Reverse Engineering & Binary Exploitation

Binary exploitation

A binary is compiled code. When a programmer writes code in a language like C, the C code isn't what gets actually ran. It is compiled into a binary and the binary is run.

Binary exploitation is the process of exploiting the vulnerabilities in a binary.

Reverse engineering

Reverse engineering is the process of figuring out how something works. It is a critical part of binary exploitation, since most of the time you are just handed a binary without any clue as to what it does. You have to figure out how it works, so you can attack it.

Compilation and execution

Computers don’t understand C code. It has to first be compiled into an executable, which is then executed.

```
$ gcc program.c -o program
$ ./program
```

What do binaries contain?

Instructions for a computer to execute. They’re basically a bunch of numbers that the computer can understand but humans cannot. Check the hexdump with

```
$ hexdump program
```

It’s not possible for humans to understand it, so it’s represented as assembly language.

Registers

Registers are essentially places that the processor can store memory. You can think of them as buckets which the processor can store information in.

General Purpose Registers

These can be used for a variety of different things

rax	rdx	r8	r11	r14
rbx	rsi	r9	r12	r15
rcx	rdi	r10	r13	

Example

```
x = 2          mov     eax, 2
x + 2          add     eax, 2
```

Disassembly

```
$ gdb ./program
(gdb) disassemble main
```

Preferably, set disassembly flavour to intel before disassembling main

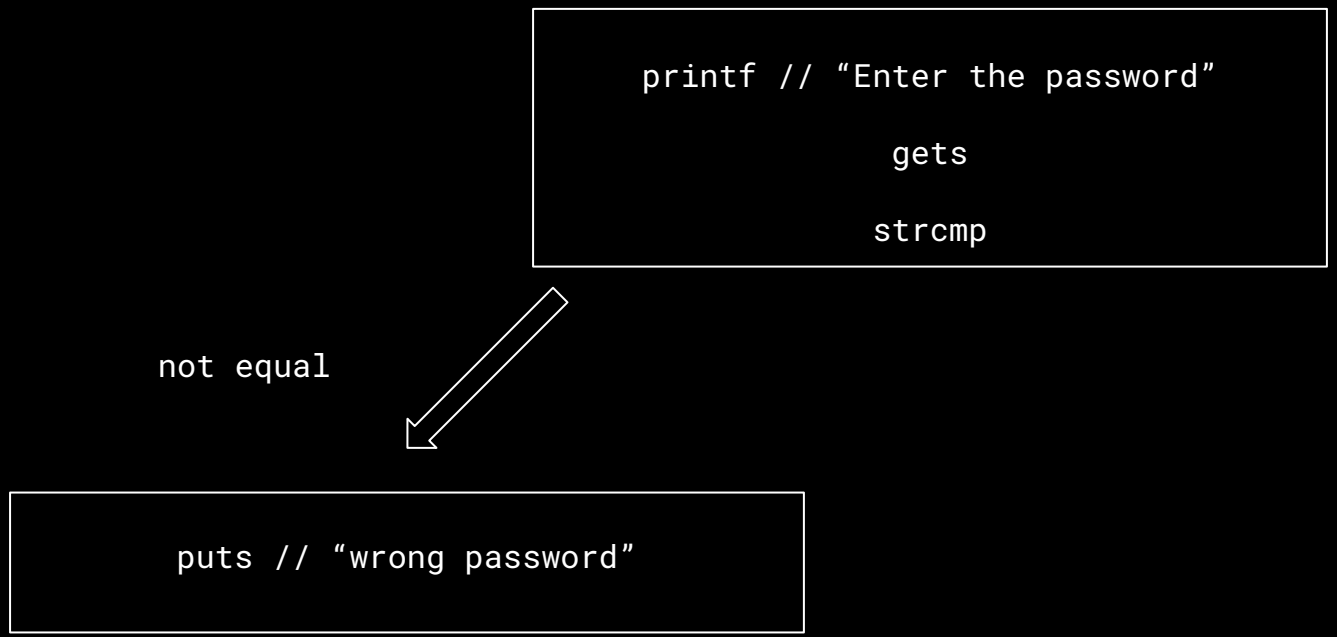
```
(gdb) set disassembly-flavor intel
(gdb) disassemble main
```

