

FCS Assignment

-Suraj Prathik Kumar(2016101)

Question 1

All the file, encrypted, decrypted, Public and the Private key has been uploaded.

1. gpg --full-gen-key

```
surajprathikkumar — -bash — 129x55
Last login: Thu Nov 15 00:12:32 on ttys000
Suraj-P-Kumar:~ surajprathikkumar$ gpg --full-gen-key
gpg (GnuPG) 2.2.11; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Please select what kind of key you want:
  (1) RSA and RSA (default)
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (2048) 4096
Requested keysize is 4096 bits
Please specify how long the key should be valid.
  0 = key does not expire
  <n> = key expires in n days
  <n>w = key expires in n weeks
  <n>m = key expires in n months
  <n>y = key expires in n years
Key is valid for? (0) 3y
Key expires at Sun Nov 14 00:24:23 2021 IST
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

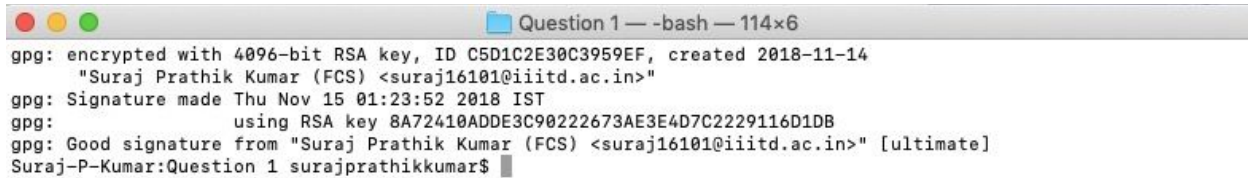
Real name: Suraj Prathik Kumar
Email address: suraj16101@iiitd.ac.in
Comment: FCS
You selected this USER-ID:
  "Suraj Prathik Kumar (FCS) <suraj16101@iiitd.ac.in>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: key E4D7C2229116D1DB marked as ultimately trusted
gpg: revocation certificate stored as '/Users/surajprathikkumar/.gnupg/openpgp-revocs.d/8A72410ADDE3C90222673AE3E4D7C2229116D1DB.
rev'
public and secret key created and signed.

pub   rsa4096 2018-11-14 [SC] [expires: 2021-11-13]
       8A72410ADDE3C90222673AE3E4D7C2229116D1DB
uid           Suraj Prathik Kumar (FCS) <suraj16101@iiitd.ac.in>
sub   rsa4096 2018-11-14 [E] [expires: 2021-11-13]

Suraj-P-Kumar:~ surajprathikkumar$
```

2. `gpg --encrypt --sign --armor -r suraj16101@iiitd.ac.in File`

A terminal window titled "Question 1 — -bash — 114x6" showing the output of a gpg command. The output indicates that a file was encrypted with a 4096-bit RSA key and signed with the user's private key. The terminal text is as follows:

```
gpg: encrypted with 4096-bit RSA key, ID C5D1C2E30C3959EF, created 2018-11-14
      "Suraj Prathik Kumar (FCS) <suraj16101@iiitd.ac.in>"
gpg: Signature made Thu Nov 15 01:23:52 2018 IST
gpg:      using RSA key 8A72410ADDE3C90222673AE3E4D7C2229116D1DB
gpg: Good signature from "Suraj Prathik Kumar (FCS) <suraj16101@iiitd.ac.in>" [ultimate]
Suraj-P-Kumar:Question 1 surajprathikkumar$
```

3. Yes, I can decrypt the file with “`gpg -d File.asc`”

I can decrypt it as I do have the private key that was associated to the public key that was used to generate the File.asc.

If someone else has the File they cannot decrypt it as they won't have the private key.

