv	Anti-debug tricks + re-infects MBR if it was changed <sup>[174] [175] [176] [177]</sup> <sup>[178] [179] [180] [181] [182]</sup> <u>Malpedia</u>
MaxSS Olmasco <sup>[Family]</sup>	2011	x86 x64	[T12.g] [T13.a] [T13.b] [T14.e]	Trojan:WinNT/Alureon.S Trojan:Win32/Alureon.FE Trojan.Win32.TDSS.bveb	[183] [184] [185]
PiXiEServ [Family] Concept	2011	x86	[T13.a]	j00ru Gynvael Coldwind	[186] [187]

				Trojan:WinNT/Bioskit.A	File name bios.sys +
Mebromi	2011	x86	[T4.i]	Trojan.Mebromi	Infects both BIOS flash
Bioskit [Family]			[T13.a]	Rootkit.Win32.Mybios.a	and MBR [188] [189]
			[T13.e]	,,,,,,	Malpedia
			<u> </u>	Trojan:W32/Smitnyl.A	Bypasses Windows File
Smitnyl [Family]	2011	x86	[T13.a]	Trojan.Win32.Smitnyl	Protection (WFP) using
					MBR file infector [190] [191]
					[192]
(= 11.3				Trojan:Win32/Popureb.E	
Popureb [Family]	2011	x86	[T13.a]	VirTool:WinNT/Popureb.A	[193] [194] [195] [196]
				Win32/Ghodow.NAG	
			[T4 c]	Trojan.Alworo	Hooks int 08h to avoid
Rovnix		x86	[T4.c] [T5]	Rootkit.Win64.Cidox	detection +
Cidox [Family]	2011	x64	T12.g]	Win32/Rovnix.D	int 01h [198] [199] [200] [201] [202]
Clubx	2011	λ0-1	[T12.l]	Win32/Rovnix.G	[203] [204] [205] [206] [207]
			[T13.b]	,	<u>Malpedia</u>
			T14.e]		
		x86	[T13.b]	Win32/Carberp.A	[208] [209] [210] [211]
Carberp [Family]	2011	x64	[T14.e]	Trojan.Carberp	[212]
					<u>Malpedia</u>
Fisp				Rookit.Win32.Fisp.a	
Fispboot [Family]	2011	x86	[T13.a]	Trojan.Fispboot	[213]
					Bypasses PatchGuard by
Evil Core [Family]	2011	x86	[T13.a]	Wolfgang Ettlinger	patching the guarded
Concept		x64	[T14.f]	Stefan Viehböck	kernel data before its
					initialization
					[214]
			[T3.a]		Hooks <i>MmMapIoSpace</i> + Hooks int 01h +
Xpaj <sup>[Family]</sup>	2012	x86	[T5.a] [T5]	Rootkit.MBR.Xpaj.A	Bypasses PatchGuard by
7.60)	2012	x64	[12.m]	Virus:Win32/Xpaj.gen	patching ntoskrnl code
			[T13.a]	, , , , , ,	before its initialization +
					Malware acts as a file
					infector [215] [216] [217]
				Backdoor.Win32.Yurn.a	File name mssounddx.sys
Yurn [Family]	2012	x86	[T13.a]	Trojan:Win32/Yurn.A	[218] [219]
				Win32/Belesak.A	
			[T2.m]		Bypasses ELAM by loading
			[T3.f]		before its initialization +
		00	[T3.g]	W:-22/C	Implements
Gapz <sup>[Family]</sup>	2012	x86 x64	[T4.c]	Win32/Gapz Trojan:Win32/Screud.A	communication with its um counterpart via a hook
Зарг	2012	X04	[T4.f] [T12.g]	Trojan: Wins2/Screud.A	that sets in Null.sys +
			[T12.g] [T12.l]	Hojan.Gapz	uses a disassembler
			[T13.a]		named «Hacker
			[T13.b]		Disassembler Engine» to
					set code hooks properly
					[220] [221] [222] [223] [224] [225]
					Disables km research tools
					and AV services +
					Communicates with hard
Guntior				Trojan.Guntor.2	drive directly via ATA PIO +

