Name :- Ronak Kumar
Roll No :- 2015080

FCS Assignment 3

Part I

A1)
I created my public and private GPG key pair (4096 bit). The keys are in the Part I Folder.

Commands :-
gpg --gen-key
gpg -a --export ronak15080@iiitd.ac.in > mypubkey.asc
gpg -a --export ronak15080@iiitd.ac.in > mysecretkey.asc

The mypubkey.asc is the public GPG key and the mysecretkey.asc is the private GPG key.

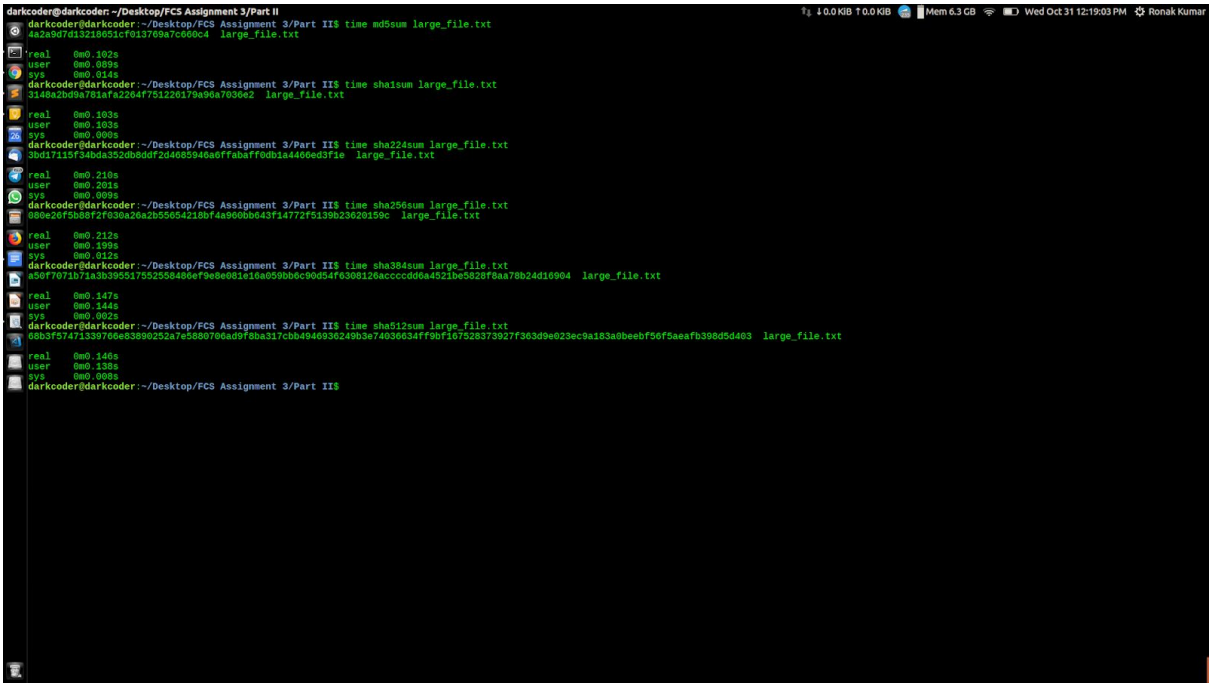
A2)
The files are in the Part I Folder

Commands :-
gpg --armor -e -r ronak15080@iiitd.ac.in file.txt
gpg --sign --default-key ronak15080@iiitd.ac.in file.txt.asc

A3)
No, it is not possible to decrypt the file since the file is first encrypted with the GPG public key whose private key I don't know. Also, since the paraphrase used during the signature phase is not available, it is not possible to decrypt the file.txt from the signature file.

Part II

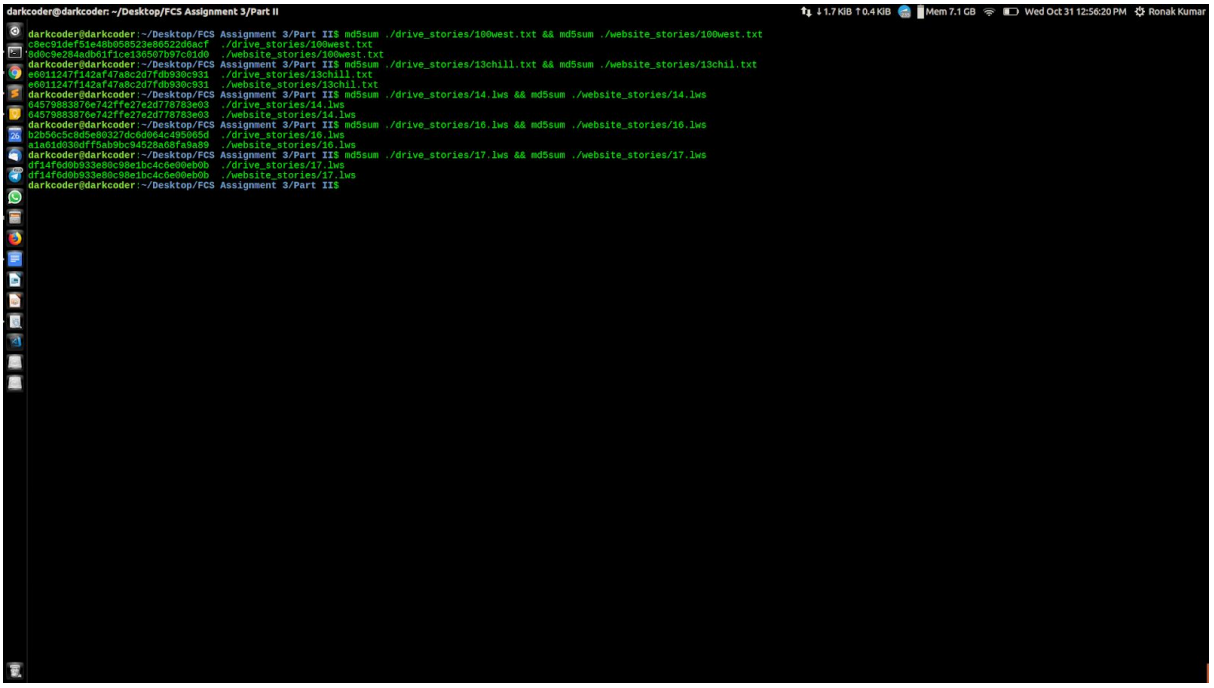
A1)
Screenshot :-



MD5 is the fastest hash algorithm among all the hashing techniques. SHA256 takes the longest time among all of them. SHA1 takes slight more time than MD5. However, this might fluctuate with CPU factors.

