



# An In-Depth Look at Windows Kernel Threats

**Sherif Magdy**  
Threat Researcher

**Mahmoud Zohdy**  
Incident Response Analyst



#### TREND MICRO LEGAL DISCLAIMER

The information provided herein is for general information and educational purposes only. It is not intended and should not be construed to constitute legal advice. The information contained herein may not be applicable to all situations and may not reflect the most current situation. Nothing contained herein should be relied on or acted upon without the benefit of legal advice based on the particular facts and circumstances presented and nothing herein should be construed otherwise. Trend Micro reserves the right to modify the contents of this document at any time without prior notice.

Translations of any material into other languages are intended solely as a convenience. Translation accuracy is not guaranteed nor implied. If any questions arise related to the accuracy of a translation, please refer to the original language official version of the document. Any discrepancies or differences created in the translation are not binding and have no legal effect for compliance or enforcement purposes.

Although Trend Micro uses reasonable efforts to include accurate and up-to-date information herein, Trend Micro makes no warranties or representations of any kind as to its accuracy, currency, or completeness. You agree that access to and use of and reliance on this document and the content thereof is at your own risk. Trend Micro disclaims all warranties of any kind, express or implied. Neither Trend Micro nor any party involved in creating, producing, or delivering this document shall be liable for any consequence, loss, or damage, including direct, indirect, special, consequential, loss of business profits, or special damages, whatsoever arising out of access to, use of, or inability to use, or in connection with the use of this document, or any errors or omissions in the content thereof. Use of this information constitutes acceptance for use in an "as is" condition.

Published by

**Trend Micro Research**

Written by

**Sherif Magdy**

Threat Researcher

**Mahmoud Zohdy**

Incident Response Analyst

Stock image used under license from  
Shutterstock.com

*For Raimund Genes (1963-2017)*

# Contents

**4**

The Windows Kernel Architecture

**9**

Why Attackers are Pursuing Kernel-Level Access

**13**

The State of Windows Kernel Threats

**21**

A Chronological View of Windows Kernel Threats

**40**


Are the First Cluster Threats Still Relevant?

**43**

The Second Cluster APT Case Study

**51**

Conclusion and Future Predictions



Most security products today tend to focus on threats that operate at the higher levels of the software stack. This is particularly true in user-mode applications, which has, for the most part, allowed the entire security industry to achieve reasonably good results. Unfortunately, these security products will have less visibility over the crucial parts of the system's lower levels, such as the kernel space, the boot process environment, and the firmware. If an attacker gains privileged access to the system and installs a malicious component at this level, a user's security products would not be able to detect and block the threat — especially threats that target the kernel's lower levels such as a rootkit, which is a program (or a collection of programs) that provide a stealthy environment for malware to execute within an infected system.<sup>1, 2</sup>

This research paper discusses the current state of low-level threats affecting the Windows platform. It will also show how these threats have evolved over the last seven years and where they stand in today's malware landscape. We will also show the major malware classes that we have observed based on the analyzed threat data contributed by the industry in this area.

We determined three types of low-level threats that either directly (loads a kernel driver) or indirectly (compromises the kernel by operating one level below it in the stack) affects the Windows kernel in today's threat landscape:

- Windows kernel-level rootkits – Threats that launch when the operating system is completely initialized
- Bootkits/bootloader – Threats that launch during the operating system's booting process
- Firmware/BIOS (basic input/output system) implants – Threats that launch in the pre-boot environment and during the firmware's initialization process.

Through this research, we will show the current nature of these threats, and discuss findings based on the analysis of our dataset, which consists of more than 60 of the most noteworthy lower-level threats observed in the wild since 2015. We will also discuss how advanced threat actors are adapting to the current defense mechanisms found in modern systems and how they are evolving their techniques.

# The Windows Kernel Architecture

In Windows systems, the kernel (also known as “ring 0”) has the most powerful access and privileged capabilities.<sup>3</sup> The kernel is a high-privilege level that deals with different kinds of hardware interfaces and primitive system components provided by the operating system, such as processes, threads, handles, modules, registries, and other objects.

With the level of access granted to the Windows kernel, the code that gets executed at the kernel level is capable of a wide range of monitoring and filtering mechanisms. Important system events or any data flow can be intercepted using appropriate kernel-level codes. The code execution at this level can prevent some events from happening if it is filtered by the registered kernel code and callbacks.

Aside from the core subsystems that are designed to be executed in the kernel space, the Windows kernel also provides extensibility to any third-party component that needs access to this privileged execution level. With the proper procedures, these third-party components can be plugged into the Windows kernel as a software plugin, which are also known as kernel drivers. These lower-level drivers are designed to run in ring 0 to provide both specific hardware-related and non-hardware related features.

The Windows kernel is modeled as a composite system that consists of several layers. Each layer provides services for the next layer that depends on it and assumes trust for the layers beneath it to function properly. If a single layer is compromised, all the subsequent layers within the stack will also be compromised. This is because the subsequent layers’ trust comes from the initially compromised layer that provides the execution environment and the required services for the other layers.

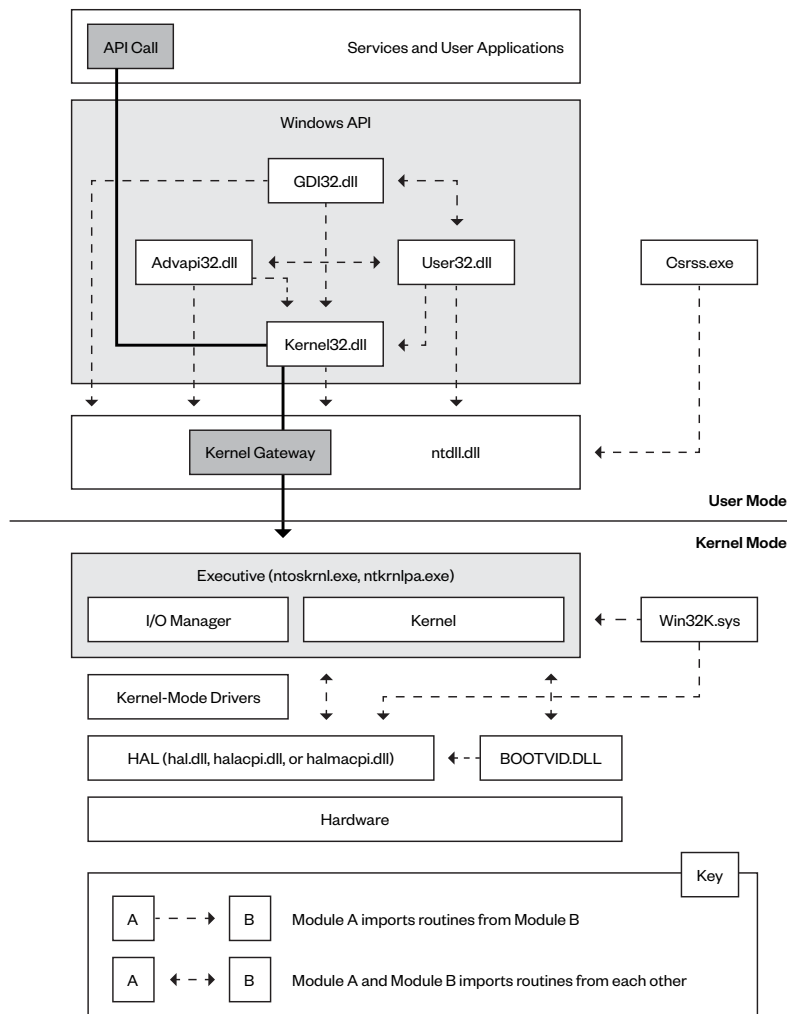


Figure 1. The composite model of Windows kernel subsystems

Source: *Windows Internals: System architecture, processes, threads, memory management, and more, Part 1 (Developer Reference) 7th Edition*

## Issues With the Windows Kernel's Architecture

The Windows kernel has a single logical address space (ring 0) and any code that executes within this space also has access to the kernel's entire physical memory. Hence, if malicious code is executed in the kernel and the kernel space is compromised, the entire system is also compromised. Traditionally, kernel code was executed with the highest level of privilege and was isolated from the regular user applications' processes. It was common to assume that the kernel space was a trusted region: Its previous architectural design was that it was required to deliver any third-party code for different system operations inside the kernel and that it was given the highest level of privilege. However, this introduced a wide attack surface for the kernel trust model and extended it significantly. To enhance this trust model, the entire Windows kernel architecture required that a new boundary be added to the kernel component itself, which we will discuss in detail in the virtualization-based security mechanisms section.

According to our telemetry data and other third-party repository data that we will show in succeeding sections, there are tens of thousands of unique kernel driver modules that are observed each day. Figure 2 shows that the number of unique kernel driver modules with revoked signatures that were submitted to a third-party malware repository has grown exponentially from 2015 to 2021. Each of these kernel driver modules could introduce a plethora of vulnerabilities inside the assumed, trusted, and guarded kernel space, on top of the other methods attackers use to get into the kernel, which we will also cover in succeeding sections of this report.

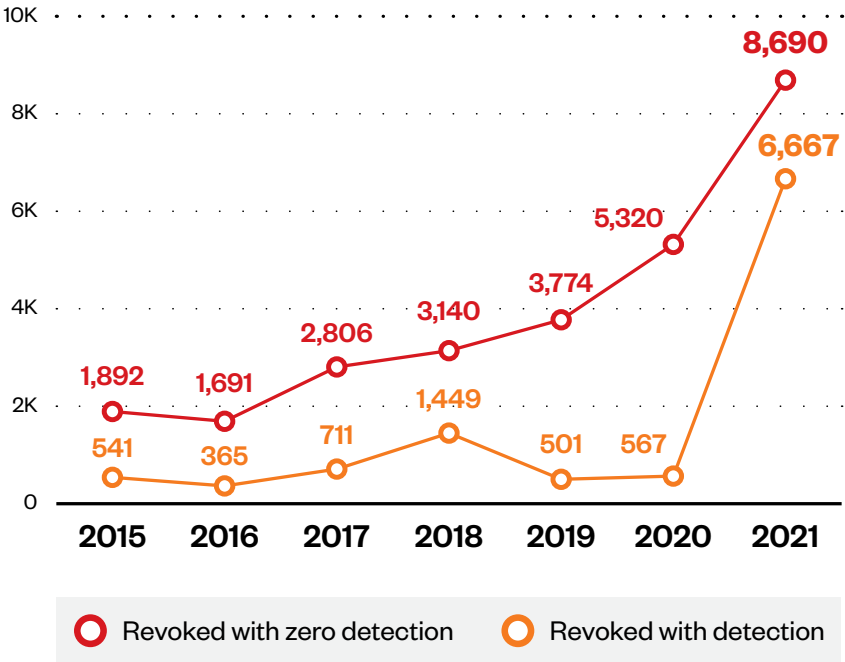


Figure 2. The number of kernel driver modules with revoked signatures that were submitted to a third-party malware repository from 2015 to 2021

Based on this data, a trust model that involves a small and well-guarded kernel space region that is completely isolated from any user-mode application tampering attempt is far from being realistic. Even with the addition of the kernel boundary, it could still be subverted by malicious code that is loaded inside the kernel, or malicious code that is made to run one layer beneath the execution environment just before a user-mode application’s complete initialization takes place. In succeeding sections, this study will show a compilation of recently observed kernel threats that abused this traditional trust model.

Another Windows kernel problem is the minimal visibility across all operating system components and the confidentiality surrounding the way in which the entire system was built. For a long time, the Windows kernel has used the security through obscurity (STO)<sup>4</sup> design pattern to maintain its security. However, it has been proven that this process favors attackers more than defenders, as attackers tend to have the time and resources to abuse undocumented interfaces and commands to launch their malicious code.

They also have the means to focus on reverse engineering system components to find the security flaws of the Windows kernel. Meanwhile, defenders usually adhere to formal documented interfaces to place their defense mechanisms.

A recent example of this pattern is evidenced by the leaked conversations of the operators of the Conti ransomware family.<sup>5</sup> These conversations showed how the Conti group was actively performing reverse engineering and fuzzing techniques to break into Intel Management Engine's firmware to gain a deep foothold into their system's software stack. This incident proves how, to this day, cybercriminals are abusing the STO design pattern to their advantage.

```
2021-06-07T18:12:59.968579 naned -> stern: Hi, things are good. I apologize for not immediately responding, I haven't communicated through a toad for a long time, I haven't seen what you wrote. Now I am finishing a full report on the mechanism of operation of the Intel ME controller and the AMT technology based on it. Recovered a bunch of undocumented commands using reverse, interface dump, and fuzzing. Unfortunately, the starting theory based on the presentation of Embedi/PositiveTechnologies reporters was not confirmed in the form in which they presented it, but there is another legal mechanism to activate AMT, but so far it has not reached the working SOFTWARE, at the moment I make a sniffer buffer that provides the HECI interface, because it is all configured in UEFI, then the sniffer took a little longer, after I fully restore the command set, the POC will be prepared. There are ideas, if we talk about the topic of uefi, then this is not just a load dropper but also perhaps some daemon of the level of SMM processors, plus since now I have tightly studied the ME controller, the idea is to test such functionality as rewriting the SPI flash drive through it. Usually this controller is allowed to write to the flash drive, which can not be said about the processor, and some commands were found that are responsible for this functionality.
```

Figure 3. Translated chat discussing the research and proof of concept (PoC) development

*Source: Eclysium.com*

## Research Scope

This research focuses on the current state of the low-level threats that are affecting the Windows kernel trust model and how these threats have evolved in the past seven years. This report also discusses the characteristics of these low-level threats, including their life cycle (from just being PoCs to their subsequent spread among actual threat actors) and their being used as part of sophisticated advanced persistent threat (APT) groups' infection chains.

More than anything else, when it comes to these low-level attacks, the threat landscape is greatly affected by the current native defense mechanisms provided by the operating system. The current innovative defenses that make use of virtualization extensions in modern CPUs are greatly reshaping how these threats are created and carried out. With the introduction of every new defense mechanism, malware actors are forced to adapt to the different abstraction layers and shift focus between the kernel and the boot process to get closer to the firmware and hardware layers. It is, therefore, important to discuss the history of these threats, what the security community is currently observing when it comes to security, and the trajectory in which this type of threat is heading toward in the future as defensive solutions shipped with modern versions of Windows systems continuously evolve.

This research covers more than 60 different low-level threats related to the Windows kernel. The samples we analyzed were clustered into several groups to identify the main patterns across different threat types, discover the types of malicious actors that were pursuing this kernel-level capabilities, determine the reasons behind seeking such privileged access levels, get insight on possible development costs, and shine a light on the potential trade-offs for adding a kernel-level capability to cybercriminals' malware arsenal.

We also covered the current observed capabilities in recently found rootkit samples and provide a statistical analysis of the current detection rate of these modules based on our telemetry. Finally, we detail what we discovered in underground marketplaces that malicious actors can use or outsource to abuse the Windows kernel. This research focuses on the threats that are mainly executing in or below the kernel space, or threats that have at least a single component that runs in the Windows kernel.



# Why Attackers are Pursuing Kernel-Level Access

Malicious actors gain many advantages once they are able to execute code at the kernel level of targeted systems, including the ability to impair their victims' security defenses and remain undetected, especially since rootkits allow malicious actors' implants to evade detection for a longer period.

The following section describes why attackers are still seeking such low-level capability for their malware families, despite new defense mechanisms that are supposed to protect this execution level; If having this capability ends up being optimal for the malicious actors' attack scenarios and objectives, it will likely be a complicated route to take. Finally, we discuss the cost that comes with adding kernel-level malware families to an attacker's malware arsenal.

## Pros

The most obvious use cases that justify the high development costs for kernel-level rootkits and other low-level attacks are as follows:

- Gaining very high-privileged access to system resources
- Hiding malicious activity on devices and making detection and response activities more difficult
- Protecting malicious artifacts from normal system filtering processes
- Executing stealth operations that can bypass detection for extended periods
- Gaining inherited trust from third-party antivirus products
- Tampering with core services' data flow on which multiple user-mode applications depend
- Tampering with third-party security products that hinder malicious activity
- Achieving a very low detection rate. According to intelligence reports, most modern rootkits remain undetected for a long period

The difficulty of detecting these threats lies in third-party security products' limited visibility over the operations performed by most of kernel modules running inside the kernel boundary. These malicious

kernel drivers are often ignored because they have the same privilege level as that of security product drivers. Malicious kernel drivers have a significant inherited trust compared to user-mode applications, which is why kernel rootkits are able to evade security controls and tools. Moreover, there are less mitigation techniques and security solutions that target malware types that are deployed as standalone kernel components. Figure 4 shows an example of how endpoint security solutions give implicit trust to kernel drivers within registered PreProcessThread callbacks.

- Exclude (PID < 8 and OperationInformation.KernelHandle == 1).
- Include (! OperationInformation.KernelHandle == 1 and ExGetPreviousMode()).

```
int __stdcall OnPreOpenProcess(int a1, _OB_PRE_OPERATION_INFORMATION *a2)
{
    char v3[4]; // [esp+4h] [ebp-10h]
    HANDLE v4; // [esp+8h] [ebp-Ch]
    PEPROCESS Process; // [esp+Ch] [ebp-8h]
    BOOL Is_System_Process; // [esp+10h] [ebp-4h]

    Is_System_Process = (unsigned int)PsGetCurrentProcessId() < 8;
    if ( Is_System_Process & a2->KernelHandle & 1 )
        return 0; // Return OB_PREOP_SUCCESS if operation is coming from the Kernel
    Process = (PEPROCESS)a2->Object;
```

Figure 4. Antivirus solutions tend to treat kernel drivers with significant trust compared to user-mode applications

## Cons

Using kernel rootkits in attacks also come with the following cons:

- Kernel rootkits are more difficult to develop and implement compared to other user-mode application malware types, which does not make them the ideal threat for most attacks.
  - The development of kernel rootkits involves highly qualified kernel-mode developers who understand the targeted operating system's internal components and have a sufficient level of competence when it comes to reverse engineering system components.
  - Since kernel rootkits are more sensitive to errors, they might reveal the whole operation if it crashed the system and triggered BSOD because of code bugs in the kernel module. Any errors in the source code of a kernel-mode rootkit will cause irreparable changes that will inevitably affect the system's stability.
- If the victim's security mechanisms are already ineffective or can be taken down via a simpler technique, introducing a kernel-mode component will complicate the attack more than support it. If a point of entry to the targeted entity is found, and malicious actors observe that that the perimeter has a weak protection and that there are significant security system flaws, it is irrational to use a kernel-level rootkit in the attack.

- Kernel-level rootkits take a lot of time to develop and test. It is a malware type that better suits APT actors' motives as opposed to other cybercrime actors who are used to abusing newly discovered exploits and attempting to make use of threats to infiltrate a victim's network with reliable and ready-to-use tools.

## The Trade-Offs

Attackers first assess the requirements before opting for sophisticated techniques such as building a kernel rootkit or even deploying an implant in a low-level layer before kernel initialization. These design choices are governed by several parameters, which include determining the number of detection vectors that a particular method will be exposed to, understanding how usable it is, and finding out how expensive it would be to have a detection for it. Malicious actors are always seeking for undocumented, inexpensive, hard-to-detect, and stable methods for their attacks.

Malicious actors are switching back and forth between different abstraction layers and system boundaries depending on the current adoption rate of new defense mechanisms such as virtualization-based security (VBS) and Hypervisor-Protected Code Integrity (HVCI).<sup>6</sup> The current platform's security mechanisms will define the most suitable execution level in which malicious actors will operate without affecting the stability and usability of their attacks' capabilities. An example of a trade-off is the kernel-mode code signing (KMCS) bypass that might reduce the system's stability or might not be usable on all versions of Windows systems.

Figure 5 shows how the initial infection point is changing in today's threats. It also shows how some threats have shifted from operating in one layer to another based on the advantages and disadvantages of each layer and the defenses deployed for each one. Once a new protection is introduced, the attackers will choose the path of least resistance and opt for the next layer in the software stack. For example, in the historical data we have analyzed, the malicious actors behind the Moriya rootkit<sup>7</sup> was observed to have shifted from operating in a user-mode execution level (IISpy) to being a kernel-mode component. The same malicious actor used a kernel-mode component to have a more powerful capability given the required stealth objectives and the available resources to bypass KMCS. The TDL3 rootkit also evolved from the kernel-level and emerged as a bootkit (TDL4) operating in the next layer. Finally, the ZeroAccess rootkit<sup>8</sup> became a widespread commodity malware that shifted from kernel-mode to user-mode when KMCS was introduced.

