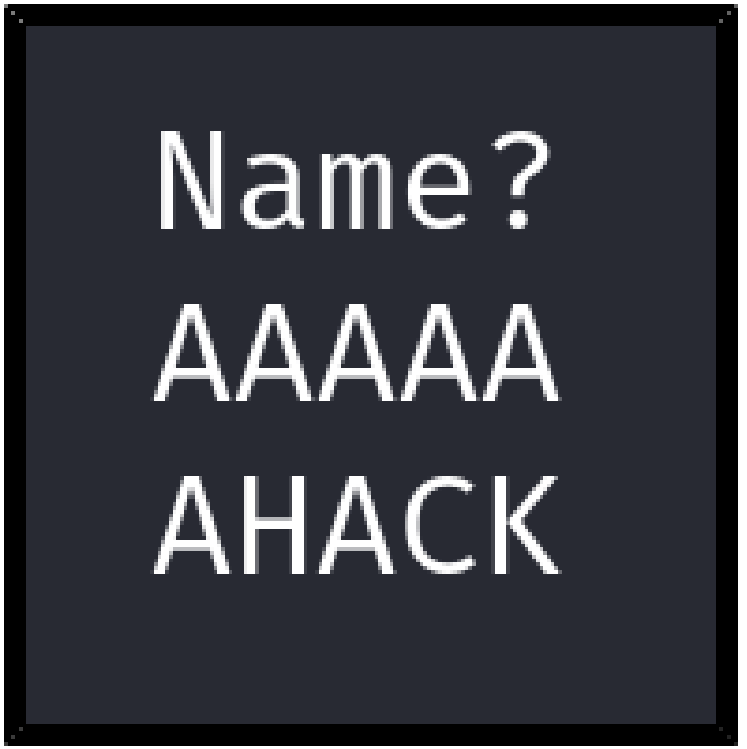
HackerFrogs Afterschool
Binary Hacking Basics: Part 2

Class:
Binary Hacking

Workshop Number:
AS-BIN-02

Document Version:
1.75

Special Requirements:
None



Welcome to HackerFrogs Afterschool!

This workshop is the second class to binary hacking

This is what we'll be covering in this session...

Buffer Overflow - Ret2Win

We will cover the following topics in this session:

