



# EXPLORANDO AMENAZAS AVANZADAS ENTRE EL ARRANQUE Y EL KERNEL: BOOTKITS Y ROOTKITS

Descifrando el desarrollo de Bootkits y Rootkits

[in/vazquez-vazquez-alejandro]

Cybersecurity Day 2024





# WHOAMI

- FRIKI (Fanático de Revolucionar Internamente Kernels e Inicios de sistema)
- Me gusta el pulpo, de ahí los B/Rootkits
- Ciberseguridad Ofensiva en Telefónica
- Docente en Máster de Análisis de Malware



# SENSITIVE CONTENT

## Age Verification

This presentation contains age-restricted materials including malware and explicit hooking techniques. By entering, you affirm that you are at least 18 years of age and you consent to viewing "hacker" stuff.

Let me In
This is real stuff

No I prefer OSINT



## CONCEPTOS

- Bootkit: Malicious program designed to load as early as possible in the boot process, in order to control all stages of the operating system start up, modifying system code and drivers before security components are loaded.
   Kaspersky
- Rootkit: Sophisticated piece of malware that can add new code to the operating system or delete and edit operating system code. Rootkits may remain in place for years because they are hard to detect, due in part to their ability to block some antivirus software and malware scanner software.
   CrowdStrike

## CONCEPTOS

 Bootkit: Malicious program designed to load as early as possible in the boot process, in order to control all stages of the operating system start up, modifying system code and drivers before security components are loaded.
 Kaspersky

**UEFI** Application

C/C++ - boot.efi

 Rootkit: Sophisticated piece of malware that can add new code to the operating system or delete and edit operating system code. Rootkits may remain in place for years because they are hard to detect, due in part to their ability to block some antivirus software and malware scanner software.

~ CrowdStrike

Kernel-Mode Driver

C/C++ - driver.sys

## PROTECCIONES

Administrator: Command Prompt

- Driver Signature Enforcement (DSE)
   Windows won't run drivers not certified by Microsoft
- Kernel Patch Protection (PatchGuard)
   Feature of 64-bit editions of Microsoft Windows
   Prevents patching the kernel
- SecureBoot
   Only software trusted by the Original Manufacturer.
   Firmware checks the signature of UEFI firmware drivers,
  - EFI applications and SO

Microsoft Windows [Version 10.0.22631.3007] (c) Microsoft Corporation. All rights reserved. C:\Windows\System32>.\sc.exe create POC type=kernel start=demand binpath= "C:\Users\user1\Documents\Rootkit\Driver.sys" [SC] CreateService SUCCESS C:\Windows\System32>.\sc.exe start POC [SC] StartService FAILED 577: Windows cannot verify the digital signature for this file. A recent hardw are or software change might have installed a file that is signed incorre ctly or damaged, or that might be malicious software from an unknown sour ce. C:\Windows\System32> Invalid signature detected. Check Secure Boot Policy in Setup

• ELAM, VBS, CFG, ...







**POST** 



**UEFI Firmware** 



**Boot Information** 

## **Windows Boot Manager** \EFI\Microsoft\Boot\ bootmgfw.efi

To specify an advanced option for this choice, press F8.





Boot order

Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi

Boot0002 = /EFI/Ubuntu/shimx64.efi Boot000x = /EFI/Vendor/bootx64.efi

#### **Windows NT OS Kernel** %SystemRoot%\system32\ ntoskrnl.exe



Executive

Processes



Safe Mode with Networking Safe Mode with Command Prompt Enable low-resolution video (640x480) Directory Services Restore Mode Disable automatic restart on system failure Disable Driver Signature Enforcement Description: View a list of system recovery tools you can use to repair startup problems, run diagnostics, or restore your system.









**POST** 



**UEFI Firmware** 



**Boot Information** 

# Windows Boot Manager \EFI\Microsoft\Boot\ bootmgfw.efi

Choose an operating system to start, or press TAB to select a tool: (Use the arrow keys to highlight your choice, then press ENTER.)

Windows Vist Windows XP

To specify an advanced option for this choice, press F8. Seconds until the highlighted choice will be started automatically: 1

Tools

Windows Memory Diagnos

ENTER=Choose

ESC=Cance1

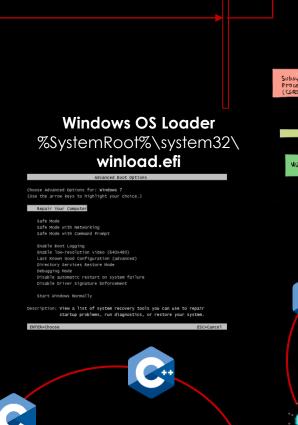


Boot order

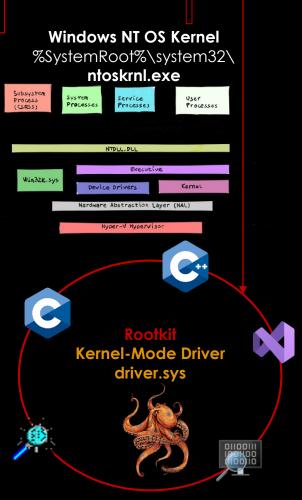
Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi

Boot0002 = /EFI/Ubuntu/shimx64.efi

Boot000x = /EFI/Vendor/bootx64.efi



UEFI Application bootmgfw.efi









**POST** 

gBS->LoadImage();



**UEFI Firmware** 



**Boot Information** 

# Windows Boot Manager \EFI\Microsoft\Boot\ bootmgfw.efi

Choose an operating system to start, or press TAB to select a tool:
(Use the arrow keys to highlight your choice, then press ENTER.)

