An Introduction to Reverse Engineering

Jerre Starink

Twente Hacking Squad
https://ths.eemcs.utwente.nl/

UNIVERSITY OF TWENTE.

How to follow along

- Download Python: https://python.org/
- Download Ghidra: https://ghidra-sre.org/ (requires JDK 17 64-bit)
- To run the challenges, you will need Linux or a Linux VM

What is this program doing?

program

Warmup

What input do the following programs expect?

What input does this program expect? (1/6)

```
def challenge1(input_password):
    if input_password == "THS{secr3t}":
        return True
    else:
        return False
```

What input does this program expect? (2/6)

```python
def challenge2(input_password):
    if len(input_password) == 17 \
        and input_password.startswith("THS{sup3r_") \
        and input_password.endswith("ecr3t}") \
        and input_password[6] == '5':
        return True
    return False
```

What input does this program expect? (3/6)

```c
int challenge3(const char* input_password) {
    if (strlen(input_password) != 8) return 0;

    return input_password[0] == 'T' && input_password[1] == 'H'
        && input_password[2] == 'S' && input_password[3] == '{'
        && input_password[4] == 'w' && input_password[5] == '0'
        && input_password[6] == 'w' && input_password[7] == '}';
}
```

UNIVERSITY OF TWENTE.

What input does this program expect? (4/6)

```c
const char SECRET[22] = "}setyb_eht_esrever{SHT";

int challenge4(const char* input_password) {
    if (strlen(input_password) != 22) return 0;

    for (int i = 0; i < 22; i++) {
        if (input_password[i] != SECRET[21 - i])
            return 0;
    }
    return 1;
}
```

UNIVERSITY OF TWENTE.

What input does this program expect? (5/6)

```c
int challenge5(const char* input_password) {
    if (strlen(input_password) != 11)             return 0;
    if (strncmp("THS{", input_password, 4) != 0) return 0;
    if (input_password[10] == '}')                return 0;

    for (int i = 4; i < 10; i++) {
        if (input_password[i] < '0' || input_password[i] > '9') return 0;
        if (i > 4 && input_password[i-1] >= input_password[i])  return 0;
    }
    return 1;
}
```

UNIVERSITY OF TWENTE.

What input does this program expect? (6/6)

```
00000000: 7f45 4c46 0201 0100 0000 0000 0000 0000  .ELF............
00000010: 0300 3e00 0100 0000 4010 0000 0000 0000  ..>.....@.......
00000020: 4000 0000 0000 0000 c834 0000 0000 0000  @........4......
00000030: 0000 0000 4000 3800 0d00 4000 1e00 1d00  ....@.8...@.....
00000040: 0600 0000 0400 0000 4000 0000 0000 0000  ........@.......
00000050: 4000 0000 0000 0000 4000 0000 0000 0000  @.......@.......
00000060: d802 0000 0000 0000 d802 0000 0000 0000  ................
00000070: 0800 0000 0000 0000 0300 0000 0400 0000  ................
00000080: 1803 0000 0000 0000 1803 0000 0000 0000  ................
00000090: 1803 0000 0000 0000 1c00 0000 0000 0000  ................
000000A0: ...
```

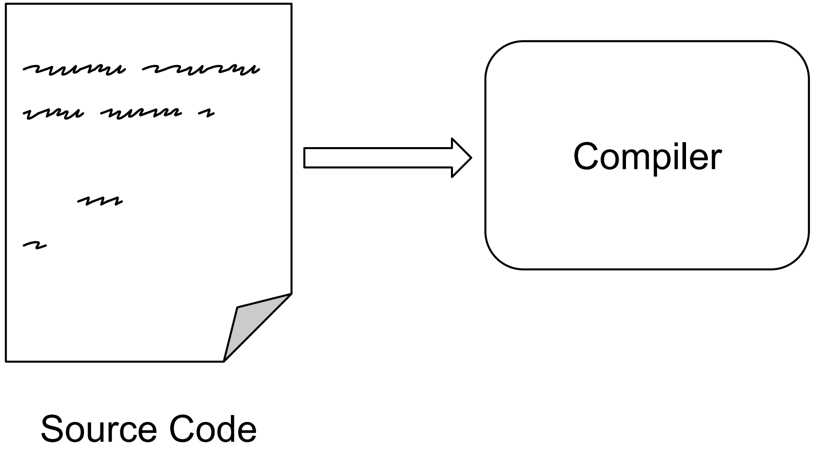
Theory: Compilers

UNIVERSITY OF TWENTE.

