



# Hack The Box Netherlands

2nd Wednesday Hack-day  
meetup

**Welcome!**

# Before we start



- When presentations are going, please **mute your microphone** (we all love the sound of mechanical keyboards, but it makes it hard to hear people)
- If you have a **question**, type it in the **chat**
- Portions of this meetup will be **recorded**

# Discord Community



<https://discord.gg/BynaNgS>

# Slides and video



You can download these slides from our companion website

<https://hackdewereld.nl>

# What to expect

- Today we have a presentation: **Just Enough Reverse Engineering (1 hour)**
- After this presentation you will have a choice:
  - Participate in a Reverse Engineering **workshop**
  - Hack on some machines in our **dedicated lab**



# Just Enough Reverse Engineering

to score those Challenge points!

Arjen / credmp



Cyber Security Faculty of NOVI University of Applied  
Science



Certified Malware Reverse Engineering Instructor



# What is reverse engineering?

*the processes of extracting knowledge or design information from anything man-made*

- Gives you a great understanding of how computer systems work
- Great fun if you like puzzles
- Allows you to peek “under the hood” of any product

# Hack The Box Challenges

- Reversing is one type of challenge
- Today you will learn enough to:
  - Complete the easy challenges
  - Get started on the harder ones

 Labs

Starting Point

Access

Machines

Challenges

Reversing

10

Crypto

10

Stego

10

Pwn

10

Web

10

Misc

10

**What are binaries?**

# How binaries are created

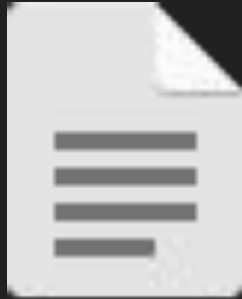
Source Code



compile



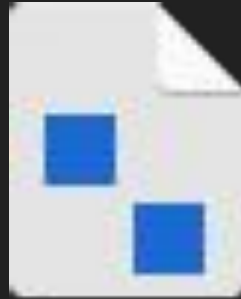
Assembly



assemble



Object File



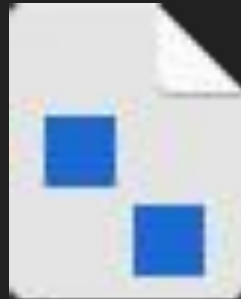
link



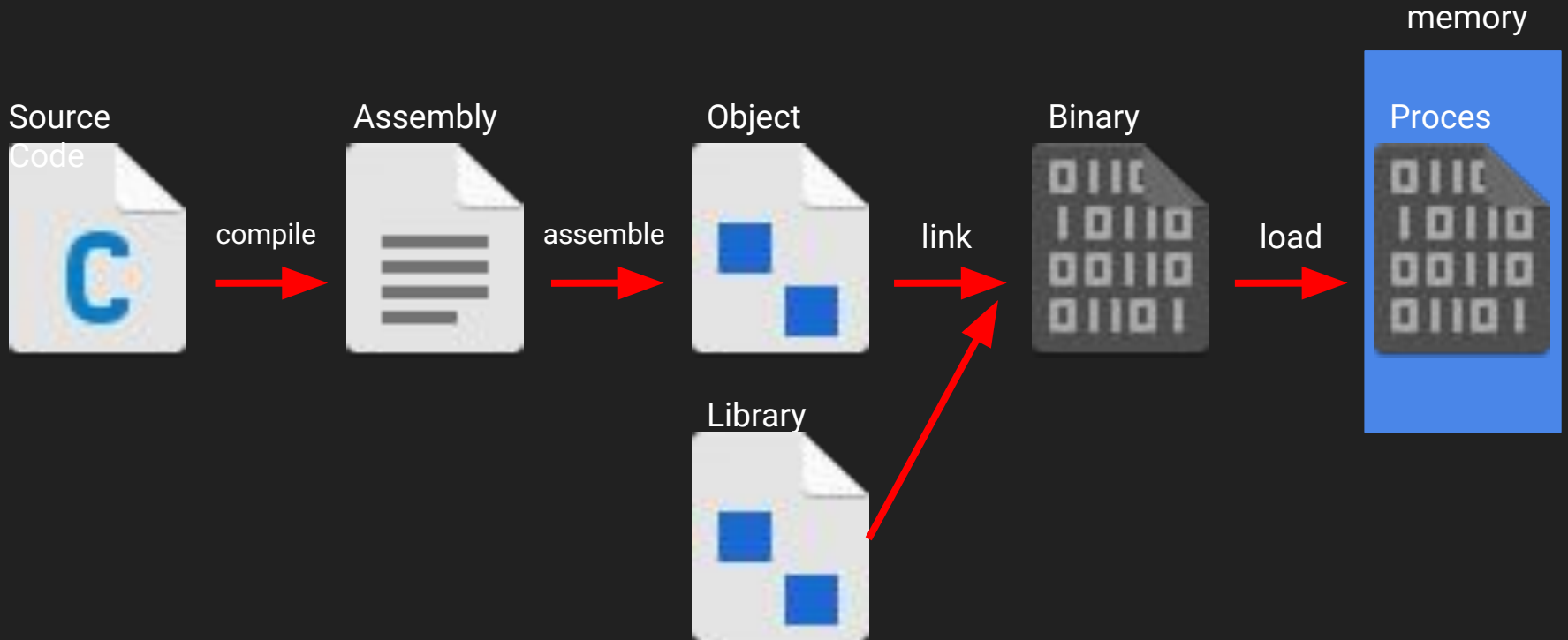
Binary File



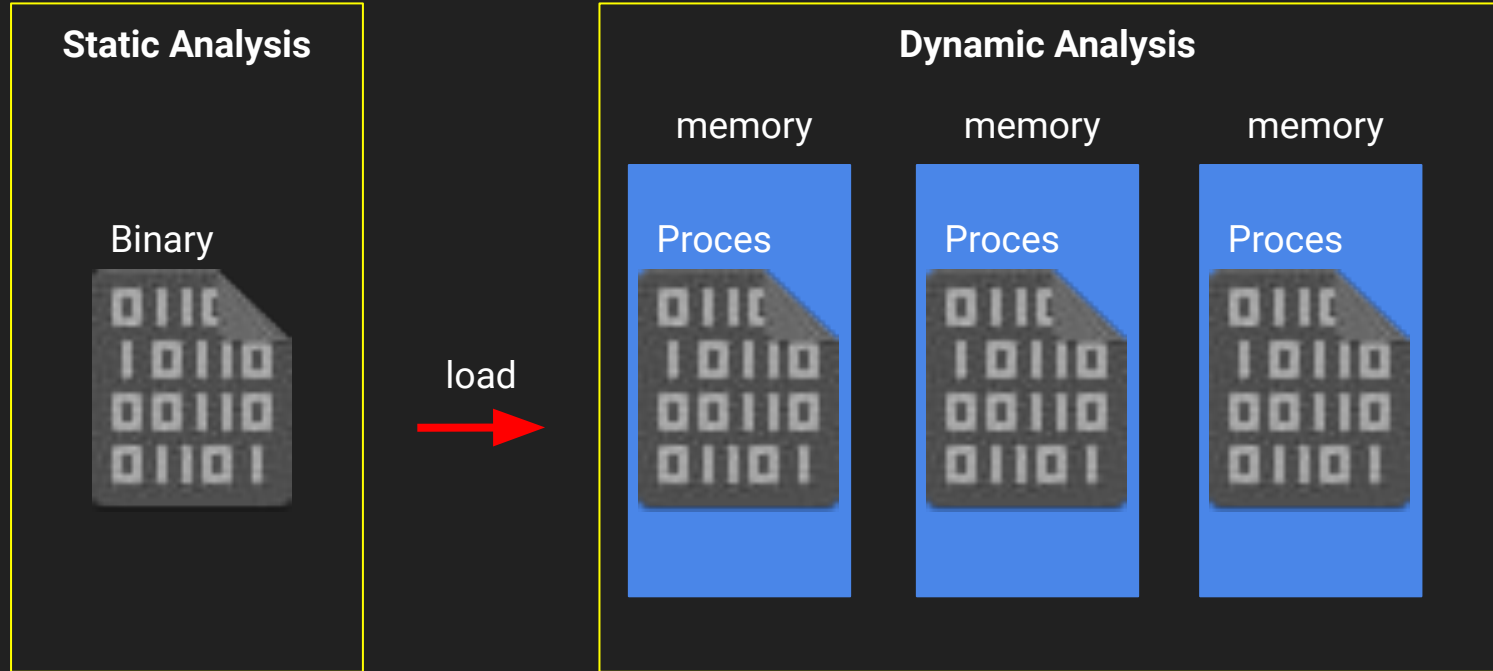
Library File



# How binaries are created




# Reverse Engineering Domain



# Static Analysis

# Do we have the right binary?

- Ensure the zip file is correct by comparing hashes
- Tool: **md5sum** or **sha256sum**

🏆 [10 Points] Baby RE [by Xh4H] [8831 solvers] 1912 👍 98 🗨️ Difficulty:  ⚠️

🔥 First Blood: mprox

Show us your basic skills! (P.S. There are 4 ways to solve this, are you willing to try them all?)