Wapomi Pincav <sup>[Family]</sup>	2012	x86	[T13.a]	Win32/Wapomi Win32/Jadtre	Doesn't protect itself on disk + Leverages PnP Manager for covert loading + Borrows the sources of the NTFS parser of Stoned bootkit [226] [227] [228]
Aduska <sup>[Family]</sup>	2012	x86	[T13.a]	Troj/AduskMbr-A Rootkit.MBR.Whistler.F	[229] [230]
Dreamboot [Family] Concept	2013	x86 x64	[12.m] [T13.c]	No detections	[231] [232]
Halcbot <sup>[Family]</sup>	2013	x86	[T1.a] [T3.a] [T4.c] [T13.a]	Win32/CsNowDown Rootkit.Win32.Lapka Trojan:Win32/Pabueri	Intercepts <i>nt!loCreateFile</i> [233] [234] [235]
Caphaw Shylock <sup>[Family]</sup>	2013	x86	[T1.a] [T4.b] [T13.a]	Win32/Wolcape.A Trojan:WinNT/Caphaw.A Backdoor:Win32/Caphaw	[236] [237] <u>Malpedia</u>
Plite [Family]	2013	x86	[T13.a]	Rootkit.MBR.Plite.A Backdoor.Win32.Plite.b Trojan.Gpb.1 Trojan:Win32/Gupboot.A	[238] [239] [240]
Simda <sup>[Family]</sup>	2013	x86	[T2.a] [T2.m] [T3.a] [T3.f] [T4.b] [T13.b]	Trojan.Win64.Simda.at Backdoor:Win32/Simda Win32/Simda.B	Intercepts <i>KiDebugRoutine</i> [241] [242] [243] <u>Malpedia</u>
Gootkit [Family]	2014	x86 x64	[T13.b]	BackDoor.Gootkit.112 Win32/Rovnix.P	[244] [245] <u>Malpedia</u>
<b>Sednit</b> [Family] (early MBR version)	2014	x86 x64	[T1.a] [T3.i] [T6.a] [T13.a]	Win32/Rootkit.Agent.OAW Backdoor.Win64.Agent.If Trojan.Sednit.15	[246]
Careto Mask <sup>[Family]</sup>	2014	?	?	?	[247] [248] <u>Malpedia</u>
Pitou <sup>[Eamily]</sup>	2015	x86 x64	[T1.a] x86 [T3.c] [T3.j] [T4.c] [T13.a] [T14.e]	VirTool:WinNT/Pitou.B Rootkit.Win32.Turla.c Trojan:Win32/Pitou.A	VM detection via <i>MmMaploSpace</i> <sup>[249] [250] [251]</sup>
Hacking Team Vector EDK <sup>[Family]</sup>	2015	x86 x64	[T13.c]	Trojan:UEFI/VectorEDK.RKL Rootkit.EFI64.MosaicRegressor Trojan.Win32.MosaicRegressor	[252] [253] [254] <u>Malpedia</u> <u>S0047</u>
Grayfish <sup>[Family]</sup>	2015	x86 x64	[T12.g] [T13.b] [T14.a]	?	To bypass DSE it abuses the driver of the CloneCD

					program named ElbyCDIO.sys <sup>[255]</sup>
HDRoot [Family] Attributed to Winnti	2015/ 2006	x86	[T4.a] [T4.b] [T4.e] [T13.a]	Rootkit.Win32.HDRoot Trojan.Boot.HDRoot	[256] [257] <u>Malpedia</u>
BOOTRASH Nemesis [Family] Attributed to FIN1	2015	x86 x64	[T12.g] [T13.b]	No public samples	[258] [259] <u>S0114</u> <u>Malpedia</u>
LoJax [Family] Attributed to Sednit	2018	x86 x64	[T13.d]	EFI/LoJax.A Win32/SPIFlash.A Backdoor:Win32/Lojax Rootkit.EFI64.DoubleAgent.a	Abuses the RWEverything driver RwDrv.sys [260] [261]  Malpedia  S0397
DarkCloud [Family]	2018	x86	[T3.a] [T3.f] [T13.a]		[262] [263]
EfiGuard [Family] Concept	2019	x64	[12.m] [T13.c] [T14.f]	Rootkit.EFI64.EfiGuard.a Hacktool.Efiguard	A portable UEFI bootkit designed to disable PatchGuard and DSE <sup>[264]</sup>
MosaicRegressor [Family]	2020	x86 x64	[T13.d]	Trojan:UEFI/VectorEDK.RKL Rootkit.EFI64.MosaicRegressor Trojan.Win32.MosaicRegressor	[265] [266] <u>Malpedia</u>
FinSpy Finfisher <sup>[Family]</sup>	2021	x86 x64	[T13.a] [T13.c]	Backdoor:Win32/FinSpy Backdoor.Win32.Finfish	[267] [268] <u>Malpedia</u> <u>S0182</u>
ESPecter [Family]	2021	x64	[T9] [T13.a] [T13.c] [T14.e]	EFI/Rootkit.ESPecter Win64/Rootkit.ESPecter	[269] Creates \Device\WebBK <u>Malpedia</u>
MoonBounce [Family]	2022	x64	[T13.d] [T14.e]	Trojan:UEFI/MoonBounce.A EFI/MoonBounce.A Rootkit.EFI64.MoonBounce.b	Backdoor <sup>[270] [271]</sup>
BlackLotus <sup>[Family]</sup>	2023	x64	[T13.c]	Win64/BlackLotus.A EFI/BlackLotus.B Rootkit.BlackLotus.A Trojan.Win64.BlackLotus	Bypasses UEFI Secure Boot with CVE-2022-21894 + Capable of disabling BitLocker, HVCI, and Windows Defender [272] [273] [274] [275] Malpedia
Glupteba Windigo <sup>[Family]</sup>	2024	x64	[12.m] [T13.c] [T14.f]	Rootkit.EFI64.EfiGuard.a Hacktool.Efiguard	Utilizes EfiGuard to disable PatchGuard and DSE <sup>[276]</sup> Malpedia
2005 eEye BootRoot					