Software (Forward) Engineering

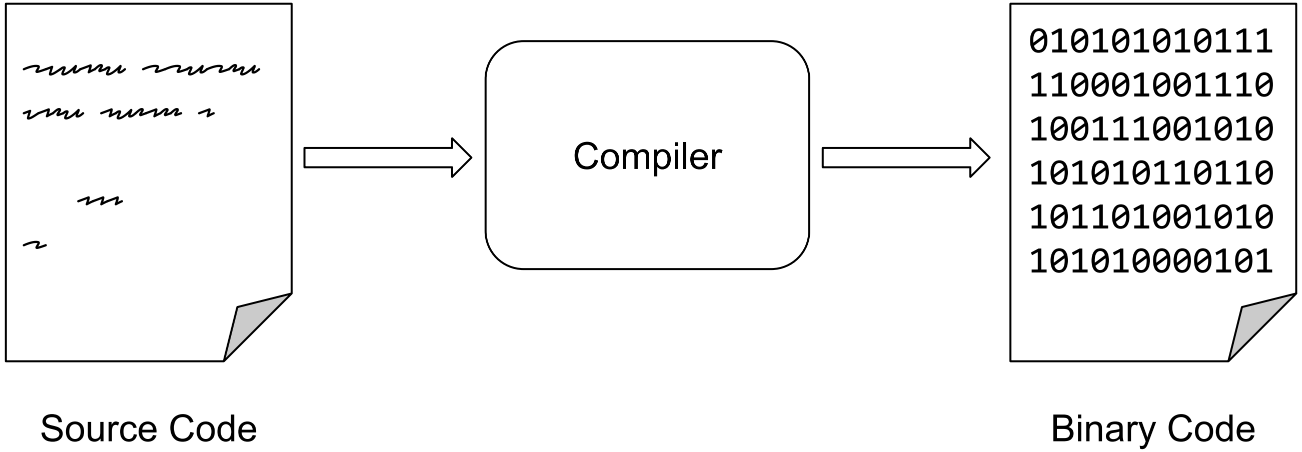
Source Code

Software (Forward) Engineering

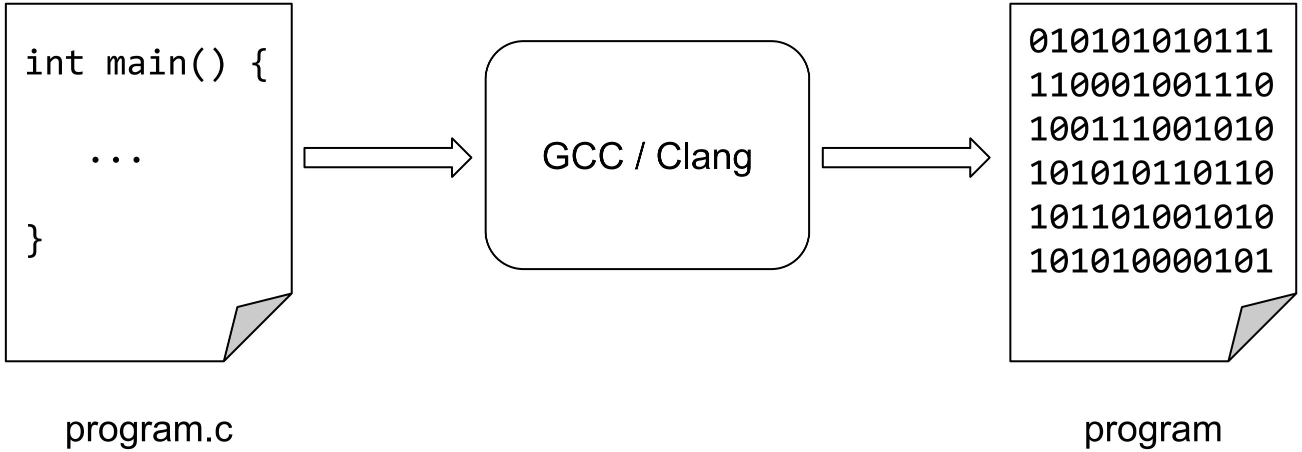


UNIVERSITY OF TWENTE.

Software (Forward) Engineering

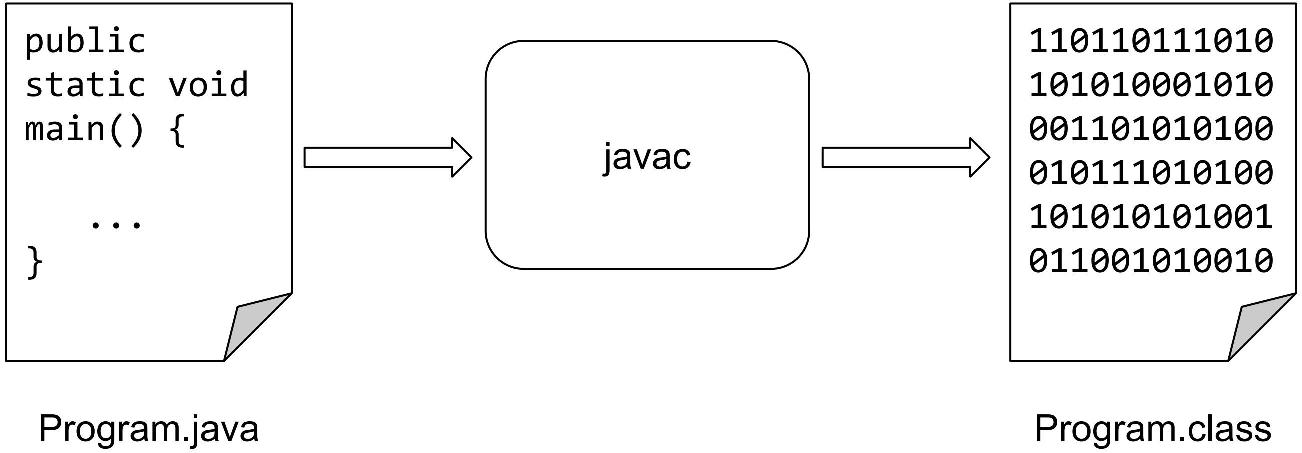


Software (Forward) Engineering



UNIVERSITY OF TWENTE.

Software (Forward) Engineering

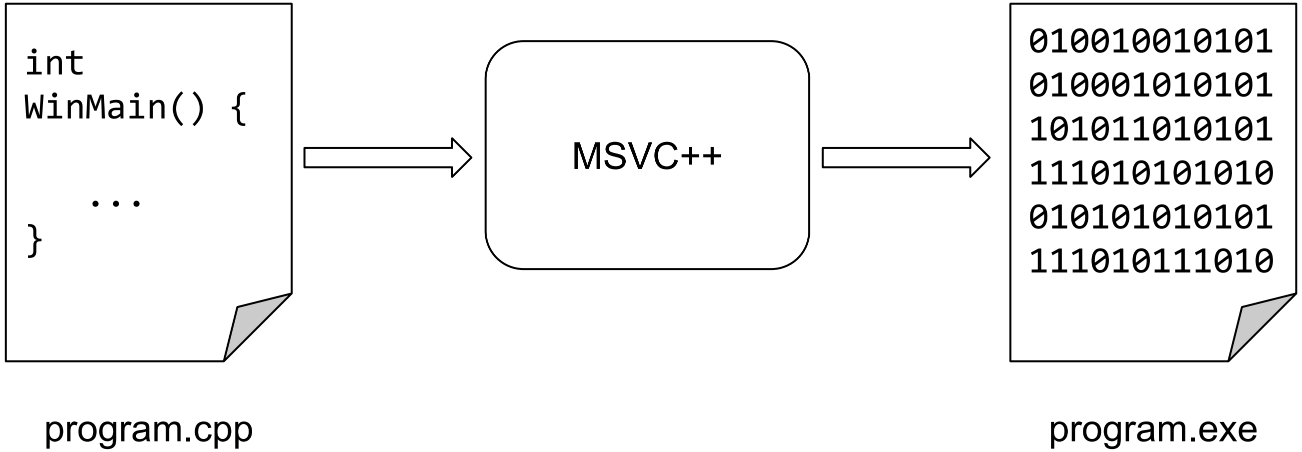


```
public
static void
main() {

   ...
}
```

```
110110111010
101010001010
001101010100
010111010100
101010101001
011001010010
```

UNIVERSITY OF TWENTE.

