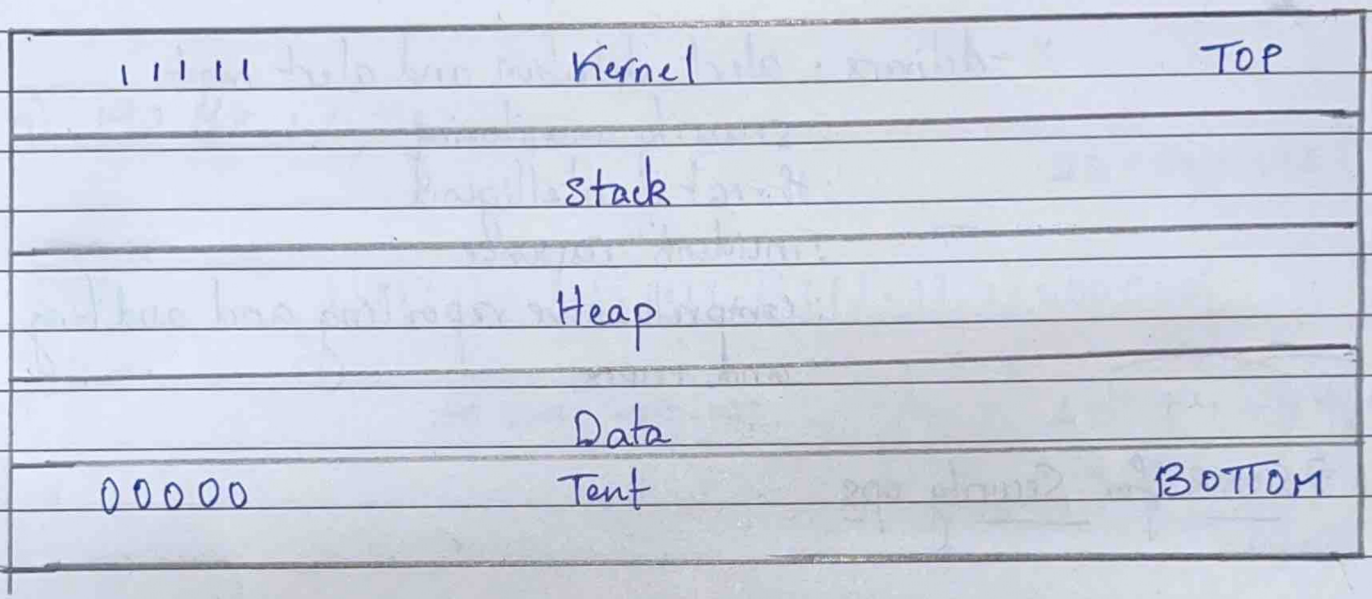
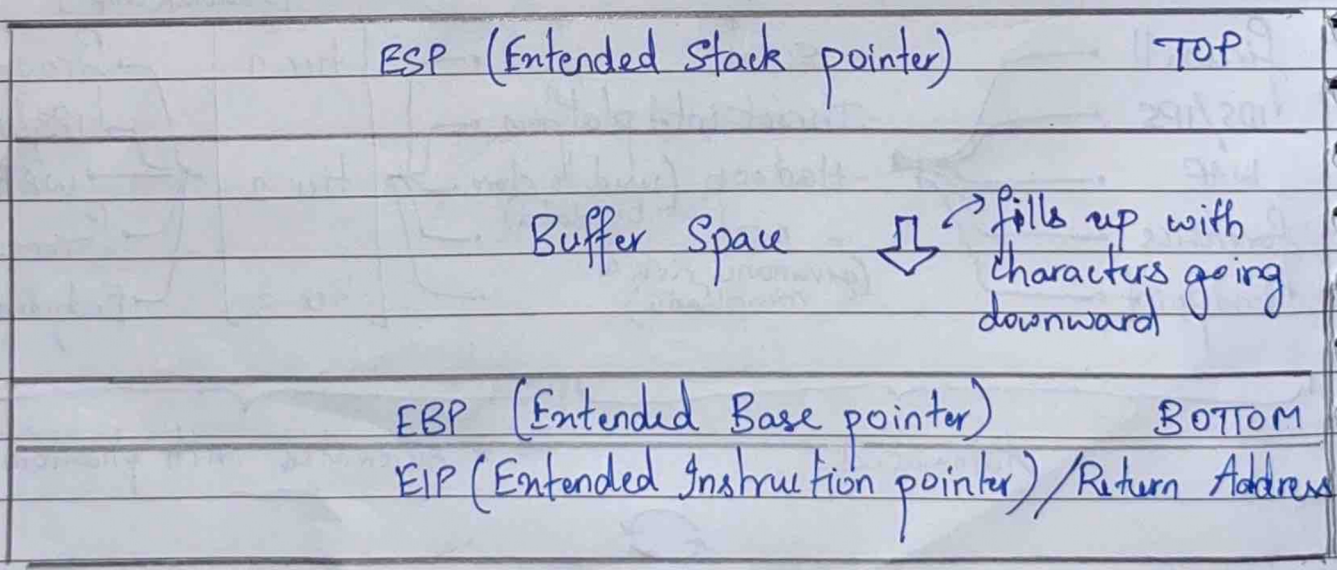
⇒ Buffer Overflows Explained

Anatomy of a memory



Anatomy of the Stack



↳ Now here, if my Buffer space fills with characters, it should fill downwards and stop when it reaches EBP.

- But in Buffer Overflow attacks, what happens is, you actually overflow the buffer space you're using and reach over EBP and into something called the EIP.

- Now EIP in layoman terms is your pointer address or return address and can be pointed to directions we instruct.

- These directions are going to be malicious codes that gives us a reverse shell.

- So what's happening in the stack is that you're overflowing Buffer space. So if you can write over the buffer space and all the way down to the EIP, you can control the stack, the pointer and eventually gain reverse shell which will lead to root.

↳ steps to conduct a buffer overflow

1) Spiking: method used to find a vulnurable part of a program
2) Fuzzing: Sending a bunch of characters out of a program to see if we can break it.
3) Finding the offset: Once fuzzing is done, we find the breaking point or the offset.
4) Overwriting the EIP: We use the offset to overwrite the EIP
5) finding bad characters: Once step 4's done, we need to do a few house cleanup things like (5) and (6)
6) find the right module:
7) Generating shell code: Once we're done with (5) & (6) we can generate malicious shell code to get reverse shell, pointing EIP to shell to gain access.
8) Root!