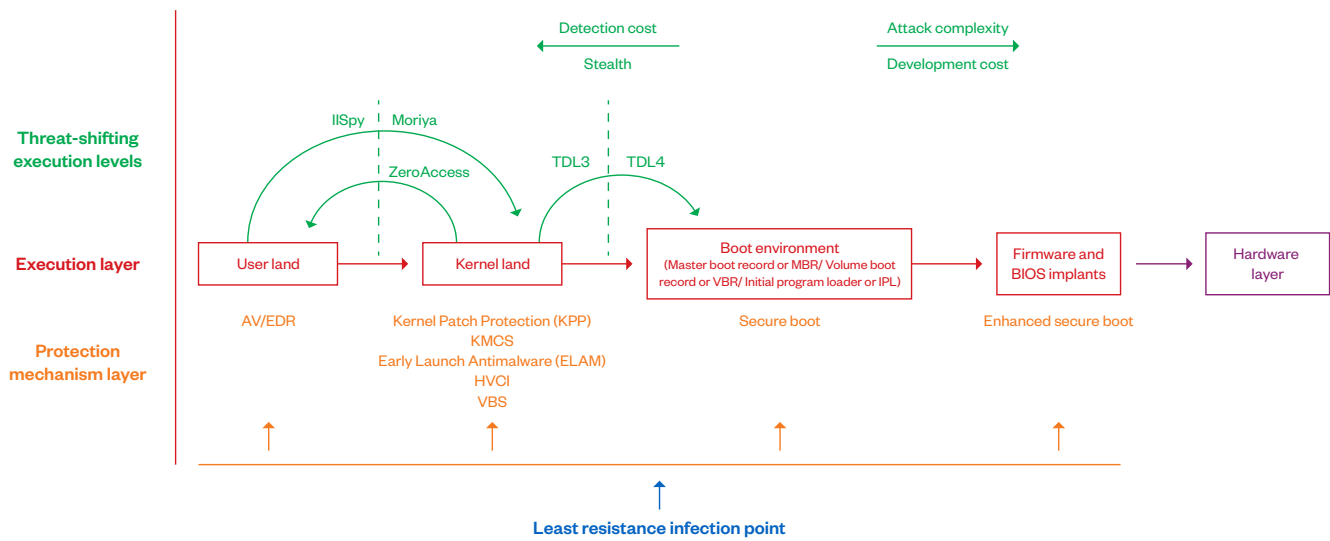


Figure 5. The dynamics of the initial infection point affected by the current platform security mechanisms

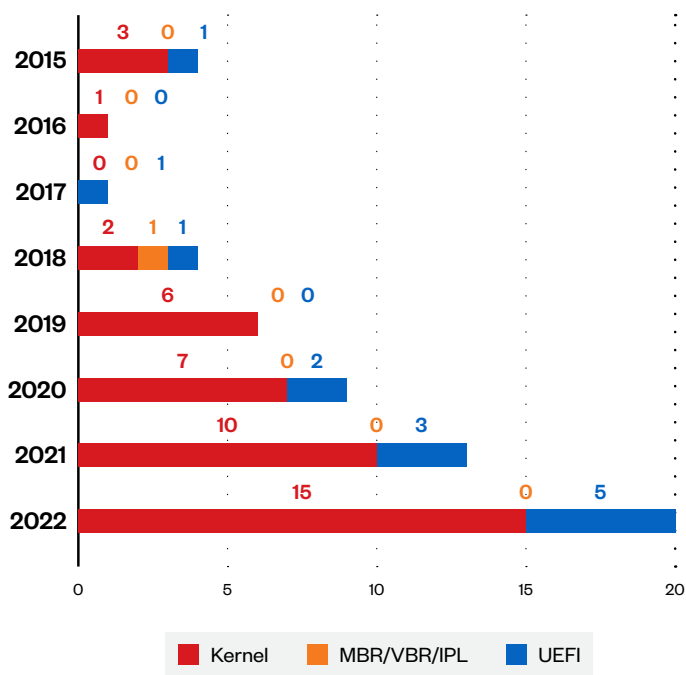


Figure 6. The number of attacks that use the Unified Extensible Firmware Interface (UEFI) as an initial infection point with kernel-level hardening over time

# The State of Windows Kernel Threats

In previous years, Microsoft has focused on adding several security mechanisms to improve the overall security posture of the kernel root of trust. This is because they were aware that if the kernel root of trust was compromised, it would immensely affect the stability and the confidentiality of the entire system. In this section, we will review the events that affected the Windows kernel trust model, the corresponding threats launched against it, the reasoning behind each security mechanism, and how attackers created their kernel-level malware arsenal and adapted to security defenses introduced in newer Windows versions.

## Pre-KMCS Era (Before Windows Vista 64-bit)

During this period, most malicious actors — regardless of motive or technical capability — were able to easily compile and load kernel-mode modules as part of their attack chains. This was because the cost of getting kernel-level execution power was significantly low. The kernel space boundary was open to anyone who could compile a Windows kernel driver. This allowed attackers to maintain a deep foothold in the kernel and do all kinds of tricks: subvert and manipulate system services, hook all kind of critical kernel objects, interfere with third-party antivirus engines, and corrupt system dispatch tables. It was obvious that third-party solutions were losing this arms race as it was difficult, and sometimes even impossible, to protect code that ran at the same level as their protection engines.

The kernel security posture at this period was chaotic because of all the untested and low-quality kernel code written by some Windows kernel developers. There was no way to trace the code to the origin entity that wrote the faulty code that caused a crash dump. Kernel rootkits also used to be implanted in all intrusions. The security industry reacted with the creation of several anti-rootkit software that featured complex heuristic scanning capabilities, the ability to analyze vague kernel data structures, and the ability to attempt restoring it upon a damage is detected. The consequence was a huge instability in the whole system, where the blue screen of death (BSOD) was very common. At this point, the root cause of kernel instability was the unverified kernel-level code shipped in the form of kernel drivers.

# KMCS Era and Beyond (From Windows Vista 64-bit Onwards)

Microsoft saw the need to control the quality of its kernel-code modules, validate the origin of their code with their respective authors, and add more restrictions to code that is executed with elevated access in 2006. After Windows developers assessed the damage caused by rootkits and other low-level malware types and the extent of their distribution, the company announced that KMCS would be shipped with Windows Vista 64-bit version systems. Alongside KMCS, enhanced kernel patch-protection mechanisms such as Microsoft KPP or PatchGuard that maintained the integrity of several core kernel objects and dispatched tables to hinder tampering by malicious code were also included in the Windows Vista systems released during this period.

This move by Microsoft made designing an attack that included a kernel module, particularly for kernel drivers that mainly depended on commodity malware, costlier for mid-level attackers. Also, the Microsoft PatchGuard security feature affected the sheer volume of techniques used by rootkit authors including Direct Kernel Object Manipulation (DKOM) and inline hooking for core system service tables. This rendered a whole class of attacks unusable and effectively limited rootkit authors' operations, even if they already succeeded in delivering their code into the Windows kernel.

These enhancements obligated hardware and software vendors to digitally sign their kernel drivers to control the quality of the code that runs in the address space of the kernel space. This meant that malware authors needed to have valid signatures to get their drivers loaded.

The rootkit threat landscape changed dramatically during this era as evidenced by the following points:

- There was a higher development cost for kernel rootkits to be loaded
- There was not as many malicious actors who had the technical capabilities to include kernel rootkits in their malware arsenal
- There were new techniques that malicious actors needed to do to bypass the security defenses Microsoft added to newer Windows versions

Figure 6 illustrates the state of kernel threats before the introduction of KMCS and after the first iteration of security enhancements were introduced to improve the kernel trust model.

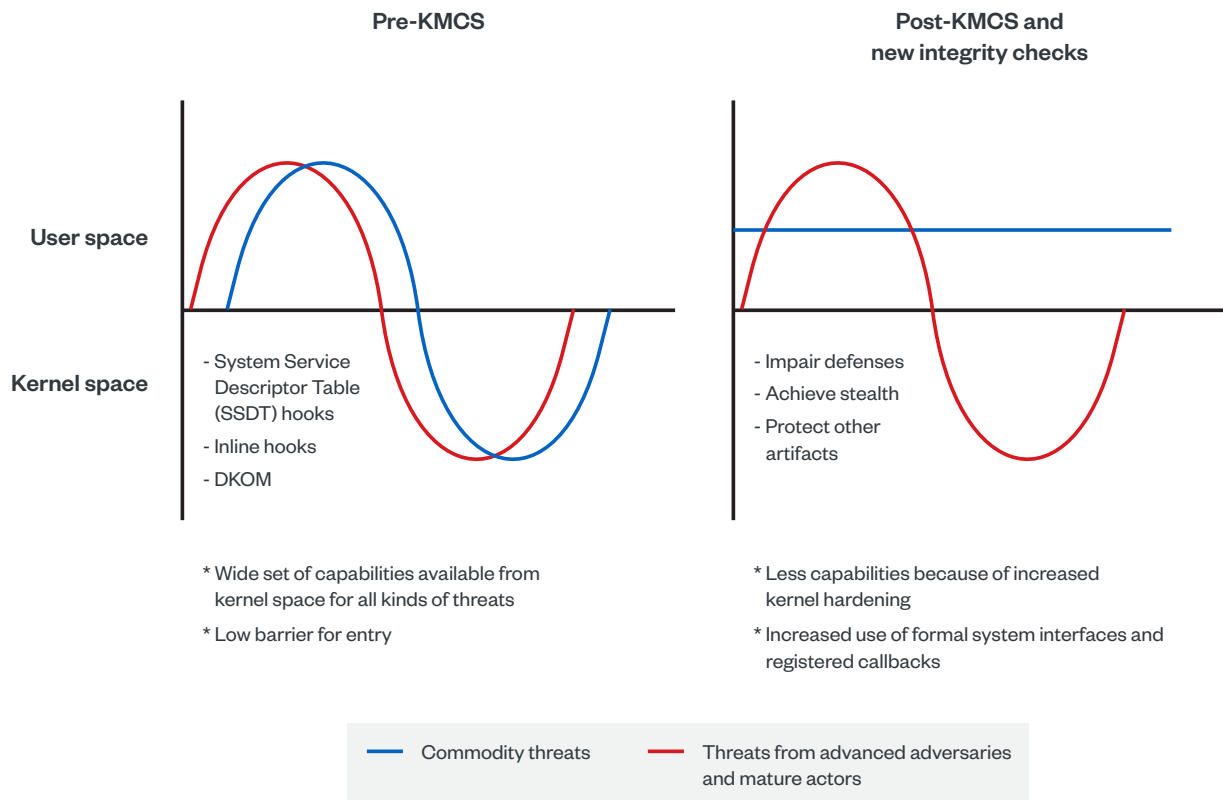


Figure 7. Hardening Windows kernel security post-KMCS adoption

## How Mature Malicious Actors Adapted to KMCS

After Microsoft introduced several security mechanisms into its newer versions, it was believed that kernel rootkits would be completely wiped out from the malware landscape — that this type of threat would fully migrate to the user space and the whole vector would eventually disappear. The research community also shifted their focus on understanding the possible bypass techniques for the security mechanisms that have been put into place. Even though KMCS reduced the sheer volume of kernel rootkits and helped in tracing malicious code back to its authors, it was not the silver bullet solution that would keep malicious actors from actively adding kernel-level access in their attacks.

Once the KMCS enforcement module was fully loaded into Windows systems, attackers found themselves unable to load unsigned code into the kernel. This forced them to look for other ways to bypass KMCS integrity checks. The following list summarizes the main clusters of observable techniques found in low-level threats discovered in the wild from April 2015 to October 2022. Each analyzed threat includes at least one kernel-level module that bypasses kernel-space access restrictions in their kill chains:

1. Threats that bypass KMCS

- Threats that disable KMCS using legitimate built-in tools
- Threats that abuse vulnerable drivers (bring your own vulnerable driver, or BYOVD, attacks)
- Threats that abuse dual-usage drivers
- Threats that target legacy systems

2. Threats that comply with KMCS using legitimate create-your-own-driver techniques

- Threats that use stolen valid code-signing certificates
- Threats that use acquired or purchased code-signing certificates

3. Threats that shifted to a lower abstraction layer

- Bootkits
- Firmware-level attacks
- BIOS implants

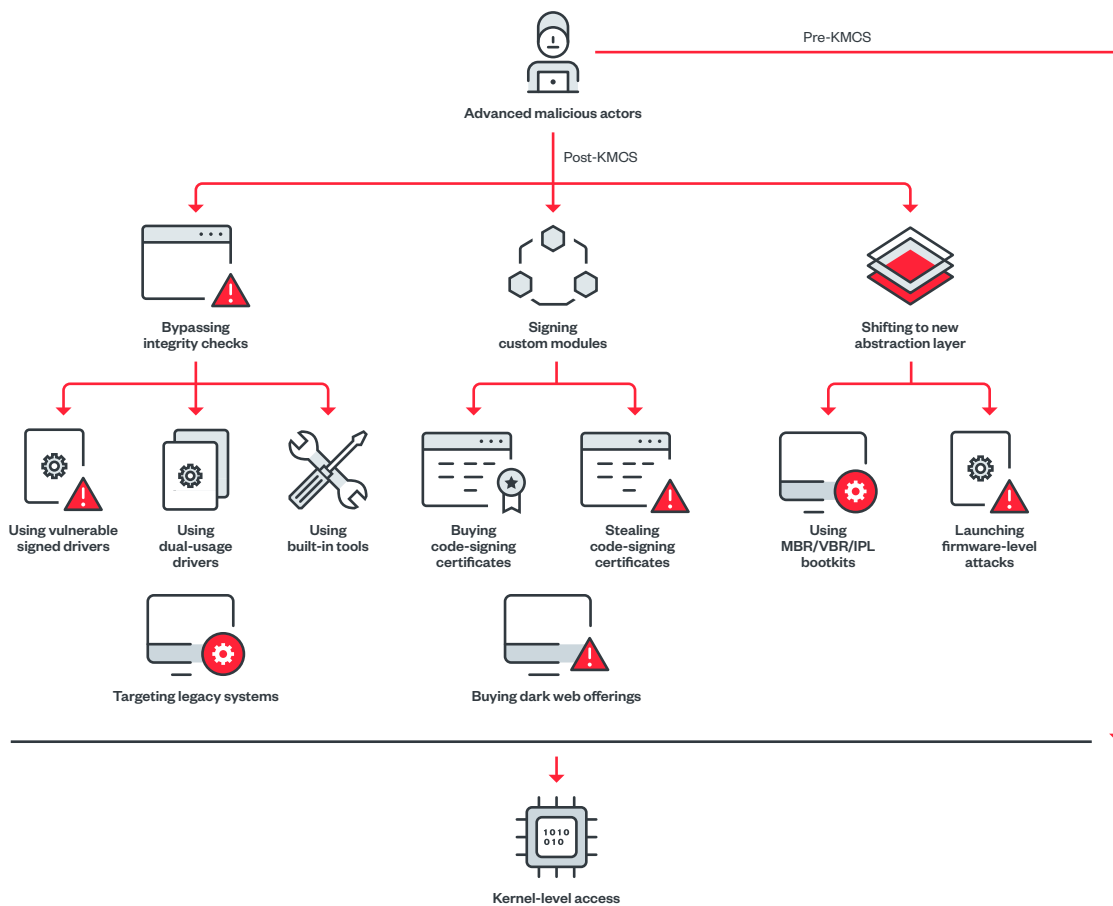


Figure 8. An illustration showing the increased number of clustered kernel-level threats post-KMCS adoption



Each of the kernel-level threats we analyzed belonged to one of the three main clusters. The first cluster depended directly on bypassing the signing restriction by different means: using legitimate administration tools to disable KMCS, compromising vulnerable code that was signed by legitimate vendors, and using dual-usage kernel drivers that had an exposed and generic interface for user-space applications that do not require proper authentication to conduct primitive kernel-level operations.

The threats in the first cluster use legitimate built-in tools that were mainly intended for debugging and testing to explicitly disable KMCS. These tools provide an interface for temporarily disabling driver verification and enabling test signing to verify the digital signature of the drivers. Inadvertently, these tools have stayed under the radar of monitoring systems.

The second technique in the first cluster, widely known as BYOVD, involves piggybacking the intended kernel code through a vulnerable driver that can be legitimately loaded into the kernel, whether it's a Windows system kernel driver or a third-party kernel driver. A recent example of this technique was when ransomware actors abused a vulnerable anti-cheat driver for the role-playing game Genshin Impact to disable antivirus processes and services.<sup>9</sup>

It should be noted that the definition of a vulnerable driver can include dual-purpose kernel drivers with generic input and output control (IOCTL) interfaces that support a wide range of kernel capabilities for user-space components. These drivers can have legitimate uses but can also be abused by attackers to bypass KMCS. Also, these drivers are useful for attackers as they can use them to completely disable the KMCS via the Code Integrity modules (CI.dll). These modules can be turned on and off with a single specific variable in the kernel memory space that controls the state of KMCS. By manipulating this variable using vulnerable drivers with memory read/write primitives, the whole integrity checks logic can be stopped. Later, Microsoft further enhanced its security in such a way that no single variable controls the Code Integrity status.

The second cluster threats take on a different approach. The threats in this cluster comply with Microsoft's signing requirements, which give them the flexibility to compile and sign the customized kernel drivers that are built for very specific tasks. This required the malicious actor to either obtain a valid code-signing certificate by impersonating a legitimate entity and following Microsoft's cross-signing certificate process (this was back when Microsoft still allowed cross-signing for kernel-mode code) or to steal someone else's certificate. An example of an attack that used the techniques in this second cluster is when a malicious actor abused the Windows Hardware Compatibility Program (WHCP) portal and submitted malicious drivers to be signed by Microsoft.<sup>10</sup> This attack, which happened in June 2021, targeted gaming environments. Compared to other techniques, the methods used in this attack came with a relatively high cost and are believed to be mainly used by state-sponsored APT actors. In December 2022, Microsoft revoked several Microsoft Windows Hardware Developer Program accounts after kernel drivers certified through the program were used in malicious campaigns, including ransomware attacks.<sup>11</sup>

The third cluster threats use a more complex yet effective strategy. This strategy involves completely moving into a lower-level layer in the software stack and operating at a new abstraction layer, as shown in figure 8. By doing so, it would be possible to load the malicious kernel code even before the full kernel and the core component that enforces the code-signing policy are initialized. Bootkit infection techniques saw a resurgence and different malicious actors were seen using it in malware families they used in actual attacks. The techniques in this cluster later evolved from just infecting MBR/VBR/IPL entries in legacy BIOS-based boot processes to abusing firmware vulnerabilities, which are located one layer closer to the hardware. The evolution in this cluster was mainly because of the introduction of more boot process security features that we will discuss in the following section.

## New Iterations of Kernel Defense Mechanisms

As malicious actors evolved their tactics and adapted to KMCS restrictions, Microsoft developers had to review their strategies once again and raise the bar against all the threats that used existing bypasses that allowed malicious actors to find their way back into the kernel. They shipped a new set of defensive capabilities with new Windows versions to further fortify the kernel boundary's overall security.

This started with the Early Launch Antimalware (ELAM) detection mechanism, which allowed third-party antivirus software to register a kernel-mode driver that is guaranteed to execute very early in the boot process before any other third-party driver is loaded.<sup>12</sup> ELAM, which was first introduced in Windows Server 2012 back in August 2012, gives the third-party antivirus software an advantage against known malicious kernel drivers, particularly those that belong in the first and second threat clusters. After the kernel components are completely initialized, it guarantees that antivirus kernel drivers would be loaded before any other malicious code that might already be registered in the system, but not against any other threats that would be dropped before the operating system kernel is loaded (i.e., threats belonging in the third cluster). ELAM was not designed to protect the lower abstraction layers in the boot process and can only monitor legitimately loaded drivers. It cannot monitor most bootkits and UEFI-level rootkits that load kernel-mode drivers using undocumented operating system features. This means that such threats can bypass this security feature and inject their code into the kernel address space.

