https://shorturl.at/bPCzW

https://retooling.io/blog

Welcome to: An unexpected journey into Microsoft Defender's signature World

Tools you need: Download in this order

1. ResourceHacker (~3MB)
2. build environment based on msys64 (~350MB)
3. labs

https://shorturl.at/bPCzW

Who we are.... *"The Italian doctors"* *and ...*

@DrCh40s

Silvio

- ❑ Co-founder of *Retooling LLC*
- ❑ Former Senior Cyber Security Architect @ LEONARDO Spa - Cyber Security Division
- ❑ Senior Security Researcher @ EMC/RSA -> DELL – Center of Excellence
- ❑ Malware reverse engineer @ Symantec - Security Response
- ❑ PhD Network Security @ University of Pisa
- ❑ M.Sc. in Computer Engineering

Antonio

- ❑ Co-founder of *Retooling LLC*
- ❑ Former Senior Cyber Security Architect @ LEONARDO Spa - Cyber Security Division
- ❑ Cyber Threat Analyst / Reverse Engineer
- ❑ PhD System Security @ University of Roma Tre
- ❑ M.Sc. in Computer Science

Davide

- ❑ Master's degree student in Cybersecurity @ University of Sapienza
- ❑ Bachelor's degee in Information Technology @ University of L'Aquila
- ❑ Passionate about malware analysis and reverse engineering

@t0nvi

@davidefont96

silvio@retooling.io

antonio@retooling.io

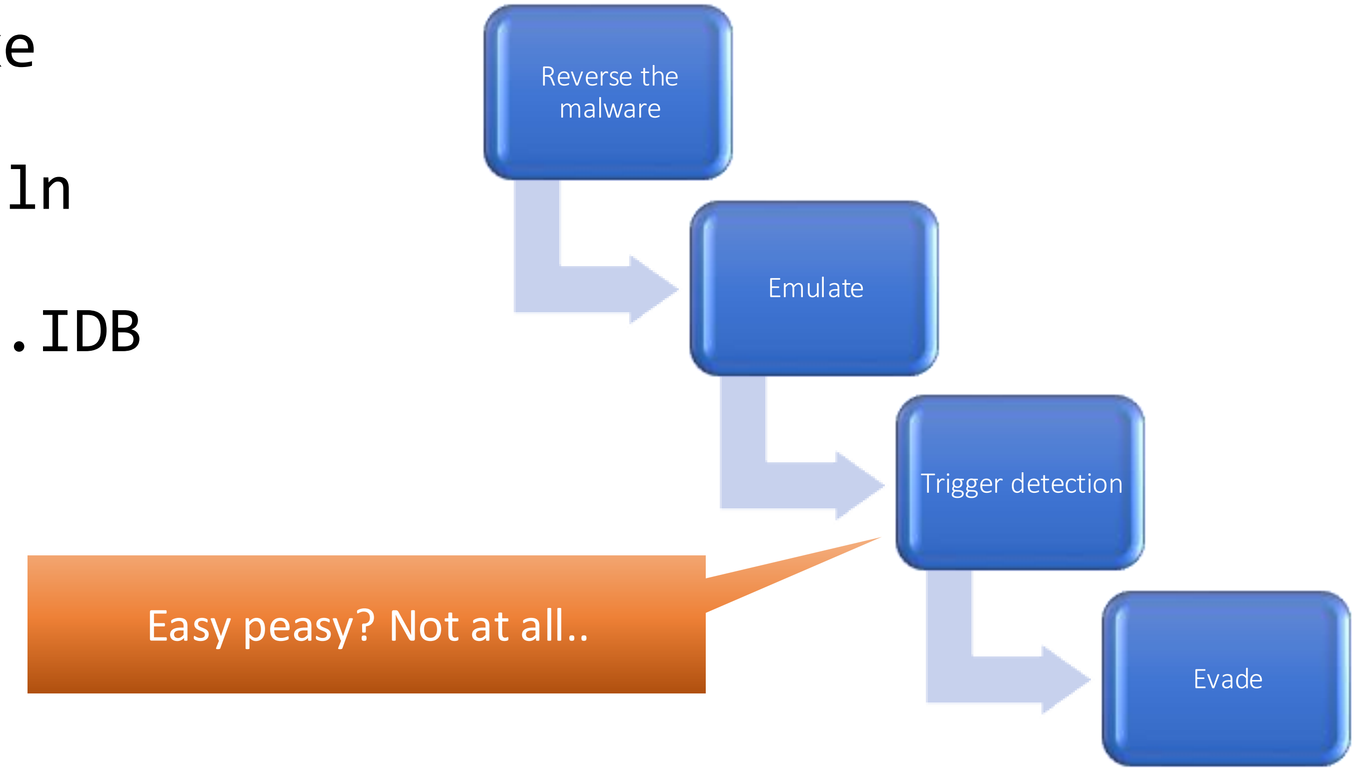
davidefontana96.df@gmail.com

An *unexpected* journey…

- ❑ **Starting point:** `PingPull.exe`
- ❑ **Initial objective:** `PingPull.sln`
- ❑ **Where we end up:** `Defender.IDB`

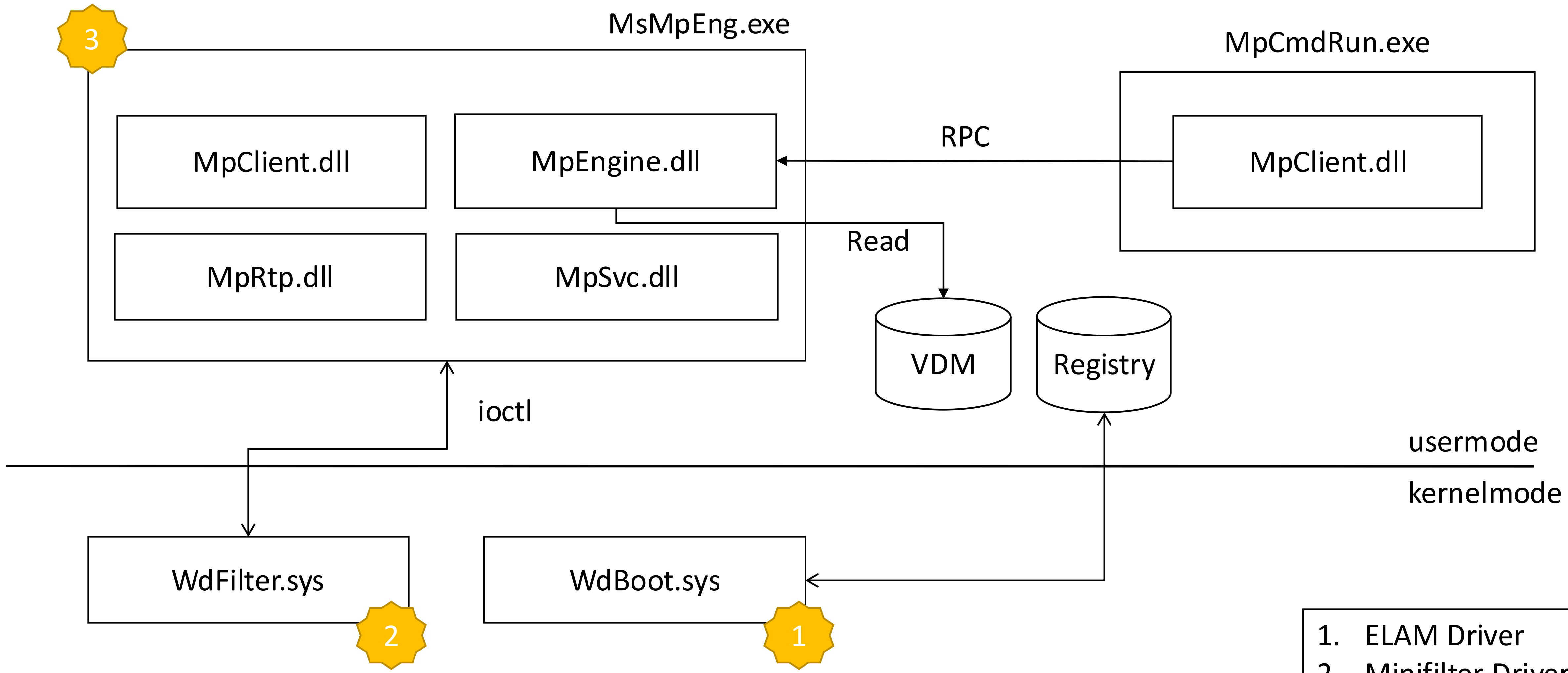
PingPull was written in Visual C++ and provides a threat actor the ability to run commands and access a reverse shell on a compromised host. There are three variants of PingPull that are all functionally the same but use different protocols for communications with their C2: ICMP, HTTP(S) and raw TCP.

Palo Alto, Unit42



Integrating new threats into Retooling Revo

Microsoft Defender Antivirus Architecture



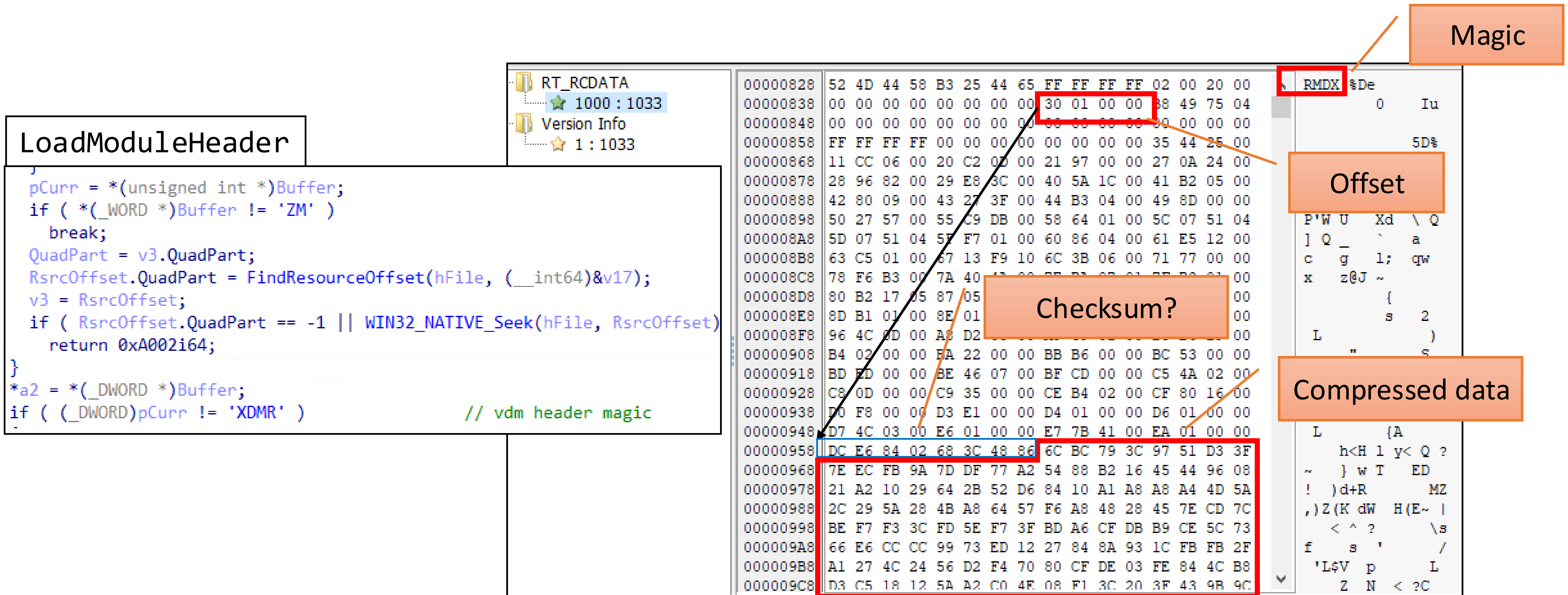
1. ELAM Driver
2. Minifilter Driver
3. PPL

Microsoft Defender's signatures files

- **Located in:** `C:\ProgramData\Microsoft\Windows Defender\Definition Updates\<RandomGUID>\`
- **Portable Executable:**
  - `mpa{s,v}base.vdm` : Updated one per month, contains antimalware/antispyware signatures
  - `mpa{s,v}dlta.vdm`: Updated constantly, contains antimalware/antispyware updates to the base vdms.
- **Focus on** `mpavbase.vdm` **and** `mpasbase.vdm`

mpavbase.vdm and mpasbase.vdm

- ❑ Both contains compressed data (signatures) inside their resource section (`.rsrc`)
- ❑ At boostrap, `mpengine` merges the `*base.vdm` files with the `*delta.vdm` files

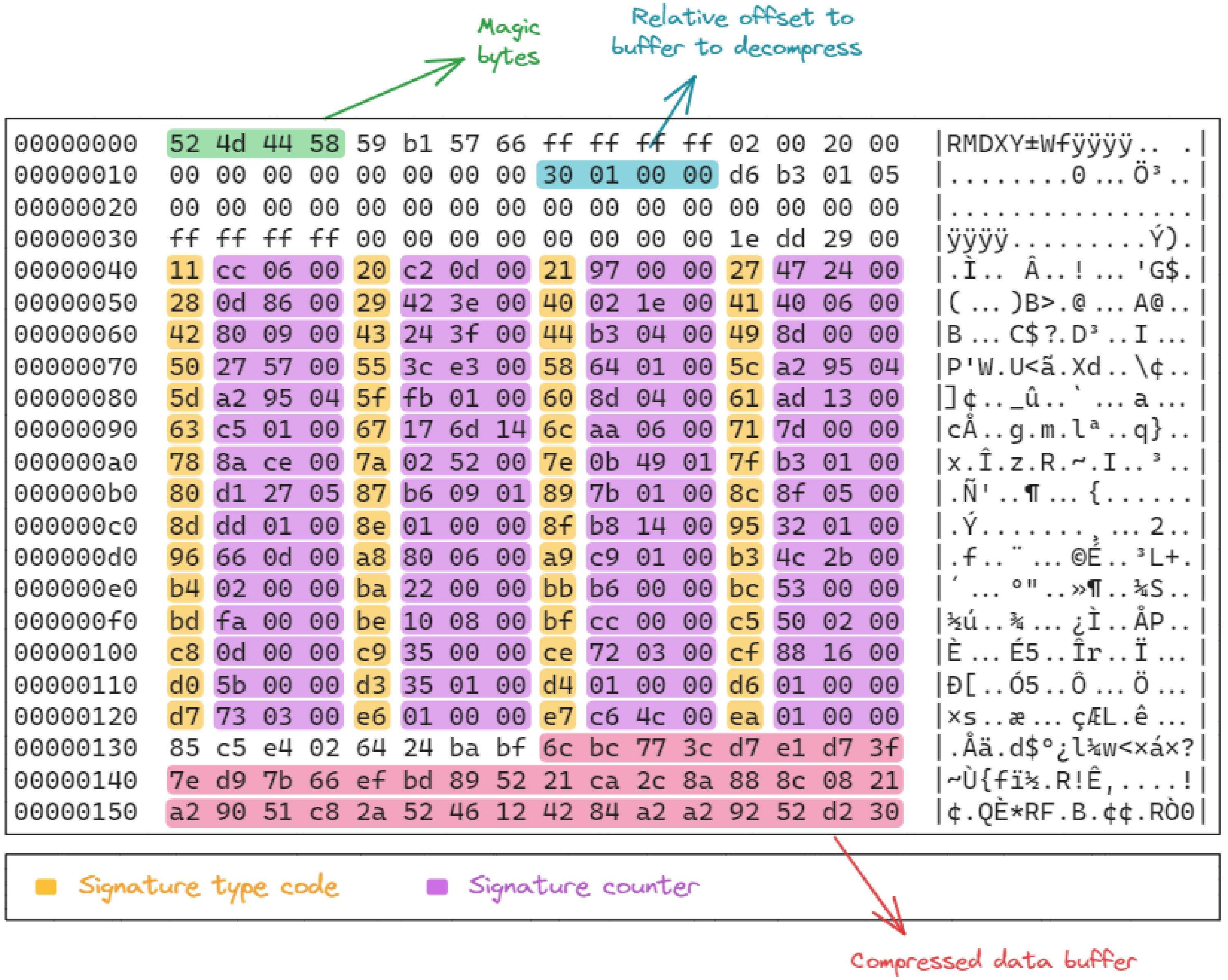


```
  pCurr = *(unsigned int *)Buffer;
  if ( *(_WORD *)Buffer != 'ZM' )
    break;
  QuadPart = v3.QuadPart;
  RsrcOffset.QuadPart = FindResourceOffset(hFile, (__int64)&v17);
  v3 = RsrcOffset;
  if ( RsrcOffset.QuadPart == -1 || WIN32_NATIVE_Seek(hFile, RsrcOffset)
    return 0xA002i64;
}
*a2 = *(_DWORD *)Buffer;
if ( (_DWORD)pCurr != 'XDMR' )          // vdm header magic
```

```
00000828  52 4D 44 58 B3 25 44 65 FF FF FF FF 02 00 20 00   RMDX %De
00000838  00 00 00 00 00 00 00 00 30 01 00 00 38 49 75 04          0    Iu
00000848  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00000858  FF FF FF FF 00 00 00 00 00 00 00 00 35 44 25 00               5D%
00000868  11 CC 06 00 20 C2 0D 00 21 97 00 00 27 0A 24 00
00000878  28 96 82 00 29 E8 3C 00 40 5A 1C 00 41 B2 05 00
00000888  42 80 09 00 43 27 3F 00 44 B3 04 00 49 8D 00 00
00000898  50 27 57 00 55 C9 DB 00 58 64 01 00 5C 07 51 04   P'W U  Xd  \ Q
000008A8  5D 07 51 04 5F F7 01 00 60 86 04 00 61 E5 12 00   ] Q _   `   a
000008B8  63 C5 01 00 67 13 F9 10 6C 3B 06 00 71 77 00 00   c   g   l;  qw
000008C8  78 F6 B3 00 7A 40 [illegible] 00   x   z@J ~
000008D8  80 B2 17 05 87 05 [illegible] 00          {
000008E8  8D B1 01 00 8E 01 [illegible] 00          s   2
000008F8  96 4C 0D 00 A8 D2 [illegible] 00    L          )
00000908  B4 02 00 00 BA 22 00 00 BB B6 00 00 BC 53 00 00        "       S
00000918  BD ED 00 00 BE 46 07 00 BF CD 00 00 C5 4A 02 00
00000928  C8 0D 00 00 C9 35 00 00 CE B4 02 00 CF 80 16 00
00000938  D0 F8 00 00 D3 E1 00 00 D4 01 00 00 D6 01 00 00
00000948  D7 4C 03 00 E6 01 00 00 E7 7B 41 00 EA 01 00 00    L       {A
00000958  DC E6 84 02 68 3C 48 86 6C BC 79 3C 97 51 D3 3F       h<H l y< Q ?
00000968  7E EC FB 9A 7D DF 77 A2 54 88 B2 16 45 44 96 08   ~    } w T   ED
00000978  21 A2 10 29 64 2B 52 D6 84 10 A1 A8 A8 A4 4D 5A   !  )d+R       MZ
00000988  2C 29 5A 28 4B A8 64 57 F6 A8 48 28 45 7E CD 7C   ,)Z(K dW  H(E~ |
00000998  BE F7 F3 3C FD 5E F7 3F BD A6 CF DB B9 CE 5C 73      < ^ ?       \s
000009A8  66 E6 CC CC 99 73 ED 12 27 84 8A 93 1C FB FB 2F   f    s  '      /
000009B8  A1 27 4C 24 56 D2 F4 70 80 CF DE 03 FE 84 4C B8    'L$V  p       L
000009C8  D3 C5 18 12 5A A2 C0 4E 08 F1 3C 20 3F 43 9B 9C      Z   N   < ?C
```

Various types of signatures

```
switch (a1)
    {
...
    case 0x79u:
        return "SIGNATURE_TYPE_VDLL_X86";
    case 0x6Bu:
        return "SIGNATURE_TYPE_WVT_EXCEPTION";
    case 0x6Cu:
        return "SIGNATURE_TYPE_REVOKED_CERTIFICATE";
    case 0x70u:
        return "SIGNATURE_TYPE_TRUSTED_PUBLISHER";
    case 0x71u:
        return "SIGNATURE_TYPE_ASEP_FILEPATH";
    case 0x73u:
        return "SIGNATURE_TYPE_DELTA_BLOB";
    case 0x74u:
        return "SIGNATURE_TYPE_DELTA_BLOB_RECINFO";
    case 0x75u:
        return "SIGNATURE_TYPE_ASEP_FOLDERNAME";
    case 0x77u:
        return "SIGNATURE_TYPE_PATTMATCH_V2";
    case 0x78u:
        return "SIGNATURE_TYPE_PEHSTR_EXT";
...
    }
```



```
00000000  52 4d 44 58 59 b1 57 66 ff ff ff ff 02 00 20 00  |RMDXY±Wfÿÿÿÿ.. .|
00000010  00 00 00 00 00 00 00 00 30 01 00 00 d6 b3 01 05  |........0...Ö³..|
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |................|
00000030  ff ff ff ff 00 00 00 00 00 00 00 00 1e dd 29 00  |ÿÿÿÿ........Ý).|
00000040  11 cc 06 00 20 c2 0d 00 21 97 00 00 27 47 24 00  |.Ì.. Â..!...'G$.|
00000050  28 0d 86 00 29 42 3e 00 40 02 1e 00 41 40 06 00  |(...)B>.@...A@..|
00000060  42 80 09 00 43 24 3f 00 44 b3 04 00 49 8d 00 00  |B...C$?.D³..I...|
00000070  50 27 57 00 55 3c e3 00 58 64 01 00 5c a2 95 04  |P'W.U<ã.Xd..\¢..|
00000080  5d a2 95 04 5f fb 01 00 60 8d 04 00 61 ad 13 00  |]¢.._û..`...a...|
00000090  63 c5 01 00 67 17 6d 14 6c aa 06 00 71 7d 00 00  |cÅ..g.m.lª..q}..|
000000a0  78 8a ce 00 7a 02 52 00 7e 0b 49 01 7f b3 01 00  |x.Î.z.R.~.I..³..|
000000b0  80 d1 27 05 87 b6 09 01 89 7b 01 00 8c 8f 05 00  |.Ñ'..¶...{......|
000000c0  8d dd 01 00 8e 01 00 00 8f b8 14 00 95 32 01 00  |.Ý.......¸...2..|
000000d0  96 66 0d 00 a8 80 06 00 a9 c9 01 00 b3 4c 2b 00  |.f..¨...©É..³L+.|
000000e0  b4 02 00 00 ba 22 00 00 bb b6 00 00 bc 53 00 00  |´...º"..»¶..¼S..|
000000f0  bd fa 00 00 be 10 08 00 bf cc 00 00 c5 50 02 00  |½ú..¾...¿Ì..ÅP..|
00000100  c8 0d 00 00 c9 35 00 00 ce 72 03 00 cf 88 16 00  |È...É5..Îr..Ï...|
00000110  d0 5b 00 00 d3 35 01 00 d4 01 00 00 d6 01 00 00  |Ð[..Ó5..Ô...Ö...|
00000120  d7 73 03 00 e6 01 00 00 e7 c6 4c 00 ea 01 00 00  |×s..æ...çÆL.ê...|
00000130  85 c5 e4 02 64 24 ba bf 6c bc 77 3c d7 e1 d7 3f  |.Åä.d$º¿l¼w<×á×?|
00000140  7e d9 7b 66 ef bd 89 52 21 ca 2c 8a 88 8c 08 21  |~Ù{fï½.R!Ê,....!|
00000150  a2 90 51 c8 2a 52 46 12 42 84 a2 a2 92 52 d2 30  |¢.QÈ*RF.B.¢¢.RÒ0|
```

🟨 Signature type code 🟪 Signature counter

Lab0: Extract Windows Defender's signatures files

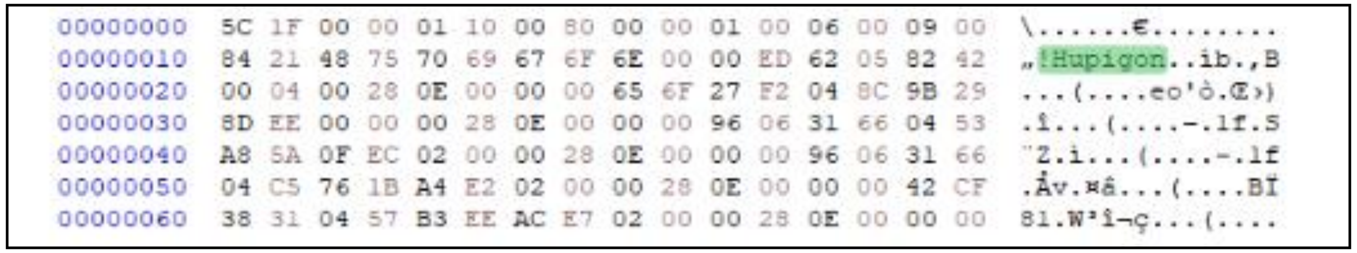
1. Open the folder `C:\ProgramData\Microsoft\Windows Defender\Definition Updates\<Your_GUID_Here>\`
2. Copy the `mpavbase.vdm` on your working folde
3. Cut the file as described to get only the compressed data. Save as `x.gz`
4. Run this `python3` script from the same folder of `x.gz`:

```
import zlib
compressed = open('x.gz', 'rb').read()
decompressed = zlib.decompress(compressed, -zlib.MAX_WBITS)
```

