SafeBreach

Windows Downdate: **Downgrade Attacks Using Windows Updates**

Alon Leviev

Security Researcher @ SafeBreach

22-years-old

Self-taught

OS internals, reverse engineering and vulnerability research

Former BJJ world and european champion

Creator of PoolParty process injection techniques

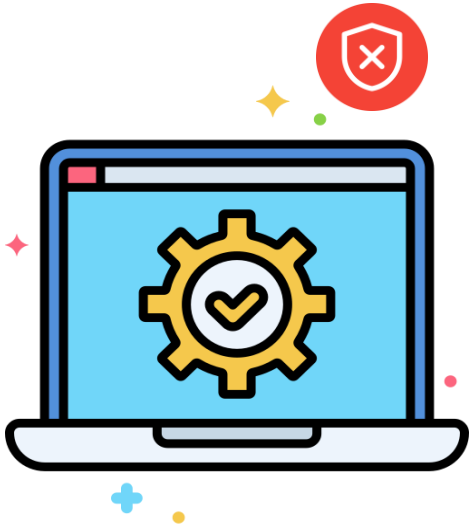
Agenda

- Research Background
- Downgrade Attacks Using Windows Updates
- Virtualization-Based Security Vulnerabilities
- Windows Update Restoration Vulnerability
- Closing Remarks

Research Background

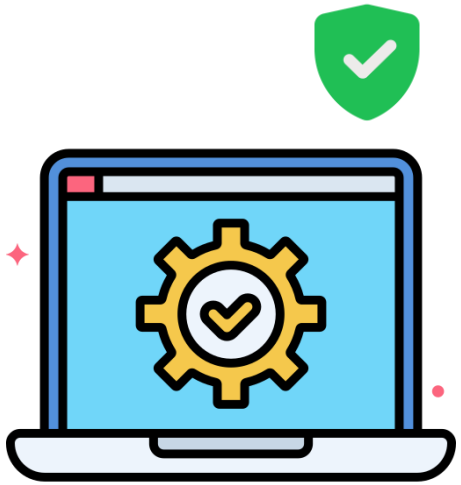
What are Downgrade Attacks?

Attacker



Vulnerable Software

Downgrade **immune** software to **vulnerable** software



Immune Software

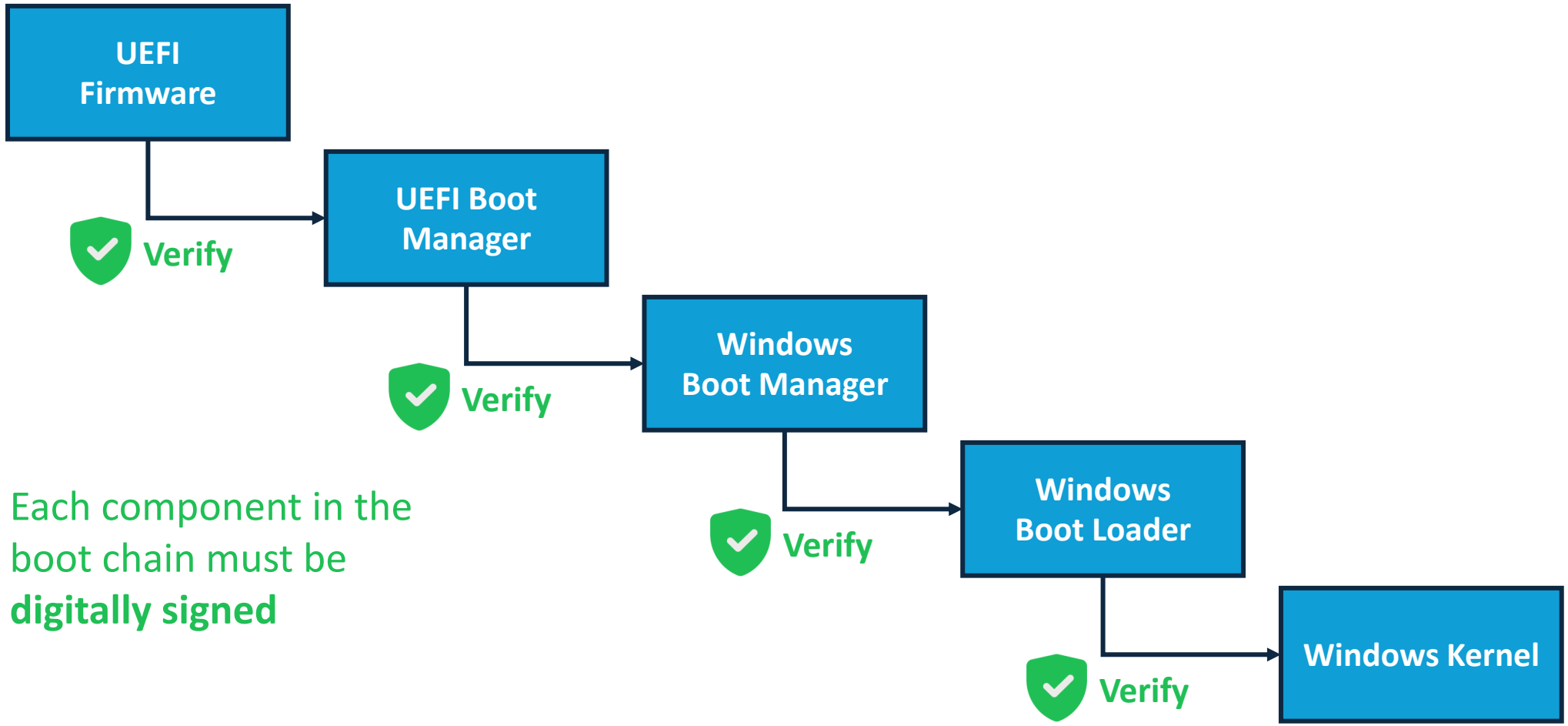
Downgrade Attacks In-The-Wild – BlackLotus UEFI Bootkit

The BlackLotus UEFI bootkit employed a downgrade attack to bypass Secure Boot

The Secure Boot bypass worked on fully updated Windows 11 machines

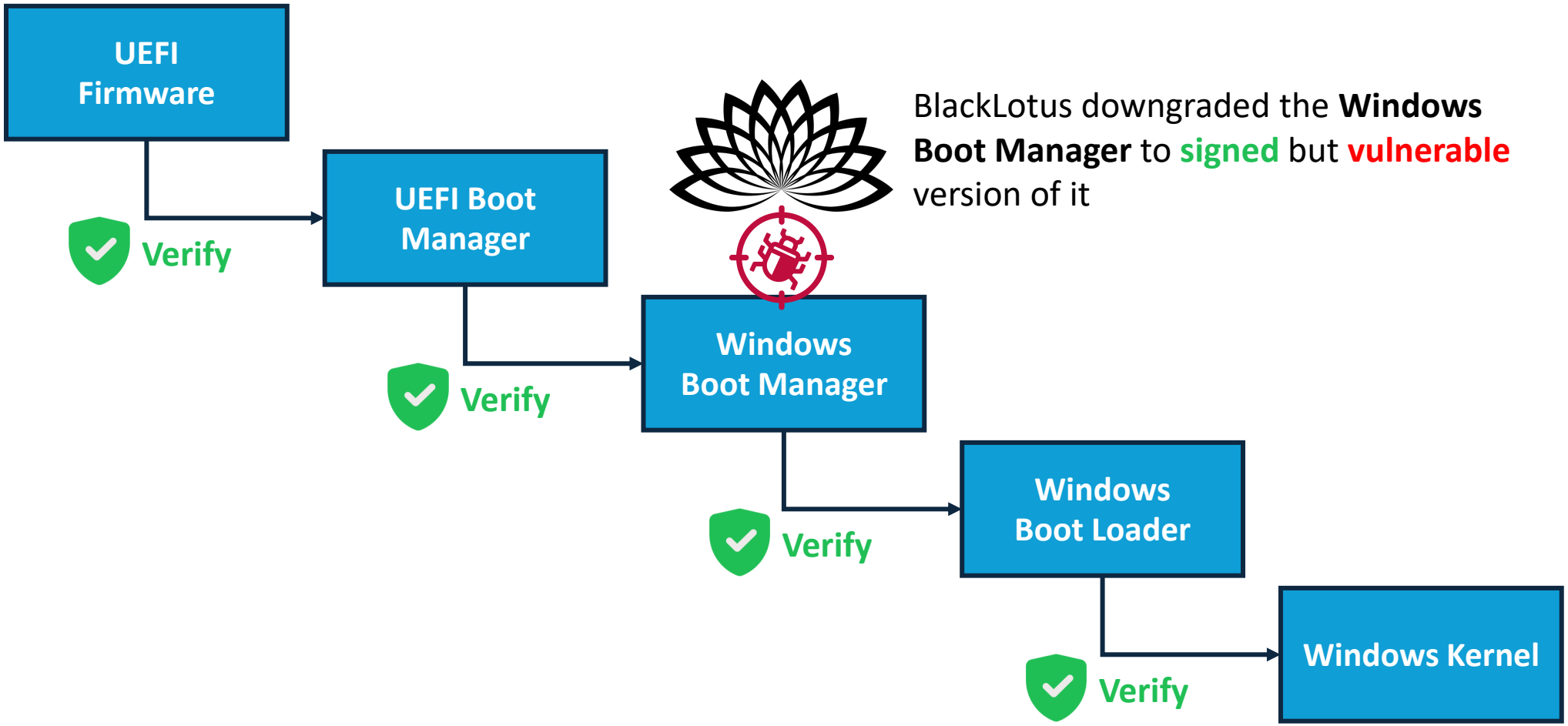
Caused a massive panic in the cyber security industry

Secure Boot In a Nutshell



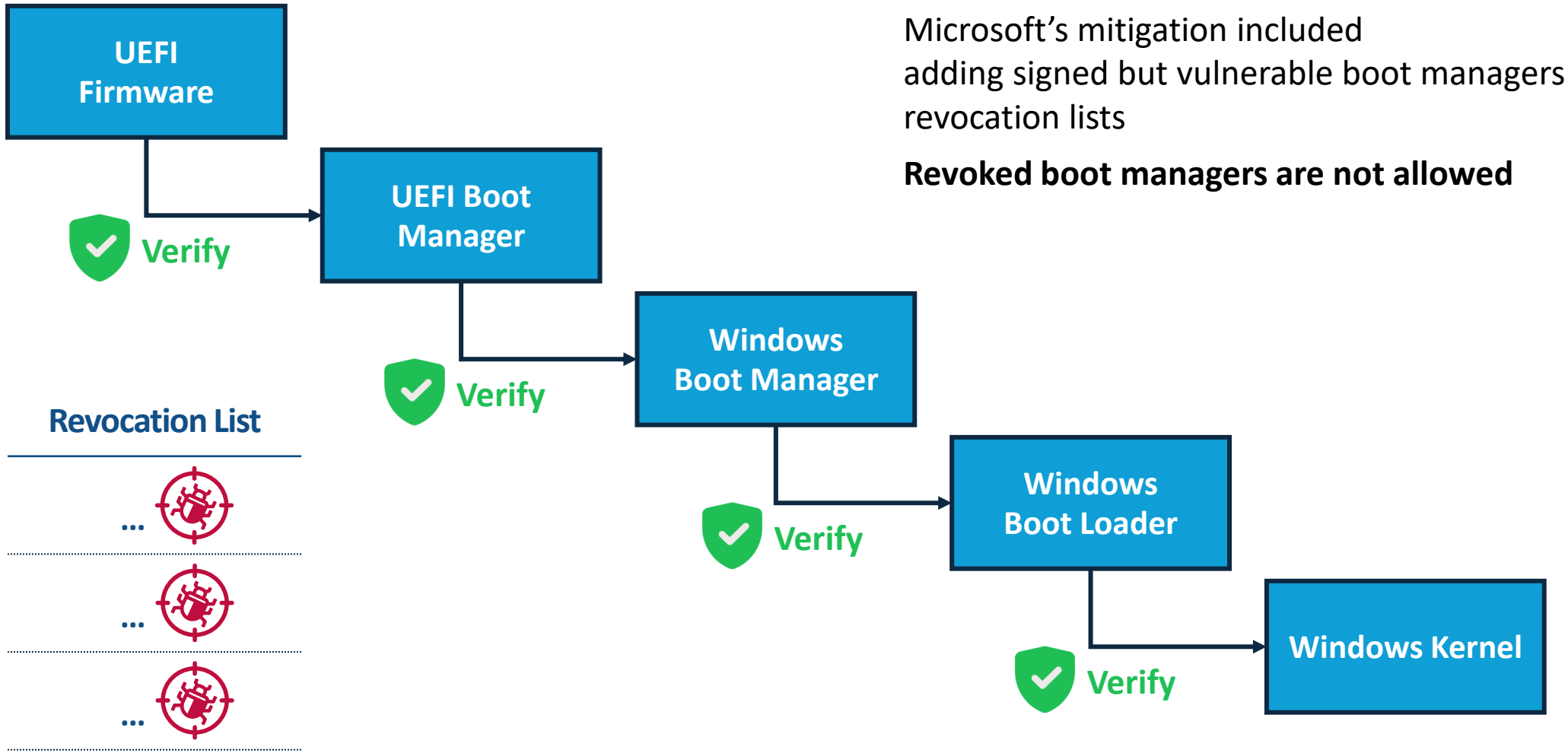
Each component in the boot chain must be **digitally signed**

BlackLotus Secure Boot Bypass



BlackLotus downgraded the **Windows Boot Manager** to **signed** but **vulnerable** version of it

Microsoft's Mitigation Against Secure Boot Downgrades



Microsoft's mitigation included adding signed but vulnerable boot managers to revocation lists

Revoked boot managers are not allowed

Research Motivation

Are there any components affected by downgrade attacks other then Secure Boot?

