REVERSE ENGINEERING – CLASS 0x03

THE STRUCTURE OF PE FILES

Cristian Rusu

LAST TIME

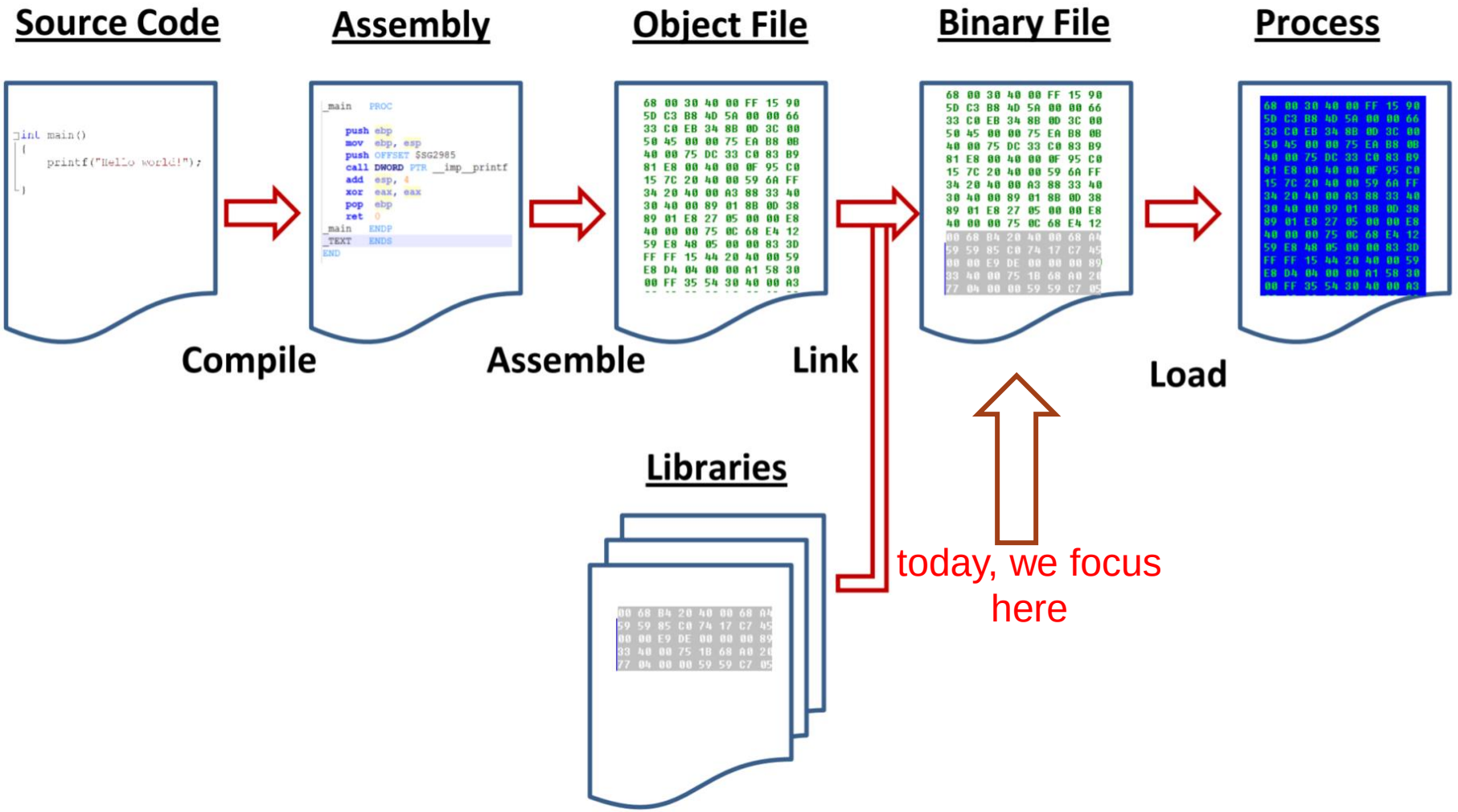
- assembly in context
- the structure of binary files
- study of the ELF binaries
- PE for next week

