PE 101 — a windows executable walk-through

PORTABLE EXECUTABLE

V2.01, 5TH DECEMBER 2012
CREATIVE COMMONS 3.0 BY

ANGE ALBERTINI
CORKAMI.COM

DISSECTED PE

SHA-1 B7AF4CB5FCE38E43E0306566E826F8FAB408CF9CB
DOWNLOAD @ PE101.CORKAMI.COM

SIMPLE.EXE

HEADER

TECHNICAL DETAILS ABOUT THE EXECUTABLE

DOS HEADER

SHOWS IT'S A BINARY

HEXADECIMAL DUMP	ASCII DUMP	FIELDS	VALUES	EXPLANATION
4D 5A 00 00-00 00 00 00-00 00 00 00-00 00 00 00	MZ..............			
Offset:0x30 00 00 00 00-00 00 00 00-00 00 00 00-40 00 00 00	............@...	e_magic	'MZ'	CONSTANT SIGNATURE
		e_lfanew	0x40	OFFSET OF THE PE HEADER 1

PE HEADER

SHOWS IT'S A MODERN BINARY

HEXADECIMAL DUMP	ASCII DUMP	FIELDS	VALUES	EXPLANATION
Offset:0x40 50 45 00 00-4C 01 03 00-00 00 00 00-00 00 00 00	PE..L...........	Signature	'PE', 0, 0	CONSTANT SIGNATURE
		Machine	0x14c [intel 386]	PROCESSOR: ARM/MIPS/INTEL/...
		NumberOfSections	3	NUMBER OF SECTIONS 2
00 00 00 00-E0 00 02 01...	....a...	SizeOfOptionalHeader	0xe0	RELATIVE OFFSET OF THE SECTION TABLE 2
		Characteristics	0x102 [32b EXE]	EXE/DLL/...

OPTIONAL HEADER

EXECUTABLE INFORMATION

HEXADECIMAL DUMP	ASCII DUMP	FIELDS	VALUES	EXPLANATION
Offset:0x58 ...0B 01 00 00-00 00 00 00	....	Magic	0x10b [32b]	32 BITS/64 BITS
00 00 00 00-00 00 00 00-00 10 00 00-00 00 00 00	....	AddressOfEntryPoint	0x1000	WHERE EXECUTION STARTS 5
00 00 00 00-00 00 40 00-00 10 00 00-00 02 00 00	......@.........	ImageBase	0x400000	ADDRESS WHERE THE FILE SHOULD BE MAPPED IN MEMORY 3
00 00 00 00-00 00 00 00-04 00 00 00-00 00 00 00	....	SectionAlignment	0x1000	WHERE SECTIONS SHOULD START IN MEMORY 2
00 40 00 00-00 02 00 00-00 00 00 00-02 00 00 00	.@..............	FileAlignment	0x200	WHERE SECTIONS SHOULD START ON FILE 2
00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00	....	MajorSubsystemVersion	4 [NT 4 or later]	REQUIRED VERSION OF WINDOWS
00 00 00 00-10 00 00 00...	....	SizeOfImage	0x4000	TOTAL MEMORY SPACE REQUIRED
		SizeOfHeaders	0x200	TOTAL SIZE OF THE HEADERS 3
		Subsystem	2 [GUI]	DRIVER/GRAPHICAL/COMMAND LINE/...
		NumberOfRvaAndSizes	16	NUMBER OF DATA DIRECTORIES 4

DATA DIRECTORIES

POINTERS TO EXTRA STRUCTURES (EXPORTS, IMPORTS,...)

HEXADECIMAL DUMP	ASCII DUMP	FIELDS	VALUES	EXPLANATION
...00 00 00 00-00 00 00 00	....			
00 20 00 00-00 00 00 00-00 00 00 00-00 00 00 00	....	ImportsVA	0x2000	RVA* OF THE IMPORTS 4
00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00	....			