2007 IceLord Sinowal 2009 Stoned Whistler 2010 TDL4 MaxSS Mebromi Popureb Carberp Fispboot Evil Core 2011 Smitnyl Храј Gapz 2012 Aduska 19 2013 Dreamboot Caphaw Plite Simda

### Rootkit references

- [1] Greg Hoglund, James Butler | Rootkits: Subverting the Windows Kernel
- [2] Bill Blunden | *The Rootkit Arsenal*
- [3] Gary Nebbett | Windows NT/2000 Native API Reference
- [4] David A. Solomon, Mark Russinovich, Alex Ionescu, Pavel Yosifovich, Andrea Allievi | Windows Internals (7th edition)
- [5] Alex Matrosov, Eugene Rodionov, Sergey Bratus | Rootkits and Bootkits: Reversing Modern Malware and Next Generation Threats
- [6] Alexander Tereshkin (90210) | Bypassing Klister 0.4 With No Hooks or Running a Controlled Thread Scheduler
- [7] Microsoft | Windows Research Kernel (WRK)
- [8] Ron Ben Yizhak of Deepinstinct | #NoFilter Abusing Windows Filtering Platform for Privilege Escalation
- [9] Jonathan Johnson | <u>Understanding Telemetry: Kernel Callbacks</u>
- [10] Microsoft | Windows Filtering Platform
- [11] Microsoft | Event Tracing for Windows (ETW)
- [12] Microsoft | Filter Manager Concepts
- [13] Sherif Magdy, Mahmoud Zohdy of Trend Micro | An In-Depth Look at Windows Kernel Threats
- [14] Eugene Wineblat | Service Hiding
- [15] Michal Poslušný of ESET | Signed kernel drivers Unguarded gateway to Windows' core
- [16] Rotem Salinas of Cyberark | Fantastic Rootkits: And Where to Find Them (Part 1)
- [17] 90210 | Rootkits: Attacking Personal Firewalls
- [18] F-Secure | Spam from the kernel
- [19] Ph4nt0m Security Team | https://pstgroup.blogspot.com/2007/07/tips.html
- [20] James Butler of HBGary | <u>DKOM (Direct Kernel Object Manipulation)</u>
- [21] Mariusz Burdach, SANS Institute | FU rootkit
- [22] CardMagic(Edward) | PortHidDemo Vista
- [23] Kimmo Kasslin of F-Secure, Elia Florio of Symantec | When malware meets rootkits
- [24] MITRE ATT&CK | Hide Artifacts: Hidden File System
- [25] Alexander Tereshkin and Rafal Wojtczuk | Introducing Ring -3 Rootkits
- **%**NTRootkit
- [26] GitHub | Rootkit com mirror hoglund
- %⊱FU
- [27] Mariusz Burdach, SANS Institute | FU rootkit
- [28] GitHub | Rootkit\_com mirror Fuzen\_op
- **╬** Phide
- [29] GitHub | Rootkit com mirror 90210 |
- % Phide2
- [30] GitHub | Rootkit com mirror 90210

**%**≻FUTo

[31] GitHub | Rootkit com mirror petersilberman

∰Myfip

[32] F-Secure | Myfip.H

**∜**⊱Fanbot

[32]

\$∱Shadow Walker

[33] Sherri Sparks, Jamie Butler | «SHADOW WALKER» Raising The Bar For Rootkit Detection

[34] GitHub | Rootkit com mirror hoglund

[35] 90210 | Defeating Shadow Walker

% Sony XCP rootkit

[36] Microsoft | Sony, Rootkits and Digital Rights Management Gone Too Far

[37] Symantec | SecurityRisk.First4DRM

[38] Mark Russinovich | Inside Sony's rootkit

[39] Deirdre K. Mulligan, Aaron Perzanowski of University of Michigan | <u>The Magnificence of the Disaster: Reconstructing the Sony BMG</u>

**☆**Apropos

[40] F-Secure | Apropos

% Rustock.A

[41] Elia Florio, Prashant Pathak of Symantec | Raising the bar: Rustock and advances in rootkits

**%** Rustock.B

[42] Clifford Weaver | Finishing up Rustock.B

[43] Ken Chiang, Levi Lloyd of Sandia National Laboratories | A Case Study of the Rustock Rootkit and Spam Bot

[44] Frank Boldewin | A Journey to the Center of the Rustock.B Rootkit

% phide\_ex

[45] Yan Wen, Jinjing Zhao, Huaimin Wang | Implicit Detection of Hidden Processes with a Local-Booted Virtual Machine

[46] Carl Jongsma of Sûnnet Beskerming | <u>Undetectable Rootkit</u>

**☆**Unreal.A

[47] Carl Jongsma of Sûnnet Beskerming | <u>Undetectable Rootkit</u>

% Rustock.C

[48] Vyacheslav Rusakoff of Dr.Web | Win32.Ntldrbot (aka Rustock.C)

[49] Lukasz Kwiatek, Stanislaw Litawa of ESET | 'Yet another Rustock analysis ...'

