DEFYING DEFENDER
by
Outsmarting Static Signatures

Cedric Van Bockhaven
August 2024

SIKKERHETS FESTIVALEN

OUTFLANK
clear advice with a hacker mindset

ABOUT YOUR SPEAKERS

Cedric Van Bockhaven - @c3c

- Red Teamer and Offensive Developer @ Outflank
- Network security background / R&D new attack vectors
- "Challenge the Cyber" CTF in The Netherlands

OUTFLANK

- Outflank Security Tooling (OST)
- Red Teaming Services

AGENDA

- Introduction
  - Context
  - Problem statement
  - Gameplan
- Executing the gameplan
  - Automating Defender
  - Localizing signatures
  - Evading signatures
- Demo
- Summary


INTRODUCTION

MDI vs MDE vs MDAV

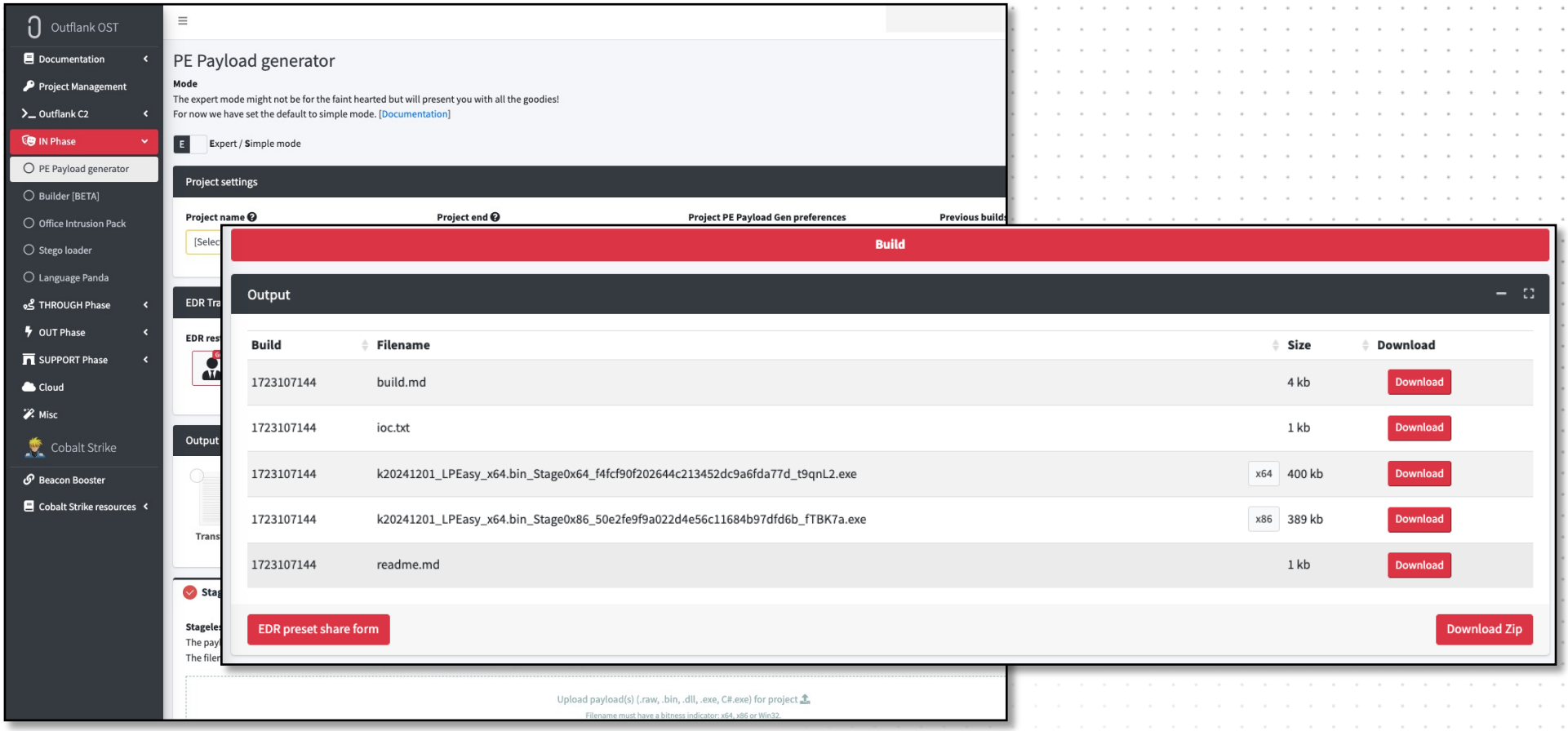
- MDI (Identity) → domain activity
- MDE (Endpoint) → cloud integrated / centralized reporting
- MDAV (Antivirus) → protection component of MDE

STATIC SIGNATURES

- Today's focus will be: **static signature types**
- Defender also has behavioral signatures and ML signatures
  - We are not looking at these