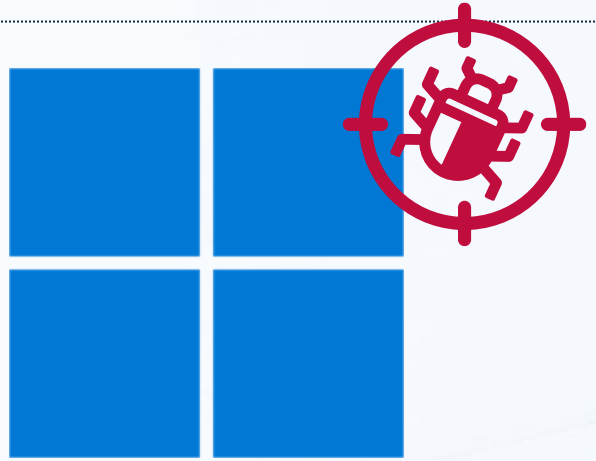
Research Goals

Evaluate the state of downgrade attacks on Windows

Find if any other critical components have been overlooked

Downgrade Vision

Bring Your Own Vulnerable Windows!



What makes a downgrade attack complete?

Fully Undetectable	The downgrade is performed in a legitimate way
Invisible	The downgraded components appear up to date
Persistent	Future updates do not overwrite the downgraded components
Irreversible	Scanning and repairing tools are unable to detect and repair corruptions

Finding the suitable component

Which component is the least expected to perform downgrades?

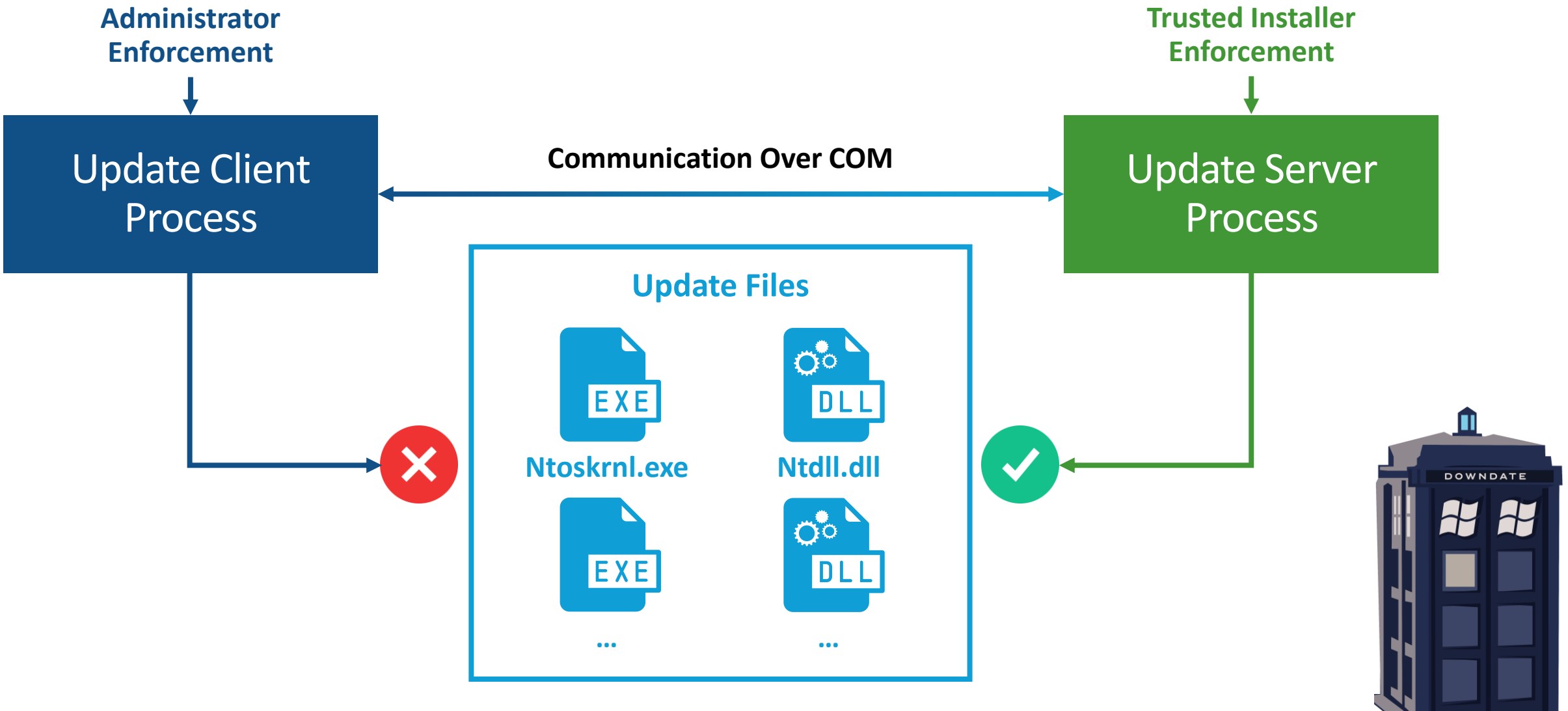
Finding the suitable component

Windows Updates!

WINDOWS DOWNDATE

Downgrade Attacks Using Windows Updates

Windows Updates Architecture



Trusted Installer enforcement – Is It Useful?

Multiple working public PoC's of Administrator to Trusted Installer elevation

It is considered malicious and EDRs detect such elevations

Even if I bypass detection, self-implementing the downgrade may seem malicious

Taking over the Windows Update process solves all of that

Update Flow

Update Client Process

1. Client asks the server to perform an update given an update folder

Update Server Process

Update Flow

Update Client Process

2. Server validates the integrity of the client supplied update folder

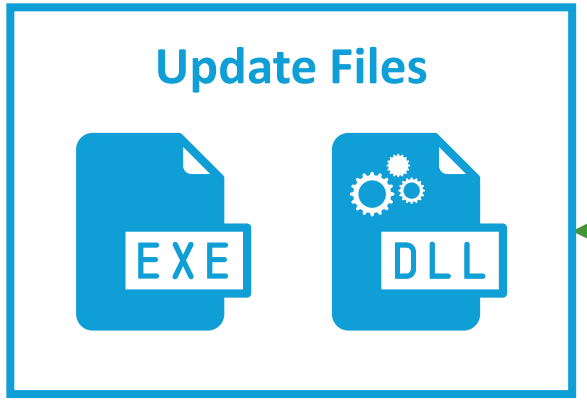
Update Server Process

Update Flow

Update Client Process

3. Server operates on the update folder to finalize the update files

Update Server Process

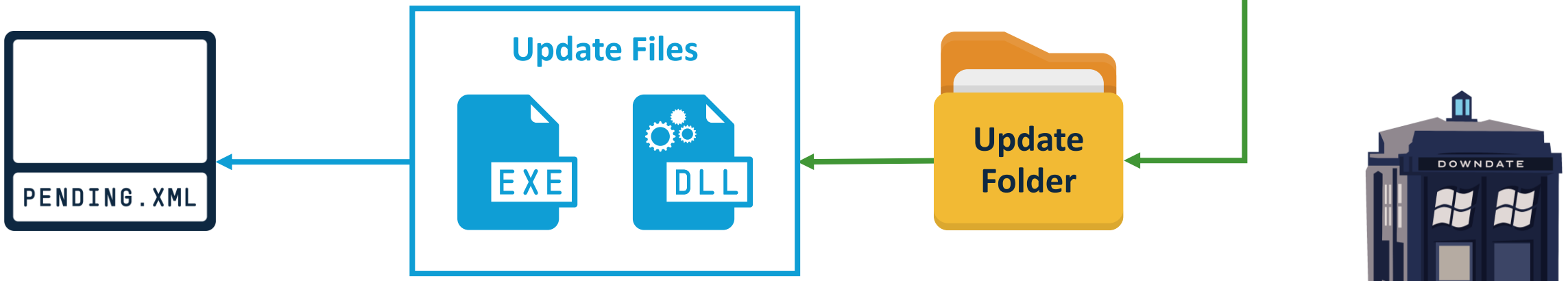


Update Flow

4. Server saves the update action list to **%WinDir%\WinSxS\Pending.xml**

Update Client Process

Update Server Process

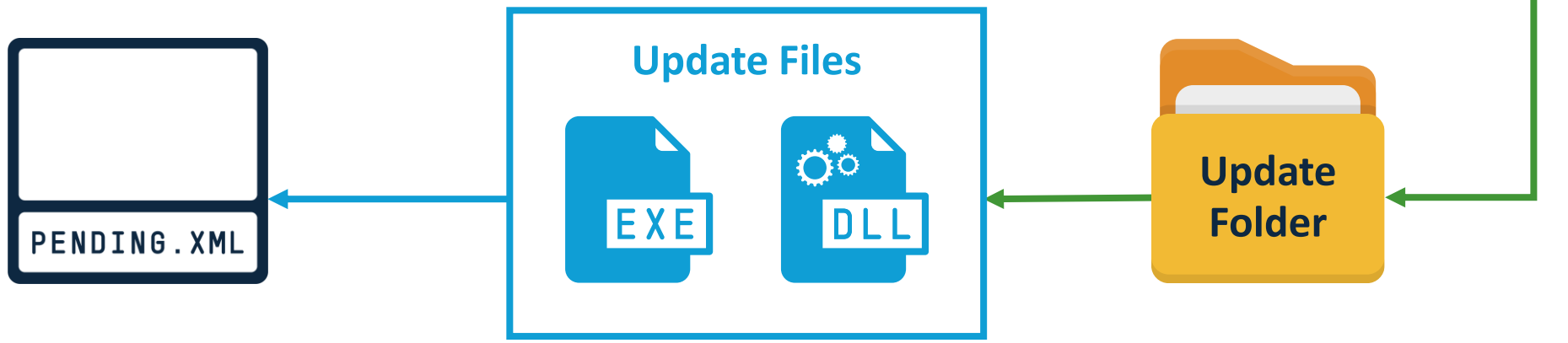


Update Flow

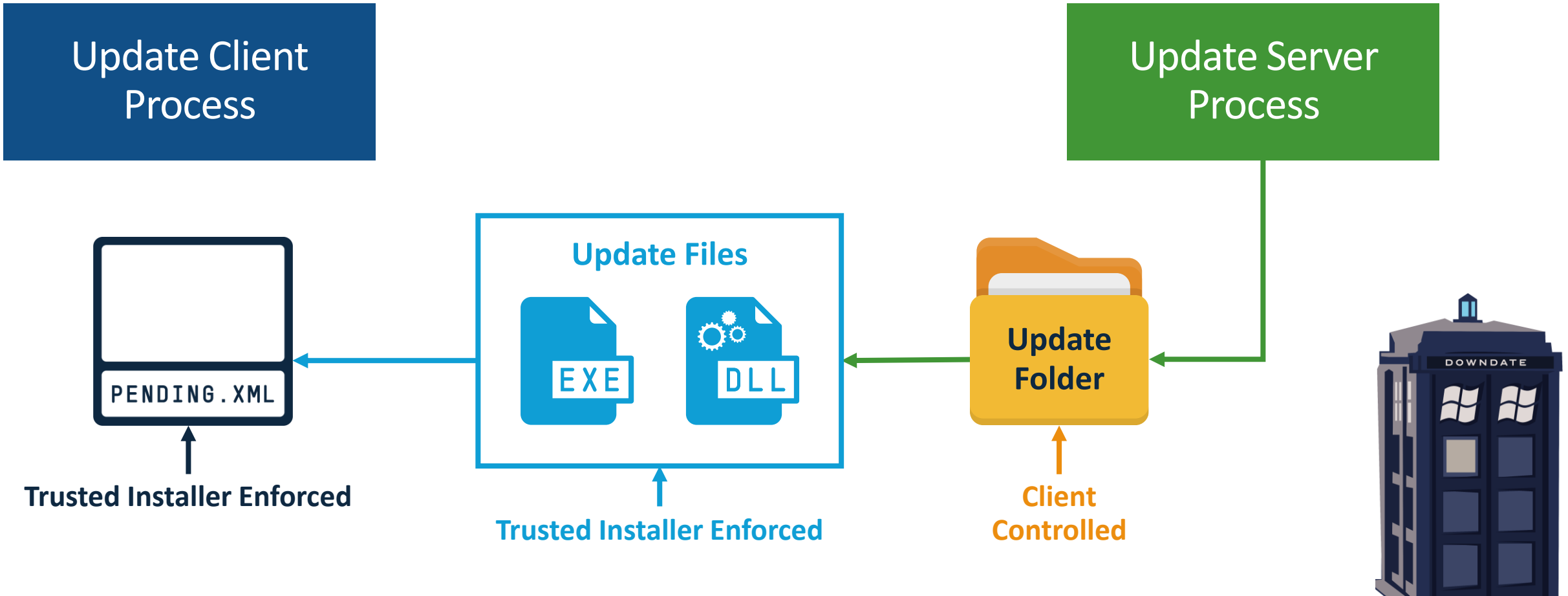
5. In the next reboot, **Pending.xml** is operated on, and the update actions are performed