SECTIONS TABLE

Offset:0x138

HEXADECIMAL DUMP	ASCII DUMP
2E 74 65 78-74 00 00 00	.text...
00 10 00 00-00 10 00 00-00 02 00 00-00 02 00 00	....
00 00 00 00-00 00 00 00-00 00 00 00-20 00 00 60	....`
2E 72 64 61-74 61 00 00-00 10 00 00-00 20 00 00	.rdata.........
00 02 00 00-00 04 00 00-00 00 00 00-00 00 00 00	....
00 00 00 00-40 00 00 40-2E 64 61 74-61 00 00 00	....@..@.data...
00 10 00 00-00 30 00 00-00 02 00 00-00 06 00 00	.....0..........
00 00 00 00-00 00 00 00-00 00 00 00-40 00 00 C0	............@..+

SECTIONS TABLE

NAME	VIRTUALSIZE	RVA*	VIRTUALADDRESS	RVA*	SIZEOFRAWDATA	PHYSICAL SIZE	POINTERTORAWDATA	PHYSICAL OFFSET	CHARACTERISTICS
.text	0x1000		0x1000		0x200		0x200		CODE EXECUTE READ
.rdata	0x1000		0x2000		0x200		0x400		INITIALIZED READ
.data	0x1000		0x3000		0x200		0x600		DATA READ WRITE

FOR EACH SECTION, A SIZEOFRAWDATA SIZED BLOCK IS READ FROM THE FILE AT POINTERTORAWDATA OFFSET.
IT WILL BE LOADED IN MEMORY AT ADDRESS IMAGEBASE + VIRTUALADDRESS IN A VIRTUALSIZE SIZED BLOCK, WITH SPECIFIC CHARACTERISTICS.

SECTIONS

CONTENTS OF THE EXECUTABLE

CODE

WHAT IS EXECUTED

Offset:0x200/RVA:0x401000

```
6A 00 68 00-30 40 00 68-17 30 40 00-6A 00 FF 15  j.h.0@.h.0@.j. .
70 20 40 00-6A 00 FF 15-68 20 40 00              p.@.j. .h.@.
```

X86 ASSEMBLY

```
push 0
push 0x403000
push 0x403017
push 0
call [0x402070]
push 0
call [0x402068]
```

EQUIVALENT C CODE

```
MessageBox(0, "Hello World!","a simple PE executable", 0);
ExitProcess(0);
```

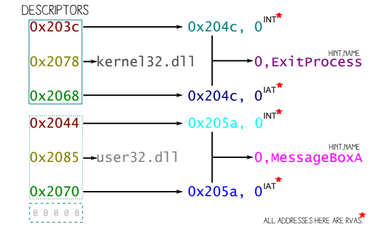
IMPORTS

LINK BETWEEN THE EXECUTABLE AND (WINDOWS) LIBRARIES

Offset:0x400/RVA:0x402000

```
3C 20 00 00-00 00 00 00-00 00 00 00-78 20 00 00  <...........x...
68 20 00 00-44 20 00 00-00 00 00 00-00 00 00 00  h...D...........
85 20 00 00-70 20 00 00-00 00 00 00-00 00 00 00  à...p...........
00 00 00 00-00 00 00 00-00 00 00 00-4C 20 00 00  ............L...
00 00 00 00-5A 20 00 00-00 00 00 00-00 00 45 78  ....Z.........Ex
69 74 50 72-6F 63 65 73-73 00 00 00-4D 65 73 73  itProcess...Mess
61 67 65 42-6F 78 41 00-4C 20 00 00-00 00 00 00  ageBoxA.L.......
5A 20 00 00-00 00 00 00-6B 65 72 6E-65 6C 33 32  Z.......kernel32
2E 64 6C 6C-00 75 73 65-72 33 32 2E-64 6C 6C 00  .dll.user32.dll.
```

IMPORTS STRUCTURES



CONSEQUENCES

AFTER LOADING,
0X402068 WILL POINT TO KERNEL32.DLL'S EXITPROCESS
0X402070 WILL POINT TO USER32.DLL'S MESSAGEBOXA

DATA

INFORMATION USED BY THE CODE

Offset:0x600/RVA:0x403000

```
61 20 73 69-6D 70 6C 65-20 50 45 20-65 78 65 63  a.simple.PE.exec
75 74 61 62-6C 65 00 48-65 6C 6C 6F-20 77 6F 72  utable.Hello.wor
6C 64 21 00                                      ld!.
```

STRINGS

```
a simple PE executable\0
Hello world!\0
```

THIS IS THE WHOLE FILE. HOWEVER, MOST PE FILES CONTAIN MORE ELEMENTS.
EXPLANATIONS ARE SIMPLIFIED, FOR CONCISENESS.