CONTEXT

- OST payload generation



CONTEXT

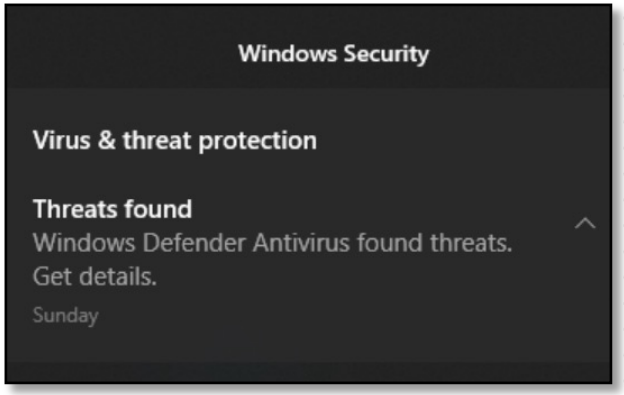
- Detection in multiple of our payloads

[redacted] Nov 13th, 2023 at 3:15 PM

Hotfix relating Defender detection

Earlier today initial reports came in from 2 customers on Microsoft Defender picking up different samples, likely due to updated definitions. An investigation was started in our labs.

After analysis we identified a combination of syscalls that were being picked up as "[redacted]Inject".
The offending signature/rule being triggered appears to be a very specific check. Hence we developed a quick fix. Around 14:00 CET, we started out rolling out a hotfix that is fully deployed right now.



ANALYSIS

- matterpreter's DefenderCheck / rastamouse's ThreatCheck

```
[!] Identified end of bad bytes at offset 0x804
00000000   8B D1 48 89 4C 24 28 48  89 5C 24 48 4C 89 4C 24   ?ÑH?L$(H?\$HL?L$
00000010   50 33 F6 4C 8B 0B 4C 89  4C 24 40 48 8B 43 08 49   P3öL?·L?L$@H?C·I
00000020   2B C1 48 C1 F8 02 4D 8B  55 00 49 8B 4D 08 49 2B   +AHAo·M?U·I?M·I+
00000030   CA 48 C1 F9 02 45 33 E4  8B C0 48 89 44 24 38 48   EHAù·E3ä?AH?D$8H
00000040   85 C0 0F 84 F5 00 00 00  44 8B F1 90 47 8B 3C A1   ?A·?o···D?ñ?G?<¡
00000050   33 FF 4D 85 F6 0F 84 D1  00 00 00 49 8B D8 49 8B   3ÿM?ö·?Ñ···I?OI?
00000060   EA 49 2B E8 0F 1F 84 00  00 00 00 00 44 3B 3C 2B   êI+è··?·····D;<+
00000070   0F 85 93 00 00 00 44 8B  1B 4C 03 DA 49 C7 C2 FF   ·??···D?·L·UIÇAÿ
00000080   FF FF FF 66 0F 1F 84 00  00 00 00 00 49 FF C2 43   ÿÿÿf··?·····IÿAC
00000090   80 3C 13 00 75 F6 49 FF  CA 49 83 FA FF 75 08 41   ?<··uöIÿEI?úÿu·A
000000A0   B8 FF FF FF FF EB 42 49  8D 4A FF 49 8B D3 E8 39   ,ÿÿÿÿëBI?JÿI?Oè9
000000B0   21 00 00 44 8B C0 43 0F  B6 14 1A 8D 42 BF 33 C9   !··D?AC···?B¿3É
000000C0   3C 19 B8 20 00 00 00 0F  46 C8 0A D1 0F BE C2 49   <·, ····FE·Ñ·_AI
000000D0   33 C0 0F B6 C8 41 C1 E8  08 48 8D 05 3C 93 02 00   3A·EAAè·H?·<?··
```

```
PS C:\temp> python3
Python 3.11.6 (tags/v3.11.6:8b6ee5b, Oct  2 2023, 14:57:12) [MSC v.1935 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> ff = open("                                                                    .dll","rb").read()
>>> open("test.bin","wb").write(ff[:0x800] + b'\x00'*0x100 + ff[0x900:])
196608
>>> exit()
PS C:\temp> .\ThreatCheck.exe -f test.bin -e Defender
[+] No threat found!
[*] Run time: 0.09s
PS C:\temp>
```

ANALYSIS

- Detection 1:

```
{
  if ( *v19 != 'Z' && v19[1] == 'w' )
    ++v9;
  goto LABEL_17;
```

- Native system call implementation
- Zw… E.g. ZwQueryInformationProcess

```
cmp byte ptr [r11], 0x5a
```