Even before ELAM was shipped together with newer Windows systems, we witnessed a rise in the adoption of a security standard known as the secure boot. Secure boot was designed to ensure the integrity of the components involved in the boot process just before the kernel is initialized.<sup>13</sup> This security standard helped in the eradication of threats belonging in the third cluster, which targeted the Windows boot process. It also created obstacles for bootkit authors who reacted to this security feature and moved their attacks one layer deeper into the stack to target the actual firmware. It is believed that once the secure boot security standard garners a larger adoption rate in corporate systems, the threats belonging in the third cluster would have to pivot and change their techniques. Instead of targeting the boot process, the system's firmware vulnerabilities will be the next infection point to be targeted in the software stack.

With Windows 10, we saw the emergence of virtualization-based security with the Virtual Secure Mode (VSM), a feature that uses the virtualization extensions of modern CPUs to provide added security for data in memory.<sup>14</sup> It supports more robust defense mechanisms that take the integrity-check logic in Windows systems to a whole new level. This feature led to the eradication of some of the threat clusters that depended on finding a route to disable the integrity checks once the kernel is compromised (i.e., the first cluster threats disable integrity checks through vulnerable drivers or dual-usage drivers). At the core of these integrity checks is the ability to run in a separate virtual environment isolated from the kernel image so that even if it is compromised, the hypervisor would still be able to enforce another boundary to the isolated Code Integrity and other security mechanisms.

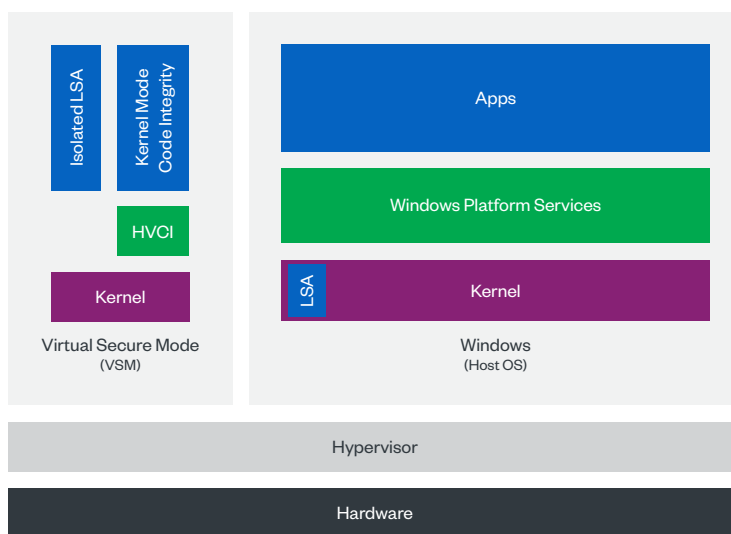


Figure 9. Windows 10 user-mode and kernel-mode levels

Source: *Windows Internals, Part 2, 7th Edition*

HVCI is a critical component that protects and hardens this virtual environment by running kernel-mode Code Integrity within it and restricting kernel memory allocations that could be used to compromise the system. Both HVCI trustlet and VBS improved the Windows kernel trust model and provided a stronger protection against modern malware targeting the Windows kernel using different exploitation attempts.

Once this design becomes more popular and gains a higher adoption rate, third-party security solutions might also start customizing their own hypervisors when Microsoft makes the appropriate interfaces available and when enough modern PC hardware becomes more readily available. This might be the future of all operating systems' core defenses. However, with all these advanced security mechanisms, we still see threats targeting the most fortified regions of the Windows kernel.<sup>15</sup>

# Why Is the Threat Still Alive?

Threat actors will always try to follow the path of least resistance to gain a high access level that could accelerate and support their main objectives.

Even though the entire software stack seems to be properly guarded by defensive features — including the early boot process, a fully loaded and initialized kernel guarded by a hypervisor layer, and all the built-in security mechanisms that come with newer Windows versions and hardware-level support — the following are the reasons why Windows kernel-level threats are still alive and will not completely disappear anytime soon:

- Legacy systems are still found in most corporate networks.
- The high hardware requirements for the modern innovative virtualization-based and secure boot mechanisms.
- The low adoption rate for the new kernel defense mechanisms due to its performance impact and backward compatibility issues.
- The new defenses are not bulletproof. It only takes one bypass in the protection mechanism in any layer to compromise the subsequent layers.<sup>16</sup>

We are still observing new kernel-level threats deployed for a plethora of objectives on targeted networks. Even with security mechanisms deployed in Windows systems, attackers will adapt by changing infection points in the software stack.

# A Chronological View of Windows Kernel Threats

## A Look at Analyzed Threat Data

In this section, we will provide a chronological view of the analyzed threats that either completely rely on a kernel driver component or have at least one module in their chain that executes in the kernel space. The analyzed data included only the threats that are observed in the wild; PoCs were not included. The following diagram shows the number of noteworthy threats and other major events that have been reported by the cybersecurity community over the last seven years. The numbers show a significant trend in the last five years.

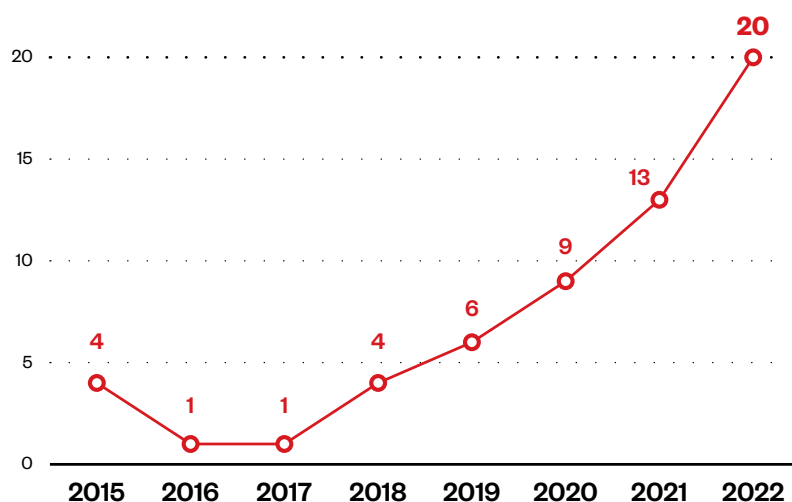


Figure 10. The number of public intelligence reports that included kernel-level threats from April 2015 to October 2022

Figure 11 shows that the third cluster has the fewest threats. Figure 12 shows how each cluster has evolved over the past seven years, revealing a notable increase in the number of third-cluster threats in the last three years.

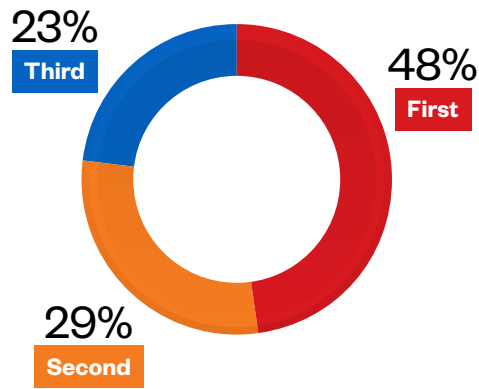


Figure 11. The distribution of kernel-level threats among the three clusters from April 2015 to October 2022

Based on Figure 11, it is apparent that the threats belonging in the first cluster is still the most popular cluster across different threat actors. At present, the threats belonging to the first cluster still represent most of the threats affecting the Windows kernel. The number of threats in this cluster is expected to rise until the adoption rate for the new hypervisor-based defense solutions introduced in Windows 10 increases, which will significantly decrease the number of threats in it. After Microsoft replaces the cross-signing process for kernel drivers with stricter procedures to verify, test, and sign kernel drivers within their portal, the volume of the vulnerable kernel drivers that are being abused by the threats in the first cluster will decrease.

The threats belonging in the second cluster are less common compared to those belonging in the first cluster due to the higher cost of developing such attacks. Despite this, the number of second cluster threats in Figure 12 has increased since 2018, but it is expected to decrease and eventually cease because of the kernel code signing policy in Windows 10 and 11. We will delve into these signing policies and how these will affect kernel-level threats in succeeding sections.

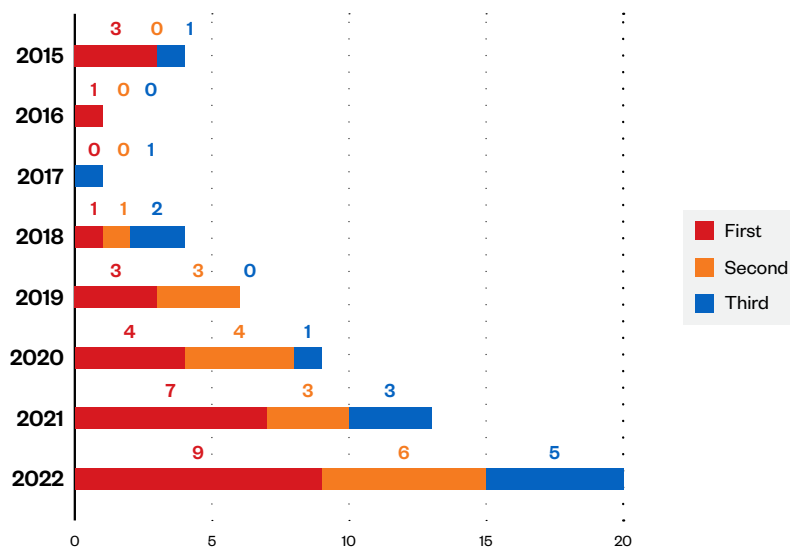


Figure 12. Kernel-level threats categorized by cluster from April 2015 to October 2022

Finally, the threats in the third cluster are the least common ones due to its complex design and they are only being used by very advanced and mature actors. We foresee that these threats will slowly increase in the coming years as attackers shift their initial infection points earlier in the process to evade the modern security mechanisms we have previously discussed. As mentioned earlier, this increase will be slow due to the level of complexity involved in developing these threats.

From the data we have analyzed for this research, we were also able to categorize the threat types that used kernel-level access, which can be seen in Figure 13.

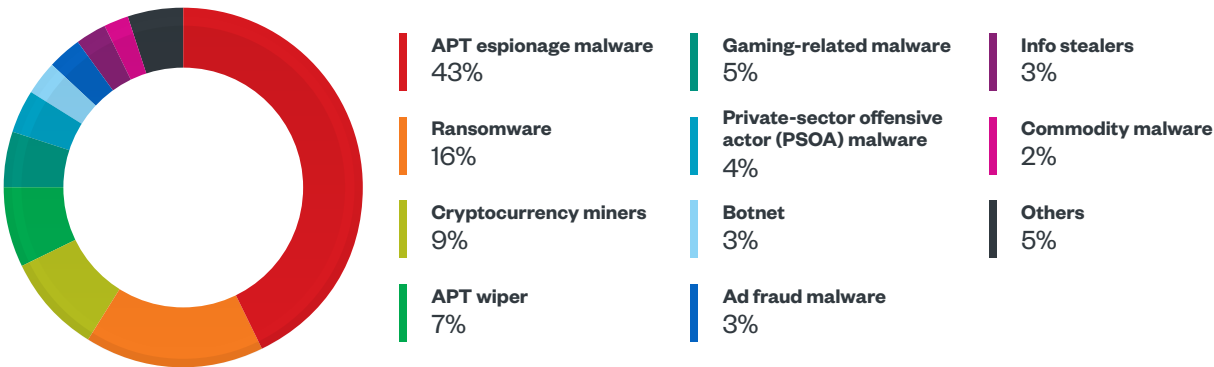


Figure 13. The types of threats that used kernel-level malware from April 2015 to October 2022

Figure 13 shows that APT espionage malware used low-level components the most in their attacks. After all, APT espionage groups are drawn to using stealthy components such as kernel rootkits and lower-level implants in their operations and have the ability and resources to develop and deploy kernel rootkits in their attacks. These threats are difficult to detect and eradicate since most of the rootkits’ instances will be associated with high-profile targeted attacks launched by advanced actors.

Ransomware actors and their affiliates also showed a high level of interest in gaining privileged-level access for the ransomware families they used in their attacks. They used ransomware families that incorporated low-level components to avoid being detected by security products once they drop their final payloads. Cryptomining threats were also observed to have a kernel foothold, and their main objective is to protect their illicit cryptomining processes, hide their different components, and fake the performance degradation caused by their cryptominers on affected systems.

By mapping these kernel-level threats’ respective kill chains, we found that most kernel-related payloads are usually found in the defense evasion phase, as shown in Figure 14. We believe that this is typical for the kind of privileged access achieved when malicious actors obtain kernel-level access.

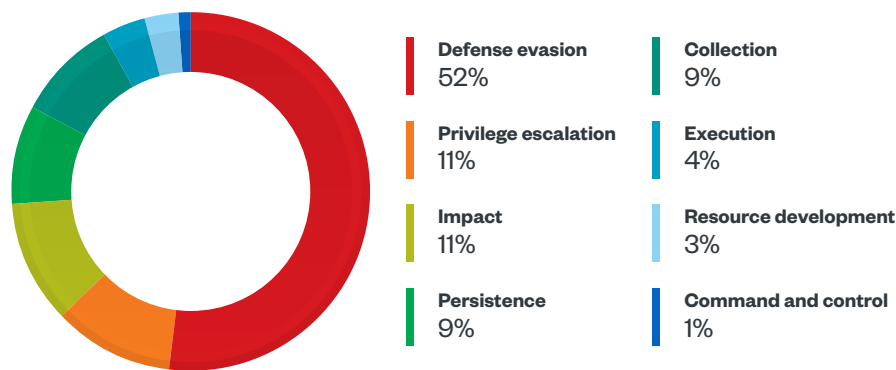


Figure 14. Identifying where kernel-level threats are found in the threats' respective kill chain phases

In the collection phase of the kill chain, malicious actors attempt to gather critical data, and doing so from the kernel space will give them unfettered and undetected access to protected resources. Meanwhile, in the resource development phase, advanced malicious actors attempt to have a wide range of capabilities that allow them to gain kernel-level access, such as obtaining code-signing certificates for loading custom drivers. This is usually planned before the actual intrusion.

Figure 14 shows how we mapped the most used MITRE Adversarial Tactics, Techniques, and Common Knowledge (ATT&CK) techniques in the kernel-level threats analyzed in this report.

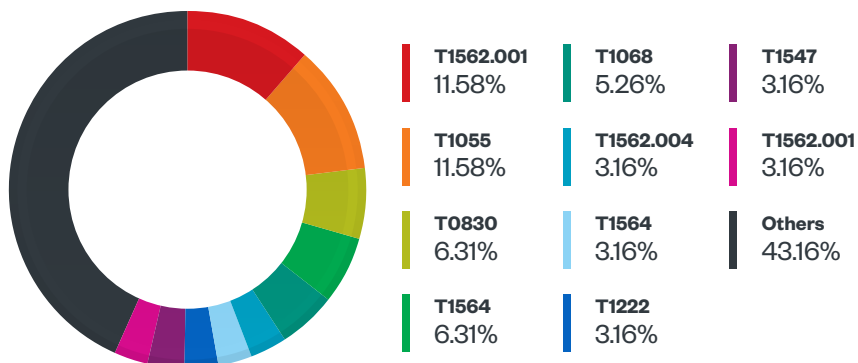


Figure 15. Mapping the kernel threats' MITRE ATT&CK techniques distribution

Based on our data, 11% of kernel-level threats attempt to gain kernel-level access to impair defenses (Impair Defenses: Disable or Modify Tools, T1562.001). Adversaries try to modify and/or disable security tools to avoid possible detection of their malware payload, tools, and activities. This use case makes sense because of the increased protection on user-land processes by endpoint protection platform (EPP) and endpoint detection and response (EDR) technologies, either on users' desktop or servers. Because of these added layers of protection, attackers will opt for the path of least resistance and get some of their code running in the kernel level to interfere with security solutions.



# VirusTotal Statistics

We also assessed our Windows kernel driver samples based on a) their signed drivers being revoked or otherwise and b) their having one or more positive detection based on malware search engines including the VirusTotal malware repository. The following is how we segregated each set of the threat samples we have gathered from January 2015 to May 2022:

Sample set	Description
Set 1	Signed drivers that have not been revoked with zero positive detection
Set 2	Signed drivers that have not been revoked with one or more positive detection from different engines
Set 3	Signed drivers that have been revoked with zero positive detection
Set 4	Signed drivers that have been revoked with one or more positive detection from different engines

We saw that the majority of the kernel-level threats that we have analyzed had signed drivers that have not been revoked with one or more positive detection from different engines followed closely by threats that had signed drivers that have not been revoked with zero positive detection.

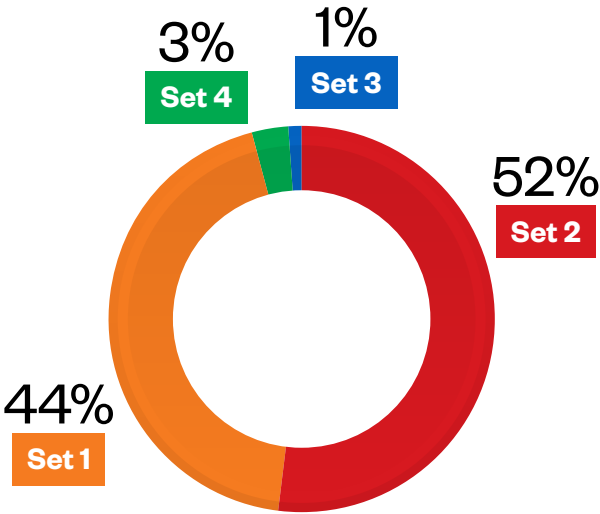


Figure 16. Windows kernel driver samples based on a) their signed drivers being revoked or otherwise and b) their having one or more positive detections

Our data shows an increase in the number of threats belonging in three of the four sample sets, namely sets 2, 3, and 4 from 2020 to May 2022.

