Finding the Needle

A Study of the PE32 Rich Header (and Respective Malware Triage)

George D. Webster, Bojan Kolosnjaji, Christian von Pentz,
Julian Kirsch*, Zachary D. Hanif, Apostolis Zarras, Claudia Eckert

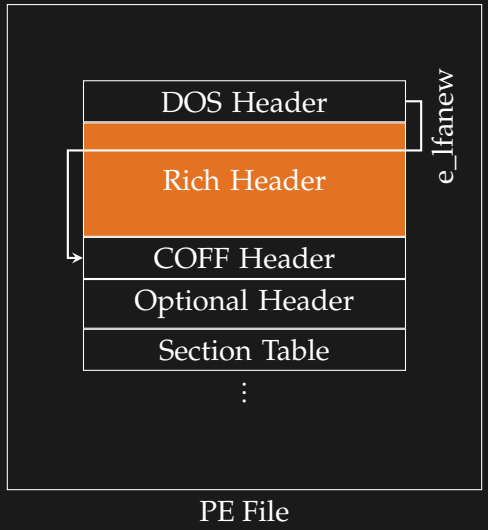
* presenting author

July 6, 2017

Overview

What this Talk is About

Rich header: **Undocumented region** of the PE file format



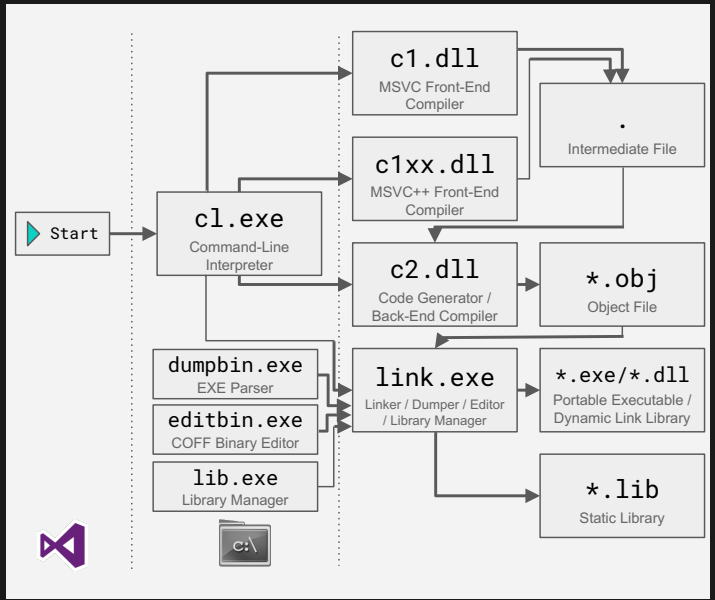
Technical University of Munich

- Not covered by academic literature

- Header present since at least Visual Studio 6 (Microsoft, 1998)
- First public mentioning (Lifewire, 2004)
- First (partial) interpretation of the header's contents (Daniel Pistelli, 2008)
- Complete understanding (Our Work, 2016)