Software (Forward) Engineering

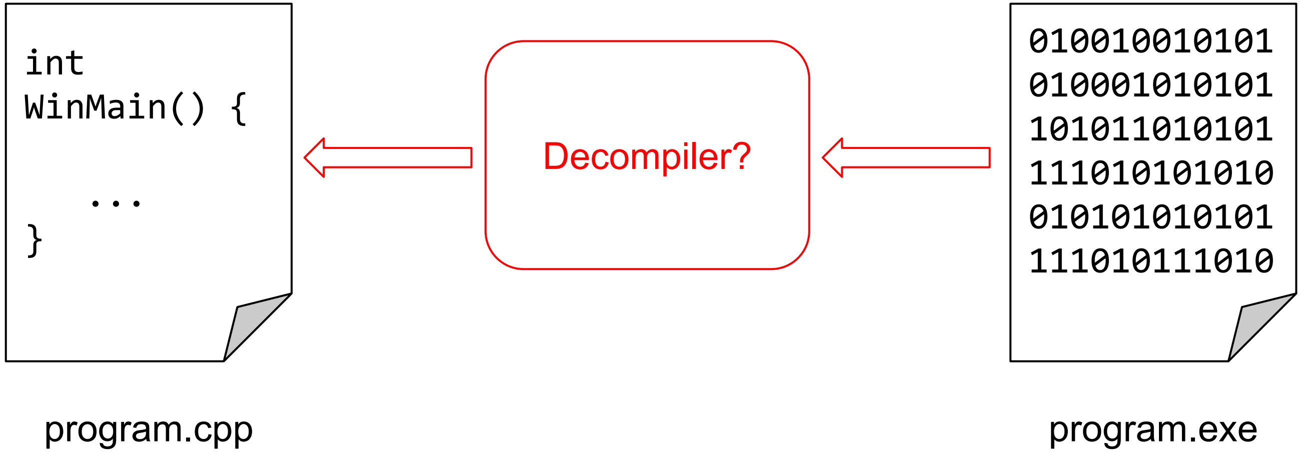


```
int
WinMain() {

    ...
}
```

```
010010010101
010001010101
101011010101
111010101010
010101010101
111010111010
```

UNIVERSITY OF TWENTE.

Software (~~Forward~~ Reverse) Engineering



UNIVERSITY OF TWENTE.

Static Analysis

Challenge: REasy (ths.eemcs.utwente.nl)

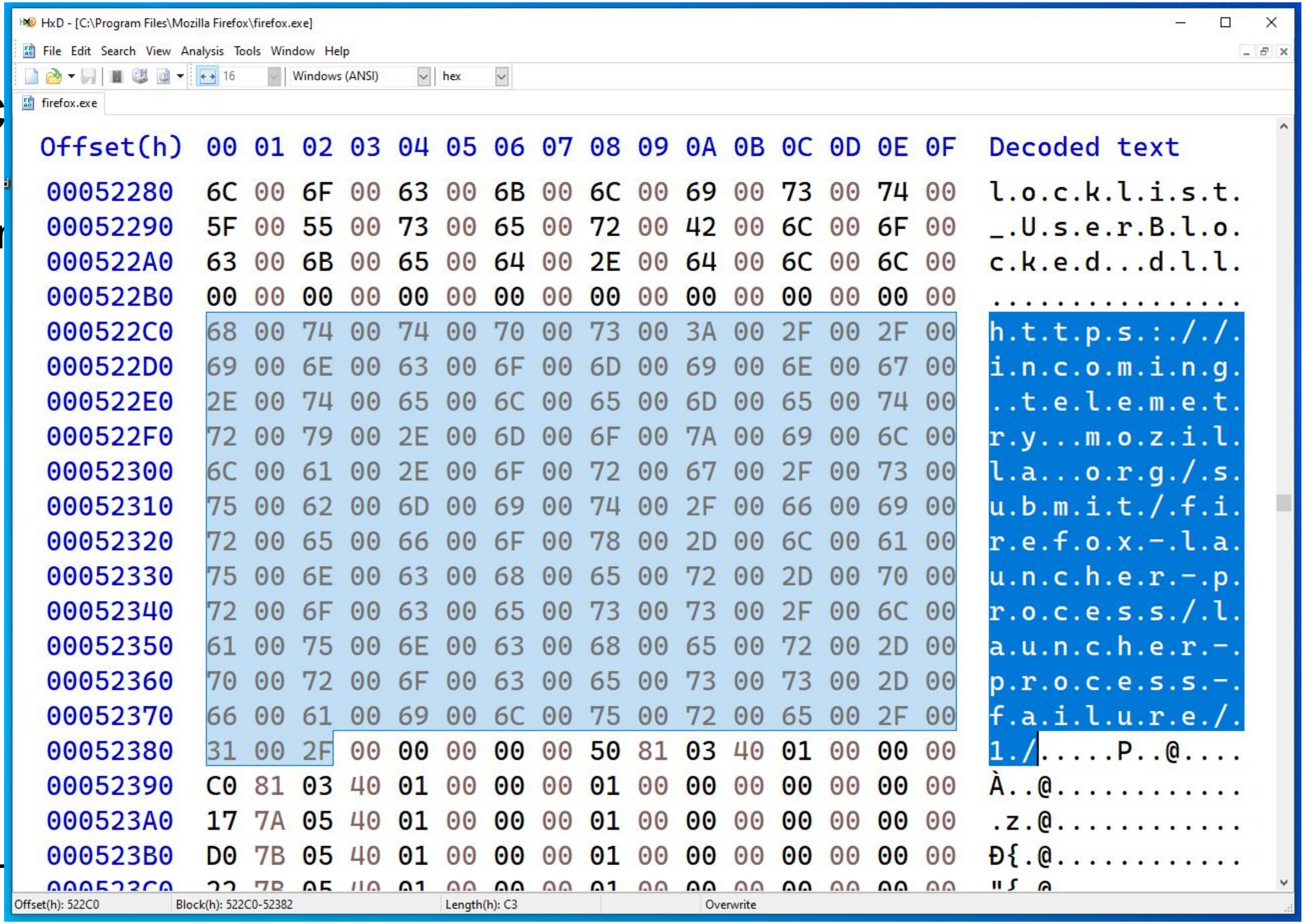
Static Analysis

Static Analysis

- Computers are deterministic:
  - Everything required to run this program is stored in this program file.
  - This includes all the **strings** and **constants** the program uses.