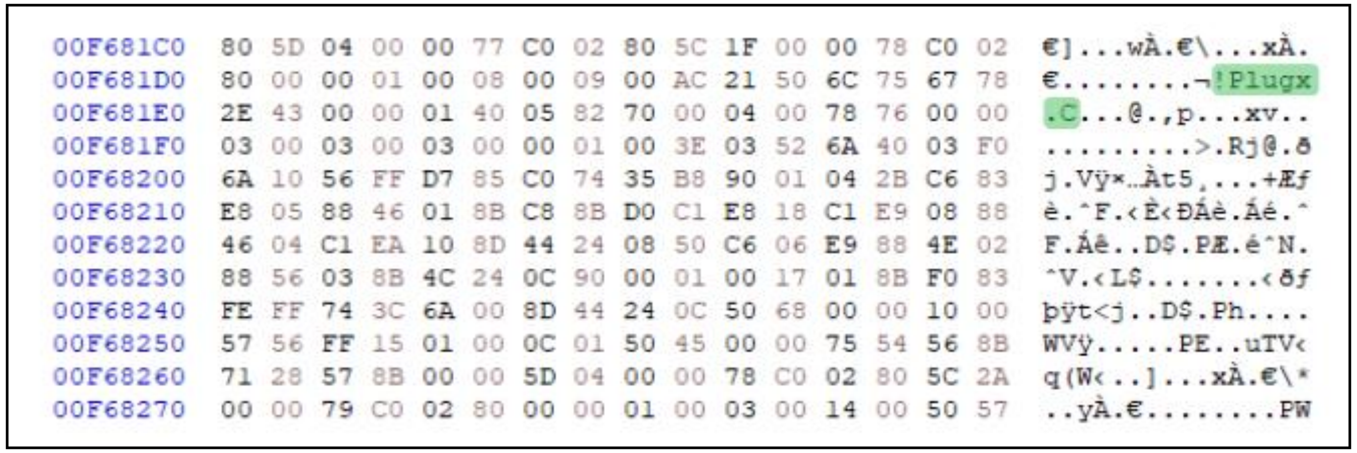
No gz header

Expected output of extracted vdm files

- ❑ Blobs with some ASCII strings referring to threats
  - ❑ !Hupigon
  - ❑ !Plugx.C
  - ❑ …
- ❑ Variable distance among threat names

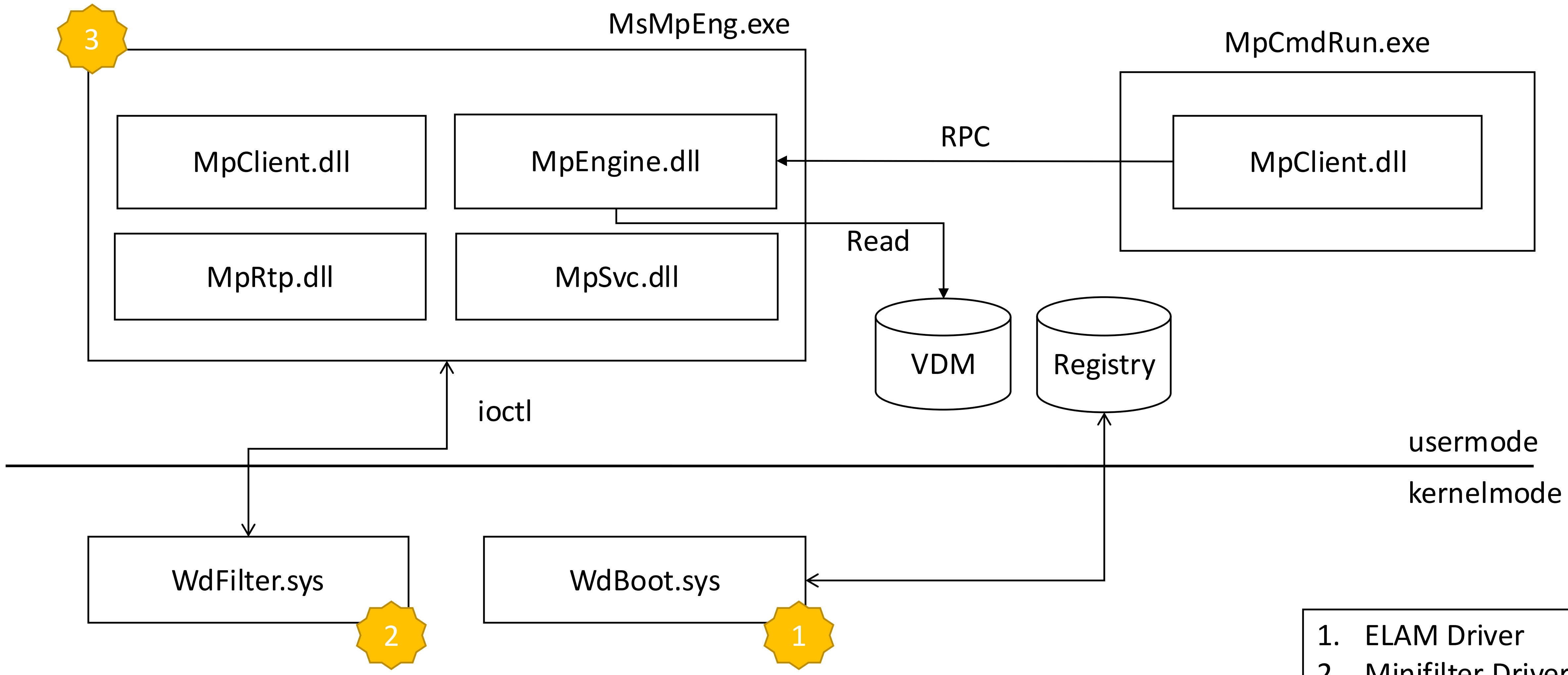


```
00000000  5C 1F 00 00 01 10 00 80 00 00 01 00 06 00 09 00  \......€........
00000010  84 21 48 75 70 69 67 6F 6E 00 00 ED 62 05 82 42  „!Hupigon..íb.‚B
00000020  00 04 00 28 0E 00 00 00 65 6F 27 F2 04 8C 9B 29  ...(....eo'ò.Œ›)
00000030  8D EE 00 00 00 28 0E 00 00 00 96 06 31 66 04 53  .î...(....–.1f.S
00000040  A8 5A 0F EC 02 00 00 28 0E 00 00 00 96 06 31 66  ¨Z.ì...(....–.1f
00000050  04 C5 76 1B A4 E2 02 00 00 28 0E 00 00 00 42 CF  .Åv.¤â...(....BÏ
00000060  38 31 04 57 B3 EE AC E7 02 00 00 28 0E 00 00 00  81.W³î¬ç...(....
```



```
00F681C0  80 5D 04 00 00 77 C0 02 80 5C 1F 00 00 78 C0 02  €]...wÀ.€\...xÀ.
00F681D0  80 00 00 01 00 08 00 09 00 AC 21 50 6C 75 67 78  €........¬!Plugx
00F681E0  2E 43 00 00 01 40 05 82 70 00 04 00 78 76 00 00  .C...@.‚p...xv..
00F681F0  03 00 03 00 03 00 00 01 00 3E 03 52 6A 40 03 F0  .........>.Rj@.ð
00F68200  6A 10 56 FF D7 85 C0 74 35 B8 90 01 04 2B C6 83  j.Vÿ×…Àt5¸....+Æƒ
00F68210  E8 05 88 46 01 8B C8 8B D0 C1 E8 18 C1 E9 08 88  è.ˆF.‹È‹ÐÁè.Áé.ˆ
00F68220  46 04 C1 EA 10 8D 44 24 08 50 C6 06 E9 88 4E 02  F.Áê..D$.PÆ.éˆN.
00F68230  88 56 03 8B 4C 24 0C 90 00 01 00 17 01 8B F0 83  ˆV.‹L$.......‹ðƒ
00F68240  FE FF 74 3C 6A 00 8D 44 24 0C 50 68 00 00 10 00  þÿt<j..D$.Ph....
00F68250  57 56 FF 15 01 00 0C 01 50 45 00 00 75 54 56 8B  WVÿ.....PE..uTV‹
00F68260  71 28 57 8B 00 00 5D 04 00 00 78 C0 02 80 5C 2A  q(W‹..]...xÀ.€\*
00F68270  00 00 79 C0 02 80 00 00 01 00 03 00 14 00 50 57  ..yÀ.€........PW
```

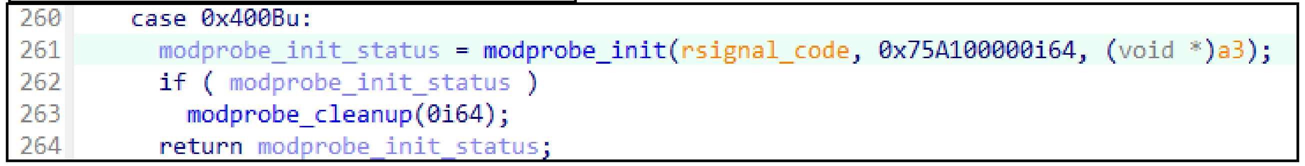
Microsoft Defender Antivirus Architecture



1. ELAM Driver
2. Minifilter Driver
3. PPL

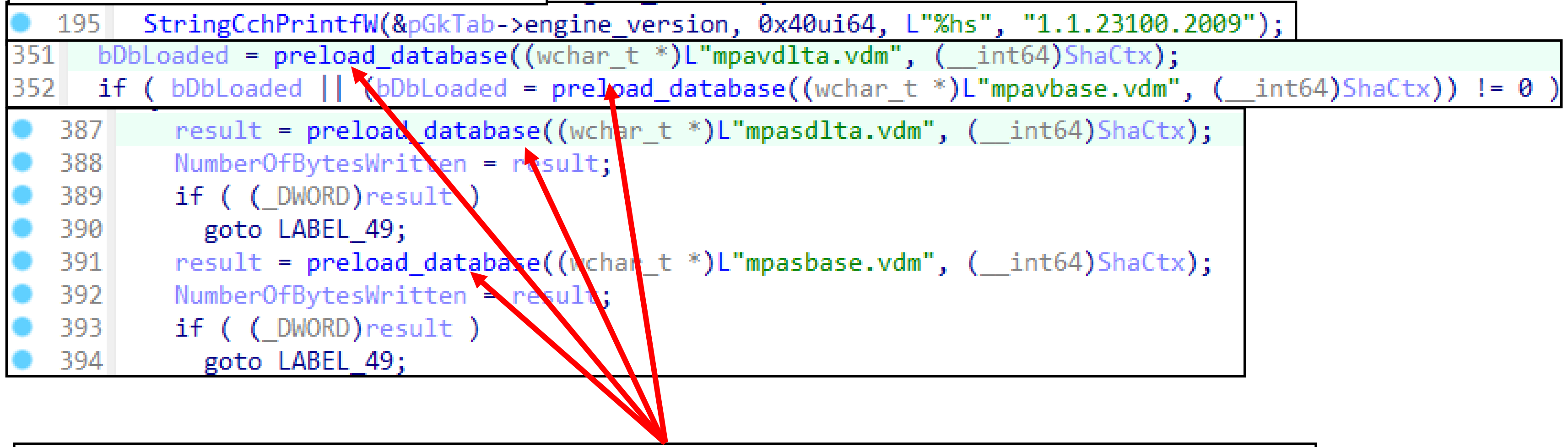
Phase1: Signatures Database preload

ksignal



```
    case 0x400Bu:
      modprobe_init_status = modprobe_init(rsignal_code, 0x75A100000i64, (void *)a3);
      if ( modprobe_init_status )
        modprobe_cleanup(0i64);
      return modprobe_init_status;
```

modprobe_init_worker



```
  StringCchPrintfW(&pGkTab->engine_version, 0x40ui64, L"%hs", "1.1.23100.2009");
```

```
  bDbLoaded = preload_database((wchar_t *)L"mpavdlta.vdm", (__int64)ShaCtx);
  if ( bDbLoaded || (bDbLoaded = preload_database((wchar_t *)L"mpavbase.vdm", (__int64)ShaCtx)) != 0 )
```

```
    result = preload_database((wchar_t *)L"mpasdlta.vdm", (__int64)ShaCtx);
    NumberOfBytesWritten = result;
    if ( (_DWORD)result )
      goto LABEL_49;
    result = preload_database((wchar_t *)L"mpasbase.vdm", (__int64)ShaCtx);
    NumberOfBytesWritten = result;
    if ( (_DWORD)result )
      goto LABEL_49;
```

It reads the header and retrieve general information s.a. signature versions and numbers
`LoadModuleHeader` : loads the database header (the first 16 bytes)

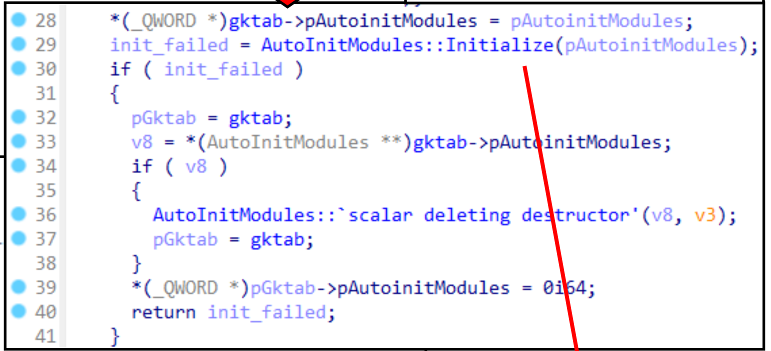
Signature version 1.401.1166.0

Once the pre-processing of signature file completes, the defender modules initialization begins...

Phase 2 Initialization of Defender modules

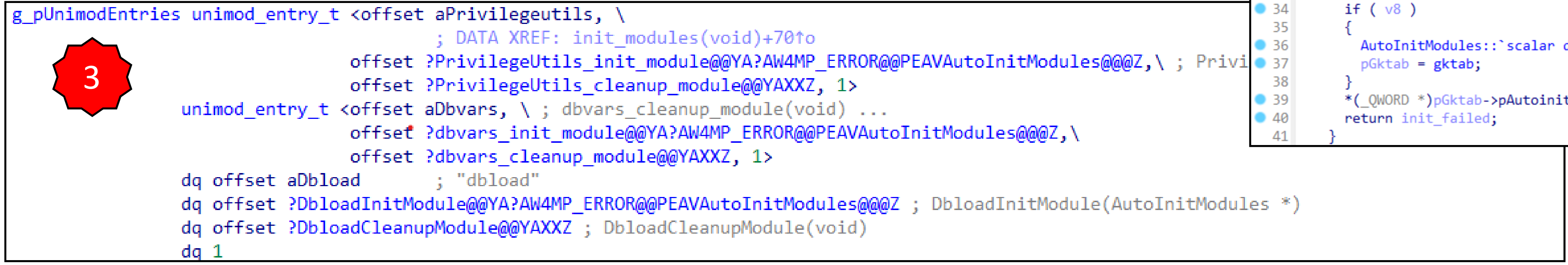
1

init_modules



```
*(_QWORD *)gktab->pAutoinitModules = pAutoinitModules;
init_failed = AutoInitModules::Initialize(pAutoinitModules);
if ( init_failed )
{
  pGktab = gktab;
  v8 = *(AutoInitModules **)gktab->pAutoinitModules;
  if ( v8 )
  {
    AutoInitModules::`scalar deleting destructor'(v8, v3);
    pGktab = gktab;
  }
  *(_QWORD *)pGktab->pAutoinitModules = 0i64;
  return init_failed;
}
```

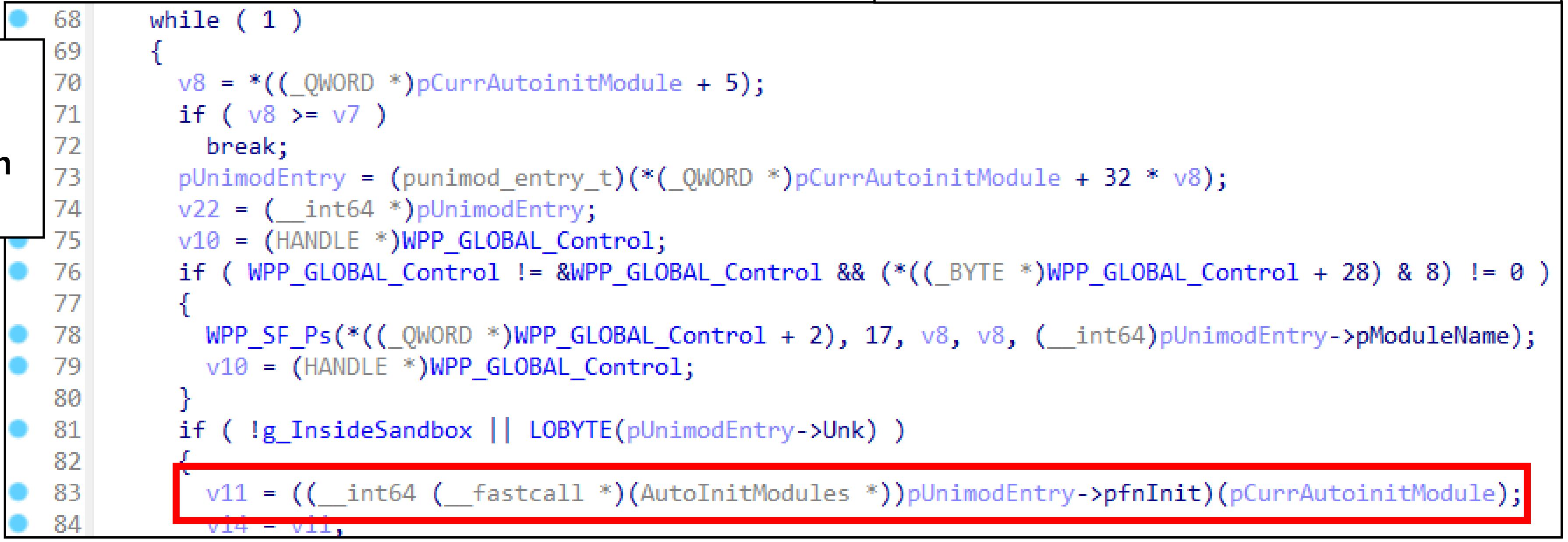
3



```
g_pUnimodEntries unimod_entry_t <offset aPrivilegeutils, \
                                        ; DATA XREF: init_modules(void)+70↑o
                        offset ?PrivilegeUtils_init_module@@YA?AW4MP_ERROR@@PEAVAutoInitModules@@@Z,\ ; Privi
                        offset ?PrivilegeUtils_cleanup_module@@YAXXZ, 1>
                 unimod_entry_t <offset aDbvars, \ ; dbvars_cleanup_module(void) ...
                        offset ?dbvars_init_module@@YA?AW4MP_ERROR@@PEAVAutoInitModules@@@Z,\
                        offset ?dbvars_cleanup_module@@YAXXZ, 1>
                 dq offset aDbload       ; "dbload"
                 dq offset ?DbloadInitModule@@YA?AW4MP_ERROR@@PEAVAutoInitModules@@@Z ; DbloadInitModule(AutoInitModules *)
                 dq offset ?DbloadCleanupModule@@YAXXZ ; DbloadCleanupModule(void)
                 dq 1
```

2

AutoInitModules::Initialize



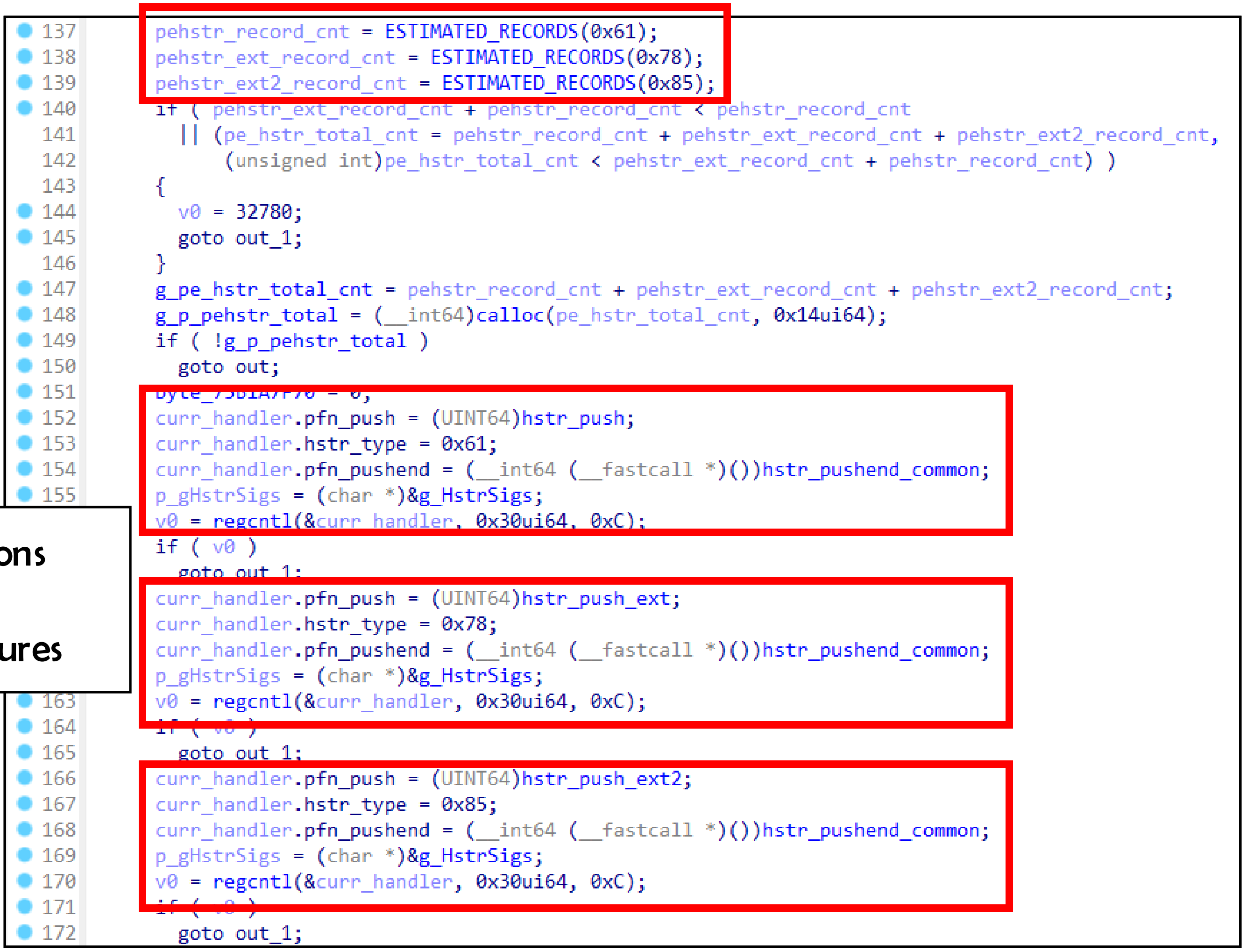
```
while ( 1 )
{
  v8 = *((_QWORD *)pCurrAutoinitModule + 5);
  if ( v8 >= v7 )
    break;
  pUnimodEntry = (punimod_entry_t)(*(_QWORD *)pCurrAutoinitModule + 32 * v8);
  v22 = (__int64 *)pUnimodEntry;
  v10 = (HANDLE *)WPP_GLOBAL_Control;
  if ( WPP_GLOBAL_Control != &WPP_GLOBAL_Control && (*((_BYTE *)WPP_GLOBAL_Control + 28) & 8) != 0 )
  {
    WPP_SF_Ps(*((_QWORD *)WPP_GLOBAL_Control + 2), 17, v8, v8, (__int64)pUnimodEntry->pModuleName);
    v10 = (HANDLE *)WPP_GLOBAL_Control;
  }
  if ( !g_InsideSandbox || LOBYTE(pUnimodEntry->Unk) )
  {
    v11 = ((__int64 (__fastcall *)(AutoInitModules *))pUnimodEntry->pfnInit)(pCurrAutoinitModule);
    v14 = v11;
```

