Regístrate gratis en
www.eniit.es/cybersecurity-day-24/

26 y 27 de junio

Cybersecurity Day 2024

El principal evento online de Seguridad Informática
¡Explora las fronteras de la ciberseguridad!

#CybersecurityDay24

ENIIT
INNOVA IT BUSINESS SCHOOL

Campus Internacional
CIBERSEGURIDAD

ENIIT
INNOVA IT BUSINESS SCHOOL

Campus Internacional
CIBERSEGURIDAD

27 de junio, 19 h

Cybersecurity Day 2024

Alejandro Vázquez

Red Team Operator en Telefónica

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

Regístrate gratis en **www.eniit.es/cybersecurity-day-24**

Cybersecurity Day 2024

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

Descifrando el desarrollo de Bootkits y Rootkits

[in/vazquez-vazquez-alejandro]

Whoami

- **FRIKI** (**F**anático de **R**evolucionar **I**nternamente **K**ernels e **I**nicios de sistema)
- Me gusta el pulpo, de ahí los B/Rootkits
- Ciberseguridad Ofensiva en Telefónica
- Docente en Máster de Análisis de Malware

[in/vazquez-vazquez-alejandro]

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

Campus Internacional
CIBERSEGURIDAD

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

- 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

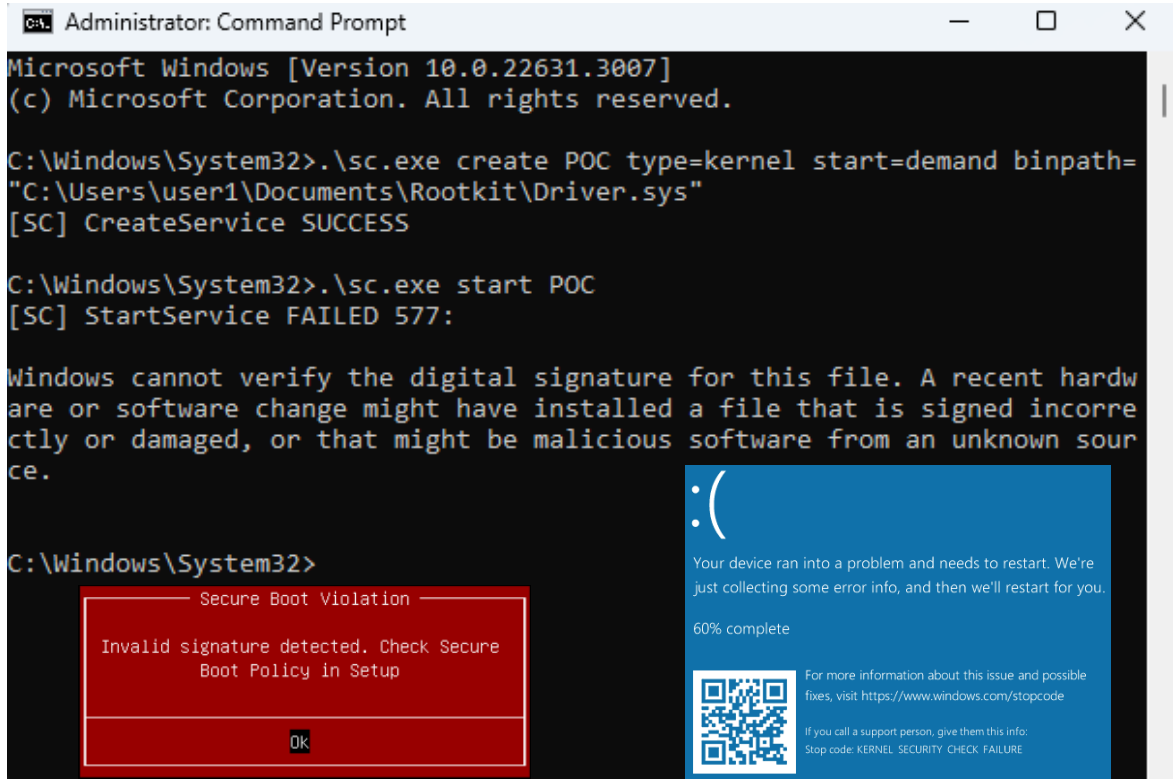
  Prevents patching the kernel

- SecureBoot

  Only software trusted by the Original Manufacturer.

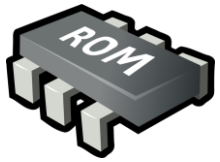
  Firmware checks the signature of UEFI firmware drivers, EFI applications and SO

- ELAM, VBS, CFG, …



Cybersecurity Day 2024

Power ON



Read Instructions

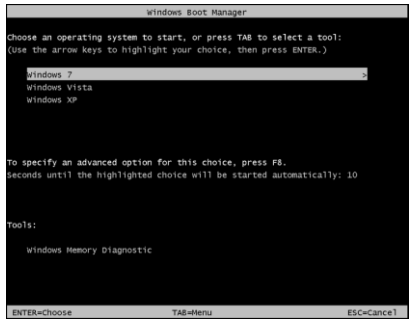
POST

UEFI Firmware

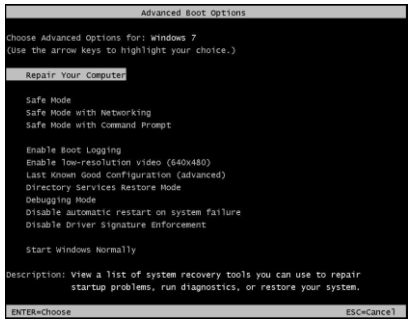
Boot Information

Boot order
Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi
Boot0002 = /EFI/Ubuntu/shimx64.efi
Boot000x = /EFI/Vendor/bootx64.efi

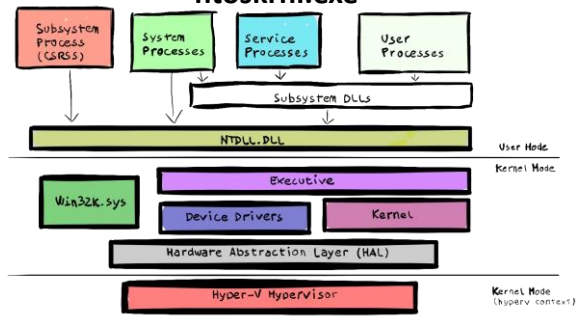
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi



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

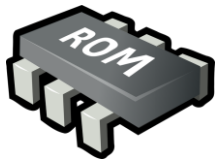


Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe



Cybersecurity Day 2024

Power ON



Read Instructions

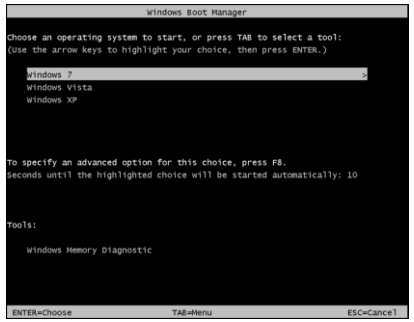
POST

UEFI Firmware

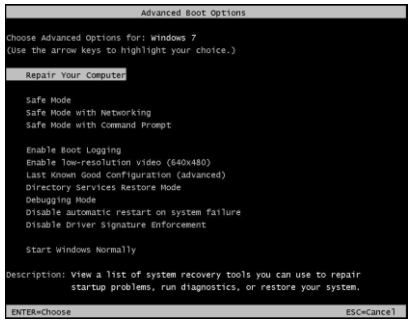
Boot Information

Boot order
Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi
Boot0002 = /EFI/Ubuntu/shimx64.efi
Boot000x = /EFI/Vendor/bootx64.efi

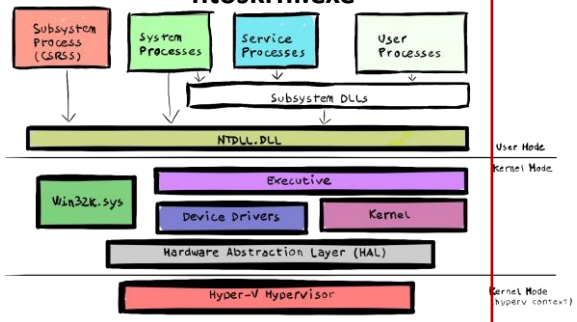
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi

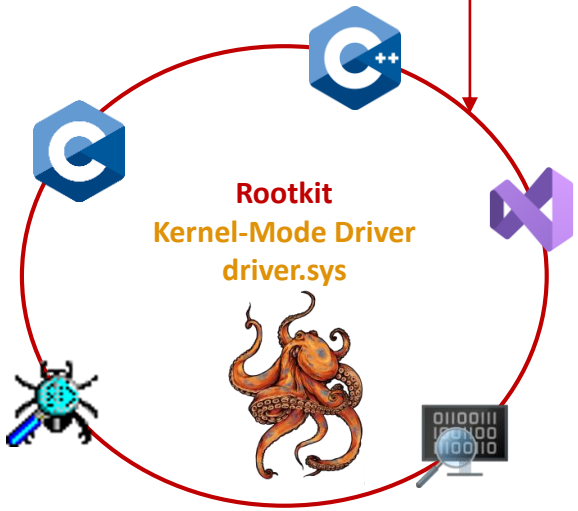


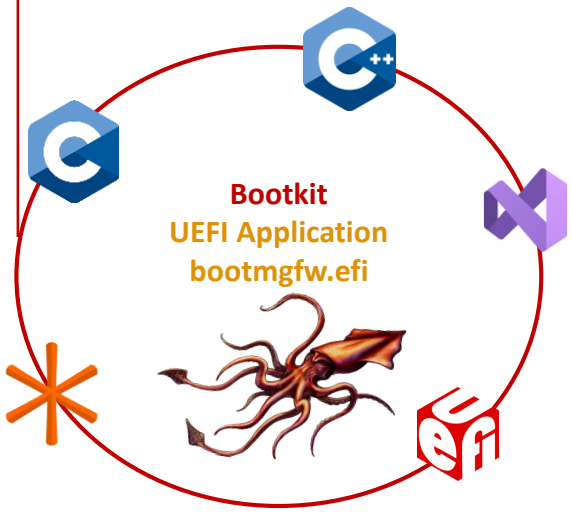
Windows OS Loader
%SystemRoot%\system32\
winload.efi



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

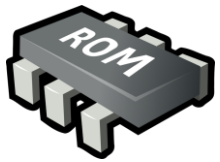






Cybersecurity Day 2024

Power ON



Read Instructions

POST

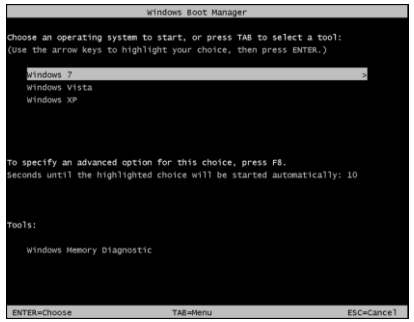
UEFI Firmware

Boot Information

Boot order
Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi
Boot0002 = /EFI/Ubuntu/shimx64.efi
Boot000x = /EFI/Vendor/bootx64.efi

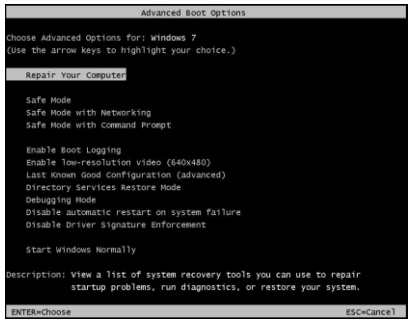
gBS->LoadImage();

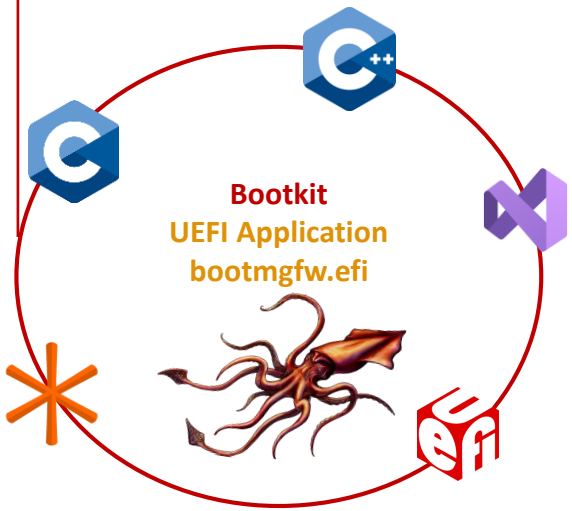
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi



ImgArchStartBootApplication();
Archpx64TransferTo64BitApplicationAsm();

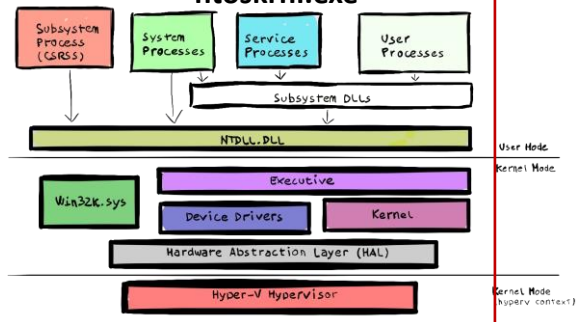
Windows OS Loader
%SystemRoot%\system32\
winload.efi

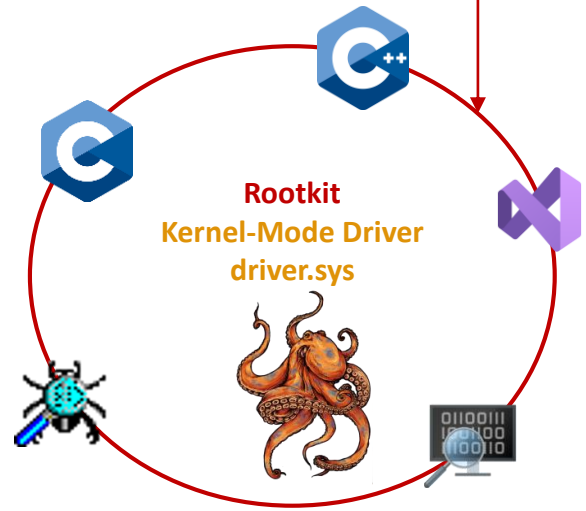




OslFwpKernelSetupPhase1();
OslArchTransferToKernel();
ExitBootServices();

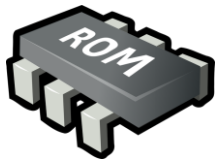
Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe





Cybersecurity Day 2024

Power ON



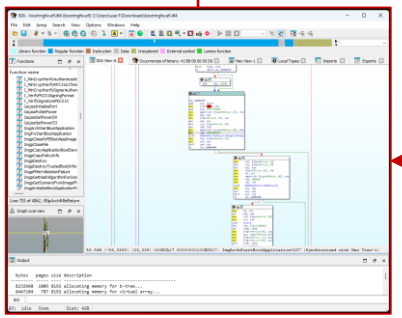
Read Instructions

POST

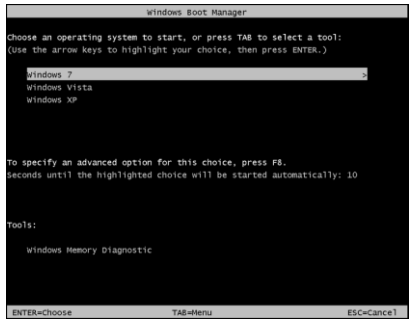
UEFI Firmware

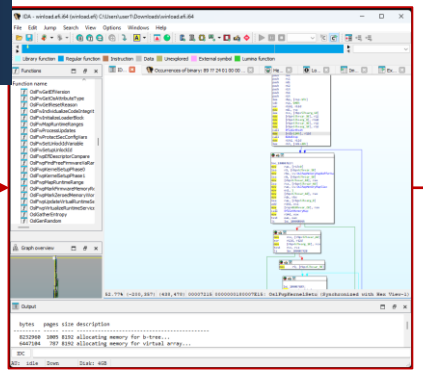
Boot Information

Boot order
Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi
Boot0002 = /EFI/Ubuntu/shimx64.efi
Boot000x = /EFI/Vendor/bootx64.efi

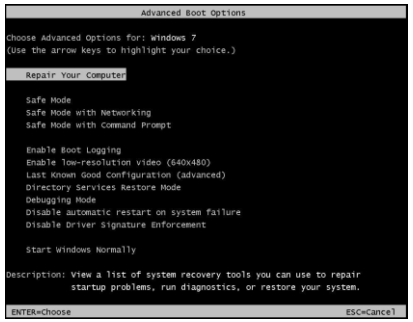


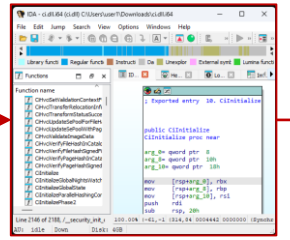
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi



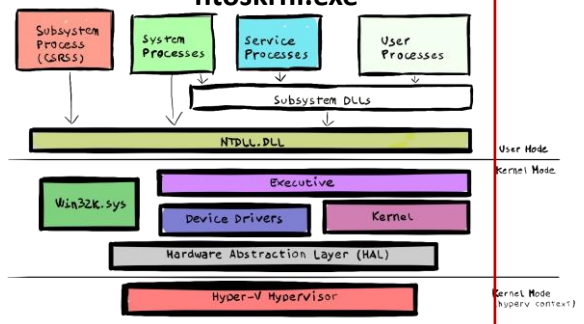


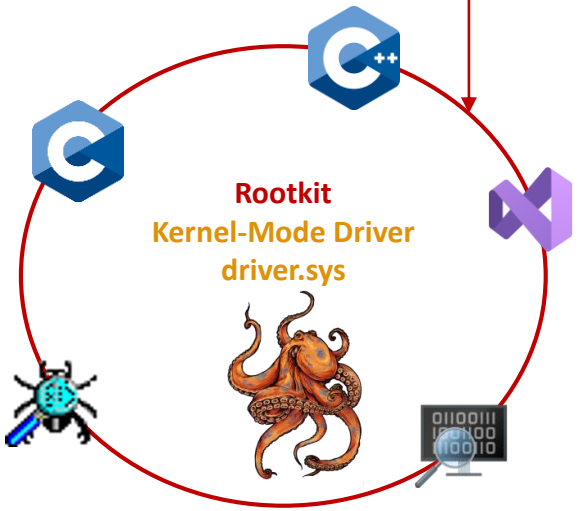
Windows OS Loader
%SystemRoot%\system32\
winload.efi