Download

Zip Password: hackthebox

sha256: 13bdad272ee08f609bd41e7d24a3e2f8581d3a07e9e42fce92d3dd35d38da160



Complete



# Unpacking on Kali

- Use 7z to extract zip archives

```
> 7z x Baby_RE.zip
```

```
7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21  
p7zip Version 16.02 (locale=en_US.utf8,Utf16=on,HugeFiles=on,64 bits,1
```

```
Scanning the drive for archives:  
1 file, 2885 bytes (3 KiB)
```

```
Extracting archive: Baby_RE.zip
```

```
--
```

```
Path = Baby_RE.zip  
Type = zip  
Physical Size = 2885
```

```
Enter password (will not be echoed):  
Everything is Ok
```

# Tool: file

- used to determine filetype
- with binaries it shows so much more!
  - Executable type (ELF, PE, etc)
  - Architecture
  - Endianess
  - Statically or Dynamically linked

```
> file baby
baby: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=25adc53b89f781335a27bf1b81f5c4cb74581022, for GNU/Linux 3.2.0, not stripped
```

# Tool: strings

- list printable strings in a binary
- allows you to determine program functions
- will show a lot of small strings as well, so need to filter

```
> strings -n 10 headache
/lib64/ld-linux-x86-64.so.2
__cxa_finalize
__libc_start_main
GLIBC_2.2.5
_ITM_deregisterTMCloneTable
__gmon_start__
_ITM_registerTMCloneTable
[]A\A]A^A_
a15abe90c112d09369d9f9da9a8c046e
Initialising
Enter the key:
Login Failed!
Login success!
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0:
HTB{not_so_easy_lo!}
GCC: (Debian 8.3.0-19) 8.3.0
```

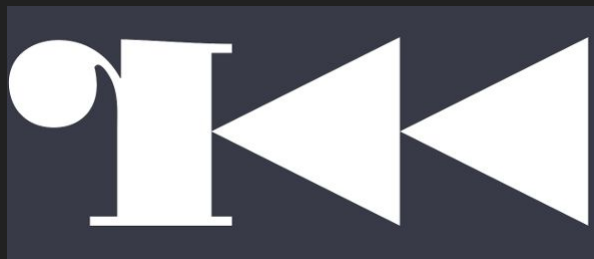
# Is it packed? (Tool: upx)

- Packing “hides” the data inside the binary
- It is only loaded at runtime and you can not examine it without unpacking
- Most popular packer is **upx**
- **upx -l <filename>** will show if the binary is packed

```
Ultimate Packer for eXecutables
Copyright (C) 1996 - 2018
UPX 3.95 SizeOfM Markus Oberhumer, Laszlo Molnar & John Reiser Aug 26th 2018
```

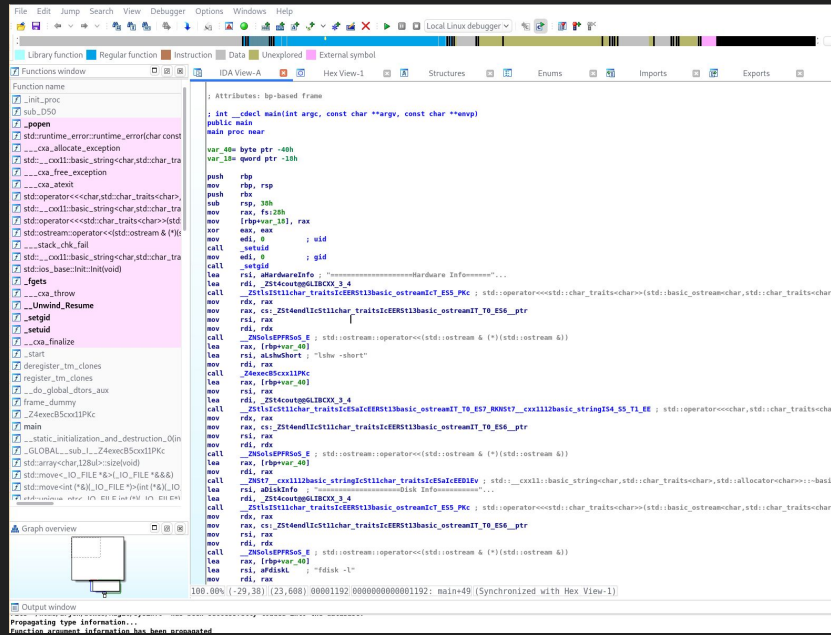
File size	Ratio	Format	Name
2202568	709524	32.21%	linux/amd64

# Disassembler



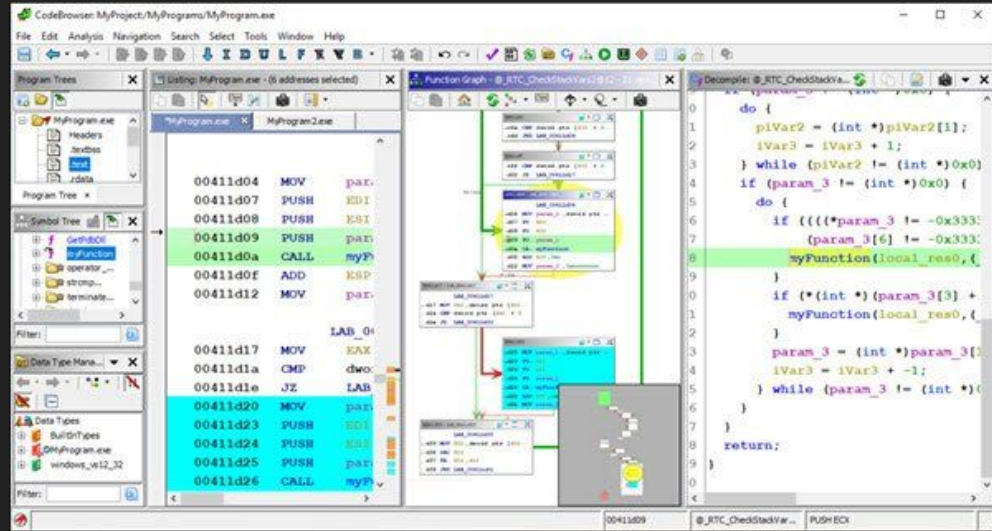
# IDA Pro / Freeware

- Commercial tool from Hex-Rays
- Loved in the industry, but very expensive!
- Comes with a debugger