Windows 7 Windows Vis

To specify an advanced option for this choice, press F8.

Seconds until the highlighted choice will be started automatically: 1

Tools

Windows Memory Diagnos

ENTER=Choose

ESC=Cance



Boot order

Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi

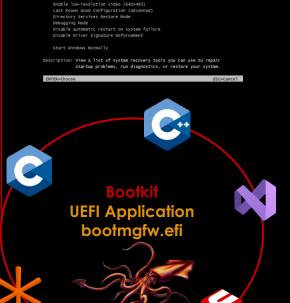
Boot0002 = /EFI/Ubuntu/shimx64.efi

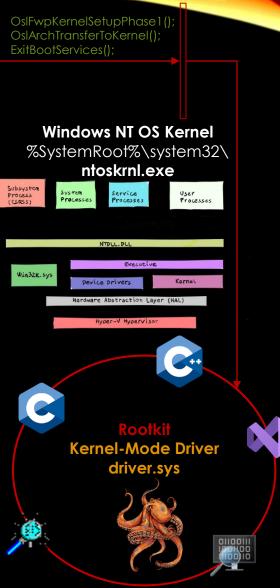
Boot000x = /EFI/Vendor/bootx64.efi

ImgArchStartBootApplication(); Archpx64TransferTo64BitApplicationAsm();

Safe Mode with Command Prompt















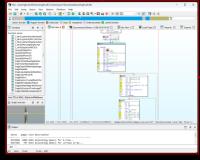
**POST** 



**UEFI Firmware** 



**Boot Information** 



Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi

Windows Boot Manager

Choose an operating system to start, or press TAB to select a tool:
(Use the arrow keys to highlight your choice, then press EMTER.)

Windows 7.

Windows Vista
Windows XP

To specify an advanced option for this choice, press FB.
Seconds until the highlighted choice will be started automatically: 10

Tools:

Windows Memory Diagnostic

ENTER-Choose

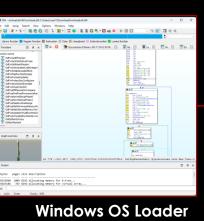
TAB-Henu ESSC-Cat



Boot order

Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi

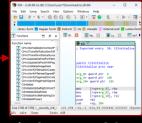
Boot0002 = /EFI/Ubuntu/shimx64.efi Boot000x = /EFI/Vendor/bootx64.efi



Windows OS Loader %SystemRoot%\system32\ winload.efi





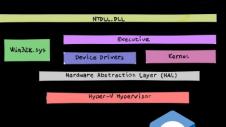


Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe























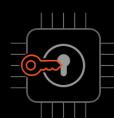
**POST** 

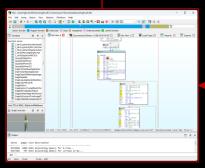


**UEFI Firmware** 



**Boot Information** 





**Windows Boot Manager** \EFI\Microsoft\Boot\ bootmgfw.efi

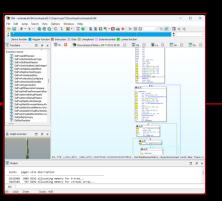




Boot order

Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi

Boot0002 = /EFI/Ubuntu/shimx64.efi Boot000x = /EFI/Vendor/bootx64.efi



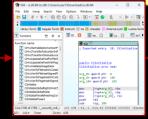
Windows OS Loader %SystemRoot%\system32\ winload.efi





**UEFI** Application bootmgfw.efi



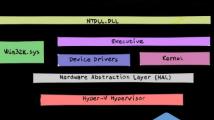


**Windows NT OS Kernel** %SystemRoot%\system32\ ntoskrnl.exe



















## **ENTORNO**

## WDK

- Develop, test and deploy drivers for Windows.
- Software toolset from Microsoft.

## EDK2

- Official development environment for UEFI applications, UEFI Drivers, DXE Drivers.
- Developed by the open-source Tianocore project (Intel, HP and Microsoft)
- Full on implementation of the UEFI specification.

