Reverse Engineering

5. Compiler Stub Recognition

Ing. Tomáš Zahradnický, EUR ING, Ph.D.
Ing. Martin Jirkal

FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Czech Technical University in Prague
Faculty of Information Technology
Department of Computer Systems

The update of these materials was supported by the University Development Fund of the Czech Ministry of Education, Youth and Sports under project No. 341/2013/B1a.

Version 2017-11-18

Table of Contents

Motivation

Question

Why would we want to identify the compiler which was used to compile an executable?

Answer

To reduce the amount of code we have to analyze. Known compiler stubs can be excluded from further analysis.

The same question can be asked for all 3rd party libraries the executable is linked with. If we can identify them, we can exclude them from the analysis.
Furthermore, the knowledge of the functions in these libraries can be used to extend our understanding of the application by giving types and meaning to variables which are passed to them.

Introduction

- Every compiler uses its own style during the compilation (e.g. data and code locations are specific to the compiler).
- Every executable links to specific runtime libraries, either statically or dynamically. Identification of runtime library code can tell us a lot about the compiler.
- To identify which compiler was used to compile a binary, we need to know where, inside the binary, to look for important pieces of information.
- The binary is usually large so knowledge of what to skip speeds-up the analysis significantly.

Compiler Recognition — Microsoft Visual C++

Marketing version	Internal version	CL.EXE version	Possible imported DLL	Release date
6	6.0	12.00	msvcrt.dll, msvcp60.dll	June 1998
.NET	7.0	13.00	msvcr70.dll, msvcp70.dll	February 13, 2002
.NET 2003	7.1	13.10	msvcr71.dll, msvcp71.dll	April 24, 2003
2005	8.0	14.00	msvcr80.dll, msvcp80.dll	November 7, 2005
2008	9.0	15.00	msvcr90.dll, msvcp90.dll	November 19, 2007
2010	10.0	16.00	msvcr100.dll, msvcp100.dll	April 12, 2010
2012	11.0	17.00	msvcr110.dll, msvcp110.dll	September 12, 2012
2013	12.0	18.00	msvcr120.dll, msvcp120.dll	October 17, 2013
2015	14.0	19.00	vcruntime140.dll	July 20, 2015
2017	14.1+	19.10	vcruntime140.dll	March 7, 2017

- MSVC compiler can be recognized by searching the import directory for the libraries above.
- MSVC-decorated symbol names usually start with a `?` sign.

```
?doSomething@CMFCApplicationView@@QAEXXZ
```

Compiler Recognition — Borland

- May import the `BORLNDMM.DLL` library.
- Mangled names start with the `@` symbol.

```
@TModule@ValidWindow$qp14TWindowsObject
```

- Delphi contains data type names at the beginning of `CODE` segment. Strings like `Boolean`, `Integer`, `TObject`, `Char` etc. seen near the start of the file make Delphi identification quite fast and easy.

Compiler Recognition — A Delphi Example

```
00400 0410 4000 0307 426F 6F6C 6561 6E01 0000   ..@...Boolean...
00410 0000 0100 0000 0010 4000 0546 616C 7365   ........@..False
00420 0454 7275 658D 4000 2C10 4000 0107 496E   .TrueŤ@.,.@...In
00430 7465 6765 7204 0000 0080 FFFF FF7F 8BC0   teger....€˙˙˙▯‹Ŕ
00440 4410 4000 0104 4279 7465 0100 0000 00FF   D.@...Byte.....˙
00450 0000 0090 5810 4000 0104 576F 7264 0300   ... X.@...Word..
00460 0000 00FF FF00 0090 6C10 4000 0108 4361   ...˙˙.. l.@...Ca
00470 7264 696E 616C 0500 0000 00FF FFFF FF90   rdinal.....˙˙˙˙
00480 8410 4000 0A06 5374 7269 6E67 9010 4000   „.@...String .@.
00490 0C07 5661 7269 616E 748D 4000 E810 4000   ..VariantŤ@.č.@.
004A0 0000 0000 0000 0000 0000 0000 0000 0000   ................
004B0 0000 0000 0000 0000 0000 0000 E810 4000   ............č.@.
004C0 0400 0000 0000 0000 AC3A 4000 B83A 4000   ........¬:@.¸:@.
004D0 BC3A 4000 C03A 4000 B43A 4000 1438 4000   Ľ:@.Ŕ:@.´:@..8@.
004E0 3038 4000 6C38 4000 0754 4F62 6A65 6374   08@.l8@..TObject
004F0 F410 4000 0707 544F 626A 6563 74E8 1040   ô.@...TObjectč.@
00500 0000 0000 0000 0006 5379 7374 656D 0000   ........System..
00510 1411 4000 0F0A 4949 6E74 6572 6661 6365   ..@...IInterface
00520 0000 0000 0100 0000 0000 0000 00C0 0000   .............Ŕ..
00530 0000 0000 4606 5379 7374 656D 0300 FFFF   ....F.System..˙˙
00540 CC83 4424 04F8 E931 5200 0083 4424 04F8   Ě D$.řé1R.. D$.ř
00550 E94F 5200 0083 4424 04F8 E959 5200 00CC   éOR.. D$.řéYR..Ě
00560 4111 4000 4B11 4000 5511 4000 0100 0000   A.@.K.@.U.@.....
```

Compiler Recognition — Other

- GCC
  - `cygwin1.dll` is imported if Cygwin was used for compilation.
  - `msvcrt.dll` is imported if MinGW was used for compilation.[1]
  - Mangled names usually start with `_Z`.

```
_Z1hv
```

- Watcom
  - Mangled names usually start with `W`.

```
W?method$_class$n__v
```

- FORTRAN
  - Can import `libifcoremd.dll`, `libifportmd.dll`, `libiomp5md.dll`.

