

Analyzing the import Address Table (PE View)

Analyzing the import Address Table (PE View)

PE : Portable Executable

It is nothing more than a **gigantic array of bytes**, which will be our **Malware**

In PE view you can see **Pfile(Address)** ,**Raw Data** , **Actual value**.

| pFile | Raw Data | Value |
|----------|---|--------------------|
| 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 |@..... |
| 00000020 | 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 F8 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 cannot |
| 00000060 | 74 20 62 65 20 72 75 6E 20 69 6E 20 44 4F 53 20 | be run in DOS |
| 00000070 | 6D 6F 64 65 2E 0D 0D 0A 24 00 00 00 00 00 00 | mode....\$. |
| 00000080 | F4 70 F6 21 B0 11 98 72 B0 11 98 72 B0 11 98 72 | .p.!...r...r...r |
| 00000090 | B9 69 0B 72 BA 11 98 72 0E 60 9D 73 A4 11 98 72 | .i...r...s...r |
| 000000A0 | 0E 60 9C 73 BC 11 98 72 0E 60 9B 73 B5 11 98 72 | .s...r...s...r |
| 000000B0 | 0E 60 99 73 B4 11 98 72 EB 79 99 73 B9 11 98 72 | .s...r...y...s...r |
| 000000C0 | B0 11 99 72 8C 11 98 72 26 63 91 73 B1 11 98 72 | ...r...r&c...s...r |
| 000000D0 | 26 63 67 72 B1 11 98 72 26 63 9A 73 B1 11 98 72 | &cgr...r&c...s...r |
| 000000E0 | 52 69 63 68 B0 11 98 72 00 00 00 00 00 00 00 | Rich...r...f...r |
| 000000F0 | 00 00 00 00 00 00 00 00 50 45 00 00 4C 01 05 00 |PE..L... |
| 00000100 | C0 B6 33 61 00 00 00 00 00 00 00 00 E0 00 02 01 | ..3a..... |
| 00000110 | 0B 01 0E 1C 00 16 00 00 00 18 00 00 00 00 00 | |
| 00000120 | F1 15 00 00 00 10 00 00 00 30 00 00 00 00 40 00 |0....@... |
| 00000130 | 00 10 00 00 00 02 00 00 06 00 00 00 00 00 00 | |
| 00000140 | 06 00 00 00 00 00 00 00 00 70 00 00 00 04 00 |p..... |
| 00000150 | 00 00 00 00 03 00 40 81 00 00 10 00 00 10 00 |@..... |
| 00000160 | 00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 | |
| 00000170 | 00 00 00 00 00 00 00 00 34 38 00 00 F0 00 00 |48..... |
| 00000180 | 00 50 00 00 E0 01 00 00 00 00 00 00 00 00 00 | .P..... |
| 00000190 | 00 00 00 00 00 00 00 00 00 60 00 00 B8 01 00 | |
| 000001A0 | 98 33 00 00 70 00 00 00 00 00 00 00 00 00 00 | .3..p..... |
| 000001B0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 000001C0 | 08 34 00 00 40 00 00 00 00 00 00 00 00 00 00 | .4..@..... |
| 000001D0 | 00 30 00 00 FC 00 00 00 00 00 00 00 00 00 00 | .0..... |
| 000001E0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 | |
| 000001F0 | 2E 74 65 78 74 00 00 00 A1 15 00 00 00 10 00 | .text..... |
| 00000200 | 00 16 00 00 00 04 00 00 00 00 00 00 00 00 00 | |
| 00000210 | 00 00 00 00 20 00 00 60 2E 72 64 64 74 64 00 | ...data |

This **MZ** The two starting byte will give you the idea what this file is.

`IMAGE_FILE_HEADER` under `Image section` will have the **time stamp** which can be correct or incorrect.

The malwares compiled with Borland Delphi Compiler Will always have a timestamp of 1992

| Malware.Unknown.exe.malz | pFile | Data | Description | Value |
|--------------------------|----------|----------|-------------------------|-----------------------------|
| IMAGE_DOS_HEADER | 000000FC | 014C | Machine | IMAGE_FILE_MACHINE_I386 |
| IMAGE_DEBUG_TYPE_ | 000000FE | 0005 | Number of Sections | |
| MS-DOS Stub Program | 00000100 | 6133B6C0 | Time Date Stamp | 2021/09/04 Sat 18:11:12 UTC |
| IMAGE_NT_HEADERS | 00000104 | 00000000 | Pointer to Symbol Table | |
| Signature | 00000108 | 00000000 | Number of Symbols | |
| IMAGE_FILE_HEADER | 0000010C | 00E0 | Size of Optional Header | |
| IMAGE_OPTIONAL_H | 0000010E | 0102 | Characteristics | |
| IMAGE_SECTION_HEAD | | 0002 | | IMAGE_FILE_EXECUTABLE_IMAGE |
| IMAGE_SECTION_HEAD | | 0100 | | IMAGE_FILE_32BIT_MACHINE |
| IMAGE_SECTION_HEAD | | | | |
| IMAGE_SECTION_HEAD | | | | |

Now , Take a good look at `IMAGE_FILE_HEADER .text` Here we can see the **virtual size** and **size of raw data** , it will be in Hex ,So use a calculator to compare their size

If **Virtual size is > Size of raw data** then we may be dealing with a `packed binary`.

| Malware.Unknown.exe.malz | pFile | Data | Description | Value |
|-----------------------------|----------|-------------|-------------------------|-----------------------|
| IMAGE_DOS_HEADER | 000001F0 | 2E 74 65 78 | Name | .text |
| IMAGE_DEBUG_TYPE_ | 000001F4 | 74 00 00 00 | | |
| MS-DOS Stub Program | 000001F8 | 000015A1 | Virtual Size | |
| IMAGE_NT_HEADERS | 000001FC | 00001000 | RVA | |
| Signature | 00000200 | 00001600 | Size of Raw Data | |
| IMAGE_FILE_HEADER | 00000204 | 00000400 | Pointer to Raw Data | |
| IMAGE_OPTIONAL_HEADER | 00000208 | 00000000 | Pointer to Relocations | |
| IMAGE_SECTION_HEADER .text | 0000020C | 00000000 | Pointer to Line Numbers | |
| IMAGE_SECTION_HEADER .rdata | 00000210 | 0000 | Number of Relocations | |
| IMAGE_SECTION_HEADER .data | 00000212 | 0000 | Number of Line Numbers | |
| IMAGE_SECTION_HEADER .rsrc | 00000214 | 60000020 | Characteristics | |
| IMAGE_SECTION_HEADER .reloc | | 00000020 | | IMAGE_SCN_CNT_CODE |
| SECTION .text | | 20000000 | | IMAGE_SCN_MEM_EXECUTE |
| SECTION .rdata | | 40000000 | | IMAGE_SCN_MEM_READ |
| IMPORT Address Table | | | | |
| IMAGE_DEBUG_DIRECTORY | | | | |
| IMAGE_LOAD_CONFIG_DIRECTORY | | | | |
| IMAGE_DEBUG_TYPE_CODEVIEW | | | | |
| IMAGE_DEBUG_TYPE | | | | |

One of the Important Info We can get in `IMPORT Address Table` Under `SECTION .rdata`

To Understand this we have to understand **Windows API** .

- Dont forgot procmon,procwatch,proc explore (John.H)
- always open three of them for use
- regshot learn about about
 - To capture wht registry changed and can compare
- cscript.exe run .js file in cmd