Update Client Process

Update Server Process

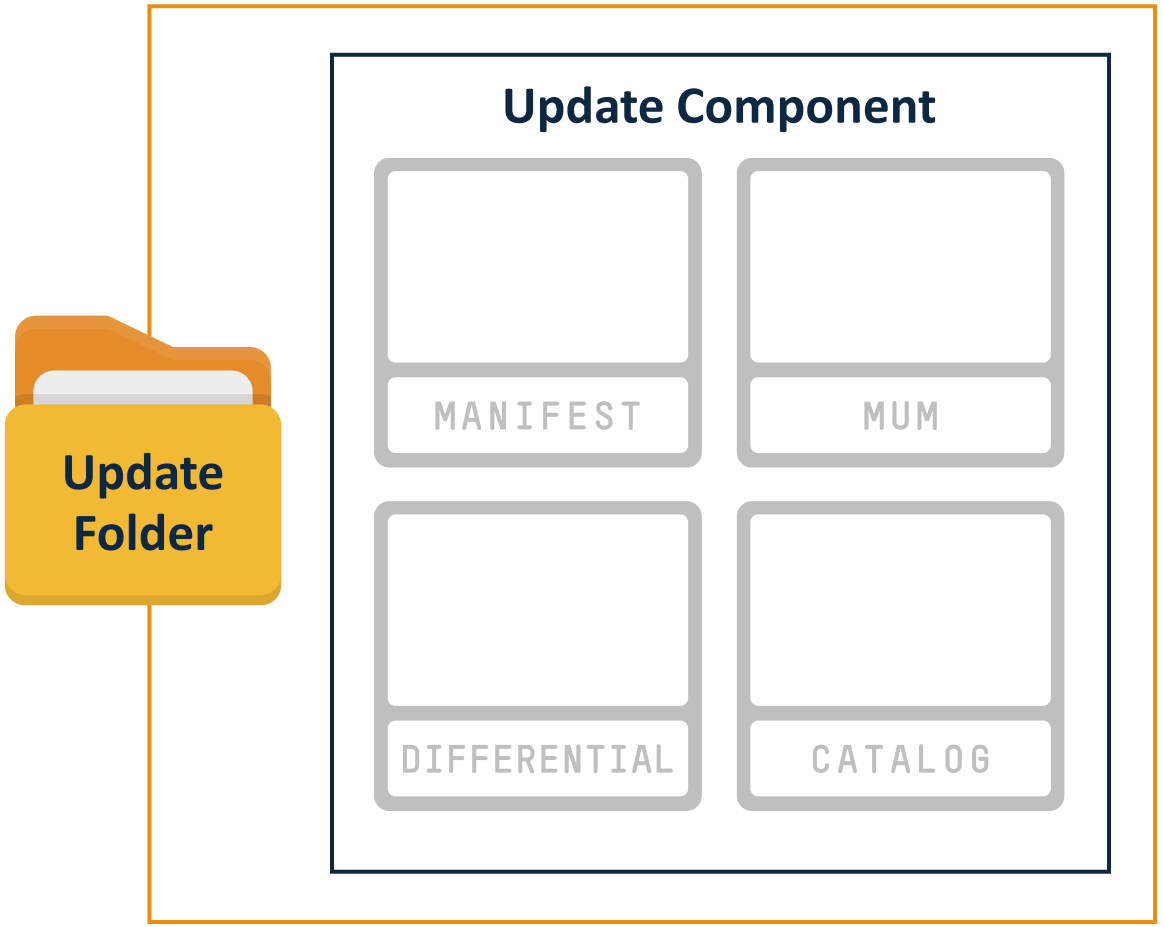


What Is Client Controlled?



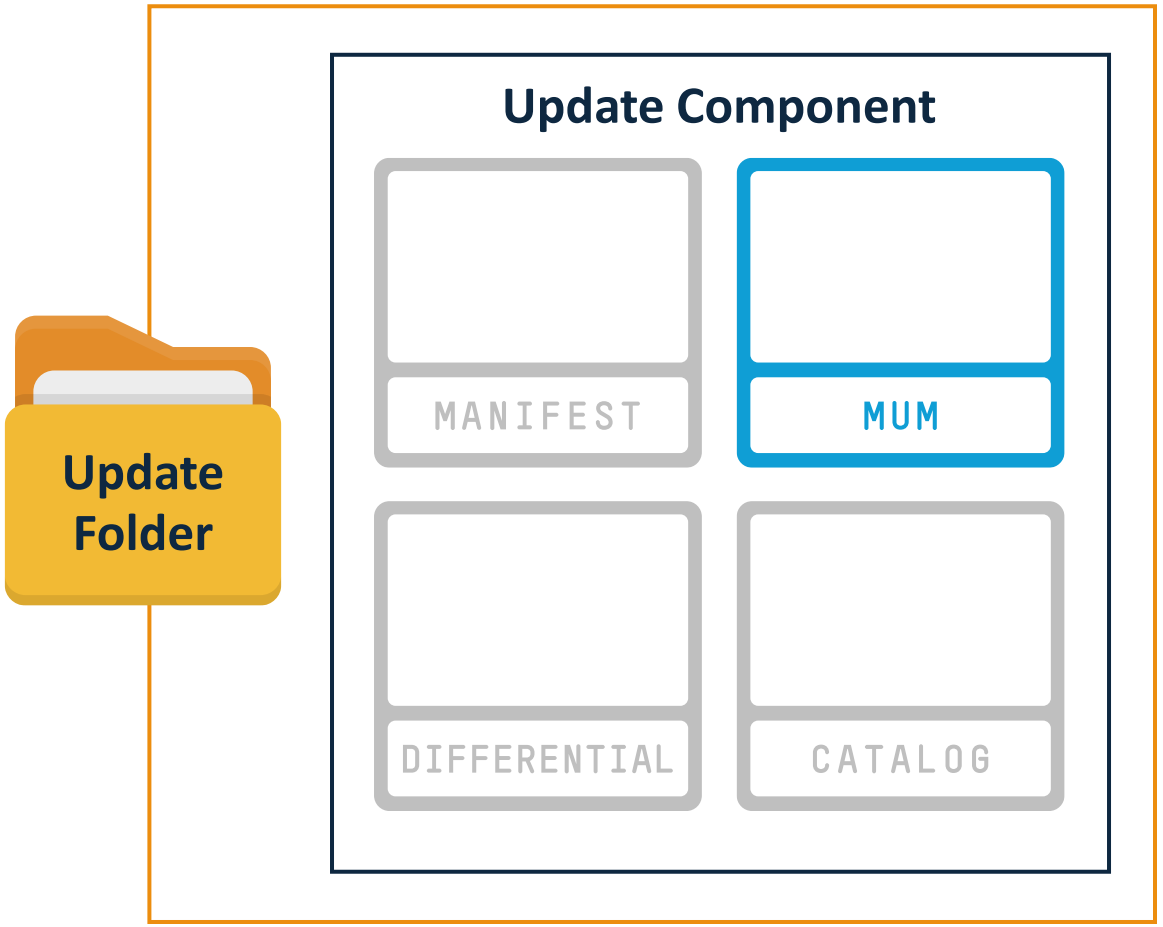
Update Folder Contents

Update folder contains update components



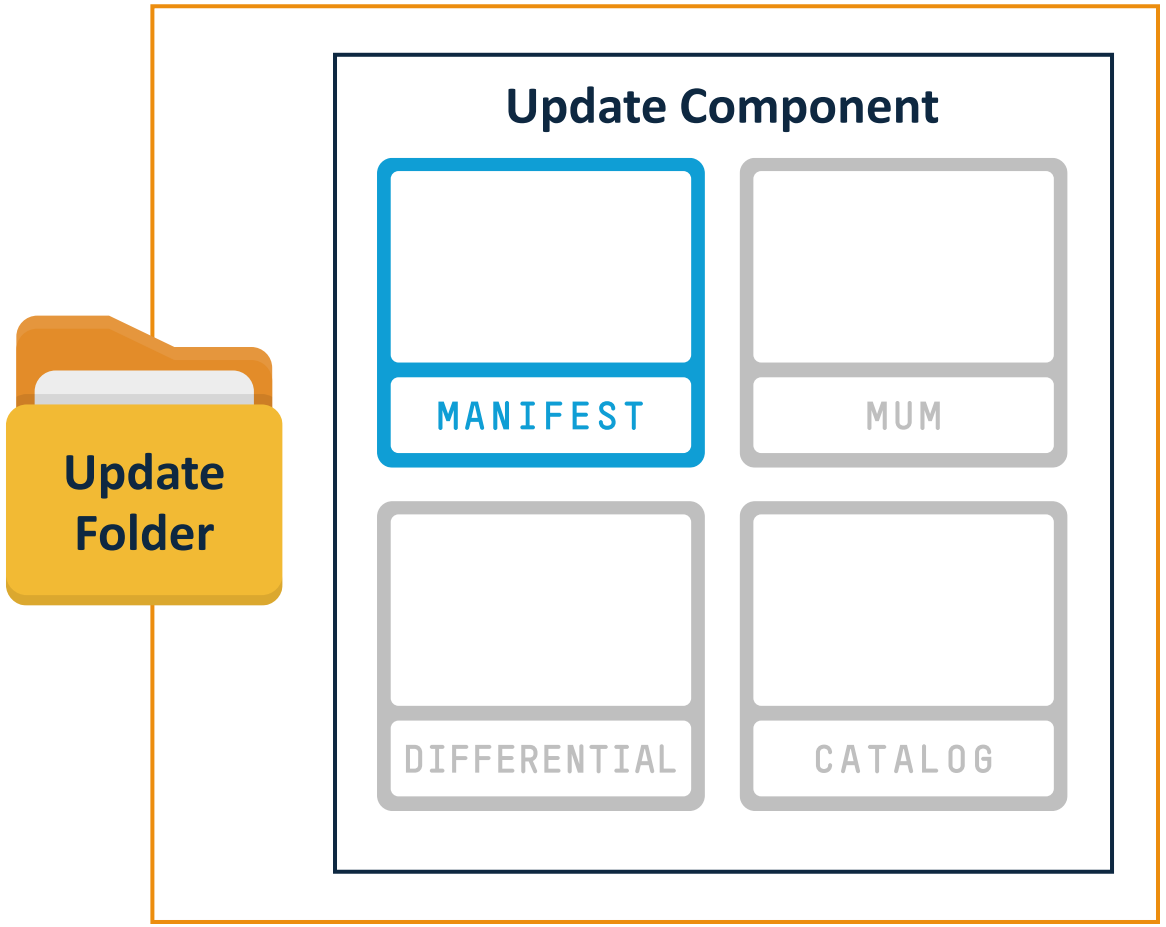
Update Folder Contents – MUM

MUM files contain component metadata, component dependencies, installation order etc.