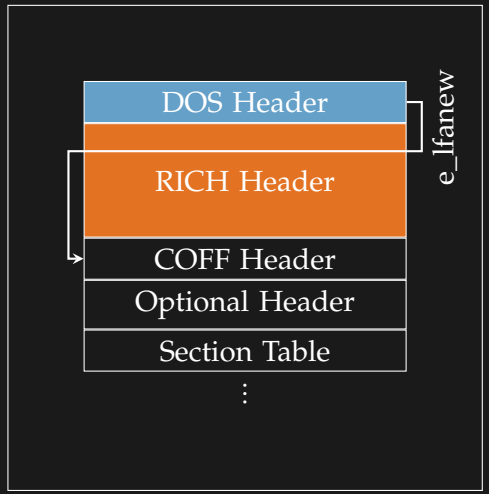
Background

The Microsoft Visual Studio Toolchain



The Rich Header

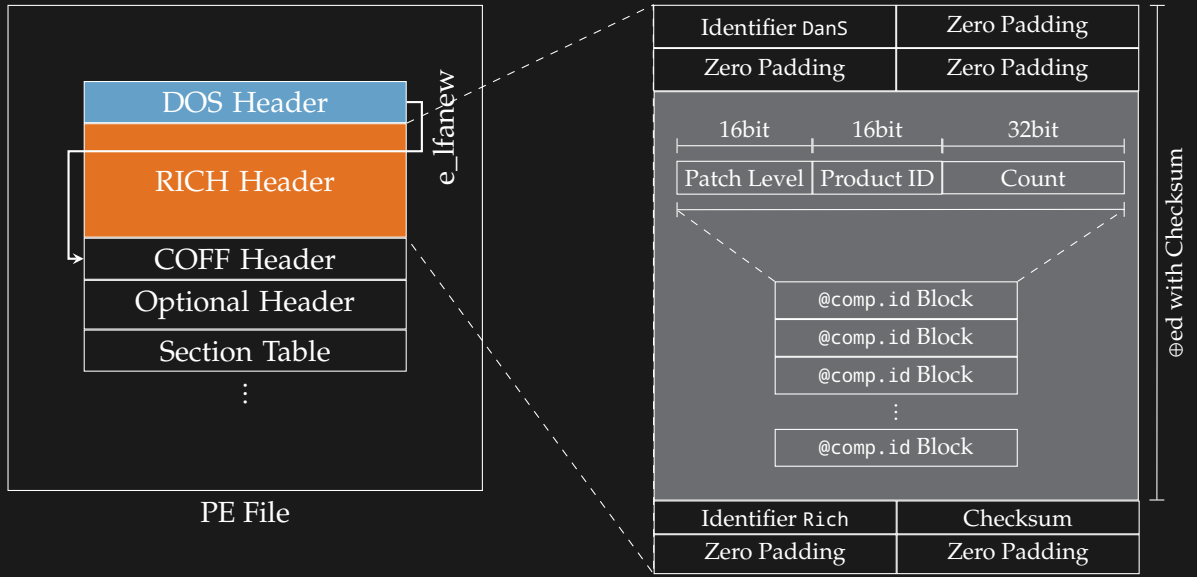
Overview



- DOS header:
  - Ensures backwards compatibility: DOS program printing `This program cannot be run in DOS mode`
  - Fully documented by Microsoft
  - Program can be replaced by any MS-DOS application (using MSVC's `/STUB` compiler flag)
- **RICH Header**:
  - No consistent explanation available
  - Never officially mentioned by Microsoft
  - Added to any Portable Executable created using the Microsoft linker
- Followed by COFF and Optional header