Static

- Co
  -
  -



HxD - [C:\Program Files\Mozilla Firefox\firefox.exe]

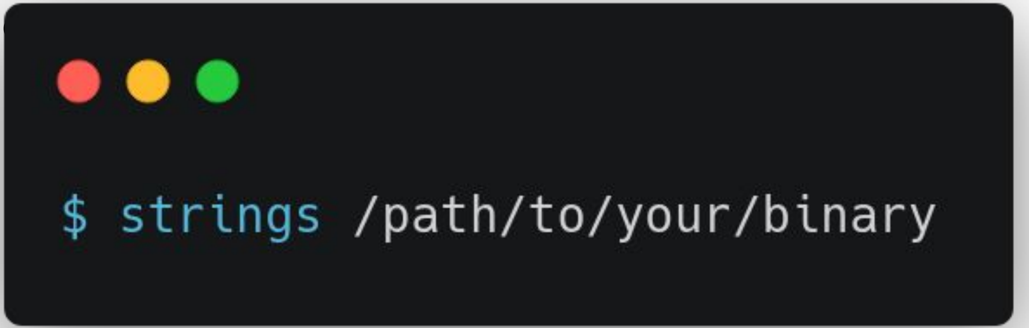
```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F  Decoded text

00052280  6C 00 6F 00 63 00 6B 00 6C 00 69 00 73 00 74 00  l.o.c.k.l.i.s.t.
00052290  5F 00 55 00 73 00 65 00 72 00 42 00 6C 00 6F 00  _.U.s.e.r.B.l.o.
000522A0  63 00 6B 00 65 00 64 00 2E 00 64 00 6C 00 6C 00  c.k.e.d...d.l.l.
000522B0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
000522C0  68 00 74 00 74 00 70 00 73 00 3A 00 2F 00 2F 00  h.t.t.p.s.:././.
000522D0  69 00 6E 00 63 00 6F 00 6D 00 69 00 6E 00 67 00  i.n.c.o.m.i.n.g.
000522E0  2E 00 74 00 65 00 6C 00 65 00 6D 00 65 00 74 00  ..t.e.l.e.m.e.t.
000522F0  72 00 79 00 2E 00 6D 00 6F 00 7A 00 69 00 6C 00  r.y...m.o.z.i.l.
00052300  6C 00 61 00 2E 00 6F 00 72 00 67 00 2F 00 73 00  l.a...o.r.g./.s.
00052310  75 00 62 00 6D 00 69 00 74 00 2F 00 66 00 69 00  u.b.m.i.t./.f.i.
00052320  72 00 65 00 66 00 6F 00 78 00 2D 00 6C 00 61 00  r.e.f.o.x.-.l.a.
00052330  75 00 6E 00 63 00 68 00 65 00 72 00 2D 00 70 00  u.n.c.h.e.r.-.p.
00052340  72 00 6F 00 63 00 65 00 73 00 73 00 2F 00 6C 00  r.o.c.e.s.s./.l.
00052350  61 00 75 00 6E 00 63 00 68 00 65 00 72 00 2D 00  a.u.n.c.h.e.r.-.
00052360  70 00 72 00 6F 00 63 00 65 00 73 00 73 00 2D 00  p.r.o.c.e.s.s.-.
00052370  66 00 61 00 69 00 6C 00 75 00 72 00 65 00 2F 00  f.a.i.l.u.r.e./.
00052380  31 00 2F 00 00 00 00 00 50 81 03 40 01 00 00 00  1./.....P..@....
00052390  C0 81 03 40 01 00 00 00 01 00 00 00 00 00 00 00  À..@............
000523A0  17 7A 05 40 01 00 00 00 01 00 00 00 00 00 00 00  .z.@............
000523B0  D0 7B 05 40 01 00 00 00 01 00 00 00 00 00 00 00  Ð{.@............
000523C0  22 7B 05 40 01 00 00 00 01 00 00 00 00 00 00 00  "{.@
```

Offset(h): 522C0   Block(h): 522C0-52382   Length(h): C3   Overwrite

UNIVERSI

Static Analysis

- Computers are deterministic:
    - Everything required to run this program is stored in this program file.
    - This includ



```
$ strings /path/to/your/binary
```

UNIVERSITY OF TWENTE.

Static Analysis

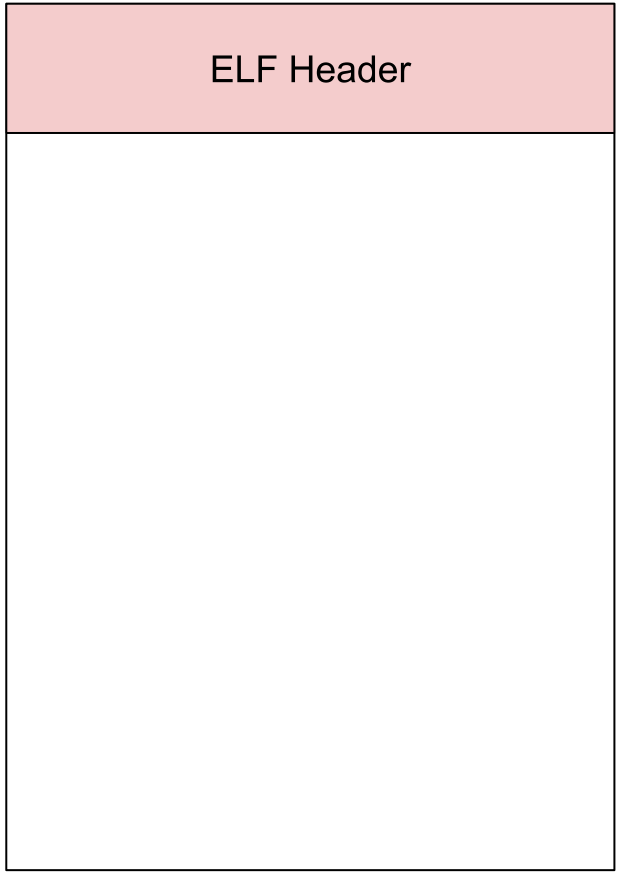
- Computers are deterministic:
    - Everything required to run this program is stored in this program file.
    - This includes all the **strings** and **constants** the program uses.

Static Analysis

- Computers are deterministic:
  - Everything required to run this program is stored in this program file.
  - This includes all the **strings** and **constants** the program uses.
- Binary files have a structure:
  - On Linux: Executable and Linkable Format (ELF).