#### • +

- Visual Studio, C/C++, Reverse Engineering, WinDbg, IDA







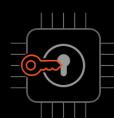
**POST** 

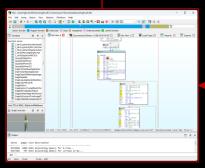


**UEFI Firmware** 



**Boot Information** 





**Windows Boot Manager** \EFI\Microsoft\Boot\ bootmgfw.efi

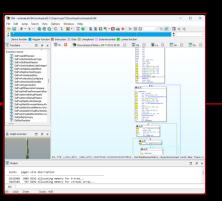




Boot order

Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi

Boot0002 = /EFI/Ubuntu/shimx64.efi Boot000x = /EFI/Vendor/bootx64.efi



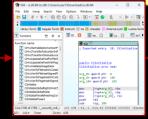
Windows OS Loader %SystemRoot%\system32\ winload.efi





**UEFI** Application bootmgfw.efi



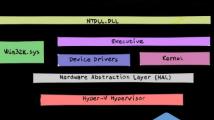


**Windows NT OS Kernel** %SystemRoot%\system32\ ntoskrnl.exe













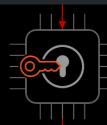


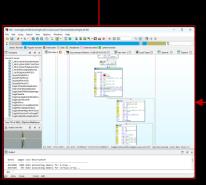




#### **★ UEFI Specification**

- 1. Introduction
- 2. Overview 3. Boot Manager
- 4. EFI System Table
- 5. GUID Partition Table (GPT) Disk
- 6. Block Translation Table (BTT) Layout
- 7. Services Boot Services
- 8. Services Runtime Services
- 9. Protocols EFI Loaded Image
- 24. Network Protocols SNP, PXE, BIS and HTTP Boot
- 25. Network Protocols Managed Network
- 26. Network Protocols Bluetooth
- 27. Network Protocols VLAN, EAP, Wi-Fi and Supplicant
- 28. Network Protocols TCP. IP. IPsec. FTP, TLS and Configurations
- 29. Network Protocols ARP, DHCP, DNS, HTTP and REST
- 30. Network Protocols UDP and

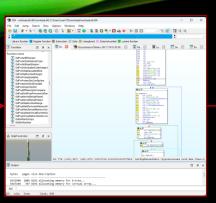




## **Windows Boot Manager** \EFI\Microsoft\Boot\ bootmgfw.efi

Choose an operating system to start, or press TAB to select a tool

To specify an advanced option for this choice, press F8.



#### **Windows OS Loader** %SystemRoot%\system32\ winload.efi

Choose Advanced Options for: Windows 7

Safe Mode with Command Prompt

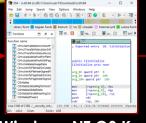
Directory Services Restore Mode

Disable automatic restart on system failure Disable Driver Signature Enforcement

Start Windows Normally

**UEFI** Application bootmgfw.efi





**Windows NT OS Kernel** %SystemRoot%\system32\ ntoskrnl.exe











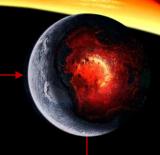


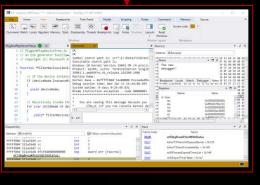


**Kernel-Mode Driver** driver.sys







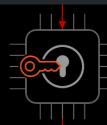


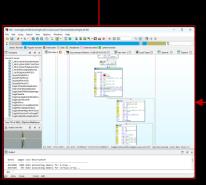




#### **★ UEFI Specification**

- 1. Introduction
- 2. Overview 3. Boot Manager
- 4. EFI System Table
- 5. GUID Partition Table (GPT) Disk
- 6. Block Translation Table (BTT) Layout
- 7. Services Boot Services
- 8. Services Runtime Services
- 9. Protocols EFI Loaded Image
- 24. Network Protocols SNP, PXE, BIS and HTTP Boot
- 25. Network Protocols Managed Network
- 26. Network Protocols Bluetooth
- 27. Network Protocols VLAN, EAP, Wi-Fi and Supplicant
- 28. Network Protocols TCP. IP. IPsec. FTP, TLS and Configurations
- 29. Network Protocols ARP, DHCP, DNS, HTTP and REST
- 30. Network Protocols UDP and

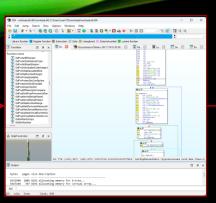




## **Windows Boot Manager** \EFI\Microsoft\Boot\ bootmgfw.efi

Choose an operating system to start, or press TAB to select a tool

To specify an advanced option for this choice, press F8.



#### **Windows OS Loader** %SystemRoot%\system32\ winload.efi

Choose Advanced Options for: Windows 7

Safe Mode with Command Prompt

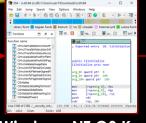
Directory Services Restore Mode

Disable automatic restart on system failure Disable Driver Signature Enforcement

Start Windows Normally

**UEFI** Application bootmgfw.efi





**Windows NT OS Kernel** %SystemRoot%\system32\ ntoskrnl.exe











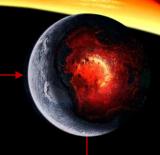


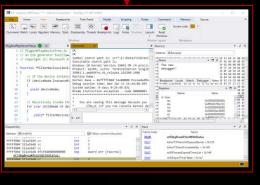


**Kernel-Mode Driver** driver.sys











## UEFI

UEFI or (Unified) Extensible Firmware Interface is a specification for x86, x86-64, ARM, and Itanium platforms that defines a software interface between the operating system and the platform firmware/BIOS.