A2)

a)



As, in the above screenshot, I computed the MD5 Checksum of the respective files. The files which are having the same checksum are 13chill.txt, 14.lws, 17.lws i.e these are the files that are not tampered. So, the files that were modified by the third party are 100west.txt & 16.lws as their checksum is different.

The MD5 checksum technique is the fast method for finding any tamperness / modification in the data since, if there is any change in the files then the checksums don't match. I used the MD5 approach as it is the fastest and a reliable source for identifying tamperness and verifying the integrity of the files. The large file taken as input is the same given in the previous Homework 2

b)

Yes, it is possible to modify the content of the file without changing the value of the checksum. A good example for this would be the use of MD5 and the SHA1 algorithms. Both the algorithms can remain the same, even if change the content. This is why it is better to move to more secure hash algorithms like SHA-256 as MD5 and SHA1 are considered to be deprecated in terms of verifying the integrity of the file.

c)

The two properties that may get violated in part b are collision resistance and second preimage resistance. This is because even if the two files have different contents in them still the SHA and the MD5 hash turns out to be the same. Hence it becomes easy to find two hashes with the same value thereby violating the collision resistance property. Also, the second property is violated in the same way since it says that it should be hard to find a different input with the same hash value although now it becomes easy to find as seen in the first example.

Part III

A1)

The code is given in the Part III Folder

A2)

The code is given in the Part III Folder

A3)

The permissions for the file /etc/passwd is "-rw-r--r-- 1 root root 2298 Oct 10 11:00". I ran the ls -l | grep passwd command in the /etc directory for getting the

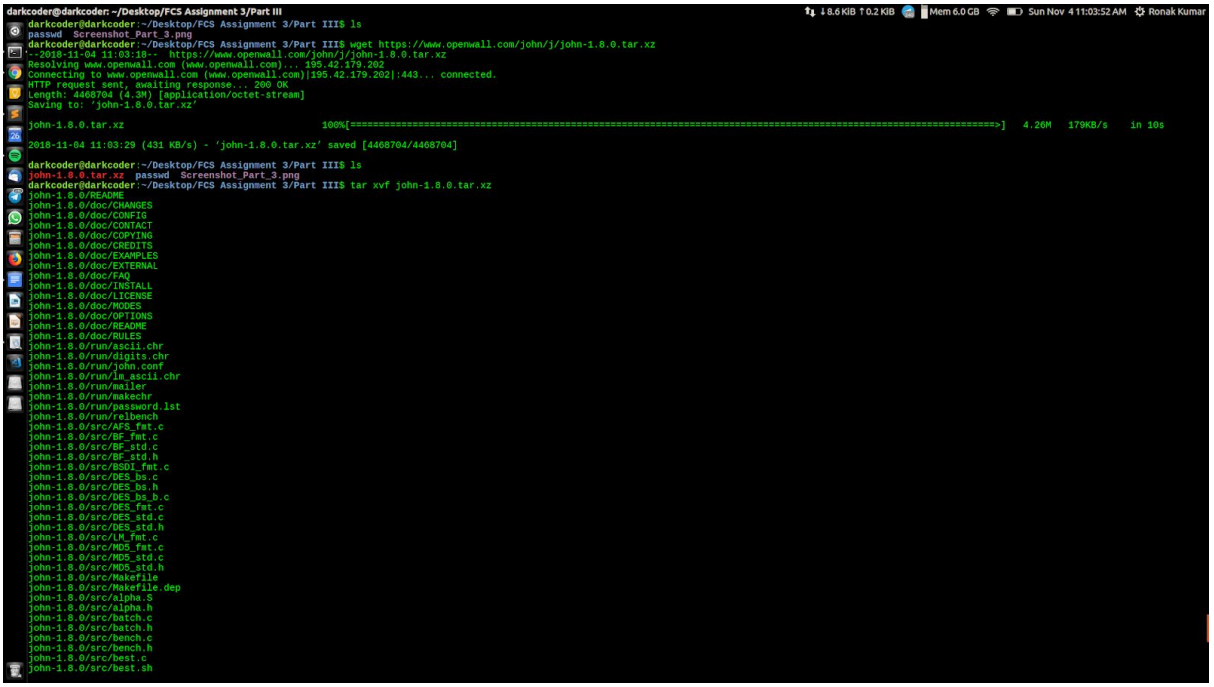
permissions. The first section is for the owner or root user, second for group and the third for normal/other users. So, it means that every user is capable of reading this file but only the root user is allowed to modify or write to the file. The password of the username is stored in the /etc/shadow file. The passwords are encrypted using hashing methods like MD5 or by using Secure Hash Algorithm.

