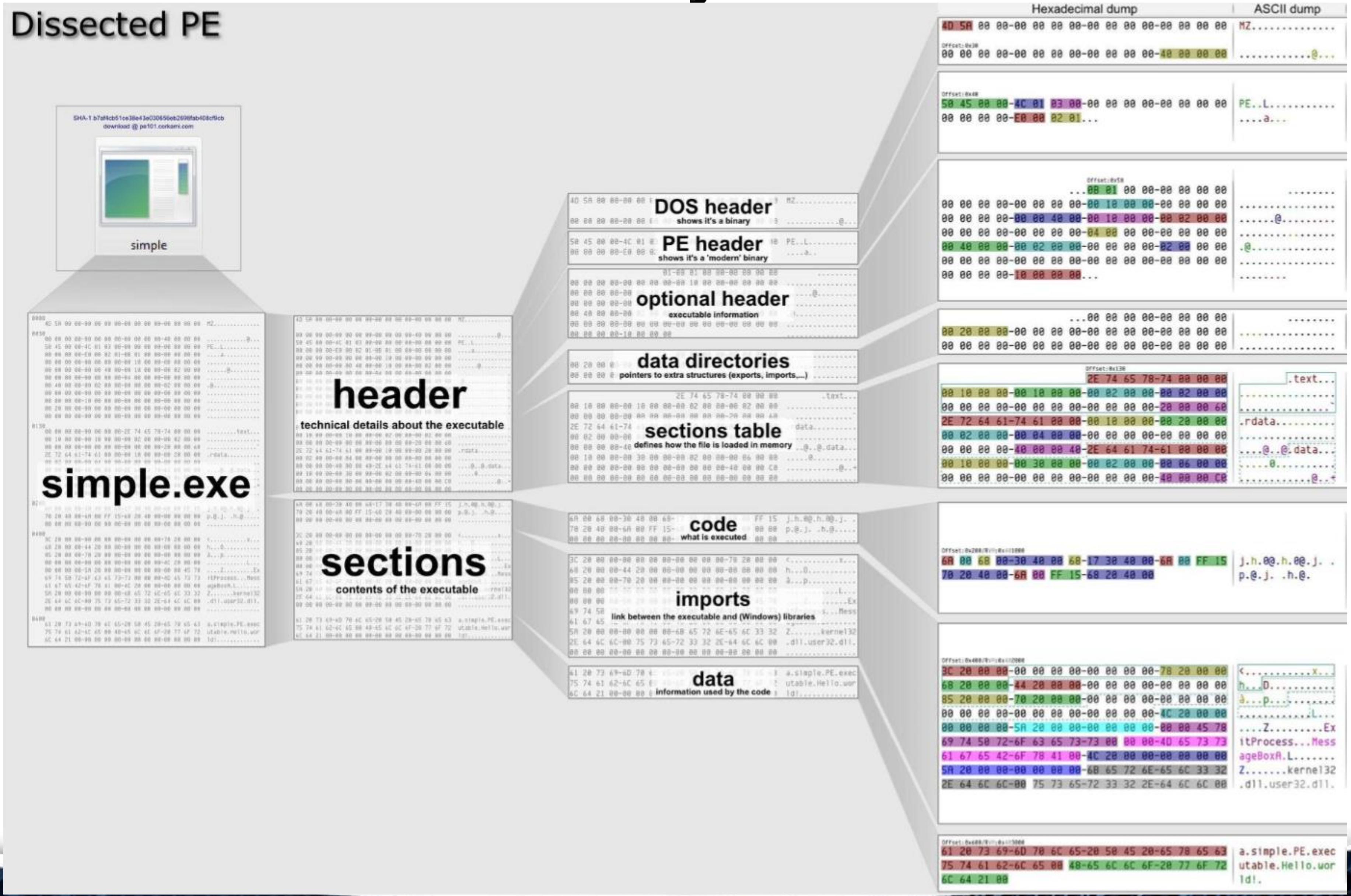
Malware Analysis

By Z-Lab team

PE file format

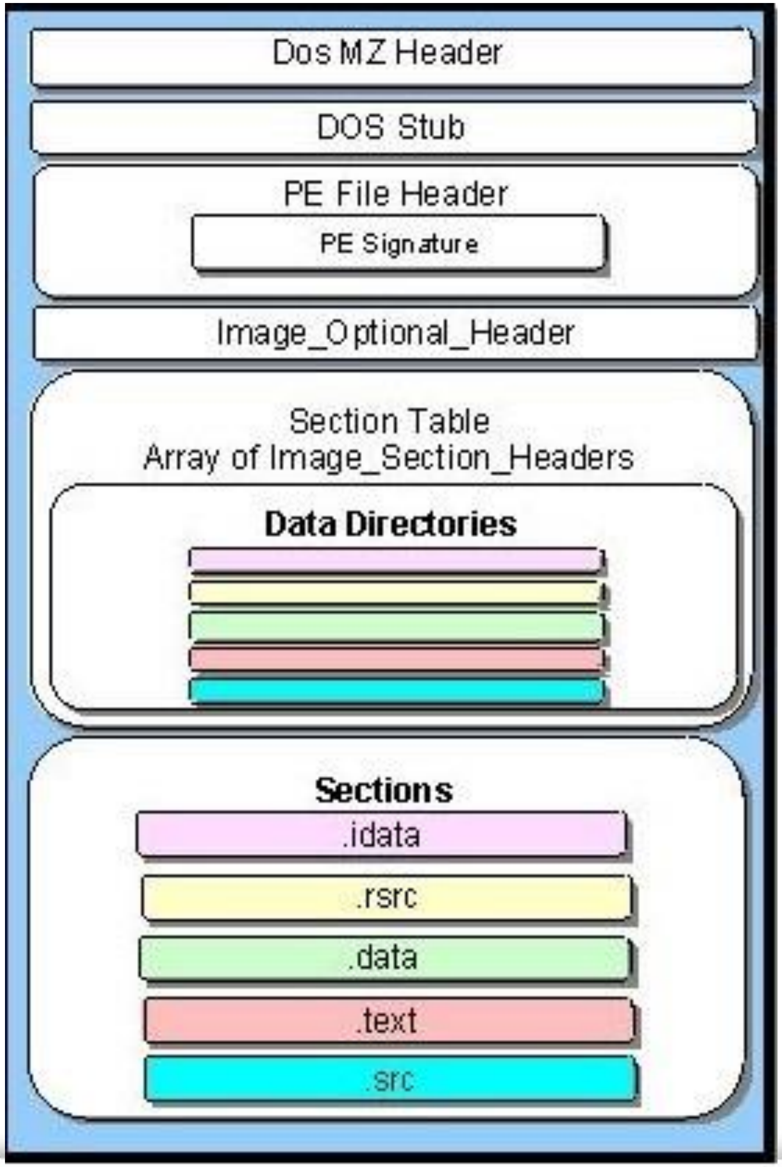
- The Portable Executable (PE) format is a file format for executables and DLLs used in 32-bit and 64-bit versions of Windows operating system.
- The term «portable» refers to the format’s versatility in numerous environments of operating system software architecture.

PE layout


Dissected PE
SHA-1 b7af4cb51ce38e43e030656eb2698fab408cf9cb
download @ pe101.corkami.com
simple
simple.exe
header
technical details about the executable
sections
contents of the executable
DOS header
shows it's a binary
PE header
shows it's a 'modern' binary
optional header
executable information
data directories
pointers to extra structures (exports, imports,...)
sections table
defines how the file is loaded in memory
code
what is executed
imports
link between the executable and (Windows) libraries
data
information used by the code
Hexadecimal dump
ASCII dump

```
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 | ............@...

Offset:0x40
50 45 00 00-4C 01 03 00-00 00 00 00-00 00 00 00 | PE..L...........
00 00 00 00-E0 00 02 01...                      | ....a...

                                     Offset:0x58
                        ...0B 01 00 00-00 00 00 00 |         ........
00 00 00 00-00 00 00 00-00 10 00 00-00 00 00 00 | ................
00 00 00 00-00 00 40 00-00 10 00 00-00 02 00 00 | ......@.........
00 00 00 00-00 00 00 00-04 00 00 00-00 00 00 00 | ................
00 40 00 00-00 02 00 00-00 00 00 00-02 00 00 00 | .@..............
00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00 | ................
00 00 00 00-10 00 00 00...                      | ........

                        ...00 00 00 00-00 00 00 00 |         ........
00 20 00 00-00 00 00 00-00 00 00 00-00 00 00 00 | ................
00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00 | ................

                                    Offset:0x130
                        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 | ............@...

Offset:0x200/RVA:0x1000
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.@.

Offset:0x400/RVA:0x2000
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.

Offset:0x600/RVA:0x3000
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!.
```