Question 2

1. Commands -

- `md5 <filename>`
- `shasum -a 1 <filename>`
- `shasum -a 224 <filename>`
- `shasum -a 256 <filename>`
- `shasum -a 384 <filename>`
- `shasum -a 512 <filename>`
- `sha3sum -a <filename>`

Speed

`md5 > sha1 > sha224 > sha256 > sha384 > sha512 > sha3`

```

[Suraj-P-Kumar:Desktop surajprathikkumar$ time md5 large_file.txt
MD5 (large_file.txt) = 4a2a9d7d13218651cf013769a7c660c4

real    0m0.110s
user    0m0.086s
sys     0m0.027s
[Suraj-P-Kumar:Desktop surajprathikkumar$ time shasum -a 1 large_file.txt
3148a2bd9a781afa2264f751226179a96a7036e2  large_file.txt

real    0m0.222s
user    0m0.122s
sys     0m0.045s
[Suraj-P-Kumar:Desktop surajprathikkumar$ time shasum -a 224 large_file.txt
3bd17115f34bda352db8ddf2d4685946a6ffabaff0db1a4466ed3f1e  large_file.txt

real    0m0.233s
user    0m0.210s
sys     0m0.019s
[Suraj-P-Kumar:Desktop surajprathikkumar$ time shasum -a 256 large_file.txt
080e26f5b88f2f030a26a2b55654218bf4a960bb643f14772f5139b23620159c  large_file.txt

real    0m0.237s
user    0m0.213s
sys     0m0.021s
[Suraj-P-Kumar:Desktop surajprathikkumar$ time shasum -a 384 large_file.txt
a50f7071b71a3b395517552558486ef9e8e081e16a059bb6c90d54f6308126acccdd6a4521be5828f8aa78b24d16904  large_file.tx
t

real    0m0.181s
user    0m0.158s
sys     0m0.019s
[Suraj-P-Kumar:Desktop surajprathikkumar$ time shasum -a 512 large_file.txt
68b3f57471339766e83890252a7e5880706ad9f8ba317cbb4946936249b3e74036634ff9bf167528373927f363d9e023ec9a183a0beebf5
6f5aeafb398d5d403  large_file.txt

real    0m0.196s
user    0m0.164s
sys     0m0.024s
[Suraj-P-Kumar:Desktop surajprathikkumar$ ]

```

2. a) File Names 14.lws.txt and 100west.txt have were modified by the third party.2 out of 5 were modified.

Methodology - Checksum used to verify the files integrity as it is very unlikely that any two non-identical files in the real world will have the same MD5 hash and Sha1 hash unless they have been specifically created to have the same hash.

```

[Suraj-P-Kumar:Desktop surajprathikkumar$ shasum -a 1 13chil.txt
0ea223d57002be138c314937555e9c34a21107a1 13chil.txt
[Suraj-P-Kumar:Desktop surajprathikkumar$ shasum -a 1 14.lws.txt
f54b7cc361b9d0e2d4d13e107e9bc1c923a0c263 14.lws.txt
[Suraj-P-Kumar:Desktop surajprathikkumar$ shasum -a 1 16.lws.txt
719661b44387e1c21ea4f2ee57cc4377ae5c5a5 16.lws.txt
[Suraj-P-Kumar:Desktop surajprathikkumar$ shasum -a 1 17.lws.txt
6ee74b0ee77bd6ab150ca604543b0a91d7c7034d 17.lws.txt
[Suraj-P-Kumar:Desktop surajprathikkumar$ shasum -a 1 100west.txt
39d44af1220fa42f82257c554de6722facbcf4db 100west.txt
[Suraj-P-Kumar:Desktop surajprathikkumar$ ]

[Suraj-P-Kumar:stories surajprathikkumar$ shasum -a 1 13chil.txt
0ea223d57002be138c314937555e9c34a21107a1 13chil.txt
[Suraj-P-Kumar:stories surajprathikkumar$ shasum -a 1 14.lws
f54b7cc361b9d0e2d4d13e107e9bc1c923a0c263 14.lws
[Suraj-P-Kumar:stories surajprathikkumar$ shasum -a 1 16.lws
203d185504fdafce1b5ac2afd32374ecd2e559b 16.lws
[Suraj-P-Kumar:stories surajprathikkumar$ shasum -a 1 17.lws
6ee74b0ee77bd6ab150ca604543b0a91d7c7034d 17.lws
[Suraj-P-Kumar:stories surajprathikkumar$ shasum -a 1 100west.txt
586ea64cf65671465eb797a675fb917a8c706cff 100west.txt
[Suraj-P-Kumar:stories surajprathikkumar$ ]