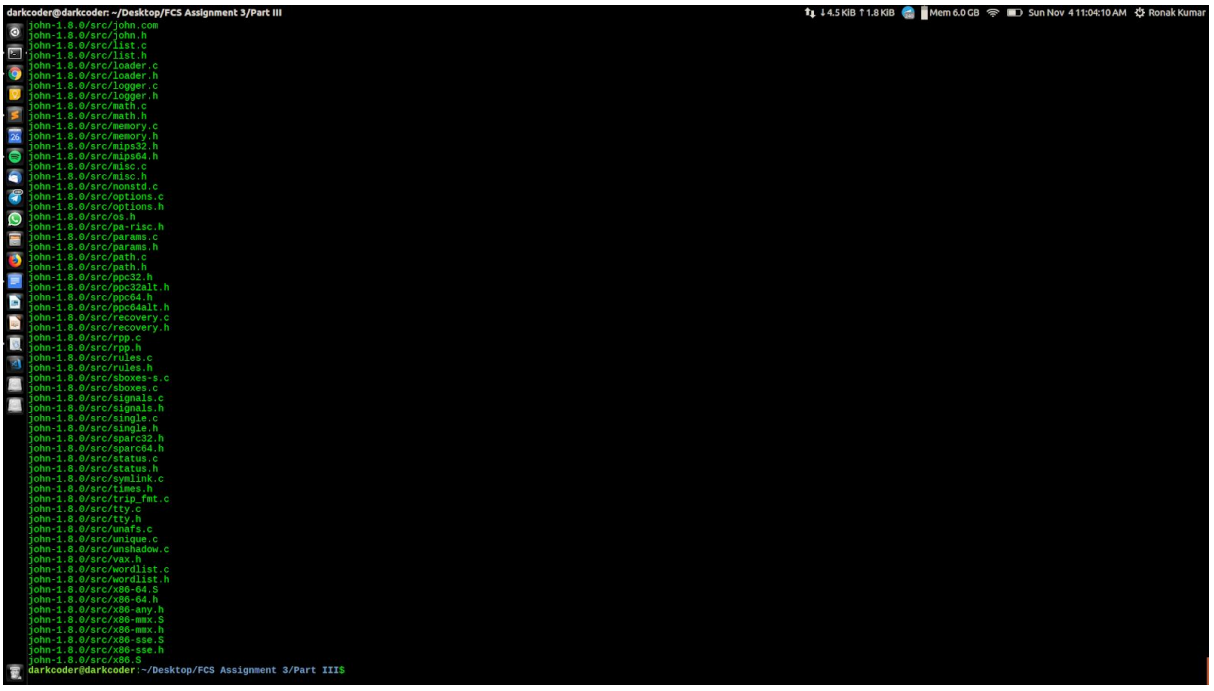
A4)

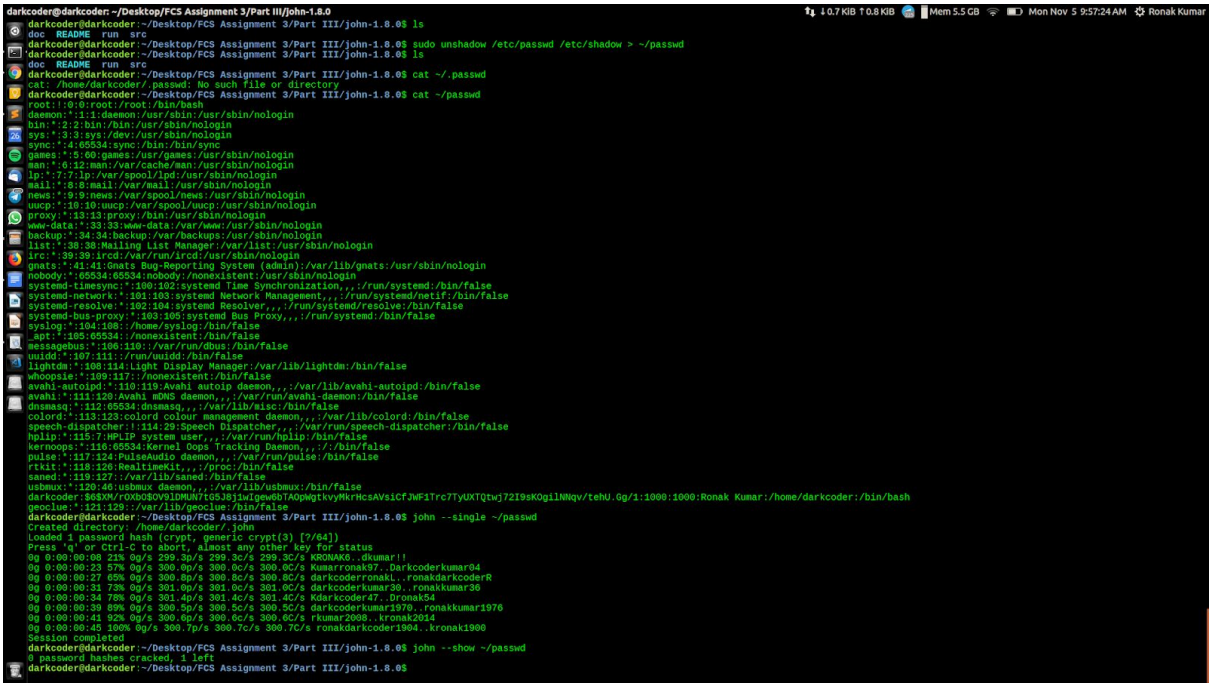
The code is given in the Part III Folder

A5)