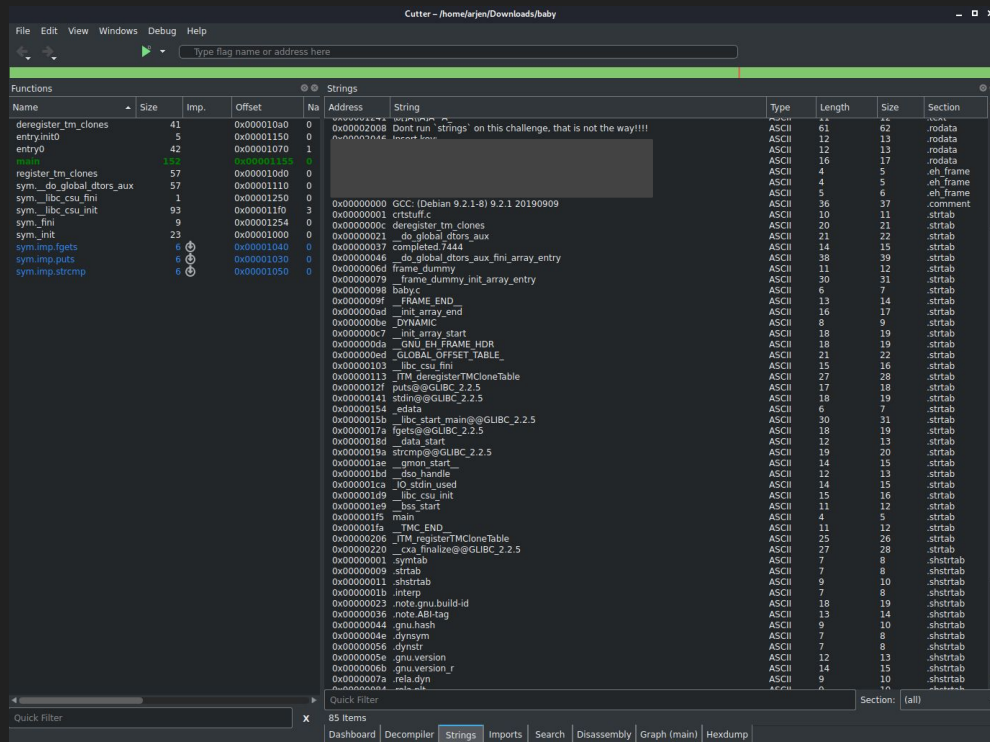
# Ghidra

- Developed by the NSA
- Has been open sourced
- Not the most intuitive, but very powerful
- Comes with a **decompiler!!!!**
- But sadly no debugger yet....



# Radare2 / Cutter

- Libre and portable reverse engineering framework
- More than just a disassembler
- Cutter (the GUI) now includes the **Ghidra decompiler**
- Features a **debugger** (Beta)





# My advice

- When you get started, use **Ghidra**
- Examine the other tools once you finish the challenges (IDA / Cutter)
- Invest some time in learning 1 of these tools very well

# Dynamic Analysis

# Tracing

- Tool: ltrace
  - Traces library calls performed by the binary
  - C functions such as **puts, fgets, strcmp**
- Tool: strace
  - Traces system calls and signals performed by the binary
  - functions related to **memory management, file operations**

```
arjen@kali:~/Downloads$ ltrace ./secret_  
__libc_start_main(0x4008fd, 1, 0x7ffe184  
time(0)  
srand(0x5ec2f2f0, 0x7ffe1844b7d8, 0x7ffe  
rand(0x7fb9795cc740, 0x7ffe1844b6a4, 0x7  
fopen("/tmp/secret", "rb")  
exit(-2 <no return ...>  
+++ exited (status 254) +++
```

# Debugger

- Use a debugger to **step through** the program
- You can also manipulate the program by **changing registers**
- Examine / dump memory of the proces
- Use IDA or Cutter or a dedicated debugger