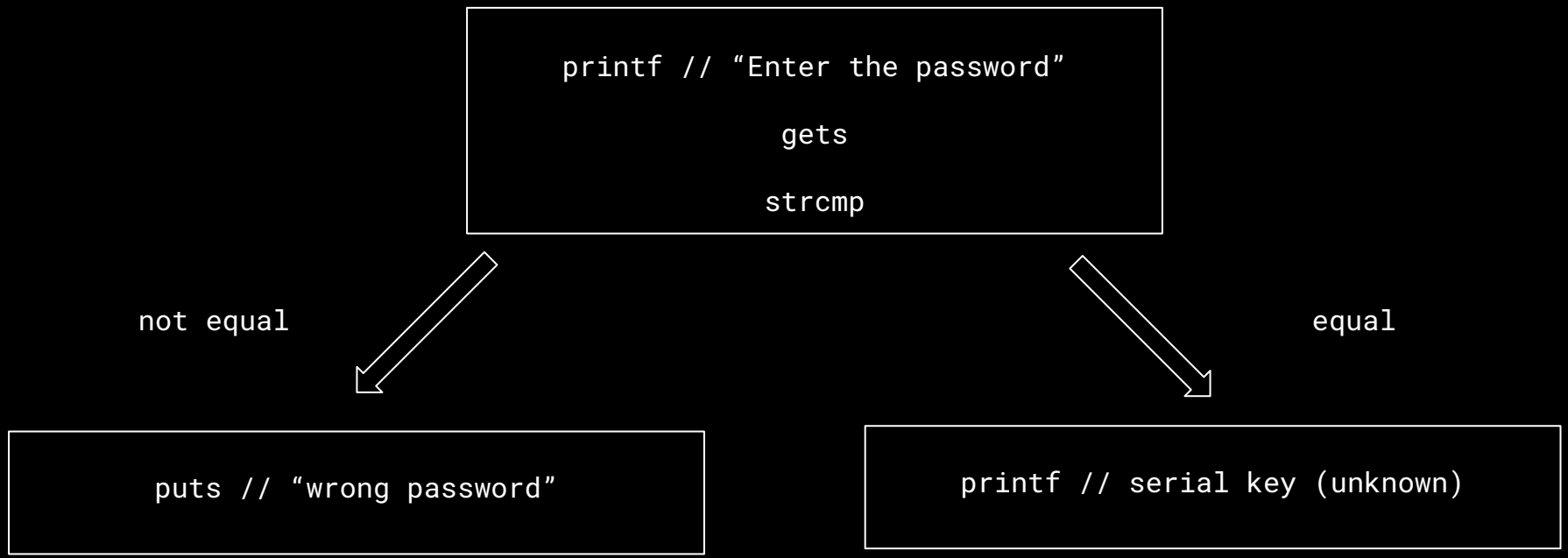
Cracking a binary

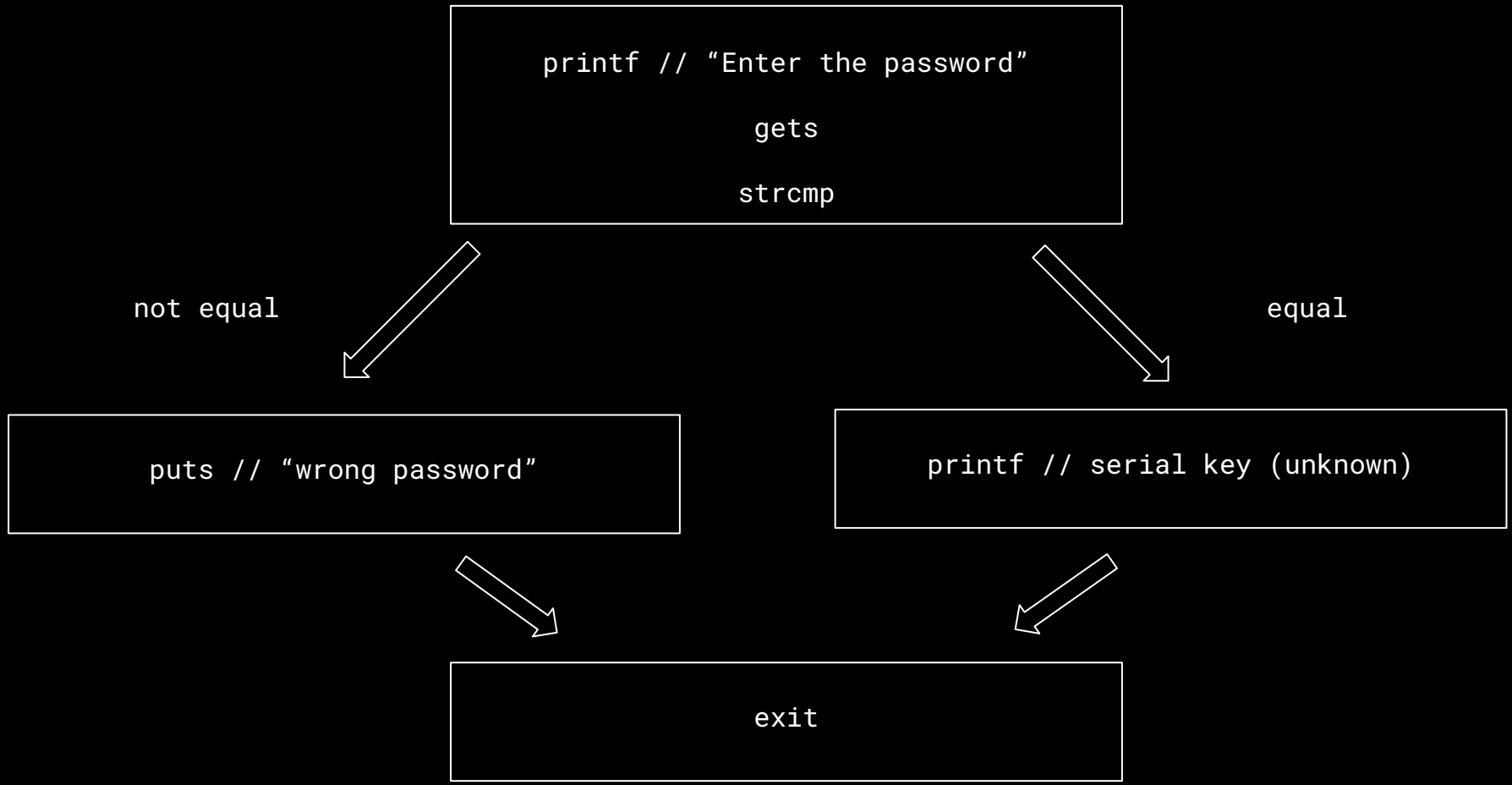
```
printf // “Enter the password”

gets

strcmp
```


printf // “Enter the password”
gets
strcmp
not equal
puts // “wrong password”


printf // “Enter the password”
gets
strcmp
not equal
equal
puts // “wrong password”
printf // serial key (unknown)


printf // “Enter the password”
gets
strcmp
not equal
equal
puts // “wrong password”
printf // serial key (unknown)
exit

Decompilation

A compiler takes source code like C, and converts it into machine code. A decompiler tries to do the opposite. It takes machine code and generates code that resembles it's source code.

Popular decompilers:

IDA (used in this workshop), Ghidra

Solving a CTF challenge

Buffer overflow

If a programmer wants to put ten bytes of data into a buffer that had only been allocated eight bytes of space, that type of action is allowed, even though it will most likely cause the program to crash.

This is known as buffer overflow, since the extra two bytes of data will overflow and spill out of the allocated memory, overwriting whatever happens to come next.

Exploiting a buffer overflow vulnerability

Resources

x86 Assembly:

https://www.youtube.com/watch?v=75gBFiFtAb8

Binary Exploitation:

https://www.youtube.com/watch?v=iyAyN3GFM7A&list=PLhixgUqwRTjxglIswKp9mpkfPNfHkzyeN

Websites:

https://guyinatuxedo.github.io/00-intro/index.html

Books:

The Art of Exploitation by Jon Erikson

Thank You