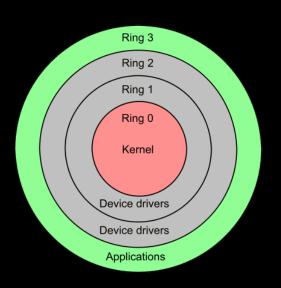
SEC (Security)	PEI (Pre-EFI Init)	DXE (Driver Execution Environment)	BDS (Boot Device Selection)	TSL (Transient System Load)	RT (Run Time)	AL (After Life)
	PEI Modules	DXE Drivers	UEFI Boot Manager	GRUB, Windows Boot Manager,	OS Kernel	
		SMM Modules				

## UEFI

Two types of services apply in an compliant system:

- Boot Services: Functions that are available before a successful call to ExitBootServices().
- Runtime Services: Functions that are available before and after any call to ExitBootServices().

## KERNEL



The *kernel* of an operating system implements the core functionality that everything else in the operating system depends upon. The Microsoft Windows kernel provides basic low-level operations such as scheduling threads or routing hardware interrupts. It is the heart of the operating system and all tasks it performs must be fast and simple.

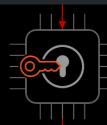
## KERNEL

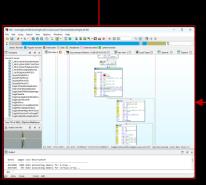
## Kernel-Mode Driver Architecture Design Guide:

- Kernel-Mode Components: Describe the primary kernel-mode managers and components of the Windows OS (Kernel Library, Memory Manager, etc).
- Windows Driver Model (WDM): To allow driver developers to write device drivers that are source-code compatible across all Microsoft Windows operating systems, WDM was introduced.
- Windows Driver Frameworks (WDF): It is an abstraction layer that takes care of much of the common code required to write a Windows driver.

#### **★ UEFI Specification**

- 1. Introduction
- 2. Overview 3. Boot Manager
- 4. EFI System Table
- 5. GUID Partition Table (GPT) Disk
- 6. Block Translation Table (BTT) Layout
- 7. Services Boot Services
- 8. Services Runtime Services
- 9. Protocols EFI Loaded Image
- 24. Network Protocols SNP, PXE, BIS and HTTP Boot
- 25. Network Protocols Managed Network
- 26. Network Protocols Bluetooth
- 27. Network Protocols VLAN, EAP, Wi-Fi and Supplicant
- 28. Network Protocols TCP. IP. IPsec. FTP, TLS and Configurations
- 29. Network Protocols ARP, DHCP, DNS, HTTP and REST
- 30. Network Protocols UDP and

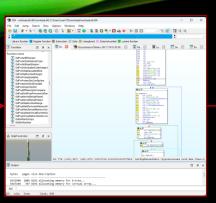




## **Windows Boot Manager** \EFI\Microsoft\Boot\ bootmgfw.efi

Choose an operating system to start, or press TAB to select a tool

To specify an advanced option for this choice, press F8.



#### **Windows OS Loader** %SystemRoot%\system32\ winload.efi

Choose Advanced Options for: Windows 7

Safe Mode with Command Prompt

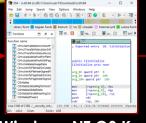
Directory Services Restore Mode

Disable automatic restart on system failure Disable Driver Signature Enforcement

Start Windows Normally

**UEFI** Application bootmgfw.efi





**Windows NT OS Kernel** %SystemRoot%\system32\ ntoskrnl.exe











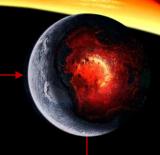


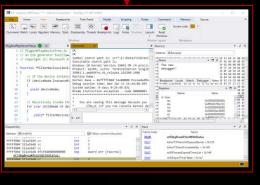


**Kernel-Mode Driver** driver.sys





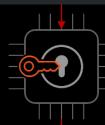


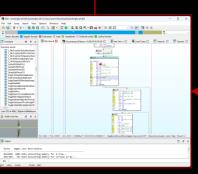




#### ★ UEFI Specification

- 1. Introduction
- 2. Overview
- 3. Boot Manager
- 4. EFI System Table
- 5. GUID Partition Table (GPT) Disk
- Layout
- 6. Block Translation Table (BTT) Layout
- 7. Services Boot Services
- 8. Services Runtime Services
- 9. Protocols EFI Loaded Image
- 24. Network Protocols SNP, PXE, BIS and HTTP Boot
- 25. Network Protocols Managed Network
- 26. Network Protocols Bluetooth
- 27. Network Protocols VLAN, EAP, Wi-Fi and Supplicant
- 28. Network Protocols TCP, IP, IPsec, FTP, TLS and Configurations
- 29. Network Protocols ARP, DHCP, DNS. HTTP and REST
- 30. Network Protocols UDP and MTFTP





# Windows Boot Manager \EFI\Microsoft\Boot\ bootmgfw.efi

Windows Boot Man

Choose an operating system to start, or press TAB to select a tool:
(Use the arrow keys to highlight your choice, then press ENTER.)