```

b) The detection technique might also fail in case of MD5 collisions, it is possible for the third party to create a second file with the same checksum, so this technique cannot protect against some forms of malicious tampering.

c) Cryptographic Hash functions are mostly used instead of hash functions to solve 3 fundamental problems associated with hash functions.

Pre-image resistance, Second Pre-image resistance, Collision resistance (Hard to find messages $\text{hash}(m1) = \text{hash}(m2)$)

Security property violated is **Collision resistance**.

Since MD5 collisions are possible in 2b and they take away the use case of cryptographic hash functions over hash functions.

Due to this Priyab will have to use some other technique to find the integrity of the files.

Question 3

1. The changes in the functions are -

```
static int is_registered(char *uname) {  
    FILE *db_file;  
  
    if ((db_file = fopen("user_db.txt", "r")) == NULL) {  
        printf("Error\n");  
        exit(1);  
    }  
  
    char user[2000], passwd[2000];  
    while (fscanf(db_file, "%s", user) == 1) {  
        fscanf(db_file, " ");  
        fscanf(db_file, "%s\n", passwd);  
        if (strcmp(uname, user) == 0) {  
            return 1;  
        }  
    }  
  
    fclose(db_file);  
  
    return 0;  
}
```

```

int register_user(char *uname, char *passwd) {
    if (access("user_db.txt", R_OK) != 0) {
        if (errno == ENOENT) {
            printf("File does not exist\n");
            return -1;
        }

        if (errno == EACCES) {
            printf("User does not have permissions to access database\n");
            return -1;
        }

        fprintf(stderr, "Error Occured\n");
        return -1;
    }

    if (is_registered(uname)) {
        fprintf(stderr, "Choose another username\n");
        return 0;
    }

    FILE *db_file = fopen("user_db.txt", "a");

    if (db_file == NULL) {
        printf("Error\n");
        exit(1);
    }

    fprintf(db_file, "%s", uname);
    fprintf(db_file, " ");
    fprintf(db_file, "%s\n", passwd);

    fclose(db_file);

    return 1;
}

```

```

int auth_user(char *uname, char *passwd) {
    if (!is_registered(uname)) {
        fprintf(stderr, "User not registered\n");
        return 0;
    }

    FILE *db_file;

    if ((db_file = fopen("user_db.txt", "r")) == NULL)
    {
        printf("Error\n");
        exit(1);
    }

    char user[2000], password[2000];
    while (fscanf(db_file, "%s", user) == 1)
    {
        fscanf(db_file, " ");
        fscanf(db_file, "%s\n", password);

        if (strcmp(uname, user) == 0)
        {
            if (strcmp(strdup(passwd), strdup(password)) == 0) {
                return 1;
            }
        }
    }

    fclose(db_file);

    return 0;
}