Loop over all the module in
`g_pUnimodEntries`
And call the module-specific init function
`pfnInit()`

cksig_init_module

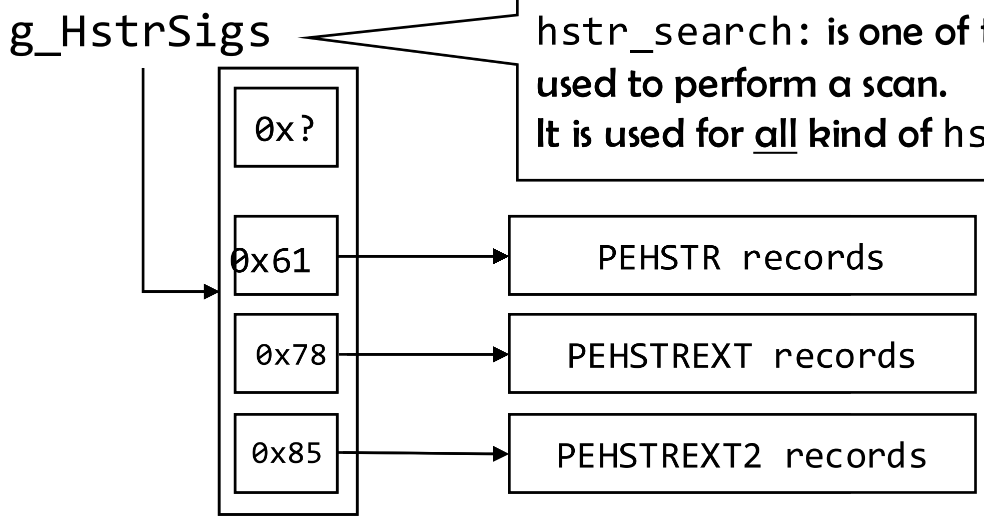
- Invokes the `pattsearch_init` function initializes the data structures that will contain the signatures: namely `g_HstrSigs` and `g_DynamicHstrSigs`
- Those symbols are pointers to an hashtable which contains all the HSTR signatures (elf, pe, macho, ...)
- The `load_database/load_database_cache` will `DispatchRecords` to the right bucket

`pattsearch_init`

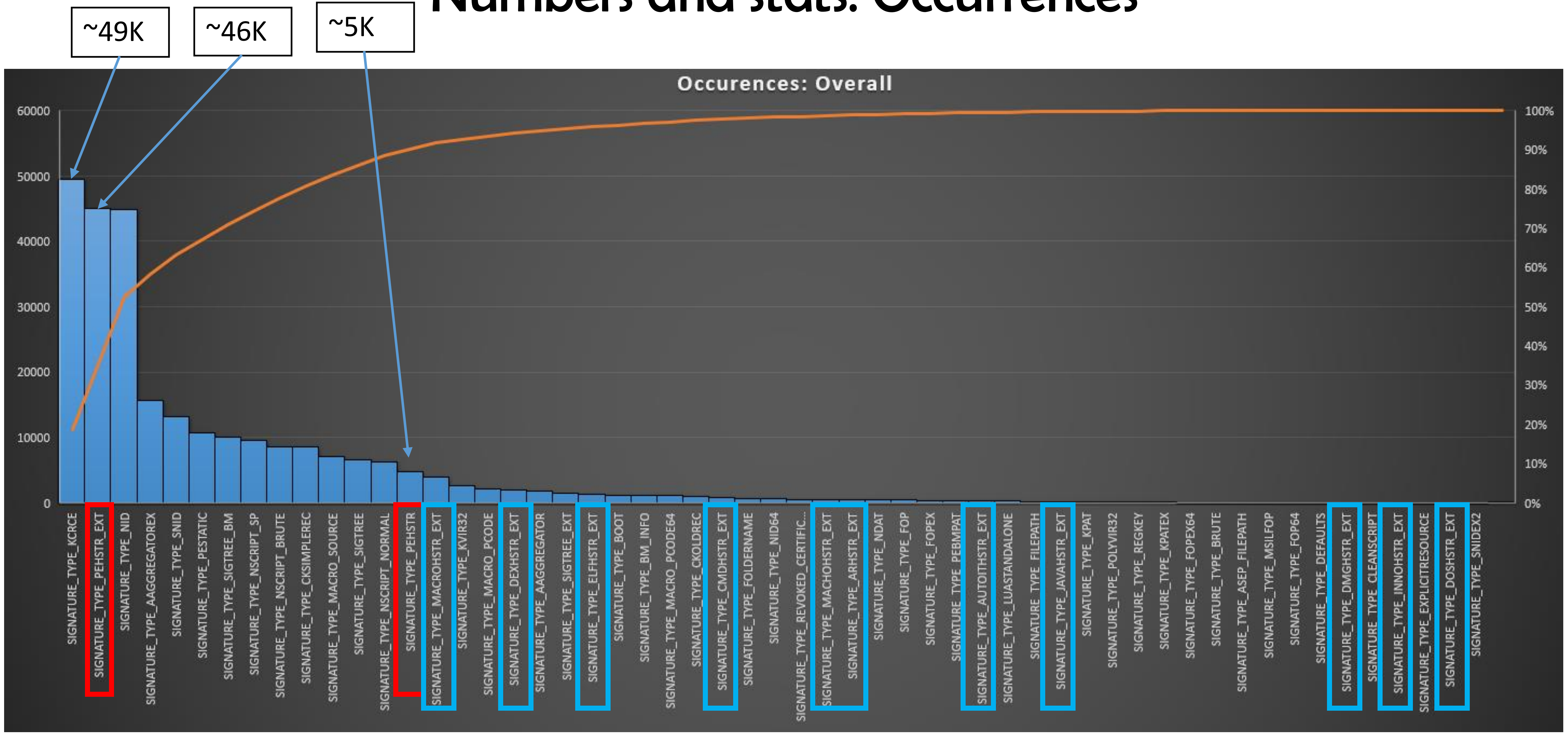


```
pehstr_record_cnt = ESTIMATED_RECORDS(0x61);
pehstr_ext_record_cnt = ESTIMATED_RECORDS(0x78);
pehstr_ext2_record_cnt = ESTIMATED_RECORDS(0x85);
if ( pehstr_ext_record_cnt + pehstr_record_cnt < pehstr_record_cnt
  || (pe_hstr_total_cnt = pehstr_record_cnt + pehstr_ext_record_cnt + pehstr_ext2_record_cnt,
     (unsigned int)pe_hstr_total_cnt < pehstr_ext_record_cnt + pehstr_record_cnt) )
{
  v0 = 32780;
  goto out_1;
}
g_pe_hstr_total_cnt = pehstr_record_cnt + pehstr_ext_record_cnt + pehstr_ext2_record_cnt;
g_p_pehstr_total = (__int64)calloc(pe_hstr_total_cnt, 0x14ui64);
if ( !g_p_pehstr_total )
  goto out;
byte_73B1A7F70 = 0;
curr_handler.pfn_push = (UINT64)hstr_push;
curr_handler.hstr_type = 0x61;
curr_handler.pfn_pushend = (__int64 (__fastcall *)())hstr_pushend_common;
p_gHstrSigs = (char *)&g_HstrSigs;
v0 = regcntl(&curr_handler, 0x30ui64, 0xC);
if ( v0 )
  goto out_1;
curr_handler.pfn_push = (UINT64)hstr_push_ext;
curr_handler.hstr_type = 0x78;
curr_handler.pfn_pushend = (__int64 (__fastcall *)())hstr_pushend_common;
p_gHstrSigs = (char *)&g_HstrSigs;
v0 = regcntl(&curr_handler, 0x30ui64, 0xC);
if ( v0 )
  goto out_1;
curr_handler.pfn_push = (UINT64)hstr_push_ext2;
curr_handler.hstr_type = 0x85;
curr_handler.pfn_pushend = (__int64 (__fastcall *)())hstr_pushend_common;
p_gHstrSigs = (char *)&g_HstrSigs;
v0 = regcntl(&curr_handler, 0x30ui64, 0xC);
if ( v0 )
  goto out_1;
```

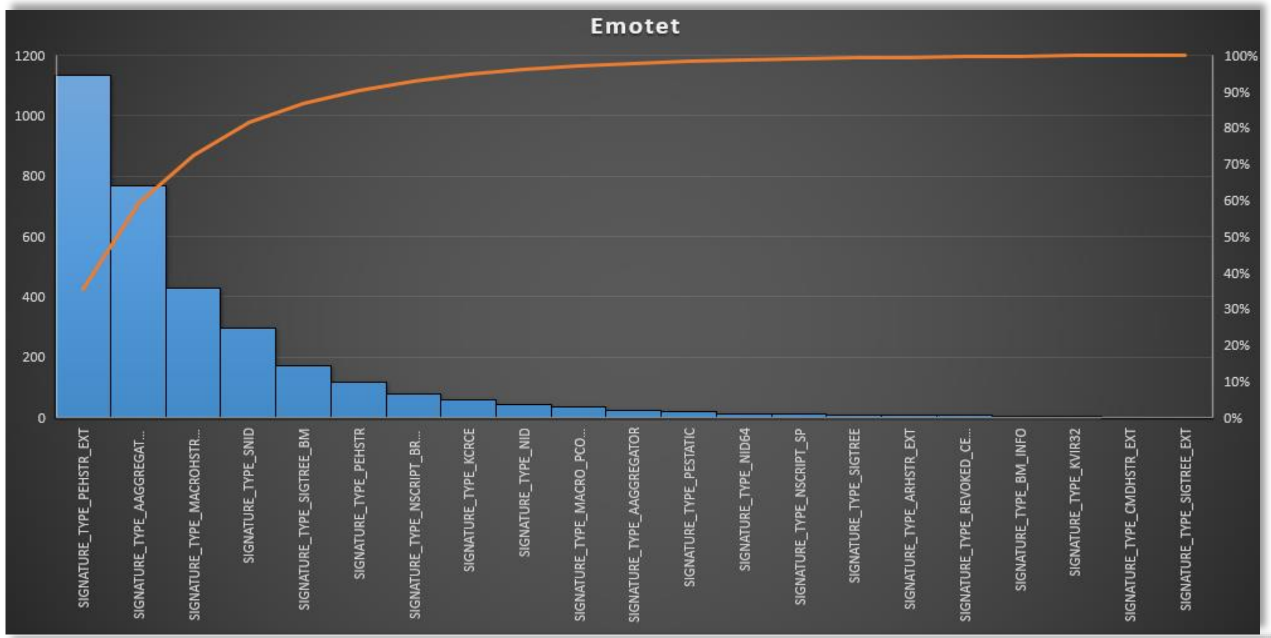
`hstr_search`: is one of the functions used to perform a scan.
It is used for <u>all</u> kind of `hstr` signatures

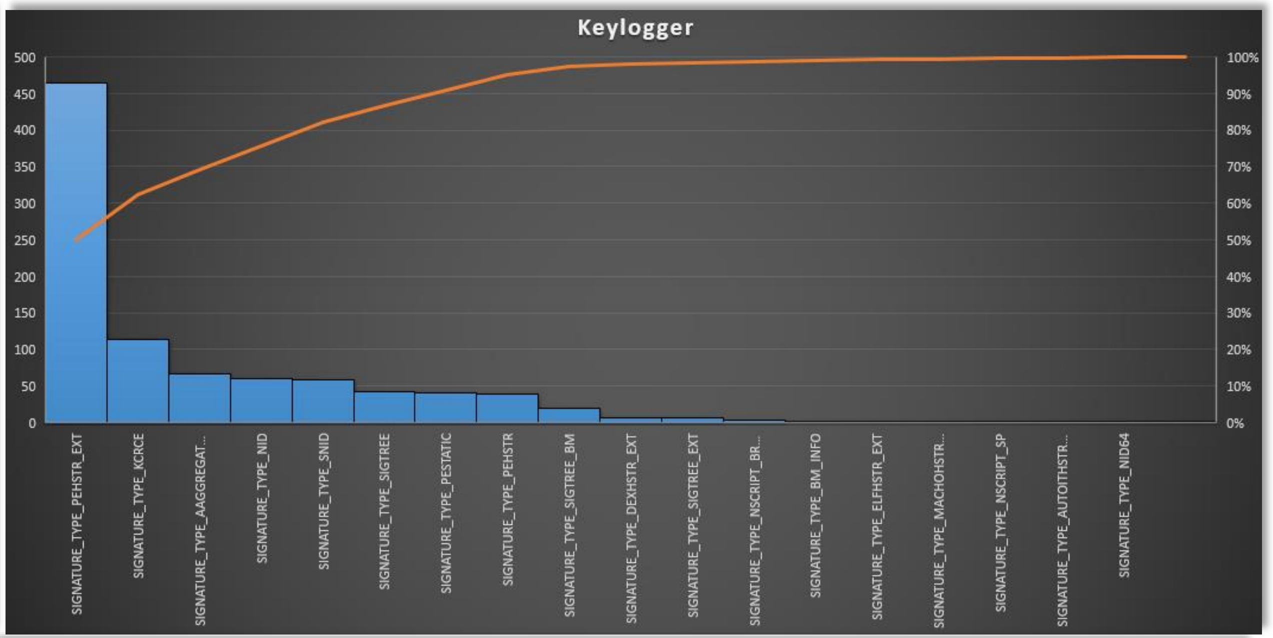


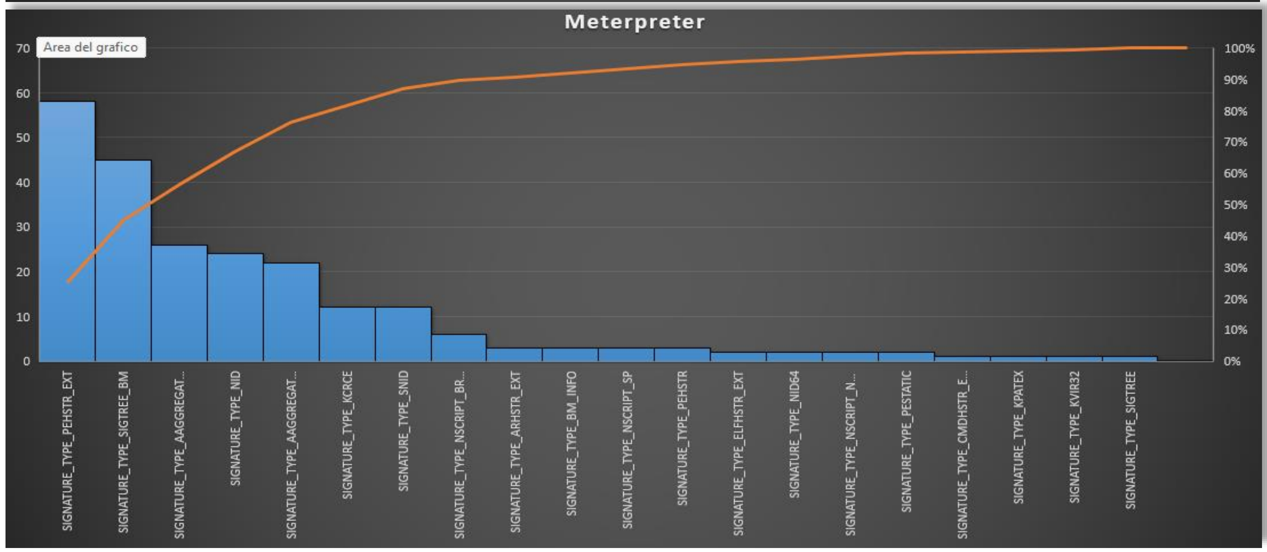
Numbers and stats: Occurrences

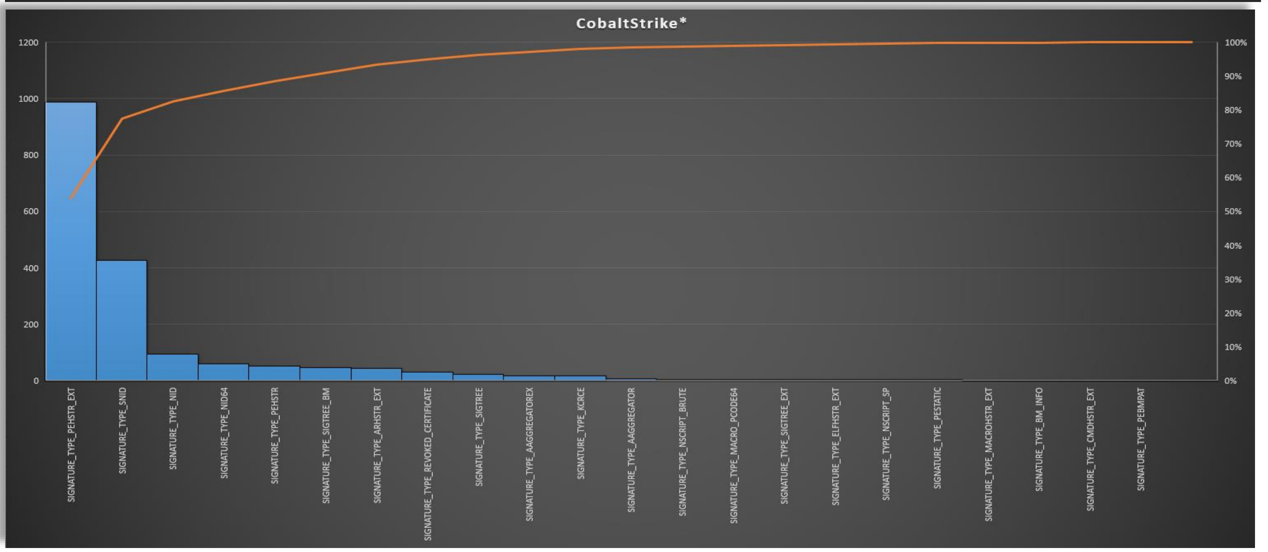


Slicing on specific threats

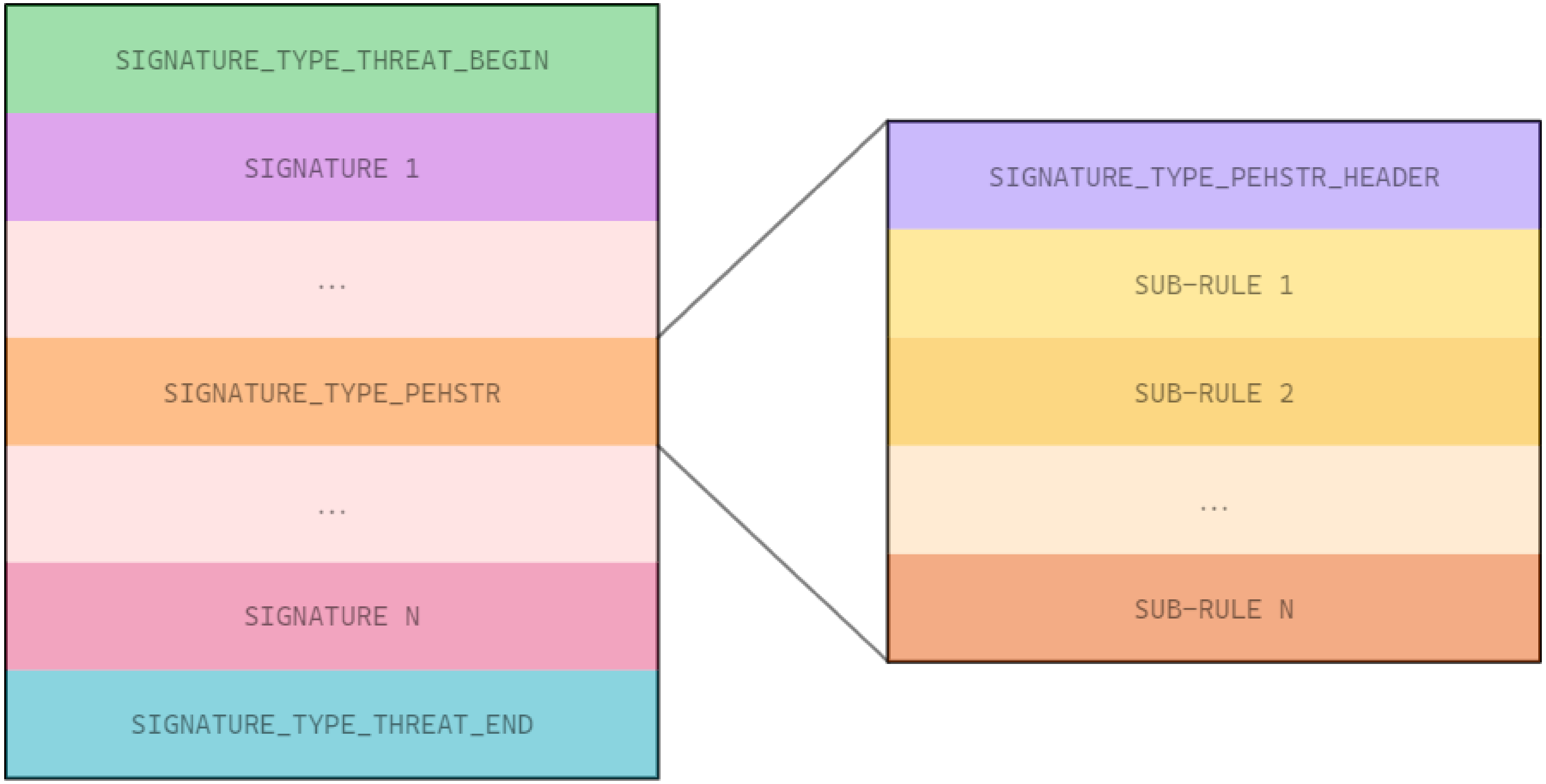








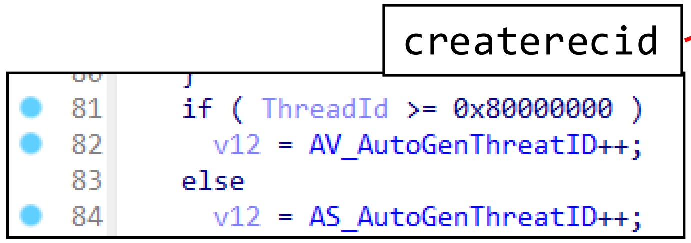
General structure of signatures



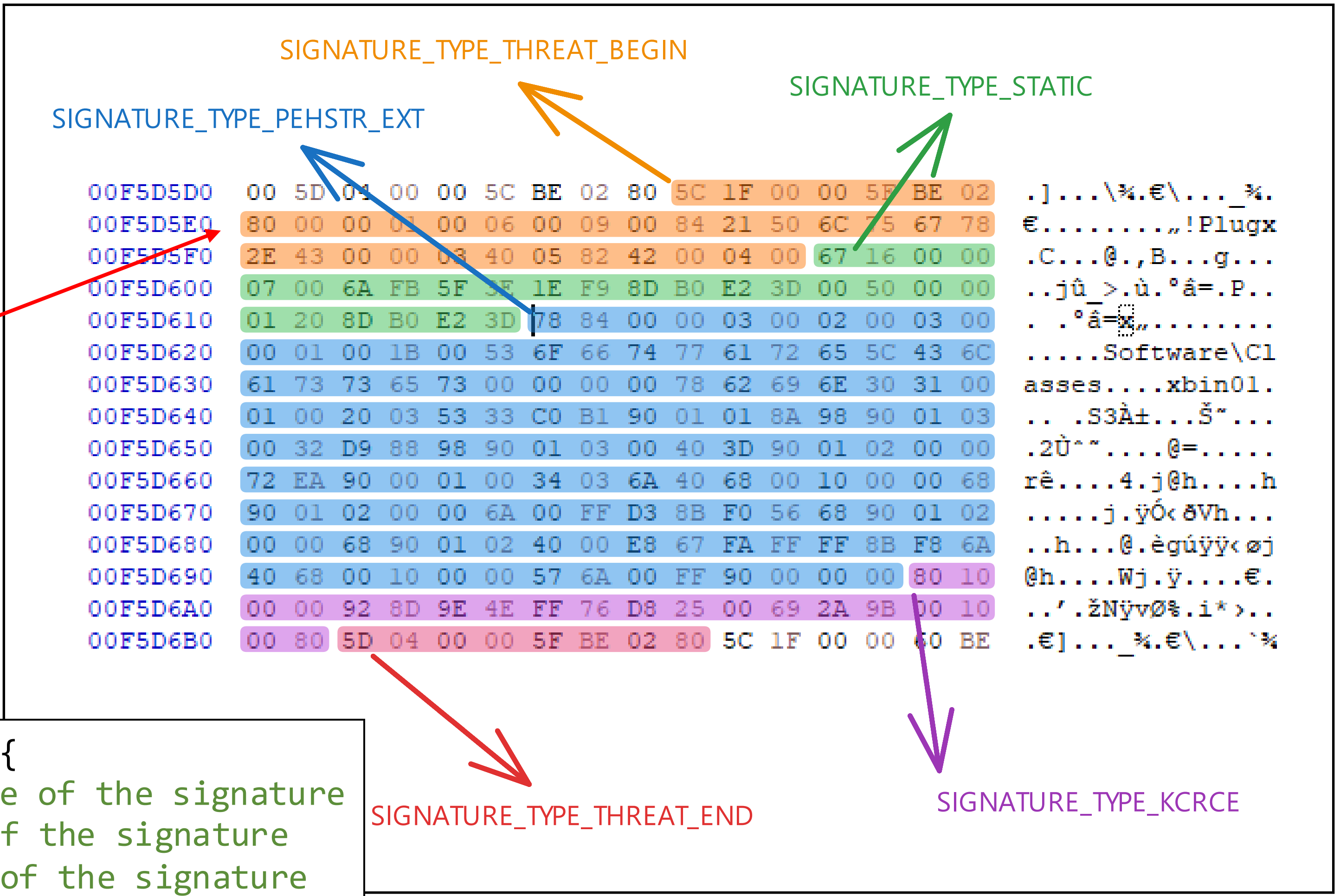
BEGIN 2 END

SIGNATURE_TYPE_THREAT_BEGIN and SIGNATURE_TYPE_THREAT_END have custom data inside them