Screenshots :-


darkcoder@darkcoder: ~/Desktop/FCS Assignment 3/Part III
↓8.6 KiB ↑0.2 KiB Mem 6.0 GB Sun Nov 4 11:03:52 AM Ronak Kumar
darkcoder@darkcoder:~/Desktop/FCS Assignment 3/Part III$ ls
passwd Screenshot_Part_3.png
darkcoder@darkcoder:~/Desktop/FCS Assignment 3/Part III$ wget https://www.openwall.com/john/j/john-1.8.0.tar.xz
--2018-11-04 11:03:18-- https://www.openwall.com/john/j/john-1.8.0.tar.xz
Resolving www.openwall.com (www.openwall.com)... 195.42.179.202
Connecting to www.openwall.com (www.openwall.com)|195.42.179.202|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4468704 (4.3M) [application/octet-stream]
Saving to: 'john-1.8.0.tar.xz'
john-1.8.0.tar.xz 100%[==========>] 4.26M 179KB/s in 10s
2018-11-04 11:03:29 (431 KB/s) - 'john-1.8.0.tar.xz' saved [4468704/4468704]
darkcoder@darkcoder:~/Desktop/FCS Assignment 3/Part III$ ls
john-1.8.0.tar.xz passwd Screenshot_Part_3.png
darkcoder@darkcoder:~/Desktop/FCS Assignment 3/Part III$ tar xvf john-1.8.0.tar.xz
john-1.8.0/README
john-1.8.0/doc/CHANGES
john-1.8.0/doc/CONFIG
john-1.8.0/doc/CONTACT
john-1.8.0/doc/COPYING
john-1.8.0/doc/CREDITS
john-1.8.0/doc/EXAMPLES
john-1.8.0/doc/EXTERNAL
john-1.8.0/doc/FAQ
john-1.8.0/doc/INSTALL
john-1.8.0/doc/LICENSE
john-1.8.0/doc/MODES
john-1.8.0/doc/OPTIONS
john-1.8.0/doc/README
john-1.8.0/doc/RULES
john-1.8.0/run/ascii.chr
john-1.8.0/run/digits.chr
john-1.8.0/run/john.conf
john-1.8.0/run/lm_ascii.chr
john-1.8.0/run/mailer
john-1.8.0/run/makechr
john-1.8.0/run/password.lst
john-1.8.0/run/relbench
john-1.8.0/src/AFS_fmt.c
john-1.8.0/src/BF_fmt.c
john-1.8.0/src/BF_std.c
john-1.8.0/src/BF_std.h
john-1.8.0/src/BSDI_fmt.c
john-1.8.0/src/DES_bs.c
john-1.8.0/src/DES_bs.h
john-1.8.0/src/DES_bs_b.c
john-1.8.0/src/DES_fmt.c
john-1.8.0/src/DES_std.c
john-1.8.0/src/DES_std.h
john-1.8.0/src/LM_fmt.c
john-1.8.0/src/MD5_fmt.c
john-1.8.0/src/MD5_std.c
john-1.8.0/src/MD5_std.h
john-1.8.0/src/Makefile
john-1.8.0/src/Makefile.dep
john-1.8.0/src/alpha.S
john-1.8.0/src/alpha.h
john-1.8.0/src/batch.c
john-1.8.0/src/batch.h
john-1.8.0/src/bench.c
john-1.8.0/src/bench.h
john-1.8.0/src/best.c
john-1.8.0/src/best.sh


darkcoder@darkcoder: ~/Desktop/FCS Assignment 3/Part III
↓4.5 KiB ↑1.8 KiB Mem 6.0 GB Sun Nov 4 11:04:10 AM Ronak Kumar
john-1.8.0/src/john.com
john-1.8.0/src/john.h
john-1.8.0/src/list.c
john-1.8.0/src/list.h
john-1.8.0/src/loader.c
john-1.8.0/src/loader.h
john-1.8.0/src/logger.c
john-1.8.0/src/logger.h
john-1.8.0/src/math.c
john-1.8.0/src/math.h
john-1.8.0/src/memory.c
john-1.8.0/src/memory.h
john-1.8.0/src/mips32.h
john-1.8.0/src/mips64.h
john-1.8.0/src/misc.c
john-1.8.0/src/misc.h
john-1.8.0/src/nonstd.c
john-1.8.0/src/options.c
john-1.8.0/src/options.h
john-1.8.0/src/os.h
john-1.8.0/src/pa-risc.h
john-1.8.0/src/params.c
john-1.8.0/src/params.h
john-1.8.0/src/path.c
john-1.8.0/src/path.h
john-1.8.0/src/ppc32.h
john-1.8.0/src/ppc32alt.h
john-1.8.0/src/ppc64.h
john-1.8.0/src/ppc64alt.h
john-1.8.0/src/recovery.c
john-1.8.0/src/recovery.h
john-1.8.0/src/rpp.c
john-1.8.0/src/rpp.h
john-1.8.0/src/rules.c
john-1.8.0/src/rules.h
john-1.8.0/src/sboxes-s.c
john-1.8.0/src/sboxes.c
john-1.8.0/src/signals.c
john-1.8.0/src/signals.h
john-1.8.0/src/single.c
john-1.8.0/src/single.h
john-1.8.0/src/sparc32.h
john-1.8.0/src/sparc64.h
john-1.8.0/src/status.c
john-1.8.0/src/status.h
john-1.8.0/src/symlink.c
john-1.8.0/src/times.h
john-1.8.0/src/trip_fmt.c
john-1.8.0/src/tty.c
john-1.8.0/src/tty.h
john-1.8.0/src/unafs.c
john-1.8.0/src/unique.c
john-1.8.0/src/unshadow.c
john-1.8.0/src/vax.h
john-1.8.0/src/wordlist.c
john-1.8.0/src/wordlist.h
john-1.8.0/src/x86-64.S
john-1.8.0/src/x86-64.h
john-1.8.0/src/x86-any.h
john-1.8.0/src/x86-mmx.S
john-1.8.0/src/x86-mmx.h
john-1.8.0/src/x86-sse.S
john-1.8.0/src/x86-sse.h
john-1.8.0/src/x86.S
darkcoder@darkcoder:~/Desktop/FCS Assignment 3/Part III$



Part IV

A1)

I disabled the echo-reply (pong) on the local lab machine. I have explained the steps in the form of commands in the order in which I executed :-

ifconfig (Got IP Address of my laptop) (Let it be IP1)
ping IP1 (Got responses)
ifconfig (Got IP Address of the Lab Computer) (Let it be IP2)
sudo iptables -L
sudo iptables -A INPUT -s IP2 -j DROP

# Now the IP on the lab machine would come in the list in my laptop
# Now if I ping my laptop from the machine i.e IP1 it would not show any results as it is now blocked
# If I again want to ping, then I would accept the IP using the command
sudo iptables -D INPUT 1