```

```

int main(int argc, char *argv[])
{
    int register_flag = 0;

    if (strcmp(argv[1], "-r") == 0) {
        printf("Register User\n");
        register_flag = 1;
    }

    else if (strcmp(argv[1], "-a") == 0)
    {
        printf("Authorise\n");
        register_flag = 0;
    }

    char uname[2000];

    printf("Enter Username: ");
    scanf("%s", uname);

    unsigned long seed[2];
    char salt[] = "$1$.....";
    char temp[] = "abcdef";
    const char *const seedchars =
        "./0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ"
        "UVWXYZabcdefghijklmnopqrstuvwxyz";
    seed[0] = 0;
    seed[1] = 0;
    char *password;
    int i;

    for (i = 0; i < 8; i++)
        salt[3 + i] = seedchars[(seed[i / 5] >> (i % 5) * 6) & 0x3f];

    password = crypt(getpass("Password:"), salt);

    if (register_flag == 1) {
        int a = register_user(uname, password);
    }
    else {
        int a = auth_user(uname, password);

        if (a == 1) {
            printf("Authorised\n");
        }
        else {
            printf("Incorrect\n");
        }
    }

    return 0;
}

```

The code hashes the password using a salt. The file permissions have also been defined as above.

2. The following code is used for the Brute force attack. The password list is saved in a file -

commands to run -

gcc -o ./attack brute_force.c

./attack <user>

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
    char username[200];

    if (argc!=2)
    {
        exit(1);
    }
    else
    {
        strcpy(username, argv[1]);

        FILE *fptr1;

        fptr1 = fopen("./passwd.txt","r");

        if (fptr1 == NULL)
        {
            printf("Error");
            exit(1);
        }
        else
        {
            char password[1000];

            while (fgets(password, 1000, fptr1) != NULL)
            {
                FILE *fp;

                fp = popen("./main","w");
                fprintf(fp, "%s\n", username);
                fprintf(fp, "%s\n", password);

                pclose(fp);
            }
        }
    }

    return 0;
}
```

3. The password and the username is saved in the /etc/shadow file.


```
root@kali:/etc# gedit shadow

shadow
/etc

root:
$6$g0hGrss7$gA1e.UKigndgogv.r59EnKp6PAVIA5/63Hzn1VPwupNVtA52MlADfngCQg9I1M0t14jyIhUfiJp0LkvPkgv3I1:1
daemon*:17820:0:99999:7:::
bin*:17820:0:99999:7:::
sys*:17820:0:99999:7:::
sync*:17820:0:99999:7:::
games*:17820:0:99999:7:::
man*:17820:0:99999:7:::
lp*:17820:0:99999:7:::
mail*:17820:0:99999:7:::
news*:17820:0:99999:7:::
uucp*:17820:0:99999:7:::
proxy*:17820:0:99999:7:::
www-data*:17820:0:99999:7:::
backup*:17820:0:99999:7:::
list*:17820:0:99999:7:::
irc*:17820:0:99999:7:::
gnats*:17820:0:99999:7:::
nobody*:17820:0:99999:7:::
```

The passwd stores general user information and shadow stores user passwd info. passwd is the file where the user information and in shadow file important information (like an encrypted form of the password of a user, the day the password expires) are stored.

5. Command - **sudo /usr/sbin/unshadow /etc/passwd /etc/shadow > /tmp/crack.password.db**

For cracking Password: **john /tmp/crack.password.db**

```
root@kali: ~/Desktop

File Edit View Search Terminal Help
fopen: /etc/passwd/: Not a directory
root@kali:~/Desktop# sudo /usr/sbin/unshadow /etc/passwd /etc/shadow > /tmp/crack.password.db
root@kali:~/Desktop# john /tmp/crack.password.db
Created directory: /root/.john
Warning: detected hash type "sha512crypt", but the string is also recognized as "crypt"
Use the "--format=crypt" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Press 'q' or Ctrl-C to abort, almost any other key for status
toor (root)
lg 0:00:00:00 DONE 1/3 (2018-11-15 11:35) 33.33g/s 266.6p/s 266.6c/s 266.6C/s root..roots
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

To show the cracked File: **john -show /tmp/crack.password.db**

```
root@kali:~/Desktop# john -show /tmp/crack.password.db
root:toor:0:0:root:/root:/bin/bash

1 password hash cracked, 0 left
```

Question 4

Note: Since i have a mac the 4th Question is done on LAB PC as the iptables dont work on it.