Windows 7 Windows Vista

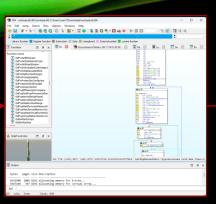
To specify an advanced option for this choice, press F8.

Seconds until the highlighted choice will be started automatically: 10

Tools:

Windows Memory Diagnos

ENTER=Choose TAB=Menu ESC=Cance1



#### Windows OS Loader %SystemRoot%\system32\ winload.efi

Choose Advanced Options for: Windows 7
(Use the arrow keys to highlight your choice.)

Repair Your Computer

Safe Mode Safe Mode with Networking Safe Mode with Command Prompt

Enable Boot Logging
Enable low-resolution video (640x480)

Last Known Good Configuration (advanced) Directory Services Restore Mode Debugging Mode

Disable automatic restart on system failure Disable Driver Signature Enforcement

Start Windows Normally

Description: View a list of system recovery tools you can use to repar

ESC=Cancel

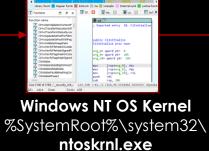


## Bootkit

UEFI Application bootmgfw.efi DXE Runtime Driver

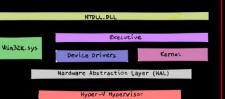
אב kuntime Dr plus.efi





bsystem rocess srss)

ystem Processes iervice Processes User Processes

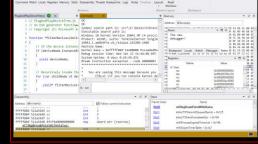




Kernel-Mode Driver driver.sys

Filter Driver plus.sys







# DESARROLLO

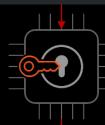
UEFI Application	DXE Runtime Driver	Kernel-Mode Driver	Filter Driver
UEFI Specification	DXE	WDF (KMDF)	WDM
Download Malware	Hooking Functions	Hide Processes	Filter Network Traffic
Boot Services	Runtime Services	Keylogger	Filter File System
C + EDK2 -> .efi	C + EDK2 -> .efi	C + WDK -> .sys	C + WDK -> .sys

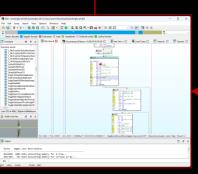
Bootkit

Rootkit

#### ★ UEFI Specification

- 1. Introduction
- 2. Overview
- 3. Boot Manager
- 4. EFI System Table
- 5. GUID Partition Table (GPT) Disk
- Layout
- 6. Block Translation Table (BTT) Layout
- 7. Services Boot Services
- 8. Services Runtime Services
- 9. Protocols EFI Loaded Image
- 24. Network Protocols SNP, PXE, BIS and HTTP Boot
- 25. Network Protocols Managed Network
- 26. Network Protocols Bluetooth
- 27. Network Protocols VLAN, EAP, Wi-Fi and Supplicant
- 28. Network Protocols TCP, IP, IPsec, FTP, TLS and Configurations
- 29. Network Protocols ARP, DHCP, DNS. HTTP and REST
- 30. Network Protocols UDP and MTFTP





# Windows Boot Manager \EFI\Microsoft\Boot\ bootmgfw.efi

Windows Boot Man

Choose an operating system to start, or press TAB to select a tool:
(Use the arrow keys to highlight your choice, then press ENTER.)

Windows 7 Windows Vista

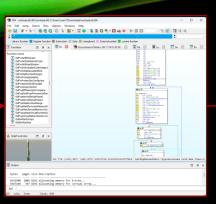
To specify an advanced option for this choice, press F8.

Seconds until the highlighted choice will be started automatically: 10

Tools:

Windows Memory Diagnos

ENTER=Choose TAB=Menu ESC=Cance1



#### Windows OS Loader %SystemRoot%\system32\ winload.efi

Choose Advanced Options for: Windows 7
(Use the arrow keys to highlight your choice.)