- One of them is the 4 bytes rule id (e.g. 0x8002be5f)



```
    if ( ThreadId >= 0x80000000 )
      v12 = AV_AutoGenThreatID++;
    else
      v12 = AS_AutoGenThreatID++;
```

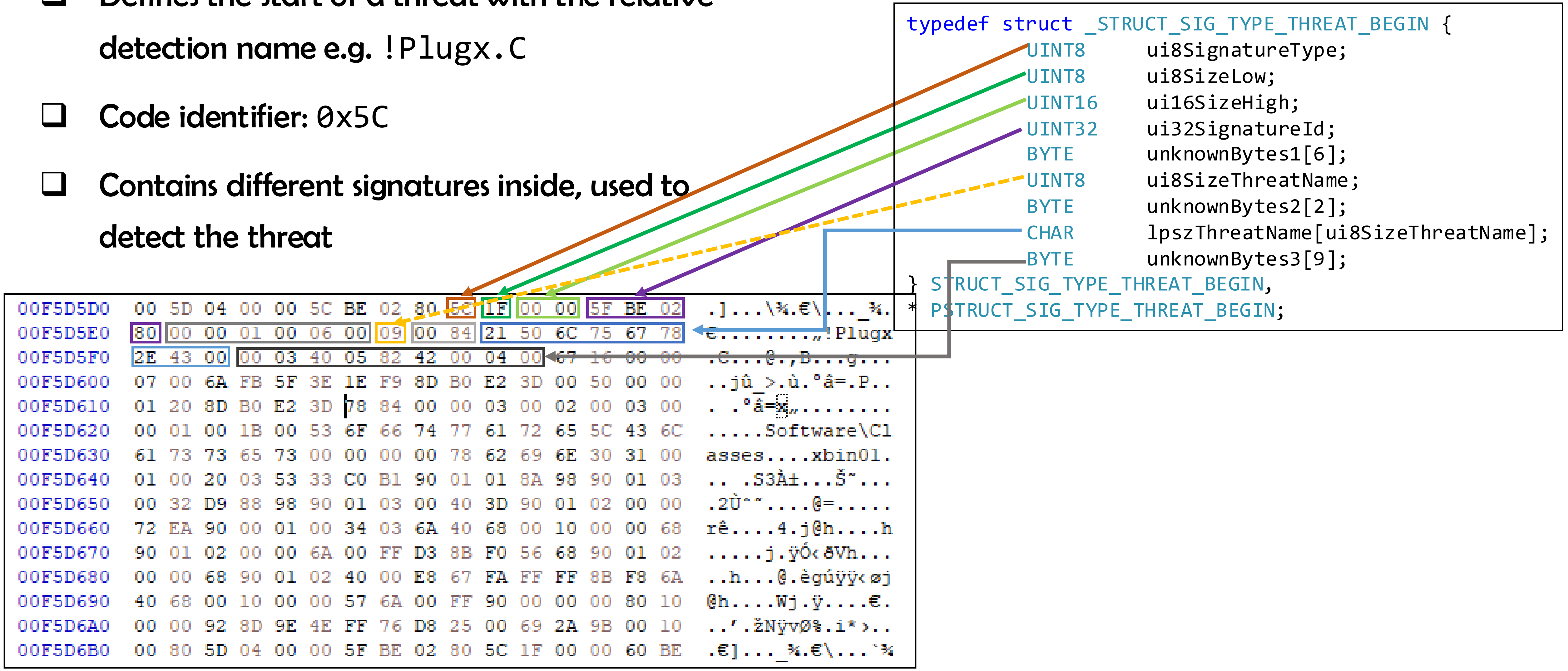

SIGNATURE_TYPE_THREAT_BEGIN
SIGNATURE_TYPE_STATIC
SIGNATURE_TYPE_PEHSTR_EXT
SIGNATURE_TYPE_THREAT_END
SIGNATURE_TYPE_KCRCE

```
00F5D5D0  00 5D 04 00 00 5C BE 02 80 5C 1F 00 00 5F BE 02  .]...\¾.€\..._¾.
00F5D5E0  80 00 00 01 00 06 00 09 00 84 21 50 6C 75 67 78  €........„!Plugx
00F5D5F0  2E 43 00 00 03 40 05 82 42 00 04 00 67 16 00 00  .C...@.‚B...g...
00F5D600  07 00 6A FB 5F 3E 1E F9 8D B0 E2 3D 00 50 00 00  ..jû_>.ù.°â=.P..
00F5D610  01 20 8D B0 E2 3D 78 84 00 00 03 00 02 00 03 00  . .°â=x„........
00F5D620  00 01 00 1B 00 53 6F 66 74 77 61 72 65 5C 43 6C  .....Software\Cl
00F5D630  61 73 73 65 73 00 00 00 00 78 62 69 6E 30 31 00  asses....xbin01.
00F5D640  01 00 20 03 53 33 C0 B1 90 01 01 8A 98 90 01 03  .. .S3À±...Š˜...
00F5D650  00 32 D9 88 98 90 01 03 00 40 3D 90 01 02 00 00  .2Ùˆ˜....@=.....
00F5D660  72 EA 90 00 01 00 34 03 6A 40 68 00 10 00 00 68  rê....4.j@h....h
00F5D670  90 01 02 00 00 6A 00 FF D3 8B F0 56 68 90 01 02  .....j.ÿÓ‹ðVh...
00F5D680  00 00 68 90 01 02 40 00 E8 67 FA FF FF 8B F8 6A  ..h...@.ègúÿÿ‹øj
00F5D690  40 68 00 10 00 00 57 6A 00 FF 90 00 00 00 80 10  @h....Wj.ÿ....€.
00F5D6A0  00 00 92 8D 9E 4E FF 76 D8 25 00 69 2A 9B 00 10  ..’.žNÿvØ%.i*›..
00F5D6B0  00 80 5D 04 00 00 5F BE 02 80 5C 1F 00 00 60 BE  .€]..._¾.€\...`¾
```

```
typedef struct _STRUCT_COMMON_SIGNATURE_TYPE {
    UINT8  ui8SignatureType; // defines the type of the signature
    UINT8  ui8SizeLow;       // low byte size of the signature
    UINT16 ui16SizeHigh;     // high byte size of the signature
    BYTE   pbRuleContent[];  // content of the rule
};
```

SIGNATURE_TYPE_THREAT_BEGIN

- ❑ Defines the start of a threat with the relative detection name e.g. `!Plugx.C`
- ❑ Code identifier: `0x5C`
- ❑ Contains different signatures inside, used to detect the threat



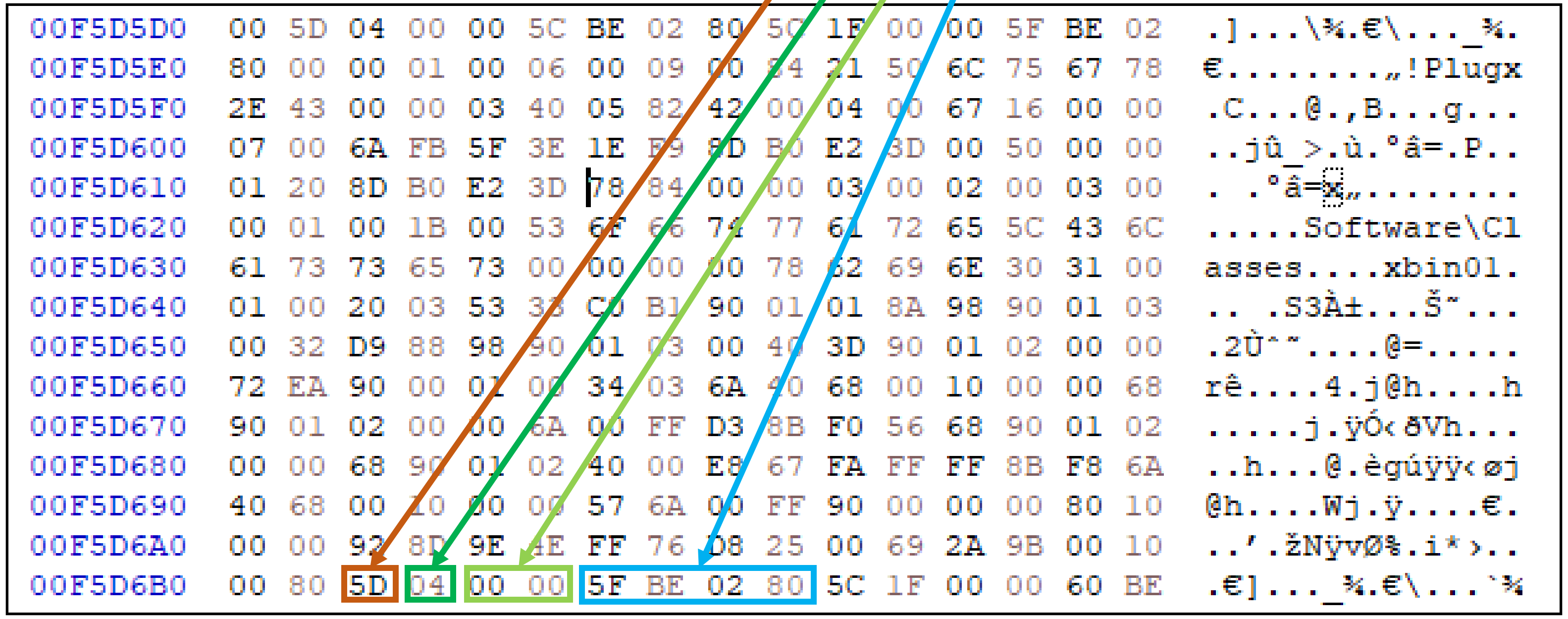
```
typedef struct _STRUCT_SIG_TYPE_THREAT_BEGIN {
    UINT8     ui8SignatureType;
    UINT8     ui8SizeLow;
    UINT16    ui16SizeHigh;
    UINT32    ui32SignatureId;
    BYTE      unknownBytes1[6];
    UINT8     ui8SizeThreatName;
    BYTE      unknownBytes2[2];
    CHAR      lpszThreatName[ui8SizeThreatName];
    BYTE      unknownBytes3[9];
} STRUCT_SIG_TYPE_THREAT_BEGIN,
* PSTRUCT_SIG_TYPE_THREAT_BEGIN;
```

```
00F5D5D0  00 5D 04 00 00 5C BE 02 80 5C 1F 00 00 5F BE 02  .]...\¾.€\..._¾.
00F5D5E0  80 00 00 01 00 06 00 09 00 84 21 50 6C 75 67 78  €.........„!Plugx
00F5D5F0  2E 43 00 00 03 40 05 82 42 00 04 00 67 16 00 00  .C...@.‚B...g...
00F5D600  07 00 6A FB 5F 3E 1E F9 8D B0 E2 3D 00 50 00 00  ..jû_>.ù.°â=.P..
00F5D610  01 20 8D B0 E2 3D 78 84 00 00 03 00 02 00 03 00  . .°â=x„........
00F5D620  00 01 00 1B 00 53 6F 66 74 77 61 72 65 5C 43 6C  .....Software\Cl
00F5D630  61 73 73 65 73 00 00 00 00 78 62 69 6E 30 31 00  asses....xbin01.
00F5D640  01 00 20 03 53 33 C0 B1 90 01 01 8A 98 90 01 03  .. .S3À±..Š˜...
00F5D650  00 32 D9 88 98 90 01 03 00 40 3D 90 01 02 00 00  .2Ùˆ˜....@=.....
00F5D660  72 EA 90 00 01 00 34 03 6A 40 68 00 10 00 00 68  rê....4.j@h....h
00F5D670  90 01 02 00 00 6A 00 FF D3 8B F0 56 68 90 01 02  .....j.ÿÓ‹ðVh...
00F5D680  00 00 68 90 01 02 40 00 E8 67 FA FF FF 8B F8 6A  ..h...@.ègúÿÿ‹øj
00F5D690  40 68 00 10 00 00 57 6A 00 FF 90 00 00 00 80 10  @h....Wj.ÿ....€.
00F5D6A0  00 00 92 8D 9E 4E FF 76 D8 25 00 69 2A 9B 00 10  ..'.žNÿvØ%.i*›..
00F5D6B0  00 80 5D 04 00 00 5F BE 02 80 5C 1F 00 00 60 BE  .€]..._¾.€\...`¾
```

SIGNATURE_TYPE_THREAT_END

- Defines the end of a threat
- Code identifier: `0x5D`
- `pbRuleContent` value is the same as the corresponding ui32SignatureId used in the SIGNATURE_TYPE_THREAT_BEGIN

```
typedef struct _STRUCT_SIG_TYPE_THREAT_END
 {
        UINT8  ui8SignatureType;
        UINT8  ui8SizeLow;
        UINT16 ui16SizeHigh;
        BYTE   pbRuleContent[];
} STRUCT_SIG_TYPE_THREAT_END,
* PSTRUCT_SIG_TYPE_THREAT_END;
```

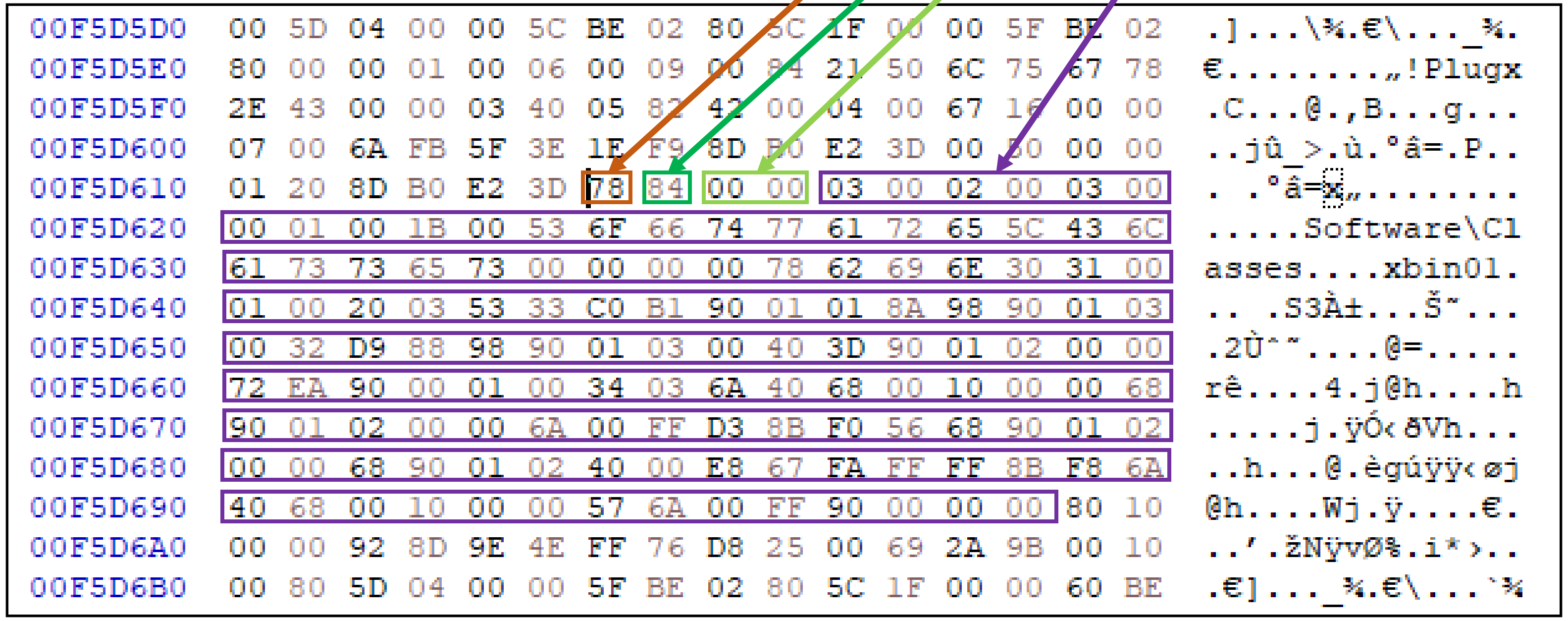


```
00F5D5D0  00 5D 04 00 00 5C BE 02 80 5C 1F 00 00 5F BE 02  .]...\¾.€\..._¾.
00F5D5E0  80 00 00 01 00 06 00 09 00 84 21 50 6C 75 67 78  €........„!Plugx
00F5D5F0  2E 43 00 00 03 40 05 82 42 00 04 00 67 16 00 00  .C...@.‚B...g...
00F5D600  07 00 6A FB 5F 3E 1E F9 8D B0 E2 3D 00 50 00 00  ..jû_>.ù.°â=.P..
00F5D610  01 20 8D B0 E2 3D 78 84 00 00 03 00 02 00 03 00  . .°â=x„........
00F5D620  00 01 00 1B 00 53 6F 66 74 77 61 72 65 5C 43 6C  .....Software\Cl
00F5D630  61 73 73 65 73 00 00 00 00 78 62 69 6E 30 31 00  asses....xbin01.
00F5D640  01 00 20 03 53 33 C0 B1 90 01 01 8A 98 90 01 03  .. .S3À±..Š˜...
00F5D650  00 32 D9 88 98 90 01 03 00 40 3D 90 01 02 00 00  .2Ùˆ˜....@=.....
00F5D660  72 EA 90 00 01 00 34 03 6A 40 68 00 10 00 00 68  rê....4.j@h....h
00F5D670  90 01 02 00 00 6A 00 FF D3 8B F0 56 68 90 01 02  .....j.ÿÓ‹ðVh...
00F5D680  00 00 68 90 01 02 40 00 E8 67 FA FF FF 8B F8 6A  ..h...@.ègúÿÿ‹øj
00F5D690  40 68 00 10 00 00 57 6A 00 FF 90 00 00 00 80 10  @h....Wj.ÿ....€.
00F5D6A0  00 00 92 8D 9E 4E FF 76 D8 25 00 69 2A 9B 00 10  ..'.žNÿvØ%.i*›..
00F5D6B0  00 80 5D 04 00 00 5F BE 02 80 5C 1F 00 00 60 BE  .€]..._¾.€\...`¾
```

SIGNATURE_TYPE_PEHSTR vs SIGNATURE_TYPE_PEHSTR_EXT

- ❑ SIGNATURE_TYPE_PEHSTR is used to perform string matching against Portable Executable
- ❑ Code identifier: 0x61

```
typedef struct _STRUCT_COMMON_SIGNATURE_TYPE {
            UINT8  ui8SignatureType;
            UINT8  ui8SizeLow;
            UINT16 ui16SizeHigh;
            BYTE   pbRuleContent[];
} STRUCT_COMMON_SIGNATURE_TYPE,
*PSTRUCT_COMMON_SIGNATURE_TYPE;
```



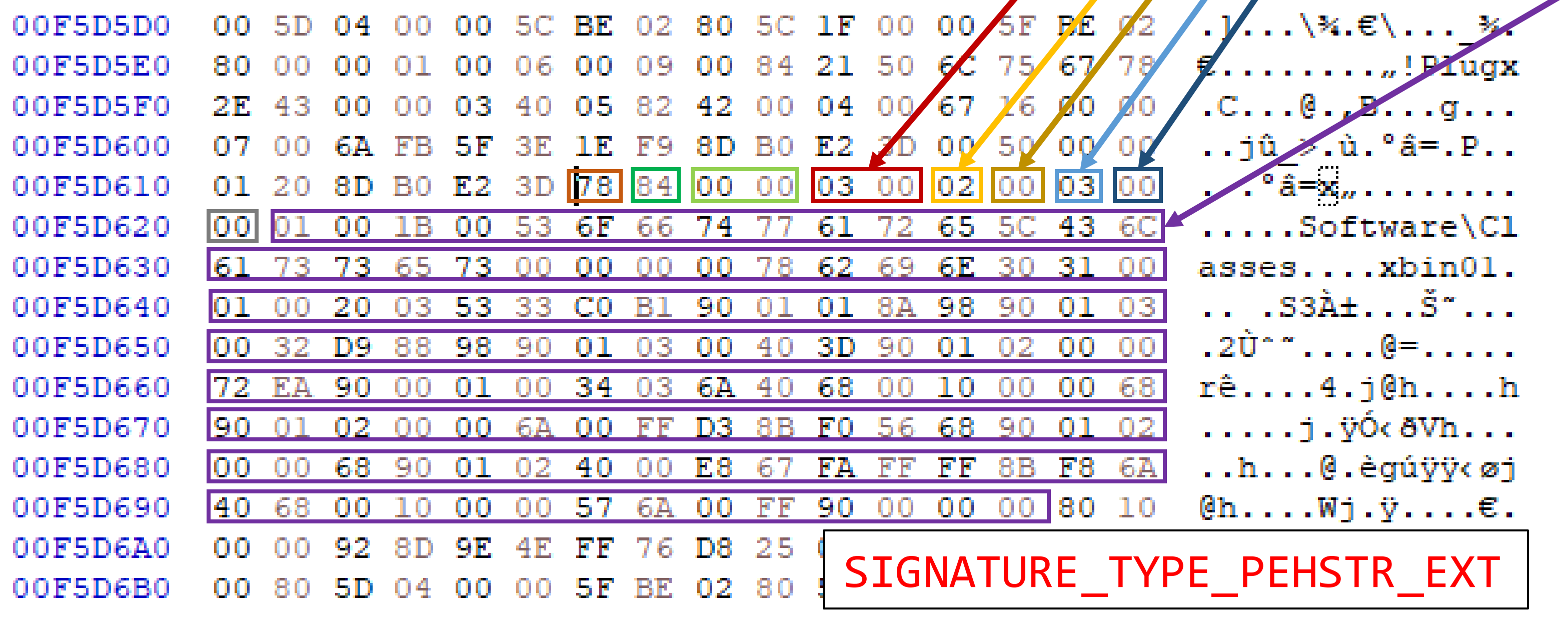
```
00F5D5D0  00 5D 04 00 00 5C BE 02 80 5C 1F 00 00 5F BE 02  .]...\¾.€\..._¾.
00F5D5E0  80 00 00 01 00 06 00 09 00 84 21 50 6C 75 67 78  €........„!Plugx
00F5D5F0  2E 43 00 00 03 40 05 82 42 00 04 00 67 16 00 00  .C...@.‚B...g...
00F5D600  07 00 6A FB 5F 3E 1E F9 8D B0 E2 3D 00 50 00 00  ..jû_>.ù.°â=.P..
00F5D610  01 20 8D B0 E2 3D 78 84 00 00 03 00 02 00 03 00  . .°â=x„........
00F5D620  00 01 00 1B 00 53 6F 66 74 77 61 72 65 5C 43 6C  .....Software\Cl
00F5D630  61 73 73 65 73 00 00 00 00 78 62 69 6E 30 31 00  asses....xbin01.
00F5D640  01 00 20 03 53 33 C0 B1 90 01 01 8A 98 90 01 03  .. .S3À±...Š˜...
00F5D650  00 32 D9 88 98 90 01 03 00 40 3D 90 01 02 00 00  .2Ùˆ˜....@=.....
00F5D660  72 EA 90 00 01 00 34 03 6A 40 68 00 10 00 00 68  rê....4.j@h....h
00F5D670  90 01 02 00 00 6A 00 FF D3 8B F0 56 68 90 01 02  .....j.ÿÓ‹ðVh...
00F5D680  00 00 68 90 01 02 40 00 E8 67 FA FF FF 8B F8 6A  ..h...@.ègúÿÿ‹øj
00F5D690  40 68 00 10 00 00 57 6A 00 FF 90 00 00 00 80 10  @h....Wj.ÿ....€.
00F5D6A0  00 00 92 8D 9E 4E FF 76 D8 25 00 69 2A 9B 00 10  ..'.žNÿvØ%.i*›..
00F5D6B0  00 80 5D 04 00 00 5F BE 02 80 5C 1F 00 00 60 BE  .€]..._¾.€\...`¾
```

- ❑ SIGNATURE_TYPE_PEHSTR_EXT is used to perform byte-matching against Portable Executable
- ❑ Code identifier: 0x78