# gdb

- The OG debugger under linux, console based
- Extendable, example of this is **GEF** (<https://gef.readthedocs.io/en/master/>)
- Extremely powerful
- Hard to learn

```
$r11 : 0x202
$r12 : 0x000055555555070 -> <start+0> xor ebp, ebp
$r13 : 0x00007fffffff180 -> 0x0000000000000001
$r14 : 0x0
$r15 : 0x0
$eflags: [zero carry PARITY adjust sign trap INTERRUPT direction overflow resume virtualx86 identification]
$cs: 0x0033 $ss: 0x002b $ds: 0x0000 $es: 0x0000 $fs: 0x0000 $gs: 0x0000

0x00007fffffff060 +0x0000: 0x0000000000000000 -> $rsp
0x00007fffffff068 +0x0008: 0x000055555555225 -> <_libc_csu_init+69> add rbx, 0x1
0x00007fffffff070 +0x0010: 0x0000000000000000
0x00007fffffff078 +0x0018: 0x0000000000000000
0x00007fffffff080 +0x0020: 0x000055555555210 -> <_libc_csu_init+0> push r15
0x00007fffffff088 +0x0028: 0x000055555555070 -> <start+0> xor ebp, ebp
0x00007fffffff090 +0x0030: 0x00007fffffff180 -> 0x0000000000000001
0x00007fffffff098 +0x0038: 0x0000555555556008 -> "Dont run 'strings' on this challenge, that is not [...]"

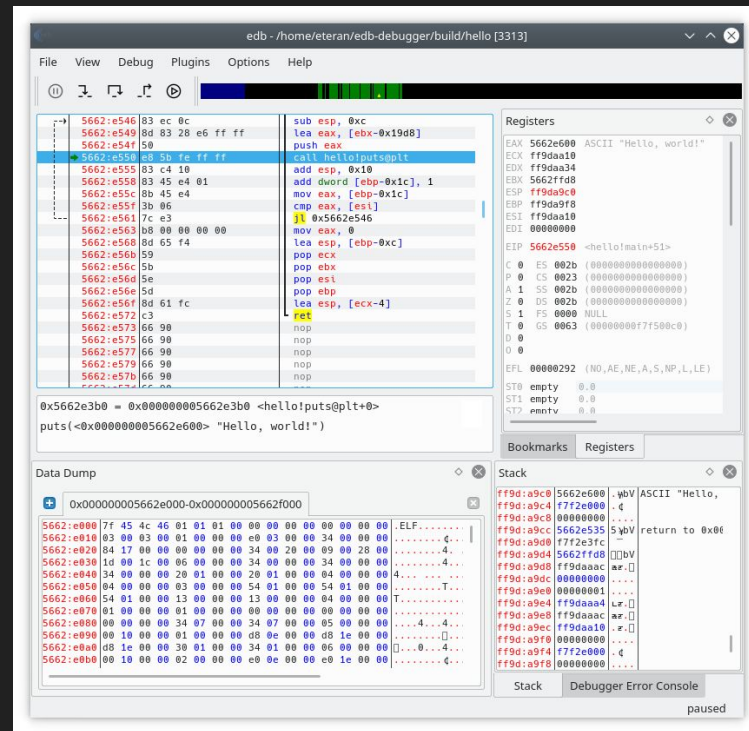
0x5555555515d <main+8> lea rax, [rip+0xea4] # 0x555555556008
0x55555555164 <main+15> mov QWORD PTR [rbp-0x8], rax
0x55555555168 <main+19> lea rdi, [rip+0xed7] # 0x555555556046
-> 0x5555555516f <main+26> call 0x55555555030 <puts@plt>
   0x555555555030 <puts@plt+0> jmp QWORD PTR [rip+0x2fe2] # 0x555555558018 <puts@got.plt>
   0x555555555036 <puts@plt+6> push 0x0
   0x55555555503b <puts@plt+11> jmp 0x55555555020
   0x555555555040 <fgets@plt+0> jmp QWORD PTR [rip+0x2fda] # 0x555555558020 <fgets@got.plt>
   0x555555555046 <fgets@plt+6> push 0x1
   0x55555555504b <fgets@plt+11> jmp 0x55555555020

puts@plt (
  $rdi = 0x0000555555556046 -> "Insert key:",
  $rsi = 0x00007fffffff188 -> 0x00007fffffff48a -> "/home/arjen/Downloads/baby",
  $rdx = 0x00007fffffff198 -> 0x00007fffffff4a5 -> "SHELL=/bin/zsh"
)

[#0] Id 1, Name: "baby", stopped, reason: SINGLE STEP
[#0] 0x5555555516f -> main()
```

# edb - Evan's debugger

- GUI based debugger
- Great alternative to GDB for most cases



CTF Thing to know

# XOR

- CTF developers love to XOR flags
- Compares on a binary level
- Output true (1) only when the values differ
- Easily implemented in any programming language (^)

0 0 1 1

0 1 0 1

0 1 1 0

0 1 0 1

0 0 1 1



**Wrapping up**

# Where to go from here?

- Reversing challenges on Hack The Box
- <https://beginners.re/> a guide to reverse engineering examining code and assembly
- <https://malwareunicorn.org/#/workshops> great workshops you can follow at home

## Additional challenges

- Advanced: <http://www.flare-on.com/> a yearly competition for RE
- <http://crackmes.cf/> archive of 1000s of crackme challenges

# You now know...

- how to perform basic static analysis
  - sha256sum, file, strings, upx, Ghidra
- how to get started with dynamic analysis
  - ltrace, strace,, edb
- how a **XOR** operation works
- **you can grab some challenge points!**

**Thanks! Questions?**

# Stick around

- There will be 2 breakout rooms
    - 1 to get started reversing
    - 2 to do some hacking on HTB
  - **credmp** will host the reversing breakout
  - **DutchPyro** will host the HTB hacking breakout
- 
- Let us know **in the chat** where you want to go and what your **HTB Username** is for the dedicated server!