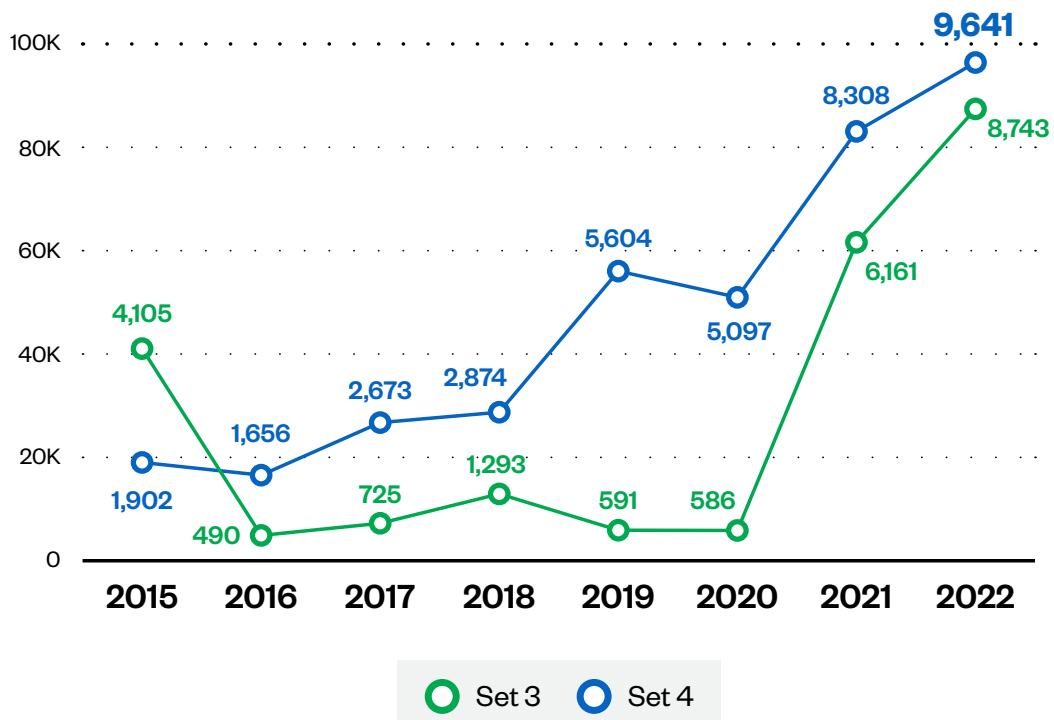
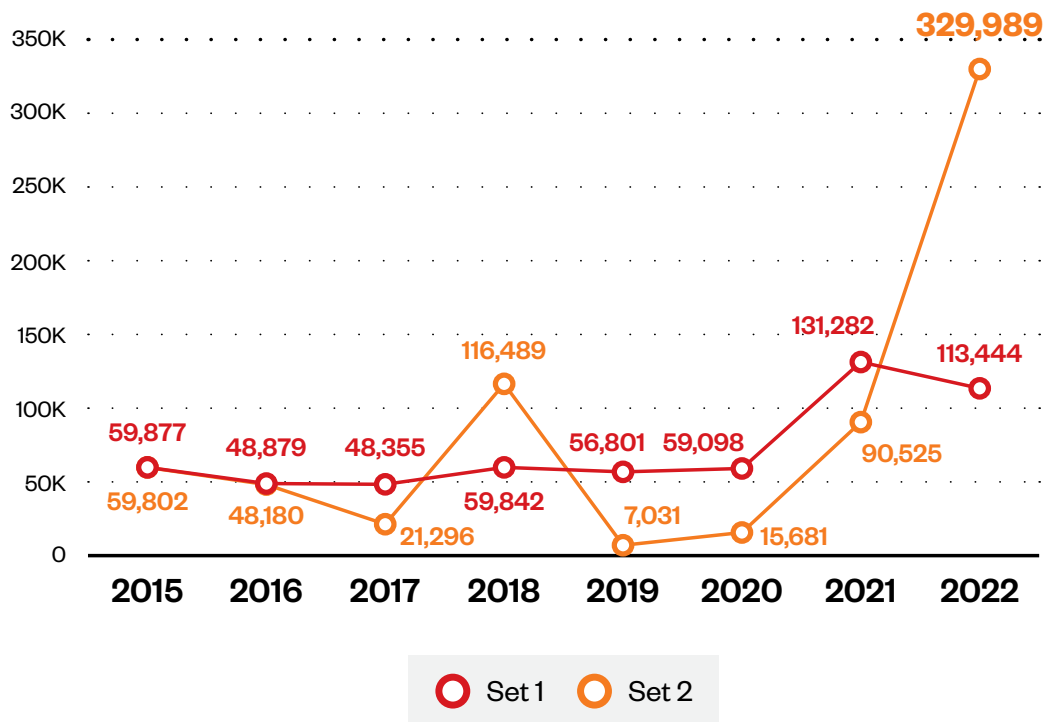
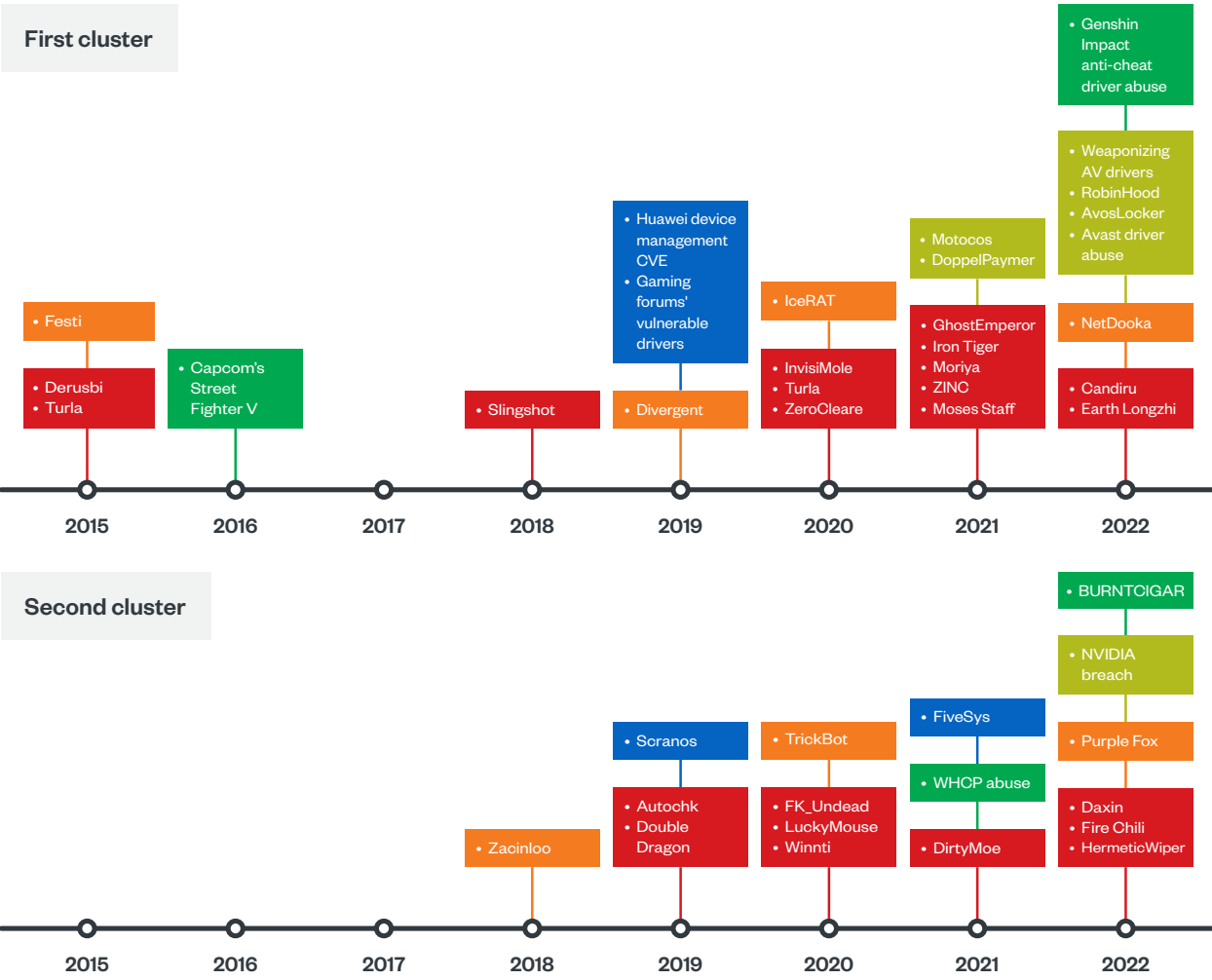


Figure 17. An increase in the number of kernel driver submissions belonging to sample sets 2, 3, and 4 from 2020 to May 2022

# Noteworthy Events

In this section, we list down the publicly disclosed and noteworthy events that affected the Windows kernel from April 2015 to October 2022. Figure 17 shows the timeline of major events for each threat cluster, while the succeeding tables provide further details on each threat, such as how it was used to achieve malicious actors' attack objectives and how it affected the Windows kernel trust model. Tables 1,2 and 3 list samples from each threat cluster. It should be noted that PoCs and attacks without publicly disclosed samples to confirm their relevance in this study were not included.



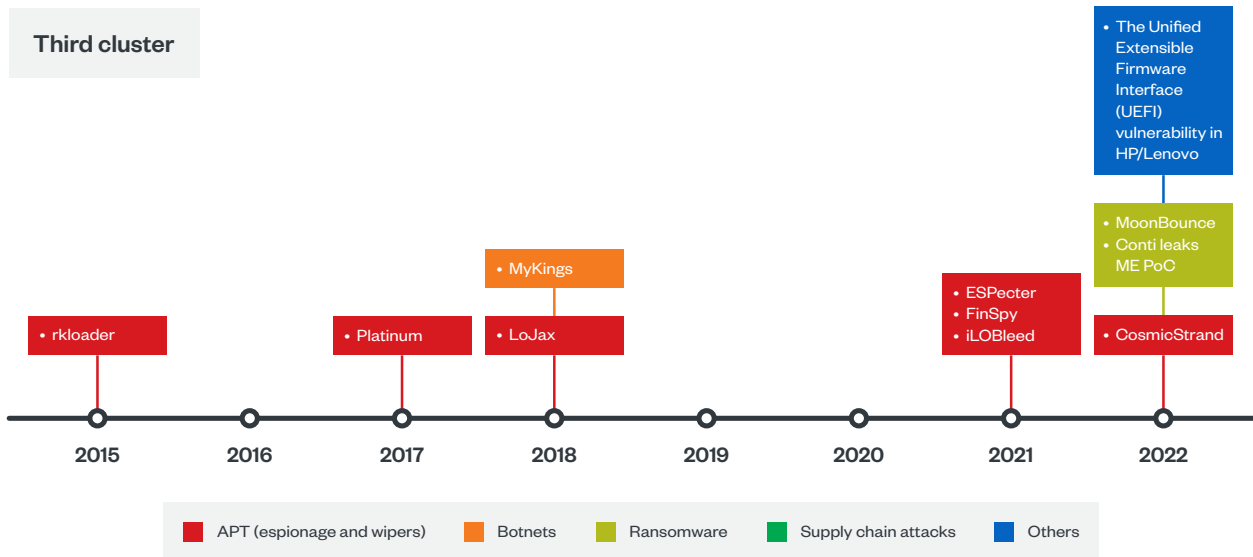


Figure 18. Noteworthy events that affected the Windows kernel from April 2015 to October 2022

Threat Name	Threat Profile
Earth Longzhi (2022) <sup>17</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> A campaign that targets multiple regions using a custom Cobalt Strike loader. The APT group targets high-profile victims in the defense, aviation, insurance, and urban development industries in Taiwan, China, Thailand, Malaysia, Indonesia, Pakistan, and Ukraine.</p> <p><b>Kernel driver operation:</b> The vulnerable driver (RTCORE64.sys) allows authenticated users to read/write any arbitrary address including the kernel space. It is used to terminate antivirus products.</p>
Genshin Impact anti-cheat driver abuse (2022) <sup>18</sup>	<p><b>Threat type:</b> Ransomware</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> A threat actor that deploys ransomware within the victim's device by abusing a vulnerable driver that provides anti-cheat functions.</p> <p><b>Kernel driver operation:</b> The driver is currently being abused by a ransomware actor to kill antivirus processes and services for the mass deployment of ransomware.</p>
Candiru (2022) <sup>19, 20, 21</sup>	<p><b>Threat type:</b> PSOA</p> <p><b>Technique:</b> Signed Driver</p> <p><b>Threat details:</b> Candiru is a private company that sells cyberweapons to government agencies via hacking-as-a-service packages. One of the weapons they sell is a spyware that targets users located in Lebanon, Turkey, Yemen, and Palestine via watering hole attacks using zero-day exploits for Google Chrome</p> <p><b>Kernel driver operation:</b> It uses a signed driver called phymem.sys. The driver's description is "Physical Memory Access Driver," and it offers a "by-design" kernel read/write capability. It is abused to proxy certain API calls via the kernel to hinder detection, including the capability to have some of the calls appear from other processes.</p>

Threat Name	Threat Profile
Weaponizing an antivirus driver (2022) <sup>22</sup>	<p><b>Threat type:</b> Ransomware</p> <p><b>Technique:</b> Dual-use driver</p> <p><b>Threat details:</b> A ransomware campaign that abuses a function in an Avast Anti Rootkit kernel driver to terminate popular AV and EDR processes.</p> <p><b>Kernel driver operation:</b> It kills processes and files belonging to endpoint security products.</p>
Avast driver (2022) <sup>23</sup>	<p><b>Threat type:</b> Ransomware</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> These vulnerabilities give malicious actors escalated privileges for disabling security products, overwriting system components, and performing other malicious operations undetected.</p> <p><b>Kernel driver operation:</b> Malicious actors trigger CVE-2022-26522 and CVE-2022-26523 in a socket connection handler in the <i>aswArPot.sys</i> kernel driver.</p>
NetDooka (2022) <sup>24</sup>	<p><b>Threat type:</b> Commodity malware</p> <p><b>Technique:</b> Built-in tools</p> <p><b>Threat details:</b> A sophisticated malware that includes a loader, a dropper, a protection driver, and a full-featured RAT that implements its own network communication protocol. Its attack is distributed via a pay-per-install (PPI) service.</p> <p><b>Kernel driver operation:</b> The driver's main functionality is to protect and hide user-mode components.</p>
AvosLocker(2022) <sup>25</sup>	<p><b>Threat type:</b> Ransomware</p> <p><b>Technique:</b> Dual-use driver</p> <p><b>Threat details:</b> A ransomware campaign that uses an exported functionality from a legitimate Avast Anti-Rootkit Driver (<i>asWarPot.sys</i>) to terminate security-related processes.</p> <p><b>Kernel driver operation:</b> A legitimate driver is used to terminate security-related processes to enable the ransomware payload to run uninterrupted and undetected.</p>
RobbinHood (2022) <sup>26</sup>	<p><b>Threat type:</b> Ransomware</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> A ransomware campaign that uses a vulnerable driver to disable signing policy to load an unsigned driver.</p> <p><b>Kernel driver operation:</b> The unsigned driver kills processes and files belonging to endpoint security products and bypasses tamper protection features to enable the ransomware payload to run uninterrupted and undetected.</p>

Threat Name	Threat Profile
DoppelPaymer (2021) <sup>27</sup>	<p><b>Threat type:</b> Ransomware</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> A ransomware campaign that uses the Process Hacker tool to terminate security-related services and processes, email servers, and database software.</p> <p><b>Kernel driver operation:</b> It uses a sideloading DLL vulnerability inside the user-mode process of the Process hacker tool to communicate with the Process Hacker driver and terminate processes.</p>
MosesStaff (2021) <sup>28</sup>	<p><b>Threat type:</b> APT wiper</p> <p><b>Technique:</b> Abusing third-party driver</p> <p><b>Threat details:</b> A campaign that targeted Israeli organizations by stealing sensitive information, encrypting victims' networks, and leaking stolen data without any demand for ransom.</p> <p><b>Kernel driver operation:</b> It uses the DiskCryptor open library to encrypt victims' machines then install custom bootloader to lock the victims' machines.</p>
GhostEmperor (2021) <sup>29, 30</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Dual-use driver</p> <p><b>Threat details:</b> A sophisticated multi-stage malware framework aimed at providing remote control over victim servers.</p> <p><b>Kernel driver operation:</b> The kernel driver hides user-mode artifacts such as file, registry, TCP connections and running services.</p>
Motocos (2021) <sup>31</sup>	<p><b>Threat type:</b> Ransomware</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> The Motocos ransomware uses Telegram to communicate with its victims. This ransomware also subjects its victims to an increasing amount of ransom with each day victims' machines are infected with it.</p> <p><b>Kernel Driver operation:</b> The Motocos ransomware shares a similar driver structure with that of the RobinHood ransomware.</p>
Moriya (2021) <sup>32</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> A campaign that targeted several prominent organizations in Asia and Africa and deployed passive backdoors on public-facing servers.</p> <p><b>Kernel driver operation:</b> The driver does network packet inspection, allowing attackers to drop the packets of interest before they are processed by the network stack, thus ensuring they are not detected by security solutions. There are also other drivers that might be related to Moriya developers that kills AV processes.</p>

Threat Name	Threat Profile
Iron Tiger (2021) <sup>33</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> A campaign that targets gambling and betting companies in Southeast Asia to install a backdoor. It takes advantage of a sideloaded injection technique present in <i>dlpumgr32.exe</i>, and utilizes a known vulnerability to disable DSE and load an unsigned driver.</p> <p><b>Kernel driver operation:</b> The kernel driver filters incoming traffic with a predefined token and injects code inside the “<i>lsass.exe</i>” process.</p>
ZINC (2021) <sup>34</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> A campaign that targeted threat researchers by sending malicious Visual Studio projects that included prebuilt binaries, one of which is a malicious DLL called <i>Browse.vc.db</i>.</p> <p><b>Kernel driver operation:</b> Attempts to exploit CVE-2017-16238 inside <i>Viraglt64.sys</i>. However, the code appears to be buggy and fails to successfully exploit the vulnerability.</p>
InvisiMole (2020) <sup>35</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> A backdoor that provides extensive espionage capabilities, such as recording victims’ webcams and microphones, tracking geolocations, and collecting recently accessed documents.</p> <p><b>Kernel driver operation:</b> A vulnerable driver that is used to inject code inside legitime user-mode processes.</p>
AcidBox (2020) <sup>36</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> Abuses Windows Security Support Providers (SSPs) for persistence and injecting DLLs inside the <i>lsass</i> process and exploiting VirtualBox to load unsigned drivers.</p> <p><b>Kernel driver operation:</b> Waits for commands from one or more components. These commands include the loading of additional registry payloads from the kernel space via the driver or the installation of new SSP DLLs.</p>
ZeroCleare (2020) <sup>37</sup>	<p><b>Threat type:</b> APT wiper</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> It is a destructive attack that aims to overwrite the Master Boot Record (MBR) and disk partitions on Windows-based machines. It exploits a vulnerability driver to disable DSE and load the <i>EIRawDisk</i> driver to access raw hard disk data.</p> <p><b>Kernel driver operation:</b> It is an unsigned driver that gives access to the hard disk.</p>

Threat Name	Threat Profile
Divergent (2019) <sup>38,39</sup>	<p><b>Threat type:</b> Ad fraud</p> <p><b>Technique:</b> Abusing third-party drivers</p> <p><b>Threat Details:</b> A new malware loader that uses NodeJS as well as a legitimate open-source utility called WinDivert. An attacker can use this malware to target corporate networks. It appears to be primarily designed to conduct click-fraud.</p> <p><b>Kernel driver operation:</b> The malware uses the WinDivert library to block AV traffic and intercept and rewrite the first SYN (synchronize) packet of the three-way TCP (Transmission Control Protocol) handshake for all outgoing connections the infected host attempts to make, the changes made to the SYN packets depend on which executable was used, either <i>divergent.exe</i> or <i>mdivergent.exe</i>.</p>
Driver vulnerability (2019) <sup>40</sup>	<p><b>Threat type:</b> Others</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> An abuse of a vulnerability in a Huawei driver that allows attackers to inject code inside other user-mode processes using Windows asynchronous procedure call (APC) from the kernel space, which will result in local privilege escalation.</p> <p><b>Kernel driver operation:</b> A vulnerability in a Huawei driver that allows an unauthorized process to inject code inside other processes.</p>
SlingShot (2018) <sup>41</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b></p> <p><b>Threat details:</b> A loader called Slingshot replaces a system DLL called “scesrv.dll” with its own (it will be loaded by Services.exe with SYSTEM privilege). For persistence and privilege escalation, the malicious DLL will exploit a vulnerable driver to disable the Driver Signature Enforcement (DSE) and load its own unsigned driver.</p> <p><b>Kernel driver operation:</b> The kernel-mode component does the following: Injects payload to user-mode processes, hides/sniffs network traffic, hooks function for anti-debugging techniques, receives code to execute in the kernel level, notifies user-mode components of event-related processes.</p>
Capcom’s Street Fighter V (2016) <sup>42</sup>	<p><b>Threat type:</b> Gaming-related</p> <p><b>Technique:</b> Vulnerable supply chain</p> <p><b>Threat details:</b> An update that installs a rootkit that grants kernel-level privileges to installed applications on affected computers.</p> <p><b>Kernel driver operation:</b> The rootkit disables the Supervisor Mode Execution Protection (SMEP) feature in the operating system to execute malicious code. After which, it reenables SMEP.</p>