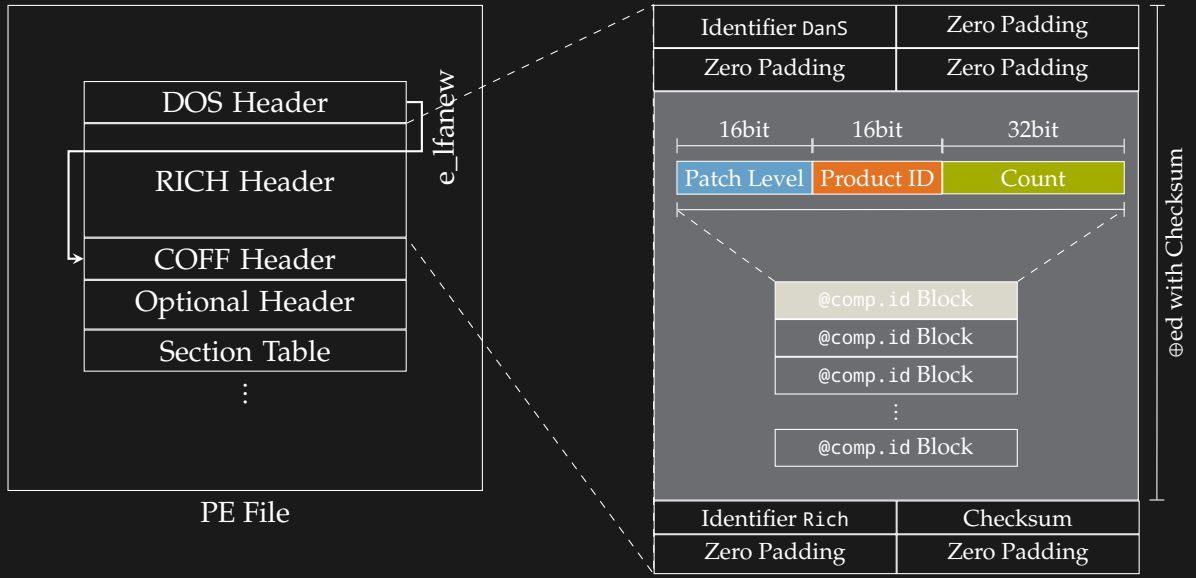
The Rich Header

Internals



The Rich Header

Internals



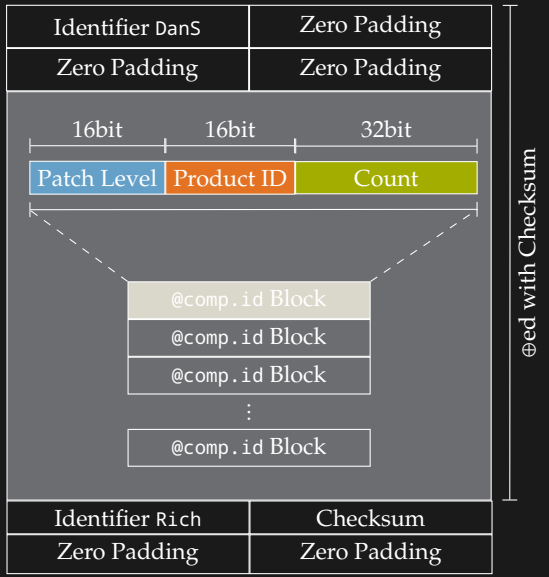
The Rich Header

Internals

Technical University of Munich

Array of 64bit `@comp.id` Blocks storing:

- Patch Level: Patch level of the MSVS tool used to create the object file
- Product ID: Unique ID indicating the type of the object file (before compilation!)
- Count: Number of symbols used from objects with the respective Patch Level / Product ID combination



The Rich Header

Example

Technical University of Munich

```c
#include <stdio.h>

int main(int argc, char **argv)
{
    puts("Hello World!");
}
```

The Rich Header

Example

Technical University of Munich

```
00000000: 4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 |MZ..............|
00000010: b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 |........@.......|
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|
00000030: 00 00 00 00 00 00 00 00 00 00 00 00 f0 00 00 00 |................|
00000040: 0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68 |........!..L.!Th|
00000050: 69 73 20 70 72 6f 67 72 61 6d 20 63 61 6e 6e 6f |is program canno|
00000060: 74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20 |t be run in DOS |
00000070: 6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00 00 |mode....$.......|
00000080: f1 bc 37 8c b5 dd 59 df b5 dd 59 df b5 dd 59 df |..7...Y...Y...Y.|
00000090: 01 41 a8 df bc dd 59 df 01 41 aa df cc dd 59 df |.A....Y..A....Y.|
000000a0: 01 41 ab df ad dd 59 df 01 41 b6 df b0 dd 59 df |.A....Y..A....Y.|
000000b0: b5 dd 58 df e0 dd 59 df d6 80 5a de a4 dd 59 df |..X...Y...Z...Y.|
000000c0: d6 80 5c de ab dd 59 df d6 80 5d de a4 dd 59 df |..\...Y...]...Y.|
000000d0: db 80 5d de b4 dd 59 df db 80 5b de b4 dd 59 df |..]...Y...[...Y.|
000000e0: 52 69 63 68 b5 dd 59 df 00 00 00 00 00 00 00 00 |Rich..Y.........|
000000f0: 50 45 00 00 4c 01 05 00 24 4c 2f 58 00 00 00 00 |PE..L...$L/X....|
                                  ⋮
```