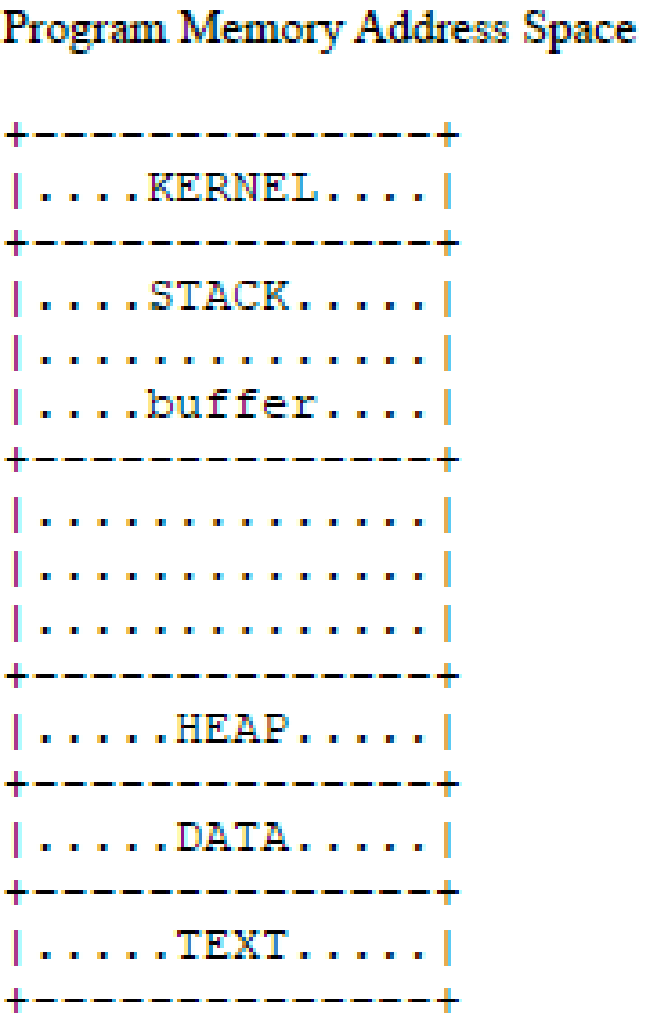
- Controlling program flow

- Pico: Buffer Overflow 1

- Pico: Heap 2

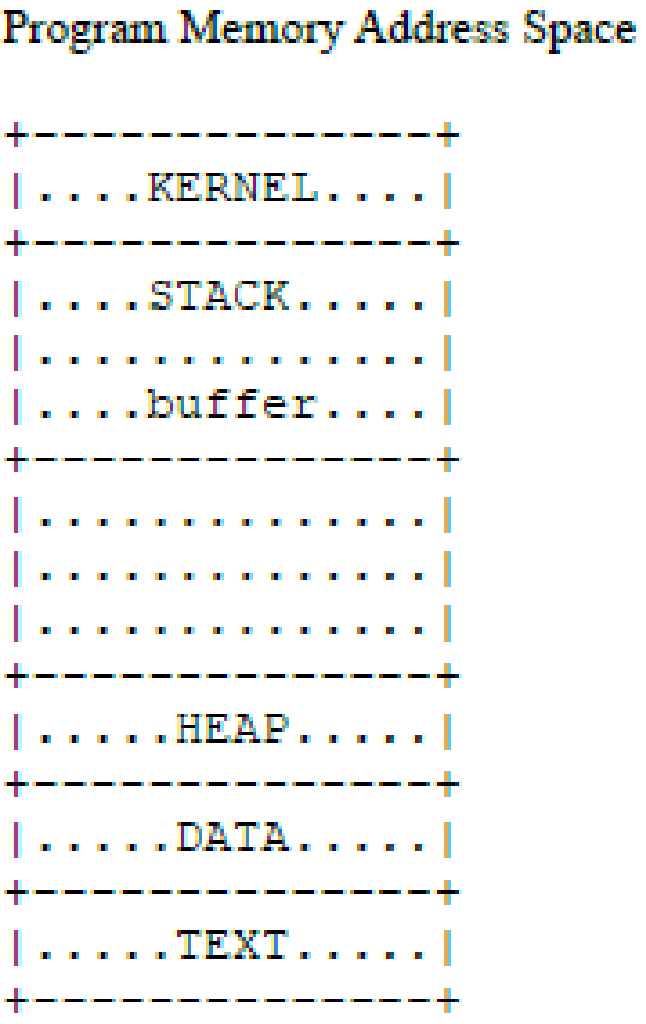
Controlling Program Flow

Through buffer overflow, it is possible to redirect the flow of program execution. We know that stack buffer overflows expand up towards lower memory addresses