TODAY

- **the structure of binary files**

- **study of the PE binaries**

FROM SOURCE CODE TO EXECUTION



BINARY FILES

- **ELF/SO**
- **PE/DLL**
- **WASM**
- **machine code (assembly translated to CPU readable instructions) is only part of the executable**
- **all of them have some particular structure we need to understand to in order to execute the binary (ABI)**

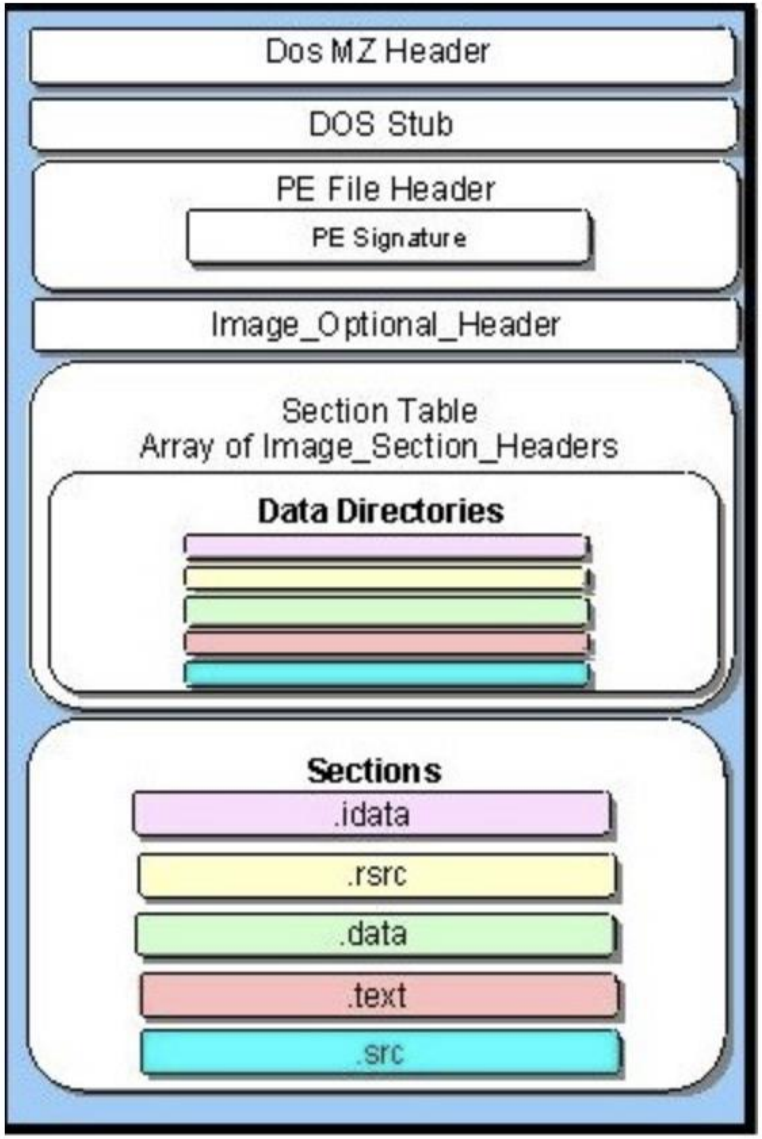
PE BINARY

- **Portable Executable**

- **for both Windows x86 and x64**

PE BINARY

- **headers & sections**



https://resources.infosecinstitute.com/topic/2-malware-researchers-handbook-demystifying-pe-file/