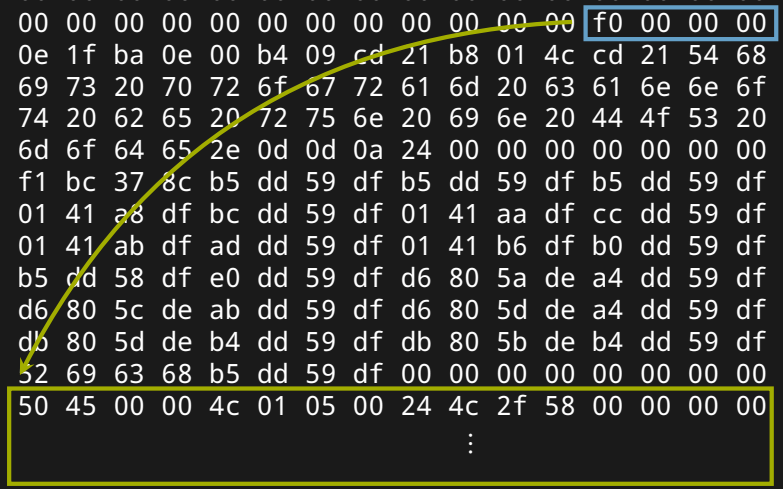
The Rich Header

Example


```
00000000: 4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 |MZ..............|
00000010: b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 |........@.......|
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|
00000030: 00 00 00 00 00 00 00 00 00 00 00 00 f0 00 00 00 |................|
00000040: 0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68 |........!..L.!Th|
00000050: 69 73 20 70 72 6f 67 72 61 6d 20 63 61 6e 6e 6f |is program canno|
00000060: 74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20 |t be run in DOS |
00000070: 6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00 00 |mode....$.......|
00000080: f1 bc 37 8c b5 dd 59 df b5 dd 59 df b5 dd 59 df |..7...Y...Y...Y.|
00000090: 01 41 a8 df bc dd 59 df 01 41 aa df cc dd 59 df |.A....Y..A....Y.|
000000a0: 01 41 ab df ad dd 59 df 01 41 b6 df b0 dd 59 df |.A....Y..A....Y.|
000000b0: b5 dd 58 df e0 dd 59 df d6 80 5a de a4 dd 59 df |..X...Y...Z...Y.|
000000c0: d6 80 5c de ab dd 59 df d6 80 5d de a4 dd 59 df |..\...Y...]...Y.|
000000d0: db 80 5d de b4 dd 59 df db 80 5b de b4 dd 59 df |..]...Y...[...Y.|
000000e0: 52 69 63 68 b5 dd 59 df 00 00 00 00 00 00 00 00 |Rich..Y.........|
000000f0: 50 45 00 00 4c 01 05 00 24 4c 2f 58 00 00 00 00 |PE..L...$L/X....|
                                  ⋮
```

The Rich Header

Example



The Rich Header

Example

Technical University of Munich

```
00000000: 4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 |MZ..............|
00000010: b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00 |........@.......|
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |................|
00000030: 00 00 00 00 00 00 00 00 00 00 00 00 f0 00 00 00 |................|
00000040: 0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68 |........!..L.!Th|
00000050: 69 73 20 70 72 6f 67 72 61 6d 20 63 61 6e 6e 6f |is program canno|
00000060: 74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20 |t be run in DOS |
00000070: 6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00 00 |mode....$.......|
00000080: f1 bc 37 8c b5 dd 59 df b5 dd 59 df b5 dd 59 df |..7...Y...Y...Y.|
00000090: 01 41 a8 df bc dd 59 df 01 41 aa df cc dd 59 df |.A....Y..A....Y.|
000000a0: 01 41 ab df ad dd 59 df 01 41 b6 df b0 dd 59 df |.A....Y..A....Y.|
000000b0: b5 dd 58 df e0 dd 59 df d6 80 5a de a4 dd 59 df |..X...Y...Z...Y.|
000000c0: d6 80 5c de ab dd 59 df d6 80 5d de a4 dd 59 df |..\...Y...]...Y.|
000000d0: db 80 5d de b4 dd 59 df db 80 5b de b4 dd 59 df |..]...Y...[...Y.|
000000e0: 52 69 63 68 b5 dd 59 df 00 00 00 00 00 00 00 00 |Rich..Y.........|
000000f0: 50 45 00 00 4c 01 05 00 24 4c 2f 58 00 00 00 00 |PE..L...$L/X....|
                                  ⋮
```