LOADING PROCESS

1 HEADERS

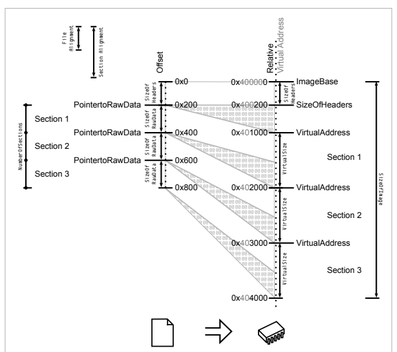
THE *DOS HEADER* IS PARSED
THE *PE HEADER* IS PARSED
(ITS OFFSET IS *DOS HEADER*'S E_LFANEW)
THE *OPTIONAL HEADER* IS PARSED
(IT FOLLOWS THE *PE HEADER*)

2 SECTIONS TABLE

SECTIONS TABLE IS PARSED
(IT IS LOCATED AT: OFFSET (*OPTIONALHEADER*) + *SIZEOFOPTIONALHEADER*)
IT CONTAINS *NUMBEROFSECTIONS* ELEMENTS
IT IS CHECKED FOR VALIDITY WITH ALIGNMENTS:
FILEALIGNMENTS AND *SECTIONALIGNMENTS*

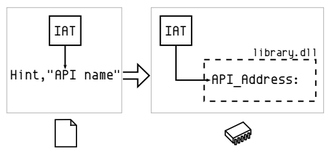
3 MAPPING

THE FILE IS MAPPED IN MEMORY ACCORDING TO:
THE *IMAGEBASE*
THE *SIZEOFHEADERS*
THE SECTIONS TABLE



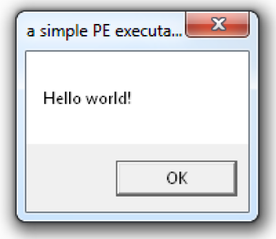
4 IMPORTS

DATADIRECTORIES ARE PARSED
THEY FOLLOW THE *OPTIONALHEADER*
THEIR NUMBER IS *NUMOFRVAANDSIZES*
IMPORTS ARE ALWAYS #2
IMPORTS ARE PARSED
EACH DESCRIPTOR SPECIFIES A *DLLNAME*
THIS DLL IS LOADED IN MEMORY
IAT AND *INT* ARE PARSED SIMULTANEOUSLY
FOR EACH API IN *INT*
ITS ADDRESS IS WRITTEN IN THE *IAT* ENTRY



5 EXECUTION

CODE IS CALLED AT THE *ENTRYPOINT*
THE CALLS OF THE CODE GO VIA THE IAT TO THE APIS



NOTES

MZ HEADER *AKA* DOS_HEADER
STARTS WITH 'MZ' (INITIALS OF *MARK ZBIKOWSKI* MS-DOS DEVELOPER)

PE HEADER *AKA* IMAGE_FILE_HEADERS / COFF FILE HEADER
STARTS WITH 'PE' (PORTABLE EXECUTABLE)

OPTIONAL HEADER *AKA* IMAGE_OPTIONAL_HEADER
OPTIONAL ONLY FOR NON-STANDARD PES BUT REQUIRED FOR EXECUTABLES

RVA RELATIVE VIRTUAL ADDRESS
ADDRESS RELATIVE TO IMAGEBASE (AT IMAGEBASE, RVA = 0)
ALMOST ALL ADDRESSES OF THE HEADERS ARE RVAS
IN CODE, ADDRESSES ARE *NOT* RELATIVE.

INT IMPORT NAME TABLE
NULL-TERMINATED LIST OF POINTERS TO HINT, NAME STRUCTURES

IAT IMPORT ADDRESS TABLE
NULL-TERMINATED LIST OF POINTERS
ON FILE IT IS A COPY OF THE INT
AFTER LOADING IT POINTS TO THE IMPORTED APIS

HINT
INDEX IN THE EXPORTS TABLE OF A DLL TO BE IMPORTED
NOT REQUIRED BUT PROVIDES A SPEED-UP BY REDUCING LOOK-UP