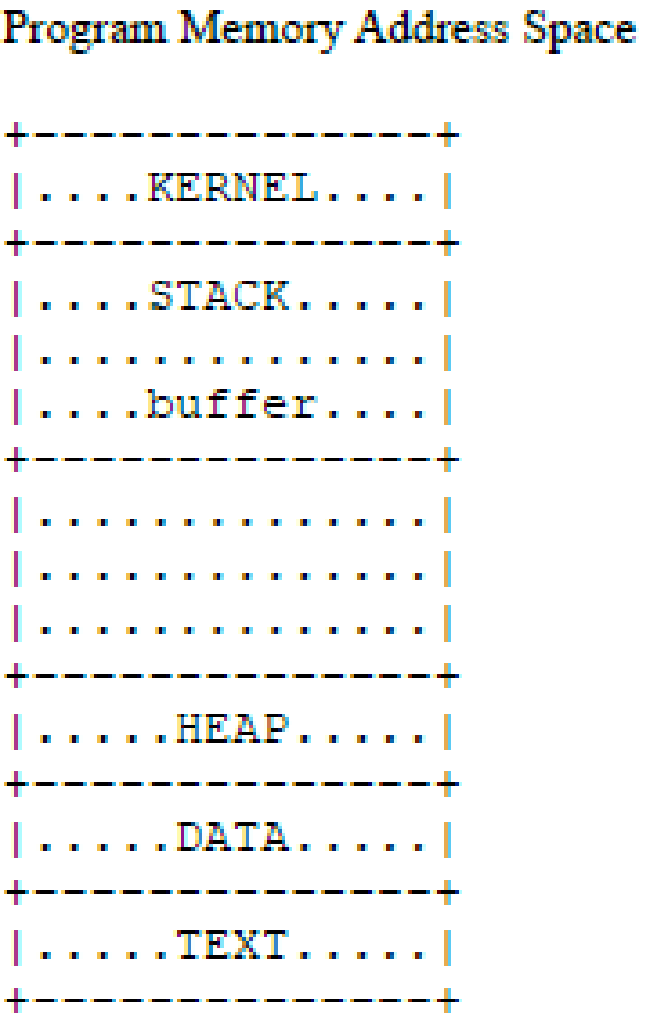
Controlling Program Flow

After sufficient overflow, the first CPU register to be overwritten is the BP (base pointer), and the second is the IP (instruction pointer)



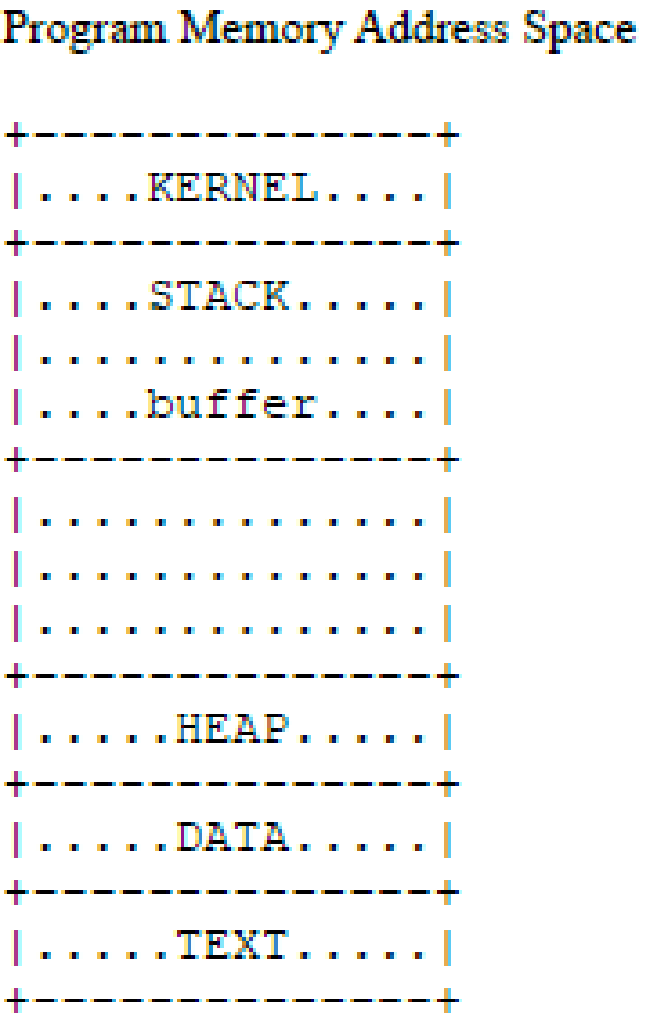
The Instruction Pointer

The instruction pointer (IP) is a CPU register which contains the memory address of the next instructions to be executed