The Rich Header

Example

Technical University of Munich

```
00000080: f1 bc 37 8c b5 dd 59 df b5 dd 59 df b5 dd 59 df |..7...Y...Y...Y.|
00000090: 01 41 a8 df bc dd 59 df 01 41 aa df cc dd 59 df |.A....Y..A....Y.|
000000a0: 01 41 ab df ad dd 59 df 01 41 b6 df b0 dd 59 df |.A....Y..A....Y.|
000000b0: b5 dd 58 df e0 dd 59 df d6 80 5a de a4 dd 59 df |..X...Y...Z...Y.|
000000c0: d6 80 5c de ab dd 59 df d6 80 5d de a4 dd 59 df |..\...Y...]...Y.|
000000d0: db 80 5d de b4 dd 59 df db 80 5b de b4 dd 59 df |..]...Y...[...Y.|
000000e0: 52 69 63 68 b5 dd 59 df 00 00 00 00 00 00 00 00 |Rich..Y.........|
```

The Rich Header

Example

Technical University of Munich

```
00000080: f1 bc 37 8c b5 dd 59 df b5 dd 59 df b5 dd 59 df |..7...Y...Y...Y.|
00000090: 01 41 a8 df bc dd 59 df 01 41 aa df cc dd 59 df |.A....Y..A....Y.|
000000a0: 01 41 ab df ad dd 59 df 01 41 b6 df b0 dd 59 df |.A....Y..A....Y.|
000000b0: b5 dd 58 df e0 dd 59 df d6 80 5a de a4 dd 59 df |..X...Y...Z...Y.|
000000c0: d6 80 5c de ab dd 59 df d6 80 5d de a4 dd 59 df |..\...Y...]...Y.|
000000d0: db 80 5d de b4 dd 59 df db 80 5b de b4 dd 59 df |..]...Y...[...Y.|
000000e0: 52 69 63 68 b5 dd 59 df 00 00 00 00 00 00 00 00 |Rich..Y.........|
```

The Rich Header

Example

Technical University of Munich

```
00000080: 44 61 6e 53 00 00 00 00 00 00 00 00 00 00 00 00 |DanS............|
00000090: b4 9c f1 00 09 00 00 00 b4 9c f3 00 79 00 00 00 |............y...|
000000a0: b4 9c f2 00 18 00 00 00 b4 9c ef 00 05 00 00 00 |................|
000000b0: 00 00 01 00 55 00 00 00 63 5d 03 01 11 00 00 00 |....U...c]......|
000000c0: 63 5d 05 01 1e 00 00 00 63 5d 04 01 11 00 00 00 |c]......c]......|
000000d0: 6e 5d 04 01 01 00 00 00 6e 5d 02 01 01 00 00 00 |n]......n]......|
000000e0: 52 69 63 68 b5 dd 59 df 00 00 00 00 00 00 00 00 |Rich..Y.........|
```

The Rich Header

Example

Technical University of Munich

```
00000080: 44 61 6e 53 00 00 00 00 00 00 00 00 00 00 00 00 |DanS............|
00000090: b4 9c f1 00 09 00 00 00 b4 9c f3 00 79 00 00 00 |............y...|
000000a0: b4 9c f2 00 18 00 00 00 b4 9c ef 00 05 00 00 00 |................|
000000b0: 00 00 01 00 55 00 00 00 63 5d 03 01 11 00 00 00 |....U...c]......|
000000c0: 63 5d 05 01 1e 00 00 00 63 5d 04 01 11 00 00 00 |c]......c]......|
000000d0: 6e 5d 04 01 01 00 00 00 6e 5d 02 01 01 00 00 00 |n]......n]......|
000000e0: 52 69 63 68 b5 dd 59 df 00 00 00 00 00 00 00 00 |Rich..Y.........|
```

Compiler Patchlevel	Product ID	Count
40116	0x00f1	0x00000009
40116	0x00f3	0x00000079
40116	0x00f2	0x00000018
40116	0x00ef	0x00000005
0	0x0001	0x00000055
23907	0x0103	0x00000011
23907	0x0105	0x0000001e
⋮	⋮	⋮