PEHSTR and PEHSTR_EXT common header

- `ui8TresholdRequiredLow`: the threshold required to obtain a detection from Windows Defender (low part)
- `ui8TresholdRequiredHigh`: the threshold required to obtain a detection from Windows Defender (high part)
- `ui8SubRulesNumberLow`: the number of sub-rules that are found inside this particular signature, to identify the threat (low part).
- `ui8SubRulesNumberHigh`: the number of sub-rules that are found inside this particular signature, to identify the threat. (high part)
- `pbRuleData[]`: contains all the sub-rules, which are used to perform byte-matching detection.

```
typedef struct _STRUCT_PEHSTR_HEADER {
        UINT16   ui16Unknown;
        UINT8    ui8TresholdRequiredLow;
        UINT8    ui8TresholdRequiredHigh;
        UINT8    ui8SubRulesNumberLow;
        UINT8    ui8SubRulesNumberHigh;
        BYTE     bEmpty;
        BYTE     pbRuleData[];
} STRUCT_PEHSTR_HEADER, * PSTRUCT_PEHSTR_HEADER;
```



```
00F5D5D0  00 5D 04 00 00 5C BE 02 80 5C 1F 00 00 5F BE 02  .]...\¾.€\..._¾.
00F5D5E0  80 00 00 01 00 06 00 09 00 84 21 50 6C 75 67 78  €........„!Plugx
00F5D5F0  2E 43 00 00 03 40 05 82 42 00 04 00 67 16 00 00  .C...@.‚B...g...
00F5D600  07 00 6A FB 5F 3E 1E F9 8D B0 E2 3D 00 50 00 00  ..jû_>.ù.°â=.P..
00F5D610  01 20 8D B0 E2 3D 78 84 00 00 03 00 02 00 03 00  ...°â=x„........
00F5D620  00 01 00 1B 00 53 6F 66 74 77 61 72 65 5C 43 6C  .....Software\Cl
00F5D630  61 73 73 65 73 00 00 00 00 78 62 69 6E 30 31 00  asses....xbin01.
00F5D640  01 00 20 03 53 33 C0 B1 90 01 01 8A 98 90 01 03  .. .S3À±...Š˜...
00F5D650  00 32 D9 88 98 90 01 03 00 40 3D 90 01 02 00 00  .2Ùˆ˜....@=.....
00F5D660  72 EA 90 00 01 00 34 03 6A 40 68 00 10 00 00 68  rê....4.j@h....h
00F5D670  90 01 02 00 00 6A 00 FF D3 8B F0 56 68 90 01 02  .....j.ÿÓ‹ðVh...
00F5D680  00 00 68 90 01 02 40 00 E8 67 FA FF FF 8B F8 6A  ..h...@.ègúÿÿ‹øj
00F5D690  40 68 00 10 00 00 57 6A 00 FF 90 00 00 00 80 10  @h....Wj.ÿ....€.
00F5D6A0  00 00 92 8D 9E 4E FF 76 D8 25
00F5D6B0  00 80 5D 04 00 00 5F BE 02 80
```

SIGNATURE_TYPE_PEHSTR_EXT

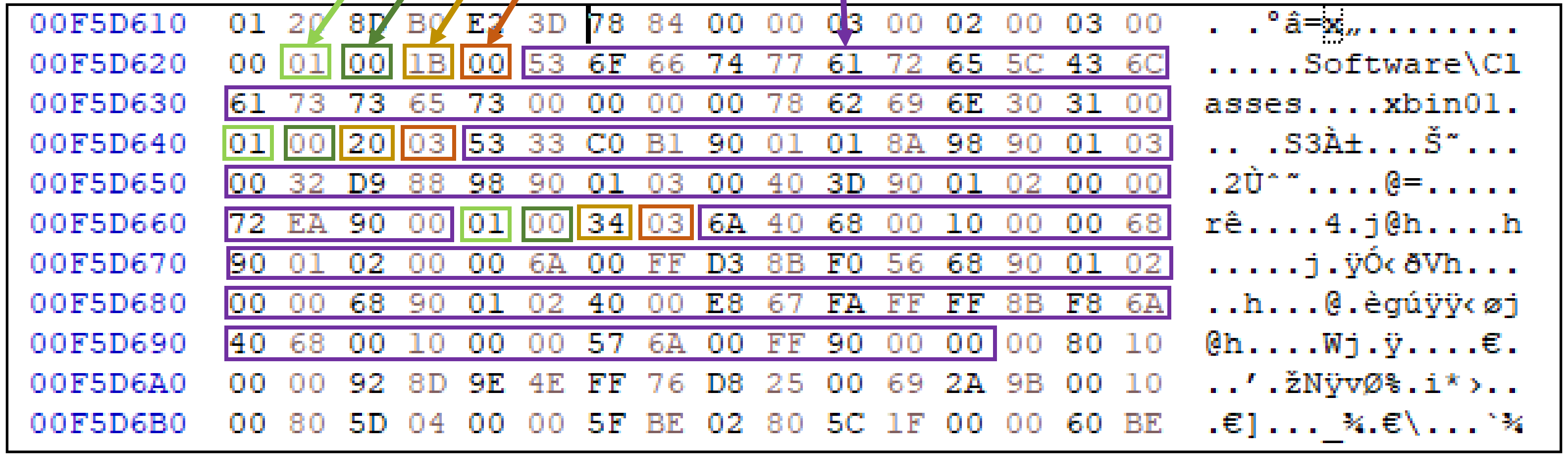
- Both types of signatures share the same structures
- The main difference resides in a slightly different format of the sub-rules structure
- SIGNATURE_TYPE_PEHSTR is used to detect "readable string"
- SIGNATURE_TYPE_PEHSTR_EXT can be used to detect opcodes and has different extra features

PEHSTR and PEHSTR_EXT sub-rule structure

- ❑ `ui8SubRuleWeightLow`: represents the weight that the sub-rule has in the detection process (low part).
- ❑ `ui8SubRuleWeightHigh`: represents the weight that the sub-rule has in the detection process (high part).
- ❑ `ui8SubRuleSize`: specify the size of the byte string to match against a given PE.
- ❑ `ui8CodeUnknown`: unknown field.
- ❑ `pbSubRuleBytesToMatch[]`: the bytes that must be found to obtain a detection.

```
typedef struct _STRUCT_RULE_PEHSTR_EXT {
        UINT8   ui8SubRuleWeightLow;
        UINT8   ui8SubRuleWeightHigh;
        UINT8   ui8SubRuleSize;
        UINT8   ui8CodeUnknown;   //_EXT only
        BYTE    pbSubRuleBytesToMatch[];
} STRUCT_RULE_PEHSTR_EXT,
*PSTRUCT_RULE_PEHSTR_EXT;
```

Example with three sub-rules



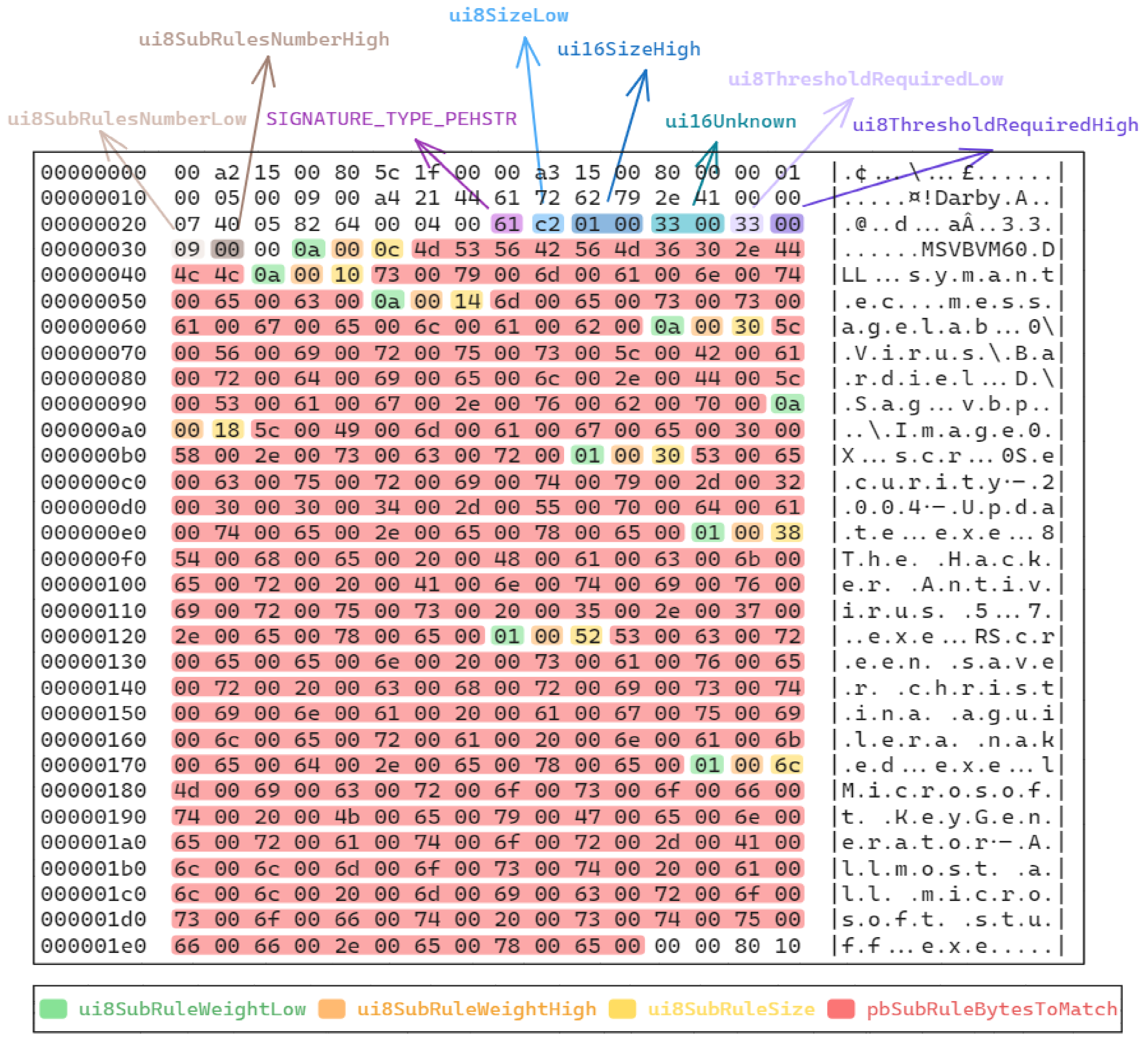
```
00F5D610  01 20 8D B0 E2 3D 78 84 00 00 03 00 02 00 03 00  . .°â=x„........
00F5D620  00 01 00 1B 00 53 6F 66 74 77 61 72 65 5C 43 6C  .....Software\Cl
00F5D630  61 73 73 65 73 00 00 00 00 78 62 69 6E 30 31 00  asses....xbin01.
00F5D640  01 00 20 03 53 33 C0 B1 90 01 01 8A 98 90 01 03  .. .S3À±...Š˜...
00F5D650  00 32 D9 88 98 90 01 03 00 40 3D 90 01 02 00 00  .2Ùˆ˜....@=.....
00F5D660  72 EA 90 00 01 00 34 03 6A 40 68 00 10 00 00 68  rê....4.j@h....h
00F5D670  90 01 02 00 00 6A 00 FF D3 8B F0 56 68 90 01 02  .....j.ÿÓ‹ðVh...
00F5D680  00 00 68 90 01 02 40 00 E8 67 FA FF FF 8B F8 6A  ..h...@.ègúÿÿ‹øj
00F5D690  40 68 00 10 00 00 57 6A 00 FF 90 00 00 00 80 10  @h....Wj.ÿ....€.
00F5D6A0  00 00 92 8D 9E 4E FF 76 D8 25 00 69 2A 9B 00 10  ..'.žNÿvØ%.i*›..
00F5D6B0  00 80 5D 04 00 00 5F BE 02 80 5C 1F 00 00 60 BE  .€]..._¾.€\...`¾
```

Lab1: `SIGNATURE_TYPE_PEHSTR`

- ❑ Open your extracted `mpavbase.vdm` with a hex editor and find all the `SIGNATURE_TYPE_PEHSTR (0x61)` belonging to threat `!Darby.A`
- ❑ Highlight all the fields of each signature (HINT: make a screenshot of the relevant bytes in the hexdump and use mspaint to highlight)
  - ❑ Identify the sub-rules
  - ❑ Identify the threshold
  - ❑ Identify the weight of each sub-rule

Solution SIGNATURE_TYPE_PEHSTR: real example

- The example in figure shows a SIGNATURE_TYPE_PEHSTR from threat !Darby.A
- _STRUCT_PEHSTR_HEADER:
  - ui16Counter1: highlighted in cyan.
  - ui16TresholdRequired: highlighted in purple.
  - ui16SubRulesNumber: highlighted in brown.
- _STRUCT_RULE_PEHSTR:
  - ui16SubRuleWeight: highlighted in green.
  - ui8UnknownCode: highlighted in orange.
  - ui8SubRuleSize: highlighted in yellow.
  - pbSubRuleBytesToMatch[]: hihlighted in red



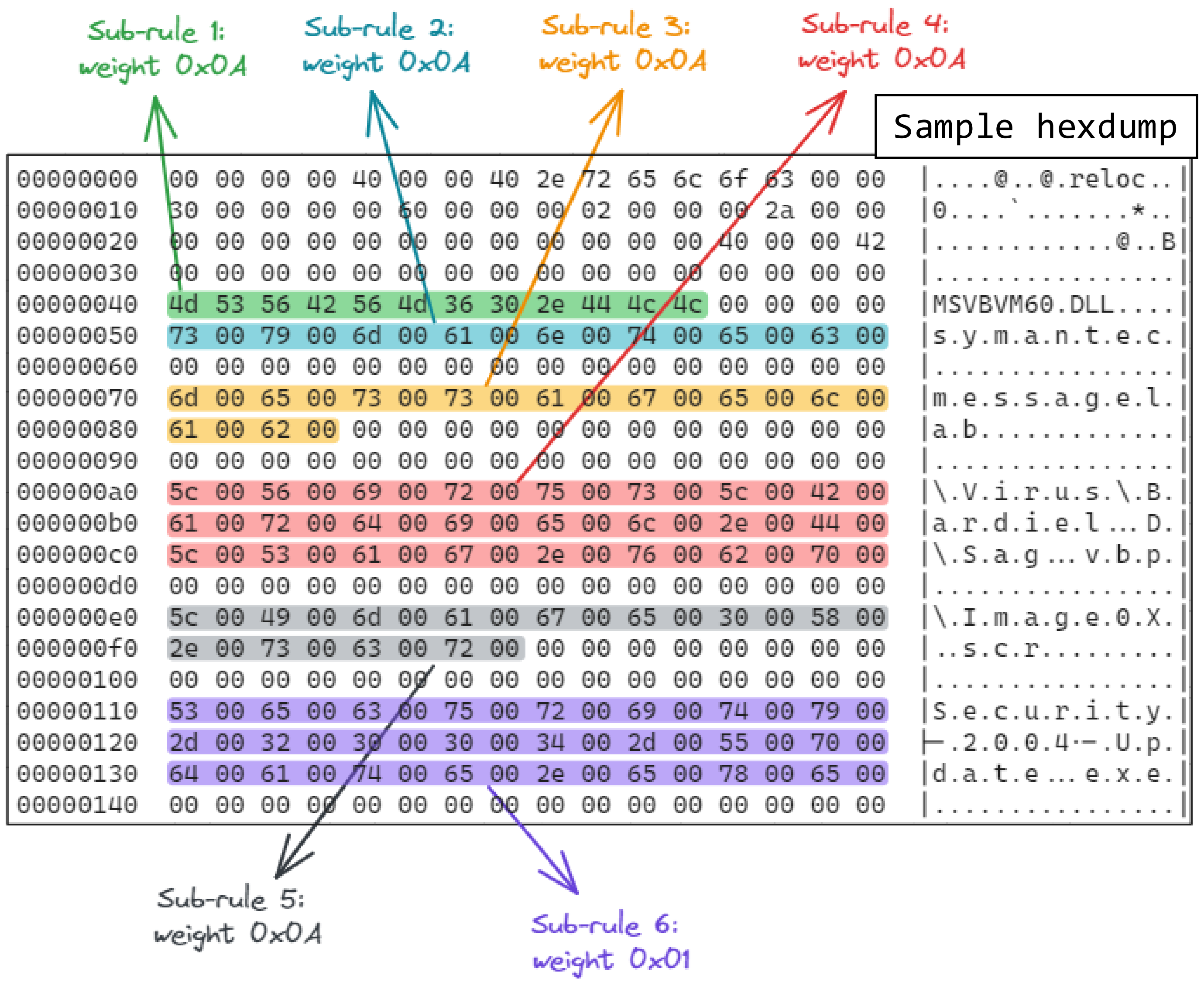
```
00000000  00 a2 15 00 80 5c 1f 00 00 a3 15 00 80 00 00 01  |.¢...\...£......|
00000010  00 05 00 09 00 a4 21 44 61 72 62 79 2e 41 00 00  |.....¤!Darby.A..|
00000020  07 40 05 82 64 00 04 00 61 c2 01 00 33 00 33 00  |.@..d...aÂ..3.3.|
00000030  09 00 00 0a 00 0c 4d 53 56 42 56 4d 36 30 2e 44  |......MSVBVM60.D|
00000040  4c 4c 0a 00 10 73 00 79 00 6d 00 61 00 6e 00 74  |LL...s.y.m.a.n.t|
00000050  00 65 00 63 00 0a 00 14 6d 00 65 00 73 00 73 00  |.e.c....m.e.s.s.|
00000060  61 00 67 00 65 00 6c 00 61 00 62 00 0a 00 30 5c  |a.g.e.l.a.b...0\|
00000070  00 56 00 69 00 72 00 75 00 73 00 5c 00 42 00 61  |.V.i.r.u.s.\.B.a|
00000080  00 72 00 64 00 69 00 65 00 6c 00 2e 00 44 00 5c  |.r.d.i.e.l...D.\|
00000090  00 53 00 61 00 67 00 2e 00 76 00 62 00 70 00 0a  |.S.a.g...v.b.p..|
000000a0  00 18 5c 00 49 00 6d 00 61 00 67 00 65 00 30 00  |..\.I.m.a.g.e.0.|
000000b0  58 00 2e 00 73 00 63 00 72 00 01 00 30 53 00 65  |X...s.c.r...0S.e|
000000c0  00 63 00 75 00 72 00 69 00 74 00 79 00 2d 00 32  |.c.u.r.i.t.y·–.2|
000000d0  00 30 00 30 00 34 00 2d 00 55 00 70 00 64 00 61  |.0.0.4·–.U.p.d.a|
000000e0  00 74 00 65 00 2e 00 65 00 78 00 65 00 01 00 38  |.t.e...e.x.e...8|
000000f0  54 00 68 00 65 00 20 00 48 00 61 00 63 00 6b 00  |T.h.e. .H.a.c.k.|
00000100  65 00 72 00 20 00 41 00 6e 00 74 00 69 00 76 00  |e.r. .A.n.t.i.v.|
00000110  69 00 72 00 75 00 73 00 20 00 35 00 2e 00 37 00  |i.r.u.s. .5...7.|
00000120  2e 00 65 00 78 00 65 00 01 00 52 53 00 63 00 72  |..e.x.e...RS.c.r|
00000130  00 65 00 65 00 6e 00 20 00 73 00 61 00 76 00 65  |.e.e.n. .s.a.v.e|
00000140  00 72 00 20 00 63 00 68 00 72 00 69 00 73 00 74  |.r. .c.h.r.i.s.t|
00000150  00 69 00 6e 00 61 00 20 00 61 00 67 00 75 00 69  |.i.n.a. .a.g.u.i|
00000160  00 6c 00 65 00 72 00 61 00 20 00 6e 00 61 00 6b  |.l.e.r.a. .n.a.k|
00000170  00 65 00 64 00 2e 00 65 00 78 00 65 00 01 00 6c  |.e.d...e.x.e...l|
00000180  4d 00 69 00 63 00 72 00 6f 00 73 00 6f 00 66 00  |M.i.c.r.o.s.o.f.|
00000190  74 00 20 00 4b 00 65 00 79 00 47 00 65 00 6e 00  |t. .K.e.y.G.e.n.|
000001a0  65 00 72 00 61 00 74 00 6f 00 72 00 2d 00 41 00  |e.r.a.t.o.r·–.A.|
000001b0  6c 00 6c 00 6d 00 6f 00 73 00 74 00 20 00 61 00  |l.l.m.o.s.t. .a.|
000001c0  6c 00 6c 00 20 00 6d 00 69 00 63 00 72 00 6f 00  |l.l. .m.i.c.r.o.|
000001d0  73 00 6f 00 66 00 74 00 20 00 73 00 74 00 75 00  |s.o.f.t. .s.t.u.|
000001e0  66 00 66 00 2e 00 65 00 78 00 65 00 00 00 80 10  |f.f...e.x.e.....|
```

🟩 ui8SubRuleWeightLow 🟧 ui8SubRuleWeightHigh 🟨 ui8SubRuleSize 🟥 pbSubRuleBytesToMatch

SIGNATURE_TYPE_PEHSTR: matching a !Darby.A signature

- The signature has a `ui16TresholdRequired` equal to `0x33`
  - To obtain a detection the threshold must be reached
- In the example the following sub-rules are involved:
  - Sub-rule 1: weight `0x0A`.
  - Sub-rule 2: weight `0x0A`.
  - Sub-rule 3: weight `0x0A`.
  - Sub-rule 4: weight `0x0A`.
  - Sub-rule 5: weight `0x0A`.
  - Sub-rule 6: weight `0x01`.

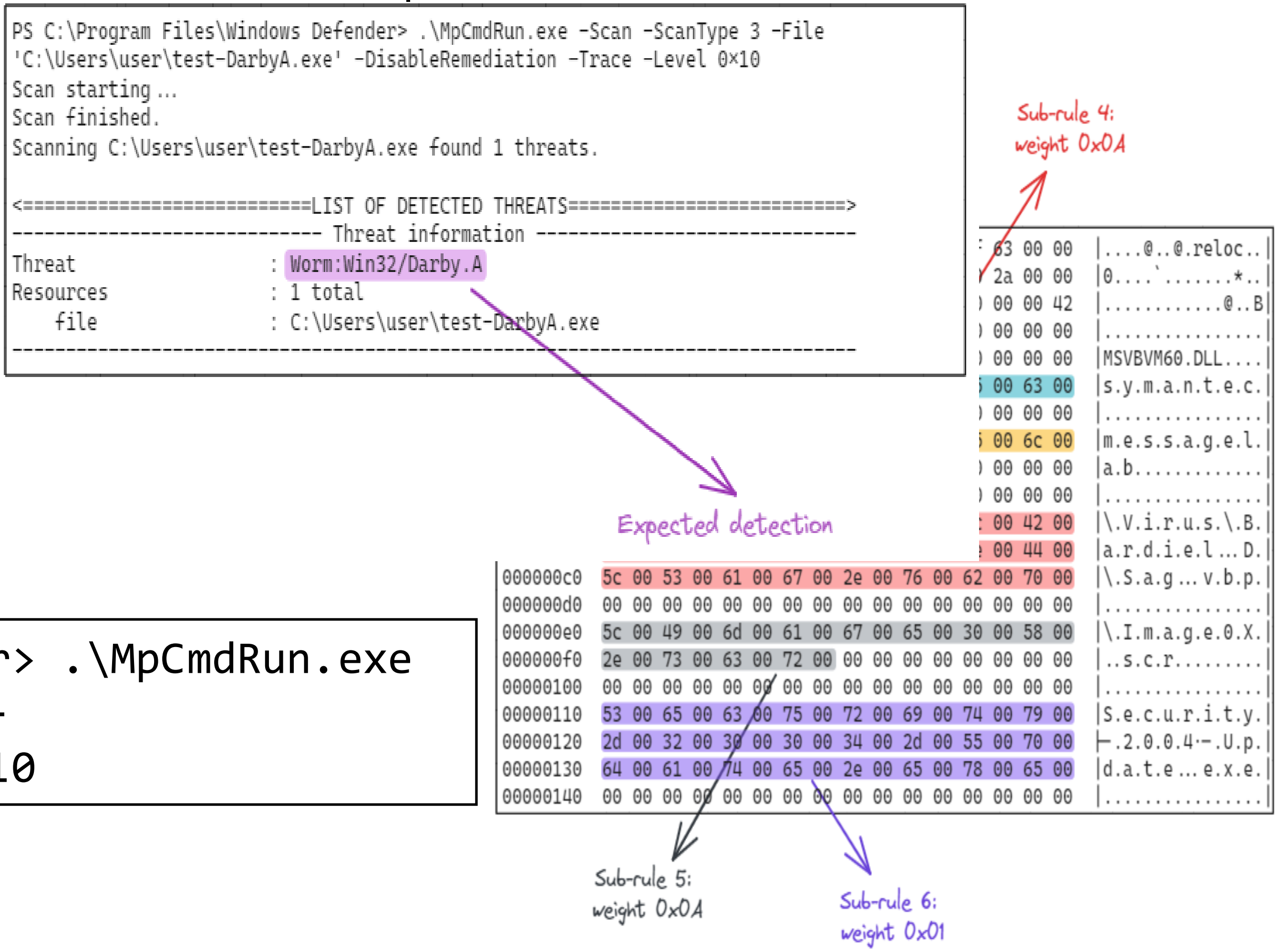
$$\sum = 0x33$$



```
00000000  00 00 00 00 40 00 00 40 2e 72 65 6c 6f 63 00 00  |....@..@.reloc..|
00000010  30 00 00 00 00 60 00 00 00 02 00 00 00 2a 00 00  |0....`.......*..|
00000020  00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 42  |............@..B|
00000030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |................|
00000040  4d 53 56 42 56 4d 36 30 2e 44 4c 4c 00 00 00 00  |MSVBVM60.DLL....|
00000050  73 00 79 00 6d 00 61 00 6e 00 74 00 65 00 63 00  |s.y.m.a.n.t.e.c.|
00000060  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |................|
00000070  6d 00 65 00 73 00 73 00 61 00 67 00 65 00 6c 00  |m.e.s.s.a.g.e.l.|
00000080  61 00 62 00 00 00 00 00 00 00 00 00 00 00 00 00  |a.b.............|
00000090  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |................|
000000a0  5c 00 56 00 69 00 72 00 75 00 73 00 5c 00 42 00  |\.V.i.r.u.s.\.B.|
000000b0  61 00 72 00 64 00 69 00 65 00 6c 00 2e 00 44 00  |a.r.d.i.e.l...D.|
000000c0  5c 00 53 00 61 00 67 00 2e 00 76 00 62 00 70 00  |\.S.a.g...v.b.p.|
000000d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |................|
000000e0  5c 00 49 00 6d 00 61 00 67 00 65 00 30 00 58 00  |\.I.m.a.g.e.0.X.|
000000f0  2e 00 73 00 63 00 72 00 00 00 00 00 00 00 00 00  |..s.c.r.........|
00000100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |................|
00000110  53 00 65 00 63 00 75 00 72 00 69 00 74 00 79 00  |S.e.c.u.r.i.t.y.|
00000120  2d 00 32 00 30 00 30 00 34 00 2d 00 55 00 70 00  |-.2.0.0.4.-.U.p.|
00000130  64 00 61 00 74 00 65 00 2e 00 65 00 78 00 65 00  |d.a.t.e...e.x.e.|
00000140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |................|
```

Fast way to check ...