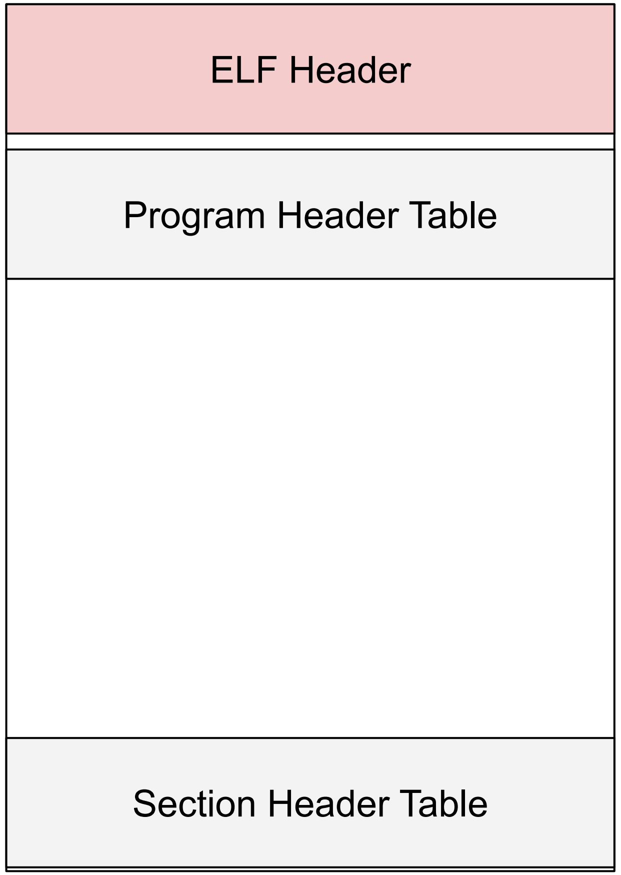
Static Analysis - ELF Files

- ELF Header



Static Analysis - ELF Files

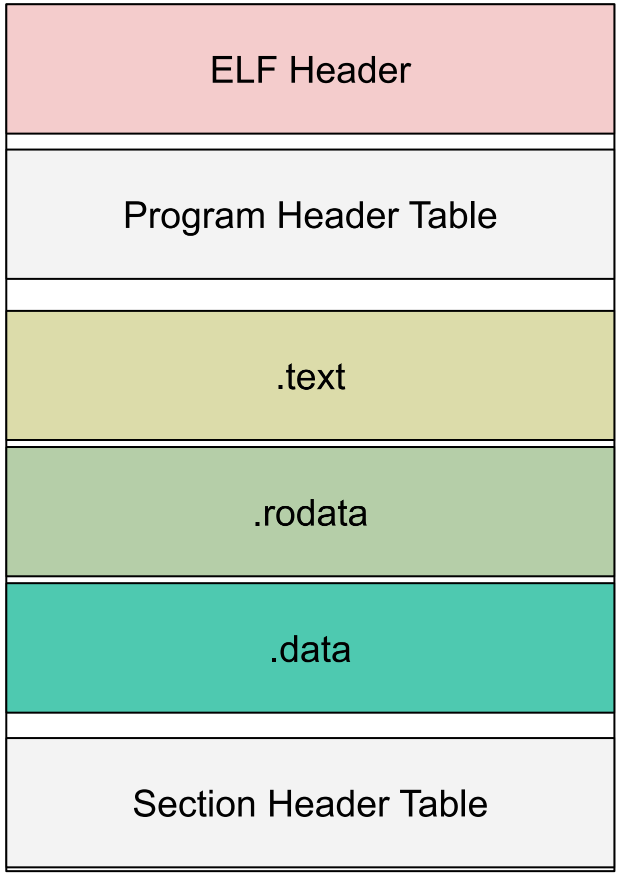
- ELF Header
- Program and Section Header Tables



UNIVERSITY OF TWENTE.