The Rich Header

Example


```
00000080: 44 61 6e 53 00 00 00 00 00 00 00 00 00 00 00 00 |DanS............|
00000090: b4 9c f1 00 09 00 00 00 b4 9c f3 00 79 00 00 00 |............y...|
000000a0: b4 9c f2 00 18 00 00 00 b4 9c ef 00 05 00 00 00 |................|
000000b0: 00 00 01 00 55 00 00 00 63 5d 03 01 11 00 00 00 |....U...c]......|
000000c0: 63 5d 05 01 1e 00 00 00 63 5d 04 01 11 00 00 00 |c]......c]......|
000000d0: 6e 5d 04 01 01 00 00 00 6e 5d 02 01 01 00 00 00 |n]......n]......|
000000e0: 52 69 63 68 b5 dd 59 df 00 00 00 00 00 00 00 00 |Rich..Y.........|
```

Compiler Patchlevel	Product ID	Count
40116	0x00f1	0x00000009
40116	0x00f3	0x00000079
40116	0x00f2	0x00000018
40116	0x00ef	0x00000005
0	0x0001	0x00000055
23907	0x0103	0x00000011
23907	0x0105	0x0000001e
⋮	⋮	⋮

The Rich Header

Example

Compiler Patchlevel	Product ID	Count	MS Internal Name	Visual Studio Release
40116	0x00f1	0x00000009	prodidMasm1210	Visual Studio 2013 (12.10)
40116	0x00f3	0x00000079	prodidUtc1810_CPP	Visual Studio 2013 (12.10)
40116	0x00f2	0x00000018	prodidUtc1810_C	Visual Studio 2013 (12.10)
40116	0x00ef	0x00000005	prodidImplib1210	Visual Studio 2013 (12.10)
0	0x0001	0x00000055	prodidImport0	Visual Studio (00.00)
23907	0x0103	0x00000011	prodidMasm1400	Visual Studio 2015 (14.00)
23907	0x0105	0x0000001e	prodidUtc1900_CPP	Visual Studio 2015 (14.00)
23907	0x0104	0x00000011	prodidUtc1900_C	Visual Studio 2015 (14.00)
23918	0x0104	0x00000001	prodidUtc1900_C	Visual Studio 2015 (14.00)
23918	0x0102	0x00000001	prodidLinker1400	Visual Studio 2015 (14.00)

⇒ `HelloWorld` (C) compiled using `c2.dll` version `14.00.23918` (VS2015)

⇒ Statically linked C standard library from `c2.dll` version `12.10.40116` (VS2013)

The Rich Header

Example

Compiler Patchlevel	Product ID	Count	MS Internal Name	Visual Studio Release
40116	0x00f1	0x00000009	prodidMasm1210	Visual Studio 2013 (12.10)
40116	0x00f3	0x00000079	prodidUtc1810_CPP	Visual Studio 2013 (12.10)
40116	0x00f2	0x00000018	prodidUtc1810_C	Visual Studio 2013 (12.10)
40116	0x00ef	0x00000005	prodidImplib1210	Visual Studio 2013 (12.10)
0	0x0001	0x00000055	prodidImport0	Visual Studio (00.00)
23907	0x0103	0x00000011	prodidMasm1400	Visual Studio 2015 (14.00)
23907	0x0105	0x0000001e	prodidUtc1900_CPP	Visual Studio 2015 (14.00)
23907	0x0104	0x00000011	prodidUtc1900_C	Visual Studio 2015 (14.00)
23918	0x0105	0x00000001	prodidUtc1900_C	Visual Studio 2015 (14.00)
23918	0x0102	0x00000001	prodidLinker1400	Visual Studio 2015 (14.00)

⇒ `HelloWorld` (C++) compiled using `cl.exe` version `14.00.23918` (VS2015)

⇒ Statically linked C standard library from `cl.exe` version `12.10.40116` (VS2013)

The Rich Header

Product IDs

Recent versions of Visual Studio are capable of producing **18** different product IDs per major release. The following are the **most important** ones for VS2015 (see last slide for full list):

Product ID	Meaning
`0x00ff`	Resource File
`0x0101`	DLL Import Stub
`0x0102`	The Linker's Own Entry (appended for PE files)
`0x0103`	ASM Object
`0x0104`	C Object
`0x0105`	C++ Object
`0x0108`	(Statically) LTCG Optimized C-Object
`0x0109`	(Statically) LTCG Optimized C++-Object
`0x010b`	(Dynamically) LTCG Instrumented C-Object
`0x010c`	(Dynamically) LTCG Instrumented C++-Object
`0x010d`	(Dynamically) LTCG Optimized C-Object
`0x010e`	(Dynamically) LTCG Optimized C++-Object