[50] Chandra Prakash of Sunbelt Software | Your filters are bypassed: Rustock.C in the kernel

[51] Chandra Prakash of Sunbelt Software | Kernel mechanics of Rustock

**%**≻Haxdoor

- [52] Microsoft | WinNT/Haxdoor
- [53] Oleg Zaitsev | <u>Backdoor.Win32.Haxdoor.gu</u>
- [54] Desmond Lobo, Paul Watters and Xin-Wen Wu | Identifying Rootkit Infections Using Data Mining
- [55] Ken Dunham of iSIGHT Partners | Introduction to advanced memory analysis
- [56] Ken Dunham of iSIGHT Partners | Memory analysis examples
- [57] Sami Al-Shaheri, Dale Lindskog, Pavol Zavarsky of Concordia University College of Alberta | <u>A Forensic Study of the Effectiveness</u> of Selected Anti-Virus Products Against SSDT Hooking Rootkits
- **%**≻BlackEnergy
- [58] Jose Nazario of Arbor Networks | BlackEnergy DDoS Bot Analysis
- [59] Secureworks | BlackEnergy Version 2 Threat Analysis
- **%**≻Srizbi
- [60] Kimmo Kasslin of F-Secure, Elia Florio of Symantec | Spam from the kernel
- **%**≻Alman
- [61] kernelmode\_info | Virus.Win32.Alman.b
- [62] Desmond Lobo, Paul Watters and Xin-Wen Wu | Identifying Rootkit Infections Using Data Mining
- %≻TDL1
- [63] Vyacheslav Rusakov, Sergey Golovanov of Kaspersky | TDSS
- %≻TDL2
- [64] Lavasoft | An Analysis of Rootkit Technologies: Part 3
- [65] kernelmode\_info | Rootkit TDL 2 (Alias TDSS, Alureon.BK)
- **%**≻TDL3
- [66] kernelmode\_info | Rootkit TDL 3 (alias TDSS, Alureon.CT, Olmarik)
- [67] Dr.Web | BackDoor.Tdss.565 and its modifications (aka TDL3)
- [68] Ace Portuguez of F-Secure | The Case of Trojan DownLoader TDL3
- [69] Nguyễn Phố Sơn (t4l) | A Detailed Analysis of TDL Rootkit 3rd Generation
- **%**≻ZeroAccess
- [70] kernelmode\_info | Rootkit ZeroAccess (alias MaxPlus, Sirefef)
- [71] Marco Giuliani of PrevX | ZeroAccess an advanced kernel mode rootkit
- [72] Marco Giuliani of Webroot | TDL3 and ZeroAccess: More of the Same?
- [73] Vasily Berdnikov (vaber) | MAX++ sets its sights on x64 platforms
- [74] Sean Hittel and Rong Zhou of Symantec | <u>Trojan.ZeroAccess Infection Analysis</u>
- % Papras
- [75] Desmond Lobo, Paul Watters and Xin-Wen Wu | Identifying Rootkit Infections Using Data Mining
- **%**≻Koutodoor
- [76] Aditya Kapoor, Rachit Mathur of McAfee | Predicting the future of stealth attacks

#### % Stuxnet

- [77] Artem Baranov | Stuxnet drivers: detailed analysis
- [78] Alexander Gostev, Igor Kuznetsov | Stuxnet/Duqu: The Evolution of Drivers
- **%**≻Bubnix
- [79] kernelmode\_info | Nixoa/Bubnix Rootkit
- \$\rightarrow 4DW4R3 (TDL 2 clone)
- [80] kernelmode\_info | Rootkit 4DW4R3 (TDL 2 clone)
- **%**≻ Cutwail
- [81] Kyle Yang of Fortinet | Cut the Cutwail
- [82] Aditya Kapoor, Rachit Mathur of McAfee | Challenges in Kernel-Mode Memory Scanning
- [83] kernelmode\_info | Win32/Cutwail
- % Sality
- [84] Artem Baranov | Sality rootkit analysis
- ⇔ BlackEnergy v2
- [85] kernelmode\_info | WinNT/BlackEnergy
- **%**≻ Ramnit
- [86] kernelmode\_info | Win32/Ramnit
- [87] Chao Chen of Fortinet | Ramnit bot
- [88] Symantec | W32.Ramnit analysis
- % Necurs
- [89] Artem Baranov | Necurs rootkit under microscope
- [90] kernelmode\_info | Necurs another x64 rootkit
- [91] Peter Ferrie of Microsoft | The curse of Necurs, part 1
- [92] Peter Ferrie of Microsoft | The curse of Necurs, part 2
- [93] Peter Ferrie of Microsoft | The curse of Necurs, part 3
- [94] Vyacheslav Zakorzhevsky of Kaspersky | An unlikely couple: 64-bit rootkit and rogue AV for MacOS
- **%**Duqu
- [95] Boldizsár Bencsáth, Gábor Pék, Levente Buttyán, Márk Félegyházi of CrySyS | Duqu: A Stuxnet-like malware found in the wild
- [96] Guillaume Bonfante, Jean-Yves Marion, Fabrice Sabatier, Aurélien Thierry | Analysis and Diversion of Duqu's Driver
- **%**≻Festi
- [97] Eugene Rodionov, Alexander Matrosov of ESET | King of Spam: Festi Botnet Analysis
- [98] Check Point | Return of the Festi Rootkit
- **∜** WindowsRegistryRootkit
- [99] Cr4sh | WindowsRegistryRootkit on GitHub
- [100] Cr4sh | Applied anti-forensics: rootkits and kernel vulnerabilities