# **Next Meetup**

**June 17th (6pm to 9pm)**

# Workshop

# Challenges to try

- BabyRE (easy)
- Exatlon (easy)
- Impossible Password (easy)
- Find the Secret Flag (medium)

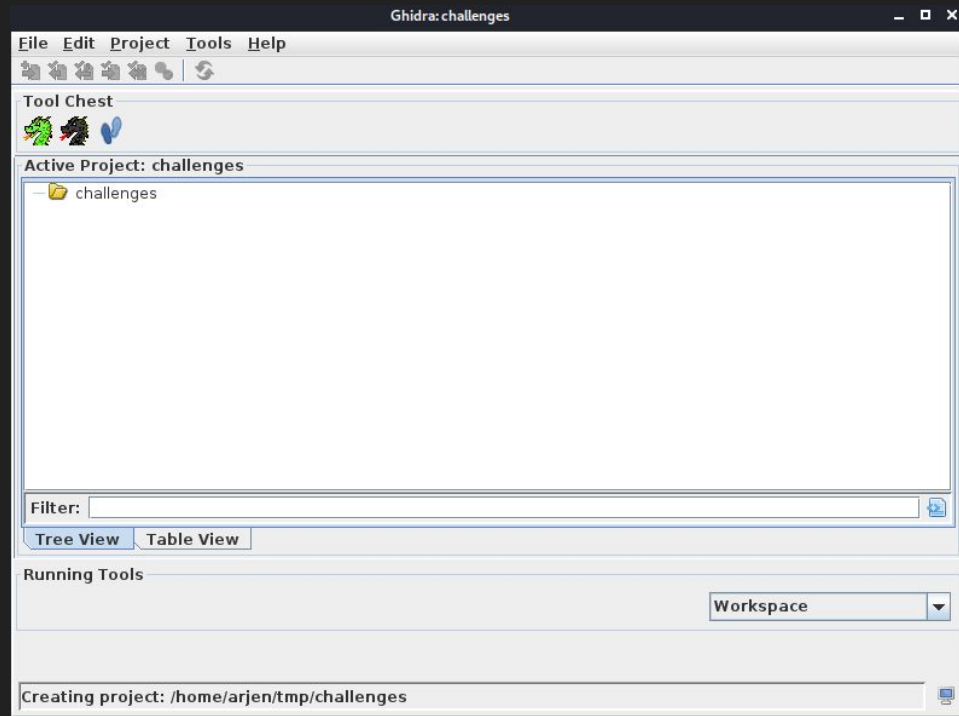


# Ghidra

- Download from <https://ghidra-sre.org/>
- Extract in your Downloads directory
- Run ghidraRun to start it