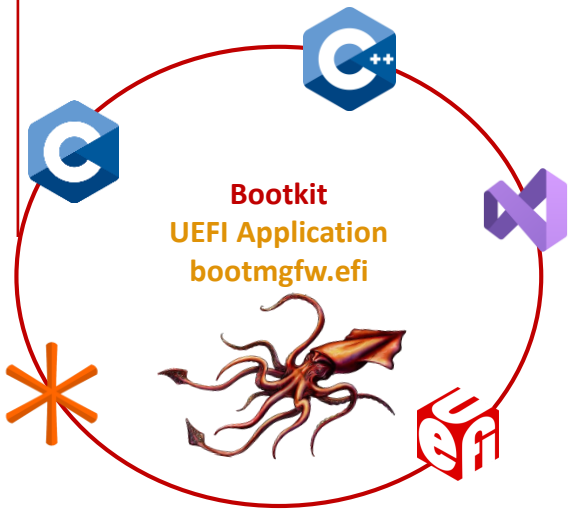




Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe

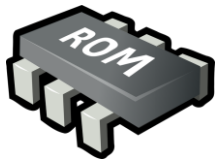






Cybersecurity Day 2024

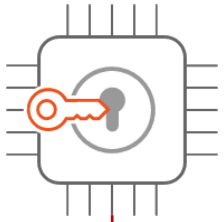
Power ON



Read Instructions

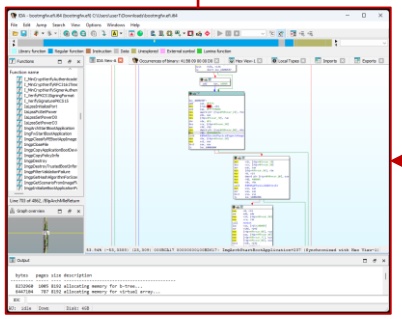
POST

UEFI Firmware

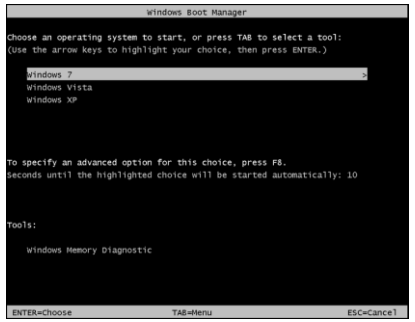


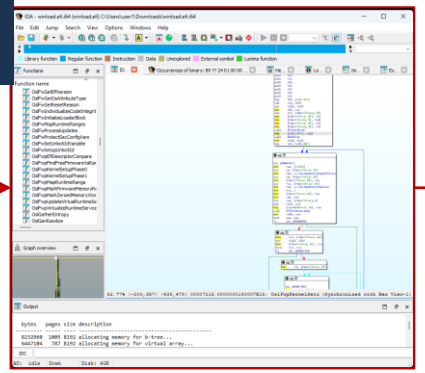
Boot Information

Boot order
Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi
Boot0002 = /EFI/Ubuntu/shimx64.efi
Boot000x = /EFI/Vendor/bootx64.efi

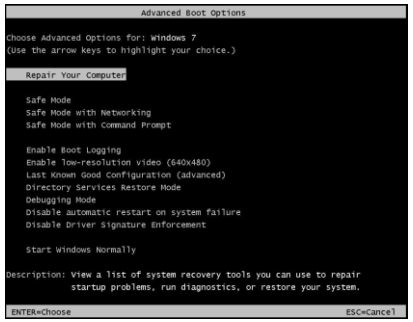


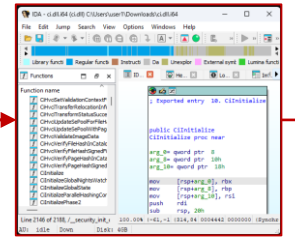
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi



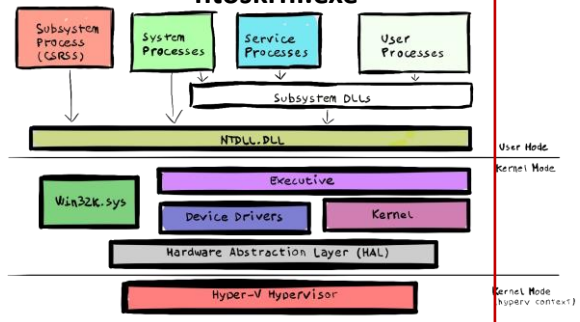


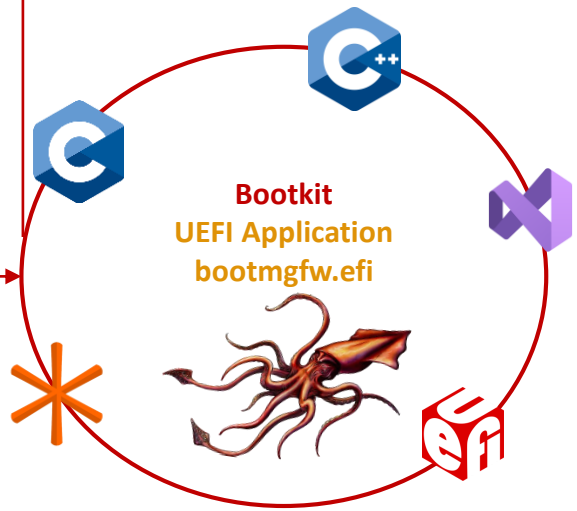
Windows OS Loader
%SystemRoot%\system32\
winload.efi

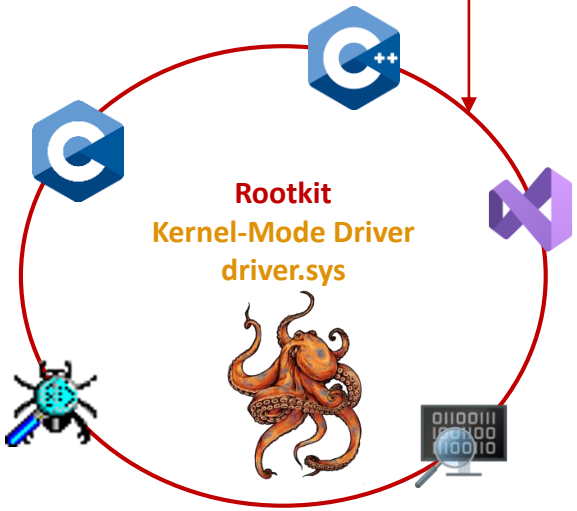




Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe







ENTORNO

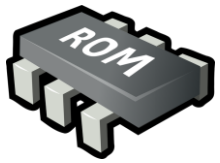
- WDK
  - Develop, test and deploy drivers for Windows.
  - Software toolset from Microsoft.

- 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

Cybersecurity Day 2024

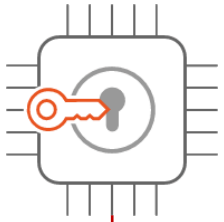
Power ON



Read Instructions

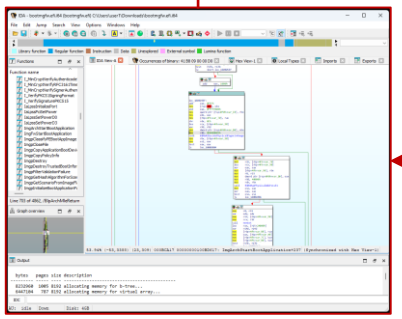
POST

UEFI Firmware

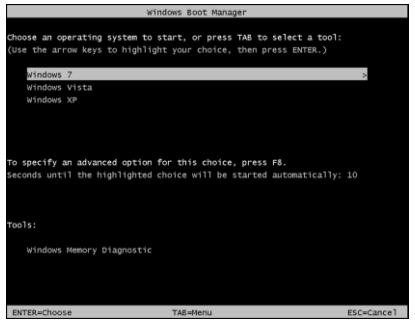


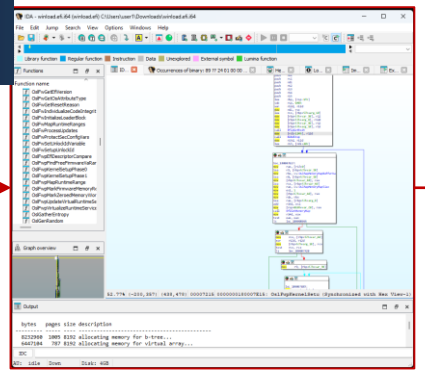
Boot Information

Boot order
Boot0001 = /EFI/Microsoft/boot/bootmgfw.efi
Boot0002 = /EFI/Ubuntu/shimx64.efi
Boot000x = /EFI/Vendor/bootx64.efi