Threat Name	Threat Profile
DERUSBI (2015) <sup>43</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> This is a Derusbi server variant which has a largely unremarkable remote access trojan (RAT). It supports basic RAT functionalities, such as file management (uploading and downloading), network tunneling, and remote command shell.</p> <p><b>Kernel driver operation:</b> The driver hooks into the Windows firewall by either using undocumented Windows Firewall hooking techniques found in Windows XP and older versions, or by using the documented Windows Filtering Platform found in Windows Vista and later versions. If a specific handshake occurs between the client and the server variant, the remainder of the communication session for the established session will be redirected. This allows an attacker to hide their communication within a cluster of network sessions originating from a single IP.</p>
Festi (2015) <sup>44</sup>	<p><b>Threat type:</b> Botnet</p> <p><b>Technique:</b> Legacy systems</p> <p><b>Threat details:</b> A bot that implements a very powerful DoS (Denial of Service) engine and sends spam messages. It is distributed mainly through a pay-per-install (PPI) scheme.</p> <p><b>Kernel driver operation:</b> The main module of the attack is responsible for Updating the configuration data from the command-and-control server (C&amp;C) and downloading additional dedicated kernel plugins to be executed (these will perform all the DDOS and spam attacks).</p>
Turla (2015) <sup>45</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Vulnerable driver</p> <p><b>Threat details:</b> An advanced malware with a sophisticated rootkit. It is based on a distributed C&amp;C architecture that can be used for a wide range of purposes, such as cyberespionage or credential theft.</p> <p><b>Kernel driver operation:</b> The kernel component's main functionality is to hide/protect its user-mode components by modifying ntoskrnl.exe and ndis.sys in memory and creating a new IDT entry.</p>

Table 1. List of threats belonging in the first cluster

Threat Name	Context
BURNTCIGAR (2022) <sup>46</sup>	<p><b>Threat type:</b> Ransomware</p> <p><b>Technique:</b> Signed driver</p> <p><b>Threat details:</b> A malicious actor that utilizes a Microsoft-signed malicious driver to try to evade multiple security products.</p> <p><b>Kernel driver operation:</b> It terminates processes or services used by a variety of endpoint security product vendors.</p>

Threat Name	Context
NVIDIA breach (2022) <sup>47, 48</sup>	<p><b>Threat type:</b> Ransomware</p> <p><b>Technique:</b> Leaked certificates</p> <p><b>Threat details:</b> Cybercrime group Lapsus\$ breached NVIDIA to steal data and digital-signing certificates.</p> <p><b>Kernel driver operation:</b> Credentials, source code, and two code-signing digital certificates that have expired were leaked to the public</p>
Fire-Chili (2022) <sup>49</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> A campaign that mainly targets financial, academic, cosmetics, and travel industries. It deploys a backdoor that is based on leaked Gh0st RAT code and uses stolen digital certificates on infected machines.</p> <p><b>Kernel driver operation:</b> The driver is digitally signed with stolen certificates from game development companies. It uses Direct Kernel Object Modification (DKOM) and hides and protects malicious artifacts from user-mode components.</p>
HermaticWiper Executable (2022) <sup>50, 51, 52</sup>	<p><b>Threat type:</b> APT wiper</p> <p><b>Technique:</b> Getting CS certificate</p> <p><b>Threat details:</b> A wiper attack that targets Ukrainian organizations.</p> <p><b>Kernel driver operation:</b> The driver acts as a proxy that allows user-mode components to write to certain sectors of the raw disk.</p>
Daxin (2022) <sup>53</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> A backdoor that is used in a long-running espionage campaign against select governments and other targeted critical infrastructures. It allows the attacker to perform various communications and data-gathering operations on infected computers.</p> <p><b>Kernel driver operation:</b> The driver hijacks legitimate TCP/IP connections, reads and writes arbitrary files, and starts arbitrary processes.</p>
PurpleFox (2022) <sup>54, 55</sup>	<p><b>Threat type:</b> Cryptomining</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> An attack that takes advantage of an infected machine's resources via cryptocurrency miners.</p> <p><b>Kernel driver operation:</b> The driver stops security products' mini-filter drivers, copies and deletes files, installs services, and kills processes.</p>

Threat Name	Context
FiveSys (2021) <sup>56, 57</sup>	<p><b>Threat type:</b> Info stealer</p> <p><b>Technique:</b> WHCP signed code</p> <p><b>Threat details:</b> An attack that targets online games with the main goal of credential theft and in-game-purchase hijacking.</p> <p><b>Kernel driver operation:</b> The driver is used to proxy traffic to internet addresses that interest the attackers, protect user-mode components, and stop other malwares drivers from loading in an infected environment.</p>
DirtyMOE (2021) <sup>58, 59</sup>	<p><b>Threat type:</b> Cryptomining</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> This campaign's main goals are performing cryptojacking and launching DDoS attacks on infected machines.</p> <p><b>Kernel driver operation:</b> The driver hides user-mode malicious activities and services. It also executes commands received from the user-mode, such as writing to file system and registry, killing processes, injecting arbitrary DLLs into targeted processes.</p>
WHCP abuse (2021) <sup>60</sup>	<p><b>Threat type:</b> Gaming-related</p> <p><b>Technique:</b> Legitimately signed driver</p> <p><b>Threat details:</b> A campaign that targets gaming environments in China.</p> <p><b>Kernel driver operation:</b> Malicious drivers are used to spoof cybercriminals' geo-locations so that they can cheat the system and play from anywhere.</p>
IceRat (2020) <sup>61, 62</sup>	<p><b>Threat type:</b> Cryptocurrency mining</p> <p><b>Technique:</b> Signed Driver</p> <p><b>Threat details:</b> This is a backdoor that enables illicit cryptocurrency-mining activities on victim machines.</p> <p><b>Kernel Driver Operation:</b> Uses the WinRing0x64 driver that allows access to the kernel space to read/write memory and access the CPU model-specific register (MSR).</p>
TrickBot (2020) <sup>63</sup>	<p><b>Threat type:</b> Botnet</p> <p><b>Technique:</b> Signed driver</p> <p><b>Threat details:</b> A firmware-level threat that is triggered by exploiting well-known vulnerabilities. By implanting malicious code in the firmware, attackers can ensure that their code is the first to run in an infected machine.</p> <p><b>Kernel driver operation:</b> It uses the REWEverything driver to directly access hardware interfaces and search for well-known vulnerabilities in the firmware level, patch the firmware, and maintain persistence on the machine.</p>

Threat Name	Context
LuckyMouse (2020) <sup>64</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> A backdoor that waits passively for the C&amp;C server to connect to, two possible communication channels, ports 3389 and 443.</p> <p><b>Kernel driver operation:</b> The driver decrypts and injects payload into memory. It also filters traffic going through RDP (Remote Desktop Protocol) port 3389 and inserts the Trojan's C2 communications into it.</p>
FK_Undead (2020) <sup>65, 66</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> A multistage spyware threat that includes at least three different rootkit modules.</p> <p><b>Kernel driver operation:</b> The drivers monitor all network traffic (inject script in webpages), add proxy to browsers, stop other malware drivers from loading, protect its registry, and monitor all access to HOSTS files then provide a malicious version of it whenever svchost.exe tries to access it.</p>
Winnti (2020) <sup>67</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> A backdoor that relies on a DNS tunneling communication channel through a custom implementation of the iodine source code and uses a stolen digital certificate to digitally sign their drivers.</p> <p><b>Kernel driver operation:</b> The driver is capable of injecting raw packets into the network and receiving special formatted packets.</p>
Autochk (2019) <sup>68</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Signed driver</p> <p><b>Threat details:</b> A China-based threat actor that targets foreign embassies to collect data on government, defense, and technology sectors.</p> <p><b>Kernel driver operation:</b> The driver redirects malware files to point to normal files (to evade security solutions). It also hides network connections to the C&amp;C server.</p>
Double Dragon (2019) <sup>69, 70</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> A threat actor that targets industries such as gaming, healthcare, high-tech, higher education, telecommunications, and travel services. it uses an exploit to gain access to victims' machines, then installs a backdoor on the compromised system.</p> <p><b>Kernel driver operation:</b> The driver conceals the network traffic and communicates with the C&amp;C servers.</p>

Threat Name	Context
Scranos (2019) <sup>71</sup>	<p><b>Threat type:</b> Info stealer</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> A spyware campaign that aims to steal login credentials from browsers and user's payment accounts, exfiltrate browsing history, inject JavaScript adware in Internet Explorer, etc.</p> <p>The operation is based around a rootkit driver that was digitally signed with a possibly stolen certificate.</p> <p><b>Kernel driver operation:</b> The driver injects the malicious payload inside a user-mode process, it also installs other components on the infected machine to bypasses DSE and PatchGuard.</p>
Zacinlo (2018) <sup>72</sup>	<p><b>Threat type:</b> Ad fraud</p> <p><b>Technique:</b> Signed code</p> <p><b>Threat details:</b> A sophisticated piece of adware that generates revenue for its operators and compromises the privacy of its victims.</p> <p><b>Kernel driver operation:</b> The kernel driver protects its user-mode component, also it has the ability to terminates other processes that affect its operation and installs a man-in-the-browser functionality to intercepts and decrypts SSL communications and inject malicious scripts.</p>

Table 2. List of threats belonging in the second cluster

Threat Name	Context
CosmicStrand (2022) <sup>73</sup>	<p><b>Threat type:</b> Others</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat details:</b> This is a UEFI firmware rootkit that is attributed to a Chinese-speaking malicious actor, the long execution chain will result in the download and deployment of a malicious component inside Windows.</p> <p><b>Boot driver operation:</b> This rootkit is located in Gigabyte or ASUS motherboard firmware images. It hooks the windows boot manager to load its driver, which will communicate with C&amp;C server.</p>
MoonBounce (2022) <sup>74, 75</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat details:</b> A threat activity that was discovered within government organizations in the Middle East using a UEFI bootkit of its own to load its driver. The driver then injects payload inside user-mode process to download other stages of the malware.</p> <p><b>Boot driver operation:</b> The UEFI firmware is used for persistent malware, and to deploy malicious code that will be run after the operating system is loaded.</p>

Threat Name	Context
System Management Mode (SMM) vulnerabilities in HP firmware (2022) <sup>76</sup>	<p><b>Threat type:</b> Others</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat details:</b> These high severity UEFI firmware vulnerabilities effect HP laptops and desktops.</p> <p><b>Boot driver operation:</b> These UEFI firmware vulnerabilities can be exploited to locally escalate to SMM privileges.</p>
Lenovo UEFI firmware vulnerabilities (2022) <sup>77, 78</sup>	<p><b>Threat type:</b> Others</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat details:</b> These Lenovo UEFI firmware buffer overflow vulnerabilities can be exploited to perform privilege escalation in infected systems.</p> <p><b>Boot driver operation:</b> These vulnerabilities achieve arbitrary code execution to disable security-related features and hijack the operating system execution flow.</p>
iLOBleed (2021) <sup>79</sup>	<p><b>Threat type:</b> APT wiper</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat details:</b> An attack that tampers with firmware modules and completely wipes data from infected systems.</p> <p><b>Boot driver operation:</b> The malicious firmware disables DSE then loads an unsigned driver, which injects the payload into user-mode processes.</p>
ESpecter (2021) <sup>80</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat details:</b> This campaign bypasses the DSE to load a malicious unsigned driver and perform espionage activities.</p> <p><b>Boot driver operation:</b> The malicious firmware patches the Windows Boot Manager to disable the DSE and load an unsigned driver.</p>
FinSpy (2021) <sup>81</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat details:</b> A spyware that is used for surveillance purposes and is distributed through a single-stage installer.</p> <p><b>Boot driver operation:</b> The malicious UEFI loads the original UEFI and patches it in memory. The patched UEFI hooks "PsCreateSystemThread," decrypts the next stage, and executes it.</p>

Threat Name	Context
MosaicRegressor (2020) <sup>82</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat details:</b> This campaign targets diplomats and NGO members from Africa, Asia, and Europe. It is aimed at espionage and data gathering.</p> <p><b>Boot driver operation:</b> The UEFI firmware is used for persistence and for deploying malicious code that will be run after the operating system is loaded.</p>
Lojax (2018) <sup>83</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat Details:</b> A campaign that uses a backdoor to target government organizations in the Balkans as well as in Central and Eastern Europe.</p> <p><b>Boot Driver Operation:</b> The malicious UEFI/BIOS makes a copy of the legitimate file <i>autochk.exe</i> then replaces it with malicious user-mode components.</p>
MyKings (2018) <sup>84, 85</sup>	<p><b>Threat type:</b> Cryptomining</p> <p><b>Technique:</b> MBR/VBR/IPL</p> <p><b>Threat details:</b> It is a botnet that typically delivers cryptominers and RATs.</p> <p><b>Boot driver operation:</b> The bootkit used to avoid detection and establish persistence that is difficult to remove or mitigate. This is done by hooking interrupt 15h and creating a thread in the kernel for its shellcode. This shellcode creates device names used in hundreds of antivirus products to prevent them from loading.</p>
Platinum (2017) <sup>86</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> Unified Extensible Firmware Interface (UEFI)</p> <p><b>Threat details:</b> This campaign targets victims from South and Southeast Asia that uses a multistage malware infrastructure in its attacks. to steal sensitive intellectual property.</p> <p><b>Boot driver operation:</b> It uses a file-transfer tool that uses the Intel® Active Management Technology (AMT) Serial-over-LAN (SOL) channel for communication.</p>
rkloader (2015) <sup>87</sup>	<p><b>Threat type:</b> APT espionage</p> <p><b>Technique:</b> UEFI</p> <p><b>Threat details:</b> The malicious actors use an UEFI BIOS rootkit to enable their Remote Control System (RCS) agent to stay within infected systems despite hard drive formats or the reinstallation of the Windows operating system.</p> <p><b>Boot driver operation:</b> The UEFI BIOS rootkit is used for persistence.</p>

Table 3. List of threats belonging in the third cluster

# Are the First Cluster Threats Still Relevant?

Historically, the threats belonging in the first cluster were the most dominant in terms of numbers. The BYOVD is commonly used because it allows malicious actors to find new vulnerabilities in third-party Windows kernel drivers of all types across the whole software stack. This cluster can be used to piggyback a custom malicious code into the Windows kernel or to disable security features and hinder KMCS from loading other modules into the kernel.

Microsoft stated that they see around a million unique driver hashes through their telemetry every month. These drivers can contain any number of vulnerabilities, and this excludes the malicious kernel-mode code that malicious actors can implant in any of them. With minimal reversing efforts, finding zero-day vulnerabilities can be trivial in OEM (Original Equipment Manufacturer) and third-party drivers. Vulnerabilities have been found in kernel drivers' modules from the biggest software vendors, including Microsoft, Intel, and NVIDIA.

The advantage of this cluster from a malicious actor's perspective is that it only requires a few primitives to escalate privileges. On the victim's side, preventing and detecting first cluster threats can be difficult because there is a need for backward compatibility for the vulnerable drivers. Blocking such drivers might even affect the system startup in case a vulnerable boot driver affecting the system boot process is abused. However, these threats come at a cost as they are known to have some compatibility and stability issues across different operating system versions depending on the found primitives.

According to Microsoft's documentation, administrator-to-kernel — or the abuse of kernel drivers that require administrator privileges to get access to the kernel — is not a security boundary.<sup>88</sup> This means that from Microsoft's perspective, the kernel boundary is defined as a non-administrative process isolated from the kernel space. A vulnerable kernel driver can be abused to allow a non-administrative user process to violate this boundary.

In other cases, the vulnerable driver might also be exposing a generic interface while restricting its device interface to system or administrator user accounts. This means that the vulnerable driver only allows administrators to communicate with its interface. Microsoft and a significant number of software vendors



would not consider this as a vulnerability per se, but malicious actors could still weaponize this capability to load their malicious kernel modules that can tamper with protected security agents to impair defenses. This pushes security solutions to rely on anti-tampering techniques to ensure that their critical kernel code is not manipulated by malicious code via this route.

An example of how ransomware operators abuse exposed generic interfaces is the DoppelPaymer's using of the Process Hacker kernel driver to kill AV services. DoppelPaymer is a part of a larger trend of various actors using this kind of kernel drivers to disable AV/EDR functionalities.<sup>89</sup>

Microsoft reacted to the surge of threats in this cluster and started to mitigate it by maintaining a blocklist for vulnerable drivers.<sup>90</sup> This blocklist is enforced by a core component in newer Windows versions that used HCVI trustlet, a VBS-based technology. This shortens the lifespan of a vulnerable driver dramatically and makes it extremely difficult to bypass the HCVI-enforced blocklist.

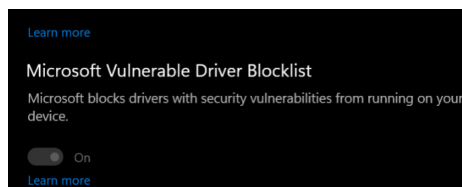


Figure 19. Microsoft enforces a vulnerable kernel drivers' blocklist via HCVI VBS technology

Trend Micro solutions actively monitor all vulnerable kernel drivers that are observed to be used by malicious actors and those that have been disclosed by the cybersecurity community. Figure 17. Shows some examples of the common vulnerable drivers that are shared in some game hacking forums. These vulnerable drivers can pose a security risk because threat actors will weaponize them in their attacks to gain a kernel foothold.

Code	Name	Signer	Description	SHA256
1.	ADVE64DRV.sys	"FUJITSU LIMITED "		04A85E3595250662338CAE86C1E5981D74A98D128920E8067583723DC1E03162
2.	Agent64.sys	"esupport.com, Inc."	DriverAgent Direct I/O for 64-bit Windows	89f992c4d192cf994462a5c100a0043c4502459909f9c89201865a27748
3.	Agent64.sys	Phoenix Technologies Ltd	DriverAgent Direct I/O for 64-bit Windows	4045AE7785981D8F13972451972EAE6F3C97BEA423E9E78F1C2F14330CD47CA
4.	Agent64.sys	Phoenix Technologies Ltd	DriverAgent Direct I/O for 64-bit Windows	6948480954137987A08E26C24CF594390960242D75F094CDEAA52E7A54FA
5.	Agent64.sys	"esupport.com, Inc"	DriverAgent Direct I/O for 64-bit Windows	8C862C5D411480E416014F808D1FD033FD4D2B0504C808908E8E86992A382058F
6.	Agent64.sys	"esupport.com, Inc."	DriverAgent Direct I/O for 64-bit Windows	81D9623325482D8821808E2D1AE33199669F7664B1078F1AD4984109414
7.	ALSys1064.sys	Artur Liberman	ALSYSIO	7196187F81EF8D1088380D3782AF8EFD0E83C4F6EFD3785DC114C609147216D
8.	ALSys1064.sys	Artur Liberman	ALSYSIO	7F375639A0DF7F51E5518CF87C3F513C58C117D8A7D28A8C615642E8188FA
9.	amf1drv64.sys	"American Megatrends, Inc."		42579A759F3F95F20A2C51D5AC2047A266A2675B3F89F46C1ED7723932A0F00
10.	AsIO.sys	ASUSTEK Computer Inc.	ASRock IO Driver	2D4330A2088409EFC51118445A824F11E0851CF30053802053785097FE40E
11.	AsIO.sys	ASUSTEK Computer Inc.	ASRock IO Driver	436CC86F62FA2D29827916E054ADE7ACE48583DE1D3E5C6C2D3DE871480E7
12.	AsIO.sys	ASUSTEK Computer Inc.	ASRock IO Driver	84D47EA790920A4531E3DF5A48480721B7FEA6849A35679F0652F1E590422602
13.	AsIO.sys	ASUSTEK Computer Inc.	ASRock IO Driver	0D6F2883F772AB8EE5904664435108791631E9C4CD81F178E5A9895995608
14.	AsrAutoChkUpdDrv.sys	ASROCK Incorporation	ASrAutoChkUpdDrv Driver	2A11808747F81E28D2E4493F50616968C703E135940521628388E4662FC4
15.	AsrDrv10.sys	ASROCK Incorporation	ASRock IO Driver	EC0A900EA089E730741499614C0917432246CE8E51599E3A18B672F402C
16.	AsrDrv101.sys	ASROCK Incorporation	RW-Everything Read & Write Driver	F4043548838984FB38945CA21A8325A51E1B5F80F045A8019748D0EC66056A8B
17.	AsrDrv101.sys	ASROCK Incorporation	RW-Everything Read & Write Driver	2A652D6868005AD92376AD3230218500A82C653A8F06E0F26120F771481E08A
18.	AsrDrv101.sys	ASROCK Incorporation	RW-Everything Read & Write Driver	95040C772021CE2E6018A2114F8E8D0A580F77E7873A4059905F52A160C9
19.	AsrRapidStartDrv.sys	ASROCK Incorporation	RW-Everything Read & Write Driver	0AAFA9F47ACFD946C9542985994F5321F00842A28DF239604930767483CB
20.	AsrSmartConnectDrv.sys	ASROCK Incorporation	RW-Everything Read & Write Driver	47F08F7D300824A8F488A98916401A37C0FD8502D308A8A91FE31128892DCC
21.	AsUpIO.sys	ASUSTEK Computer Inc.	ATI Diagnostics Hardware Abstraction Sys	8944E46A5D80FEDD1837EAD9589F9F9EFD41E801AD73D5185DC08627F8CF
22.	atill164.sys	"ATI Technologies, Inc"	ATI Diagnostics Hardware Abstraction Sys	5C04C27A4708C9A7093E338E340E61190C2927A767A1006493996F87360A
23.	BS_Def64.sys	ASUSTEK Computer Inc.	Default BIOS Flash Driver	004015302888E27E84F1EAC6855039E1A057370F5E8C615724F5A218DA0A3
24.	BS_Def64.sys	ASUSTEK Computer Inc.	Default BIOS Flash Driver	3326E2D3288A069FE86024809AFC56C7E39241E8E70A53728C7E80995422A5
25.	BS_Def64.sys	ASUSTEK Computer Inc.	Default BIOS Flash Driver	3689E31240A80341873C7092863E2E0F2C482962E0F9825271C3A121687669EB
26.	CTIDRV_A064.sys	IBM Polska Sp. z o.o.		29E0824017A8387F2527A7608A80F4D54C0E97680276C2590A381304