Repair Your Computer

Safe Mode Safe Mode with Networking Safe Mode with Command Prompt

Enable Boot Logging
Enable low-resolution video (640x480)

Last Known Good Configuration (advanced) Directory Services Restore Mode Debugging Mode

Disable automatic restart on system failure Disable Driver Signature Enforcement

Start Windows Normally

Description: View a list of system recovery tools you can use to repar

ESC=Cancel

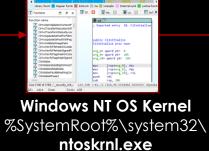


## Bootkit

UEFI Application bootmgfw.efi DXE Runtime Driver

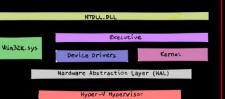
אב kuntime Dr plus.efi





bsystem rocess srss)

ystem Processes iervice Processes User Processes

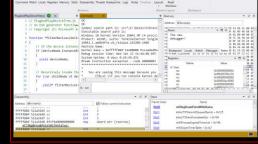




Kernel-Mode Driver driver.sys

Filter Driver plus.sys







## **Glupteba Overview**

Glupteba is built to be modular, which allows it to download and execute additional components or payloads. This modular design makes Glupteba adaptable to different attack scenarios and environments, and it also allows its operators to adapt to different security solutions.

Over the years, malware authors have introduced new modules, allowing the threat to perform a variety of tasks including the following:

- Delivering additional payloads
- Stealing credentials from various software
- Stealing sensitive information, including credit card data
- Enrolling the infected system in a cryptomining botnet
- Crypto hijacking and delivering miners
- Performing digital advertising fraud
- Stealing Google account information
- Bypassing UAC and having both rootkit and bootkit components
- Exploiting routers to gain credentials and remote administrative access

GREAT

GREAT

FinSpy, also known as FinFisher or Wingbird, is an infamous surveillance toolset. Kaspersky has been tracking deployments of this spyware since 2011. Historically, its Windows implant was distributed through a single-stage installer. This version was detected and researched several times up to 2018. Since that year, we observed a decreasing detection rate of FinSpy for Windows. While the nature of this anomaly remained unknown, we began detecting some suspicious installers of legitimate applications, backdoored with a relatively small obfuscated downloader. We were unable to cluster those packages until the middle of 2019 when we found a host that served these installers among FinSpy Mobile implants for Android. Over the course of our investigation, we found out that the backdoored installers are nothing more than first stage implants that are used to download and deploy further payloads before the actual FinSpy Trojan.

Apart from the Trojanized installers, we also observed infections involving usage of a UEFI or MBR bootkit. While the MBR infection has been known since at least 2014, details on the UEFI bootkit are publicly revealed in this article for the first time.

## **BOOTKIS IN THE WILD**

#### ESET RESEARCH

# UEFI threats moving to the ESP: Introducing ESPecter bootkit

ESET research discovers a previously undocumented UEFI bootkit with roots going back all the way to at least 2012



Martin Smolár



Anton Cherepanov

ESET researchers have analyzed a previously undocumented, real-world UEFI bootkit that persists on the EFI System Partition (ESP). The bootkit, which we've named ESPecter, can bypass Windows Driver Signature Enforcement to load its own unsigned driver, which facilitates its espionage activities. Alongside Kaspersky's recent discovery of the unrelated FinSpy bootkit, it is now safe to say that real-world UEFI threats are no longer limited to SPI flash implants, as used by Lojax.

The days of UEFI (Unified Extensible Firmware Interface) living in the shadows of the legacy BIOS are gone for good. As a leading technology embedded into chips of modern computers and devices, it plays a crucial role in securing the pre-OS environment and loading the operating system. And it's no surprise that such a widespread technology has also become a tempting target for threat actors in their search for ultimate persistence.

#### ESET RESEARCH

# BlackLotus UEFI bootkit: Myth confirmed

The first in-the-wild UEFI bootkit bypassing UEFI Secure Boot on fully updated UEFI systems is now a reality



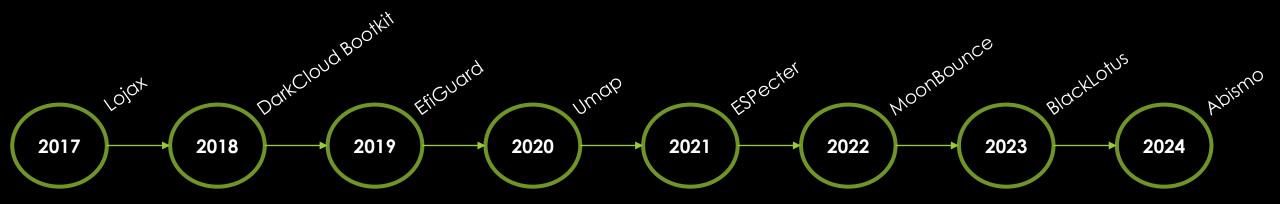
Martin Smolár

The number of UEFI vulnerabilities discovered in recent years and the failures in patching them or revoking vulnerable binaries within a reasonable time window hasn't gone unnoticed by threat actors. As a result, the first publicly known UEFI bootkit bypassing the essential platform security feature – UEFI Secure Boot – is now a reality. In this blogpost we present the first public analysis of this UEFI bootkit, which is capable of running on even fully-up-to-date Windows 11 systems with UEFI Secure Boot enabled. Functionality of the bootkit and its individual features leads us to believe that we are dealing with a bootkit known as **BlackLotus**, the UEFI bootkit being sold on hacking forums for \$5,000 since at least October 2022.