PE BINARY

- **DOS header**
  - first 64 bytes of binary
  - MZ (magic number, just as .ELF)
  - offset to the start of the PE

```
struct DOS_Header
{
// short is 2 bytes, long is 4 bytes
    char signature[2] = { 'M', 'Z' };
    short lastsize;
    short nblocks;
    short nreloc;
    short hdrsize;
    short minalloc;
    short maxalloc;
    void *ss; // 2 byte value
    void *sp; // 2 byte value
    short checksum;
    void *ip; // 2 byte value
    void *cs; // 2 byte value
    short relocpos;
    short noverlay;
    short reserved1[4];
    short oem_id;
    short oem_info;
    short reserved2[10];
    long  e_lfanew; // Offset to the 'PE\0\0' signature relative to the beginning of the file
}
```

https://en.wikipedia.org/wiki/Mark_Zbikowski
https://github.com/Alexpux/mingw-w64/blob/master/mingw-w64-tools/widl/include/winnt.h and look for _IMAGE_DOS_HEADER

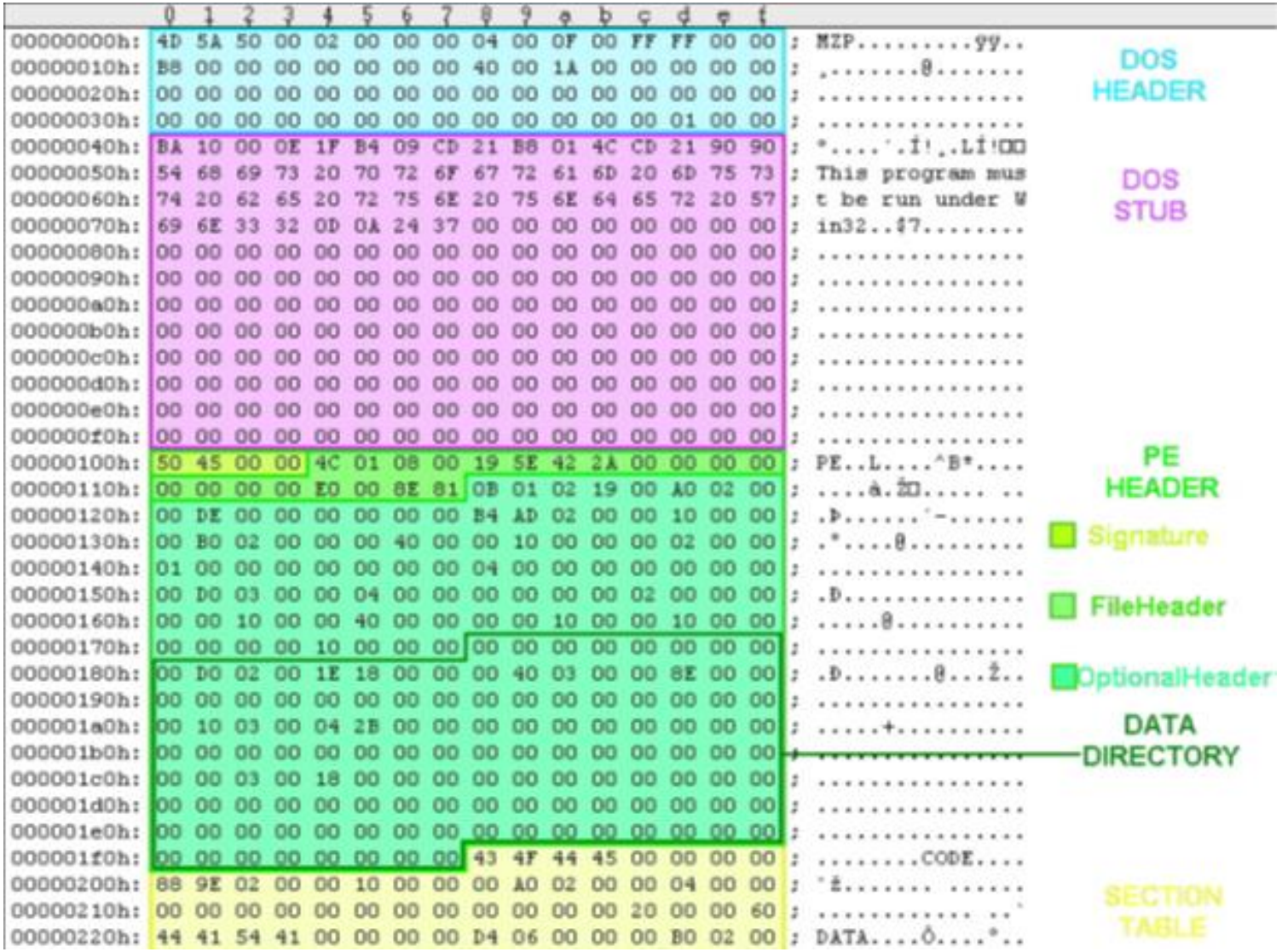
PE BINARY

- **DOS header**
- **DOS stub**
  - „This program cannot be run in DOS mode”
- **PE file header**
  - Signature
    - PE followed by two zeros
  - Machines
    - Target system: intel, AMD, etc.
  - Number of sections
    - Size of the section table
  - Size of optional header, contains information about: binary, such as initial stack size, program entry point location, preferred base address, operating system version, section alignment information

https://github.com/Alexpux/mingw-w64/blob/master/mingw-w64-tools/widl/include/winnt.h look for _IMAGE_FILE_HEADER

PE BINARY

- **dissasembly**



https://resources.infosecinstitute.com/topic/2-malware-researchers-handbook-demystifying-pe-file/

PE BINARY

- **tools for PE analysis**

- **PE Studio**
  - a utility for inspecting PE formatted binaries such as windows EXEs and DLLs

- **CFF Explorer**
  - a freeware suite of tools including a PE editor and a process viewer

- **PE bear**
  - a multiplatform reversing tool for PE files. Its objective is to deliver fast and flexible “first view” for malware analysts, stable and capable to handle malformed PE files

https://www.winitor.com/download
https://github.com/cybertechniques/site/blob/master/analysis_tools/cff-explorer/index.md
https://github.com/hasherezade/pe-bear