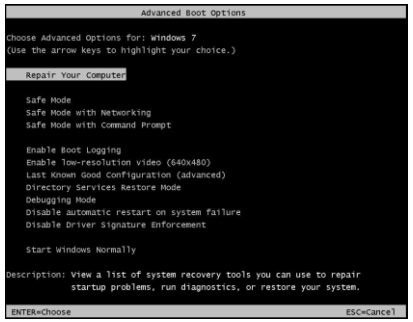


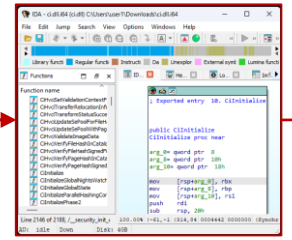
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi



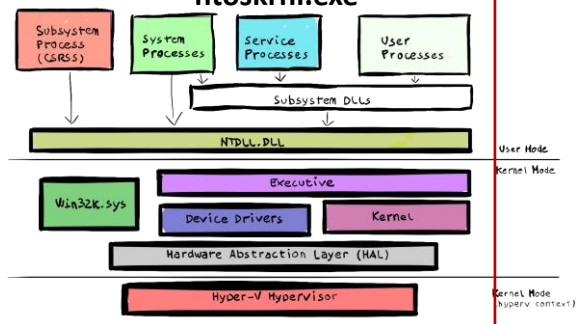


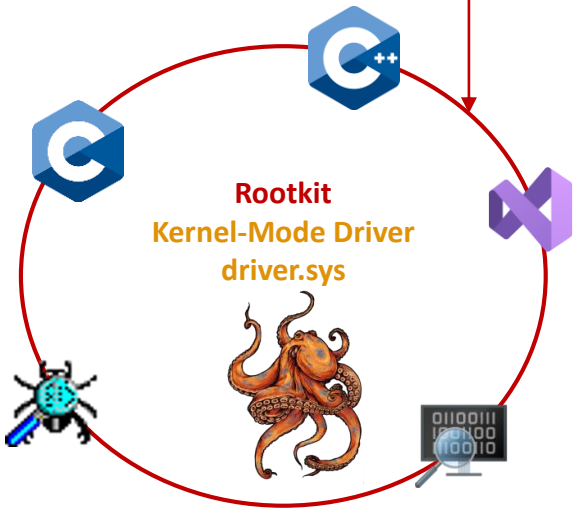
Windows OS Loader
%SystemRoot%\system32\
winload.efi

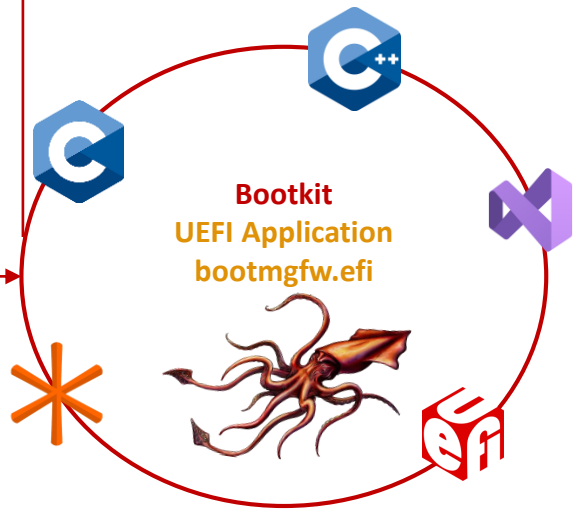


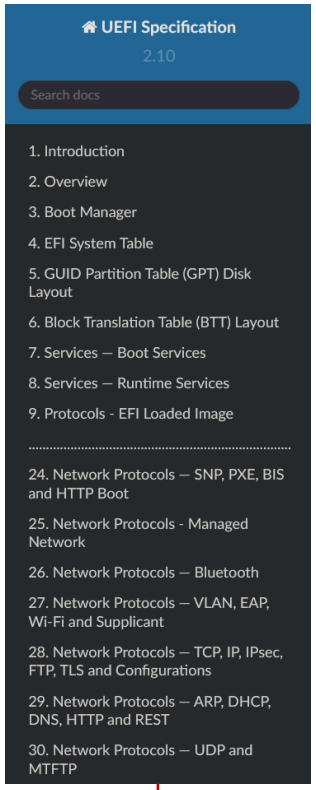


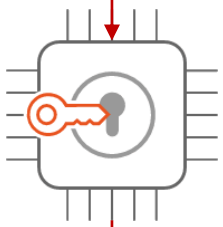
Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe

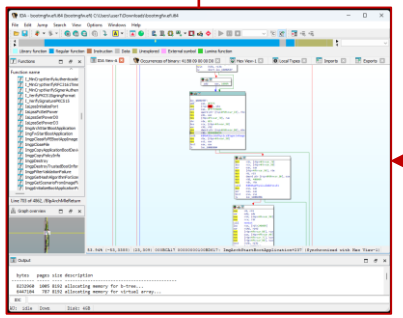




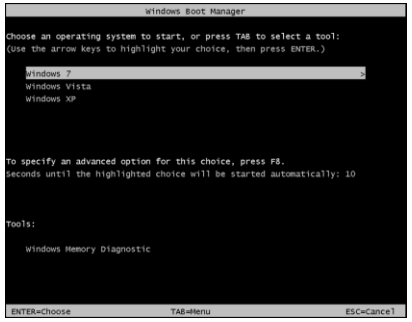


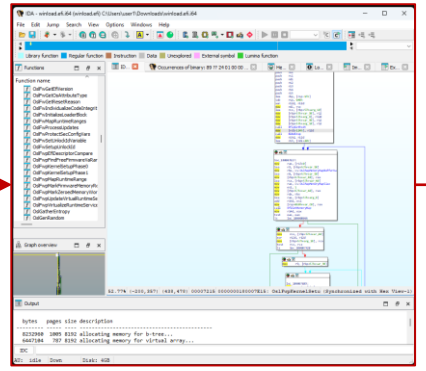




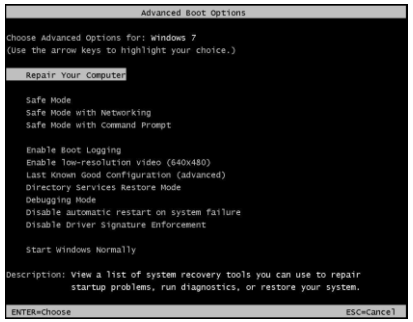


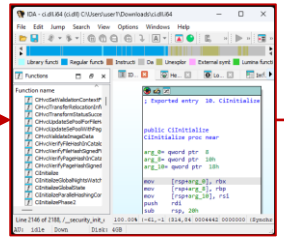
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi



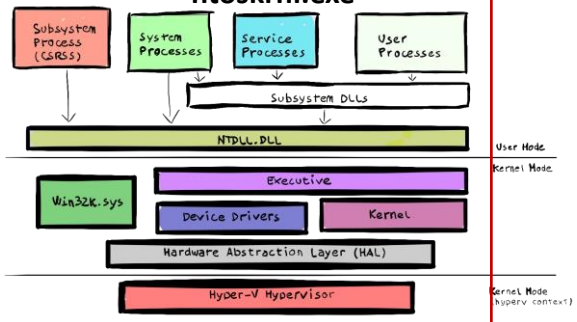


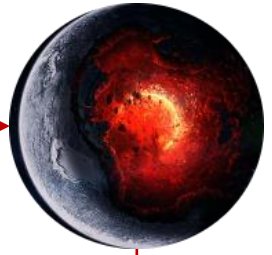
Windows OS Loader
%SystemRoot%\system32\
winload.efi

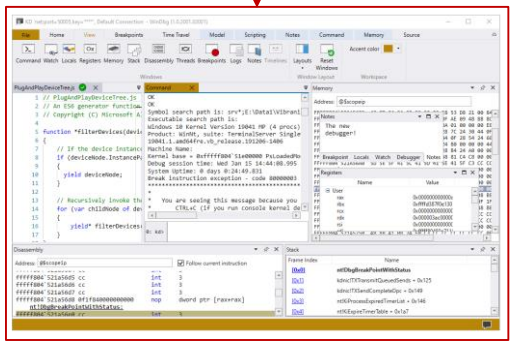


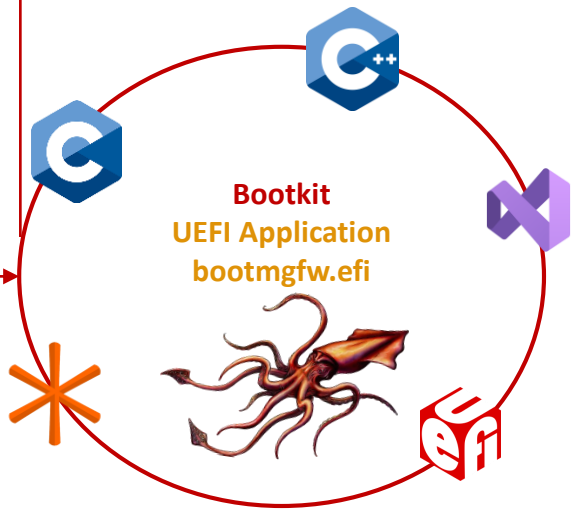


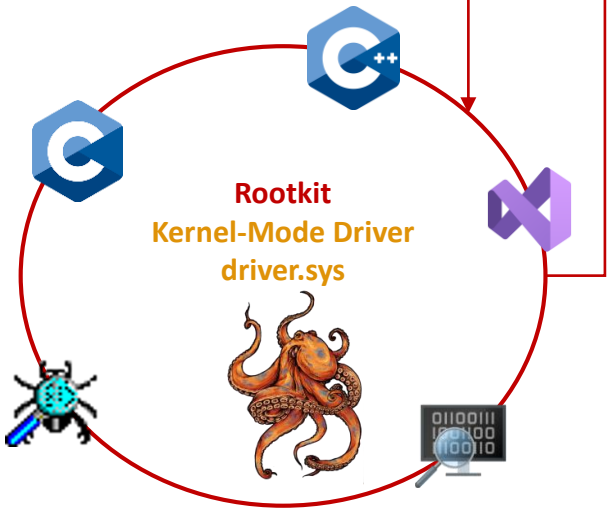
Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe