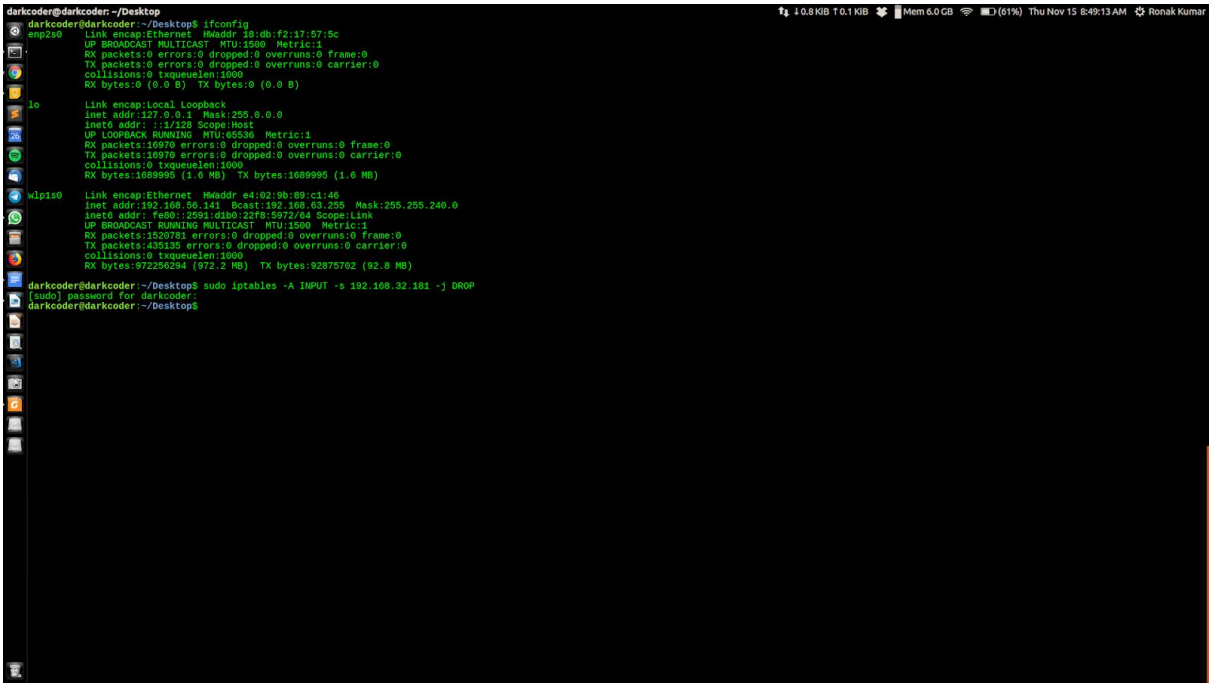
Then I will be able to ping my laptop i.e IP1 from the lab machine

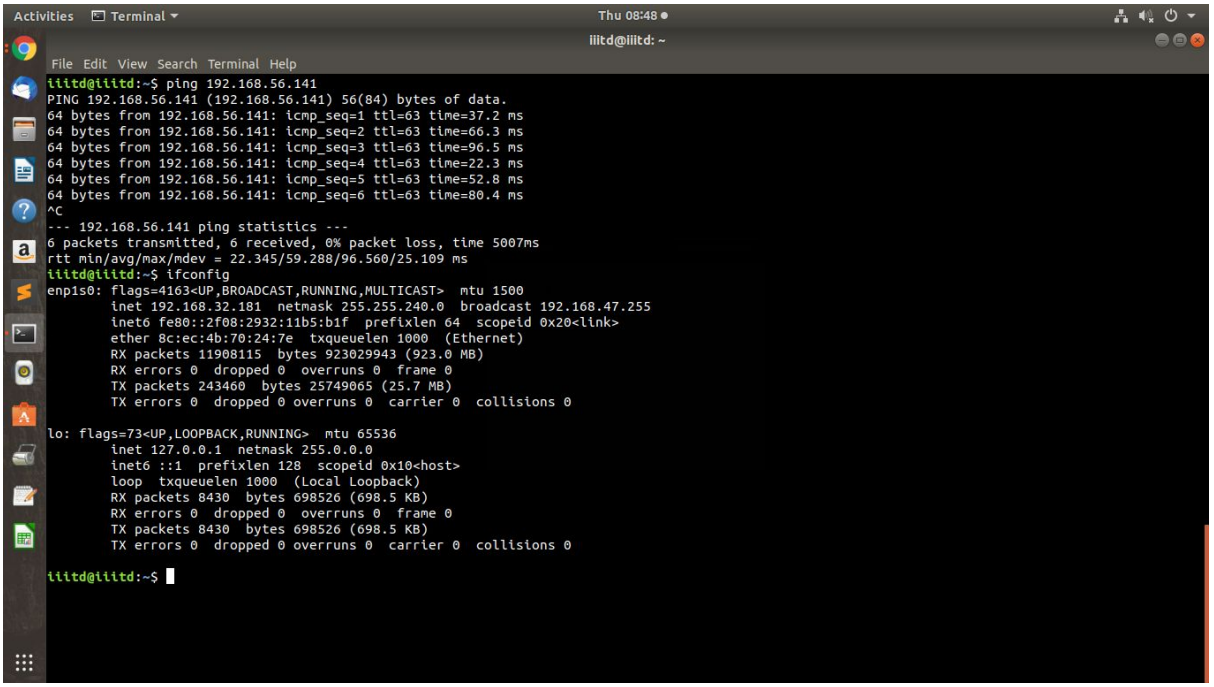
For blocking all the IPs I used the ACCEPT and the DROP command simultaneously

