

PE Header Walkthrough

0x0000 4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00 MZ@.....

0x0010 b5 b5 00 00 00 00 00 00 00 40 00 00 00 00 00 00@.....

0x0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00@.....

0x0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00@.....

0x0040 00 1f 1f 1f 1f 1f 1f 1f 21 b8 01 4c 2d 15 64 00 e_magic@.....

0x0050 69 73 20 70 7e 67 67 72 61 6d 63 61 6e 6e 6f is.program.canno

0x0060 74 20 62 65 20 72 75 7e 20 6d 63 20 44 4f 53 20 t.be.run.in.DOS.

0x0070 6d 6f 64 65 2e 0d 0a 2e 20 6d 63 20 44 4f 53 20 mode.....\$.

0x0080 a5 6d 16 9b e1 0c 78 c8 e1 0c 78 c8 e1 0c 78 c8 .m.....x.....x.....

0x0090 1b 2f 38 c8 e0 0c 78 c8 e1 0c 78 c8 e1 0c 78 c8 /B.....x.....x.....

0x00a0 1b 2f 61 c8 f2 0c 78 c8 e1 0c 79 c8 f2 0c 78 c8 /a.....x.....#.....

0x00b0 76 2f 3d c8 e0 0c 78 c8 3b 2f 64 c8 23 0c 78 c8 v/.....x...../d.....

0x00c0 1b 2f 45 c8 e0 0c 78 c8 52 69 63 68 e1 0c 78 c8 ./E.....Rich.....

0x00d0 00 00 00 00 00 00 00 00 50 45 00 00 00 00 00 00 PE.....

0x00e0 00 00 00 00 00 00 00 00 00 00 00 00 4c 01 03 00).....

0x00f0 00 84 7d 3b 00 00 00 00 00 00 00 00 00 00 01 01).....n.....

0x0100 e0 01 07 00 00 0e 00 00 a6 00 00 00 00 00 00 01 j.....

0x0110 e0 6a 00 00 00 10 00 00 00 00 00 00 00 00 00 01).....

0x0120 00 10 00 00 00 02 00 00 05 00 01 00 05 00 01 00).....

0x0130 04 00 00 00 00 00 00 00 30 01 00 00 04 00 00).....

0x0140 53 08 01 00 02 00 00 00 30 00 00 00 04 00 00).....

0x0150 00 00 10 00 00 10 00 00 00 00 00 00 10 00 00 U.....

0x0160 00 00 00 00 00 00 00 00 20 6d 00 00 c8 00 00 00 m.....

0x0170 00 a0 00 00 48 89 00 00 00 00 00 00 00 00 00 00 H.....

0x0180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x0190 40 13 00 00 1c 00 00 00 00 00 00 00 00 00 00 00 @.....

0x01a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x01b0 00 00 00 00 00 00 00 00 58 02 00 00 d0 00 00 00

0x01c0 00 10 00 00 24 03 00 00 00 00 00 00 00 00 00 00 \$.

0x01d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

0x01e0 2e 74 65 78 74 00 00 00 72 6d 00 00 10 00 00 00text.....rm.....

0x01f0 00 6e 00 00 00 04 00 00 00 00 00 00 00 00 00 00 n.....

0x0200 00 00 00 00 20 00 00 60 2e 64 61 74 61 00 00 00data.....

0x0210 a8 1b 00 00 00 00 00 00 00 06 00 00 00 72 00 00

0x0220 00 00 00 00 00 00 00 00 00 00 00 00 40 00 c0

0x0230 2e 72 73 72 63 00 00 00 48 89 00 00 a0 00 00 00 rsrc.....H.....

0x0240 00 8a 00 00 00 78 00 00 00 00 00 00 00 00 00 00

0x0250 00 00 00 00 40 00 00 16 fe 7d 3b 58 00 00 00 00

0x0260 0f fe 7d 3b 65 00 00 29 fe 7d 3b 71 00 00 00 00).....;.....X.....