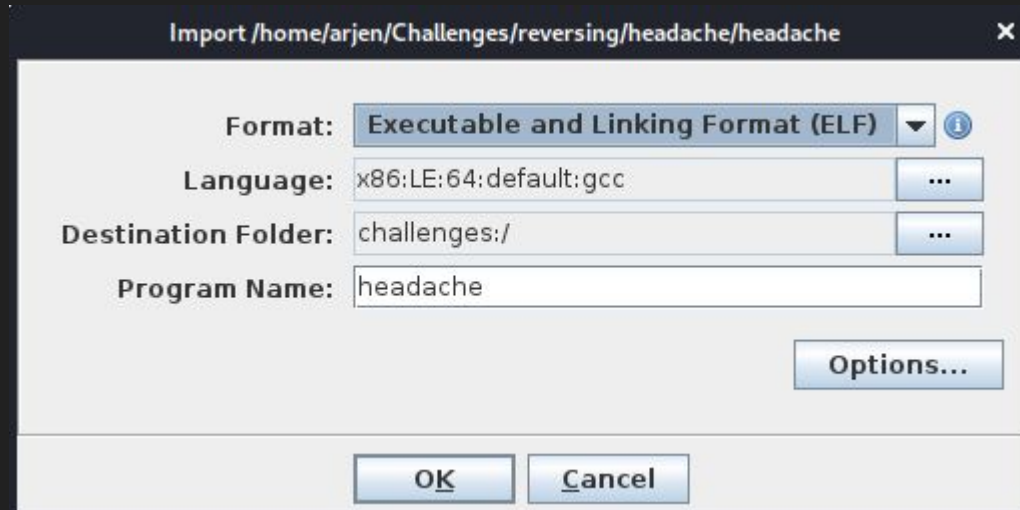
# Ghidra first step

- File > **New Project**
- Give it a name (**challenges**)
- Click on **Finish**



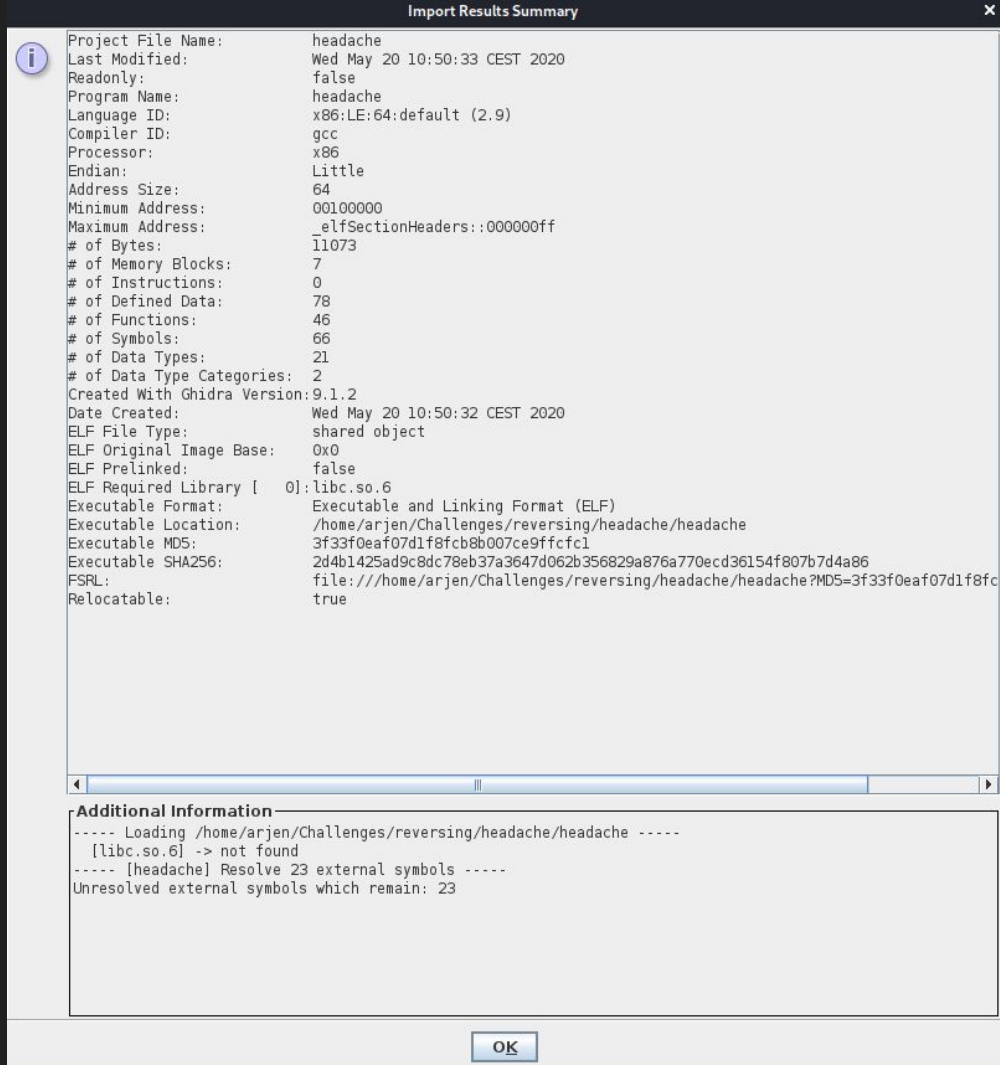
# Add a binary

- File > Import File
- Select the unpacked challenge file
- Hit Ok in the popup dialog