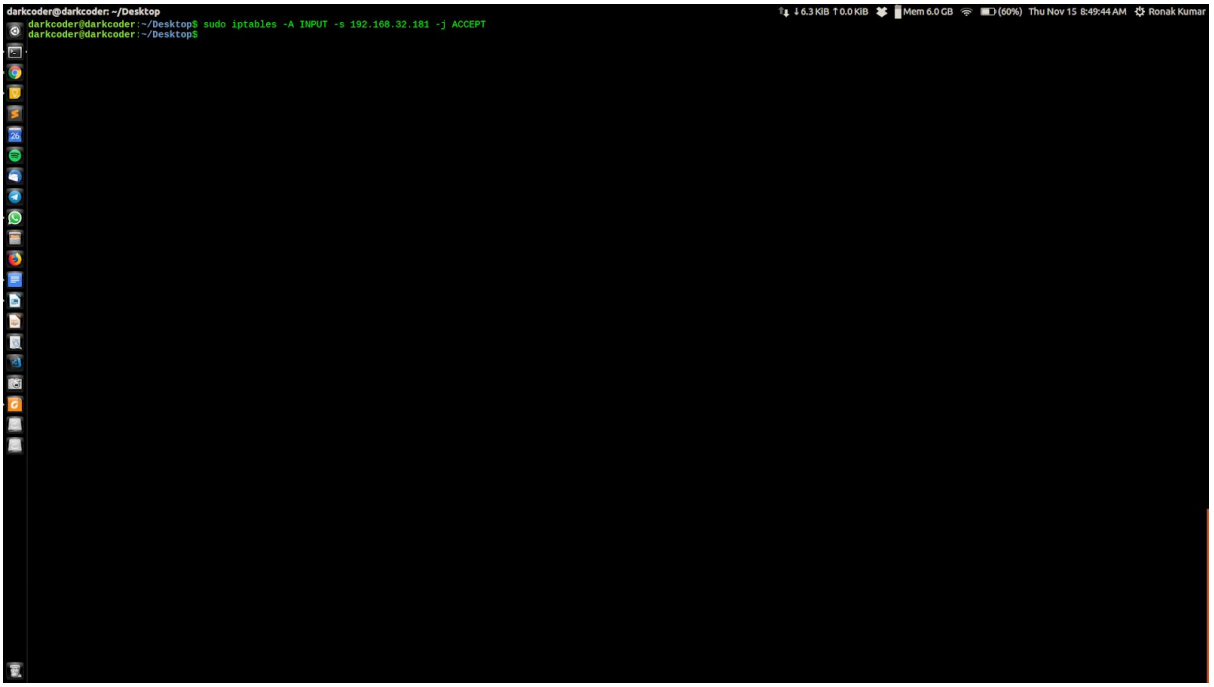
For connecting to the mobile, I used the following commands