- Extremely wide signature

ANALYSIS

- Detection 2:
  - CRC function resolution

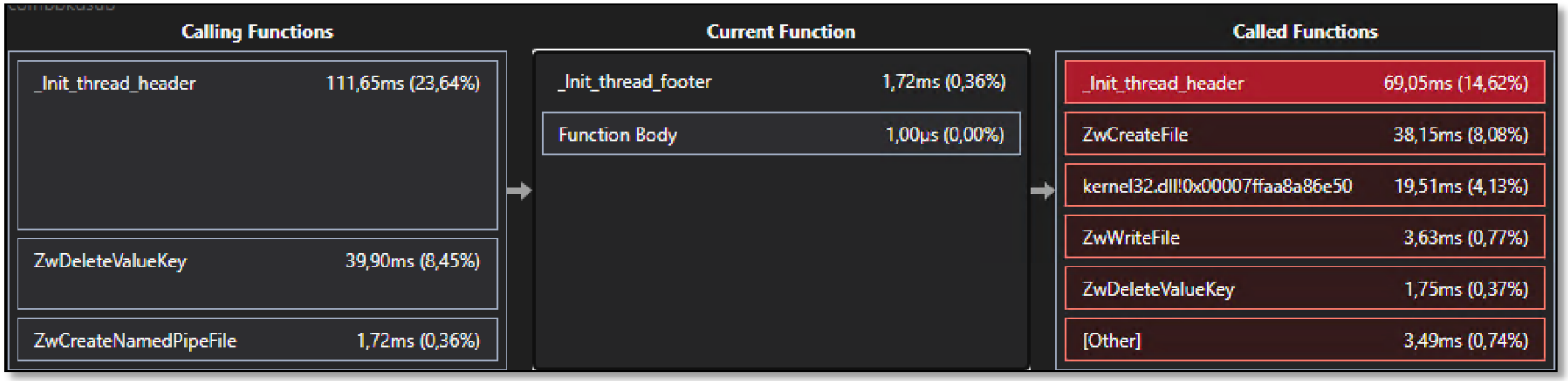
```
//runtime combine (case insensitive)
uint32_t combine_crc32_s_lower(size_t idx, const char* str, uint32_t part)
{
    char c = str[idx];
    c |= ((((uint8_t)(c - (char)N((int)'A'))) < (byte)N((int)' ')) << 5);
    return (part >> 8) ^ crc_table[(part ^ c) & 0x000000FF];
}

uint32_t combine_crc32_s_lower(size_t idx, const wchar_t* wstr, uint32_t part)
{
    wchar_t c = wstr[idx];
    c |= ((((uint16_t)(c - (char)N((int)'A'))) <= (byte)N((int)' ')) << 5);
    return (part >> 8) ^ crc_table[(part ^ c) & 0x000000FF];
}
```

```
if ( *(_WORD *)(v15 + 6) )
{
  v18 = *(unsigned __int16 *)(v15 + 20);
  while ( 1 )
  {
    v19 = (unsigned int *)(v15 + v18 + 40i64 * v16);
    v20 = -1i64;
    do
      ++v20;
    while ( *((_BYTE *)v19 + v20 + 24) );
    v21 = v20 - 1;
    if ( v21 != -1 )
    {
      v22 = sub_180003580(v21 - 1, v19 + 6, v8, v12);
      v24 = *((_BYTE *)v19 + v23 + 24);
      v25 = 0;
      if ( (unsigned __int8)(v24 - 65) <= 25u )
        v25 = 32;
      if ( ~(dword_1800295E0[(unsigned __int8)(v22 ^ (v25 | v24))] ^ (v22 >> 8)) == 0xA21C1EA3 )
        break;
    }
    if ( ++v16 >= v17 )
```

ANALYSIS

- Profiler → Defender sigs on functions with high number of calls



C:\dev\	472,36ms (100,00%)	0ns (0,00%)	
kernel32.dll!0x00007ffaa8a86e50	19,51ms (4,13%)	19,51ms (4,13%)	646915
crc32_ns::crc32_getA	158,10µs (0,03%)	93,70µs (0,02%)	1086
	34,60µs (0,01%)	34,60µs (0,01%)	1086
	29,80µs (0,01%)	29,80µs (0,01%)	1086
strcpy_s	4,90µs (0,00%)	4,90µs (0,00%)	86
kernelbase.dll!0x00007ffaa6e30720	1,70µs (0,00%)	1,70µs (0,00%)	4
wcscat_s	500,00ns (0,00%)	500,00ns (0,00%)	4
kernel32.dll!0x00007ffaa8a77d00	300,00ns (0,00%)	300,00ns (0,00%)	3
ntdll.dll!0x00007ffaa97b9700	300,00ns (0,00%)	300,00ns (0,00%)	3

ANALYSIS

- Must be a better way to do things?
- Can we find out exactly what triggers detection
- Additionally: find out early if a new signature leads to detection

- **Automate!**
  - Pull in signature updates periodically
  - Parse signatures, find offending parts
  - Modify code, avoid the signature