Update Folder Contents – Manifest

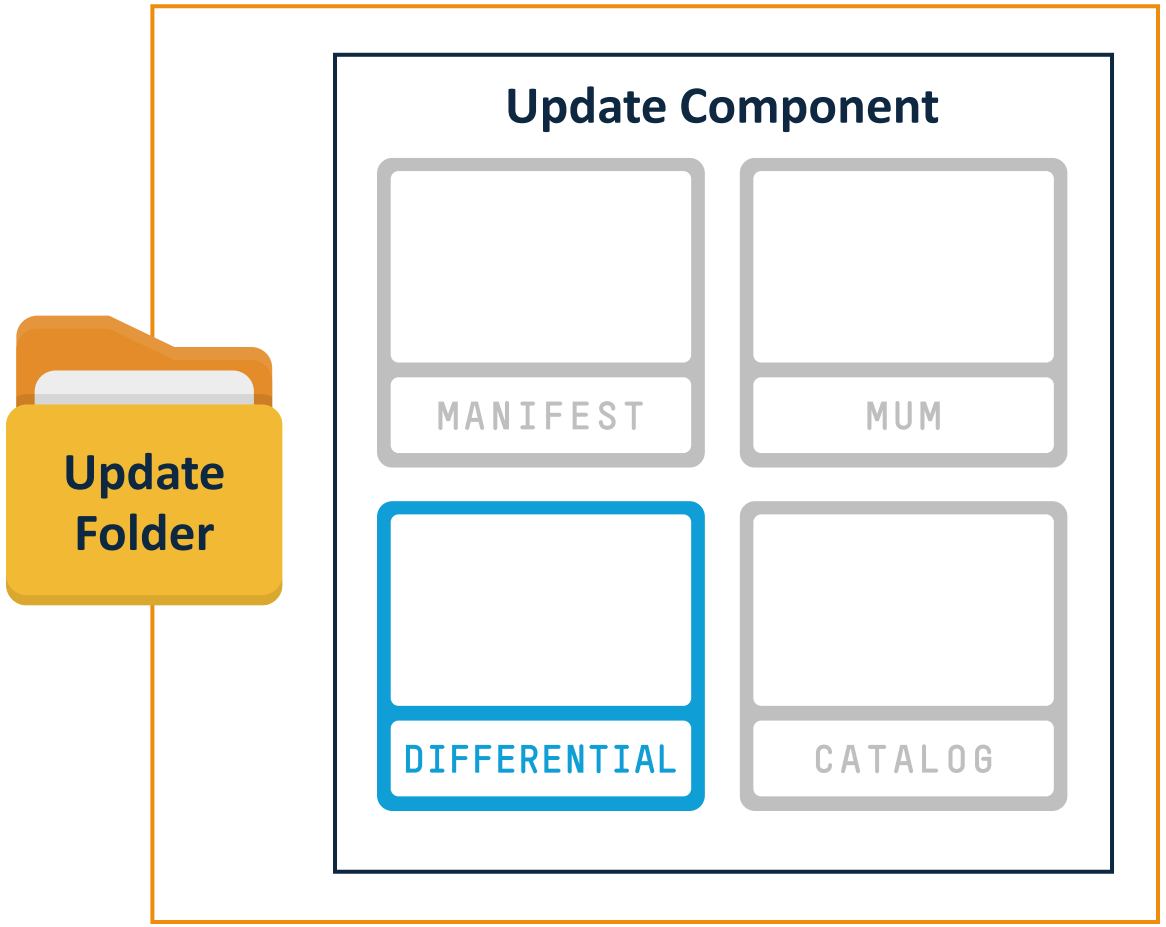
Manifest files contain installation specific data such as file paths, registry keys, and installers to execute



Update Folder Contents – Differential

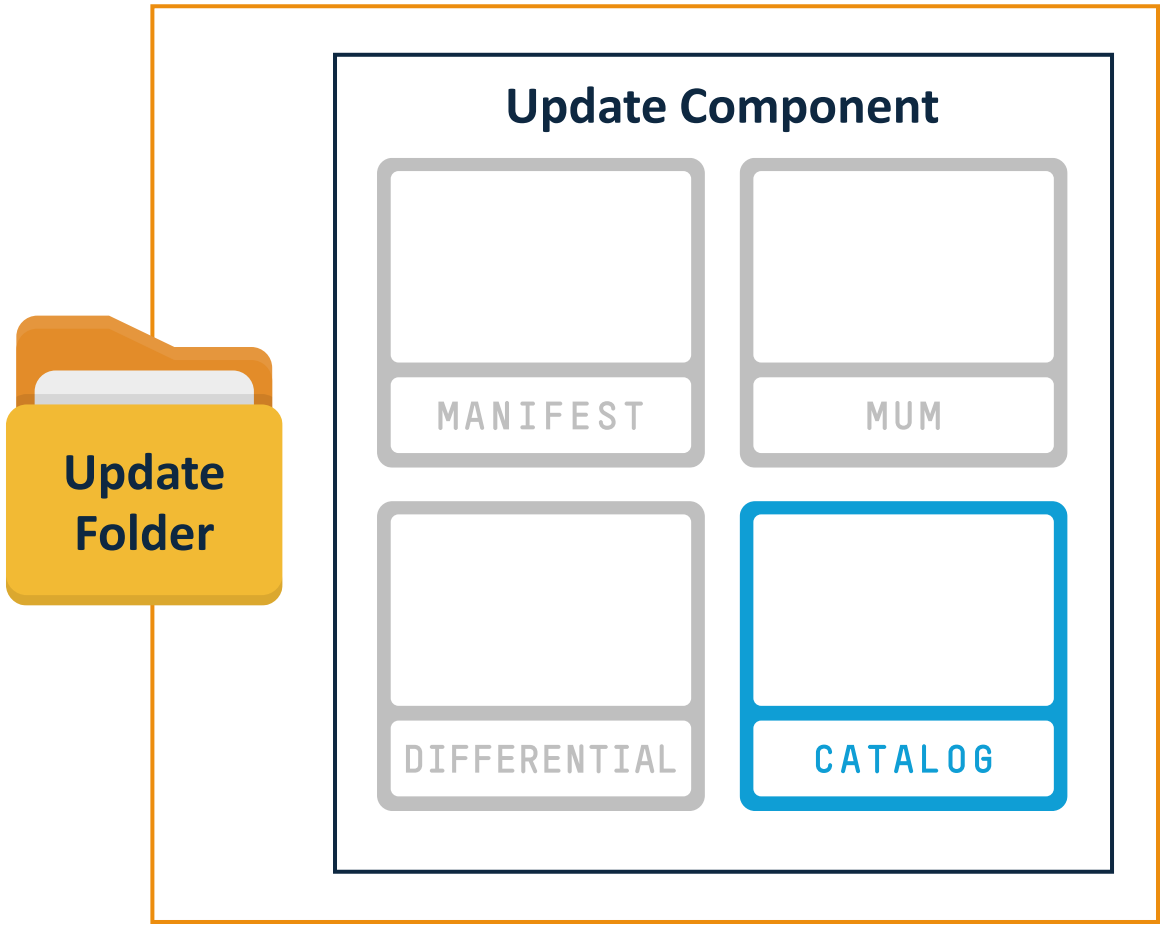
Differential files
are deltas from the base files

Base + Differential = Full Update File



Update Folder Contents – Catalog

Catalog files are the digital signatures of **MUM** and **Manifest** files



Update Folder Contents – Recap

Only **Catalogs** are explicitly digitally signed

Manifests and **MUMs** are not explicitly digitally signed, but are signed in **Catalogs**

Differentials are not digitally signed

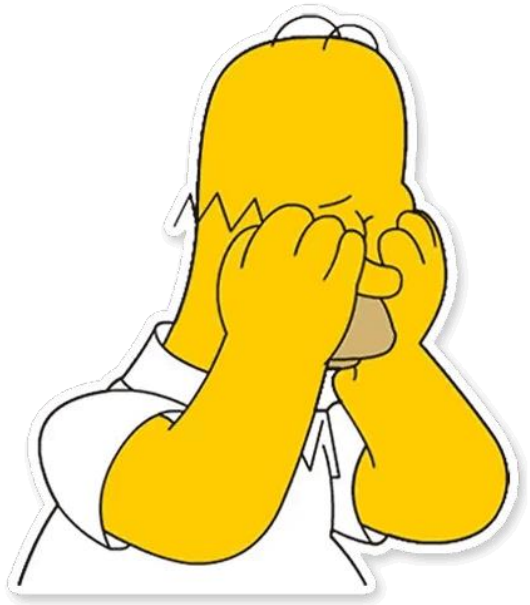
Differentials control the actual final update file content

Targeting Differential Files

Any chance that **differential** files were left behind in terms of verification?

Targeting Differential Files – Impossible

Expected full update files hashes are hardcoded in the **manifests**



Sha256
(Full Update File)

Targeting The Action List

The action list is Trusted Installer enforced. Since operated on during reboots, the system must save its state somewhere.

Targeting The Action List – Possible!

Action List path is saved in the registry and is **not Trusted Installer enforced!**

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\SideBySide\Configuration\PoqexecCmdline

PoqExec.exe Pending.xml [more args]

What Can We Do With The Action List?

```
<POQ postAction="reboot">
    <CreateFile path="C:\Windows\System32\Create.exe" fileAttributes="0x00000000"/>
    <MoveFile source="C:\UpdateDir\Source.exe" destination="C:\Windows\System32\Destination.exe"/>
    <HardlinkFile source="C:\UpdateDir\Source.exe" destination="C:\Windows\System32\Destination.exe"/>
    <SetFileInformation path="C:\UpdateDir\Source.exe" securityDescriptor="binary base64:[BASE64-BLOB]" flags="0x00000040"/>
    <DeleteFile path="C:\Windows\System32\Delete.exe"/>
    <CreateDirectory path="C:\Windows\System32\Directory" fileAttribute="0x00000080" securityDescriptor="binary base64:[BASE64-BLOB]"/>
    <CreateKey path="\Registry\Machine\Key"/>
    <SetKeyValue path="\Registry\Machine\Key" name="Name" type="0x00000001" encoding="base64" value="[BASE64-BLOB]"/>
    <SetKeySecurity path="\Registry\Machine\Key" securityDescriptor="binary base64:[BASE64-BLOB]" flags="0x00000001"/>
    <DeleteKeyValue path="\Registry\Machine\Key" name="Value"/>
    <DeleteKey flags="0x00000001" path="\Regsitry\Machine\Key"/>
</POQ>
```

How To Downgrade Files?

The **HardlinkFile** action can be used to downgrade system files

```
<HardlinkFile source="C:\UpdateDir\Source.exe" destination="C:\Windows\System32\Destination.exe"/>
```

Initiating Update

1. Set Trusted Installer service as Auto-Start

2. Add Pending.xml path in registry

HKLM\.....\PoqexecCmdline

3. Add Pending.xml identifier in registry

HKLM\COMPONENTS\PendingXmlIdentifier

Downgrade Attack via Windows Update Achieved!

Ability to “update” the system with a downgrading Pending.xml

All integrity verification checks are bypassed

No Trusted Installer elevation is required

Complete Windows Update takeover!

Complete Downgrade Attack – Fully Undetectable

The downgrade is fully undetectable, it is performed
in the most legitimate way

Complete Downgrade Attack – Invisible

The system will appear up to date, as we “updated” the system

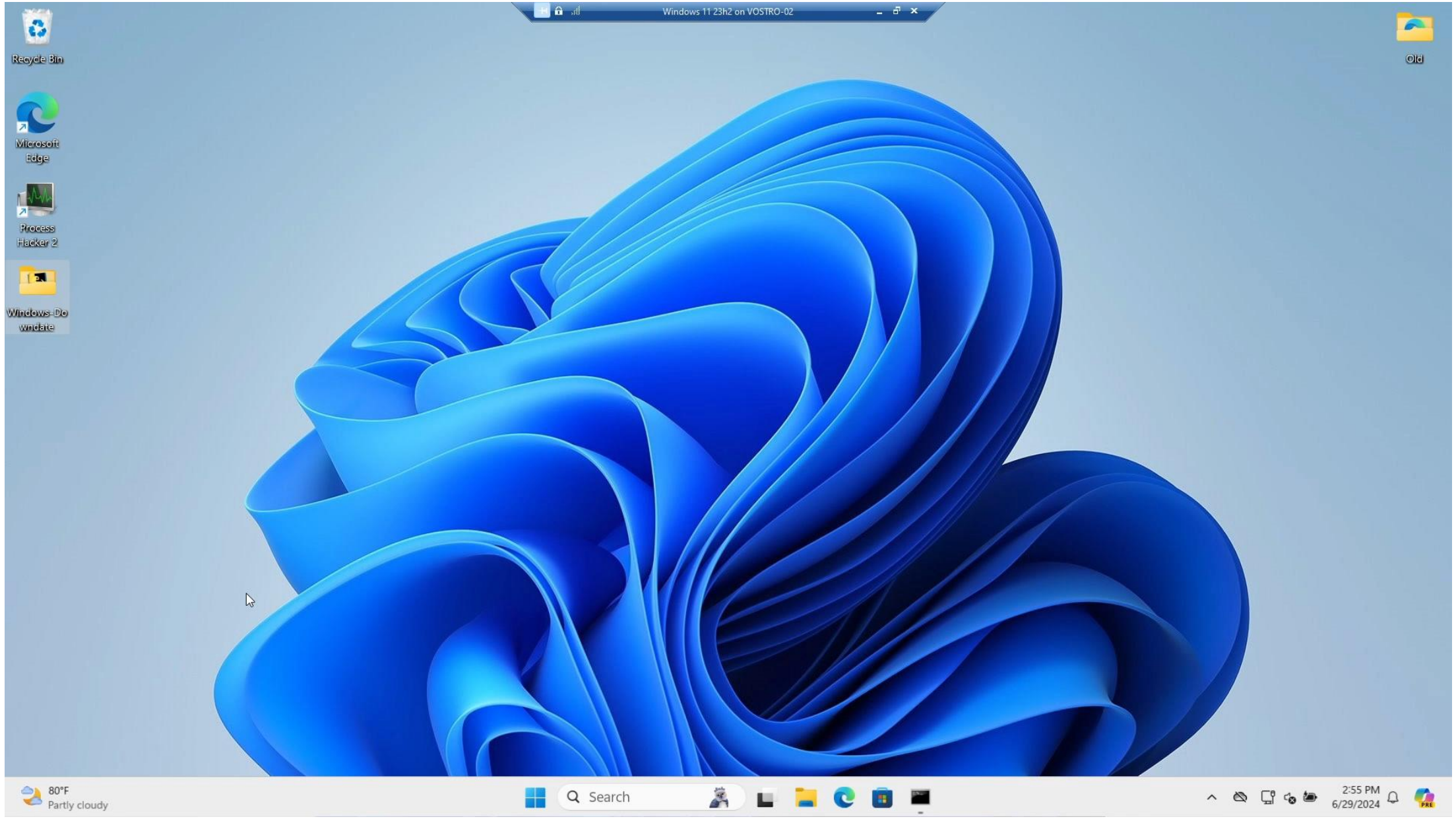
Complete Downgrade Attack – Persistent

The action list parser PoqExec.exe is not digitally signed, and can be patched to install empty updates

Complete Downgrade Attack – Irreversible

The System Integrity Check and Repair utility SFC.exe is not digitally signed, and can be patched to never detect or repair corruptions

Demo #1


Windows 11 23h2 on VOSTRO-02
Recycle Bin
Old
Microsoft Edge
Process Hacker 2
Windows-Update
80°F
Partly cloudy
Search
2:55 PM
6/29/2024

Admin to Kernel – Security Boundary?