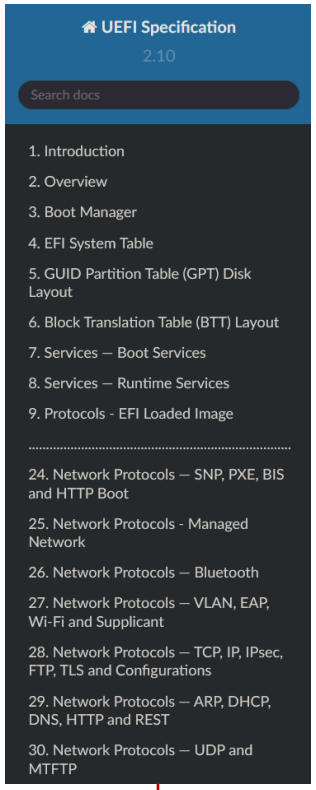


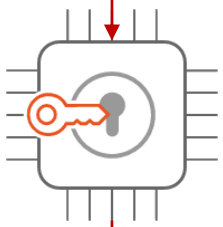


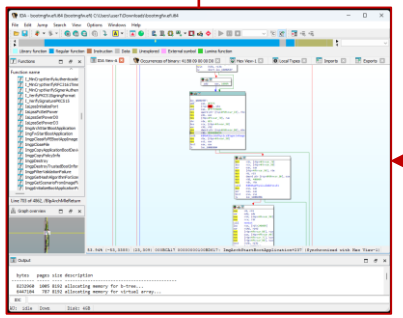




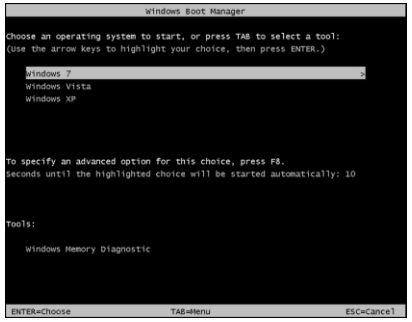

DEMO
LOADING...

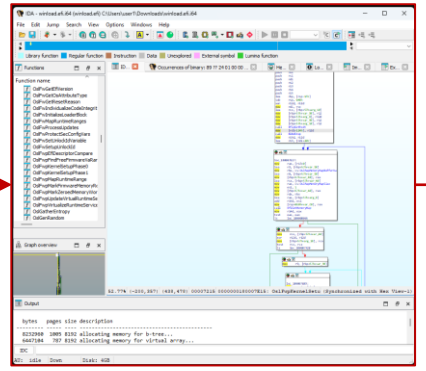




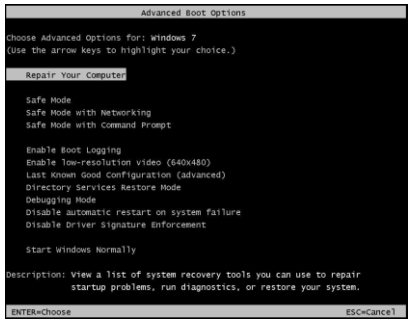


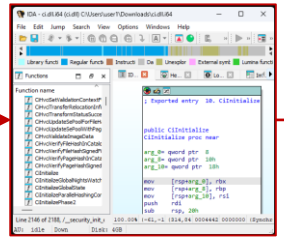
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi



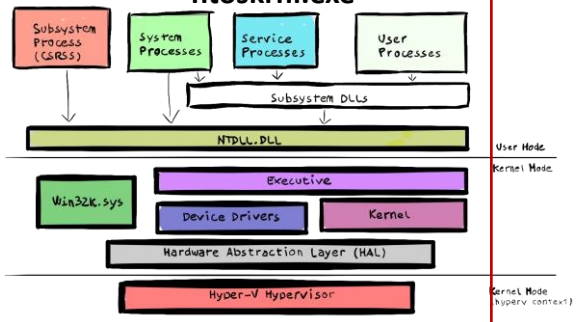


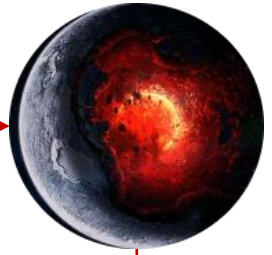
Windows OS Loader
%SystemRoot%\system32\
winload.efi

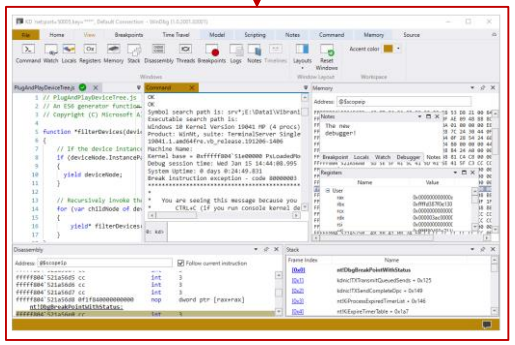


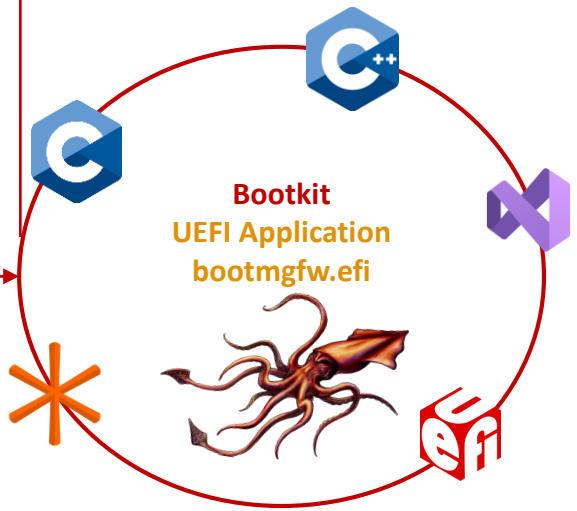


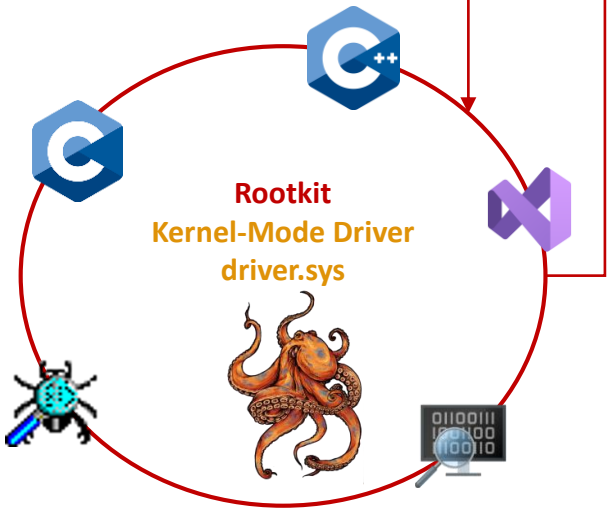
Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe





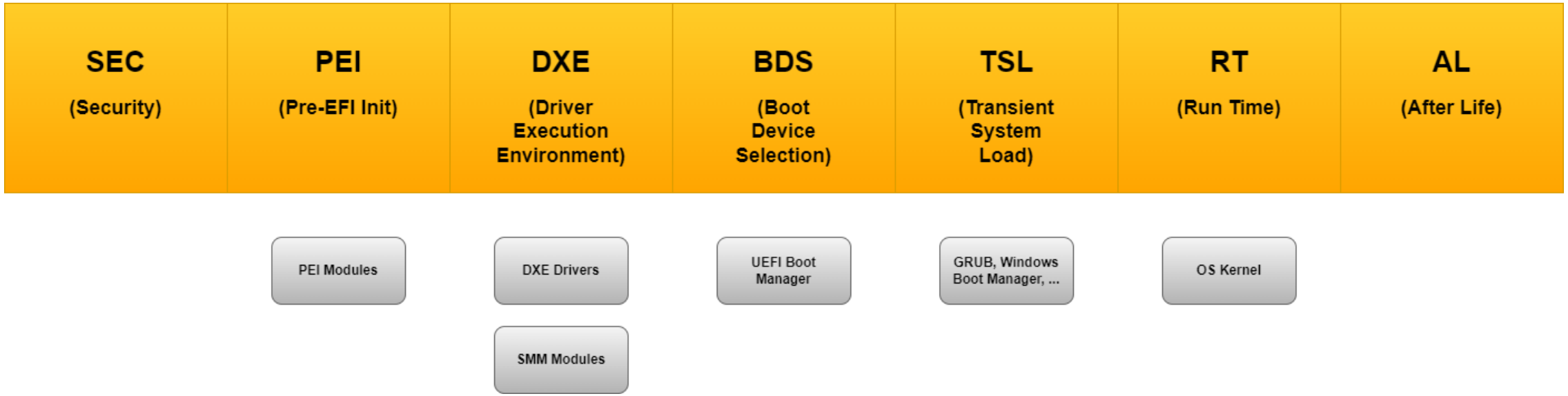






UEFI

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.