GAMEPLAN

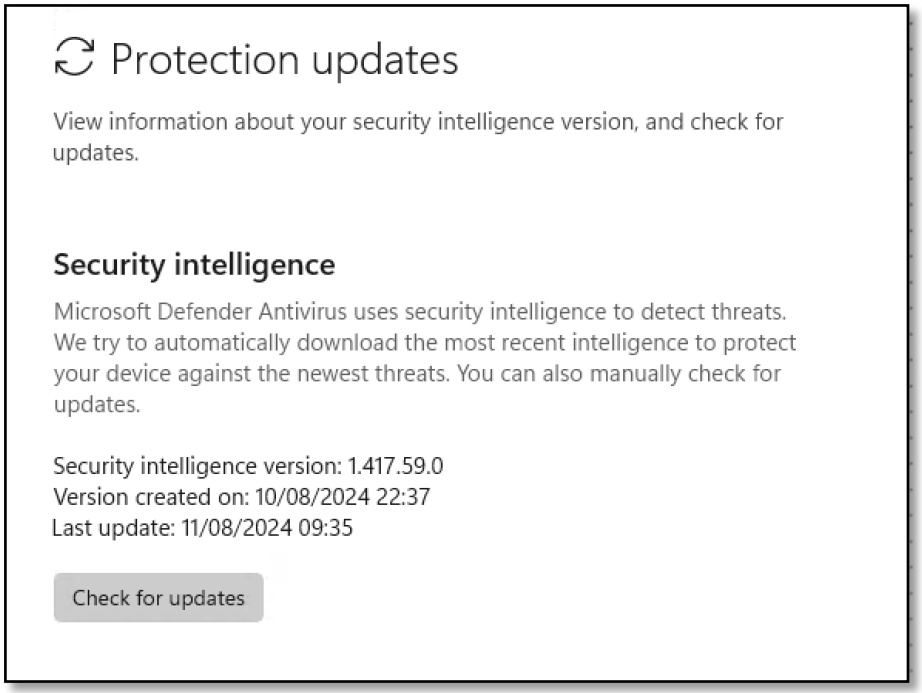
- Obtain detections/signatures for an input file
- Modify our input file accordingly
- Repeat

1. Automate Defender
2. Localize signature
3. Modify code:
   - With source code: LLVM obfuscation passes?
   - No source code: Alcatraz?

AUTOMATING DEFENDER

MDAV ARCHITECTURE

- Defender updates and scans:
  - Update frequency:
    - Intelligence (signatures):
      - Multiple times per day by MS
      - Downloaded once per day (default)
    - Platform:
      - Updated once per month by MS
  - Endpoint scan frequency (default):
    - Directly after update
    - Daily at 2AM

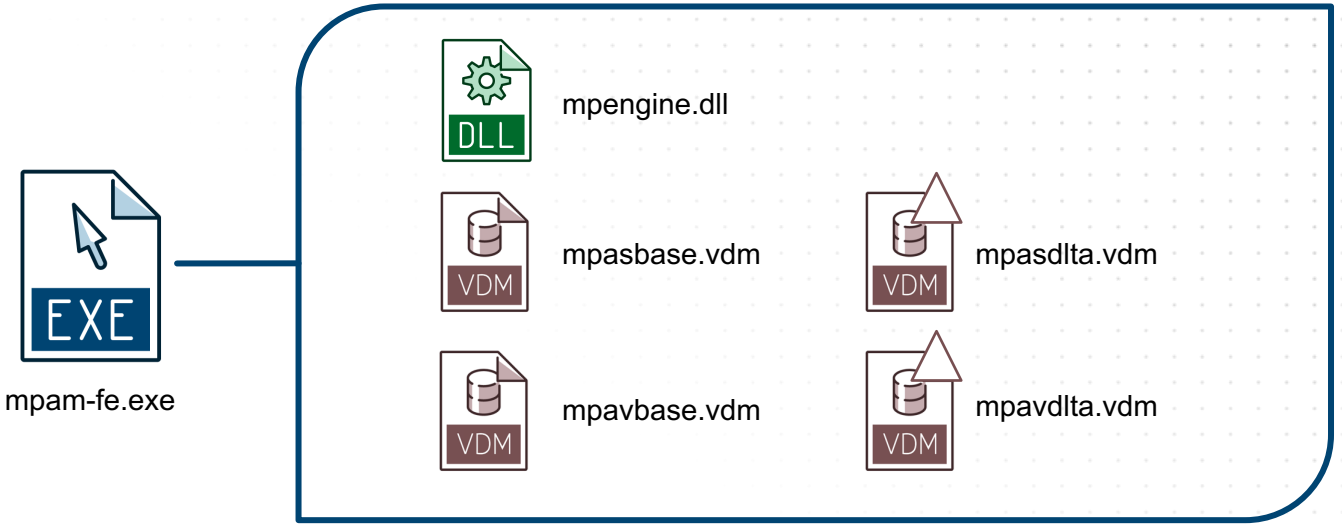


Source: https://learn.microsoft.com/en-us/defender-endpoint/schedule-antivirus-scans-group-policy

Source: https://learn.microsoft.com/en-us/windows-hardware/customize/desktop/unattend/security-malware-windows-defender-signatureupdateinterval

MDAV ARCHITECTURE

- Update format:
  - MPAM-FE.exe (Microsoft Protection Antimalware Front End)
  - Cabinet format
    - Engine
    - VDM / Virus definition module: base (monthly) + delta (daily)



DEFENDER PRETENDER

- Research by SafeBreach Labs (Tomer Bar, Omer Attias)
  - Analysis of the Defender Update Process
  - How deltas are applied to the base VDM files
  - Dissecting signature types