**%**≻Avatar

[101] Aleksandr Matrosov, Anton Cherepanov of ESET | <u>Mysterious Avatar rootkit with API, SDK, and Yahoo Groups for C&C communication</u>

[102] Aleksandr Matrosov, Anton Cherepanov of ESET | Avatar rootkit: the continuing saga

[103] kernelmode\_info | Win32/Rootkit.Avatar

**%**Sednit

[104] Artem Baranov | A note about Sednit rootkit

[105] ESET | En Route with Sednit Part 3: A Mysterious Downloader

**%**≻Turla

[106] deresz, tecamac | <u>Uroburos: the snake rootkit</u>

[107] Adam Crosser of praetorian | Developing a Hidden Virtual File System Capability That Emulates the Uroburos Rootkit

[108] hfiref0x | TDL (Turla Driver Loader)

[109] G Data | Uroburos Highly complex espionage software with Russian roots

[110] G Data | <u>Uroburos – Deeper travel into kernel protection mitigation</u>

[111] Paul Rascagneres of G DATA | <u>Uroburos</u>

https://www.kernelmode.info/forum/viewtopicbfb0.html?f=16&t=3193

**⇔**Duqu 2.0

[112] Kaspersky | The Mystery of Duqu 2.0: a sophisticated cyberespionage actor returns

**%**≻Grayfish

[113] Artem Baranov | GrayFish rootkit analysis

[114] Check Point | A Deep Dive Into DoubleFeature, Equation Group's Post-Exploitation Dashboard

**%**≻EquationDrug

[115] Artem Baranov | EquationDrug rootkit analysis (mstcp32.sys)

[116] Kaspersky | Inside the EquationDrug Espionage Platform

% Hacker's Door

[117] BlackBerry | Threat Spotlight: Opening Hacker's Door

% Derusbi

[118] Airbus | Newcomers in the Derusbi family

[119] Joe Desimone, Gabriel Landau | Kernel Mode Threats And Practical Defences

[120] CrowdStrike | Deep Panda

**%**→Wingbird

[121] Artem Baranov | Wingbird rootkit analysis

[122] Andrea Allievi and Elia Florio of Microsoft | FinFisher exposed: A researcher's tale of defeating traps, tricks, and complex virtual machines

**%**≻Finfisher

[123] Filip Kafka of ESET | ESET's guide to deobfuscating and devirtualizing FinFisher

[124] Microsoft | Office 365 Advanced Threat Protection defense for corporate networks against recent Office exploit attacks

[125] Kaspersky | FinSpy: unseen findings

★ Purple Fox

[126] Trend | A Look Into Purple Fox's New Arrival Vector

% Cahnadr

[127] Kaspersky | The Slingshot APT

**%**≻HideProc

[128] dhruval | HideProc on GitHub

[129] Ignacio Sanmillan of ESET | Ramsay: A cyber-espionage toolkit tailored for air-gapped networks

☆CEIDPageLock

[130] Israel Gubi of Check Point | CeidPageLock: A Chinese RootKit

**%**≻Zacinlo

[131] Claudiu Cobliș, Cristian Istrate, Cornel Punga, Andrei Ardelean of Bitdefender | Six Years and Counting: Inside the Complex Zacinlo Ad Fraud Operation

**%**≻Autochk

[132] Ori Damari (repnz) | Autochk Rootkit Analysis

**∜**FK\_Undead

[133] Lab52 | Recent FK Undead rootkit samples found in the wild

[134] Security Leopard | Legendary private servers hide hidden dangers, and the undead virus is raging in the arena

**%**≻Cronos

[135] XaFF-XaFF | Cronos-Rootkit on GitHub

**%** Demodex

[136] Mark Lechtik, Aseel Kayal, Paul Rascagneres, Vasily Berdnikov of Kaspersky | GhostEmperor: From ProxyLogon to kernel mode