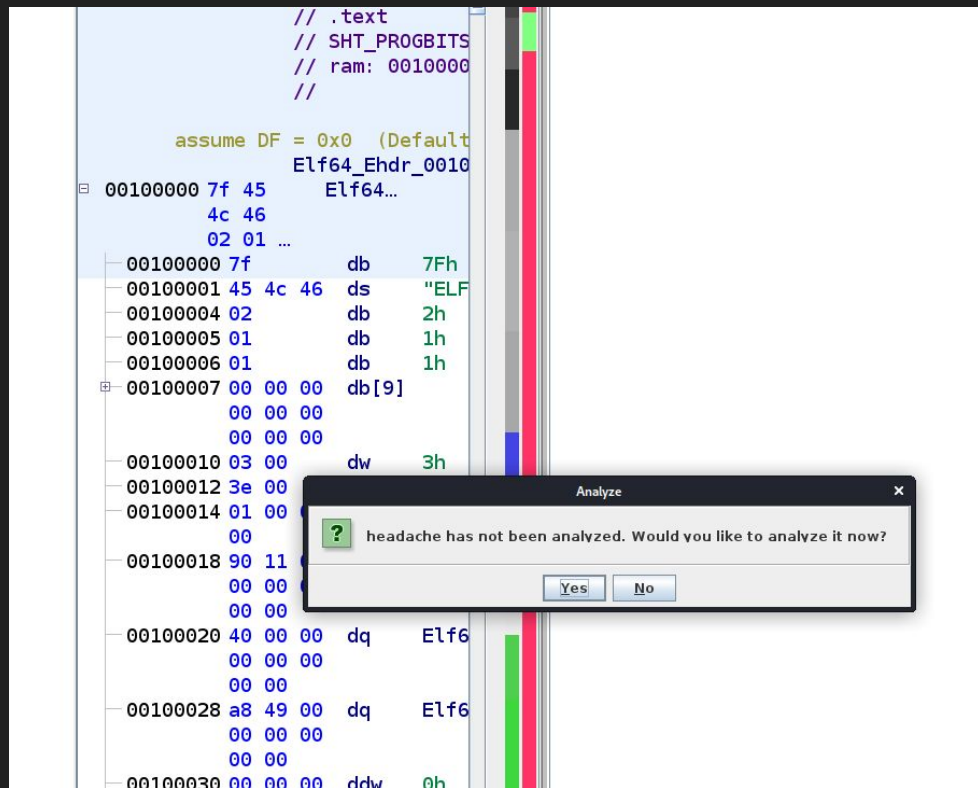
# Details

- Scan it, hit **OK**



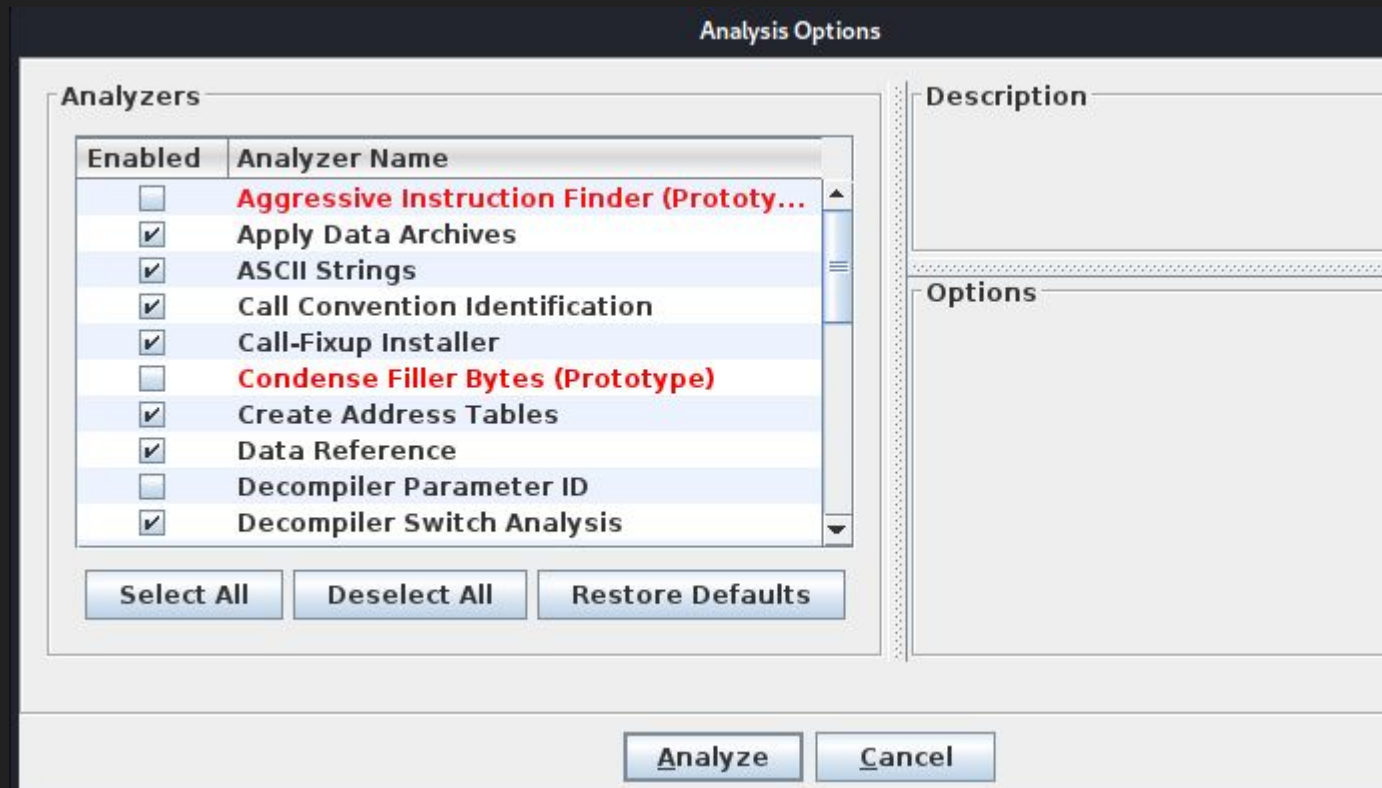
# Start analyzing

- Double click on the binary
- Select Yes to analyse your binary



# Analyze!

- Hit Analyze



**Start your journey....**