  - CVE-2023-24934: apply fake updates (removing signatures)

  - wd-pretender.py
    - Implements CVE-2023-24934
    - Remove signatures from signature database
    - Python parser

DEFENDER PRETENDER

```
C:\wd\wd-pretender>python wd-pretender.py bypass mimikatz

        -- Defender-Pretender: v1.0.0 (SafeBreach Labs) --

[+] Getting Signatures Location ...
[+] Definitions Path: C:\ProgramData\Microsoft\Windows Defender\Definition Updates\{C5284901-BDA1-43BD-A800-F7852FE50DC9}
[+] Loading mpasbase.vdm
[+] Loading mpasdlta.vdm
[+] Loading mpavbase.vdm
[+] Loading mpavdlta.vdm
[+] Enumerating Anti-Virus Definitions
[+] Threats Containing: mimikatz
        Deleting => b'\xd8!Mimikatz'
        Deleting => b'HackTool:Win64/MIMIKATZ'
        Deleting => b'\xd8!Mimikatz!dha'
        Deleting => b'HackTool:Win64/Mimikatz!dha'
        Deleting => b'Behavior:Win32/InvokeMimikatz.A!dha'
        Deleting => b'Behavior:Win32/LsassMimikatzOpen.A'
        Deleting => b'Trojan:PowerShell/Mimikatz'
        Deleting => b'\xd8!Mimikatz.A!dha'
        Deleting => b'\xd8!Mimikatz!rfn'
        Deleting => b'HackTool:Win64/Mimikatz.A'
        Deleting => b'\xd8!Mimikatz.A!dha!!Mikatz.gen!A'
        Deleting => b'HackTool:PowerShell/Mimikatz'
        Deleting => b'Trojan:PowerShell/Mimikatz.A'
        Deleting => b'Behavior:Win32/MimikatzTrigger'
        Deleting => b'Behavior:Win32/MimikatzTrigger.B'
        Deleting => b'Behavior:Win32/MimikatzTriggerv2'
        Deleting => b'Behavior:Win32/MimikatzTrigger.C'
        Deleting => b'Behavior:Win32/MimikatzTrigger.D'
        Deleting => b'\xad\x01Mimikatz'
        Deleting => b'\xac!Mimikatz'
        Deleting => b'\xd8!Mimikatz.B'
        Deleting => b'HackTool:PowerShell/Mimikatz.C'
        Deleting => b'\xd9AMimikatz'
```

Hier de schakel:

wd-pretender is handig voor signatures te gaan analy

we hebben ook manier nodig om automatisch te valideren

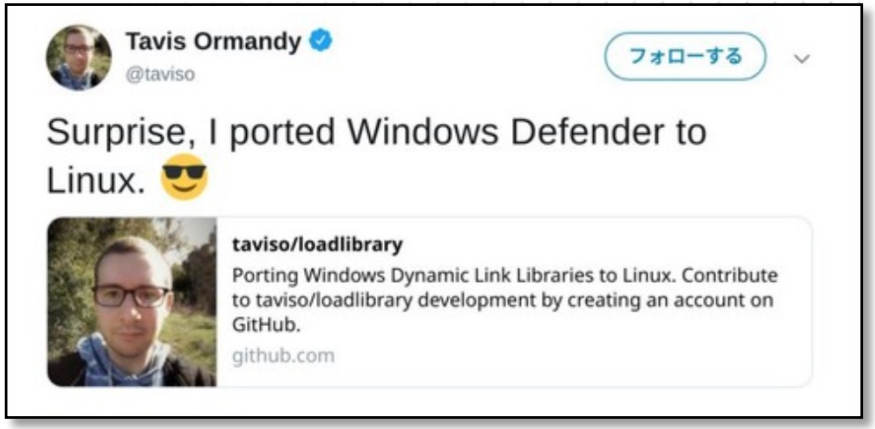
extra slide aan start van section:

what do we need to automate?

- automated signature analysis
- validate signatures

DEFENDER ON LINUX

- "LoadLibrary", research by Tavis Ormandy
  - Linux PE loader, derived from ndiswrapper
    - *A custom PE/COFF loader derived from ndiswrapper. The library will process the relocations and imports, then provide a dlopen-like API. The code supports debugging with gdb (including symbols), basic block coverage collection, and runtime hooking and patching.*
  - Call functions in a DLLs from a Linux environment
    - Replacement/stub functions for Windows API calls
  - Originally also for AV research: mpclient (2017)



LOADLIBRARY

- MpEngine startup fails

```
CloseThreadpoolTimer(): 0x41414141
EtwUnregister(): 455457
AcquireSRWLockExclusive(): 0x5ad8af84
EtwUnregister(): 455457
ReleaseSRWLockExclusive(): 0x5ad8af84
SetThreadpoolTimer(): 0x41414141, (nil), 0, 0
WaitForThreadpoolTimerCallbacks(): 0x41414141, 1
CloseThreadpoolTimer(): 0x41414141
EtwUnregister(): 455457
main(): __rsignal(RSIG_BOOTENGINE) returned failure, missing definitions?
main(): Make sure the VDM files and mpengine.dll are in the engine directory
main(): usage: ./mpclient [filenames...]
```