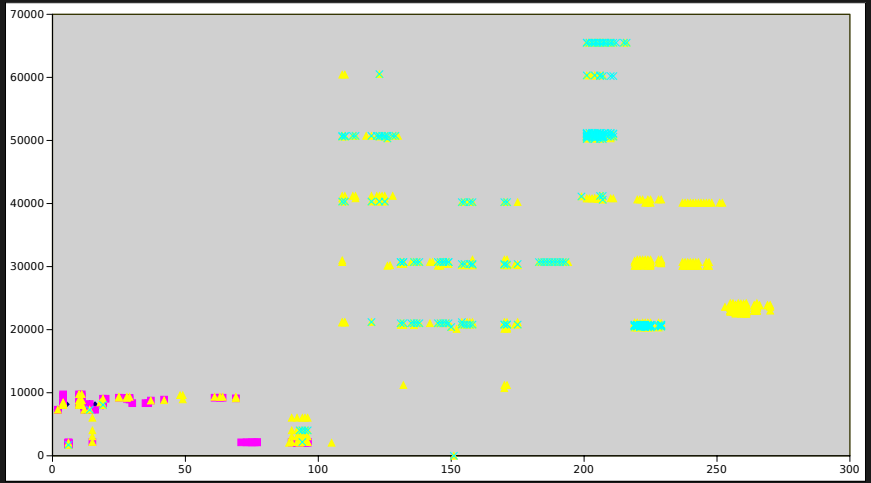
(LTCG = Link Time Code Generation)

The Rich Header

Product IDs

X-Axis: ProdID, Y-Axis: Compiler Patchlevel

Win 98 (Dark Blue) Win 2000 (Purple) Win 7 (Light Blue) Win 8.1 (Yellow)



The Rich Header

The Checksum Value

Technical University of Munich

```python
## Rotate left helper function
rol32 = lambda v, n: ((v << (n & 0x1f)) & 0xffffffff) | \
                     (v >> (32 - (n & 0x1f)))

## raw_dat is a bytearray containing the exe's data
## compids is the list of deciphered @compid structs
## off is the offset to the start of the Rich Header
def calc_csum(raw_dat, compids, off):
    csum = off
    for i in range(off):
        ## skip e_lfanew as it's not initialized yet
        if i in range(0x3c, 0x40):
            continue
      csum += rol32(raw_dat[i], i)

    for c in compids:
        csum += rol32((c['prodid'] << 16) | c['patchlvl'], c['count'])

    return csum & 0xffffffff
```

Technical University of Munich

Observation:

$$m \equiv n \mod 32 \implies \text{rol32}(v, n) = \text{rol32}(v, m)$$
$$\implies \text{rol32}(x, 0x5) == \text{rol32}(x, 0x25) == \text{rol32}(x, 0x345765)$$

⇒ Only the **lowest 5 bits** of the 32bit `count` field **matter** for the checksum value!
⇒ Only 64 – (32 – 5) = 37 bits per `@comp.id` are checksummed.

The Rich Header

Prevalence in Malware

Family	Total	Rich Header	Percent
Random Set	964 816	683 238	71%
APT1	292	286	98%
Zeus-Citadel	1928	717	37%
Mediyes	1873	30	2%

- Rich header allows fingerprinting build environments
- Similar programs built in the same environment lead to similar Rich header
- Details in the paper

Summary & Future Work

Takeaway:

All (non-modified) binaries generated by MSVC include information about type and number of pre-compile-time objects.

Research projects in which the Rich header might be useful:

- Improve static **signatures** to identify binary functions
- Fingerprint Microsoft's compiler **build chain**
- ...?

Contact

Technical University of Munich

- mail@kirschju.re
  PGP: F949 CFBD 140A 6DD0 71E9 0B8C DC24 396B 6D45 1038

- **Sources** available – documentation pending :-)
  → My Rich Header Toolchain:
  https://github.com/kirschju/richheader
  → Raw Data & Large Scale Processing Toolchain:
  https://holmesprocessing.github.io
  → Slides:
  https://kirschju.re/static/2017-dimva-rich.pdf

Thanks!