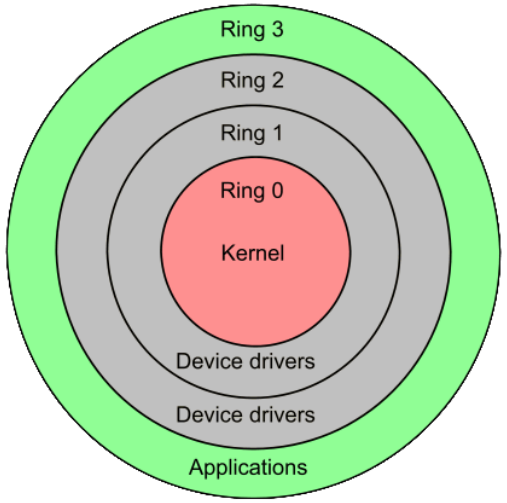


UEFI

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

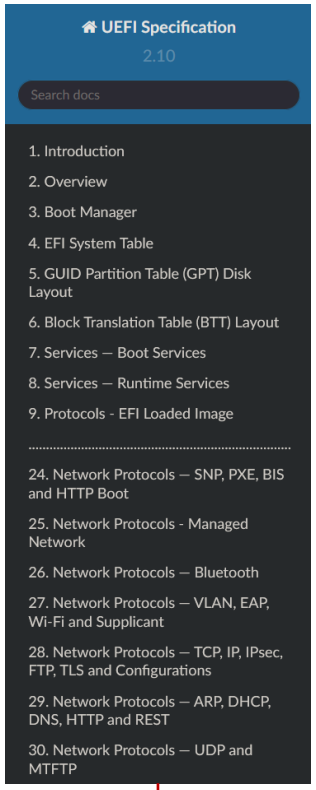


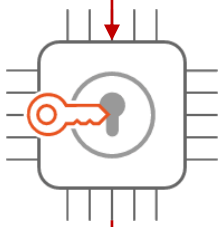
The *kernel* of an operating system implements the core functionality that everything else in the operating system depends upon. The Microsoft Windows kernel provides basic low-level operations such as scheduling threads or routing hardware interrupts. It is the heart of the operating system and all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

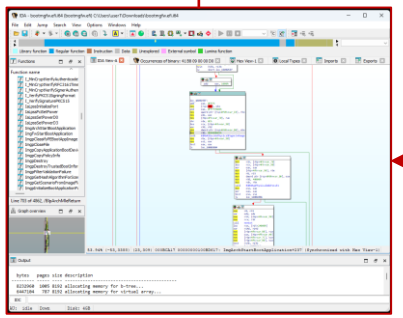
KERNEL

Kernel-Mode Driver Architecture Design Guide:

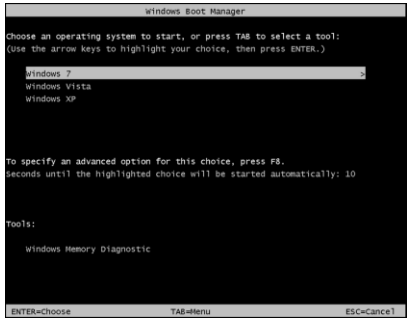
- Kernel-Mode Components: Describe the primary kernel-mode managers and components of the Windows OS (Kernel Library, Memory Manager, etc).
- Windows Driver Model (WDM): To allow driver developers to write device drivers that are source-code compatible across all Microsoft Windows operating systems, 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.

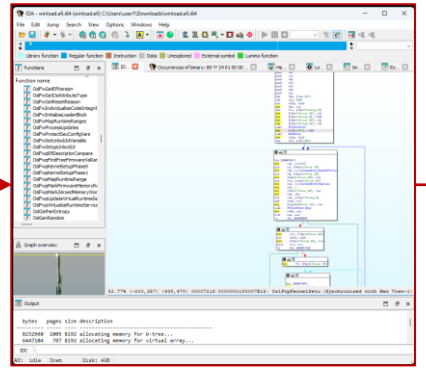




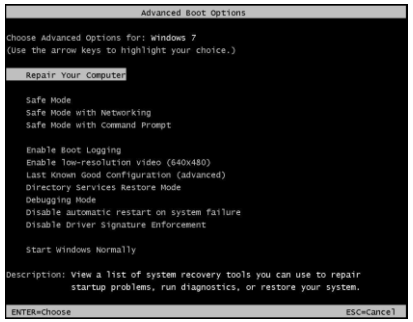


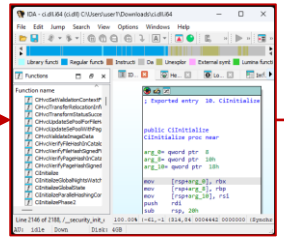
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi



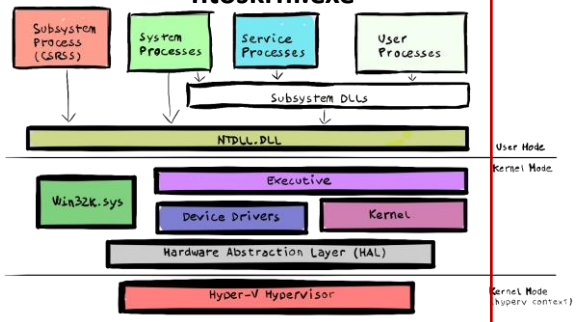


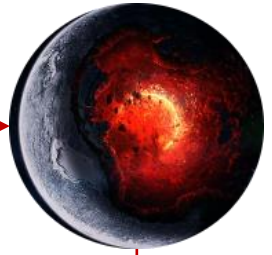
Windows OS Loader
%SystemRoot%\system32\
winload.efi

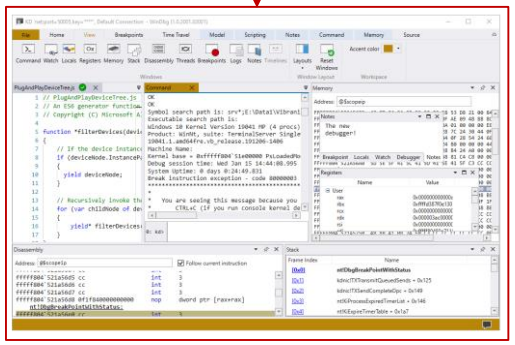


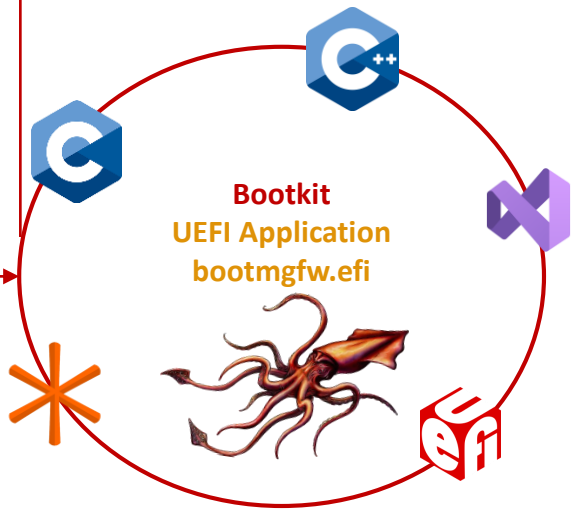


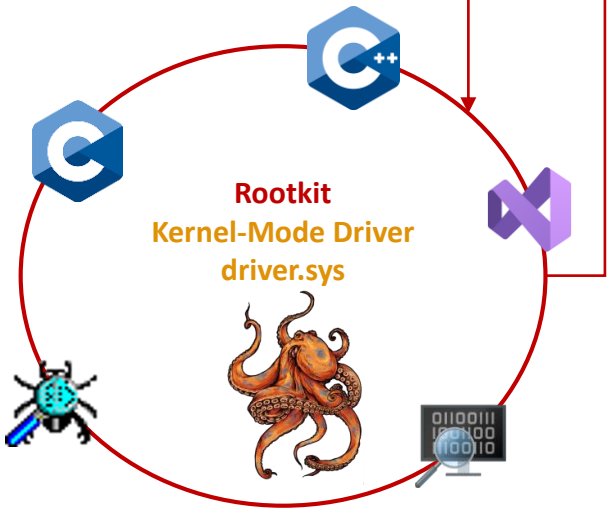
Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe

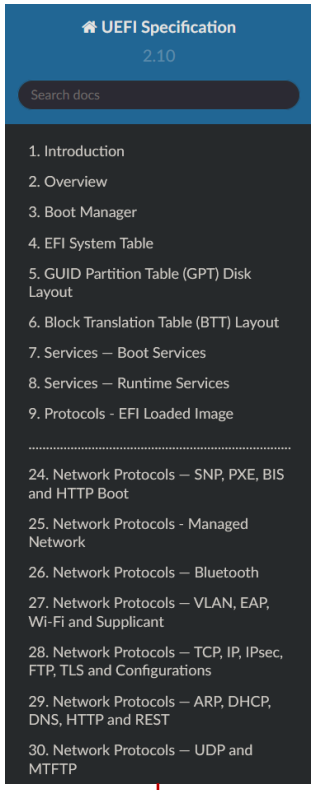


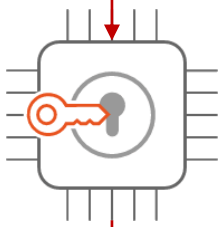


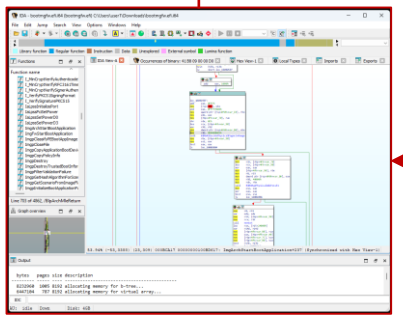




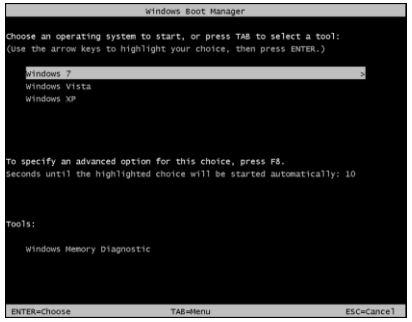


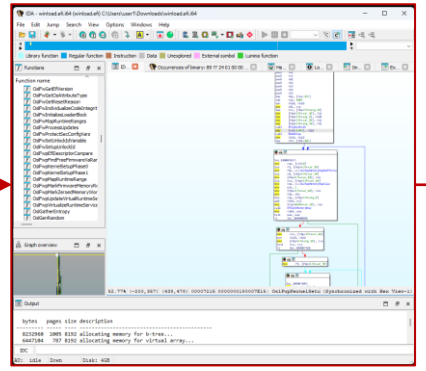




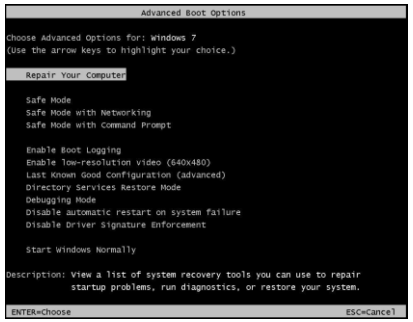


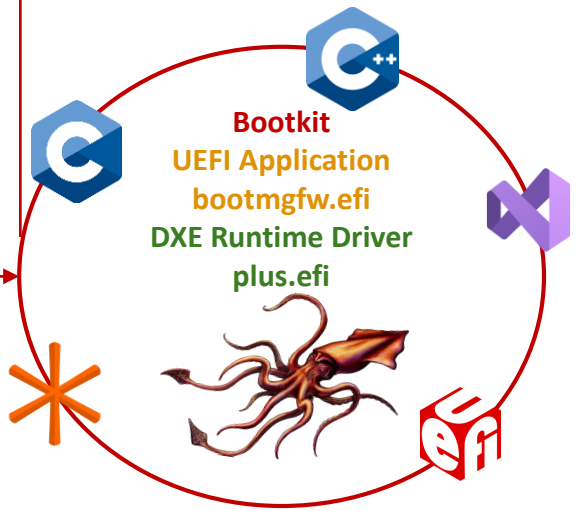
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi

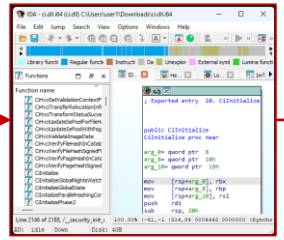




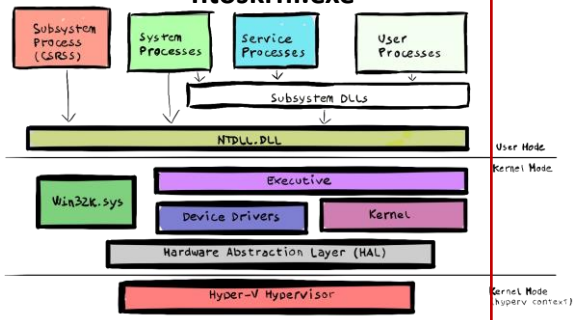
Windows OS Loader
%SystemRoot%\system32\
winload.efi

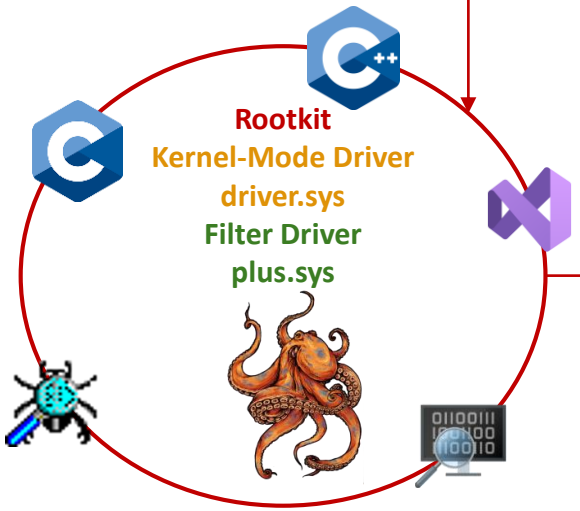


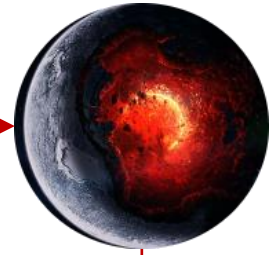


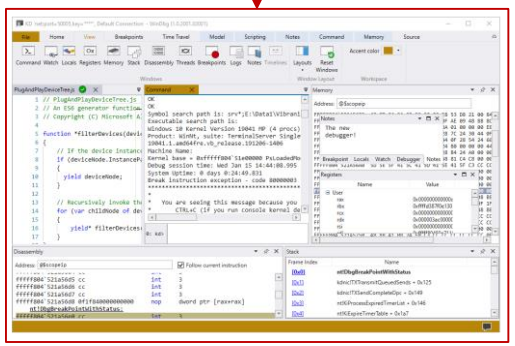


Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe







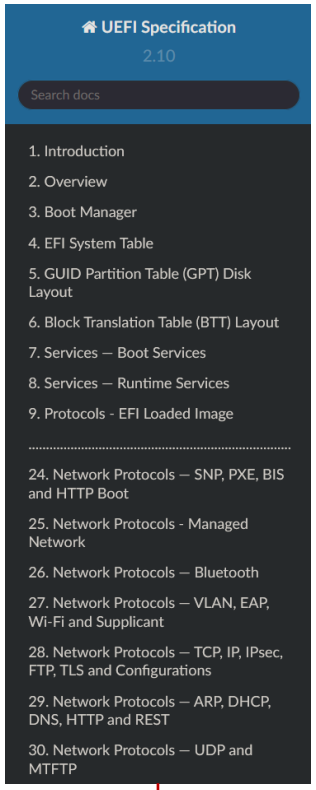


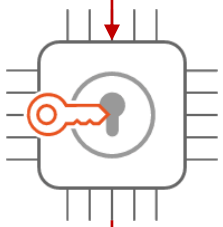
DESARROLLO

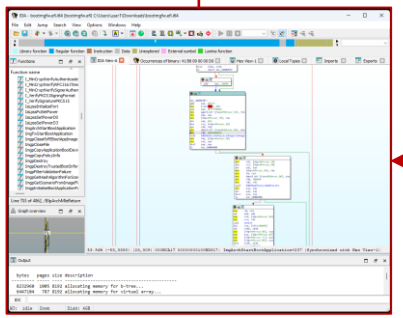
UEFI Application	DXE Runtime Driver	Kernel-Mode Driver	Filter Driver
UEFI Specification	DXE	WDF (KMDF)	WDM
Download Malware	Hooking Functions	Hide Processes	Filter Network Traffic
Boot Services	Runtime Services	Keylogger	Filter File System
C + EDK2 -> .efi	C + EDK2 -> .efi	C + WDK -> .sys	C + WDK ->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