- Find old versions of mpam-fe.exe
  - Somewhere between 4 and 11 Aug 2021 the engine was modified

INTERNET ARCHIVE
WayBackMachine

Name	Date modified
engine_20210602035005_working	02/08/2024 14:13
engine_20210714035002_working	02/08/2024 14:18
engine_20210804035003_working	02/08/2024 14:35
engine_20210811035003_fail	02/08/2024 15:05
engine_20210901035002_fail	02/08/2024 15:06

LOADLIBRARY

```
        v15 = *((_DWORD *)this + 261) - v14;
        HIDWORD(Block) = i + 1;
        v16 = v26;
        if ( i + 1 >= (unsigned int)(v15 >> 4) )
          break;
        if ( *(_QWORD *)(v14 + 16 * i + 8) > *(_QWORD *)(v14 + 16 * i + 16) )
        {
          if ( WPP_GLOBAL_Control != &WPP_GLOBAL_Control && (*((_BYTE *)WPP_GLOBAL_Control + 28) & 8) != 0 )
            WPP_SF_(0x1Bu, &WPP_bd8b580b46723a06a0cf7ab171f0eabd_Traceguids, *((_QWORD *)WPP_GLOBAL_Control + 2));
          return 0x80070570;
        }
      }
      *((_BYTE *)this + 1024) = 0;
      *(_DWORD *)v34 = ValidateTrust::ValidateTrustPluginPe::VerifyEmbeddedSignature((struct PEFileReader **)this);
      v28 = 0;
      v29 = 0;
      v30 = 0;
```

- Now verifies embedded signatures
  - `ValidateTrust::ValidateTrustPluginPe::VerifyEmbeddedSignature`
  - Fails for whatever reason ¯_(ツ)_/¯
  - Patch the function to return SUCCESS status

DEFENDER ON LINUX

```
C:\wd\loadlibrary\inputbins>"%ProgramFiles%\Windows Defender\MpCmdRun.exe" -Scan -ScanType 3 -File C:\wd\loadlibrary\inputbi
Scan starting...
Scan finished.
Scanning C:\wd\loadlibrary\inputbins\mimikatz_trunk.7z found 1 threats.

<=========================LIST OF DETECTED THREATS=========================>
---------------------------- Threat information -----------------------------
Threat                  : HackTool:Win32/Mimikatz
Resources               : 1 total
    file                : C:\wd\loadlibrary\inputbins\mimikatz_trunk.7z
-----------------------------------------------------------------------------
```

```
user@DESKTOP-M89B81E:/mnt/c/wd/loadlibrary$ ./mpclient inputbins/mimikatz_trunk.7z
fix_pe_image(): imgbase = 0x5a100000
main(): Scanning inputbins/mimikatz_trunk.7z...
EngineScanCallback(): Scanning input
EngineScanCallback(): Threat HackTool:Win32/Mimikatz identified.
user@DESKTOP-M89B81E:/mnt/c/wd/loadlibrary$
```

LOADLIBRARY

- Some limitations:
  - Single DLL load
  - MSVCRT are forwarded to the linux CRT
  - Other functions are stubs / reimplementations

- Cabinet.DLL -> Compression of CAB files
  - E.g. compress with LZX

```
user@DESKTOP-M89B81E:/mnt/c/wd/loadlibrary$ ./compress
fix_pe_image(): imgbase = 0x10000000
main(): fcicreate result 0x57c46350
main(): erf: 0 0 0
main(): fciaddfile: 1
main(): fciflushcabinet: 1
main(): fcidestroy: 1
```

- WASI SDK → Defender in the browser?

INSERT MORE ABOUT SIGNATURES

And how they relate to the original file

And what sig types there are

This is a major challenge: the type of signature set might make it easy or hard to evade.


LOCALIZING SIGNATURES

LOCALIZING SIGNATURES


SIGNATURE EVASION

APPROACHES

- Source not available (PE)

- Binary transformation
  - Alcatraz
    https://github.com/weak1337/Alcatraz
- More error prone