Figure 20. Vulnerable kernel drivers' information published on a game hacking forum

Vulnerable Driver Megathread	
Collection of signed system drivers that let you read/write privileged memory or expose some other serious vulnerability. If the driver has input buffer validation for IOCTL codes please state so before submitting to the list.	
<b>ASUS</b>	<p><b>EIO64.sys</b> MmMapIoSpace/MmUnmapIoSpace  <b>IOMap64.sys</b> MmMapIoSpace/MmUnmapIoSpace</p> <p><b>ATSZIO64.sys</b> ZwMapViewOfSection/ZwUnmapViewOfSection/MmGetPhysicalAddress</p> <p>Device Name: "\\.\ATSZIO"  Map Physical IOCTL: 0x8807200C  Unmap Physical IOCTL: 0x88072010</p> <p>Example: <a href="https://[redacted]">https://[redacted]</a></p>
<b>ATI</b>	<p><b>atilk64.sys</b> MmMapIoSpace/MmUnmapIoSpace/MmBuildMdlForNonPagedPool/MmMapLockedPages</p> <p>Device Name: "\\.\atilk64"  Map/Unmap IOCTLs: 0x9C402534, 0x9C402538, 0x9C402544, 0x9C402548  MDL IOCTLs: 0x9C40254C, 0x9C402558, 0x9C402560, 0x9C402564</p>
<b>Avast</b>	<p><b>aswVmm.sys</b> SSDT Hooking</p> <p>Device Name: "\\.\aswVmm"  Hook IOCTL: 0xA000E804</p> <p>Example: <a href="https://[redacted]">https://[redacted]</a></p>
<b>Biostar</b>	<p><b>BS_Flash64.sys</b> MmMapIoSpace/MmUnmapIoSpace/MmMapLockedPages/ExAllocatePoolWithTag/ExFreePoolWithTag</p> <p>Device Name: "\\.\BS_Flash64"  Map/Unmap IOCTL: 0x222000  Allocate IOCTL: 0x22203C</p> <p><b>BS_I2c64.sys</b> MmMapIoSpace/MmUnmapIoSpace  <b>BSMEMx64.sys</b> MmMapIoSpace/MmUnmapIoSpace/MmGetPhysicalAddress  <b>BSMIXP64.sys</b> MmMapIoSpace/MmUnmapIoSpace/MmGetPhysicalAddress</p>
<b>Capcom</b>	<p><b>Capcom.sys</b> MmGetSystemRoutineAddress</p> <p>Device Name: "\\.\Htsysm72FB"  Execute IOCTL: 0xAA013044</p>

Figure 21. A list of vulnerable kernel drivers per vendor

As previously mentioned, we expect that this cluster will decrease over time due to the increase in the adoption of virtualization-based security features that can help mitigate multiple exploitation techniques using vulnerable kernel drivers.<sup>91</sup> In addition, the enforcement of stricter rules that are compatible with VBS technology in the kernel driver development process will help eliminate certain bug classes. Third-party security vendors might also deploy their own custom hypervisors similar to Microsoft's native hypervisor used by VBS. This will allow their security solutions to provide better protection against first cluster threats.

# The Second Cluster APT Case Study

As discussed in the “How Mature Malicious Actors Adapted to KMCS” section, second cluster attacks comply with the code-signing requirements that Microsoft requires. This gives malicious actors the flexibility to compile kernel modules designed for very specific tasks and sign them with customized drivers based on the current Microsoft kernel modules’ signing rules. Under this cluster, attackers can do one of the following approaches:

1. Use a code-signing certificate that was leaked, stolen from a compromised environment, or purchased from the underground market.
2. Obtain a new valid code-signing certificate by impersonating a legitimate entity and following Microsoft’s process for getting the cross-signing certificate (back when Microsoft still allowed cross-signing for kernel-mode code), abusing Microsoft’s portal for issuing signed kernel modules, and purchasing valid code-signing certificates and/or EV (Extended Validation) certificates that are tied to real identities from the underground market.

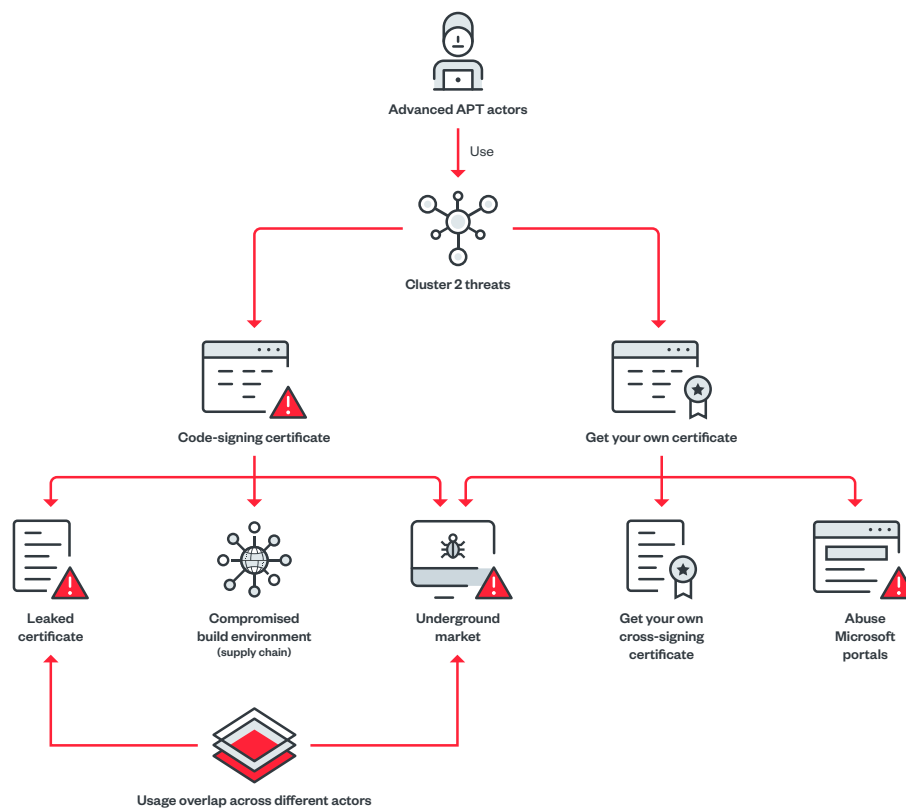


Figure 22. A diagram showing the two approaches malicious actors can take when launching threats belonging in the second cluster

# APTs Use Code-signing Certificates

Based on our analysis of historical data and intelligence feeds, Earth Baku or APT41 is one of the actors that regularly used code-signing certificates to sign malware used in their campaigns. Most of the digital certificates they used were valid, unrevoked digital certificates stolen from game development studios. According to an advertisement in an underground marketplace post, the success rate of installing a payload increases by 50% when signing files with valid digital certificates. We identified Earth Baku's objectives in opting to sign their malware arsenal in the user space or kernel space:

- Load kernel modules in an environment with KMCS
- Ensure compatibility with targeted systems and to potentially avoid detection
- Significantly decrease the likelihood that a malicious payload would be detected and attract less suspicion even when malicious payloads are detected
- Circumvent automated scanning security solutions and bypass Windows group policies that restrict unsigned code

## Stolen Code-Signing Certificates

The first approach under the second cluster combines the benefits of buying a new certificate with having an increased level of anonymity. Instead of malicious actors buying their own code-signing certificate, they can just use someone else's code signing certificate. Also, previous research efforts have also shown that the Windows kernel driver loader will still load kernel drivers regardless if it is signed with expired or revoked certificates — several of which are already publicly posted especially in gaming forums. Moreover, we observed that several exposed Amazon Simple Storage Service (Amazon S3) buckets that could be scanned for private keys in specific file extensions, such as .pfx and .p12, using online scanner services including GrayhatWarfare.<sup>92</sup> This could eventually be used for malicious abuses.

It is highly possible that advanced threat actors adopt these two approaches to gain code-signing capabilities:

- Directly targeting software vendors
  - Targeting build systems to have a malicious code signed without stealing the actual certificate (supply chain compromise)
- Stealing code-signing certificates from their build systems
  - Using underground marketplaces services and public leaks

During a supply chain attack, malicious actors can compromise a software vendor's build environment to inject signed malicious code before it is compiled. Our investigation only found user-mode applications

that have been signed using this approach. We have also observed that advanced cybercrime group APT41 signs malicious updates with legitimate certificates. In this case, all updates were required to be signed by the breached entity. This means that APT41 had to use the code-signing certificate to subvert the update mechanism. Our analysis revealed that APT41 injected malicious code into the package prior to compilation, circumventing the need to steal the code-signing certificate, after which APT41 compiled the package on their own.

Although this same approach is theoretically possible to use for signing kernel-mode code if the underlying build environment is compromised, we were unable to find any kernel modules in our analysis.

In APT41 operations that targeted the video game industry in 2018, the group used the same approach and accessed production environments to inject malicious code into legitimate video game files to distribute malware. The files were signed with valid code-signing certificates and were widely distributed to end users. This most likely indicates that APT41 had access to the victims' production environments, which facilitated a supply chain compromise and the signing of malicious files using legitimate digital certificates from the same compromised organization. The group's distinct use of supply chain compromises and its consistent use of compromised code-signing certificates show just how creative and well-resourced these threat actors are.

APT41 is known for using stolen digital certificates from video game studios to sign their malware components including malicious kernel drivers. The group was previously reported to have abused at least 19 stolen code-signing certificates.

The problem in mitigating the threat of stolen legitimate code-signing certificates is that it is difficult to just completely block all the executables that were signed with leaked or stolen code-signing certificates. It is highly likely that these certificates were published years ago and have been used to sign multiple legitimate modules during its entire lifetime.

A wide block strategy for stolen certificates is not a very practical resolution, which is why most malicious drivers that have been signed with leaked certificates have low coverage by most antivirus solutions. Also, certificate authorities are the ones responsible for revoking compromised digital certificates. Because of this, response times can vary, and digital certificates can still be continuously abused long after they are first identified to have been misused. Even if these code-signing certificates become expired or revoked, they are still usable in kernel module signing activities.

## Overlaps in Using Stolen Certificates

By abusing the trust between software vendors and certificate authorities, malicious actors can steal private keys by compromising an organization's infrastructure to access and steal code-signing certificates.

For example, APT41 used a code-signing certificate from one game publisher against other gaming

industry entities. Another digital certificate from the same breached organization was used by several other APT operators, including other China-based cyberespionage groups. This shows that there is an overlap in the use of stolen certificates and how different threat actors share them with one other.

We observed another pattern related to stolen code-signing certificates used by different cybercrime groups distributing cryptocurrency miners. Different malicious actors, such as those behind the FiveSys rootkit and the Purple Fox malware, use stolen signature information to identify and block one another.<sup>93</sup>

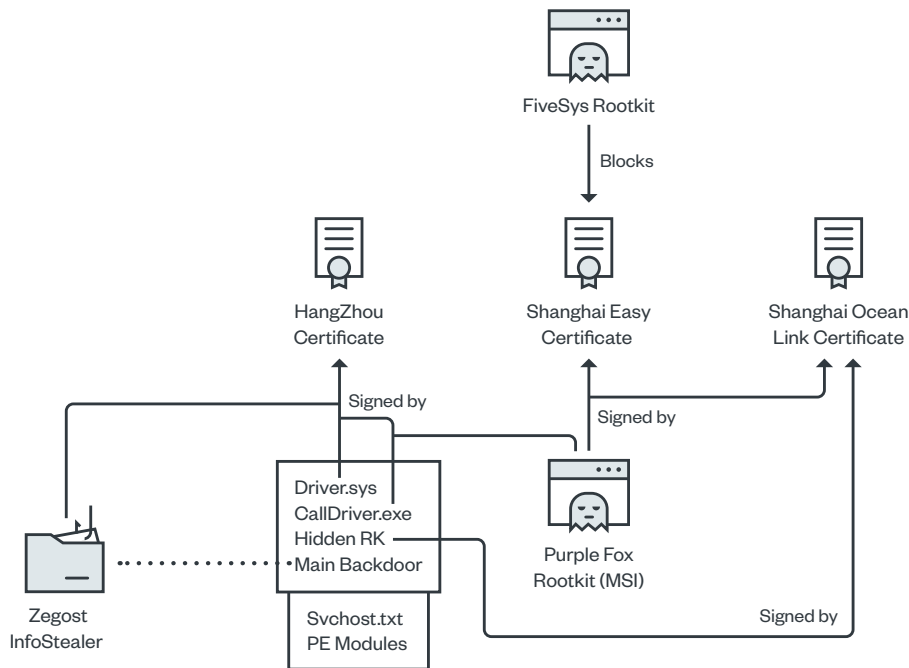


Figure 23. FiveSys operators block stolen code-signing certificates used by Purple Fox operators

Source: Trend Micro Research, News, and Perspectives<sup>94</sup>

The malware authors have left debug messages revealing the list of signatures it monitors:

00000478	186.51918030	[MY-1]MD5-0:9D9F343EAA8FB4045A4B7D05437AC02B
00000480	186.51918030	[MY-1]MD5-1:A269121725987B766740D43964E83CF3
00000482	186.51918030	[MY-1]MD5-2:698FD84F0AABAA65F8BD3E7AD417F4D4
00000484	186.51919556	[MY-1]MD5-3:CE7D7EE076A74D3C532265D8F6BBFF09
00000486	186.51919556	[MY-1]Sign-0:Zhang Zhengqi
00000488	186.51921082	[MY-1]Sign-1:Haining shengdun Network Information Technology Co., Ltd
00000490	186.51921082	[MY-1]Sign-2:SHENZHEN LIRINUOS
00000492	186.51921082	[MY-1]Sign-3:Shanghai easy kradar Information Consulting Co.Ltd

Figure 24. The FiveSys Rootkit's block list includes the Shanghai easy kradar code-signing certificate used by Purple Fox operators.

Source: Trend Micro Research, News, and Perspectives<sup>95</sup>

# Acquiring New Code-signing Certificates or Valid Signatures

The second approach under this cluster involves malicious actors proactively acquiring code-signing certificates on their own as part of the resource development phase (T1587.002). State-sponsored threat actors who can afford this complex process are typically the ones who use this approach in their campaigns.

An example of this second approach happened in 2021, when malicious actors submitted malicious third-party-built drivers for certification through the WHCP portal.<sup>96</sup> In December 2022, Mandiant published a report stating how malicious actors abused the Windows Hardware Compatibility Program to sign malicious drivers to kill EDR agents on endpoints.<sup>97</sup>

This approach comes with a relatively high cost compared to other techniques. Buying one's own code-signing certificate might be a good option when it comes to targeted attacks and red teaming activities. However, purchasing a code-signing certificate is not fool-proof, as these certificates can be fully revoked by the certificate authority (CA) and will also require buyers to disclose their identities.

Figure 23 shows the evolution of Microsoft's requirements for successfully signing and loading a kernel module. An operating system milestone can be seen in the release of Windows 10, which was when it stopped the use of cross-signing certificates and negatively affected malicious actors who used such certificates in their low-level attacks. Since the acquired certificate is cross-signed by a certificate authority that Microsoft trusts, this ensures that no malicious actor could tamper with the kernel code and that Microsoft trusts that you are who you claim you are.

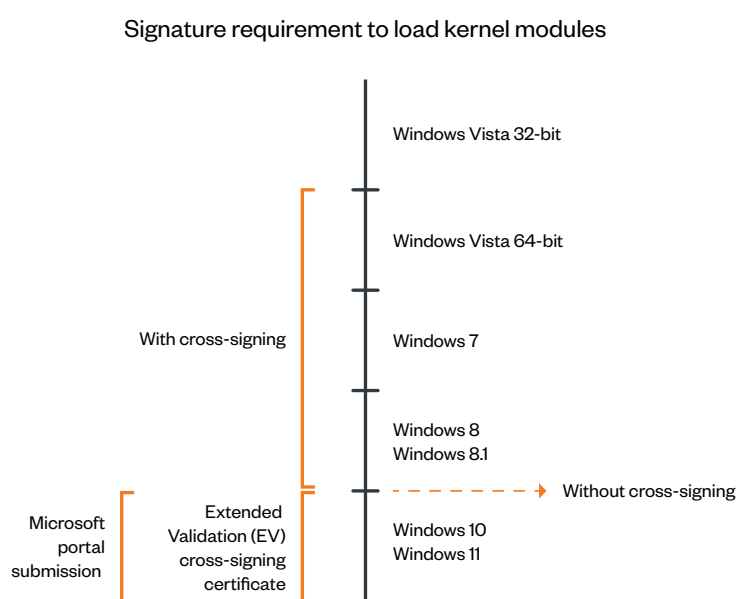


Figure 25. Microsoft kernel code signing requirements

Operating system	Signature requirement
Windows XP	None
Windows Vista 32-bit	None
Windows Vista 64-bit	Microsoft Code Verification Root (MCVR0 certificate and SHA-1
Windows 7	MCVR and (SHA 1 or KB3033929) & SHA-2
Windows 8, 8.1	MCVR and SHA-256
Windows 10, 11	MCVR, SHA-256, and Portal

Table 4. Windows kernel code-signing conditions

Requirement	Description
<b>MCVR</b>	In the tables above, MCVR means the kernel module signature's chain of trust must go back to the Microsoft Code Verification Root certificate, or some other certificate that the kernel trusts. Any signature that goes through the <b>WHQL</b> process should already satisfy this requirement. This means that the kernel does not have access to the Trusted Root Certification Authorities list. <b>A cross-signing certificate is typically needed to satisfy this requirement.</b>
<b>Portal</b>	Microsoft announced on April 1, 2015, that all new Windows 10 kernel mode drivers must be submitted to and digitally signed by the <b>Windows Hardware Developer Center Dashboard portal</b> . For backwards compatibility, Windows 10 will still allow kernel mode drivers with signatures from older certificates under certain conditions, but this requires an older certificate, so it is not very practical. <b>The portal only accepts driver submissions that are already signed with an EV certificate</b> , which is typically more expensive than a normal certificate.
<b>SHA-1</b>	A signature must be present and it must not use SHA-256. This applies to the signature of the kernel driver itself and the signatures that secure the chain of trust to a certificate.
<b>SHA-256</b>	SHA-1 will eventually not be a trusted hash in Windows. The operating system will use SHA-256 (or higher) for everything, including the file digest, main certificate, timestamp digest, and timestamp certificate.

Table 5. Description of each kernel code-signing requirement



By requiring kernel driver modules to be submitted to their portal, Microsoft proves that they want to ensure the quality of the code that will be loaded in the kernel. Microsoft will only add their signature if all of their requirements are met. For this purpose, an additional EV code-signing certificate will be needed to send a submission to their portal. The EV code-signing certificate is expensive and will complicate the process for malicious actors who are thinking of abusing this process, as it comes with a USB hardware token that is difficult to copy.