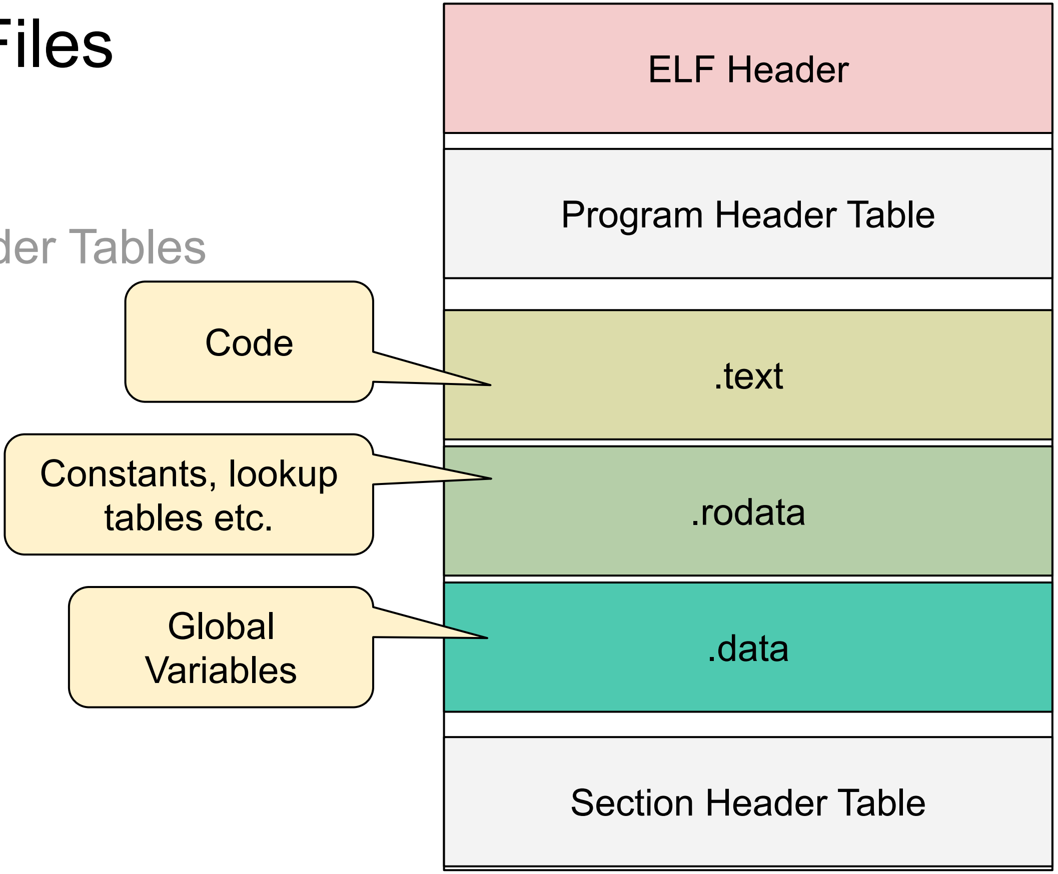
Static Analysis - ELF Files

- ELF Header
- Program and Section Header Tables
- Sections



Static Analysis - ELF Files

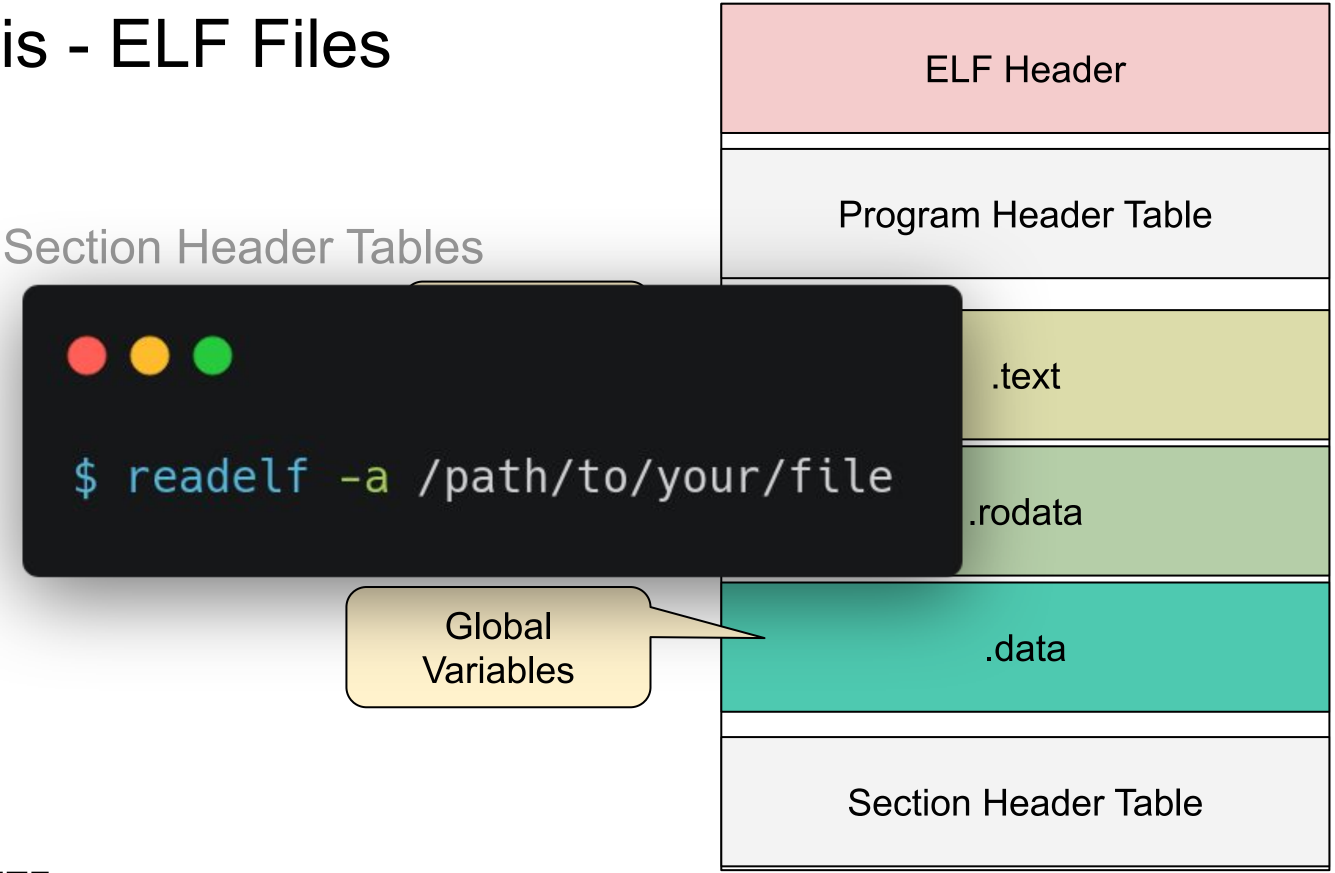
- ELF Header
- Program and Section Header Tables
- Sections



UNIVERSITY OF TWENTE.

Static Analysis - ELF Files

- ELF Header
- Program and Section Header Tables
- Sections



```
$ readelf -a /path/to/your/file
```

UNIVERSITY OF TWENTE.

Static Analysis

- Computers are deterministic:
  - Everything required to run this program is stored in this program file.
  - This includes all the **strings** and **constants** the program uses.
- Binary files have a structure:
  - On Linux: Executable and Linkable Format (ELF).

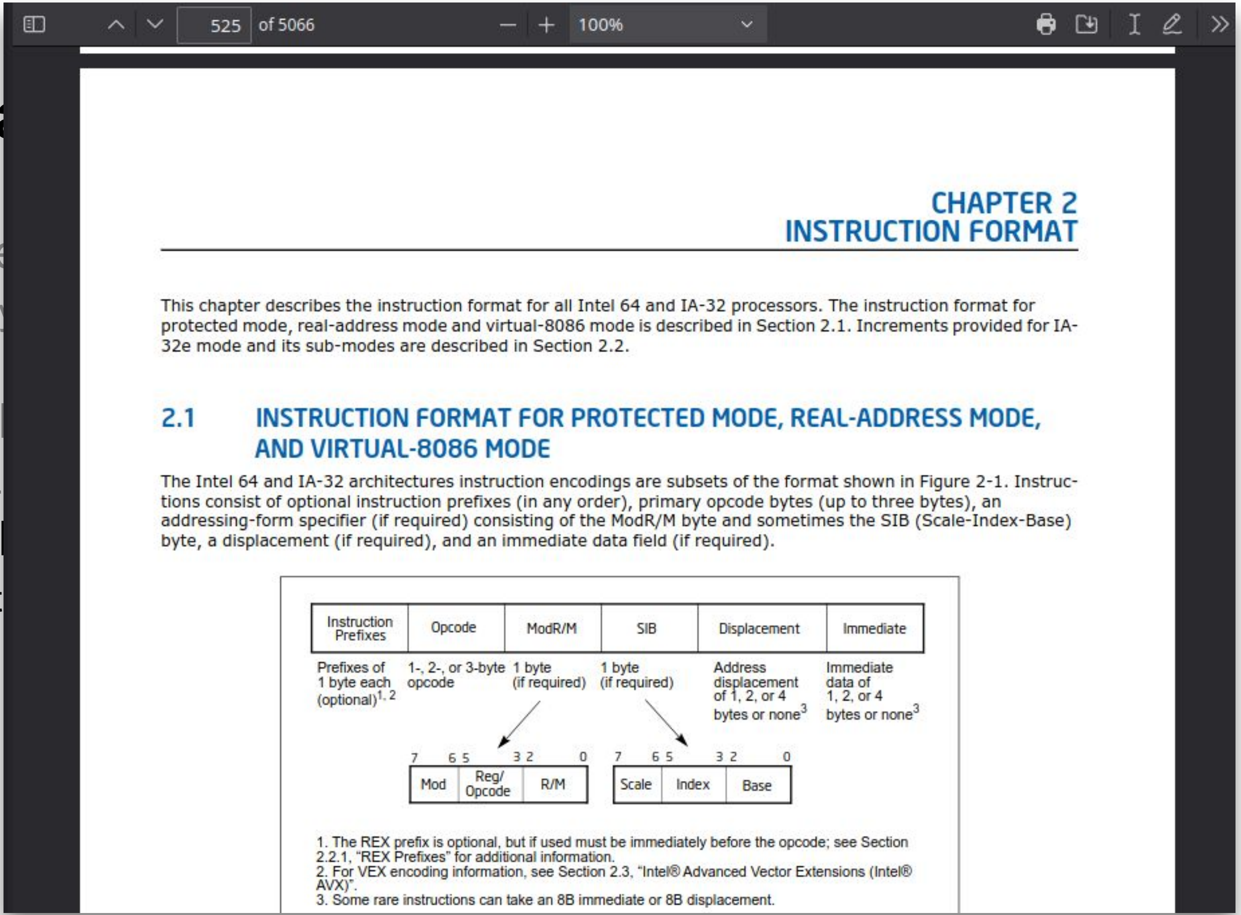
UNIVERSITY OF TWENTE.