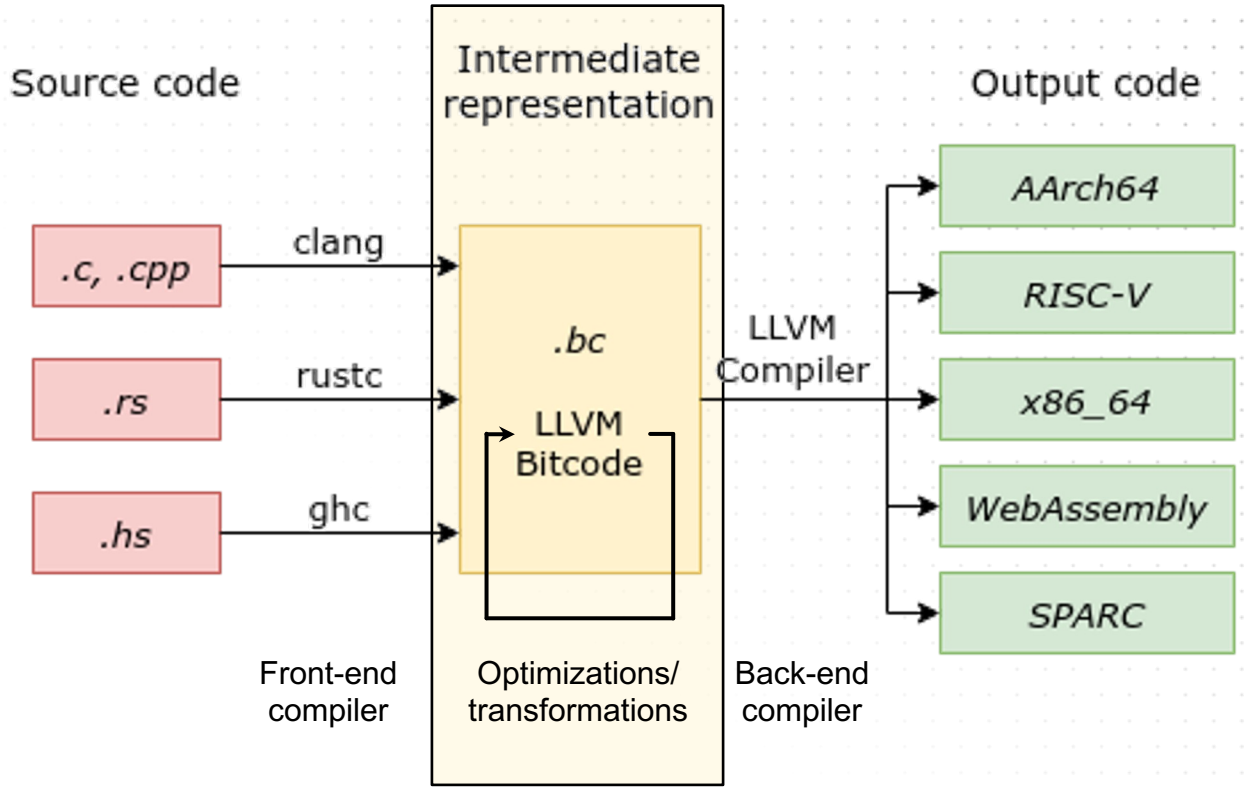
- Source is available

- Source/IR transformation
  - LLVM passes
- More control

- Could still lift the native code to LLVM bitcode (https://github.com/lifting-bits/remill)
- Or combine both

LLVM (Low Level Virtual Machine)

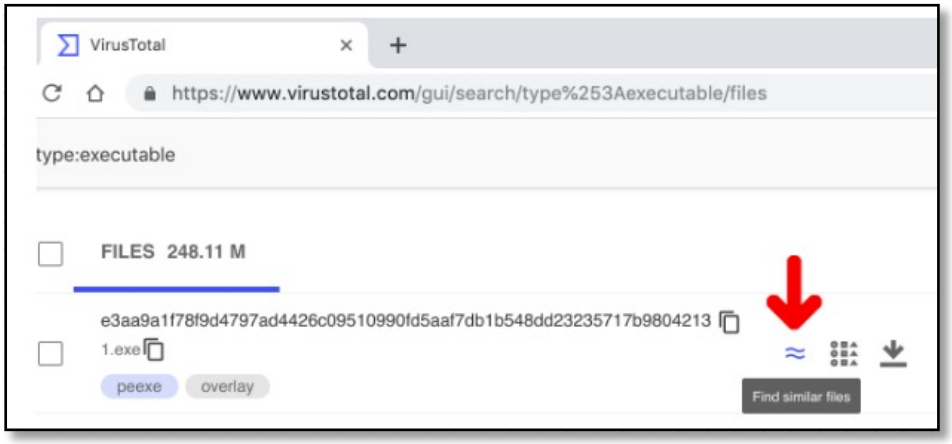
- Construct/optimize/produce intermediate and/or binary machine code
- Compiler framework
  - "front-end" (parser and lexer)
  - "back-end" (LLVM bitcode to binary machine code)



