Solutions Class Exercise—ROP and CFI
ENEE 457
9/16/20

1. In this exercise, we will take a closer look at how to launch a Return Oriented Programming (ROP) attack that starts a shell. To do this, we need to implement the assembly instructions corresponding to the shellcode. Actually, we will only have time to implement a few of these instructions. Specifically, the first steps of launching the shell are to (1) zero out the %eax register (2) load string "/bin" (0x6e69622f in hex) somewhere in memory---say into memory address 0x404 (3) load string "//sh" (0x68732f2f in hex) to the consecutive location in memory.

We are given the following gadgets.

```
0x9b4550: mov [%ebx] %ecx
          ret
```

```
0x9c27e0: pop %ecx
          ret
```

```
0x9b8d20: xor %eax %eax
          ret
```

```
0x9c61b0: pop %ebx
          ret
```

Assume that we are currently executing a function func(). What should the stack look like right before "ret" is called at the end of the execution of func() in order for steps (1) (2) and (3) to be completed upon return from func()?

1 buf
2 stored %ebp
3 0x9b8d20 [zero out %eax register]
4 0x9c61b0 [pop %ebx, we want %ebx to contain address 0x404]
5 0x404
6 0x9c27e0 [pop %ecx, we want %ecx to contain "/bin"]
7 0x6e69622f [hex for string "/bin"]
8 0x9b4550 [move contents of %ecx to memory location pointed to by %ebx]
9 0x9c61b0 [pop %ebx, we want %ebx to contain address 0x408, next consecute addr]
10 0x408
11 0x9c27e0 [pop %ecx, we want %ecx to contain "//sh"]
12 0x68732f2f [hex for string "//sh"]
13 0x9b4550 [move contents of %ecx to memory location pointed to by %ebx]

**Note that we can switch the order and do 6,7 before 4,5. Similarly, we can do 11,12 before 9,10