BINARY ANALYSIS

- **general tools**

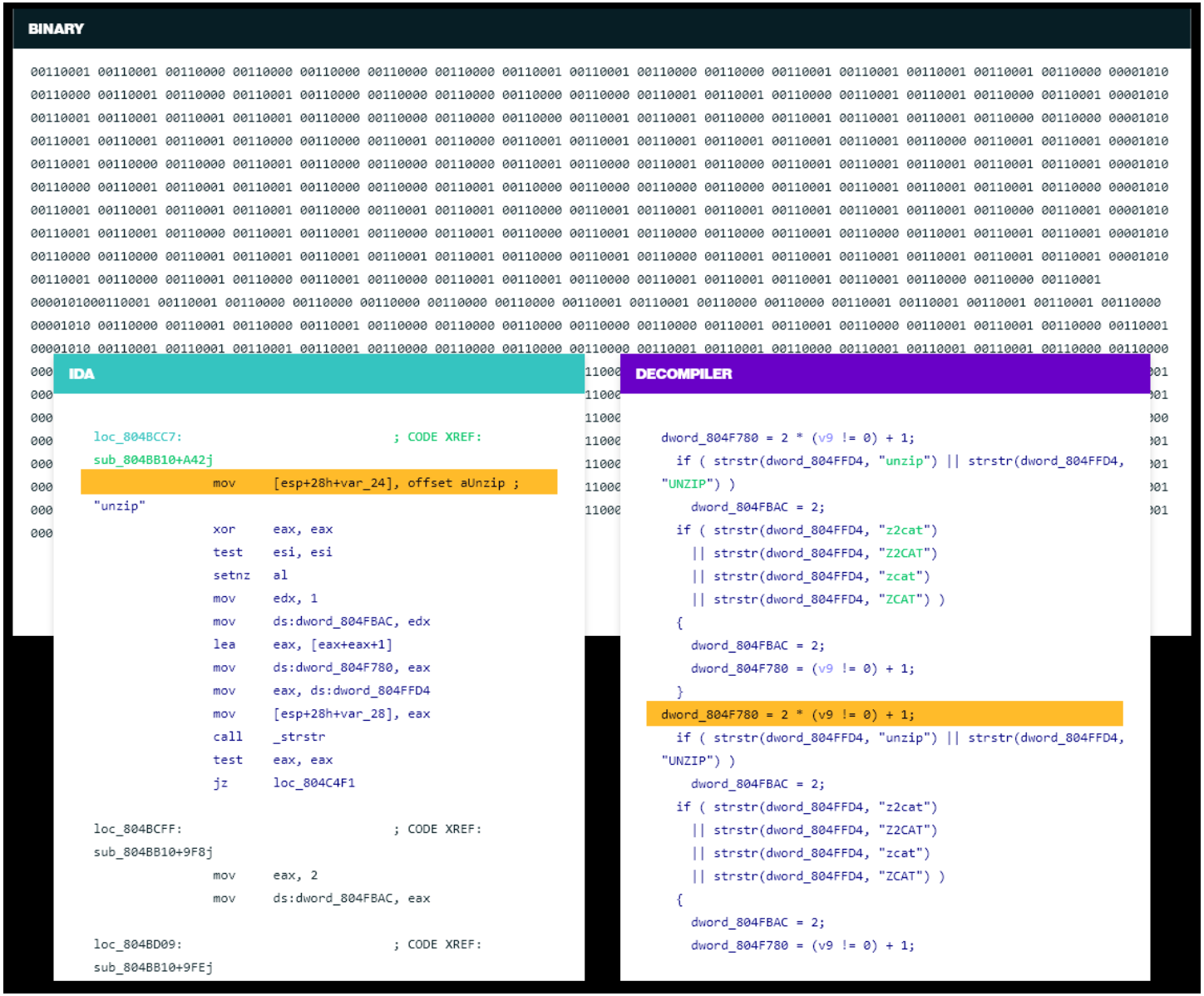
- **Ghidra**
  - Open-sourced NSA tool
  - Pro: free and hackable
  - Pro: decompiles anything it can disassemble
  - Con: looks horrible (UI/UX skills zero)
  - Con: sometimes the decompilation is hard/impossible to follow
  - Prefers gotos (no for loop support)

- **IDA**
  - Swiss army knife of Reverse Engineering
  - Pro: Tried and tested
  - Pro: Analyze most executable file formats
  - Pro: Disassemble most architectures (x86, arm, mips, z80, etc)
  - Pro: Decompile some architectures (x86/amd64, arm/arm64, ppc/ppc64, mips32)
  - Con: Too expensive
  - Con: Piracy is rampant

https://ghidra-sre.org/
https://hex-rays.com/ida-free/

IDA SHOWCASE

- go from machine code back to source code (ideally)



https://hex-rays.com/ida-free/

WHAT WE DID TODAY

- **PE binaries**
  - DOS/PE structure
- **general static binary analysis tools**
  - Gidra
  - IDA (las session today)

.

NEXT TIME …

- **Dynamic analysis & reverse engineering**

REFERENCES

- **Decompiler explorer**
  - https://dogbolt.org/
- **PE insights**
  - https://en.wikibooks.org/wiki/X86_Disassembly/Windows_Executable_Files
  - https://resources.infosecinstitute.com/topic/2-malware-researchers-handbook-demystifying-pe-file/
- **PE 101,** https://web.cse.ohio-state.edu/~reeves.92/CSE2421au12/HelloWorldGoal.pdf
- https://github.com/corkami/pocs/tree/master/PE
- **Introdution to IDA pro**
  - https://www.youtube.com/watch?v=qCQRKLaz2nQ
- **Intro to RE with IDA on Pes**
  - https://www.youtube.com/watch?v=1MotMBPX7tY

.