Source: praj.in

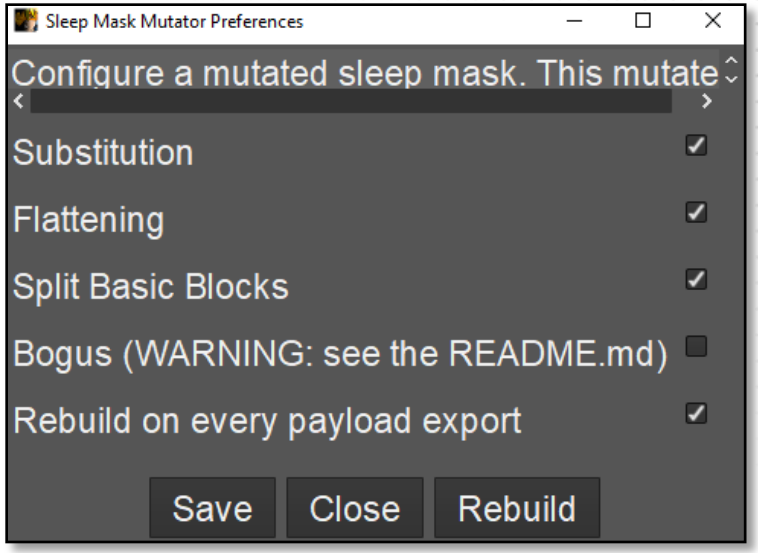
LLVM EXPERIMENTS

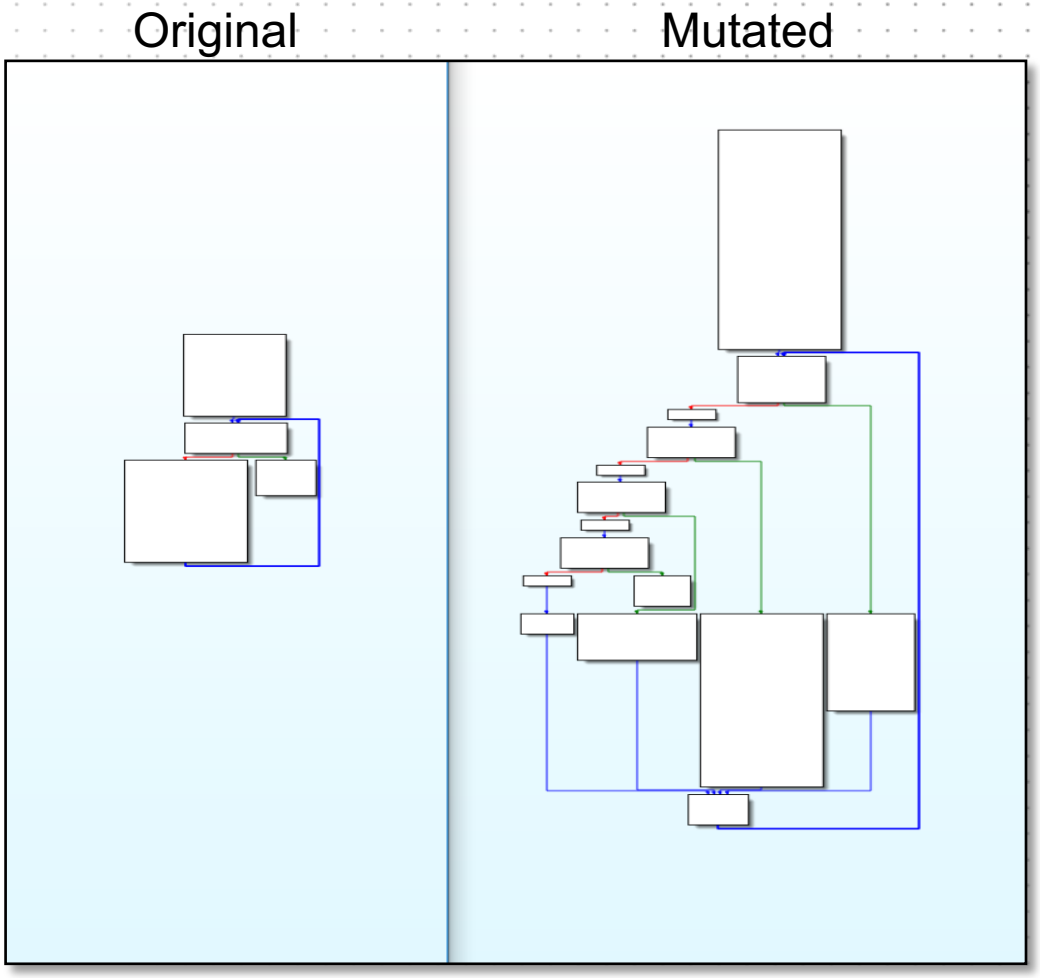
- Blog post by TrustedSec
  - Using LLVM to obfuscate input files
  - Conclusion: not hugely effective
  - But: still nice to introduce variety
  - Question: File similarity?
    - Use other legitimate files to embed our payload
    - Weave multiple files into one
    - Works wonders on “ML” scanners



MUTATOR KIT

- Cobalt Strike
  - Mutated sleep mask
  - Break in-memory scans





LLVM TARGETED OBFU

- Custom passes

DEMO

- Demo!

IN SUMMARY

- Executed the gameplan
  - Automating Defender
  - Localizing signatures
  - Evading signatures

- Periodically fetching new signatures (every 4 hours)
  - Early indication of old payloads getting detected with new signatures

- No silver bullet
  - Use in combination with other techniques

THANKS!

OUTFLANK

clear advice with a hacker mindset

Cedric Van Bockhaven
cedric@outflank.nl
www.outflank.nl/cedric
@c3c