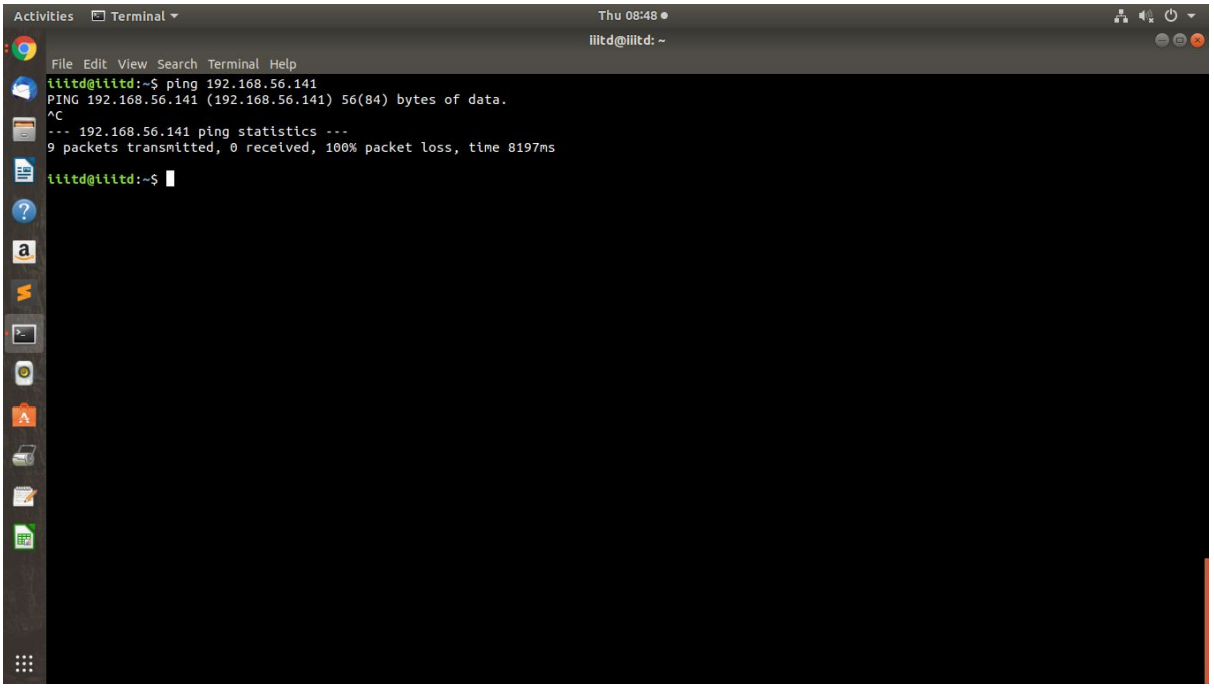
In the folder

Screenshots :-







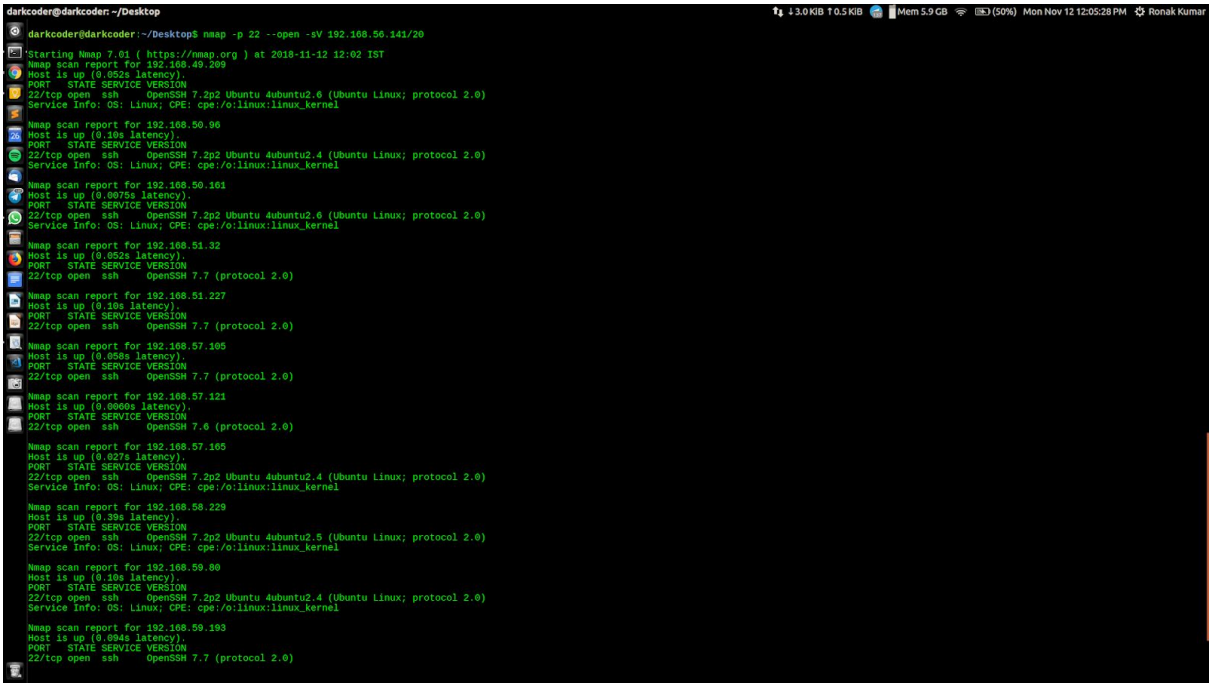


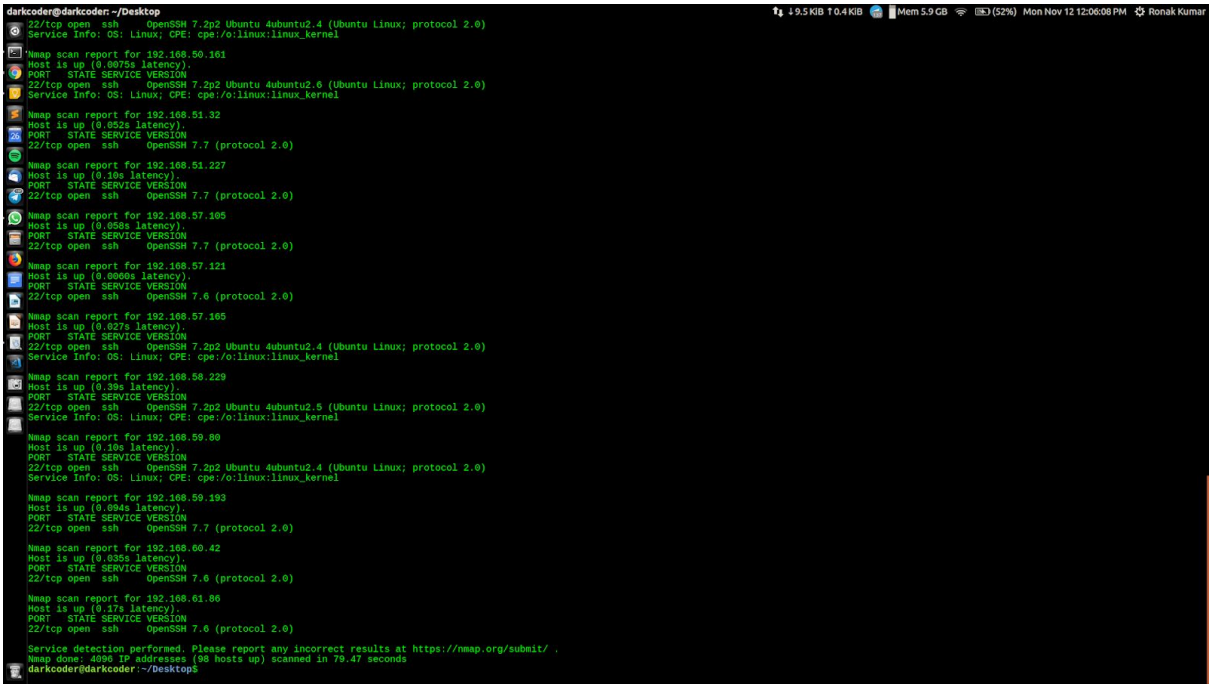
A2)

a)

These are the IP addresses of the computers connected to the B-519 subnet and provide access to SSH

Screenshots :-





b)