## Underground Marketplaces (Certificate-as-a-Service)

EV certificates are a popular commodity across underground markets, especially since these EV certificates can be used by malicious actors to load malicious kernel modules into recent Windows releases and bypass most security solutions. These can assist threat actors in conducting bigger attacks that include a legitimately signed kernel module.

This section shows an example of a malicious actor that specializes in selling EV certificates in underground marketplaces. The actor offers EV code-signing certificates issued by the DigiCert and Sectigo digital security companies, and allegedly could provide a certificate issued from a custom publisher name for a higher price. This actor maintains a positive reputation on the underground forum with multiple positive reviews at the time of writing. The malicious actor most likely used either fake companies or stolen information from real companies to obtain certificates from certificate authorities.

HermeticWiper is a recent example of a threat that confirms the use of legitimate company data to obtain a code-signing certificate.<sup>98, 99</sup> This malware had been signed using Hermetica Digital's digital certificate, which is a small business specializing in videogame design. The company reported that they had nothing to do with the attack, stating that they never sought a digital certificate and had no idea one had been issued to them.

The EV certificates that malicious actors offered copied them on USB tokens, which they offered to ship to customers. Code-signing certificates issued with custom company names cost US\$15,000, based on the claims of some malicious actors. Malicious actors also claimed that these certificates' life spans depended on how they were used: some certificates could last from a week to a year.

We are selling EV code-signing certificates available in stock or have them issued to order.  
EV Code Signing Certificate for sale.

⚡ Here's a Telegram channel where we will publish information on available certificates. Join it.  
Created a channel where information on the sale of certificates will be published. Follow.

[https://](https://t.me/...)

#### Item and cost

EV Code Signing certificate: USD 4,000 and up (it takes 2 to 4 weeks to get one issued)  
EV Code Signing Certificate - from \$ 4000 (production time 2-4 weeks)

Having the certificate put on our USB Token and delivered within Russia costs USD 350.

- \* Get a USD 50 discount for a review.
- \* EV certificates require a physical USB token.
- \* 50 \$ discount per review
- \* EV certificates require a physical USB token.

#### Tokens that are suitable for EV certificates:

USB tokens for EV certificate:

- SafeNet eToken 5100
- SafeNet eToken 5105
- SafeNet eToken 5110
- SafeNet eToken 5200
- SafeNet eToken 5205
- SafeNet eToken PRO 72K
- SafeNet eToken Pro Anywhere
- iKey 4000

A certificate can be issued in a specific company name. The conditions are negotiated on a case-by-case basis.  
Registration of code signing certificates for a specific company name is possible, conditions are discussed individually.

#### Contact details

Telegram: @

(Add a message with the username to your 'Favorites' and follow the link. Thus, you will be able to avoid fake accounts.)

Jabber: provided on request

Contact us; we'll be glad to cooperate with you!

We will be glad to cooperation!

#### What is a certificate for?

EV Code Signing certificates are specialized digital certificates that can be used to sign software, various scripts, applications, macros, etc in order to avoid copyright issues.  
- [The certificates are used] for executable x86/x64 files (.exe, .cab, .dll, .msi)  
- for drivers, including the ones for Windows 10  
- for VBA macros and JavaScript scripts

Figure 26. An underground marketplace post selling EV certificates

Aside from purchasing EV certificates from underground marketplaces, malicious actors also abuse unsecured hardware tokens to get EV certificates, impersonate the identity of the company, and use them in their attacks.

By analyzing related services offered in underground forums and channels, we found that the most common service was selling code-signing certificates and EV certificates.

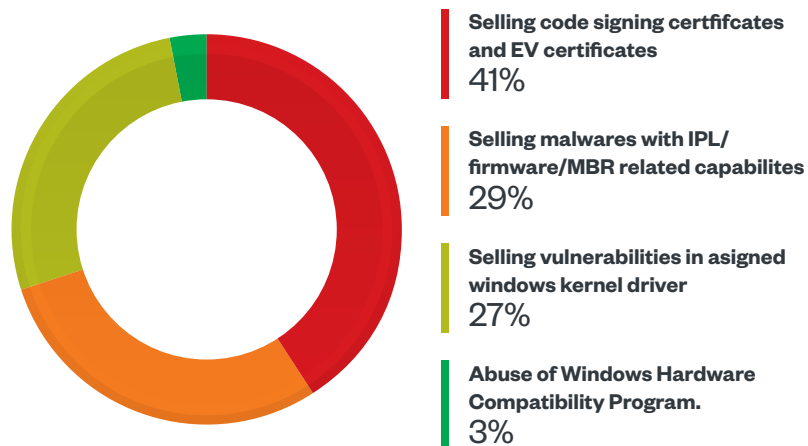


Figure 27. An analysis of related services offered in underground forums and channels

# Conclusion and Future Predictions

From the analysis of the major threats that affected the Windows kernel trust model over the past seven years, we noticed that the threats that involve low-level components are evolving rather than completely disappearing. The transformation of these threats is mainly induced by the innovative modern security mechanisms that were built into the modern Windows kernel. Despite these advancements in technology, based on our analysis, it still only takes one bypass to compromise the whole software stack. Hence, it is important to understand that these built-in technologies are not bulletproof when it comes to detecting and thwarting new threats that are continuously progressing alongside the operating system.

Advanced and mature threat actors are still actively seeking high-privilege access to the Windows operating system. And these attackers use techniques that attempt to combat the increased protection on userland processes via endpoint protection platform (EPP) and endpoint detection and response (EDR) technologies on users' desktop or servers. Because of these added layers of protection, attackers tend to opt for the path of least resistance and get their malicious code running from the kernel or on a lower level. This is why we believe that such threats will not disappear from threat actors' toolkits anytime soon.

After clustering all of the published kernel threats into three main groups, our analysis showed that several clusters are more likely to become obsolete in the future. Meanwhile, as attackers focus on shifting the initial infection point in the software stack just below the defensive mechanisms to bypass security features, other clusters are expected to increase in time.

Malicious actors will continue to use rootkits to hide malicious code from security tools, impair security defenses, and fly under the radar for long periods. Rootkits will continue to be used primarily by highly qualified groups that have the skills to reverse low-level system components and the required resources to develop such tools. These groups have sufficient financial resources to buy rootkits on the dark web or buy code-signing certificates to build a rootkit. This means that the main danger of attacks involving these kinds of rootkits lie in their ability to hide complex targeted attacks that will be used early in the kill chain. This will impair defenses before the actual payloads are launched in victim environments.

# References

- 1 Trend Micro. (n.d.) *Trend Micro Security News*. “What is Rootkit?” Accessed on Sep. 30, 2022, at <https://www.trendmicro.com/vinfo/us/security/definition/rootkit>.
- 2 Stephen J. Bigelow and Jessica Lulka. (n.d.). *TechTarget*. “Kernel.” Accessed on Oct. 20, 2022, at <https://www.techtarget.com/searchdatacenter/definition/kernel>.
- 3 Windows Hardware Developer. (Dec. 15, 2021). *Microsoft*. “Windows Kernel-Mode Kernel Library.” Accessed on Oct. 3, 2022, at <https://learn.microsoft.com/en-us/windows-hardware/drivers/kernel/windows-kernel-mode-kernel-library>.
- 4 Techopedia. (July 3, 2013). *Techopedia*. “Security Through Obscurity (STO).” Accessed on Oct. 3, 2022, at <https://www.techopedia.com/definition/21985/security-through-obscurity-sto>.
- 5 Eclypsium. (June 2, 2022). *Eclypsium*. “Conti Targets Critical Firmware.” Accessed on Oct. 3, 2022, at <https://eclypsium.com/2022/06/02/conti-targets-critical-firmware/>.
- 6 Windows Hardware Developer. (Dec. 15, 2021). *Microsoft*. “Hypervisor-Protected Code Integrity (HVCI).” Accessed on Oct. 3, 2022, at <https://learn.microsoft.com/en-us/windows-hardware/drivers/bringup/device-guard-and-credential-guard>.
- 7 Charlie Osborne. (May 7, 2021). *ZDNet*. “New Moriya rootkit stealthily backdoors Windows systems.” Accessed on Oct. 3, 2022, at <https://www.zdnet.com/article/new-moriya-rootkit-stealthily-backdoors-windows-systems/>.
- 8 James Wyke. (June 6, 2012). *Naked Security by Sophos*. “Major shift in strategy for ZeroAccess rootkit malware, as it shifts to user-mode.” Accessed on Oct. 4, 2022, at <https://nakedsecurity.sophos.com/2012/06/06/zeroaccess-rootkit-usermode/>.
- 9 Ryan Soliven and Hitomi Kimura. (Aug. 24, 2022). *Trend Micro Research, News, and Perspectives*. “Ransomware Actor Abuses Genshin Impact Anti-Cheat Driver to Kill Antivirus.” Accessed on Oct. 20, 2022, at [https://www.trendmicro.com/en\\_us/research/22/h/ransomware-actor-abuses-genshin-impact-anti-cheat-driver-to-kill-antivirus.html](https://www.trendmicro.com/en_us/research/22/h/ransomware-actor-abuses-genshin-impact-anti-cheat-driver-to-kill-antivirus.html).
- 10 Microsoft Security Response Center Team. (June 25, 2021). *Microsoft Security Response Center*. “Investigating and Mitigating Malicious Drivers.” Accessed on Oct. 7, 2022, at <https://msrc-blog.microsoft.com/2021/06/25/investigating-and-mitigating-malicious-drivers/>.
- 11 Microsoft Security Response Center. (Dec. 13, 2022). *Microsoft Security Response Center*. “Guidance on Microsoft Signed Drivers Being Used Maliciously.” Accessed on Dec. 28, 2022, at <https://msrc.microsoft.com/update-guide/vulnerability/ADV220005>.
- 12 Windows Hardware Developer. (March 18, 2022). *Microsoft*. “Overview of Early Launch AntiMalware.” Accessed on Oct. 7, 2022, at <https://learn.microsoft.com/en-us/windows-hardware/drivers/install/early-launch-antimalware>.
- 13 Windows Hardware Developer. (May 7, 2022). *Microsoft*. “Secure boot.” Accessed on Oct. 7, 2022, at <https://learn.microsoft.com/en-us/windows-hardware/design/device-experiences/oem-secure-boot>.
- 14 Windows Hardware Developer. (July 7, 2022). *Microsoft*. “Virtual Secure Mode.” Accessed on Oct. 7, 2022, at <https://learn.microsoft.com/en-us/virtualization/hyper-v-on-windows/tifs/vsm>.
- 15 Dan Goodin. (Oct. 15, 2022). *Ars Technica*. “How a Microsoft blunder opened millions of PCs to potent malware attacks.” Accessed on Dec. 28, 2022, at <https://arstechnica.com/information-technology/2022/10/how-a-microsoft-blunder-opened-millions-of-pcs-to-potent-malware-attacks/>.
- 16 Dan Goodin. (Oct. 15, 2022). *Ars Technica*. “How a Microsoft blunder opened millions of PCs to potent malware attacks.” Accessed on Dec. 28, 2022, at <https://arstechnica.com/information-technology/2022/10/how-a-microsoft-blunder-opened-millions-of-pcs-to-potent-malware-attacks/>.
- 17 Hara Hiroaki and Ted Lee. (Nov. 9, 2022). *Trend Micro Research, News, and Perspectives*. Accessed on Dec. 28, 2022, at [https://www.trendmicro.com/en\\_us/research/22/k/hack-the-real-box-apt41-new-subgroup-earth-longzhi.html?ite=159715&itq=3152&itq=0da8a7fe-8a78-414e-9c0b-7f76ad466508&itx%5Bidio%5D=2098312](https://www.trendmicro.com/en_us/research/22/k/hack-the-real-box-apt41-new-subgroup-earth-longzhi.html?ite=159715&itq=3152&itq=0da8a7fe-8a78-414e-9c0b-7f76ad466508&itx%5Bidio%5D=2098312).
- 18 Ryan Soliven and Hitomi Kimura. (Aug. 24, 2022). *Trend Micro Research, News, and Perspectives*. Accessed on Dec. 28, 2022, at [https://www.trendmicro.com/en\\_us/research/22/h/ransomware-actor-abuses-genshin-impact-anti-cheat-driver-to-kill-antivirus.html](https://www.trendmicro.com/en_us/research/22/h/ransomware-actor-abuses-genshin-impact-anti-cheat-driver-to-kill-antivirus.html).

- 19 Microsoft Threat Intelligence Center (MSTIC). (July 15, 2021). *Microsoft Security*. "Protecting customers from a private-sector offensive actor using 0-day exploits and DevilsTongue malware." Accessed on Oct. 12, 2022, at <https://www.microsoft.com/security/blog/2021/07/15/protecting-customers-from-a-private-sector-offensive-actor-using-0-day-exploits-and-devilstongue-malware/>.
- 20 Jan Vojtěšek. (July 21, 2022). *Decoded*. "The Return of Candiru: Zero-days in the Middle East." Accessed on Oct. 12, 2022, at <https://decoded.avast.io/janvojtesek/the-return-of-candiru-zero-days-in-the-middle-east/>.
- 21 Bill Marczak et al. (July 15, 2021). *The Citizen Lab*. "Hooking Candiru: Another Mercenary Spyware Vendor Comes into Focus." Accessed on Oct. 12, 2022, at <https://citizenlab.ca/2021/07/hooking-candiru-another-mercenary-spyware-vendor-comes-into-focus/>.
- 22 Eduardo Mattos and Rob Homewood. (Feb. 26, 2022). *Aon's Cyber Labs*. "Yours Truly, Signed AV Driver: Weaponizing An Antivirus Driver." Accessed on Oct. 12, 2022, at [https://www.aon.com/cyber-solutions/aon\\_cyber\\_labs/yours-truly-signed-av-driver-weaponizing-an-antivirus-driver/](https://www.aon.com/cyber-solutions/aon_cyber_labs/yours-truly-signed-av-driver-weaponizing-an-antivirus-driver/).
- 23 Kasif Dekel. (May 5, 2022). *Sentinel Labs*. "Vulnerabilities in Avast And AVG Put Millions At Risk." Accessed on Oct. 12, 2022, at <https://www.sentinelone.com/labs/vulnerabilities-in-avast-and-avg-put-millions-at-risk/>.
- 24 Aliakbar Zahravi and Leandro Froes. (May 5, 2022). *Trend Micro Research, News, and Perspectives*. "NetDooka Framework Distributed via PrivateLoader Malware as Part of Pay-Per-Install Service." Accessed on Oct. 12, 2022, at [https://www.trendmicro.com/en\\_us/research/22/e/netdooka-framework-distributed-via-privateloader-ppi.html](https://www.trendmicro.com/en_us/research/22/e/netdooka-framework-distributed-via-privateloader-ppi.html).
- 25 Christopher Ordonez and Alvin Nieto. (May 2, 2022). *Trend Micro Research, News, and Perspectives*. "AvosLocker Ransomware Variant Abuses Driver File to Disable Antivirus, Scans for Log4shell." Accessed on Oct. 12, 2022, at [https://www.trendmicro.com/en\\_us/research/22/e/avoslocker-ransomware-variant-abuses-driver-file-to-disable-anti-virus-scans-log4shell.html](https://www.trendmicro.com/en_us/research/22/e/avoslocker-ransomware-variant-abuses-driver-file-to-disable-anti-virus-scans-log4shell.html).
- 26 Andrew Brandt and Mark Loman. (Feb. 6, 2020). *Sophos News*. "Living off another land: Ransomware borrows vulnerable driver to remove security software." Accessed on Oct. 12, 2022, at <https://news.sophos.com/en-us/2020/02/06/living-off-another-land-ransomware-borrows-vulnerable-driver-to-remove-security-software/>.
- 27 Trend Micro Research. (Jan. 5, 2021). *Trend Micro Research, News, and Perspectives*. "An Overview of the DoppelPaymer Ransomware." Accessed on Oct. 12, 2022, at [https://www.trendmicro.com/en\\_us/research/21/a/an-overview-of-the-doppelpaymer-ransomware.html](https://www.trendmicro.com/en_us/research/21/a/an-overview-of-the-doppelpaymer-ransomware.html).
- 28 Check Point Research. (Nov. 15, 2021). *Check Point Research*. "Uncovering MosesStaff techniques: Ideology over Money." Accessed on Oct. 12, 2022, at <https://research.checkpoint.com/2021/mosesstaff-targeting-israeli-companies/>.
- 29 Mark Lechtik et al. (Sep. 30, 2021). *Securelist by Kaspersky*. "GhostEmperor: From ProxyLogon to kernel mode." Accessed on Oct. 12, 2022, at <https://securelist.com/ghostemperor-from-proxylogon-to-kernel-mode/104407/>.
- 30 Kaspersky. (July 21, 2021). *Kaspersky*. "GhostEmperor APT targets high-profile victims using unknown rootkit." Accessed on Oct. 12, 2022, at [https://usa.kaspersky.com/about/press-releases/2021\\_ghostemperor-apt-targets-high-profile-victims-using-unknown-rootkit](https://usa.kaspersky.com/about/press-releases/2021_ghostemperor-apt-targets-high-profile-victims-using-unknown-rootkit).
- 31 TrendMicroRSRCH. (May 28, 2021). *Twitter*. "[1/5] We spotted Motocos #ransomware using @telegram..." Accessed on Oct. 25, 2022, at <https://twitter.com/TrendMicroRSRCH/status/1398270334068011016>.
- 32 Mark Lechtik and Giampaolo Dedola. (May 6, 2021). *Securelist by Kaspersky*. "Operation TunnelSnake." Accessed on Oct. 12, 2022, at <https://securelist.com/operation-tunnelsnake-and-moriya-rootkit/101831/>.
- 33 Daniel Lunghi and Kenney Lu. (April 9, 2021). *Trend Micro Research, News, and Perspectives*. "Iron Tiger APT Updates Toolkit With Evolved SysUpdate Malware." Accessed on Oct. 12, 2022, at [https://www.trendmicro.com/en\\_us/research/21/d/iron-tiger-apt-updates-toolkit-with-evolved-sysupdate-malware-va.html](https://www.trendmicro.com/en_us/research/21/d/iron-tiger-apt-updates-toolkit-with-evolved-sysupdate-malware-va.html).
- 34 Microsoft Threat Intelligence Center (MSTIC) and Microsoft Defender Threat Intelligence. (Jan. 28, 2021). *Microsoft Security*. "ZINC attacks against security researchers." Accessed on Oct. 12, 2022, at <https://www.microsoft.com/security/blog/2021/01/28/zinc-attacks-against-security-researchers/>.
- 35 Zuzana Hromcová and Anton Cherepanov (n.d.). *WeLiveSecurity by ESET*. "INVISIMOLE: THE HIDDEN PART OF THE STORY UNEARTHING INVISIMOLE'S ESPIONAGE TOOLSET AND STRATEGIC COOPERATIONS." Accessed on Oct. 12, 2022, at [https://www.welivesecurity.com/wp-content/uploads/2020/06/ESET\\_InvisiMole.pdf](https://www.welivesecurity.com/wp-content/uploads/2020/06/ESET_InvisiMole.pdf).
- 36 Dominik Reichel and Esmid Idrizovic. (June 17, 2020). *Unit42*. "AcidBox: Rare Malware Repurposing Turla Group Exploit Targeted Russian Organizations." Accessed on Oct. 25, 2022, at <https://unit42.paloaltonetworks.com/acidbox-rare-malware/>.