[1] Note that non-MinGW applications also frequently link against `msvcrt.dll`.

Startup, Runtime, and Library Code

- As explained in Lecture 2, an executable starts execution at the main entry point. That location is far away from the `main()` function.
- The startup code is heavily dependent on the compiler, compiler options and the runtime library.
- It is **mostly** uninteresting for analysis and analyzing it would be a waste of time.
- For this reason it is useful to mark the prologue code as "library code", as it is of no interest to the reverse engineer.
- We can do the same for libraries we find in the program.

Identifying Libraries

- With compiler identified, we can try to find as many library routines as possible and exclude them from the analysis.
- This is done by performing pattern matching against signatures generated for each library.
- We search the executable for known signatures and notice all matched functions. These are:
  - renamed to the name of the matched function;
  - marked down as library code.
- IDA Pro uses an approach called F.L.I.R.T. (Fast Library Identification and Recognition Technology) [1], which does just this.

Creating Signatures

- For each function in a library we have source code for can be described with a pattern.
- The pattern could be the first X bytes from the function start in the machine code, or the entire function code up to the `ret` instruction.
  - Note that the exact version of the library as well as its build variant, such as Debug or Release, is important here.
- Though we can set X to an arbitrary value or make a pattern out of the entire function, functions with the same body and different names cannot be distinguished one from the other (e.g. `htonl` and `ntohl`). In that case we choose only 1 of the functions and discard the other.
- Patterns for the whole library are then combined into a signature file.
- IDA Pro provides a FLAIR SDK for this purpose.

Libraries, how to find them? I

- How can we find that an executable uses a particular library?
- If the library is dynamically linked, it is easy:
    - Windows: `dumpbin`, CFF Explorer, Dependency Walker.
    - Linux: `objdump`, `readelf`, or just `ldd`.
    - OS X: `otool` or `dyldinfo`.
- If the library is statically linked, use `strings` to find:
    - copyright statements,
    - version statements,
    - error messages.

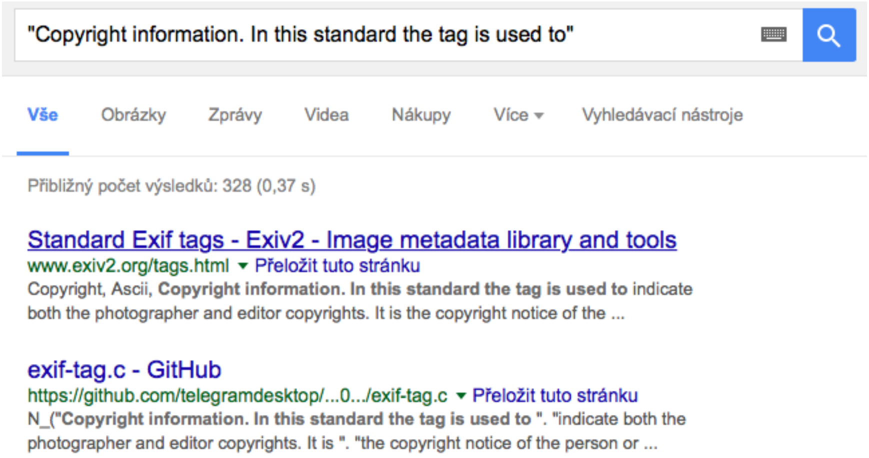
  Then paste the discovered statements, quoted, into your favorite search engine!
    - This will, usually, not find the exact version, but will provide a good initial approximation.

Libraries, how to find them? II

```
mail:MacOS admin$ strings -a MediaManager
...
Unknown Exif Version
Exif Version %d.%d
0100
FlashPix Version 1.0
0101
FlashPix Version 1.01
...
Copyright information. In this standard the tag is used to indicate both the...
...
This tag is used to record the name of an audio file related to the image...
...
```

Figure : Running the `strings` tool on the executable to discover any valuable library identification strings.

Libraries, how to find them? III


"Copyright information. In this standard the tag is used to"

Přibližný počet výsledků: 328 (0,37 s)

Standard Exif tags - Exiv2 - Image metadata library and tools
www.exiv2.org/tags.html ▾ Přeložit tuto stránku
Copyright, Ascii, **Copyright information. In this standard the tag is used to** indicate both the photographer and editor copyrights. It is the copyright notice of the ...

exif-tag.c - GitHub
https://github.com/telegramdesktop/...0.../exif-tag.c ▾ Přeložit tuto stránku
N_(**"Copyright information. In this standard the tag is used to** ". "indicate both the photographer and editor copyrights. It is ". "the copyright notice of the person or ...

Figure : An identified library — libexif.

Bibliography

Chris Eagle: *The Ida Pro Book 2nd Edition*, no starch press, 2011.

Eldad Eilam: *Reversing: Secrets of Reverse Engineering*, Wiley Publishing, Inc., 2005.

Reverend Bill Blunden: *The Rootkit Arsenal*, Wordware Publishing, Inc., 2009.