**☆** DirtyMoe

[137] Martin Chlumecký of Avast | <u>DirtyMoe: Rootkit Driver</u>

% FiveSys

[138] Cristian Alexandru ISTRATE, Balazs BIRO, Rares Costin BLEOTU, Claudiu Stefan COBLIS of Bitdefender | Digitally-Signed Rootkits are Back – A Look at FiveSys and Companions

**%**≻FudModule

[139] AhnLab | Lazarus Group's Rootkit Attack Using BYOVD

[140] Jan Vojtěšek of Avast | Lazarus and the FudModule Rootkit: Beyond BYOVD with an Admin-to-Kernel Zero-Day

[141] Jonathan Johnson | <u>Understanding Telemetry: Kernel Callbacks</u>

[142] Pierre Ciholas, Jose Miguel Such, Angelos K. Marnerides, Benjamin Green, Jiajie Zhang, Utz Roedig | Fast and Furious: Outrunning Windows Kernel Notification Routines from User-Mode

[143] Dylan (@batsec) of MDSec | Bypassing Image Load Kernel Callbacks

% Fire Chili

[144] Rotem Sde-Or and Eliran Voronovitch of Fortinet | New Milestones for Deep Panda: Log4Shell and Digitally Signed Fire Chili Rootkits

∰InvisiMole

[145] Michal Poslušný | Signed kernel drivers – Unguarded gateway to Windows' core

**☆** Daxin

[146] Symantec | Daxin Backdoor: In-Depth Analysis, Part One

**%**Winnti

[147] Stéfan Le Berre of ExaTrack | From tweet to rootkit

[148] Chen Erlich, Fusao Tanida, Ofir Ozer, Akihiro Tomita, Niv Yona, Daniel Frank, Assaf Dahan of Cybereason | Operation CuckooBees: A Winnti Malware Arsenal Deep-Dive

## **Bootkit references**

% eEye BootRoot

[149] Derek Soeder and Ryan Permeh | <u>eEye BootRoot: A Basis for Bootstrap-Based Windows Kernel Code</u>

[150] Przemysław Gmerek | Stealth MBR rootkit

% lceLord

[151] Bernhard Grill, Andrei Bacs, Christian Platzer, Herbert Bos | «Nice Boots!» - A Large-Scale Analysis of Bootkits and New Ways to Stop Them

[152] Icelord | BIOS Rootkit: Welcome home, my Lord!

[153] Webroot | Mebromi: the first BIOS rootkit in the wild

% Vboot Kit

[154] Nitin Kumar, Vipin Kumar of nvLabs.in | Vboot Kit: Compromising Windows Vista Security

[155] Nitin Kumar, Vipin Kumar | Vboot Kit sources on GitHub

[156] Nitin Kumar, Vipin Kumar | VBootKit 2.0 – Attacking Windows 7 via Boot Sectors

★ Mebroot (Sinowal, Maosboot)

[157] Kimmo Kasslin of F-Secure, Elia Florio of Symantec | Your computer is now stoned (...again!)

[158] Marco Giuliani of Prevx | From Gromozon to Mebroot - A Reflection on Rootkits Today

[159] Michael Sandee et al of FOXiT | Post mortem report on the sinowal/nu nl incident

[160] Andrea Allievi of SaferBytes | Sinowal: MBR rootkit never dies!

[161] Kimmo of F-Secure | MBR Rootkit, A New Breed of Malware

[162] Sergey Golovanov, Alexander Gostev, Alexey Monastyrsky | Bootkit: the challenge of 2008

[163] A thread on km + samples

[164] Peter Kleissner | Analysis of Sinowal

%⊱Stoned

[165] Peter Kleissner | Stoned Bootkit

[166] Peter Kleissner | The Art of Bootkit Development

[167] Source code on GitHub

**%**≻Whistler

[168] Răzvan Stoica of Bitdefender | Whistler Bootkit Flies Under the Radar

[169] Răzvan Stoica of Bitdefender | <u>Unencrypted Whistler Variant in the Wild</u>

[170] gjf | Whistler Bootkit — нашему полку прибыло

[171] A thread km + samples

**%**≻Mebratix

[172] A thread on km + samples

#### [173] Peter Kleissner | Analysis of Mebratix



- [174] Joe Johnson of Microsoft | Alureon: The First In The Wild 64-Bit Windows Rootkit
- [175] Vyacheslav Rusakov of Kaspersky | TDSS. TDL-4
- [176] David Harley, Alexander Matrosov, Eugene Rodionov of ESET | TDL4 rebooted
- [177] David Harley, Alexander Matrosov, Eugene Rodionov of ESET | TDL4 reloaded: Purple Haze all in my brain
- [178] Eugene Rodionov, Alexander Matrosov of ESET | The Evolution of TDL: Conquering x64
- [179] Alexander Matrosov of ESET | Defeating x64: The Evolution of the TDL Rootkit
- [180] A L Johnson of Symantec | Tidserv 64-bit Goes Into Hiding
- [181] Mircea Ciubotariu of Symantec | Backdoor. Tidserv and x64
- [182] A thread on km + samples