- 37 imor Kessem. (Dec. 4, 2019). *Security Intelligence*. "New Destructive Wiper ZeroCleare Targets Energy Sector in the Middle East." Accessed on Dec. 14, 2022, at [https://securityintelligence.com/posts/new-destructive-wiper-zeroclare-targets-energy-sector-in-the-middle-east/?fbclid=IwAR1Del8GX\\_nvhrVhSkbWTU7hDE-pnEflSbAVBhYdJAG9ykCTztOaph3-OU](https://securityintelligence.com/posts/new-destructive-wiper-zeroclare-targets-energy-sector-in-the-middle-east/?fbclid=IwAR1Del8GX_nvhrVhSkbWTU7hDE-pnEflSbAVBhYdJAG9ykCTztOaph3-OU).
- 38 Talos Intelligence. (Sep. 26, 2019). *Talos Intelligence*. "Divergent: "Fileless" NodeJS Malware Burrows Deep Within the Host." Accessed on Oct. 11, 2022, at <https://blog.talosintelligence.com/2019/09/divergent-analysis.html?m=1>.
- 39 Ori Damari. (Nov. 13, 2019). *Low Level Pleasure*. "Abusing Signed Windows Drivers." Accessed on Oct. 11, 2022, at <https://repnz.github.io/posts/abusing-signed-drivers/>.
- 40 Microsoft Defender Security Research Team. (March 25, 2019). *Microsoft Security*. "From alert to driver vulnerability: Microsoft Defender ATP investigation unearths privilege escalation flaw." Accessed on Oct. 11, 2022, at <https://www.microsoft.com/security/blog/2019/03/25/from-alert-to-driver-vulnerability-microsoft-defender-atp-investigation-unearths-privilege-escalation-flaw/>.
- 41 Kaspersky Content Hub. (March 6, 2018). *Kaspersky*. "The Slingshot APT." Accessed on Oct. 11, 2022, at [https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/09133534/The-Slingshot-APT\\_report\\_ENG\\_final.pdf](https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2018/03/09133534/The-Slingshot-APT_report_ENG_final.pdf).
- 42 Chris Williams. (Sep. 23, 2016). *The Register*. "Double KO! Capcom's Street Fighter V installs hidden rootkit on PCs." Accessed on Oct. 11, 2022, at [https://www.theregister.com/2016/09/23/capcom\\_street\\_fighter\\_v](https://www.theregister.com/2016/09/23/capcom_street_fighter_v).
- 43 Novetta. (n.d.). *Novetta*. "Derusbi (Server Variant) Analysis." Accessed on Oct. 11, 2022, at <https://www.novetta.com/wp-content/uploads/2014/11/Derusbi.pdf>.
- 44 WeLiveSecurity. (May 11, 2012). *WeLiveSecurity by ESET*. "King of Spam: Festi botnet analysis." Accessed on Oct. 11, 2022, at <https://www.welivesecurity.com/2012/05/11/king-of-spam-festi-botnet-analysis/>.
- 45 Eset. (March 30, 2017). *Eset*. "Cyber espionage group Turla and its latest malware under the microscope." Accessed on Oct. 28, 2022, at <https://www.eset.com/sg/about/newsroom/press-releases1/awards/cyber-espionage-group-turla-and-its-latest-malware-under-the-microscope-1/>.
- 46 Andreas Klopsch and Andrew Brandt. (Dec. 13, 2022). *Sophos*. "Signed driver malware moves up the software trust chain." Accessed on Dec. 28, 2022, at <https://news.sophos.com/en-us/2022/12/13/signed-driver-malware-moves-up-the-software-trust-chain/>.
- 47 Cedric Pernet. (March 7, 2022). *TechRepublic*. "Nvidia's breach might help cybercriminals run malware campaigns." Accessed on Oct. 13, 2022, at <https://www.techrepublic.com/article/nvidias-breach-might-help-cybercriminals-run-malware-campaigns/>.
- 48 DEFCONConference. (Aug. 6, 2020). *YouTube*. "DEF CON Safe Mode - Bill Demirkapi - Demystifying Modern Windows Rootkits." Accessed on Oct. 13, 2022, at [https://www.youtube.com/watch?v=1H9tEfkjFXs&t=320s&ab\\_channel=DEFCONConference](https://www.youtube.com/watch?v=1H9tEfkjFXs&t=320s&ab_channel=DEFCONConference).
- 49 Eugene Rodionov and Aleksandr Matrosov. (May 11, 2012). *WeLiveSecurity by ESET*. "King of Spam: Festi botnet analysis." Accessed on Oct. 13, 2022, at <https://www.welivesecurity.com/2012/05/11/king-of-spam-festi-botnet-analysis/>.
- 50 Michele Kambas and James Pearson. (Feb. 25, 2022). *Reuters*. "Cyprus games writer denies links to malware found before Russian invasion." Accessed on Oct. 13, 2022, at <https://www.reuters.com/world/europe/cyprus-games-writer-denies-links-malware-found-before-russian-invasion-2022-02-24/>.
- 51 ESET. (March 1, 2022). *ESET*. "ESET Research: Ukraine hit by destructive attacks before and during the Russian invasion with HermeticWiper and IsaacWiper." Accessed on Oct. 13, 2022, at <https://www.eset.com/int/about/newsroom/press-releases/research/eset-research-ukraine-hit-by-destructive-attacks-before-and-during-the-russian-invasion-with-hermet/>.
- 52 Threat Intelligence Team. (March 4, 2022). *Malwarebytes Labs*. "HermeticWiper: A detailed analysis of the destructive malware that targeted Ukraine." Accessed on Oct. 13, 2022, at <https://www.malwarebytes.com/blog/threat-intelligence/2022/03/hermeticwiper-a-detailed-analysis-of-the-destructive-malware-that-targeted-ukraine>.
- 53 Symantec Threat Hunter Team. (Feb. 28, 2022). *Broadcom Software Blogs*. "Designed for Attacks Against Hardened Networks." Accessed on Oct. 13, 2022, at <https://symantec-enterprise-blogs.security.com/blogs/threat-intelligence/daxin-backdoor-espionage>.
- 54 Sherif Magdy et al. (March 25, 2022). *Trend Micro Research, News, and Perspectives*. "Purple Fox Uses New Arrival Vector and Improves Malware Arsenal." Accessed on Oct. 13, 2022, at [https://www.trendmicro.com/en\\_us/research/22/c/purple-fox-uses-new-arrival-vector-and-improves-malware-arsenal.html](https://www.trendmicro.com/en_us/research/22/c/purple-fox-uses-new-arrival-vector-and-improves-malware-arsenal.html).

- 55 Natalie Zargarov. (March 1, 2022). *Minerva Labs*. "Malicious Telegram Installer Drops Purple Fox Rootkit." Accessed on Oct. 13, 2022, at <https://minerva-labs.com/blog/malicious-telegram-installer-drops-purple-fox-rootkit/>.
- 56 Cristian Alexandru Istrate. (Oct. 20, 2021). *BitDefender*. "Digitally-Signed Rootkits are Back – A Look at FiveSys and Companions." Accessed on Oct. 13, 2022, at <https://www.bitdefender.com/blog/labs/digitally-signed-rootkitsare-back-a-look-atfivesys-and-companions/>.
- 57 Cristian Alexandru Istrate. (n.d.). *BitDefender*. "Digitally-Signed Rootkits are Back – A Look at FiveSys and Companions." Accessed on Oct. 13, 2022, at <https://www.bitdefender.com/files/News/CaseStudies/study/405/Bitdefender-DT-Whitepaper-Fivesys-creat5699-en-EN.pdf>.
- 58 Martin Chlumecký. (Aug. 11, 2021). *Decoded*. "DirtyMoe: Rootkit Driver." Accessed on Oct. 13, 2022, at <https://decoded.avast.io/martinchlumecky/dirtymoe-rootkit-driver/>.
- 59 Martin Chlumecký. (Sep. 17, 2021). *Decoded*. "DirtyMoe: Code Signing Certificate." Accessed on Oct. 13, 2022, at <https://decoded.avast.io/martinchlumecky/dirtymoe-3/>.
- 60 Microsoft Security Response Center Team. (June 25, 2021). *Microsoft Security Response Center*. "Investigating and Mitigating Malicious Drivers." Accessed on Oct. 13, 2022, at <https://msrc-blog.microsoft.com/2021/06/25/investigating-and-mitigating-malicious-drivers/>.
- 61 Karsten Hahn. (Dec. 1, 2020). *G Data Blog*. "IceRat evades antivirus by running PHP on Java VM." Accessed on Oct. 11, 2022, at <https://www.gdatasoftware.com/blog/icerat-evades-antivirus-by-using-jphp>.
- 62 Renato Marinho. (Dec. 24, 2021). *SANS Internet Storm Center*. "Example of how attackers are trying to push crypto miners via Log4Shell." Accessed on Oct. 11, 2022, at <https://isc.sans.edu/forums/diary/Example-of-how-attackers-are-trying-to-push-crypto-miners-via-Log4Shell/28172/>.
- 63 Eclipsium. (Dec. 3, 2020). *Eclipsium*. "TRICKBOT NOW OFFERS 'TRICKBOOT': PERSIST, BRICK, PROFIT." Accessed on Oct. 13, 2022, at <https://eclipsium.com/2020/12/03/trickbot-now-offers-trickboot-persist-brick-profit/>.
- 64 Global Research & Analysis Team, Kaspersky Lab. (Sep. 10, 2018). *SecureList by Kaspersky*. "LuckyMouse signs malicious NDISProxy driver with certificate of Chinese IT company." Accessed on Oct. 13, 2022, at <https://securelist.com/luckymouse-ndisproxy-driver/87914/>.
- 65 Lab52. (June 9, 2020). *Lab52*. "Recent FK\_Undead rootkit samples found in the wild." Accessed on Oct. 12, 2022, at <https://lab52.io/blog/recent-fk-undead-rootkit-samples-found-in-the-wild/>.
- 66 Avertium. (June 22, 2020). *Avertium*. "NEW FK\_UNDEAD MALWARE MODULES." Accessed on Oct. 12, 2022, at [https://www.avertium.com/blog/fk\\_undead-malware-modules](https://www.avertium.com/blog/fk_undead-malware-modules).
- 67 QuoIntelligence. (April 20, 2020). *QuoIntelligence*. "WINNTI GROUP: Insights From the Past." Accessed on Oct. 12, 2022, at <https://quointelligence.eu/2020/04/winnti-group-insights-from-the-past/>.
- 68 Ori Damari. (Nov. 1, 2019). *Low Level Pleasure*. "Autochk Rootkit Analysis." Accessed on Oct. 12, 2022, at <https://reprnz.github.io/posts/autochk-rootkit-analysis/>.
- 69 FireEye. (n.d.). *FireEye*. "Double Dragon: APT41, a dual espionage and cyber crime operation." Accessed on Oct. 12, 2022, at <https://content.fireeye.com/apt-41/rpt-apt41>.
- 70 Alex Pennino and Matt Bromiley. (Aug. 19, 2019). *Mandiant*. "GAME OVER: Detecting and Stopping an APT41 Operation." Accessed on Oct. 12, 2022, at <https://www.mandiant.com/resources/blog/game-over-detecting-and-stopping-an-apt41-operation>.
- 71 Bogdan Botezatu. (April 16, 2019). *Bitdefender*. "Inside Scranos – A Cross Platform, Rootkit-Enabled Spyware Operation." Accessed on Oct. 12, 2022, at <https://www.bitdefender.com/blog/labs/inside-scranos-a-cross-platform-rootkit-enabled-spyware-operation/>.
- 72 Bogdan Botezatu. (June 18, 2018). *Bitdefender*. "Six Years and Counting: Inside the Complex Zacinlo Ad Fraud Operation." Accessed on Oct. 12, 2022, at <https://www.bitdefender.com/blog/labs/six-years-and-counting-inside-the-complex-zacinlo-ad-fraud-operation/>.
- 73 Global Research & Analysis Team, Kaspersky Lab. (July 25, 2022). *SecureList by Kaspersky*. "CosmicStrand: the discovery of a sophisticated UEFI firmware rootkit." Accessed on Oct. 13, 2022, <https://securelist.com/cosmicstrand-uefi-firmware-rootkit/106973/>.
- 74 Mark Lechtik et al. (Jan. 20, 2022). *SecureList by Kaspersky*. "MoonBounce: the dark side of UEFI firmware." Accessed on Oct. 13, 2022, at <https://securelist.com/moonbounce-the-dark-side-of-uefi-firmware/105468/>.

- 75 Kaspersky. (n.d.). *Kaspersky*. "Technical details of MoonBounce's implementation." Accessed on Oct. 13, 2022, at [https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2022/01/19115831/MoonBounce\\_technical-details\\_eng.pdf](https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/2022/01/19115831/MoonBounce_technical-details_eng.pdf).
- 76 Assaf Carlsbad & Itai Liba. (March 10, 2022). *Sentinel Labs*. "Another Brick in the Wall: Uncovering SMM Vulnerabilities in HP Firmware." Accessed on Oct. 13, 2022, at <https://www.sentinelone.com/labs/another-brick-in-the-wall-uncovering-smm-vulnerabilities-in-hp-firmware/>.
- 77 Martin Smolár. (April 19, 2022). *WeLiveSecurity by ESET*. "When "secure" isn't secure at all: High-impact UEFI vulnerabilities discovered in Lenovo consumer laptops." Accessed on Oct. 13, 2022, at <https://www.welivesecurity.com/2022/04/19/when-secure-isnt-secure-uefi-vulnerabilities-lenovo-consumer-laptops/>.
- 78 Ravie Lakshmanan. (July 13, 2022). *The Hacker News*. "New UEFI Firmware Vulnerabilities Impact Several Lenovo Notebook Models." Accessed on Oct. 13, 2022, at <https://thehackernews.com/2022/07/new-uefi-firmware-vulnerabilities.html>.
- 79 Padvish Threats Database. (Dec. 28, 2021). *Padvish Threats Database*. "Implant.ARM.iLOBleed.a." Accessed on Oct. 13, 2022, at <https://threats.amnparadaz.com/en/2021/12/28/implant-arm-ilobleed-a/>.
- 80 Martin Smolár and Anton Cherepanov. (Oct. 5, 2021). *WeLiveSecurity by ESET*. "UEFI threats moving to the ESP: Introducing ESPecter bootkit." Accessed on Oct. 13, 2022, at <https://www.welivesecurity.com/2021/10/05/uefi-threats-moving-esp-introducing-especter-bootkit/>.
- 81 Global Research & Analysis Team, Kaspersky Lab. (Sep. 28, 2021). *SecureList by Kaspersky*. "FinSpy: unseen findings." Accessed on Oct. 13, 2022, at <https://securelist.com/finspy-unseen-findings/104322/>.
- 82 Mark Lechtik. (Oct. 5, 2020). *SecureList by Kaspersky*. "MosaicRegressor: Lurking in the Shadows of UEFI." Accessed on Oct. 13, 2022, at <https://securelist.com/mosaicregressor/98849/>.
- 83 ESET Research. (Sep. 27, 2018). *WeLiveSecurity by ESET*. "LoJax: First UEFI rootkit found in the wild, courtesy of the Sednit group." Accessed on Oct. 13, 2022, at <https://www.welivesecurity.com/2018/09/27/lojax-first-uefi-rootkit-found-wild-courtesy-sednit-group/>.
- 84 Gabor Szappanos. (n.d.). *Sophos*. "MyKings: The Slow But Steady Growth of a Relentless Botnet." Accessed on Oct. 13, 2022, at <https://www.sophos.com/en-us/medialibrary/PDFs/technical-papers/sophoslabs-uncut-mykings-report.pdf>.
- 85 Nirmal Singh. (Sep. 13, 2018). *Zscaler*. "DarkCloud Bootkit." Accessed on Oct. 13, 2022, at <https://www.zscaler.com/blogs/security-research/darkcloud-bootkit>.
- 86 Microsoft Defender Security Research Team. (June 7, 2017). *Microsoft Security*. "PLATINUM continues to evolve, find ways to maintain invisibility." Accessed on Oct. 13, 2022, at <https://www.microsoft.com/security/blog/2017/06/07/platinum-continues-to-evolve-find-ways-to-maintain-invisibility/>.
- 87 Charlie Osborne. (July 15, 2015). *ZDNet*. "Hacking Team stealthy spyware rootkit stays entrenched through disk removal." Accessed on Oct. 25, 2022, at <https://www.zdnet.com/article/hacking-team-stealthy-spyware-rootkit-stays-entrenched-through-hard-disk-removal/>.
- 88 Microsoft Security Response Center. (n.d.). *Microsoft Security Response Center*. "Microsoft Security Servicing Criteria for Windows." Accessed on Oct. 13, 2022, <https://www.microsoft.com/en-us/msrc/windows-security-servicing-criteria>.
- 89 Shaun Hurley. (Dec. 7, 2021). *CrowdStrike Blog*. "Critical Hit: How DoppelPaymer Hunts and Kills Windows Processes." Accessed on Oct. 13, 2022, <https://www.crowdstrike.com/blog/how-doppelpaymer-hunts-and-kills-windows-processes/>.
- 90 Microsoft Ignite. (Oct. 11, 2022). *Microsoft Ignite*. "Microsoft recommended driver block rules." Accessed on Oct. 13, 2022, <https://learn.microsoft.com/en-us/windows/security/threat-protection/windows-defender-application-control/microsoft-recommended-driver-block-rules>.
- 91 Michal Poslušný. (Jan. 11, 2022). *WeLiveSecurity by ESET*. "Signed kernel drivers – Unguarded gateway to Windows' core." Accessed on Oct. 13, 2022, at <https://www.welivesecurity.com/2022/01/11/signed-kernel-drivers-unguarded-gateway-windows-core/>.
- 92 GrayhatWarfare. (n.d.). *GrayhatWarfare*. Homepage. Accessed on Oct. 13, 2022, at <https://buckets.grayhatwarfare.com/>.
- 93 Cristian Alexandru Istrate et al. (n.d.). *BitDefender*. "Digitally-Signed Rootkits are Back – A Look at FiveSys and Companions." Accessed on Oct. 16, 2022, at <https://www.bitdefender.com/files/News/CaseStudies/study/405/Bitdefender-DT-Whitepaper-Fivesys-creat5699-en-EN.pdf>.



- 94 Sherif Magdy et al. (March 25, 2022). *Trend Micro Research, News, and Perspectives*. "Purple Fox Uses New Arrival Vector and Improves Malware Arsenal." Accessed on Oct. 28, 2022, at [https://www.trendmicro.com/en\\_us/research/22/c/purple-fox-uses-new-arrival-vector-and-improves-malware-arsenal.html](https://www.trendmicro.com/en_us/research/22/c/purple-fox-uses-new-arrival-vector-and-improves-malware-arsenal.html).
- 95 Sherif Magdy et al. (March 25, 2022). *Trend Micro Research, News, and Perspectives*. "Purple Fox Uses New Arrival Vector and Improves Malware Arsenal." Accessed on Oct. 28, 2022, at [https://www.trendmicro.com/en\\_us/research/22/c/purple-fox-uses-new-arrival-vector-and-improves-malware-arsenal.html](https://www.trendmicro.com/en_us/research/22/c/purple-fox-uses-new-arrival-vector-and-improves-malware-arsenal.html).
- 96 Microsoft Security Response Center Team. (June 25, 2021). *Microsoft Security Response Center*. "Investigating and Mitigating Malicious Drivers." Accessed on Oct. 16, 2022, at <https://msrc-blog.microsoft.com/2021/06/25/investigating-and-mitigating-malicious-drivers/>.
- 97 Mandiant Intelligence. (Dec. 13, 2022). *Mandiant*. "I Solemnly Swear My Driver Is Up to No Good: Hunting for Attestation Signed Malware." Accessed on Dec. 28, 2022, at <https://www.mandiant.com/resources/blog/hunting-attestation-signed-malware>.
- 98 ESET. (March 1, 2022). *ESET*. "ESET Research: Ukraine hit by destructive attacks before and during the Russian invasion with HermeticWiper and IsaacWiper." Accessed on Oct. 16, 2022, at <https://www.eset.com/int/about/newsroom/press-releases/research/eset-research-ukraine-hit-by-destructive-attacks-before-and-during-the-russian-invasion-with-hermet/>.
- 99 Michele Kambas and James Pearson. (Feb. 25, 2022). *Reuters*. "Cyprus games writer denies links to malware found before Russian invasion." Accessed on Oct. 16, 2022, at <https://www.reuters.com/world/europe/cyprus-games-writer-denies-links-malware-found-before-russian-invasion-2022-02-24/>.



## TREND MICRO™ RESEARCH

Trend Micro, a global leader in cybersecurity, helps to make the world safe for exchanging digital information.

Trend Micro Research is powered by experts who are passionate about discovering new threats, sharing key insights, and supporting efforts to stop cybercriminals. Our global team helps identify millions of threats daily, leads the industry in vulnerability disclosures, and publishes innovative research on new threat techniques. We continually work to anticipate new threats and deliver thought-provoking research.

[www.trendmicro.com](http://www.trendmicro.com)