Bootkit

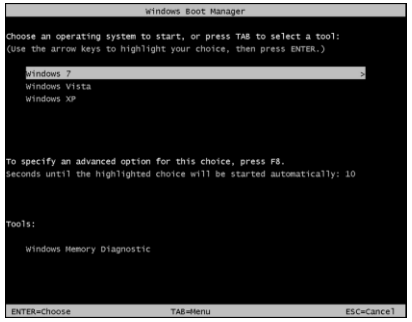
Rootkit

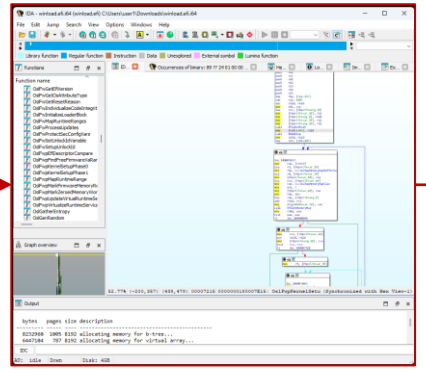




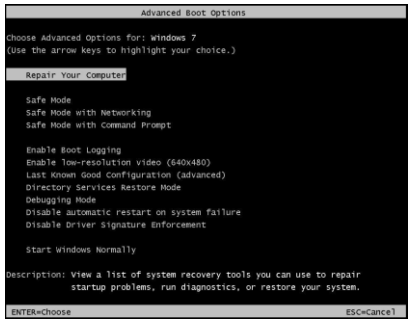


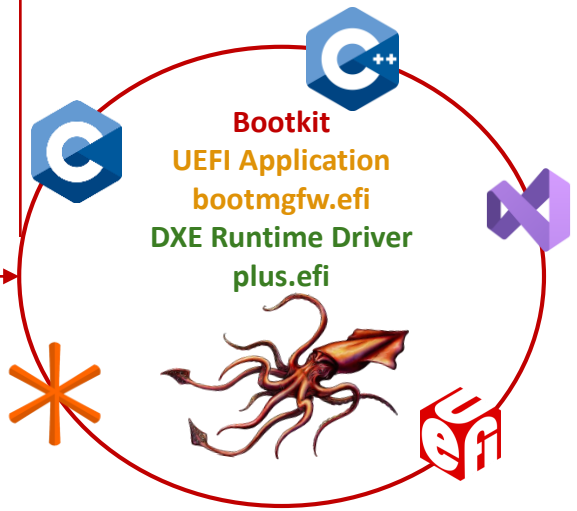
Windows Boot Manager
\EFI\Microsoft\Boot\
bootmgfw.efi

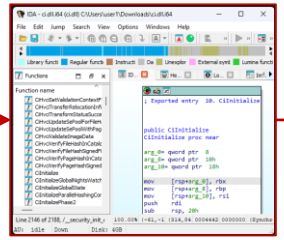




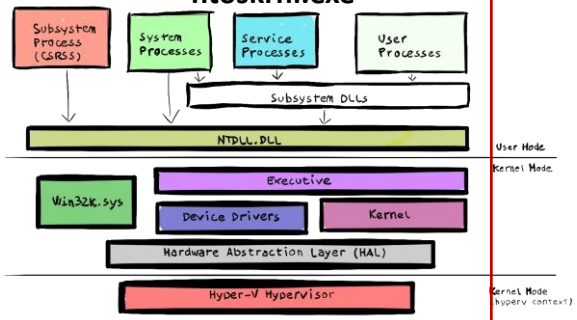
Windows OS Loader
%SystemRoot%\system32\
winload.efi

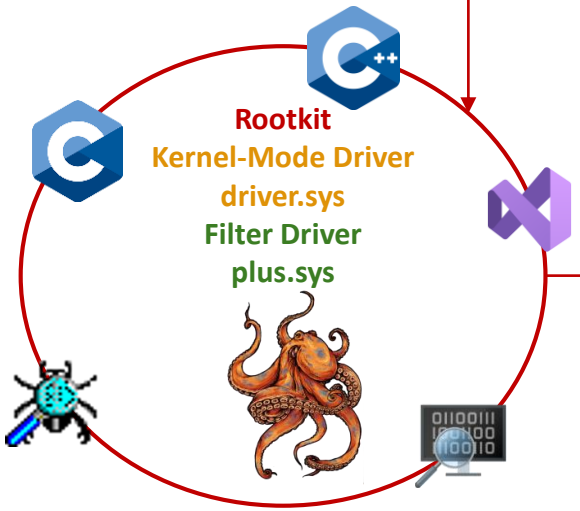


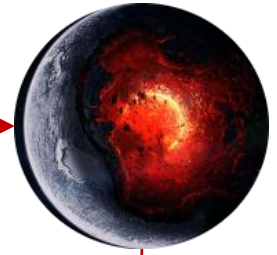


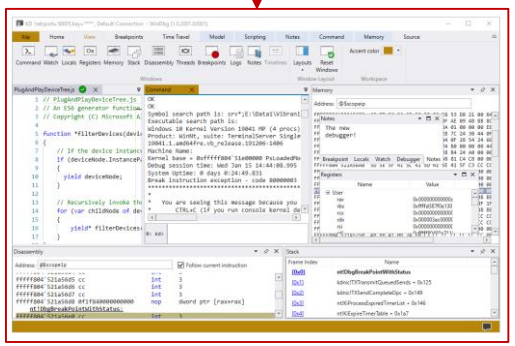


Windows NT OS Kernel
%SystemRoot%\system32\
ntoskrnl.exe









BOOTKITS IN THE WILD

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 ac

Expert GREAT

FinSpy, also known as FinFisher or Wing surveillance toolset. Kaspersky has bee 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.

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

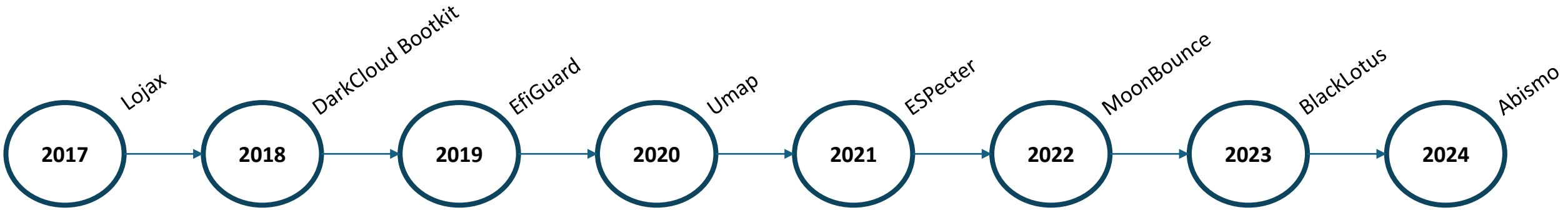
 Martin Smolár

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).

BOOTKITS 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

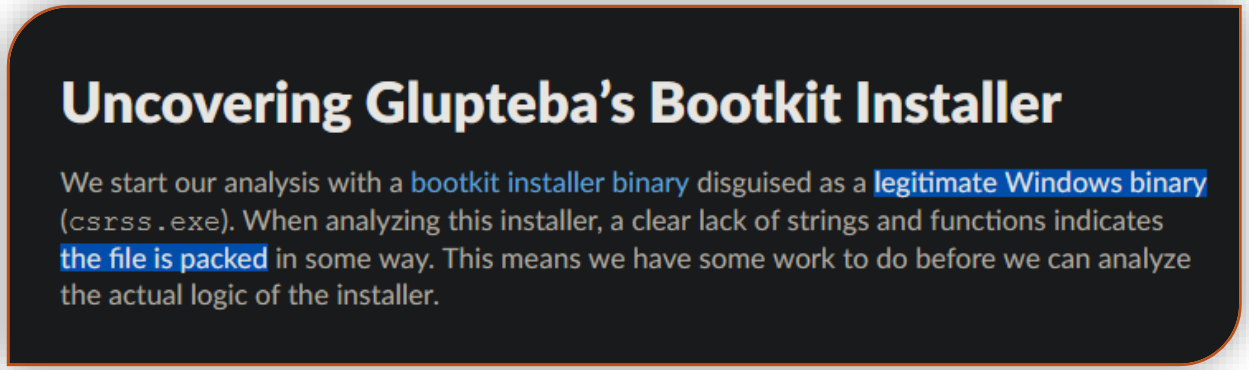
By Lior Rochberger and Dan Yashnik
February 12, 2024 at 6:00 AM
Category: Malware
Tags: Advanced Threat Prevention, Advanced URL Filtering, Advanced WildFire, Cloud-Delivered Security Services, coin miner, Cortex XDR, credential stealer, DNS security, next-generation firewall, Prisma Cloud, RedLine infostealer, 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



1. 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

FINFISHER

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.

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.

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

welivesecurity BY eset

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++ Copy

```
typedef struct _KEYBOARD_INPUT_DATA {
  USHORT UnitId;
  USHORT MakeCode;
  USHORT Flags;
  USHORT Reserved;
  ULONG  ExtraInformation;
} KEYBOARD_INPUT_DATA, *PKEYBOARD_INPUT_DATA;
```

```
#include <ntddk.h>

/**
    @brief      Function to read keystrokes from keyboard device.

    @param      pDeviceObject       Pointer to a DEVICE_OBJECT structure representing the device.
    @param      pIrp                Pointer to the I/O request packet (IRP) for reading keystrokes.

    @return     A NTSTATUS value indicating success or an error code if operation fails.
**/


NTSTATUS
DriverReadKeystrokes(
    _In_    PDEVICE_OBJECT      pDeviceObject,
    _In_    PIRP                pIrp
)
{
    IoCopyCurrentIrpStackLocationToNext(pIrp);

    IoSetCompletionRoutine(pIrp, ReadOperationFinished, NULL, TRUE, 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.

Cybersecurity Day 2024

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

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