The DOS Header

The **DOS** header can be found starting at offset zero in all *Portable Executable* files. Nowadays its main objective is to indicate the offset of the main headers containing the actual information about the *PE* file, the **NT** headers. The offset where to find those headers is stored in the **e_lfanew** member.

```

C:\Program Files\Foxit Software\Foxit Reader>dumpbin /headers
Dump of file C:\Program Files\Foxit Software\Foxit Reader\Foxit Reader.exe
PE header
Signature:
0x0000 4d 5a 90 00 03 00 00 00 00 04 00 00 ff ff 00 00 MZ.....@.....
b8 00 00 00 00 00 00 00 00 40 00 00 00 00 00 00
0x0020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0030 00 00 00 00 00 00 00 00 00 00 00 00 e8 00 00
0x0040 0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68
0x0050 69 73 10 70 72 6f 67 72 61 6d 60 63 61 6e 6e 6f
0x0060 74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20
0x0070 6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00
0x0080 a5 6d 16 9b e1 0c 78 c8 e1 0c 78 c8 e1 0c 78 c8
0x0090 1b 2f 38 c8 e1 0c 78 c8 e1 0c 78 c8 e1 0c 78 c8
0x00a0 1b 2f 61 c8 f2 0c 78 c8 e1 0c 79 c8 2f 0c 78 c8
0x00b0 7b 2f 38 c8 e0 0c 78 c8 2f 64 c8 f2 0c 78 c8
0x00c0 7b 2f 45 c8 e0 0c 78 c8 52 64 c8 68 e1 0c 78 c8
0x00d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00e0 00 00 00 00 00 00 00 00 e0 45 00 00 4c 01 03 00
0x00f0 8d 74 7b 3b 00 00 00 00 00 00 00 e0 00 01 00
0x0100 0b 01 07 00 00 00 00 00 00 a5 00 00 00 00 00 00
0x0110 e0 6a 00 00 00 00 00 00 00 00 00 00 00 01
0x0120 00 00 00 00 00 00 00 00 00 00 00 00 00 01
0x0130 04 00 00 00 00 00 00 00 00 00 30 01 00 00 04 00
0x0140 55 08 01 00 00 00 00 00 00 00 80 00 04 00 00 00
0x0150 00 00 10 00 00 00 10 00 00 00 00 00 00 10
0x0160 00 00 00 00 00 00 00 00 00 00 20 6d 00 c8 00 00 00
0x0170 00 a0 00 00 48 89 00 00 00 00 00 00 00 00 00 00
0x0180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0190 40 13 00 00 1c 00 00 00 00 00 00 00 00 00 00 00
0x01a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x01b0 00 00 00 00 00 00 00 00 58 02 00 00 d0 00 00 00
0x01c0 00 10 00 00 24 03 00 00 00 00 00 00 00 00 00 00
0x01d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x01e0 2e 74 65 78 74 00 00 00 72 6d 00 00 10 00 00 00
0x01f0 00 6e 00 00 00 04 00 00 00 00 00 00 00 00 00 00
0x0200 00 00 00 00 20 00 00 60 2e 64 61 74 61 00 00 00
0x0210 a8 1b 00 00 00 80 00 00 00 06 00 00 72 00 00 00
0x0220 00 00 00 00 00 00 00 00 00 00 00 00 40 00 c0
0x0230 2e 72 73 72 63 00 00 48 89 00 00 a0 00 00 00
0x0240 00 8a 00 00 00 78 00 00 00 00 00 00 00 00 00
0x0250 00 00 00 00 40 00 00 10 16 7d 3b 58 00 00 00
0x0260 0f fe 7d 3b 65 00 00 29 fe 7d 3b 71 00 00 00
  
```

NT Headers

The **NT** headers contain three members, a signature and two other structures defining the **File** header and the **Optional** header. The signature is the standard doubleword **0x50450000** with *ASCII* representation "**PE**". Some of the important members of the **File** header are **Machine**, specifying the target architecture for which this *PE* file is compiled, and the self-describing **SizeOfOptionalHeader** and **NumberOfSections**.

[illegible]

Optional Header

The **Optional** header member describes elements of the file such as the import and export directories that make possible to locate and link *DLL* libraries (which are *PE* files as well). Other entries provide structural information about the layout of the file, such as the alignment of its sections.

The slight irony behind the name *Optional* (it contains a wealth of critical information about an *EXE* or *DLL* file) comes from the fact that the *PE* format can also describe object files that are not meant to be run or otherwise need any of the information contributed by this header.

VirtualAddress	Size
0x0000	2d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00
0x0010	b8 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0020	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0030	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0040	0e 1f ba 0e 04 b9 cd 21 b8 01 4c cd 21 54 68
0x0050	69 73 20 70 72 6f 67 62 61 6d 20 63 61 6e 6e 6f
0x0060	74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20
0x0070	6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00 00
0x0080	a5 6d 16 9b e1 0c 78 c8 e1 0c 78 c8 e1 0c 78 c8
0x0090	1b 2f 38 c8 e0 0c 78 c8 e1 0c 78 c8 e0 0c 78 c8
0x00a0	1b 2f 61 c8 f2 0c 78 c8 e1 0c 79 c8 23 0c 78 c8
0x00b0	76 2f 3d c8 e0 0c 78 c8 3b 2f 64 c8 f2 0c 78 c8
0x00c0	1b 2f 45 c8 e0 0c 78 c8 52 69 63 68 e1 0c 78 c8
0x00d0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00e0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x00f0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0100	e0 7a 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0110	e0 7a 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x0120	00 10 00 00 00 02 00 00 00 00 00 00 00 00 00 00
0x0130	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Export	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Resource	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Security	00 a0 00 00 48 89 00 00 00 00 00 00 00 00 00 00
Debug	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
GlobalPtr	40 13 00 00 1c 00 00 00 00 00 00 00 00 00 00 00
LoadConfig	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
IAT	00 10 00 00 24 03 00 00 00 00 00 00 00 00 00 00
COM Descriptor	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Import	2e 74 65 78 74 00 00 00 72 6d 00 00 00 10 00 00
Exception	00 6e 00 00 00 04 00 00 00 00 00 00 00 00 00 00
BaseReloc	00 00 00 00 20 00 00 00 60 2e 64 61 74 61 00 00
Copyright	a8 1b 00 00 00 00 00 00 00 00 00 00 00 00 00 00
TLS	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 c0
BoundImport	2e 72 73 72 63 00 00 00 48 89 00 00 a0 00 00
Delayimport	00 8a 00 00 00 00 00 00 00 00 00 00 00 00 00 00
	00 00 00 00 00 00 00 00 16 fe 7d 3b 58 00 00
	0f fe 7d 3b 65 00 00 00 29 fe 7d 3b 71 00 00