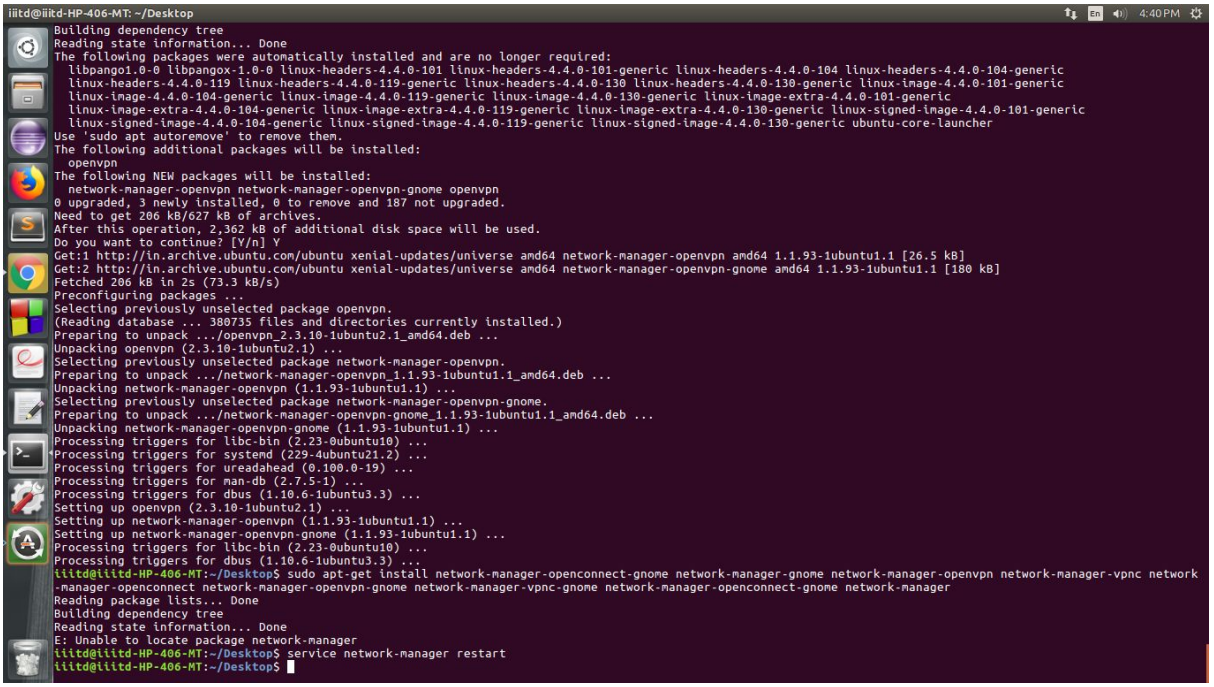
The command used for OS fingerprinting is

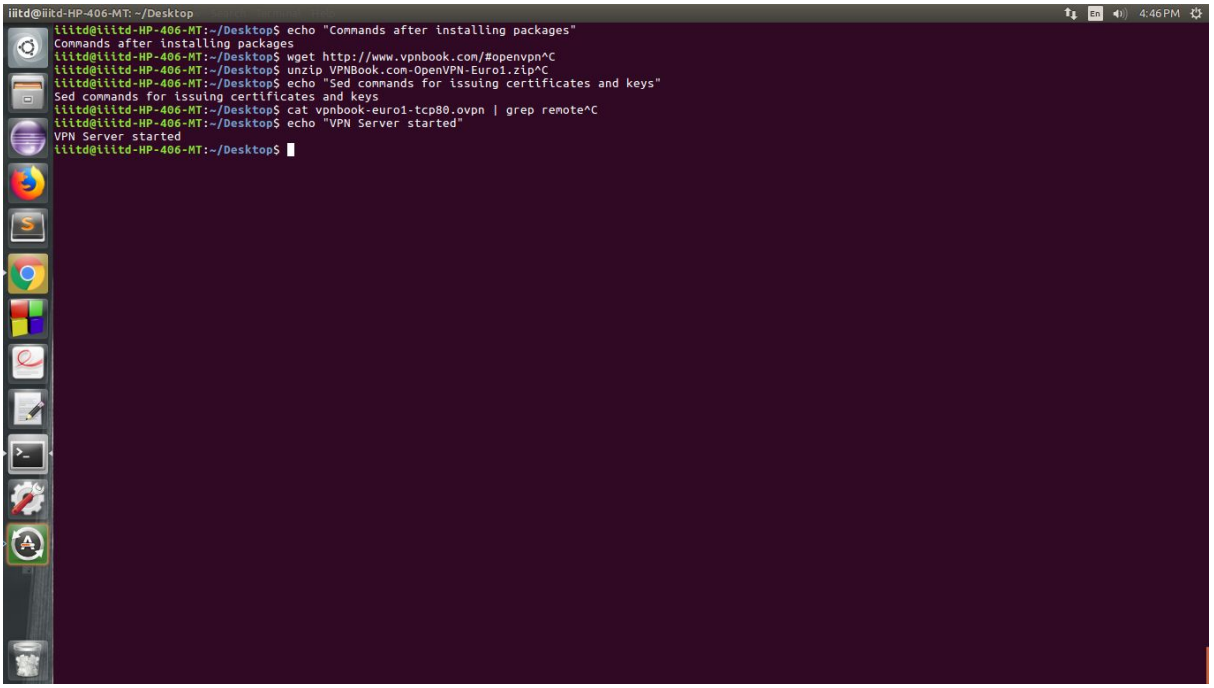
nmap -sS -o [IP Address]

A3)

I used the commands given in the screenshot for configuring the VPN in the local lab machine.

Screenshots :-





Part V

A1

a)
Some of the devices that are connected to the network with their MAC Addresses are :-

IP	MAC
98.139.239.224	8c:a9:82:50:f0:a6
74.125.225.129	00:26:08:e5:66:07

etc..

b)
The network inside the pcap file appears to be an ISP since it involves a lot of private IP addresses. ISP's usually distribute IP's in this manner. Some of the IP's got from Wireshark are 10.0.2.3, 10.0.2.2, 74.125.225.143 etc.

c)
No, in my opinion FTP in basic cannot be regarded as a safe protocol. This is because all of the information that is being transferred is being sent in the form of raw/plain text and can be sniffed. Another version of FTP known as the SFTP involving SSL encryption with FTP can be regarded as secure. Also, we can use HTTPS for encrypting all the data before sending it.

d)
The client tried to connect with "https://vimeo.com/". Yes, it is possible that HTTPS Server can protect the information leaking issue in the previous question by encrypting the entire data being send in the request before sending. Through this no one would be able to see the content of the request and the information being sent. A cipher suite is a set of cryptographic algorithms and can be considered as secure however if there is some vulnerability in the algorithm itself then it can become risky for both the cipher suite and TLS/SSL. Then, it may be prone to attacks like Downgrade attack on TLS and cipher suite.

e)
Since, the client has connected Facebook using HTTPS in his browser, but site Vimeo can redirect the user to a fake website by using malicious scripts. This may cause the problem of phishing. Here, the HTTPS does not encourage any security.

A2)

I have attached the python code in the Part V Folder. The IP address that is malicious is given in the output.txt file in the same folder. Ignore the set written that is just the data structure I used.