**What a Buffer Overflow is and how to leverage it**

**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

The buffer space goes downward. If you are properly sanitizing your buffer space, by the time the information reaches the EBP it should stop and contain the characters that you are sending. However, if you have a buffer overflow attack you reach over the EBP and into the EIP.

A screenshot of a cell phone

Description automatically generated

The EIP is a Pointer/Return Address. We can use this address to point to directions that we instruct. The directions will be malicious code that will give us a reverse shell. If you can write past the EBP to the EIP you can take control of the stack.

Steps to Conduct a Buffer Overflow:

1. Spiking: A method that we use to find the vulnerable part of the program
2. Fuzzing: Throwing a bunch of characters at a program to see if we can break it
3. Finding the Offset: Finding out at what point we did break it
4. Use the Offset to overwrite the EIP
5. Finding the Bad Characters
6. Finding the Right Module
7. Generating Shellcode
8. Root

**Spiking multiple commands (e.g. SRUN, TRUN, GMON) to find vulnerabilities:**

Kali:

A screenshot of a computer

Description automatically generated

GMON (Can be any selection)

Spiking Script for gmon: Name: gmon.spk

[

s\_readline();

s\_string(“GMON”);

s\_string\_variable(“0”):

]



A screenshot of a cell phone

Description automatically generated

**Fuzzing Tools (Building a Python Script):**

A screenshot of a cell phone

Description automatically generated

A close up of text on a white background

Description automatically generated

­