# CSO Assignment 2
# Question 3

**All information can be obtained by running the hardinfo command on ubuntu systems.**

**Operating System:**

| | |
|---|---|
| Kernel | : Linux 5.13.0-51-generic (x86_64) |
| Distribution | : Ubuntu 20.04.2 LTS |

**Kernel Modules:**

Name and description of the first 20 kernel modules (due to space constraints) is as follows:

| | |
|---|---|
| binfmt_misc | |
| xt_state | : ip[6]_tables connection tracking state match module |
| ipt_REJECT | : Xtables: packet &quot;rejection&quot; target for IPv4 |
| nf_reject_ipv4 | |
| nf_nat_h323 | : H.323 NAT helper |
| nf_conntrack_h323 | : H.323 connection tracking helper |
| nf_nat_pptp | : Netfilter NAT helper module for PPTP |
| nf_conntrack_pptp | : Netfilter connection tracking helper module for PPTP |
| nf_nat_tftp | : TFTP NAT helper |
| nf_conntrack_tftp | : TFTP connection tracking helper |
| nf_nat_sip | : SIP NAT helper |
| nf_conntrack_sip | : SIP connection tracking helper |
| nf_nat_irc | : IRC (DCC) NAT helper |
| nf_conntrack_irc | : IRC (DCC) connection tracking helper |
| nf_nat_ftp | : ftp NAT helper |
| nf_conntrack_ftp | : ftp connection tracking helper |
| ccm | : Counter with CBC MAC |
| rfcomm | : Bluetooth RFCOMM ver 1.11 |
| veth | : Virtual Ethernet Tunnel |
| xt_nat | : SNAT and DNAT targets support |

There are 227 kernel modules

**File Systems:**

loop 0 to loop 39, developer kernel file systems and temporary file systems
48 file systems

**Processor:**

multi core (8 cores) processor, all of the cores are Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz and frequency 3600.00 MHz

**Memory:**

Total Memory 7900700 KiB, Free Memory 192180 KiB, MemAvailable 2665616 KiB

**PCI Devices:**

| | |
|---|---|
| Host bridge | : Intel Corporation Device 8a12 (rev 03) |
| VGA compatible controller | : Intel Corporation Device 8a56 (rev 07) (prog-if 00 [VGA controller]) |
| Signal processing controller | : Intel Corporation Device 8a03 (rev 03) |
| USB controller | : Intel Corporation Ice Lake Thunderbolt 3 USB Controller (rev 03) (prog-if 30 [XHCI]) |
| Serial controller | : Intel Corporation Device 34fc (rev 30) (prog-if 00 [8250]) |

USB controller : Intel Corporation Ice Lake-LP USB 3.1 xHCI Host Controller (rev 30) (prog-if 30 [XHCI])
RAM memory : Intel Corporation Device 34ef (rev 30)
Network controller : Intel Corporation Killer Wi-Fi 6 AX1650i 160MHz Wireless Network Adapter (201NGW) (rev 30)

**USB Devices:**
Realtek Semiconductor Corp. RTL8153 Gigabit Ethernet Adapter, Linux Foundation 3.0 root hub, Microdia Integrated_Webcam_HD, Shenzhen Goodix Technology Co.,Ltd. FingerPrint Intel Corp., Linux Foundation 2.0 root hub, Linux Foundation 3.0 root hub, Linux Foundation 2.0 root hub

**Battery:**
Capacity: 100 / Full, Battery Technology: Li-polymer, Manufacturer: SMP

**Sensors:**
multiple thermal/thermal_zone sensors, multiple coretemp/temp sensors, one fan, battery sensor

**Storage:**
160 GB total
146 GB used
5.4 GB left

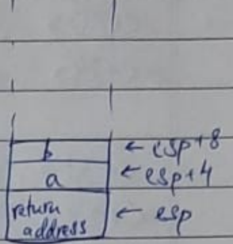
**DMI:**
Name: Inspiron 5400 2n1, -BIOS- Version: 1.3.1, -Board- Version: A00

**CPU Blowfish Benchmarking:**
Results for Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz:
Threads: 8
Machine:
Board: Dell Inc. 0032PT
CPU Name: Intel® Core (TM) i5-1035G1 CPU @ 1.00 GHz
CPU Descripton: 1 physical processor, 4 cores; 8 threads
CPU Config: 8x 3600.00 MHz
Threads Available: 8
OpenGL Renderer: Mesa Intel® UHD Graphics (ICL GT1)
Memory: 7900700 kiB