Controlling Program Flow

We can redirect program execution with buffer overflow by overwriting the BP and IP, replacing the contents of the IP with the memory address of whatever instructions we want to execute



Ret2Win /w
Pico Buffer Overflow 1

To learn more about buffer overflows, let's look at a challenge in Pico CTF:

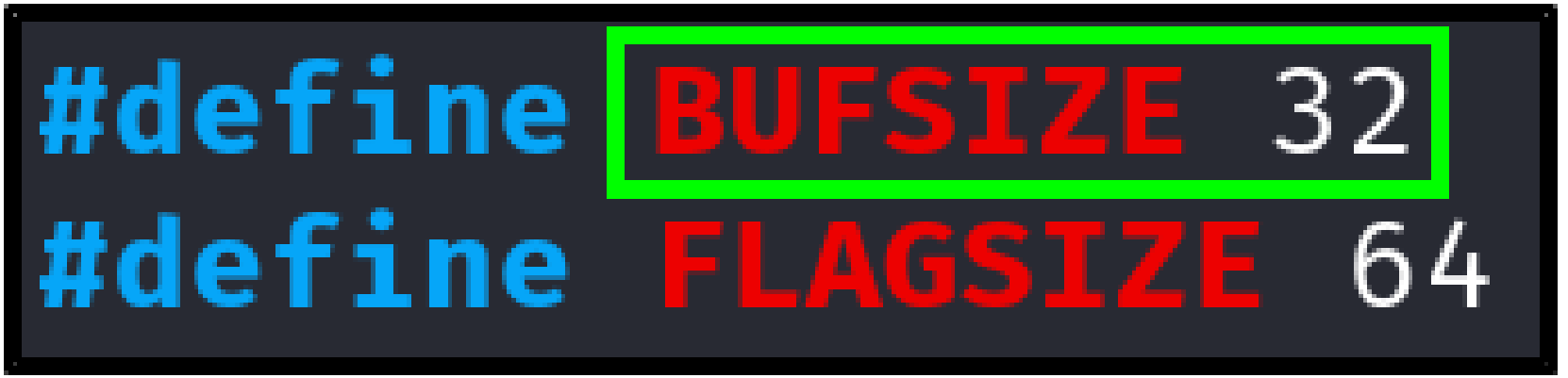
https://play.picoctf.org/practice/challenge/258

The Win Function

```
0×08049350    4     101 sym.__libc_csu_init
0×080491f6    3     139 sym.win
0×08049120    1       5 sym._dl_relocate_static_pie
```