The Data Directories

These entries, contained within the **Optional** header, point to a wide selection of miscellaneous information about the file. Imported and exported symbols, debug information, resource information (icon data, version information) and others.

All of these are optional, but few *PE* files go without having a symbol import or export table that would allow them to link to (or have its symbols used by) other *PE* files.

The diagram illustrates the structure of a PE file header. The left side shows a hex dump of the header data, and the right side shows the corresponding ASCII representation. Red arrows and boxes highlight key fields:

- Beginning of Section Headers:** Points to the `e_lfanew` field (offset 0x118).
- SizeOfOptionalHeader:** Points to the `SizeOfOptionalHeader` field (offset 0x119).
- PE_L:** Points to the `PE_L` field (offset 0x11A).

The hex dump shows the standard PE header layout, including the `PE_L` field, `SizeOfOptionalHeader`, and `e_lfanew` field. The ASCII representation shows the `PE_L` field, `SizeOfOptionalHeader`, and `e_lfanew` field.

Locating the Section Headers

The **Section** headers follow immediately after the **Optional** header. The procedure to find their starting offset is to add the value from the **File** header member **SizeOfOptionalHeader** to the starting offset of the **Optional** header. The resulting value will point to the first section header. The number of sections is specified by the field **NumberOfSections** in the **File** header.

[illegible]

The Section Headers

The **Section** headers describe each of the sections making up the file. Sections can contain code (often referred to as text, hence the common section name `'text'`), initialized and uninitialized data, more information describing the *PE* file itself such as resources or any other data the developer wishes to add. There can be an arbitrary number of sections in a *PE* file.

The diagram illustrates the structure of a PE file header, specifically the `IMAGE_DOS_HEADER` and `IMAGE_NT_HEADERS` sections. The header is represented as a table of hexadecimal values, with various fields highlighted and labeled with callouts.

Header Fields and Annotations:

- Signature:** The first three bytes (0x00000000) are labeled "Signature".
- Machine:** The next two bytes (0x00000000) are labeled "Machine".
- NumberOfLinesNumbers:** The next two bytes (0x00000000) are labeled "NumberOfLinesNumbers".
- Characteristics:** The next two bytes (0x00000000) are labeled "Characteristics".
- PointerToRelocation:** The next four bytes (0x00000000) are labeled "PointerToRelocation".
- VirtualSize:** The next four bytes (0x00000000) are labeled "VirtualSize".
- VirtualAddress:** The next four bytes (0x00000000) are labeled "VirtualAddress".
- SizeOfRawData:** The next four bytes (0x00000000) are labeled "SizeOfRawData".
- PointerToRawData:** The next four bytes (0x00000000) are labeled "PointerToRawData".
- NumberOfRelocations:** The next two bytes (0x00000000) are labeled "NumberOfRelocations".
- PointerToLineNumbers:** The next four bytes (0x00000000) are labeled "PointerToLineNumbers".

The diagram also shows the `IMAGE_SECTION_HEADER` section, which contains information about the sections of the file. The first section is labeled ".text" and has a "data" label pointing to its `PointerToRawData` field. The second section is labeled ".rsrc" and has a "rsfc" label pointing to its `PointerToRawData` field. The third section is labeled ".data" and has a "data" label pointing to its `PointerToRawData` field.

The Section Headers

Each **Section** header structure contains the details needed to find it within the file (**PointerToRawData**), its size on disk (**SizeOfRawData**) and once loaded (**VirtualSize**) and where to load it in memory (**VirtualAddress**) relative to the **Optional** header field **ImageBase**. Whether the section contains executable code, can be read from, written to or has other properties is specified by the **Characteristics** field.

A Walk Through the PE32 Format

Display of the main headers describing the basic information contained in a *Portable Executable* file and how it maps to the data in a simple executable.

More details about the *Portable Executable* format can be found at:

http://en.wikipedia.org/wiki/Portable_Executable