Not a security boundary

Admin to Kernel – Not a Boundary, But Still a Threat

Lots of users are still running
as Administrator

Microsoft's Solution – Deprivileging the Kernel

Microsoft decided to deprivilege the kernel to make kernel access less valuable

WINDOWS DOWNDATE

Virtualization-Based Security

What Is VBS?

Secure and isolated virtual environment powered by the Hyper-V hypervisor

Why VBS was created?

Kernel is assumed compromised

Need a secure place for security features and key storage

VBS Security Features

Credential Guard

Hypervisor-Protected Code Integrity (aka. HVCI)

System Guard Secure Launch

Shielded VMs

And more!

Windows Architecture – Before VBS

User Mode (Ring 3)

Process A

Process B

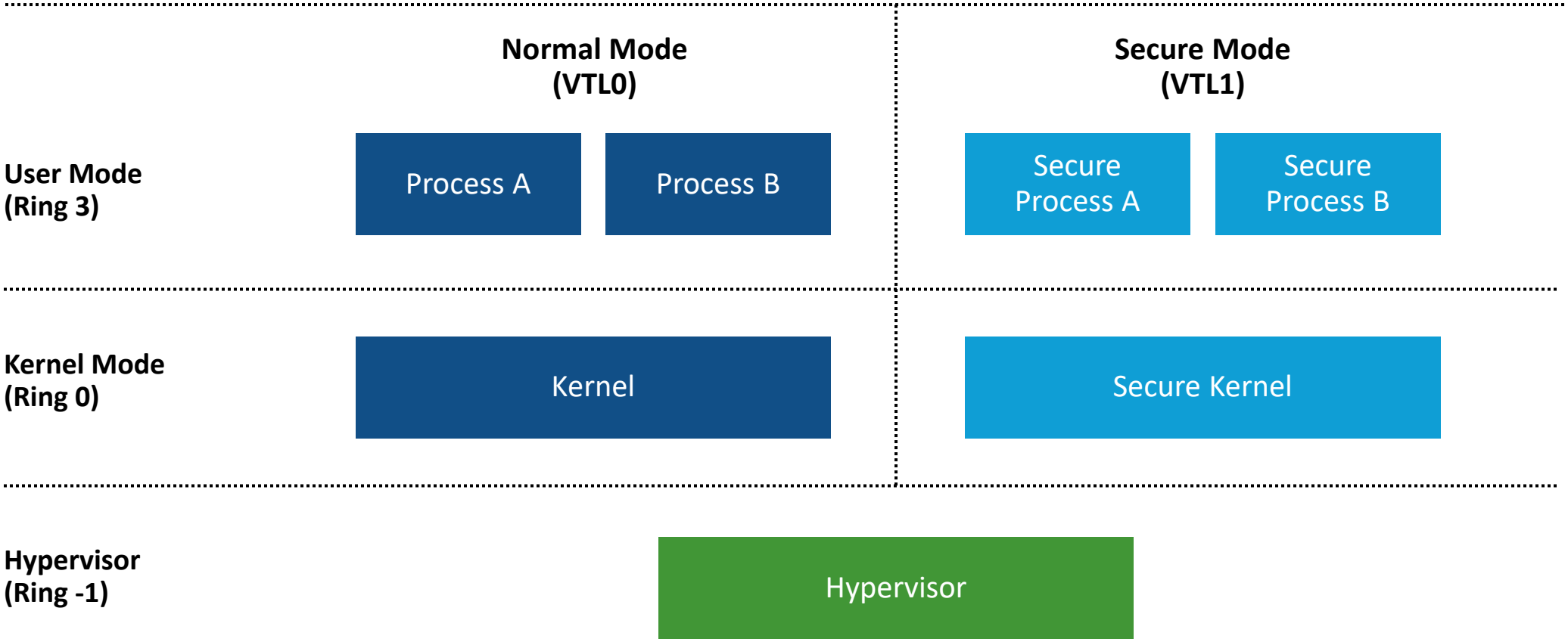
Kernel Mode (Ring 0)

Kernel

Hypervisor (Ring -1)

Hypervisor

Windows Architecture – After VBS



VBS Remote Disablement Protection via UEFI Locks

Boot service UEFI variable is used as configuration source instead of Windows Registry

VBS Remote Disablement Protection via UEFI Locks

Disabling UEFI lock protected feature requires loading a dedicated EFI application that **requires physical approval** to clear the UEFI lock

SecConfig.efi → Ask user to **physically approve** disablement → Clear lock → **NV|BS UEFI Variable** VbsPolicy

VBS Remote Disablement Protection via UEFI Locks

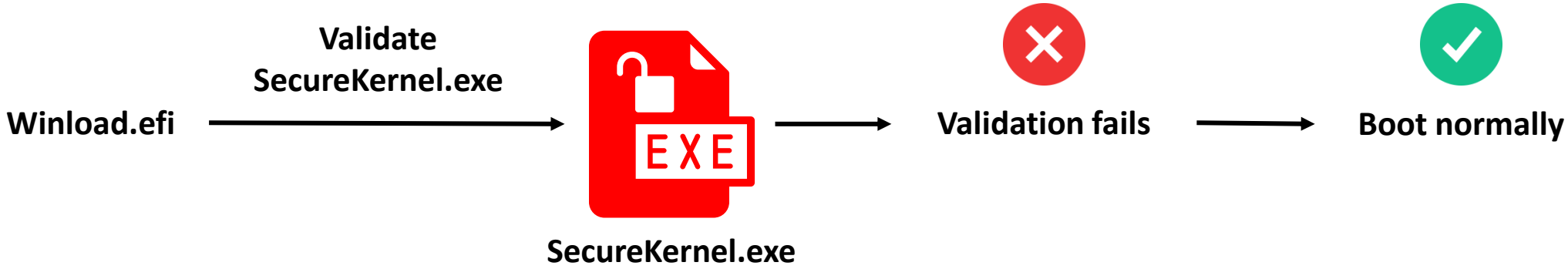
What will happen if we **invalidate** VBS files?
How will VBS react?

SecureKernel.exe

Hvix64.exe

VBS Remote Disablement Protection via UEFI Locks

Windows boots normally, abandoning VBS
Even when enforced with UEFI locks!



Demo #2 – Chaining It All Together

What are we going to see?	Settings	How will it happen?
Credential extraction against the most restrictive settings	PPL enabled for LSASS with UEFI lock	PPL bypass by reverting the PPLFault patch
	Credential Guard enabled with UEFI lock	Credential Guard disablement bypassing UEFI lock
	Windows Defender up and running	Turning Windows Defender unfunctional

Demo #2 –
What If only Credential Guard Is Bypassed?

LSASS can not be dumped

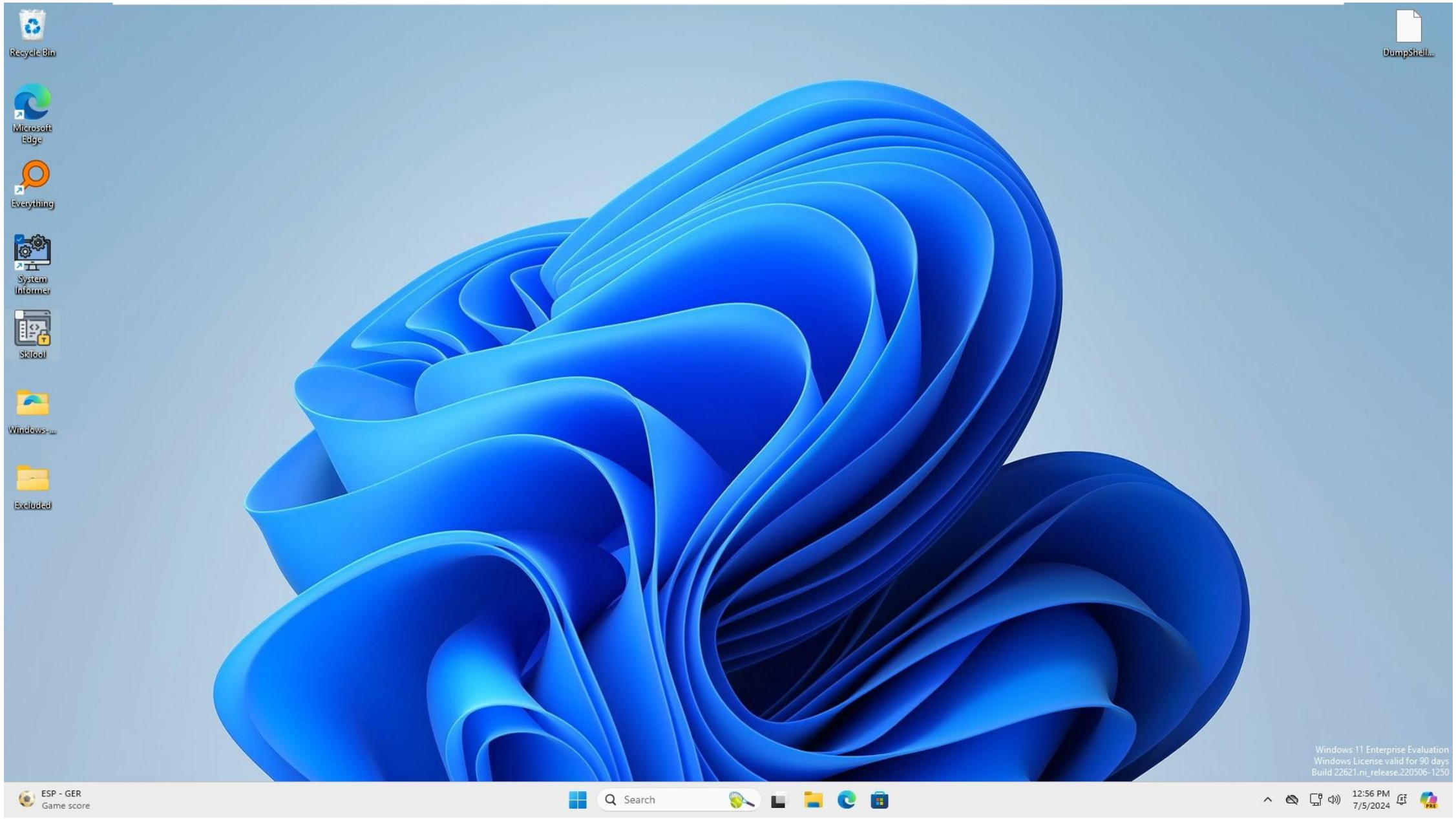
```
C:\Users\Alon\Desktop\Excluded>PPLFault.exe -v 760 lsass.dmp
 [+] No cleanup necessary.  Backup does not exist.
 [+] GetShellcode: 528 bytes of shellcode written over DLL entrypoint
 [+] Benign: C:\Windows\System32\EventAggregation.dll.bak
 [+] Payload: C:\PPLFaultTemp\PPLFaultPayload.dll
 [+] Placeholder: C:\PPLFaultTemp\EventAggregationPH.dll
 [+] Acquired exclusive oplock to file: C:\Windows\System32\devobj.dll
 [+] Ready.  Spawning WinTcb.
 [+] SpawnPPL: Waiting for child process to finish.
 [+] FetchDataCallback called.
 [+] Hydrating 90112 bytes at offset 0
 [+] Switching to payload
 [+] Emptying system working set
 [+] Working set purged
 [+] Give the memory manager a moment to think
 [+] Hydrating 90112 PAYLOAD bytes at offset 0
 [!] Did not find expected dump file: lsass.dmp
```

Demo #2 –
What If only PPL Is Bypassed?