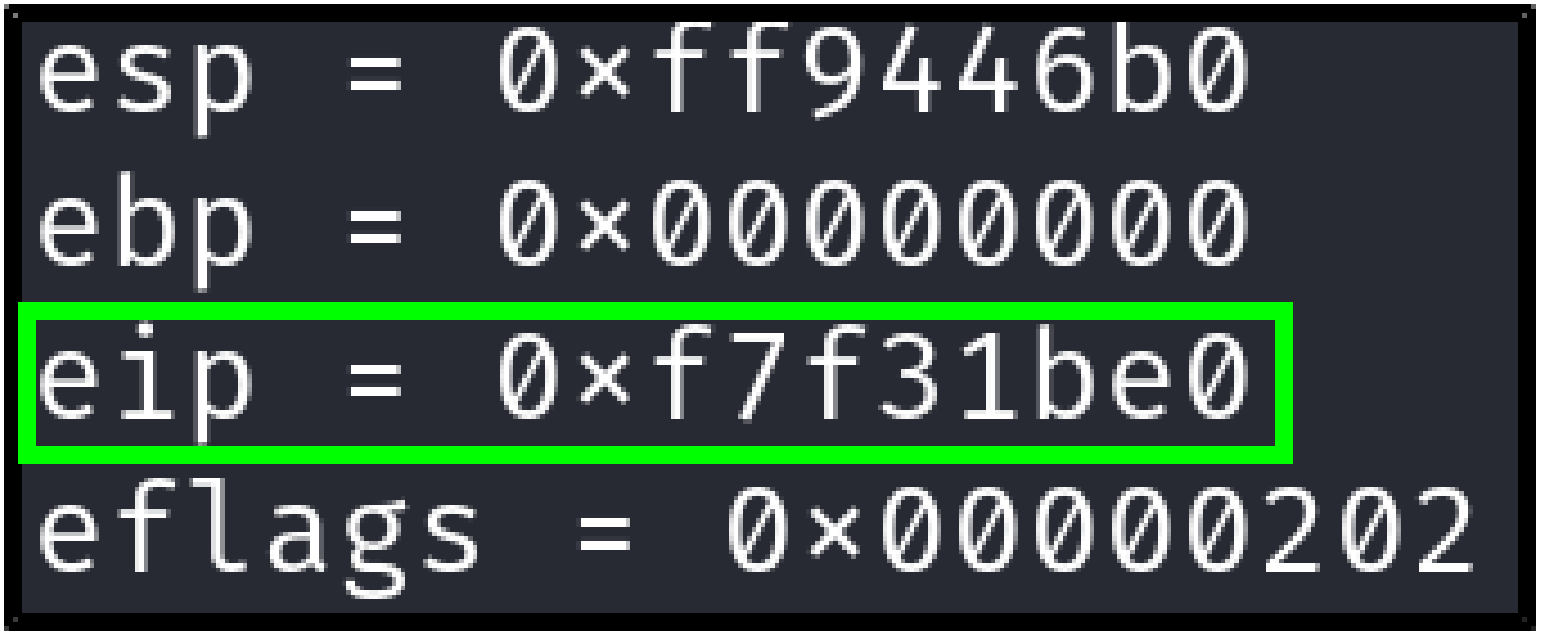
In this challenge, we can read the flag if we redirect program execution to the sym.win function, and our debugger lets us know where the function exists in program memory

The Win Function



We know that the user input buffer is 32 bytes long, and input in excess of 32 bytes will overflow the stack buffer

The Win Function



So the goal is to send 32 bytes, then figure out how many more bytes we need to send to the program to overflow the BP, then the IP register