Scan your stuffs with `MpCmdRun.exe` utility provided by Windows Defender itself



```
PS C:\Program Files\Windows Defender> .\MpCmdRun.exe -Scan -ScanType 3 -File
'C:\Users\user\test-DarbyA.exe' -DisableRemediation -Trace -Level 0x10
Scan starting...
Scan finished.
Scanning C:\Users\user\test-DarbyA.exe found 1 threats.

<===========================LIST OF DETECTED THREATS==========================>
----------------------------- Threat information -----------------------------
Threat                  : Worm:Win32/Darby.A
Resources               : 1 total
    file                : C:\Users\user\test-DarbyA.exe
-------------------------------------------------------------------------------
```

```
                                                      63 00 00   |....@..@.reloc..|
                                                      2a 00 00   |0....`.......*..|
                                                      00 00 42   |...........@..B|
                                                      00 00 00   |................|
                                                      00 00 00   |MSVBVM60.DLL....|
                                                      00 63 00   |s.y.m.a.n.t.e.c.|
                                                      00 00 00   |................|
                                                      00 6c 00   |m.e.s.s.a.g.e.l.|
                                                      00 00 00   |a.b.............|
                                                      00 00 00   |................|
                                                      00 42 00   |\.V.i.r.u.s.\.B.|
                                                      00 44 00   |a.r.d.i.e.l...D.|
000000c0  5c 00 53 00 61 00 67 00 2e 00 76 00 62 00 70 00   |\.S.a.g...v.b.p.|
000000d0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   |................|
000000e0  5c 00 49 00 6d 00 61 00 67 00 65 00 30 00 58 00   |\.I.m.a.g.e.0.X.|
000000f0  2e 00 73 00 63 00 72 00 00 00 00 00 00 00 00 00   |..s.c.r.........|
00000100  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   |................|
00000110  53 00 65 00 63 00 75 00 72 00 69 00 74 00 79 00   |S.e.c.u.r.i.t.y.|
00000120  2d 00 32 00 30 00 30 00 34 00 2d 00 55 00 70 00   |-.2.0.0.4.-.U.p.|
00000130  64 00 61 00 74 00 65 00 2e 00 65 00 78 00 65 00   |d.a.t.e...e.x.e.|
00000140  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   |................|
```

```
PS C:\Program Files\Windows Defender> .\MpCmdRun.exe
-Scan -ScanType 3 -File <filepath> -
DisableRemediation -Trace -Level 0x10
```

Lab2: Remove Darby signature

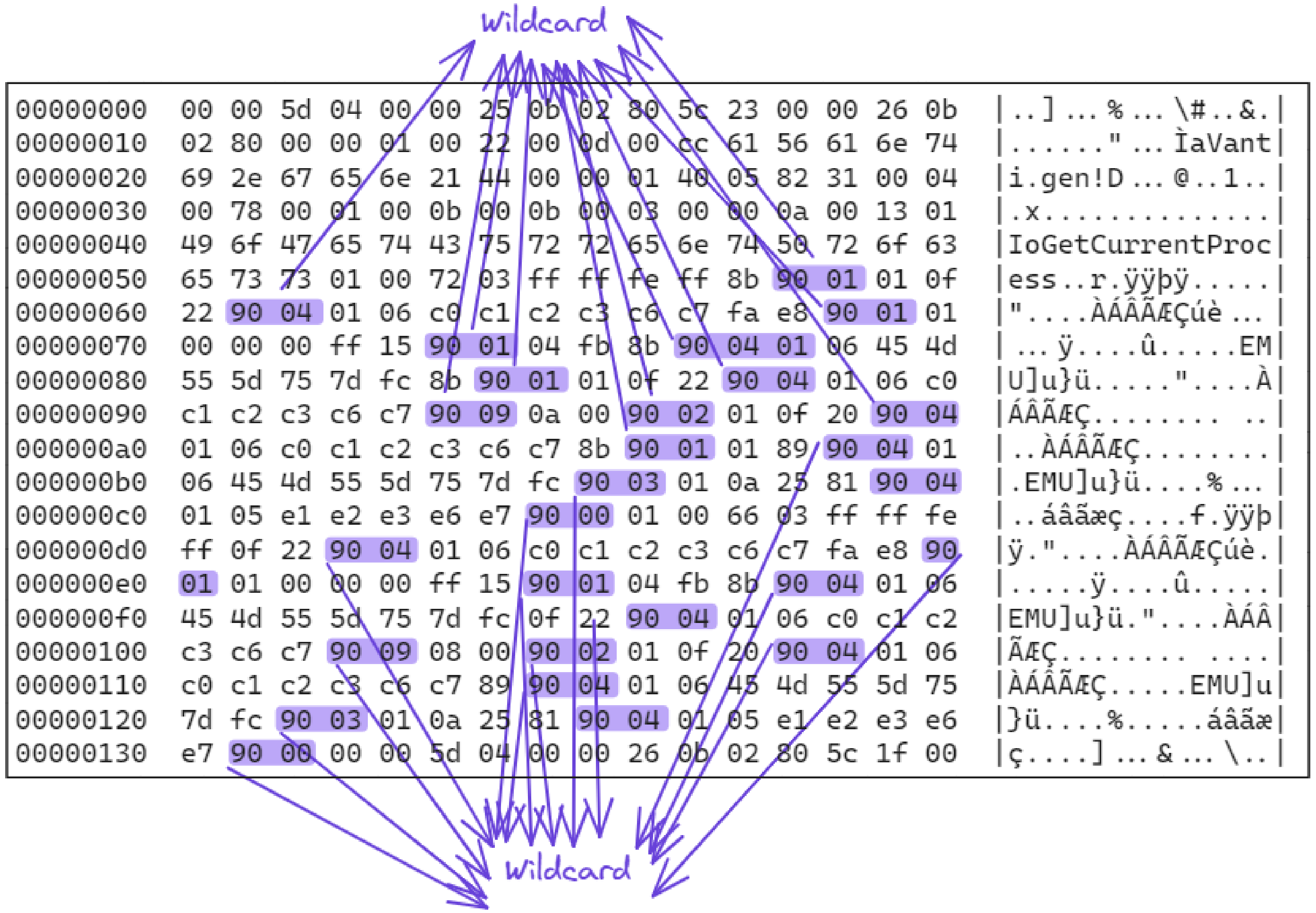
- ❑ Add a folder to the Defenders exclusions

```
PS> Add-MpPreference -ExclusionPath 'C:\YOUR_PATH_HERE'
```

- ❑ Copy the Darby zip into the excluded folder and uncompress it (pwd:infected)
- ❑ Open the binary with an hex editor
- ❑ Identify which bytes trigger the signature and modify them to evade the detection
  - ❑ What is the minimum number of bytes that you have to modify to avoid the dection?
- ❑ How the total weight is affected when the same sub-rule appear more than once?
  - ❑ Suppose that the string `S_1` with weight `W_1` appears twice in the binary. Does the binary get a weight of `2*W_1`?

Give some power to EXT

- Multiple patterns are present inside sub-rules in `SIGNATURE_TYPE_PEHSTR_EXT`
- It can be used to detect opcodes and more
- Used to match specific sequences of bytes
- Wildcard identified:
  - `90 01 XX`
  - `90 02 XX`
  - `90 03 XX YY`
  - `90 04 XX YY`
  - `90 05 XX YY`
- Wildcard still unknown:
  - `90 06 -> 90 20`



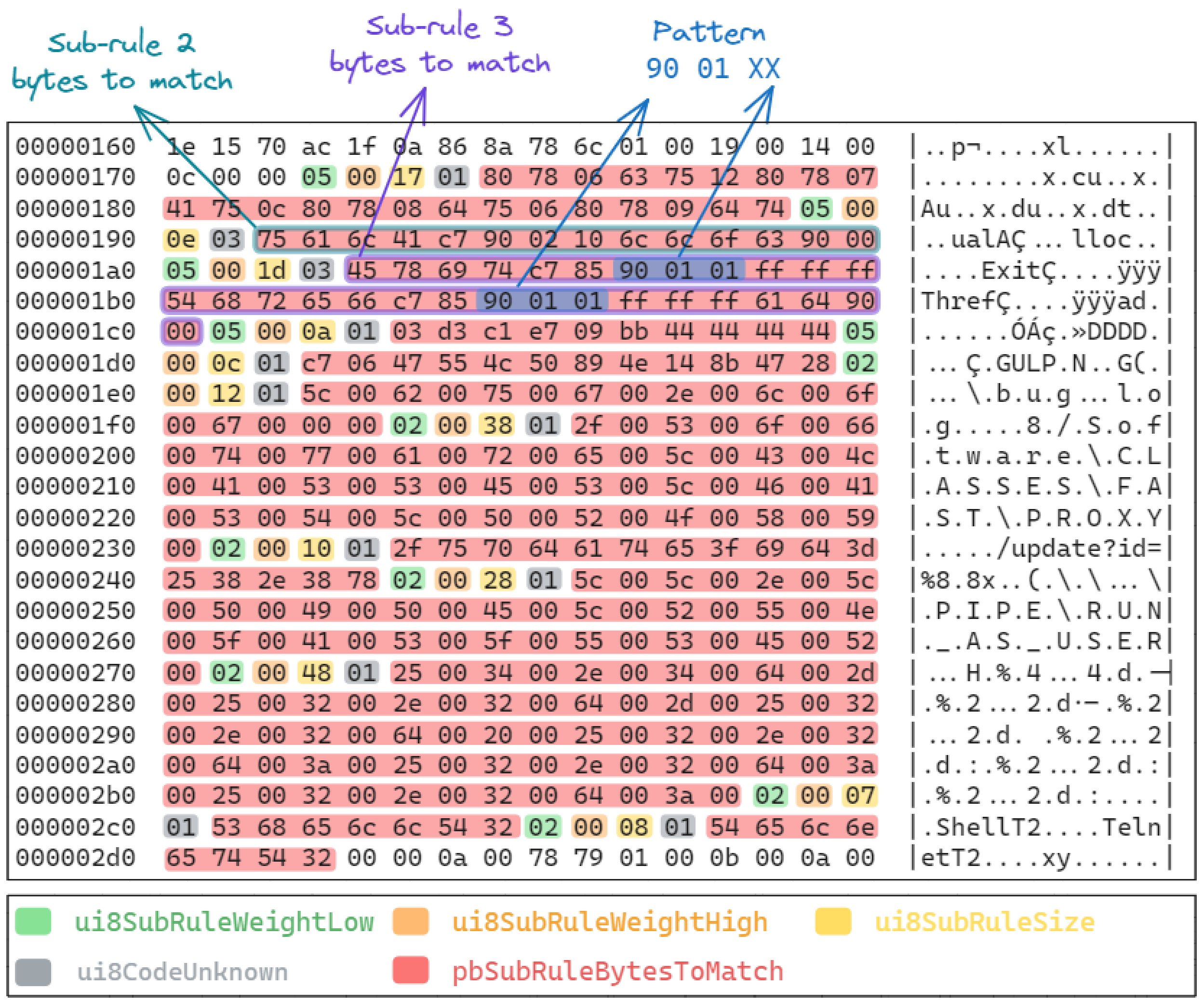
```
00000000  00 00 5d 04 00 00 25 0b 02 80 5c 23 00 00 26 0b  |..]...%...\#..&.|
00000010  02 80 00 00 01 00 22 00 0d 00 cc 61 56 61 6e 74  |......"...ÌaVant|
00000020  69 2e 67 65 6e 21 44 00 00 01 40 05 82 31 00 04  |i.gen!D...@..1..|
00000030  00 78 00 01 00 0b 00 0b 00 03 00 00 0a 00 13 01  |.x..............|
00000040  49 6f 47 65 74 43 75 72 72 65 6e 74 50 72 6f 63  |IoGetCurrentProc|
00000050  65 73 73 01 00 72 03 ff ff fe ff 8b 90 01 01 0f  |ess..r.ÿÿþÿ.....|
00000060  22 90 04 01 06 c0 c1 c2 c3 c6 c7 fa e8 90 01 01  |"....ÀÁÂÃÆÇúè...|
00000070  00 00 00 ff 15 90 01 04 fb 8b 90 04 01 06 45 4d  |...ÿ....û.....EM|
00000080  55 5d 75 7d fc 8b 90 01 01 0f 22 90 04 01 06 c0  |U]u}ü....."....À|
00000090  c1 c2 c3 c6 c7 90 09 0a 00 90 02 01 0f 20 90 04  |ÁÂÃÆÇ....... ..|
000000a0  01 06 c0 c1 c2 c3 c6 c7 8b 90 01 01 89 90 04 01  |..ÀÁÂÃÆÇ........|
000000b0  06 45 4d 55 5d 75 7d fc 90 03 01 0a 25 81 90 04  |.EMU]u}ü....%...|
000000c0  01 05 e1 e2 e3 e6 e7 90 00 01 00 66 03 ff ff fe  |..áâãæç....f.ÿÿþ|
000000d0  ff 0f 22 90 04 01 06 c0 c1 c2 c3 c6 c7 fa e8 90  |ÿ."....ÀÁÂÃÆÇúè.|
000000e0  01 01 00 00 00 ff 15 90 01 04 fb 8b 90 04 01 06  |.....ÿ....û.....|
000000f0  45 4d 55 5d 75 7d fc 0f 22 90 04 01 06 c0 c1 c2  |EMU]u}ü."....ÀÁÂ|
00000100  c3 c6 c7 90 09 08 00 90 02 01 0f 20 90 04 01 06  |ÃÆÇ........ ....|
00000110  c0 c1 c2 c3 c6 c7 89 90 04 01 06 45 4d 55 5d 75  |ÀÁÂÃÆÇ.....EMU]u|
00000120  7d fc 90 03 01 0a 25 81 90 04 01 05 e1 e2 e3 e6  |}ü....%.....áâãæ|
00000130  e7 90 00 00 00 5d 04 00 00 26 0b 02 80 5c 1f 00  |ç....]...&...\..|
```

Patterns: 90 01 XX

Pattern 90 01 XX:

- ❑ Used in sub-rules in SIGNATURE_TYPE_PEHSTR_EXT
- ❑ Match a sequence of bytes that has a specific length defined by XX,
  - ❑ The sequence must appear just after the XX byte
- ❑ An example is highlighted in blue

```
PlugxA-Sub-Rule3-Example{
    strings:
      $sub_rule_3_hex = {
                    45 78 69 74 C7 85 ?? FF FF FF 54 68
                    72 65 66 C7 85 ?? 04 FF FF FF 61 64
                }
    condition:
            $sub_rule_3_hex
}
```



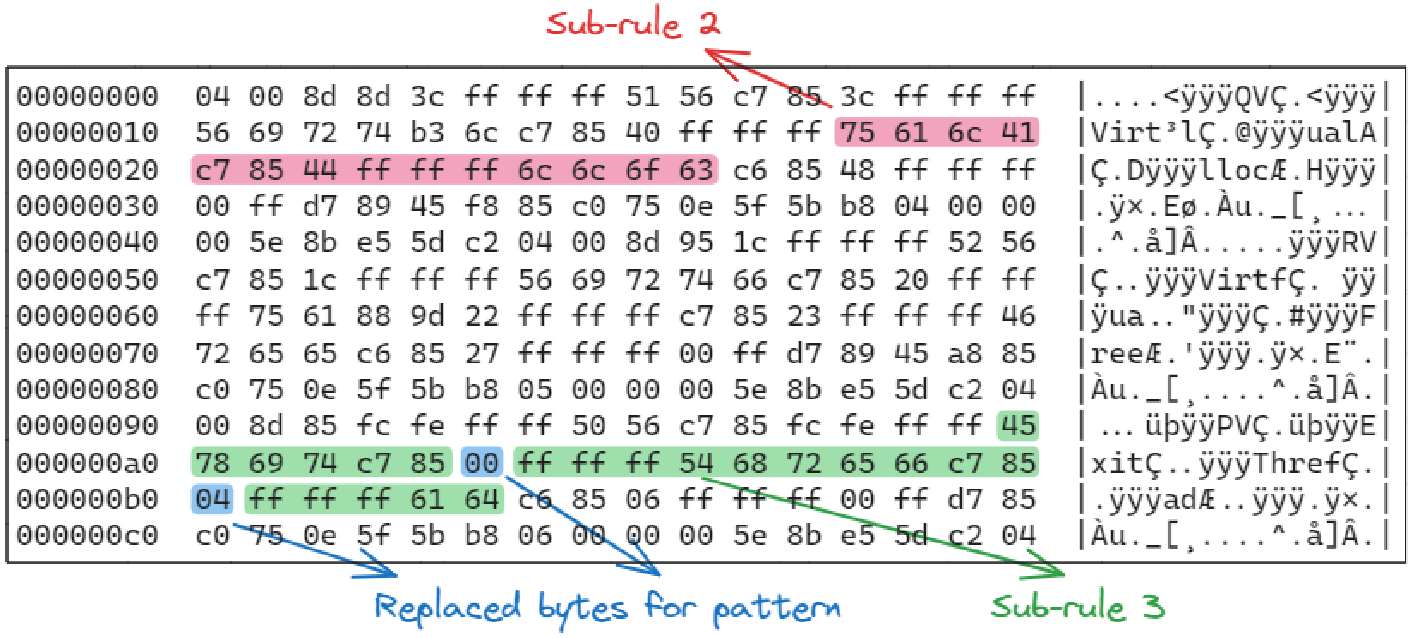
```
00000160  1e 15 70 ac 1f 0a 86 8a 78 6c 01 00 19 00 14 00  |..p¬....xl......|
00000170  0c 00 00 05 00 17 01 80 78 06 63 75 12 80 78 07  |.........x.cu..x.|
00000180  41 75 0c 80 78 08 64 75 06 80 78 09 64 74 05 00  |Au..x.du..x.dt..|
00000190  0e 03 75 61 6c 41 c7 90 02 10 6c 6c 6f 63 90 00  |..ualAÇ...lloc..|
000001a0  05 00 1d 03 45 78 69 74 c7 85 90 01 01 ff ff ff  |....ExitÇ....ÿÿÿ|
000001b0  54 68 72 65 66 c7 85 90 01 01 ff ff ff 61 64 90  |ThrefÇ....ÿÿÿad.|
000001c0  00 05 00 0a 01 03 d3 c1 e7 09 bb 44 44 44 44 05  |......ÓÁç.»DDDD.|
000001d0  00 0c 01 c7 06 47 55 4c 50 89 4e 14 8b 47 28 02  |...Ç.GULP.N..G(.|
000001e0  00 12 01 5c 00 62 00 75 00 67 00 2e 00 6c 00 6f  |...\.b.u.g...l.o|
000001f0  00 67 00 00 00 02 00 38 01 2f 00 53 00 6f 00 66  |.g.....8./.S.o.f|
00000200  00 74 00 77 00 61 00 72 00 65 00 5c 00 43 00 4c  |.t.w.a.r.e.\.C.L|
00000210  00 41 00 53 00 53 00 45 00 53 00 5c 00 46 00 41  |.A.S.S.E.S.\.F.A|
00000220  00 53 00 54 00 5c 00 50 00 52 00 4f 00 58 00 59  |.S.T.\.P.R.O.X.Y|
00000230  00 02 00 10 01 2f 75 70 64 61 74 65 3f 69 64 3d  |...../update?id=|
00000240  25 38 2e 38 78 02 00 28 01 5c 00 5c 00 2e 00 5c  |%8.8x..(.\.\...\|
00000250  00 50 00 49 00 50 00 45 00 5c 00 52 00 55 00 4e  |.P.I.P.E.\.R.U.N|
00000260  00 5f 00 41 00 53 00 5f 00 55 00 53 00 45 00 52  |._.A.S._.U.S.E.R|
00000270  00 02 00 48 01 25 00 34 00 2e 00 34 00 64 00 2d  |...H.%.4...4.d.-|
00000280  00 25 00 32 00 2e 00 32 00 64 00 2d 00 25 00 32  |.%.2...2.d·-.%.2|
00000290  00 2e 00 32 00 64 00 20 00 25 00 32 00 2e 00 32  |...2.d. .%.2...2|
000002a0  00 64 00 3a 00 25 00 32 00 2e 00 32 00 64 00 3a  |.d.:.%.2...2.d.:|
000002b0  00 25 00 32 00 2e 00 32 00 64 00 3a 00 02 00 07  |.%.2...2.d.:....|
000002c0  01 53 68 65 6c 6c 54 32 02 00 08 01 54 65 6c 6e  |.ShellT2....Teln|
000002d0  65 74 54 32 00 00 0a 00 78 79 01 00 0b 00 0a 00  |etT2....xy......|
```

🟩	ui8SubRuleWeightLow	🟧	ui8SubRuleWeightHigh
🟨	ui8SubRuleSize	⬜	ui8CodeUnknown
🟥	pbSubRuleBytesToMatch		

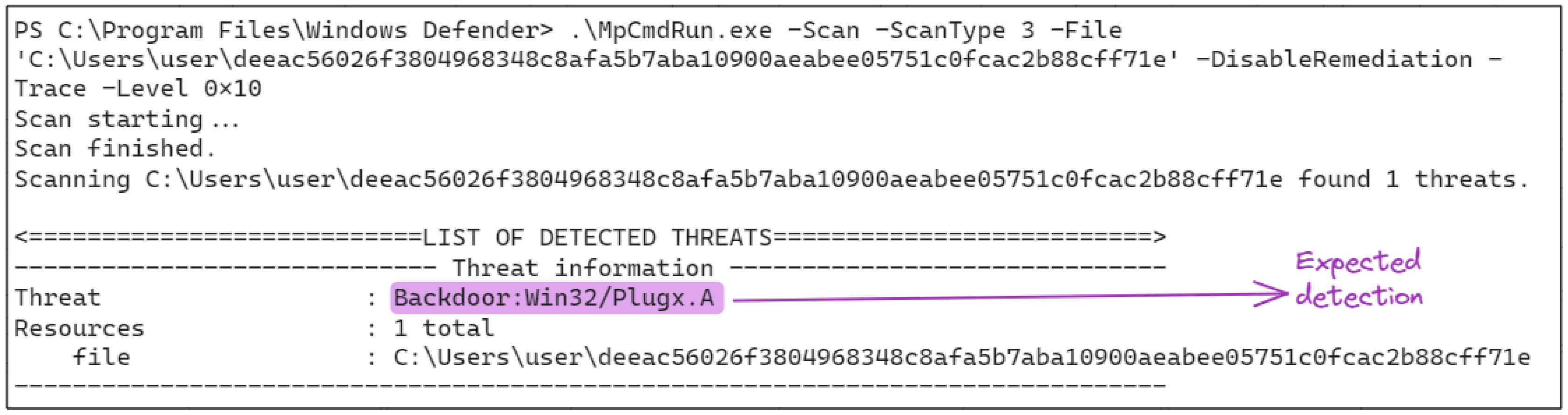
Patterns: `90 01 XX`

Pattern `90 01 XX` detection:

- ❑ **Using `MpCmdRun.exe`**
- ❑ **The bytes placed in place of the pattern `90 01 01` are (Highlighted in blue):**
  - ❑ `0x00`
  - ❑ `0x04`
- ❑ **In red sub-rule 2**
- ❑ **In green sub-rule 3**
- ❑ **Expected detection: `Plugx.A`**



```
00000000  04 00 8d 8d 3c ff ff ff 51 56 c7 85 3c ff ff ff  |....<ÿÿÿQVÇ.<ÿÿÿ|
00000010  56 69 72 74 b3 6c c7 85 40 ff ff ff 75 61 6c 41  |Virt³lÇ.@ÿÿÿualA|
00000020  c7 85 44 ff ff ff 6c 6c 6f 63 c6 85 48 ff ff ff  |Ç.DÿÿÿllocÆ.Hÿÿÿ|
00000030  00 ff d7 89 45 f8 85 c0 75 0e 5f 5b b8 04 00 00  |.ÿ×.Eø.Àu._[¸...|
00000040  00 5e 8b e5 5d c2 04 00 8d 95 1c ff ff ff 52 56  |.^.å]Â.....ÿÿÿRV|
00000050  c7 85 1c ff ff ff 56 69 72 74 66 c7 85 20 ff ff  |Ç..ÿÿÿVirtfÇ. ÿÿ|
00000060  ff 75 61 88 9d 22 ff ff ff c7 85 23 ff ff ff 46  |ÿua.."ÿÿÿÇ.#ÿÿÿF|
00000070  72 65 65 c6 85 27 ff ff ff 00 ff d7 89 45 a8 85  |reeÆ.'ÿÿÿ.ÿ×.E¨.|
00000080  c0 75 0e 5f 5b b8 05 00 00 00 5e 8b e5 5d c2 04  |Àu._[¸....^.å]Â.|
00000090  00 8d 85 fc fe ff ff 50 56 c7 85 fc fe ff ff 45  |...üþÿÿPVÇ.üþÿÿE|
000000a0  78 69 74 c7 85 00 ff ff ff 54 68 72 65 66 c7 85  |xitÇ..ÿÿÿThrefÇ.|
000000b0  04 ff ff ff 61 64 c6 85 06 ff ff ff 00 ff d7 85  |.ÿÿÿadÆ..ÿÿÿ.ÿ×.|
000000c0  c0 75 0e 5f 5b b8 06 00 00 00 5e 8b e5 5d c2 04  |Àu._[¸....^.å]Â.|
```



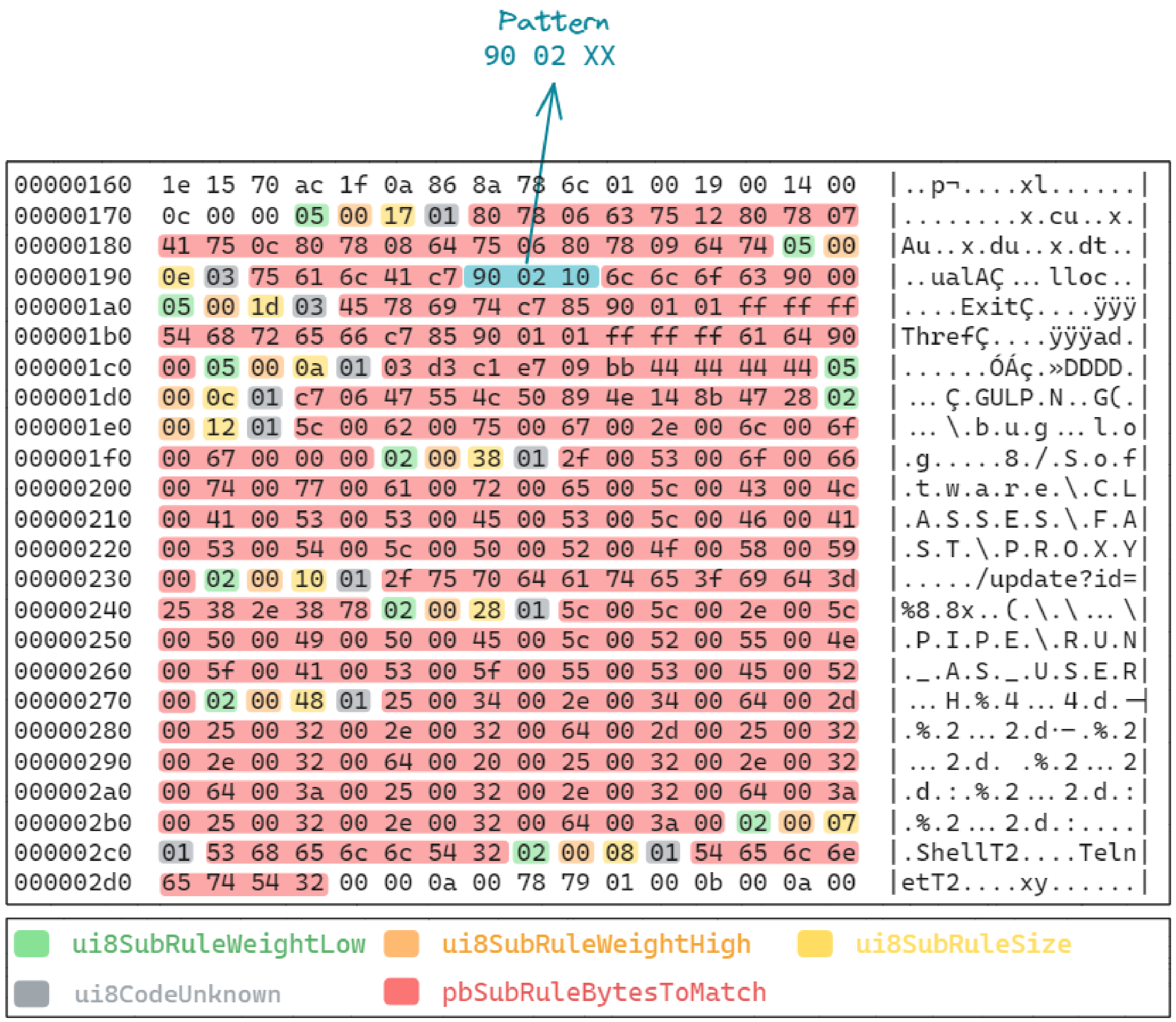
```
PS C:\Program Files\Windows Defender> .\MpCmdRun.exe -Scan -ScanType 3 -File
'C:\Users\user\deeac56026f3804968348c8afa5b7aba10900aeabee05751c0fcac2b88cff71e' -DisableRemediation -
Trace -Level 0x10
Scan starting...
Scan finished.
Scanning C:\Users\user\deeac56026f3804968348c8afa5b7aba10900aeabee05751c0fcac2b88cff71e found 1 threats.

<===========================LIST OF DETECTED THREATS==========================>
----------------------------- Threat information ------------------------------
Threat                   : Backdoor:Win32/Plugx.A
Resources                : 1 total
    file                 : C:\Users\user\deeac56026f3804968348c8afa5b7aba10900aeabee05751c0fcac2b88cff71e
-------------------------------------------------------------------------------
```

Patterns: `90 02 XX`

Pattern `90 02 XX`:

- ❑ Used as a placeholder to match up to `XX` bytes in a specific position
- ❑ Example of pattern highlighted in cyan

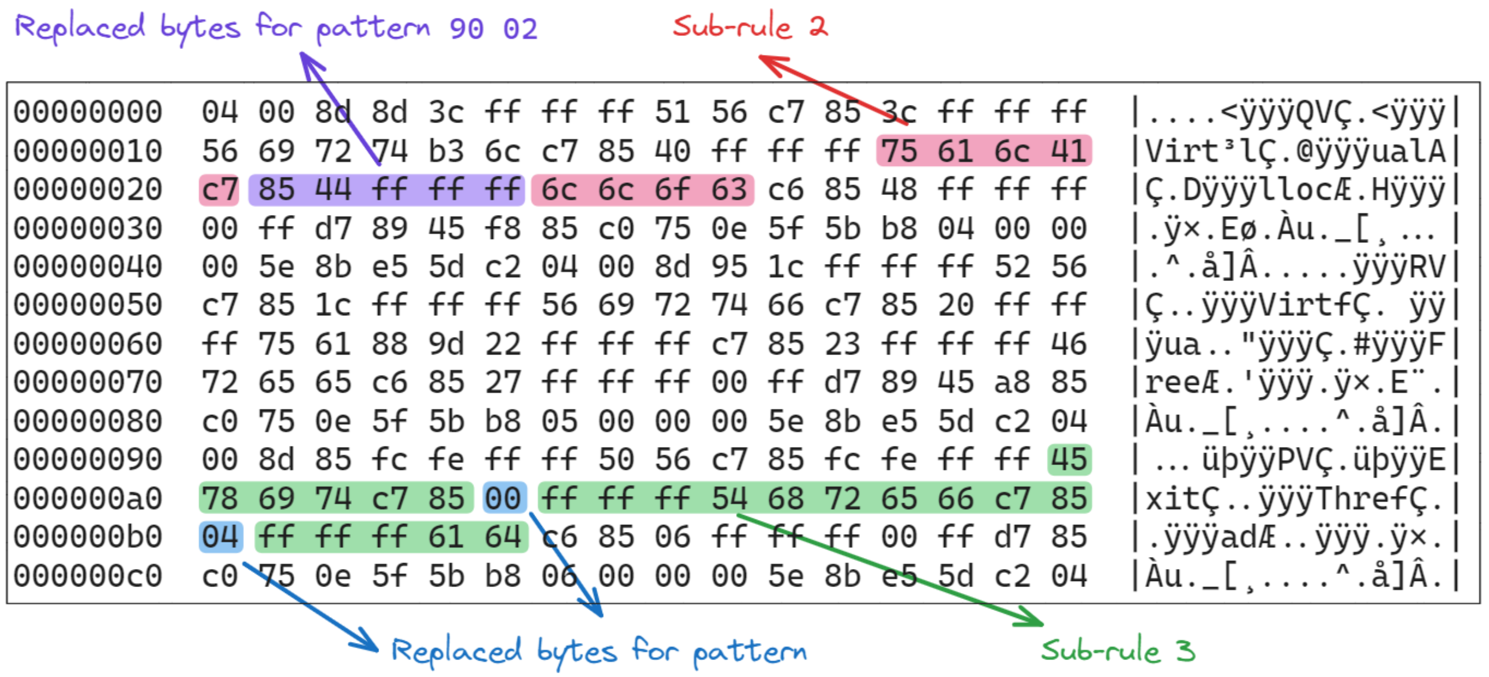
```
rule PlugxA-Sub-Rule2-Example {
strings:
        $sub_rule_2_hex = {75 61 6C 41 C7 [0-16] 6C 6C 6F 63 }
condition:
        $sub_rule_2_hex
}
```



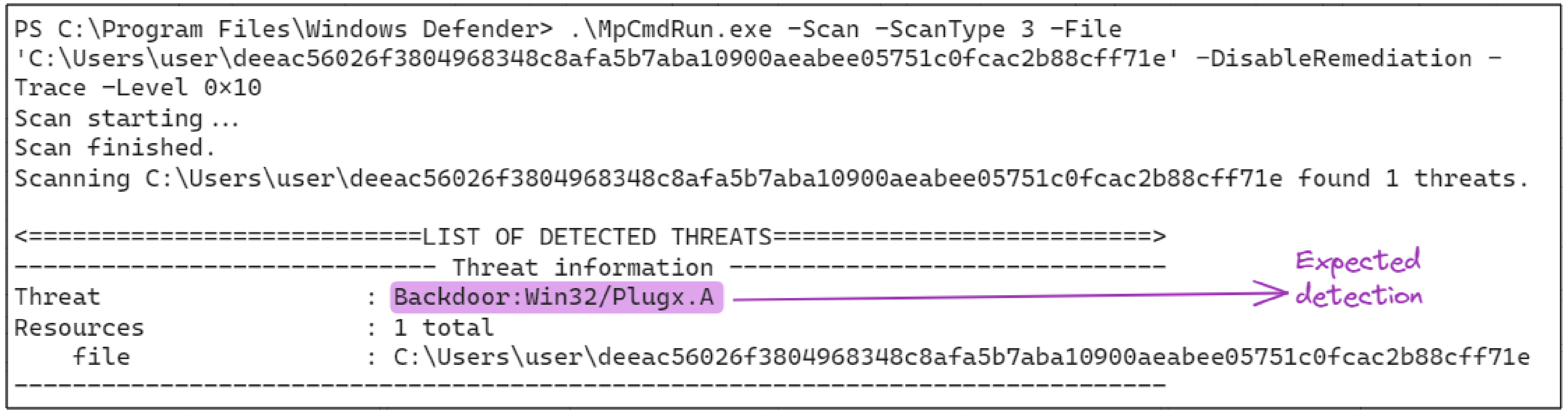
```
00000160  1e 15 70 ac 1f 0a 86 8a 78 6c 01 00 19 00 14 00  |..p¬....xl......|
00000170  0c 00 00 05 00 17 01 80 78 06 63 75 12 80 78 07  |........x.cu..x.|
00000180  41 75 0c 80 78 08 64 75 06 80 78 09 64 74 05 00  |Au..x.du..x.dt..|
00000190  0e 03 75 61 6c 41 c7 90 02 10 6c 6c 6f 63 90 00  |..ualAÇ...lloc..|
000001a0  05 00 1d 03 45 78 69 74 c7 85 90 01 01 ff ff ff  |....ExitÇ....ÿÿÿ|
000001b0  54 68 72 65 66 c7 85 90 01 01 ff ff ff 61 64 90  |ThrefÇ....ÿÿÿad.|
000001c0  00 05 00 0a 01 03 d3 c1 e7 09 bb 44 44 44 44 05  |......ÓÁç.»DDDD.|
000001d0  00 0c 01 c7 06 47 55 4c 50 89 4e 14 8b 47 28 02  |...Ç.GULP.N..G(.|
000001e0  00 12 01 5c 00 62 00 75 00 67 00 2e 00 6c 00 6f  |...\.b.u.g...l.o|
000001f0  00 67 00 00 00 02 00 38 01 2f 00 53 00 6f 00 66  |.g.....8./.S.o.f|
00000200  00 74 00 77 00 61 00 72 00 65 00 5c 00 43 00 4c  |.t.w.a.r.e.\.C.L|
00000210  00 41 00 53 00 53 00 45 00 53 00 5c 00 46 00 41  |.A.S.S.E.S.\.F.A|
00000220  00 53 00 54 00 5c 00 50 00 52 00 4f 00 58 00 59  |.S.T.\.P.R.O.X.Y|
00000230  00 02 00 10 01 2f 75 70 64 61 74 65 3f 69 64 3d  |...../update?id=|
00000240  25 38 2e 38 78 02 00 28 01 5c 00 5c 00 2e 00 5c  |%8.8x..(.\.\...\|
00000250  00 50 00 49 00 50 00 45 00 5c 00 52 00 55 00 4e  |.P.I.P.E.\.R.U.N|
00000260  00 5f 00 41 00 53 00 5f 00 55 00 53 00 45 00 52  |._.A.S._.U.S.E.R|
00000270  00 02 00 48 01 25 00 34 00 2e 00 34 00 64 00 2d  |...H.%.4...4.d.-|
00000280  00 25 00 32 00 2e 00 32 00 64 00 2d 00 25 00 32  |.%.2...2.d.-.%.2|
00000290  00 2e 00 32 00 64 00 20 00 25 00 32 00 2e 00 32  |...2.d. .%.2...2|
000002a0  00 64 00 3a 00 25 00 32 00 2e 00 32 00 64 00 3a  |.d.:.%.2...2.d.:|
000002b0  00 25 00 32 00 2e 00 32 00 64 00 3a 00 02 00 07  |.%.2...2.d.:....|
000002c0  01 53 68 65 6c 6c 54 32 02 00 08 01 54 65 6c 6e  |.ShellT2....Teln|
000002d0  65 74 54 32 00 00 0a 00 78 79 01 00 0b 00 0a 00  |etT2....xy......|
```

Patterns: `90 02 XX`

- The bytes in place of the pattern `90 02 10` and are highlighted in violet
- The entire sub-rule 2 is highlighted in red



```
00000000  04 00 8d 8d 3c ff ff ff 51 56 c7 85 3c ff ff ff  |....<ÿÿÿQVÇ.<ÿÿÿ|
00000010  56 69 72 74 b3 6c c7 85 40 ff ff ff 75 61 6c 41  |Virt³lÇ.@ÿÿÿualA|
00000020  c7 85 44 ff ff ff 6c 6c 6f 63 c6 85 48 ff ff ff  |Ç.DÿÿÿllocÆ.Hÿÿÿ|
00000030  00 ff d7 89 45 f8 85 c0 75 0e 5f 5b b8 04 00 00  |.ÿ×.Eø.Àu._[¸...|
00000040  00 5e 8b e5 5d c2 04 00 8d 95 1c ff ff ff 52 56  |.^.å]Â.....ÿÿÿRV|
00000050  c7 85 1c ff ff ff 56 69 72 74 66 c7 85 20 ff ff  |Ç..ÿÿÿVirtfÇ. ÿÿ|
00000060  ff 75 61 88 9d 22 ff ff ff c7 85 23 ff ff ff 46  |ÿua.."ÿÿÿÇ.#ÿÿÿF|
00000070  72 65 65 c6 85 27 ff ff ff 00 ff d7 89 45 a8 85  |reeÆ.'ÿÿÿ.ÿ×.E¨.|
00000080  c0 75 0e 5f 5b b8 05 00 00 00 5e 8b e5 5d c2 04  |Àu._[¸....^.å]Â.|
00000090  00 8d 85 fc fe ff ff 50 56 c7 85 fc fe ff ff 45  |...üþÿÿPVÇ.üþÿÿE|
000000a0  78 69 74 c7 85 00 ff ff ff 54 68 72 65 66 c7 85  |xitÇ..ÿÿÿThrefÇ.|
000000b0  04 ff ff ff 61 64 c6 85 06 ff ff ff 00 ff d7 85  |.ÿÿÿadÆ..ÿÿÿ.ÿ×.|
000000c0  c0 75 0e 5f 5b b8 06 00 00 00 5e 8b e5 5d c2 04  |Àu._[¸....^.å]Â.|
```



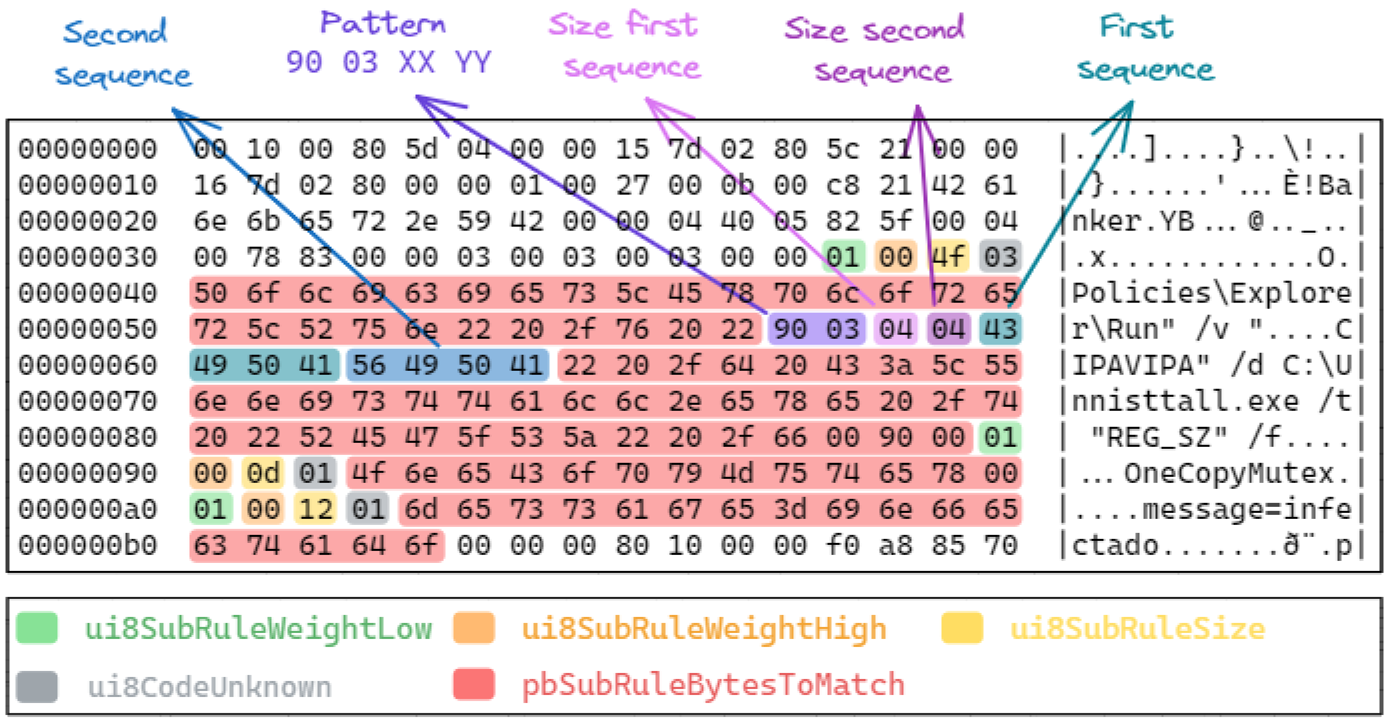
```
PS C:\Program Files\Windows Defender> .\MpCmdRun.exe -Scan -ScanType 3 -File
'C:\Users\user\deeac56026f3804968348c8afa5b7aba10900aeabee05751c0fcac2b88cff71e' -DisableRemediation -
Trace -Level 0x10
Scan starting...
Scan finished.
Scanning C:\Users\user\deeac56026f3804968348c8afa5b7aba10900aeabee05751c0fcac2b88cff71e found 1 threats.

<===========================LIST OF DETECTED THREATS==========================>
----------------------------- Threat information ------------------------------
Threat                    : Backdoor:Win32/Plugx.A
Resources                 : 1 total
    file                  : C:\Users\user\deeac56026f3804968348c8afa5b7aba10900aeabee05751c0fcac2b88cff71e
-------------------------------------------------------------------------------
```

Patterns: 90 03 XX YY

Pattern 90 03 XX YY:

- ❑ XX : the length of the first sequence (<u>Sequence_A)</u> of bytes following the pattern in pink
- ❑ YY : the length of the second sequence (<u>Sequence_B</u>) of bytes following the pattern in grape

❑ In the matching sample either <u>Sequence_A</u> or <u>Sequence_B</u> may appear



```
00000000  00 10 00 80 5d 04 00 00 15 7d 02 80 5c 21 00 00  |....]....}..\!..|
00000010  16 7d 02 80 00 00 01 00 27 00 0b 00 c8 21 42 61  |.}......'...È!Ba|
00000020  6e 6b 65 72 2e 59 42 00 00 04 40 05 82 5f 00 04  |nker.YB...@.._..|
00000030  00 78 83 00 00 03 00 03 00 03 00 00 01 00 4f 03  |.x............O.|
00000040  50 6f 6c 69 63 69 65 73 5c 45 78 70 6c 6f 72 65  |Policies\Explore|
00000050  72 5c 52 75 6e 22 20 2f 76 20 22 90 03 04 04 43  |r\Run" /v "....C|
00000060  49 50 41 56 49 50 41 22 20 2f 64 20 43 3a 5c 55  |IPAVIPA" /d C:\U|
00000070  6e 6e 69 73 74 74 61 6c 6c 2e 65 78 65 20 2f 74  |nnisttall.exe /t|
00000080  20 22 52 45 47 5f 53 5a 22 20 2f 66 00 90 00 01  | "REG_SZ" /f....|
00000090  00 0d 01 4f 6e 65 43 6f 70 79 4d 75 74 65 78 00  |...OneCopyMutex.|
000000a0  01 00 12 01 6d 65 73 73 61 67 65 3d 69 6e 66 65  |....message=infe|
000000b0  63 74 61 64 6f 00 00 00 80 10 00 00 f0 a8 85 70  |ctado.......ð¨.p|
```

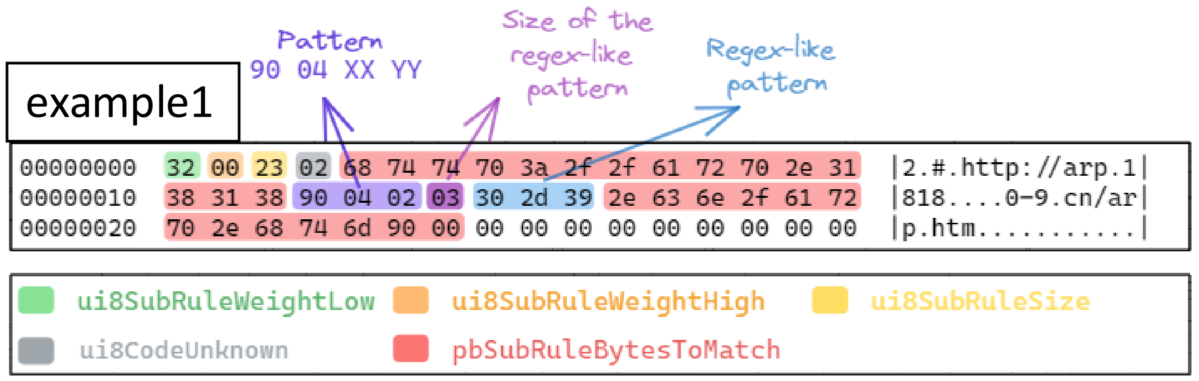
🟩 ui8SubRuleWeightLow 🟧 ui8SubRuleWeightHigh 🟨 ui8SubRuleSize
⬜ ui8CodeUnknown 🟥 pbSubRuleBytesToMatch

```
rule BankerYB_Sub_Rule1_Example{
    strings:
        $sub_rule_1_hex = { 50 6f 6c 69 63 69 65 73 5c 45 78 70 6c 6f 72 65 72 5c 52 75 6e 22 20 2f 76 20 22
                            (43 49 50 41|56 49 50 41) 22 20 2f 64 20 43 3a 5c 55 6e 6e 69 73 74 74 61 6c 6c 2e
                            65 78 65 20 2f 74 20 22 52 45 47 5f 53 5a 22 20 2f 66 00 90 00
        }
    condition:      $sub_rule_1_hex
}
```