Credentials are encrypted

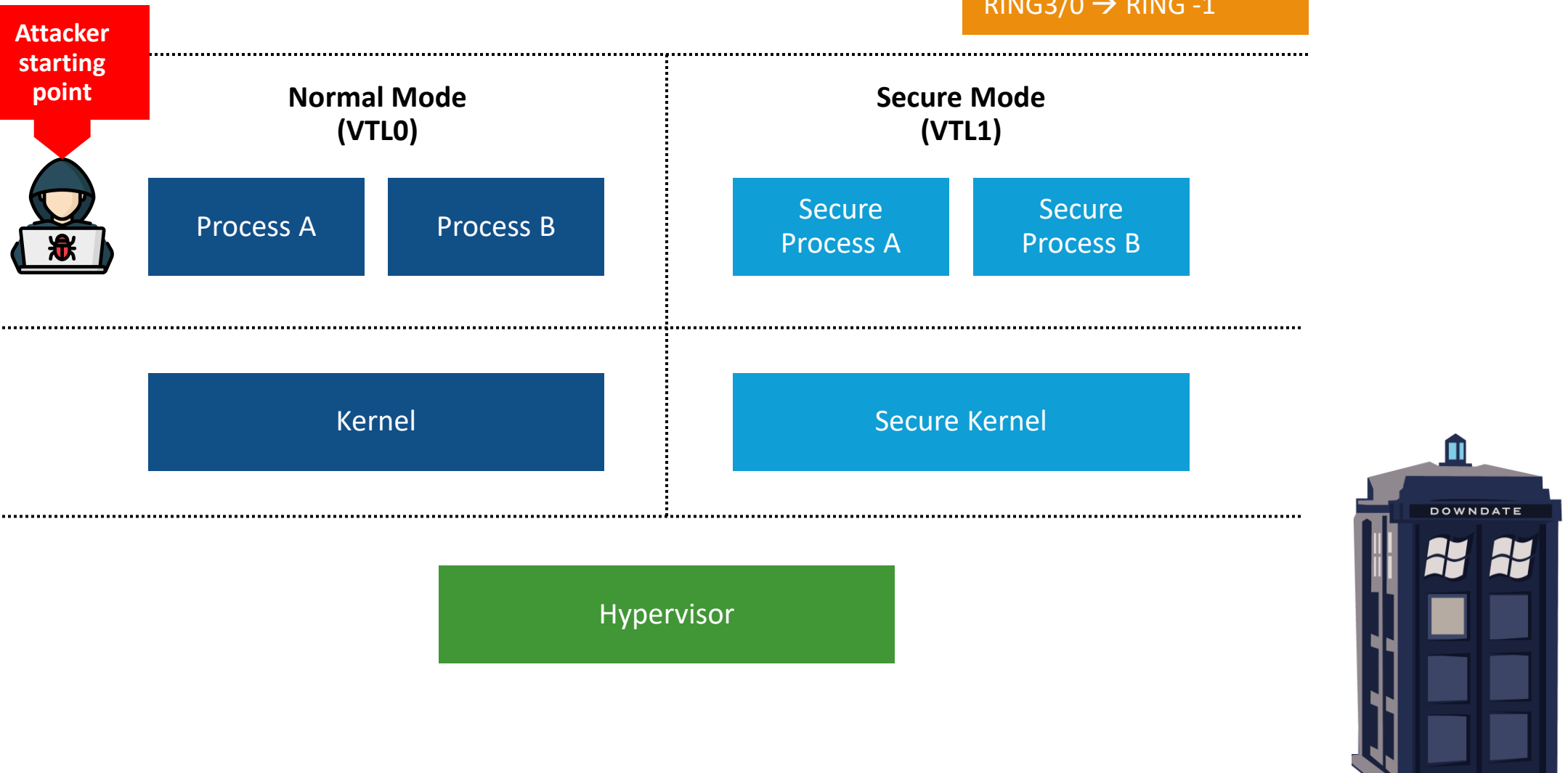
```
Authentication Id : 0 ; 466417 (00000000:00071df1)
Session           : Interactive from 1
User Name         : fcastle
Domain            : DEKU
Logon Server      : DEKU-DC
Logon Time        : 7/5/2024 12:13:03 PM
SID               : S-1-5-21-1272771627-2707523857-1367165163-1103
        msv :
         [00000003] Primary
         * Username : fcastle
         * Domain   : DEKU
           * LSA Isolated Data: NtlmHash
             KdfContext: db0f131340c395ea790ee6896f146184b1c08a6dd3b7f56a874e4383560905a4
             Tag       : 197ed55ab8ca142dd067c9f8ccc38e93
             AuthData  : 010000000000000000000000000000001000000340000004e746c6d48617368
             Encrypted : 9c601dc3c53f49df535bee643ddf5fcf6e051cb703e6ddff8e32ba19f452d2852634daee854ef3196c32ccc661958d2ffa296816
         * DPAPI    : 732646f2969ff7857b666e68ba3dca9b
        tspkg :
        wdigest :       KO
        kerberos :
         * Username : fcastle
         * Domain   : DEKU.LOCAL
         * Password : (null)
        ssp :
        credman :
        cloudap :
```

Demo #2


Recycle Bin
DumpShell...
Microsoft Edge
Everything
System Informer
SkTool
Windows-...
Excluded
Windows 11 Enterprise Evaluation
Windows License valid for 90 days
Build 22621.ni_release.220506-1250
ESP - GER
Game score
Search
12:56 PM
7/5/2024

VBS Security Boundaries

Security Boundaries:
VTL0 → VTL1
RING3/0 → RING -1



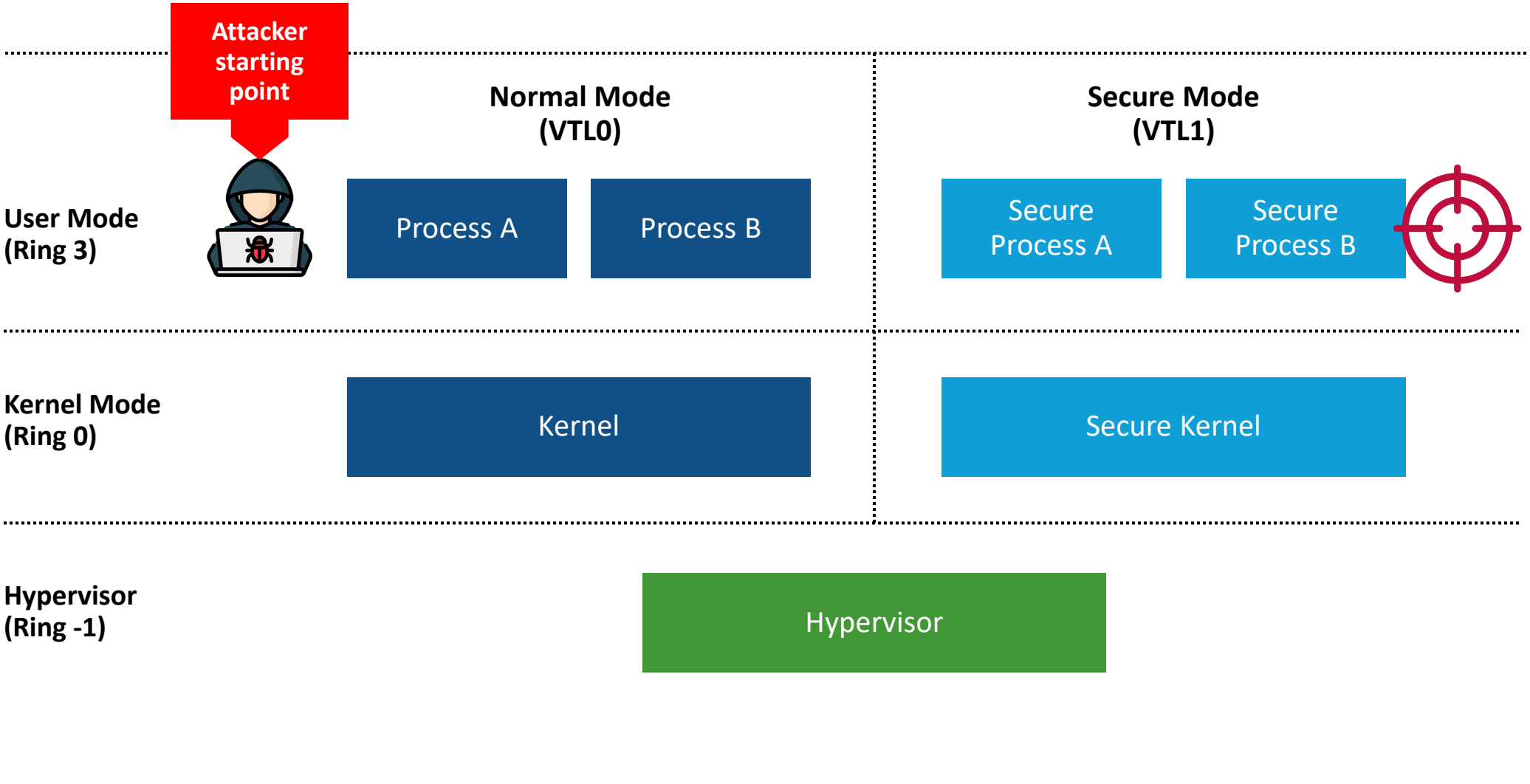
VBS Downgrades Goals

Understand if downgrade mitigation exists in the virtualization stack components

Aim to downgrade to vulnerable code

Major downgrade without vulnerable code is still a vulnerability

VBS Target – Isolated User Mode



Targeting Credential Guard Isolated User Mode Process

Implemented in Ring3-VTL1 as an Isolated User Mode process **LsaIso.exe**

LsaIso.exe contains secrets instead of the original Lsass.exe

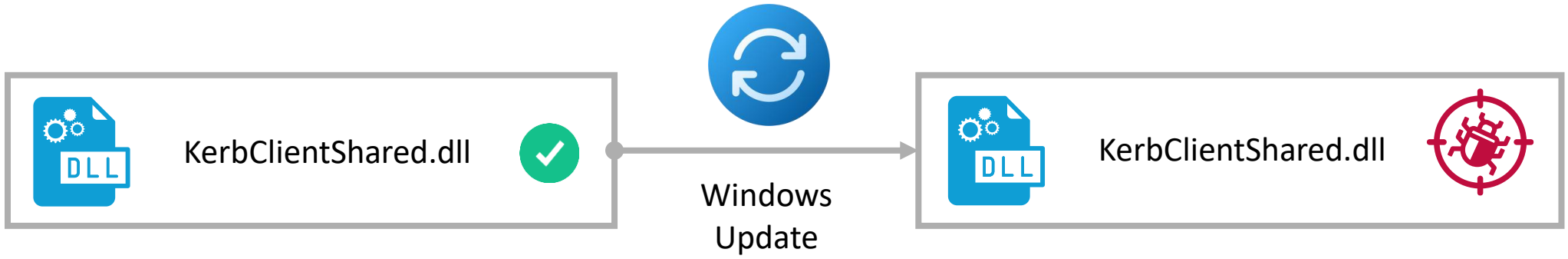
Lsass.exe proxy authentication through LsaIso.exe

Bringing CVE-2022-34709 Back To Life – Credential Guard Elevation of Privilege

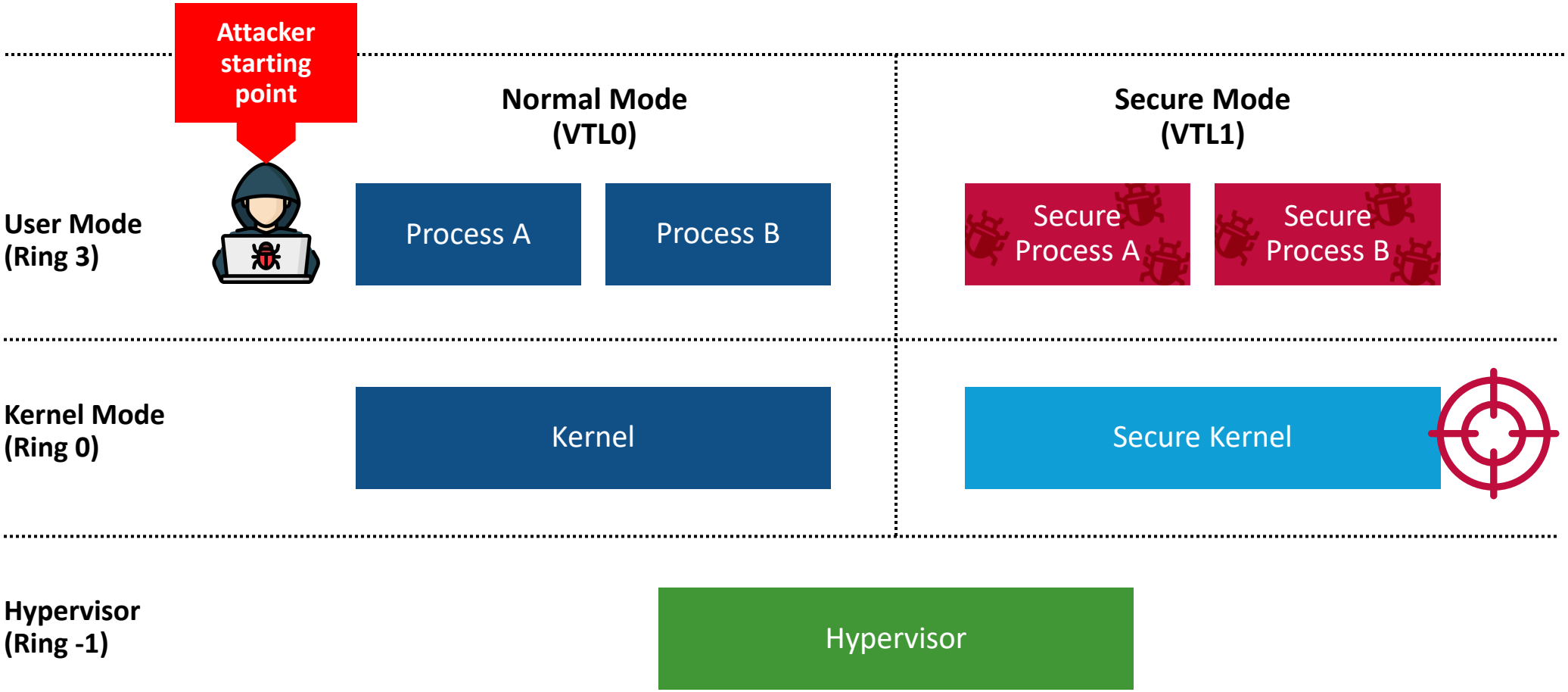
Vulnerable module is **KerbClientShared.dll (10.0.22000.856)**

Downgrading KernClientShared.dll to its vulnerable version worked!