Static Analysis

- Computers are deterministic:
  - Everything required to run this program is stored in this program file.
  - This includes all the **strings** and **constants** the program uses.
- Binary files have a structure:
  - On Linux: Executable and Linkable Format (ELF).
- Binary files use a known instruction set:
  - Most consumer PCs: x86, x86-64.

Static Ana

- Compute
  - Every
  - This
- Binary fi
  - On L
- Binary fi
  - Most



525 of 5066 100%

CHAPTER 2
INSTRUCTION FORMAT

This chapter describes the instruction format for all Intel 64 and IA-32 processors. The instruction format for protected mode, real-address mode and virtual-8086 mode is described in Section 2.1. Increments provided for IA-32e mode and its sub-modes are described in Section 2.2.

2.1 INSTRUCTION FORMAT FOR PROTECTED MODE, REAL-ADDRESS MODE, AND VIRTUAL-8086 MODE

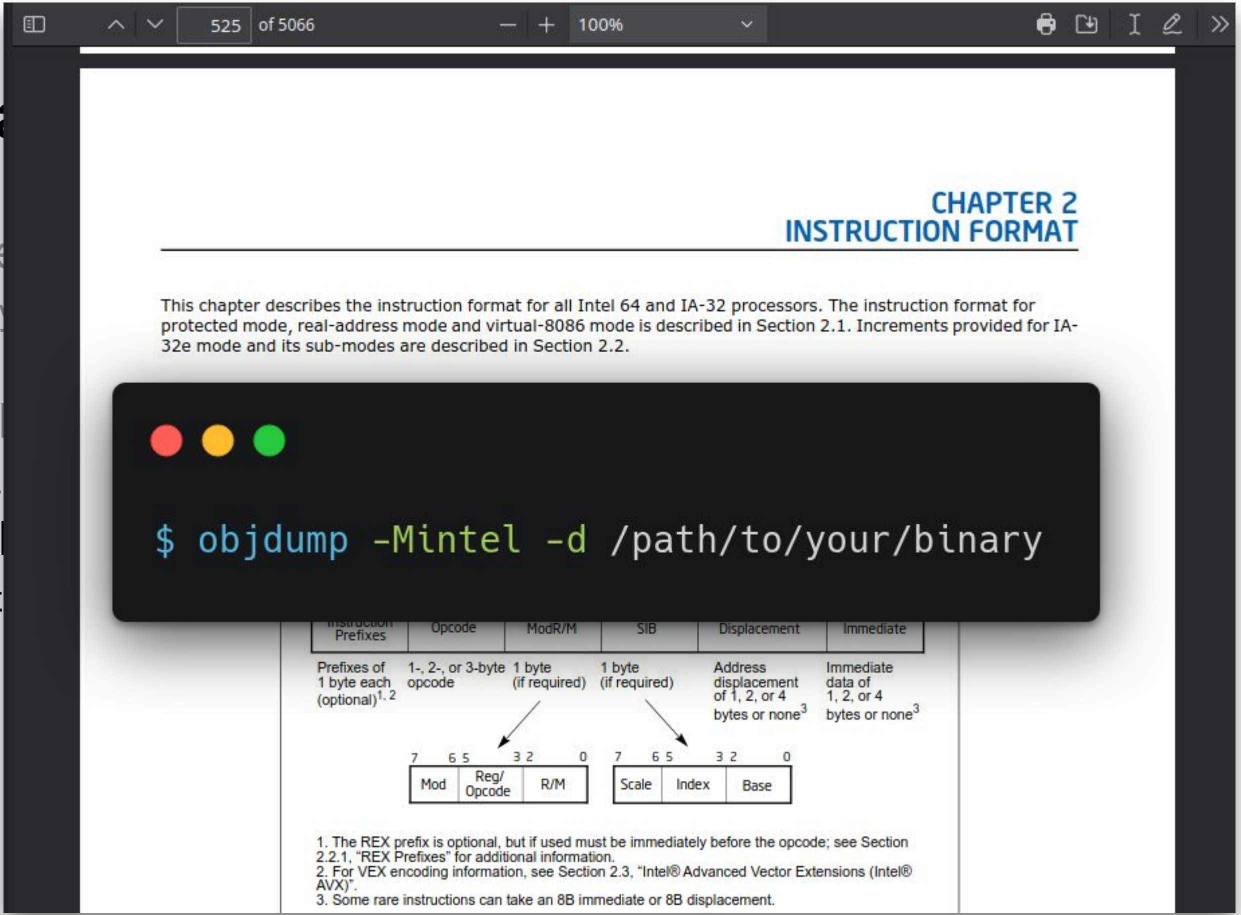
The Intel 64 and IA-32 architectures instruction encodings are subsets of the format shown in Figure 2-1. Instructions consist of optional instruction prefixes (in any order), primary opcode bytes (up to three bytes), an addressing-form specifier (if required) consisting of the ModR/M byte and sometimes the SIB (Scale-Index-Base) byte, a displacement (if required), and an immediate data field (if required).

1. The REX prefix is optional, but if used must be immediately before the opcode; see Section 2.2.1, "REX Prefixes" for additional information.
2. For VEX encoding information, see Section 2.3, "Intel® Advanced Vector Extensions (Intel® AVX)".
3. Some rare instructions can take an 8B immediate or 8B displacement.