UEFI bootkits are very powerful threats, having full control over the OS boot process and thus capable of disabling various OS security mechanisms and deploying their own kernel-mode or user-mode payloads in early OS startup stages. This allows them to operate very stealthily and with high privileges. So far, only a few have been discovered in the wild and publicly described (e.g., multiple malicious EFI samples we discovered in 2020, or fully featured UEFI bootkits such as our discovery last year – the ESPecter bootkit – or the FinSpy bootkit discovered by researchers from Kaspersky).

# BOOTKIS IN THE WILD

An ESP malware implant executes code before Windows boots, undermining security features. An SPI flash memory implant offers more control, executing earlier in the boot process but requires higher privileges, increasing complexity. ~ Malware Developers



## **GLUPTEBA**

## Glupteba Overview

Glupteba is built to be modular, which allows it to download and execute additional components or payloads. This modular design makes Glupteba adaptable to different attack scenarios and environments, and it also allows its operators to adapt to different security solutions.

Over the years, malware authors have introduced new modules, allowing the threat to perform a variety of tasks including the following:

- Delivering additional payloads
- Stealing credentials from various software
- Stealing sensitive information, including credit card data
- Enrolling the infected system in a cryptomining botnet
- Crypto hijacking and delivering miners
- Performing digital advertising fraud
- Stealing Google account information
- Bypassing UAC and having both rootkit and bootkit components
- Exploiting routers to gain credentials and remote administrative access

## **Diving Into Glupteba's UEFI Bootkit**



By Lior Rochberger and Dan Yashnik February 12, 2024 at 6:00 AM Category: Malware

Tags: Advanced Threat Prevention, Advanced URL Filtering, Advanced WildFire, Cloud-Delivered Security Services, coin miner, Cortex XDR, credential stealer, DNS security, next-generation firewall, Prisma Cloud, RedLine infostealer, Smoke Loader

## **Executive Summary**

Glupteba is advanced, modular and multipurpose malware that, for over a decade, has mostly been seen in financially driven cybercrime operations. This article describes the infection chain of a new campaign that took place around November 2023.

Despite being active for over a decade, certain capabilities that Glupteba's authors have added have remained undiscovered or unreported – until now. We will focus on one intriguing and previously undocumented feature: a Unified Extensible Firmware Interface (UEFI) bootkit. This bootkit can intervene and control the OS boot process, enabling Glupteba to hide itself and create a stealthy persistence that can be extremely difficult to detect and remove.

## **GLUPTEBA**

## **Uncovering Glupteba's Bootkit Installer**

We start our analysis with a bootkit installer binary disguised as a legitimate Windows binary (csrss.exe). When analyzing this installer, a clear lack of strings and functions indicates the file is packed in some way. This means we have some work to do before we can analyze the actual logic of the installer.

- The main\_mountEFI function mounts the ESP into the B: drive
- 2. B:\EFI\Microsoft\Boot\bootmgfw.efi is renamed to B:\EFI\Microsoft\Boot\fw.efi
- 3. B:\EFI\Boot\bootx64.efi is renamed to B:\EFI\Boot\old.efi
- 4. The asset embedded\bootmgfw.efi is written to B:\EFI\Microsoft\Boot\bootmgfw.efi and to B:\EFI\Boot\bootx64.efi
- 5. The asset embedded \EfiGuardDxe.efi is written to B:\EFI\Boot\EfiGuardDxe.efi

#### **UEFI** infection

During our research, we found a UEFI bootkit that was loading FinSpy. All machines infected with the UEFI bootkit had the Windows Boot Manager (bootmgfw.efi) replaced with a malicious one. When the UEFI transfers execution to the malicious loader, it first locates the original Windows Boot Manager. It is stored inside the efi\microsoft\boot\en-us\ directory, with the name consisting of hexadecimal characters. This directory contains two more files: the Winlogon Injector and the Trojan Loader. Both of them are encrypted with RC4. The decryption key is the EFI system partition GUID, which differs from one machine to another.

P:\EFI\Microsoft\Boot\en-US							
n Name	Size						
	Up						
050ad6a5	468480 Encrypted Backdoor Loader						
4182b569	1492 K Original Windows Boot Manager						
82056bd2	6236 Encrypted Winlogon Injector						
bootmgfw.efi.mui	77112						
bootmgr.efi.mui	77112 Clean files						
memtest.efi.mui	44856 J						

Sample contents of the \efi\microsoft\boot\en-us\ directory

Once the original bootloader is located, it is loaded into memory, patched and launched. The patched launcher:

- Patches the function of the OS loader that transfers execution to the kernel
- The patched function hooks the kernel's PsCreateSystemThread function, which, when called for the first time, creates an additional thread that decrypts the next loader stage and launches it.