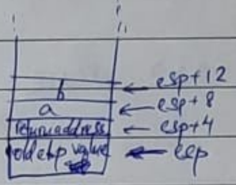
CSO Assignment 2
Question 4

Arun Khattawala
2020113008

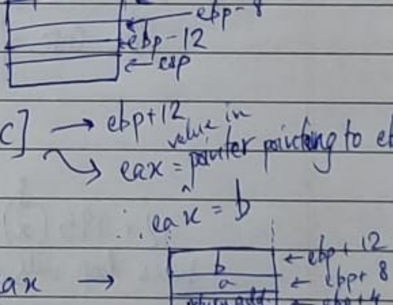
When assembly code is called w/ a & b,
it pushes b, then a & then the $\overset{\text{core}}{\text{code}}$ is executed.
∴ $\overset{\text{upon}}{\text{calling}}$ assemblycode, stack status is:



Now, let's look at the assembly code in detail w/ comments & stack status at each point:
assemblycode:
<+0>: push ebp → ~~stack~~

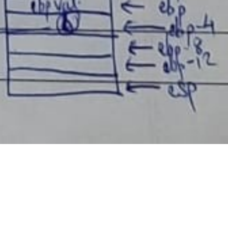


<+1>: mov ebp, esp → move esp to ebp. ∴ now, ebp = new ebp = esp
<+2>: sub esp, 0X10 → ~~esp~~ subtract 0x10 (16 in decimal) from esp ⇒ esp = esp - 16
∴ esp moves down 16 from prev. stack status
∴



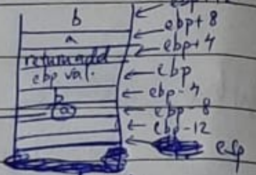
<+6>: mov eax, DWORD PTR [ebp+0xC] → ebp+12
↳ eax = $\overset{\text{value in}}{\text{pointer}}$ pointing to ebp+12
∴ eax = b

<+9>: mov DWORD PTR [ebp - 0x4], eax →



Date
Page

<+12> : mov eax, DWORD PTR [ebp+0x8] → eax = a
<+15> : mov DWORDPTR [ebp-0x8], eax →



<+18> : jmp 0x50c <asm2+31> → jump to line <+31>
<+20> : add DWORDPTR [ebp-0x4], 0x1 → *(ebp-4)+1 ⇒ b = b+1
<+24> : add DWORDPTR [ebp-0x8], 0xaf → *(ebp-8)+175 ⇒ a = a+175

0xaf in decimal

<+31> : cmp DWORDPTR [ebp-0x8], 0xa3d3 → 0xa3d3 in decimal
<+38> : jle 0x501 <asm2+20> ⟶ if a ≤ 41939, jump to line <+20>

<+40> : mov eax, DWORDPTR [ebp-0x4] → mov b to eax
∴ eax = b

<+43> : leave → deallocate dynamic stack that has been allocated by doing the following:
esp = ebp & pop ebp
↳ empty the stack

<+44> : ret

Now, we have understood how the code works along w/ the stack status at each stage of the code.
If a = 0xC & b = 0x15, a = 12 & b = 21 (in decimal).
∴ The core of the code is:

```
while (a ≤ 41939){
    b++;
    a += 175;} &
```

when we exit this return b.

∴ $12 + 175(x) > 41939$ & $12 + 175(x-1) < 41939$ where x is no. of times loop will run.

∴ $x = 240$.

∴ b++ happens 240 times.
∴ $\underline{b} = 21 + 240 = \underline{\underline{261}}$. → return value ↓ in hexadecimal,
return value = 0x105.

classmate
Date ______
Page ______

CSO Assignment 2
Question 5

Aum Khatlawala
2020113008

a) To tackle this problem I compiled the given q5.c file into an executable (↳ a.out) & ran the file a.out command & file q5.out command one after the other.
One small difference I noticed was that ~~the~~ q5.out was made for GNU/Linux 4.4.0 while the ~~file~~ a.out [illegible] was made for GNU/Linux 3.2.0. [illegible]
But the main difference I noticed led [illegible] me to the mistake in q5.out. The interpreter ~~of~~ [illegible] corresponding to q5.out is "./libc6-amd64_2.27-3ubuntu1_i386.ld" which means that the .ld file reqd. by the interpreter is being searched for in the same directory ("./") as the q5.out file & this doesn't exist. The interpreter for the a.out is "/lib64/ld-linux-x86-64.so.2" and this seems to work correctly as it gives the correct output.
To reconfirm the mistake I found to be true, I ran the file <filename> command on the other a.out files in my system & they also gave the output as "/lib64/ld-linux-x86-64.so.2".

b) The q5.out has the following properties:
1) It is an ELF 64-bit LSB shared object.
↳ Executable & Linkable format
↳ Least significant bit (Little Endian)
→ indivisible unit which can be generated from one or more relocatable objects
2) It is dynamically linked.
↳ a pointer to the file being linked in the executable is included in the executable & not the contents of the file being linked
3) It is made for Linux Kernel 4.4.0.