Patterns: 90 04 XX YY

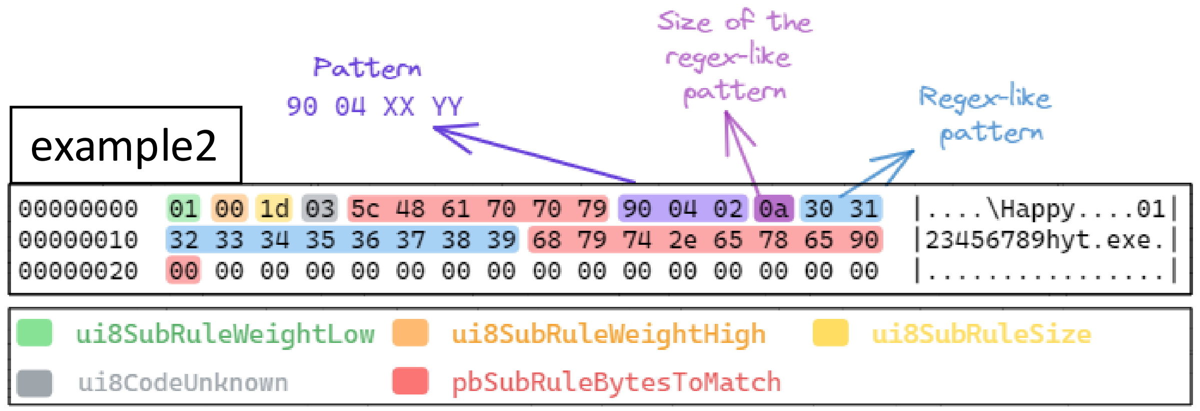
- ❑ XX : the length of the expected bytes
- ❑ YY : the length of the regex-like pattern in the figure highlighted in violet
- ❑ The bytes following 90 04 XX YY describes the pattern itself, in a regex-like fashion:
  - ❑ In this example the bytes are 30 2d 39, highlighted in blue which is 0-9

```
rule Pattern-90-04-example {
   strings:

      $example1_90_04 =
{ 68 74 74 70 3a 2f 2f 61 72 70 2e 31 38 31 38 [30-39] [30-
39] 2e 63 6e 2f 61 72 70 2e 68 74 6d 90 00 }

      $example2_90_04 =
{ 5c 48 61 70 70 79 [30-39] [30-39] 68 79 74 2e 65 78 65 90
00 }

   condition:
      $ example1_90_04 and $ example2_90_04
}
```



```
00000000  32 00 23 02 68 74 74 70 3a 2f 2f 61 72 70 2e 31  |2.#.http://arp.1|
00000010  38 31 38 90 04 02 03 30 2d 39 2e 63 6e 2f 61 72  |818....0-9.cn/ar|
00000020  70 2e 68 74 6d 90 00 00 00 00 00 00 00 00 00 00  |p.htm...........|
```

🟩 ui8SubRuleWeightLow 🟧 ui8SubRuleWeightHigh 🟨 ui8SubRuleSize
⬜ ui8CodeUnknown 🟥 pbSubRuleBytesToMatch

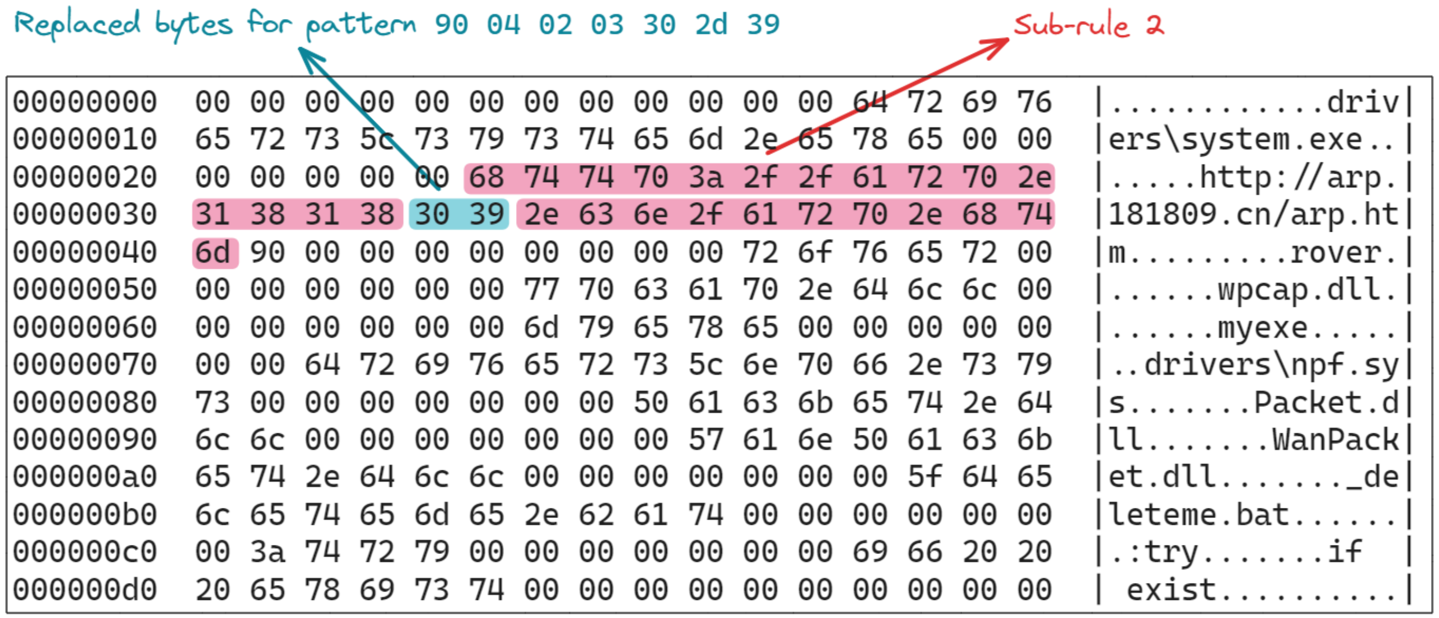


```
00000000  01 00 1d 03 5c 48 61 70 70 79 90 04 02 0a 30 31  |....\Happy....01|
00000010  32 33 34 35 36 37 38 39 68 79 74 2e 65 78 65 90  |23456789hyt.exe.|
00000020  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  |................|
```

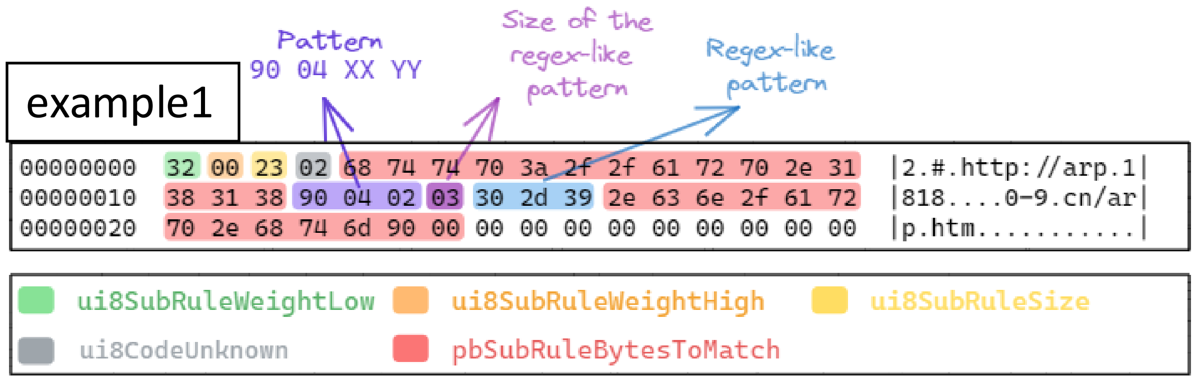
🟩 ui8SubRuleWeightLow 🟧 ui8SubRuleWeightHigh 🟨 ui8SubRuleSize
⬜ ui8CodeUnknown 🟥 pbSubRuleBytesToMatch

Patterns: 90 04 XX YY

- The bytes replacing the pattern 90 04 02 03 30 2D 39 (example1) are:
  - 0x30
  - 0x39
  - Highlighted in cyan
- In red the bytes matching the sub-rule



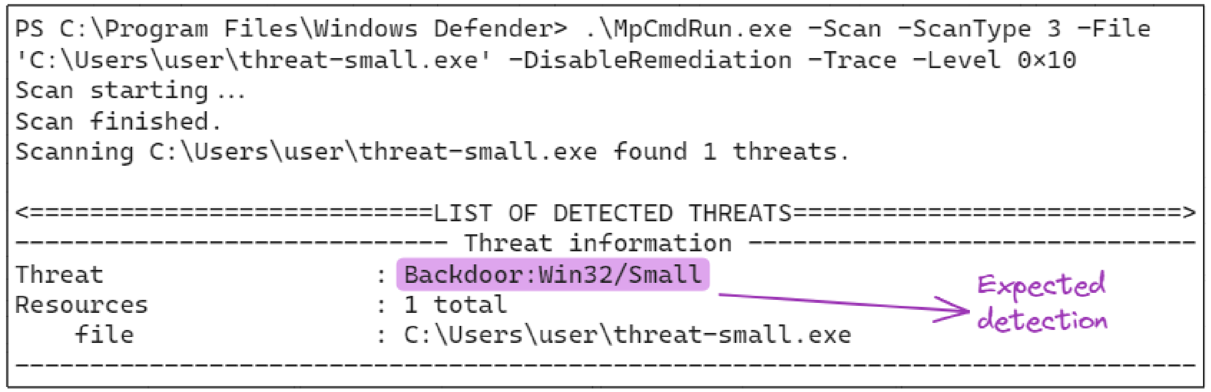
```
00000000   00 00 00 00 00 00 00 00 00 00 00 00 64 72 69 76  |............driv|
00000010   65 72 73 5c 73 79 73 74 65 6d 2e 65 78 65 00 00  |ers\system.exe..|
00000020   00 00 00 00 00 68 74 74 70 3a 2f 2f 61 72 70 2e  |.....http://arp.|
00000030   31 38 31 38 30 39 2e 63 6e 2f 61 72 70 2e 68 74  |181809.cn/arp.ht|
00000040   6d 90 00 00 00 00 00 00 00 00 72 6f 76 65 72 00  |m.........rover.|
00000050   00 00 00 00 00 00 77 70 63 61 70 2e 64 6c 6c 00  |......wpcap.dll.|
00000060   00 00 00 00 00 00 6d 79 65 78 65 00 00 00 00 00  |......myexe.....|
00000070   00 00 64 72 69 76 65 72 73 5c 6e 70 66 2e 73 79  |..drivers\npf.sy|
00000080   73 00 00 00 00 00 00 50 61 63 6b 65 74 2e 64     |s.......Packet.d|
00000090   6c 6c 00 00 00 00 00 00 00 57 61 6e 50 61 63 6b  |ll.......WanPack|
000000a0   65 74 2e 64 6c 6c 00 00 00 00 00 00 00 5f 64 65  |et.dll......._de|
000000b0   6c 65 74 65 6d 65 2e 62 61 74 00 00 00 00 00 00  |leteme.bat......|
000000c0   00 3a 74 72 79 00 00 00 00 00 00 00 69 66 20 20  |.:try.......if  |
000000d0   20 65 78 69 73 74 00 00 00 00 00 00 00 00 00 00  | exist..........|
```



example1

```
00000000   32 00 23 02 68 74 74 70 3a 2f 2f 61 72 70 2e 31  |2.#.http://arp.1|
00000010   38 31 38 90 04 02 03 30 2d 39 2e 63 6e 2f 61 72  |818....0-9.cn/ar|
00000020   70 2e 68 74 6d 90 00 00 00 00 00 00 00 00 00 00  |p.htm...........|
```

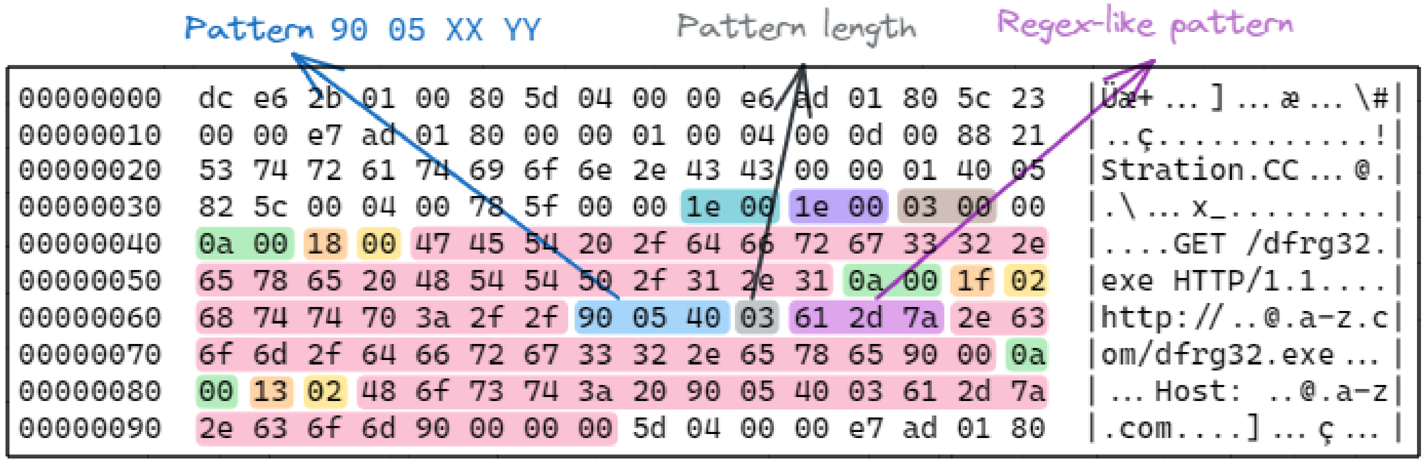
🟩 ui8SubRuleWeightLow 🟧 ui8SubRuleWeightHigh 🟨 ui8SubRuleSize
⬜ ui8CodeUnknown 🟥 pbSubRuleBytesToMatch



```
PS C:\Program Files\Windows Defender> .\MpCmdRun.exe -Scan -ScanType 3 -File
'C:\Users\user\threat-small.exe' -DisableRemediation -Trace -Level 0x10
Scan starting...
Scan finished.
Scanning C:\Users\user\threat-small.exe found 1 threats.

<===========================LIST OF DETECTED THREATS==========================>
---------------------------- Threat information ------------------------------
Threat                 : Backdoor:Win32/Small
Resources              : 1 total
    file               : C:\Users\user\threat-small.exe
------------------------------------------------------------------------------
```

Patterns: 90 05 XX YY

- XX : the max length of the expected bytes
- YY : the length of the regex-like pattern in the figure highlighted in grape
- Differently from pattern 04, this pattrns is case insensitive
- The bytes following 90 05 XX YY describes the pattern itself, in a regex-like format



```
00000000  dc e6 2b 01 00 80 5d 04 00 00 e6 ad 01 80 5c 23  |Üa+ ... ] ... æ ... \#|
00000010  00 00 e7 ad 01 80 00 00 01 00 04 00 0d 00 88 21  |..ç............!|
00000020  53 74 72 61 74 69 6f 6e 2e 43 43 00 00 01 40 05  |Stration.CC ... @.|
00000030  82 5c 00 04 00 78 5f 00 00 1e 00 1e 00 03 00 00  |.\ ... x_.........|
00000040  0a 00 18 00 47 45 54 20 2f 64 66 72 67 33 32 2e  |....GET /dfrg32.|
00000050  65 78 65 20 48 54 54 50 2f 31 2e 31 0a 00 1f 02  |exe HTTP/1.1....|
00000060  68 74 74 70 3a 2f 2f 90 05 40 03 61 2d 7a 2e 63  |http://..@.a-z.c|
00000070  6f 6d 2f 64 66 72 67 33 32 2e 65 78 65 90 00 0a  |om/dfrg32.exe ...|
00000080  00 13 02 48 6f 73 74 3a 20 90 05 40 03 61 2d 7a  |... Host:  ..@.a-z|
00000090  2e 63 6f 6d 90 00 00 00 5d 04 00 00 e7 ad 01 80  |.com....] ... ç ...|
```

```
rule Pattern-90-05-example{
      strings:

         $example_90_05 =
         "http://[a-zA-Z]{0,64}\\.com/dfrg32\\.exe“

      condition:
            $example_90_05

}
```

Lab3: Match the detection

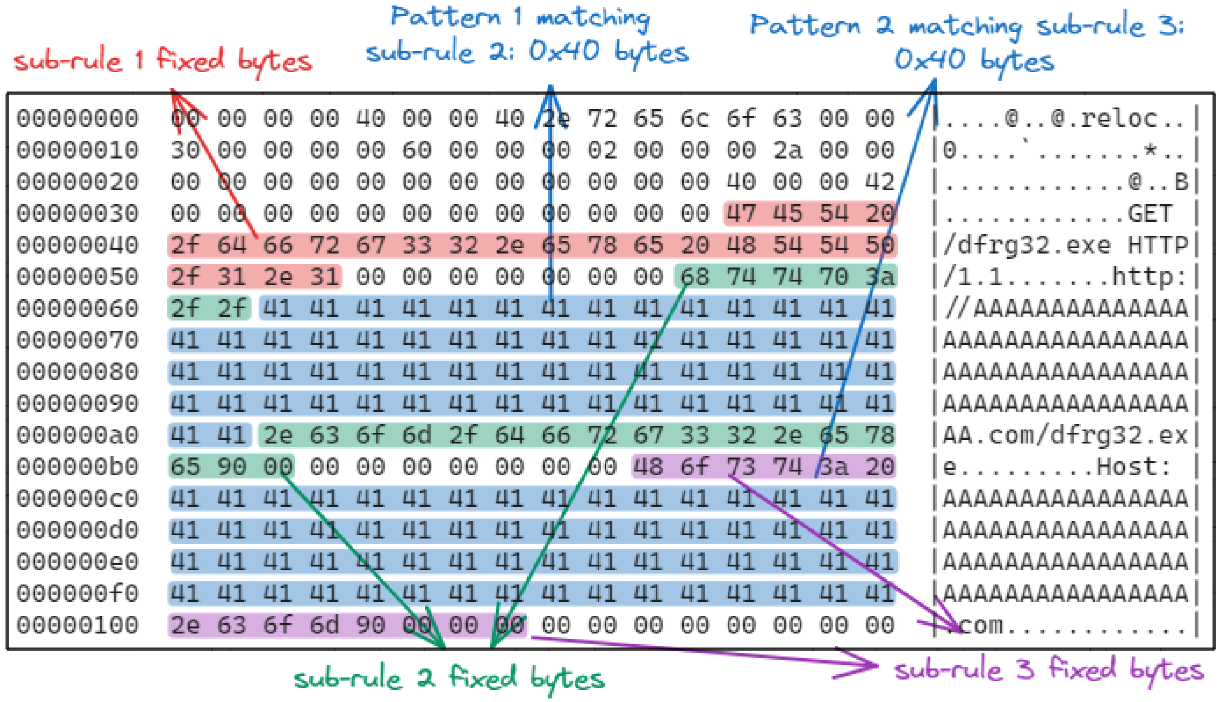
- ❑ Open `msys64` folder and run `msys64.exe`
- ❑ Change the current folder to the root of the lab using the following command

```
cd /c/<your_path>/lab3_stration/Exercise
```

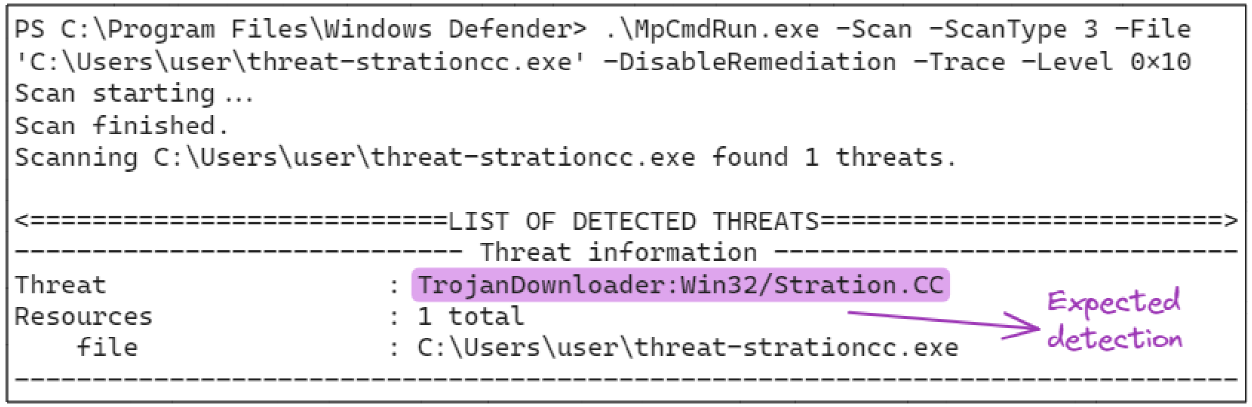
- ❑ Analyze the Stration.CC PEHSTR signature
  - ❑ Understand weights and wildcards
- ❑ Modify the provided `StrationCC.c` file in such a way that once it is compiled, matches the Stration.CC detection
- ❑ To compile use the `build.sh` script

Solution: Patterns: `90 05 XX YY`

- The bytes replacing the pattern `90 05 40 03 61 2D 7A` in sub-rule 2 are highlighted in blue
- In red fixed bytes of sub-rule 2
- Expected detection: `Stration.CC`


sub-rule 1 fixed bytes
Pattern 1 matching sub-rule 2: 0x40 bytes
Pattern 2 matching sub-rule 3: 0x40 bytes
sub-rule 2 fixed bytes
sub-rule 3 fixed bytes

```
00000000  00 00 00 00 40 00 00 40 2e 72 65 6c 6f 63 00 00  |....@..@.reloc..|
00000010  30 00 00 00 00 60 00 00 00 02 00 00 00 2a 00 00  |0....`.......*..|
00000020  00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 42  |............@..B|
00000030  00 00 00 00 00 00 00 00 00 00 00 00 47 45 54 20  |............GET |
00000040  2f 64 66 72 67 33 32 2e 65 78 65 20 48 54 54 50  |/dfrg32.exe HTTP|
00000050  2f 31 2e 31 00 00 00 00 00 00 00 68 74 74 70 3a  |/1.1.......http:|
00000060  2f 2f 41 41 41 41 41 41 41 41 41 41 41 41 41 41  |//AAAAAAAAAAAAAA|
00000070  41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
00000080  41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
00000090  41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
000000a0  41 41 2e 63 6f 6d 2f 64 66 72 67 33 32 2e 65 78  |AA.com/dfrg32.ex|
000000b0  65 90 00 00 00 00 00 00 00 00 48 6f 73 74 3a 20  |e.........Host: |
000000c0  41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
000000d0  41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
000000e0  41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
000000f0  41 41 41 41 41 41 41 41 41 41 41 41 41 41 41 41  |AAAAAAAAAAAAAAAA|
00000100  2e 63 6f 6d 90 00 00 00 00 00 00 00 00 00 00 00  |.com............|
```



```
PS C:\Program Files\Windows Defender> .\MpCmdRun.exe -Scan -ScanType 3 -File
'C:\Users\user\threat-strationcc.exe' -DisableRemediation -Trace -Level 0x10
Scan starting...
Scan finished.
Scanning C:\Users\user\threat-strationcc.exe found 1 threats.

<==========================LIST OF DETECTED THREATS==========================>
----------------------------- Threat information ------------------------------
Threat                  : TrojanDownloader:Win32/Stration.CC
Resources               : 1 total
    file                : C:\Users\user\threat-strationcc.exe
-------------------------------------------------------------------------------
```

Final lab

- ❑ GOAL: implement a working example that triggers the Defender signature `Backdoor:Win64/Havoc.A!MTB`

1. Open the extracted signature database and find the signature
   - ❑ Understand the type of signature
   - ❑ Understand what the signature bytes represents
2. Decompile the provided sample in `lab4_havoc\Exercise\sample.zip` (it is a real MALWARE, so handle with care. PWD: infected)
   - ❑ Identify and analyze the function that triggers the detection
3. Modify the `lab4_havoc\Exercise\havoc_emu_asm.S` to include the same implementation present within the provided sample for the `XorAlgorithm`
4. To compile use the `build.sh` script

https://retooling.io/blog

That's All Folks

silvio@retooling.io
antonio@retooling.io

Reference

- https://www.safebreach.com/blog/defender-pretender-when-windows-defender-updates-become-a-security-risk/
- https://gist.githubusercontent.com/mattifestation/3af5a472e11b7e135273e71cb5fed866/raw/15be4f2ae75b2d62465cf9faef72a2f61147a393/ExpandDefenderSig.ps1
- https://learn.microsoft.com/en-us/defender-endpoint/command-line-arguments-microsoft-defender-antivirus