## **FINFISHER**

- Spyware
- UEFI Bootkit
- Windows, Linux and MacOS
- Obfuscation
- C&C

## *v*elivesecurity west

By patching the Windows Boot Manager, attackers achieve execution in the early stages of the system boot process (see Figure 1), before the operating system is fully loaded. This allows ESPecter to bypass Windows Driver Signature Enforcement (DSE) in order to execute its own unsigned driver at system startup. This driver then injects other user-mode components into specific system processes to initiate communication with ESPecter's C&C server and to allow the attacker to take control of the compromised machine by downloading and running additional malware or executing C&C commands.

Even though Secure Boot stands in the way of executing untrusted UEFI binaries from the ESP, over the last few years we have been witness to various UEFI firmware vulnerabilities affecting thousands of devices that allow disabling or bypassing Secure Boot (e.g. VU#758382, VU#976132, VU#631788, ...). This shows that securing UEFI firmware is a challenging task and that the way various vendors apply security policies and use UEFI services is not always ideal.

Previously, we have reported multiple malicious EFI samples in the form of simple, single-purpose UEFI applications without extensive functionality. These observations, along with the concurrent discovery of the ESPecter and FinFisher bootkits, both fully functional UEFI bootkits, show that threat actors are not relying only on UEFI firmware implants when it comes to pre-OS persistence, but also are trying to take advantage of disabled Secure Boot to execute their own ESP implants.

We were not able to attribute ESPecter to any known threat actor, but the Chinese debug messages in the associated user-mode client component (as seen in Figure 2) leads us to believe with a low confidence that an unknown Chinese-speaking threat actor is behind ESPecter. At this point, we don't know how it was distributed.

## **ESPECTER**

- Spyware
- UEFI Bootkit
- Windows
- DSE
- Mode-Kernel Driver

# **ESPECTER**

# KEYBOARD\_INPUT\_DATA structure (ntddkbd.h) KEYBOARD\_INPUT\_DATA contains one packet of keyboard input data. Syntax C++ Lypedef struct \_KEYBOARD\_INPUT\_DATA { USHORT UnitId; USHORT MakeCode; USHORT Reserved; ULONG ExtraInformation; } KEYBOARD\_INPUT\_DATA, \*PKEYBOARD\_INPUT\_DATA;

```
@brief
               Function to read keystrokes from keyboard device.
               pDeviceObject
                                   Pointer to a DEVICE OBJECT structure representing the device.
    @param
                                   Pointer to the I/O request packet (IRP) for reading keystrokes.
    @param
                pIrp
               A NTSTATUS value indicating success or an error code if operation fails.
    @return
NTSTATUS
DriverReadKeystrokes(
           PDEVICE_OBJECT
                               pDeviceObject,
    In
    In
            PIRP
                               pIrp
    IoCopyCurrentIrpStackLocationToNext(pIrp);
    IoSetCompletionRoutine(pIrp, ReadOperationFinished, NULL, TRUE, TRUE);
    pendingKey++;
    return IoCallDriver(((PDEVICE EXTENSION))pDeviceObject->DeviceExtension)->LowerKbdDevice, pIrp);
```

## BLACKLOTUS

## A new name for old tricks

Are there any new techniques in the BlackLotus bootkit? My opinion is that BlackLotus is a good combination of well-known techniques. Proof of concept code for CVE-2022-21894 (public since August 2022) was taken from the GitHub repository of the researcher who found the vulnerability.

Binarly REsearch discovered new interesting data points about the nature of the bootkit code. It appears the author of the BlackLotus bootkit based their development on code from the Umap GitHub project (Windows UEFI bootkit that loads a generic driver manual mapper without using a UEFI runtime driver) or coincidently arrived at the same ideas. According to the first commit, Umap was released in April 2020.

The kernel driver is responsible for four main tasks:

- 1. Injecting the HTTP downloader into winlogon.exe and reinjecting it in case the thread terminated.
- 2. Protecting bootkit files deployed on the ESP from being removed.
- 3. Disarming the user-mode Windows Defender process MsMpEngine.exe.
- 4. Communicating with the HTTP downloader and if necessary, performing any commands.



# MÁSTER EN REVERSING, ANÁLISIS DE MALWARE Y BUG HUNTING







## Reversing en Sistemas Operativos Windows:

- Windows architecture (User mode and Kernel mode)
- Windows protections (DSE, KPP, VBS, CFG)
- Malware hunting with SysInternals tools
- Windows kernel opaque structures (EPROCESS, ETHREAD)
- Windows kernel debugging
- WinDbg scripting (Commands, Javascript, PyKd)
- Rootkit hooking techniques (IDT, SSDT)
- Rootkit development (Kernel Mode Drivers)
- Bootkit development (UEFI Applications)
- Bootkit analysis (ESPecter, BlackLotus)
- Kernel exploitation (Vulnerable drivers, Write-What-Where)



# MUCHAS GRACIAS

- <u>github.com/TheMalwareGuardian/Awesome-Bootkits-Rootkits-Development</u>
- <u>github.com/TheMalwareGuardian/Abismo</u>
- github.com/TheMalwareGuardian/Bentico