Crossed security boundary is Ring3-VTL0 to Ring3-VTL1



VBS Target – Secure Kernel



Secure Kernel

SecureKernel.exe serves as the kernel for Secure Mode (VTL1)

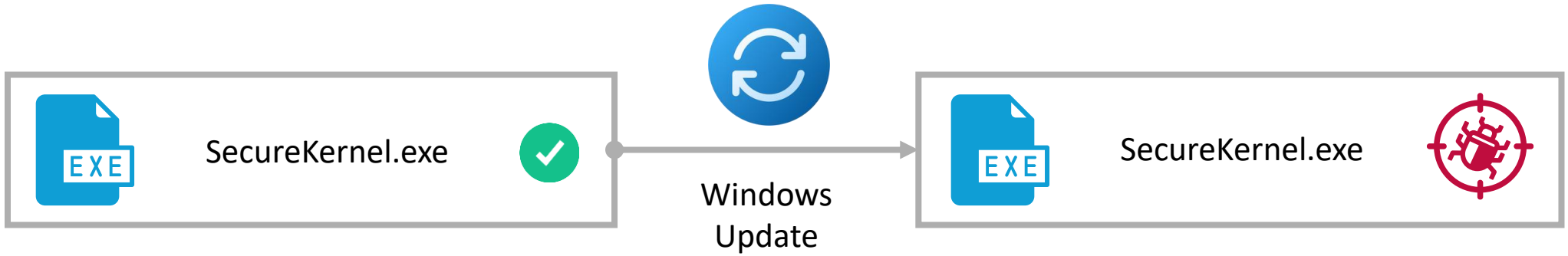
Implements security features such as HVCI, HyperGuard and more.

Bringing CVE-2021-27090 Back To Life – Secure Kernel Elevation of Privilege

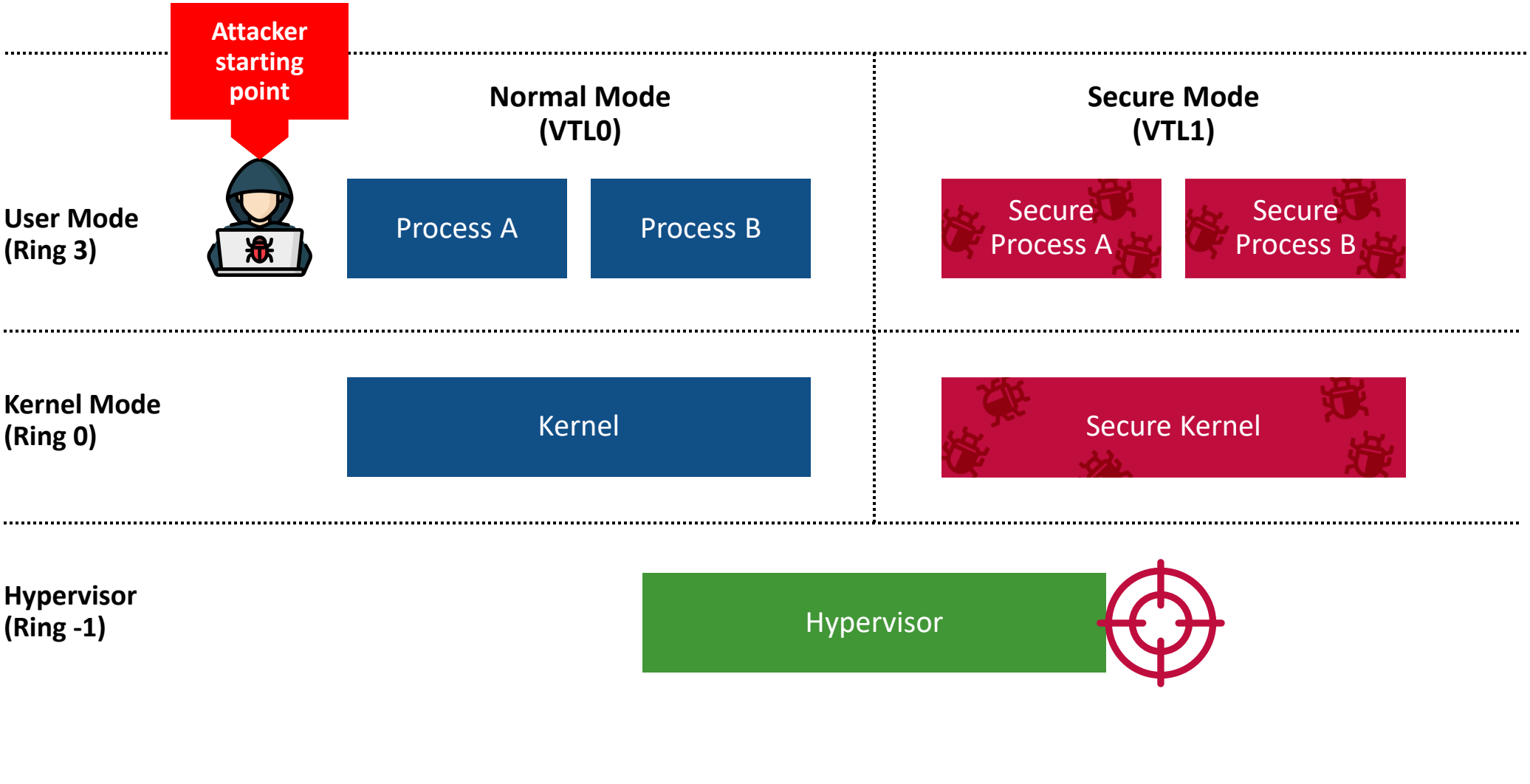
Vulnerable module is **SecureKernel.exe (10.0.19041.207)**

Downgrading **SecureKernel.exe** with some of its dependencies such as **SKCI.dll** and **CI.dll** worked!

Crossed security boundary is **Ring3-VTL0** to **Ring0-VTL1**



VBS Target – Hyper-V's Hypervisor



Hyper-V Hypervisor

The Hyper-V hypervisor is
Hvix64.exe (Intel) or **Hvax64.exe** (AMD)

The hypervisor is a standalone micro-kernel – valuable target for downgrade

Downgrading the Hyper-V Hypervisor to a two-year-old hypervisor

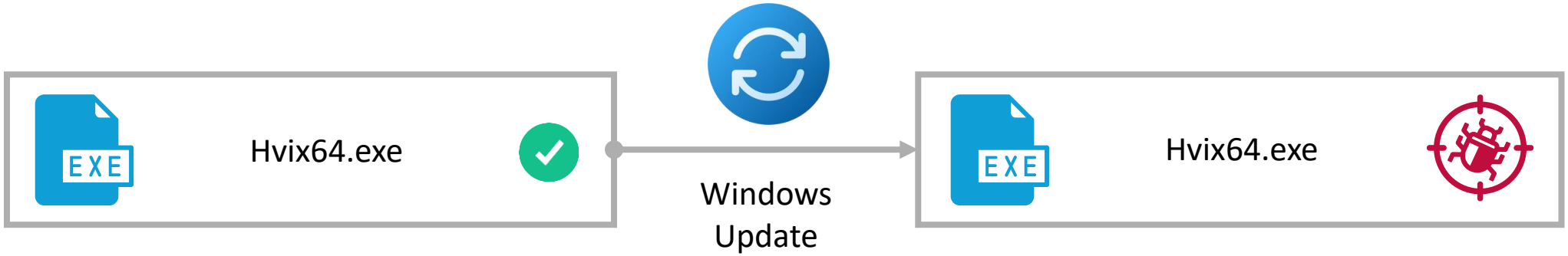
Many Hyper-V Elevation of Privileges have been found in the last two years

Microsoft does not share the vulnerable component in the Hyper-V stack

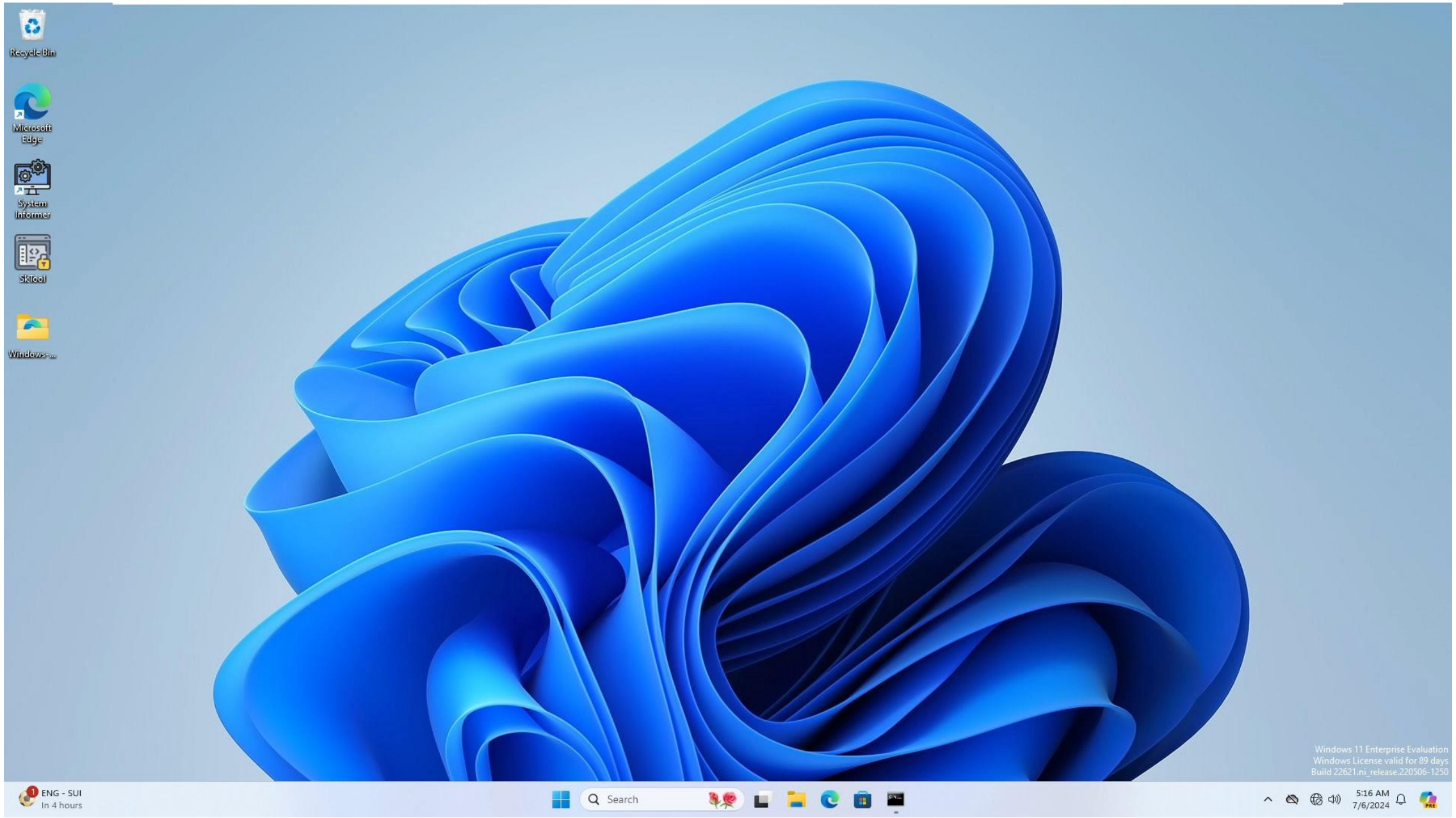
I decided to go **two years backward (10.0.22000.282)** to prove the vulnerability

Downgrading the hypervisor with its loader **HvLoader.dll** worked!