- [183] Alexander Matrosov of ESET | Olmasco bootkit: next circle of TDL4 evolution (or not?)
- [184] Răzvan Stoica of Bitdefender | TDSS Bootkit Spawns Clones
- [185] A thread on km + samples
- ☆ PiXiEServ bootkit
- [186] Mateusz Jurczyk | PiXiEServ out for public
- [187] A thread on km + samples
- ★ Mebromi (Bioskit, Wador)
- [188] Webroot | Mebromi: the first BIOS rootkit in the wild
- [189] A thread on km + samples
- %¦≻Smitnyl
- [190] Low Chin Yick of F-Secure | Analysis of MBR File System Infector
- [191] A thread on km forum + samples
- [192] Sudonull | Analysis of Smitnyl.A, the first hybrid bootkit and file infection
- **%**≻Popureb
- [193] Symantec | MBR Confusion
- [194] Techkings.org | Don't write it, read it instead!
- [195] Marco Giuliani of Webroot | Removing Popureb Doesn't Require a Windows Reinstall
- [196] Oscar Abendan of Trend Micro | POPUREB: Launchpad for Future Threats
- [197] A thread on km + samples
- % Rovnix (Mayachok, Cidox, BKLoader)
- [198] Alexander Matrosov of ESET | Rovnix.D: the code injection story

- [199] Alexander Matrosov of ESET | Rovnix bootkit framework updated
- [200] Eugene Rodionov, Aleksandr Matrosov, David Harley of ESET | Rovnix Reloaded: new step of evolution
- [201] Vyacheslav Zakorzhevsky of Kaspersky | Cybercriminals switch from MBR to NTFS
- [202] Aleksandr Matrosov, Eugene Rodionov, David Harley of ESET | Hasta La Vista, Bootkit: Exploiting the VBR
- [203] Răzvan Stoica of Bitdefender | Mayachok Hooks INT8 to Dodge Emulators
- [204] Chun Feng of Microsoft | The evolution of Rovnix: Private TCP/IP stacks
- [205] McAfee | Cidox Trojan Spoofs HTTP Host Header to Avoid Detection
- [206] Marcus Hutchins | Rovnix new evolution
- [207] A thread on km + samples
- **%**≻Carberp
- [208] Aleksandr Matrosov, Eugene Rodionov, David Harley of ESET and Dmitry Volkov of Group-IB | <u>Evolution of Win32Carberp</u>
- [209] Aleksandr Matrosov, Eugene Rodionov, David Harley of ESET and Dmitry Volkov of Group-IB | Win32/Carberp
- [210] Aleksandr Matrosov, Eugene Rodionov, David Harley of ESET and Dmitry Volkov of Group-IB | Carberp Evolution
- [211] Carberp sources on GitHub
- [212] A thread on km + samples
- % Fisp (Fispboot)
- [213] Vyacheslav Zakorzhevsky | The Chinese bootkit
- **%**≻Evil Core
- [214] Wolfgang Ettlinger, Stefan Viehböck | Evil Core Bootkit, Pwning Multiprocessor Systems
- % Xpaj
- [215] Vyacheslav Rusakov of Kaspersky | Xpaj: Reversing a Windows x64 Bootkit
- [216] Răzvan Stoica of Bitdefender | Xpaj the bootkit edition
- [217] A thread on km + samples
- %\≻Yurn
- [218] Răzvan Stoica of Bitdefender | Yurn trojan adds bootkit functionality
- [219] A thread on km + samples
- **%**→Gapz
- [220] Dr.Web | Trojan.Gapz.1 infecting Windows in a new manner
- [221] Eugene Rodionov, Aleksandr Matrosov of ESET | Mind The Gapz: The Most Complex Bootkit Ever Analyzed?
- [222] Eugene Rodionov of ESET | Win32/Gapz: New Bootkit Technique
- [223] Aleksandr Matrosov of ESET | Win32/Gapz: steps of evolution
- [224] 0x16/7ton | Win32/Gapz family ring0 payload

[225] A thread on km + samples

% Guntior

[226] Artem Baranov | Guntior - the story of an advanced bootkit that doesn't rely on Windows disk drivers

[227] Paul of ZeroSecurity | Guntior Bootkit upgraded

[228] A thread on km + samples

**%**≻Aduska

[229] AhnLab | Adusca bootkit distributed to domestic PC users

[230] A thread km + samples

**\$**Dreamboot

[231] Sources on GitHub

[232] S'ebastien Kaczmarek of Quarkslab | <u>UEFI and Dreamboot</u>

**%** Halcbot

[233] AhnLab | Bootkit that steals online game users' account information

[234] AhnLab | Detailed analysis of Halcbot bootkit tampering with MBR

[235] A thread on km + samples

**%**Caphaw

[236] Aleksandr Matrosov of ESET | Caphaw attacking major European banks using webinject plugin