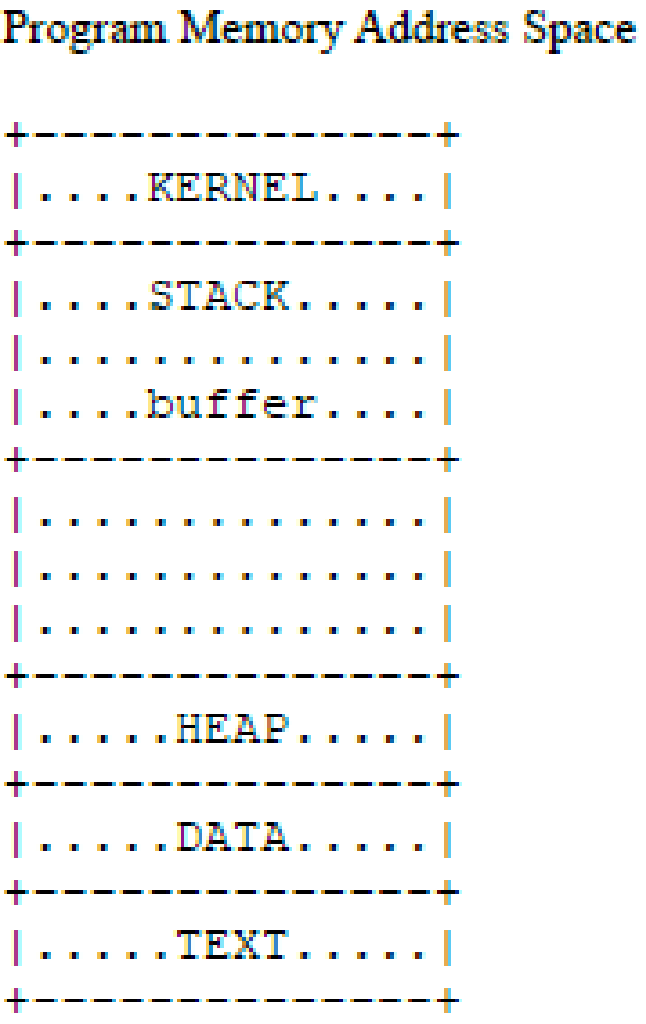
Heap-Based Ret2Win /w Pico Heap 2

To learn more about buffer overflows, let's look at another challenge in Pico CTF:

https://play.picoctf.org/practice/challenge/435

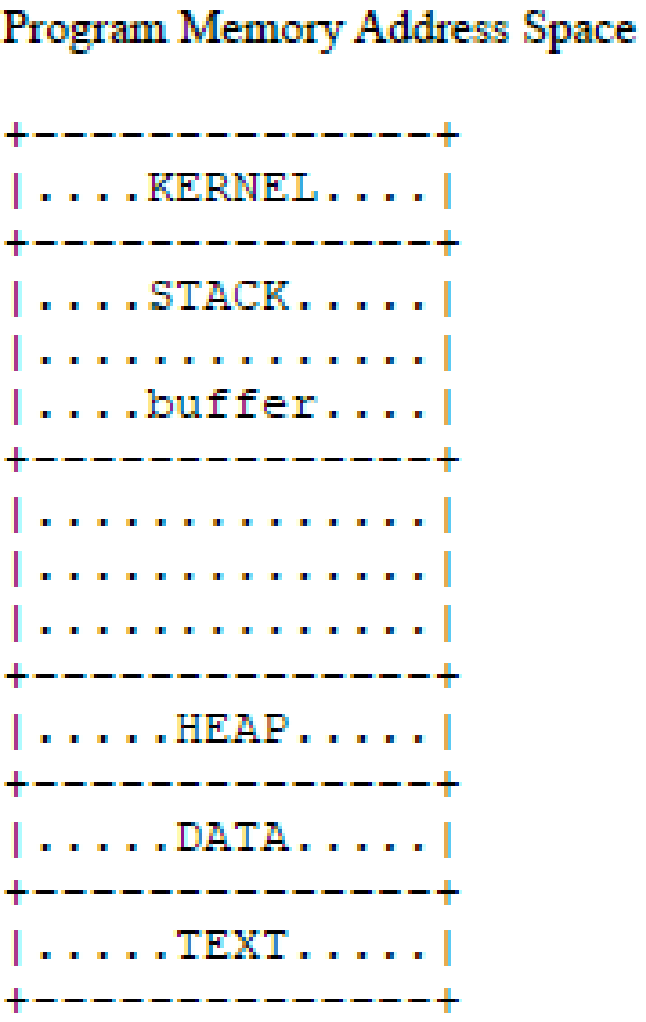
What is Heap Memory?

Heap memory is the section of program memory which holds variables that are intentionally allocated through C functions like **malloc**



What is Heap Memory?

Buffer overflows that happen in heap memory are different from stack overflows
in that they may be used to overwrite the values of other heap objects



Heap Buffer Overflow Targets

```
input_data = malloc(5);
strncpy(input_data, "pico", 5);
x = malloc(5);
strncpy(x, "bico", 5);
```

In this program, since the variables `input_data` and `x` are defined with `malloc` function, these memory buffers are created in heap memory

Heap Buffer Overflow Targets

```
input_data = malloc(5);
strncpy(input_data, "pico", 5);
x = malloc(5);
strncpy(x, "bico", 5);
```

Since `input_data` is defined first, and is user-controlled, the value of `x` can be overwritten through use of heap buffer overflow

Summary

Let's review the concepts we learned in today's workshop:

Until Next Time, HackerFrogs!