PE layout

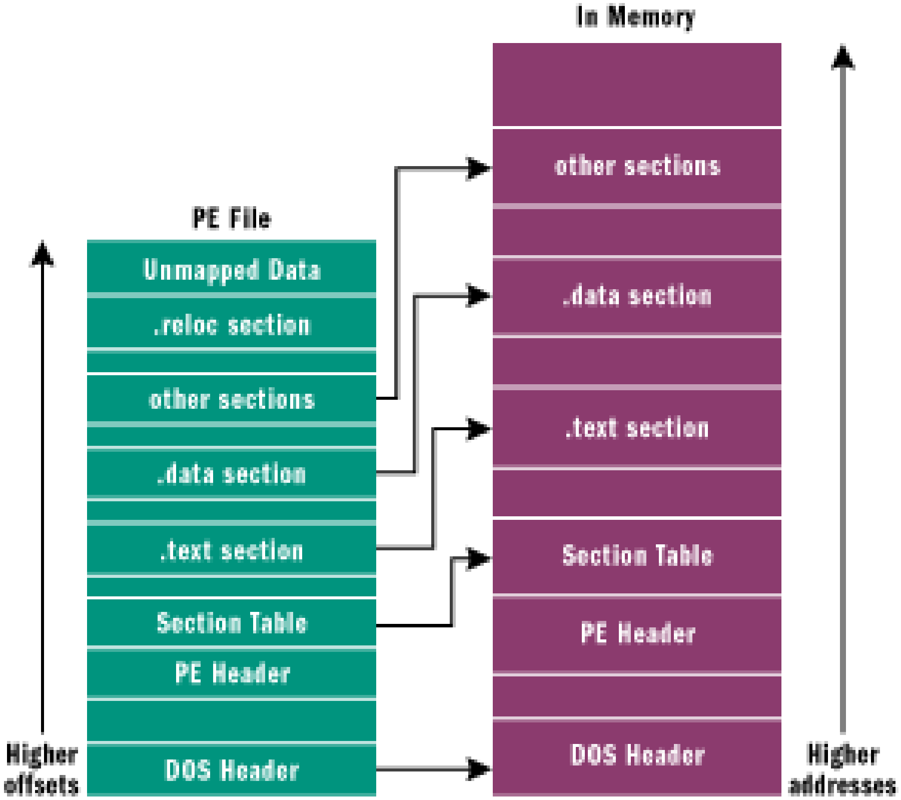

Dos MZ Header
DOS Stub
PE File Header
PE Signature
Image_Optional_Header
Section Table
Array of Image_Section_Headers
Data Directories
Sections
.idata
.rsrc
.data
.text
.src

Some sections

- .text
  - Contains the executable code
- .data
  - Contains inizialized data
- .reloc
  - Contains relocation information
- .rsrc
  - Contains resource info of a module
- .idata
  - Contains import data

Memory mapping

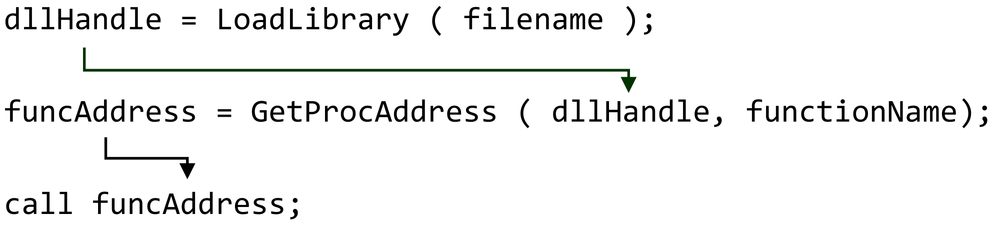
- – Direct mapping in memory



DLL

- Dynamic-link Library
  - Shared library between many processes
  - It is a PE file with the IMAGE_FILE_DLL flag set
  - It exports some functions

- Linking a DLL:
  - Dynamic Linking: the OS loads the DLLs in memory using IAT
  - Runtime Linking: when needs the DLL, the process uses



```
dllHandle = LoadLibrary ( filename );

funcAddress = GetProcAddress ( dllHandle, functionName);

call funcAddress;
```