[237] A thread on km + samples

% Plite (PBBot, Gpb)

[238] Răzvan Stoica of Bitdefender | Plite Bootkit Spies on Gamers

[239] Dr.Web | Trojan.GBPBoot.1 MBR infector

[240] A thread on km + samples

**%**Simda

[241] Microsoft | WinNT/Simda

[242] 0x16/7ton | Win32/Simda family ring0 payload

[243] A thread on km + samples

**%**├Gootkit

[244] Dr.Web | BackDoor.Gootkit.112

[245] A thread on km + samples

% Sednit

[246] ESET | En Route with Sednit: A Mysterious Downloader

% Careto, Mask

[247] Kaspersky | The Careto/Mask APT: Frequently Asked Questions

[248] Kaspersky | <u>Unveiling "Careto" - The Masked APT</u>

% Pitou (Backboot)

[249] Gianfranco Tonello of TG Soft | Bootkits are not dead. Pitou is back!

[250] F-Secure | Pitou The «silent» resurrection of the notorious Srizbi kernel spambot

[251] A thread on km + samples

☆ Hacking Team Vector EDK

[252] Sources on GitHub

[253] Trend Micro | Hacking Team Uses UEFI BIOS Rootkit

[254] Kaspersky | MosaicRegressor: Lurking in the Shadows of UEFI

**%**├Grayfish

[255] Kaspersky | Equation Group: Questions and Answers

**%**≻HDRoot

[256] Dmitry Tarakanov of Kaspersky | Lam HDRoot! Part 1

[257] Dmitry Tarakanov of Kaspersky | <u>I am HDRoot! Part 2</u>

**%**≻BOOTRASH

[258] Dimiter Andonov, Willi Ballenthin, Nalani Fraser, Will Matson, Jay Taylor of FireEye | Thriving Beyond The Operating System: Financial Threat Group Targets Volume Boot Record

[259] Christopher Glyer of FireEye | Boot What?

**%**≻LoJax

[260] ESET | LoJax: First UEFI rootkit found in the wild, courtesy of the Sednit group

[261] ESET | LoJax First UEFI rootkit found in the wild, courtesy of the Sednit group

**%** DarkCloud

[262] Nirmal Singh of Zscaler | DarkCloud Bootkit

[263] jaemanshin of AhnLab | CoinMiner Infecting MBR is Distributed in Korea (DarkCloud Bootkit)

**%**≻EfiGuard

[264] Sources of GitHub

**%**≻MosaicRegressor

[265] Mark Lechtic, Igor Kuznetsov, Yury Parshin of Kaspersky | MosaicRegressor: Lurking in the Shadows of UEFI

[266] Kaspersky | MosaicRegressor: Lurking in the Shadows of UEFI, Technical Details

⟨⟨⟨⟩ FinSpy (Finfisher)

[267] Kaspersky | FinSpy: unseen findings

[268] Andrea Allievi and Elia Florio of Kaspersky | FinFisher exposed: A researcher's tale of defeating traps, tricks, and complex virtual machines

**%**≻ESPecter

[269] Martin Smolár and Anton Cherepanov of ESET | <u>UEFI threats moving to the ESP: Introducing ESPecter bootkit</u>

**%**≻MoonBounce

[270] Mark Lechtic, Vasily Berdnikov, Denis Legezo, Ilya Borisov of Kaspersky | MoonBounce: the dark side of UEFI firmware

[271] Binarly | A deeper UEFI dive into MoonBounce

**%**≻BlackLotus

[272] Martin Smolár | BlackLotus UEFI bootkit: Myth confirmed

[273] Sources on GitHub

[274] Alexander Matrosov of Binarly | The Untold Story of the BlackLotus UEFI Bootkit

[275] Microsoft | Guidance for investigating attacks using CVE-2022-21894: The BlackLotus campaign

∜⇔Glupteba

[276] Lior Rochberger and Dan Yashnik of Palo Alto Networks | Diving Into Glupteba's UEFI Bootkit

∜⇔Other

[277] Lars Haukli of Blue Coat Systems | Exposing Bootkits with BIOS Emulation

[278] Vyacheslav Rusakov and Sergey Golovanov of Kaspersky | Attacks before system startup

[279] Alexander Matrosov and Eugene Rodionov | <u>UEFI Firmware Rootkits: Myths and Reality</u>

[280] Takahiro Haruyama of VMware | Detecting UEFI Bootkits in the Wild

[281] Eclypsium | Trickbot Now Offers «TrickBoot»: Persist, Brick, Profit

[282] Eugene Rodionov, Aleksandr Matrosov, David Harley of ESET | Bootkit Threat Evolution in 2011

[283] BIOS Based Rootkits

[284] Rafal Wojtczuk and Alexander Tereshkin of Invisible Things Lab | Attacking Intel BIOS

[285] aLS and Alfredo of Core Security | Persistent BIOS infection