UNIVERSITY OF TWENTE. https://www.intel.com/content/www/us/en/developer/articles/technical/intel-sdm.html

Static Ana

- Compute
  - Every
  - This
- Binary fi
  - On L
- Binary fi
  - Most



525 of 5066 100%

CHAPTER 2
INSTRUCTION FORMAT

This chapter describes the instruction format for all Intel 64 and IA-32 processors. The instruction format for protected mode, real-address mode and virtual-8086 mode is described in Section 2.1. Increments provided for IA-32e mode and its sub-modes are described in Section 2.2.

```
$ objdump -Mintel -d /path/to/your/binary
```

Instruction Prefixes	Opcode	ModR/M	SIB	Displacement	Immediate
Prefixes of 1 byte each (optional)[1, 2]	1-, 2-, or 3-byte opcode	1 byte (if required)	1 byte (if required)	Address displacement of 1, 2, or 4 bytes or none[3]	Immediate data of 1, 2, or 4 bytes or none[3]

1. The REX prefix is optional, but if used must be immediately before the opcode; see Section 2.2.1, "REX Prefixes" for additional information.
2. For VEX encoding information, see Section 2.3, "Intel® Advanced Vector Extensions (Intel® AVX)".
3. Some rare instructions can take an 8B immediate or 8B displacement.

UNIVERSITY OF TWENTE.

https://www.intel.com/content/www/us/en/developer/articles/technical/intel-sdm.html

Static Analysis

- Computers are deterministic:
  - Everything required to run this program is stored in this program file.
  - This includes all the **strings** and **constants** the program uses.
- Binary files have a structure:
  - On Linux: Executable and Linkable Format (ELF).
- Binary files use a known instruction set:
  - Most consumer PCs: x86, x86-64.

Static Analysis Tools

UNIVERSITY OF TWENTE.

Static Analysis Tools

- Quick and dirty tools:
  - strings,
  - readelf,
  - objdump.

UNIVERSITY OF TWENTE.

Static Analysis Tools

- Quick and dirty tools:
  - strings,
  - readelf,
  - objdump.
- General purpose tools (including decompilers):
  - Ghidra (https://ghidra-sre.org/),
  - Cutter / rizin / radare2 (https://cutter.re/),
  - angr-management (https://github.com/angr/angr-management),
  - Hex-Rays IDA (https://hex-rays.com/, commercial),
  - Binary Ninja (https://binary.ninja/, commercial).

GHIDRA

https://ghidra-sre.org/

Where do you start?

- Before you dive into the code, run the program first and see what happens!

Where do you start?

- Before you dive into the code, run the program first and see what happens!
- Look for interesting strings:
  - Prompts (e.g., “Input:”),
  - Error messages (e.g., “Please enter a valid name.”),
  - URLs (e.g., “evil.com”),
  - Paths (e.g., “flag.txt”).
  - Flags (e.g., grep for “THS{”).

Where do you start?

- Before you dive into the code, run the program first and see what happens!
- Look for interesting strings:
  - Prompts (e.g., “Input:”),
  - Error messages (e.g., “Please enter a valid name.”),
  - URLs (e.g., “evil.com”),
  - Paths (e.g., “flag.txt”).
  - Flags (e.g., grep for “THS{”).
- Look for interesting symbols:
  - Exported Functions (e.g., main),
  - Imported Functions (e.g., printf, scanf, gets, puts, strcmp, …).

Where do you start?

- Before you dive into the code, run the program first and see what happens!
- Look for interesting strings:
  - Prompts (e.g., “Input:”),
  - Error messages (e.g., “Please enter a valid name.”),
  - URLs (e.g., “evil.com”),
  - Paths (e.g., “flag.txt”).
  - Flags (e.g., grep for “THS{”).
- Look for interesting symbols:
  - Exported Functions (e.g., main),
  - Imported Functions (e.g., printf, scanf, gets, puts, strcmp, …).
- Look up documentation, rename variables, help the decompiler.

Challenges

- THS (ths.eemcs.utwente.nl)
  - REasy
  - KeyGenie
- HackTheBox (hackthebox.com):
  - Simple Encryptor
  - Exactlon
  - Impossible Password (retired but still a good introductory challenge)

UNIVERSITY OF TWENTE.

Dynamic Analysis

Dynamic Analysis

- Reading code and trying to understand it takes a lot of time.

Dynamic Analysis

- Reading code and trying to understand it takes a lot of time.
- We can also run the program and pause it at specific breakpoints:
    - The program’s current memory and registers can tell you a lot.

Dynamic Analysis

```
SECRET = b'\x12\x86\x01 ... (truncated encrypted data)'

def magic_decrypt_function(data):
    # ... Extremely complicated math-heavy code here ...
    return result

def challenge7(input_password):
    if input_password == magic_decrypt_function(SECRET):
        return True
    else:
        return False
```

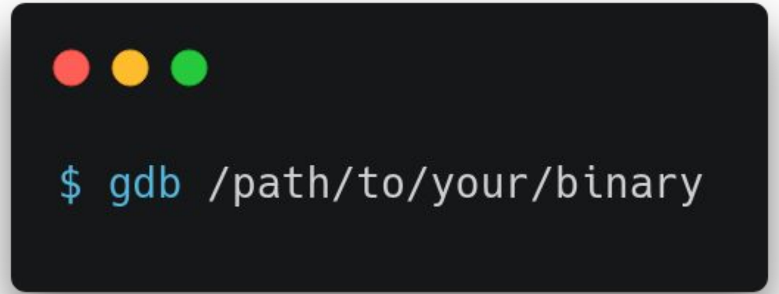
Dynamic Analysis

```
SECRET = b'\x12\x86\x01 ... (truncated encrypted data)'

def magic_decrypt_function(data):
    # ... Extremely complicated math-heavy code here ...
    return result

def challenge7(input_password):
    if input_password == magic_decrypt_function(SECRET):
        return True
    else:
        return False
```

If we can pause the program right after the “magic_function” call, the correct password should be visible in memory.

Dynamic Analysis

- Reading code and trying to understand it takes a lot of time.
- We can also run the program and pause it at specific breakpoints:
  - The program's current memory and registers can tell you a lot.



```
$ gdb /path/to/your/binary
```

UNIVERSITY OF TWENTE.

GDB Cheat Sheet

- Start a new GDB instance: `gdb /path/to/your/file`
- Common GDB commands:

Run/Restart Program	run, r, starti
Pause Execution	Ctrl+C
Continue Execution	c
Set Breakpoint	b *0x55557812, b *main, b *main+123
View registers	info reg
View memory	x /10bx 0x55557812, x /10bx $rsi

UNIVERSITY OF TWENTE.

Challenges

- THS (ths.eemcs.utwente.nl)
  - REasy
  - KeyGenie
  - break-in
- HackTheBox (hackthebox.com):
  - Simple Encryptor
  - Exactlon
  - Impossible Password (retired but still a good introductory challenge)

UNIVERSITY OF TWENTE.