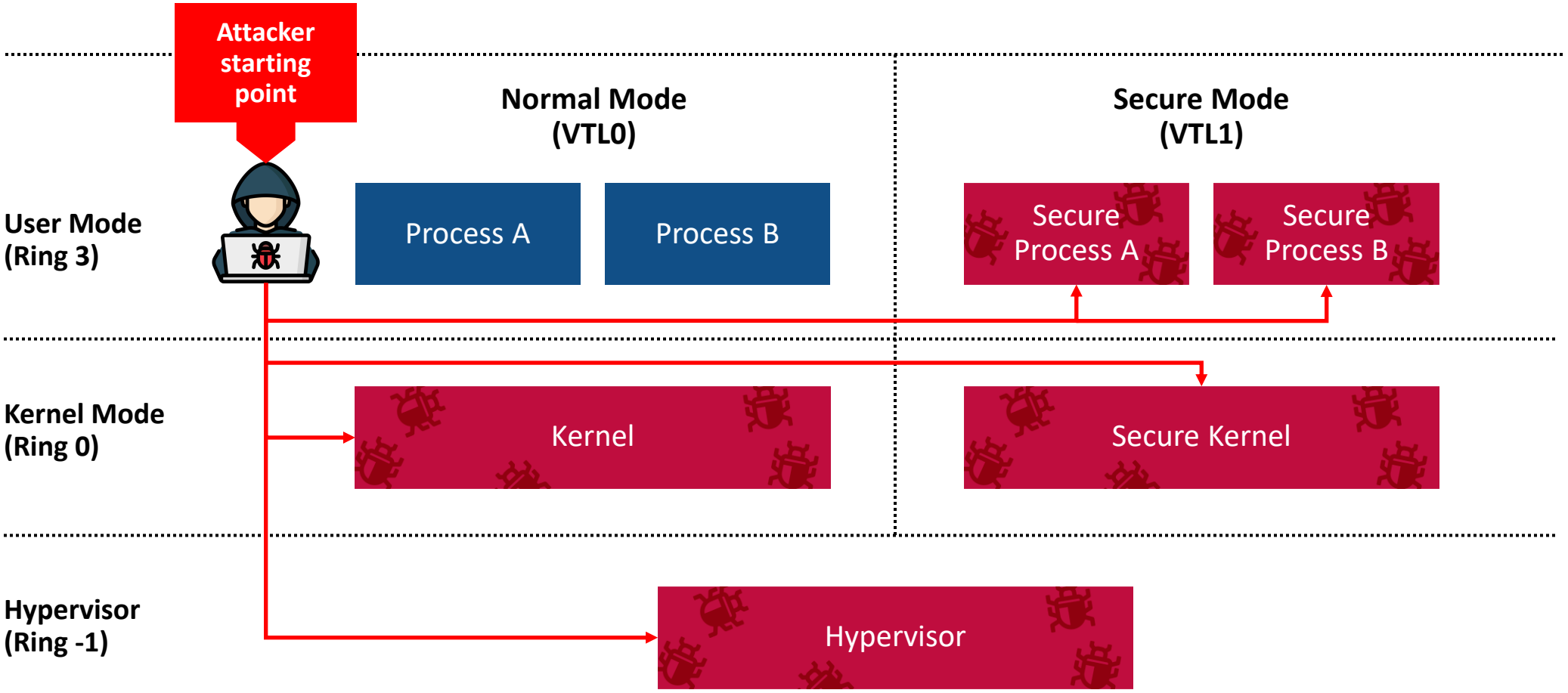
Crossed security boundary is **Ring3-VTL0** to **Ring -1**



Demo #3


Recycle Bin
Microsoft Edge
System Informer
SkTool
Windows-...
Windows 11 Enterprise Evaluation
Windows License valid for 89 days
Build 22621.ni_release.220506-1250
ENG - SUI
In 4 hours
Search
5:16 AM
7/6/2024

Downgrade Attacks Implications



WINDOWS DOWNDATE

Windows Update Restoration Vulnerability

It All Started With Windows.old

Windows Quality Updates saves the old operating system in **C:\Windows.old** for restoration purposes

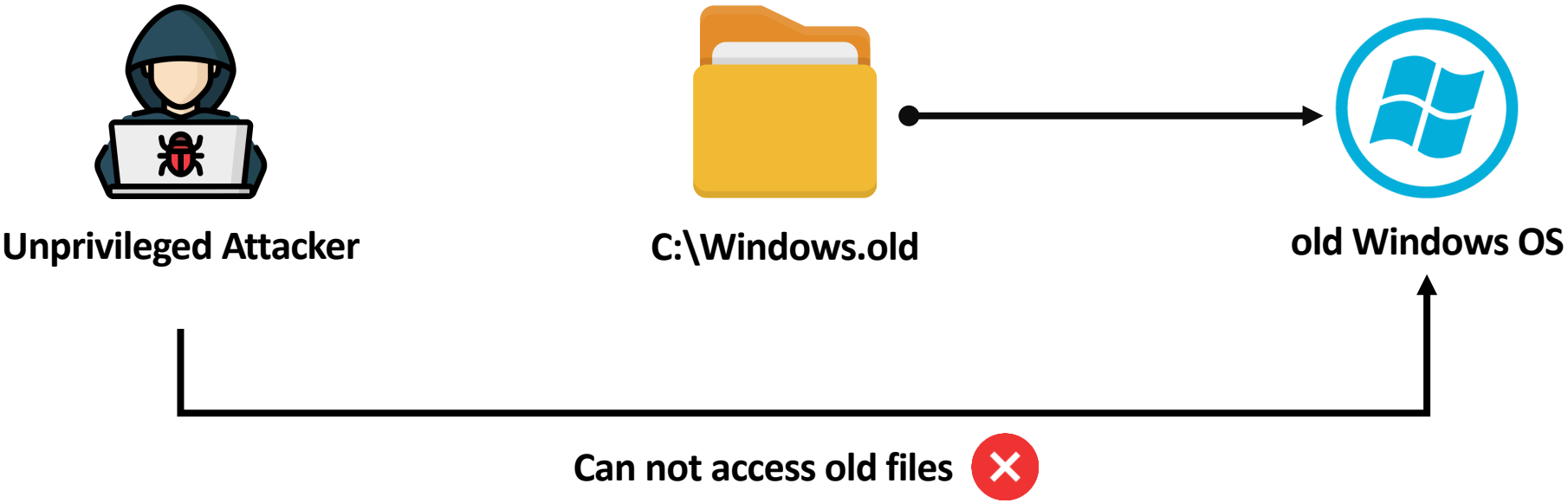
C:\Windows.old

old Windows OS

Can Windows.old contents be tampered with?

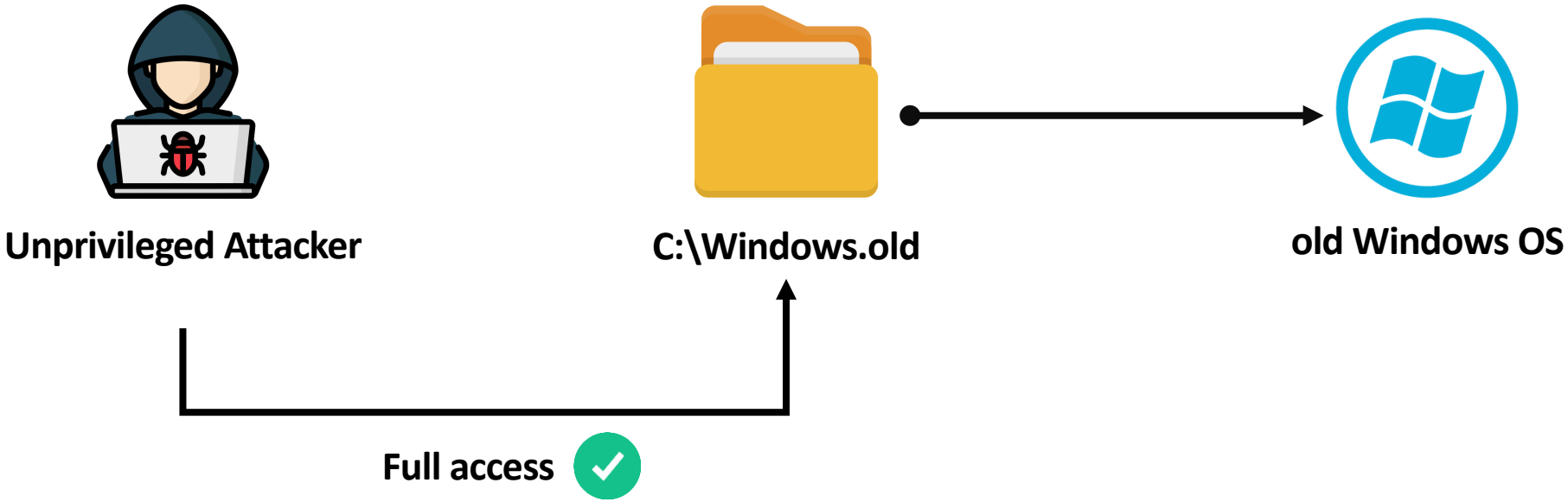
Access lists are copied from the old OS

It is impossible to temper with files that could not be previously tampered with



Can Windows.old be tampered with?

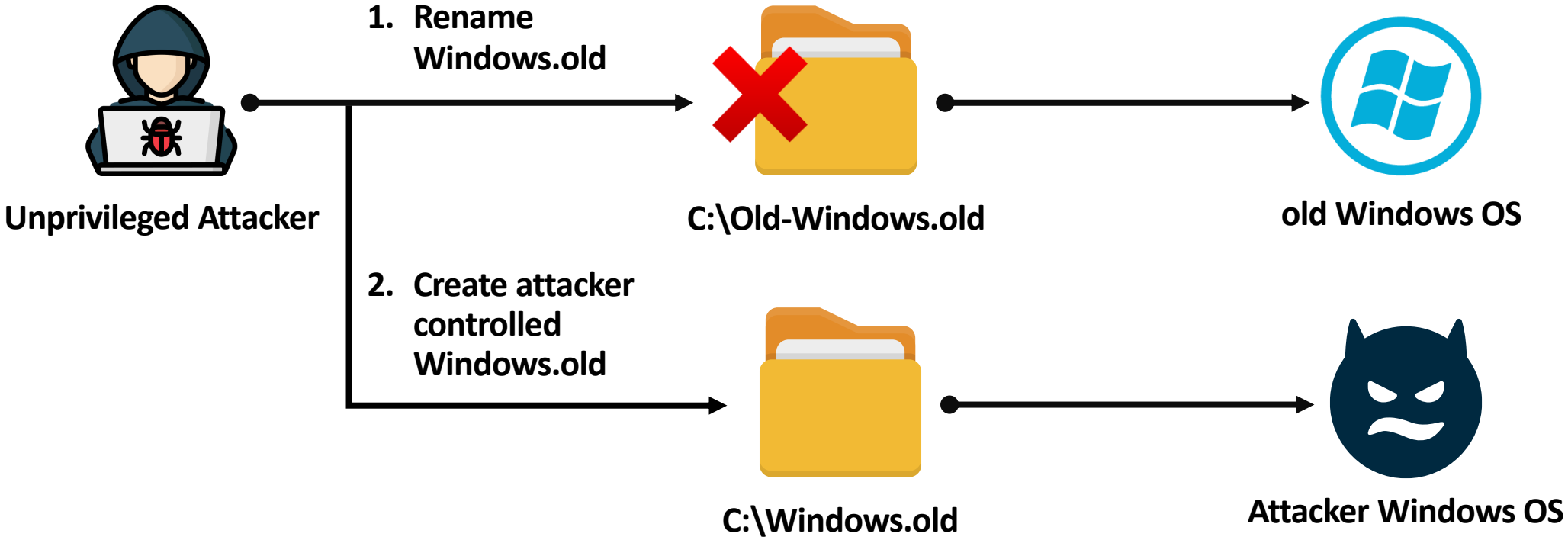
Unprivileged users have **full access** to C:\Windows.old itself!



Exploitation Strategy

Attacker can rename C:\Windows.old and re-create
an attacker-controlled Windows.old

As a result, the attacker-controlled OS is used in case of update restoration!



Closing Remarks

Responsible Disclosure and CVE

We responsibly disclosed all the research findings to Microsoft in February 2024

Microsoft issued **CVE-2024-21302**

Microsoft’s Official Response

We appreciate the work of SafeBreach in identifying and responsibly reporting this vulnerability through a coordinated vulnerability disclosure. We are actively developing mitigations to protect against these risks while following an extensive process involving a thorough investigation, update development across all affected versions, and compatibility testing, to ensure maximized customer protection with minimized operational disruption.

Next Steps

Are there additional Windows features vulnerable to downgrade attacks?

Linux Virtualization-Based Security (LVBS) was introduced, does the same design issues exist in the Linux implementation?

Are other operating systems such as Linux or MacOS vulnerable to downgrade attacks?

Takeaways

Awareness and mitigations against OS downgrade attacks

Takeaways

Design must be regarded as a relevant attack surface

Takeaways

Thoroughly examine and expand in-the-wild attacks

Credits

James Forshaw **@tiraniddo**		**CVE-2022-34709**
Saar Amar **@AmarSaar**		**CVE-2021-27090**
Gabriel Landau **@GabrielLandau**		**PPLFault**
Valentina Palmiotti **@chompie1337**	Ruben Boonen **@FuzzySec**	**CVE-2023-21768 Exploit**
Benjamin Delphi **@gentilkiwi**		**Mimikatz**

Thank You!

 **@_0xDeku**

 **linkedin.com/in/alonleviev**

 **alon.leviev@safebreach.com**