1. a) Command -

Sudo iptables -A INPUT -p icmp --icmp-type echo-request -j DROP

Sudo iptables -A OUTPUT -p icmp --icmp-type echo-reply -j DROP

```
iiiitd@NameNode: ~
iiiitd@NameNode:~$ sudo iptables -A INPUT -p icmp --icmp-type echo-request -j DROP
iiiitd@NameNode:~$ sudo iptables -A OUTPUT -p icmp --icmp-type echo-reply -j DROP
iiiitd@NameNode:~$ time ping 192.168.33.47
PING 192.168.33.47 (192.168.33.47) 56(84) bytes of data.
^C
--- 192.168.33.47 ping statistics ---
12 packets transmitted, 0 received, 100% packet loss, time 11086ms

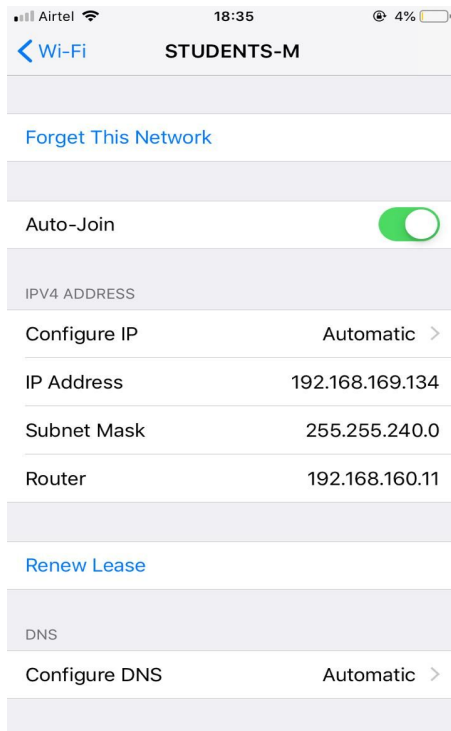
real    0m11.312s
user    0m0.000s
sys      0m0.000s
iiiitd@NameNode:~$ ping fb.com
PING fb.com (157.240.24.35) 56(84) bytes of data.
64 bytes from edge-star-mini-shv-01-sin2.facebook.com (157.240.24.35): icmp_seq=1 ttl=51 time=78.3 ms
64 bytes from edge-star-mini-shv-01-sin2.facebook.com (157.240.24.35): icmp_seq=2 ttl=51 time=78.2 ms
64 bytes from edge-star-mini-shv-01-sin2.facebook.com (157.240.24.35): icmp_seq=3 ttl=51 time=78.3 ms
64 bytes from edge-star-mini-shv-01-sin2.facebook.com (157.240.24.35): icmp_seq=4 ttl=51 time=78.6 ms
64 bytes from edge-star-mini-shv-01-sin2.facebook.com (157.240.24.35): icmp_seq=5 ttl=51 time=78.1 ms
64 bytes from edge-star-mini-shv-01-sin2.facebook.com (157.240.24.35): icmp_seq=6 ttl=51 time=78.3 ms
64 bytes from edge-star-mini-shv-01-sin2.facebook.com (157.240.24.35): icmp_seq=7 ttl=51 time=78.2 ms
64 bytes from edge-star-mini-shv-01-sin2.facebook.com (157.240.24.35): icmp_seq=8 ttl=51 time=78.6 ms
64 bytes from edge-star-mini-shv-01-sin2.facebook.com (157.240.24.35): icmp_seq=9 ttl=51 time=78.4 ms
^C
--- fb.com ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8010ms
rtt min/avg/max/mdev = 78.184/78.401/78.692/0.384 ms
iiiitd@NameNode:~$
```

Checking if the ping to system is blocked. “**ping IP address**”

No response and checked for 11sec.

Checking if the ping to any other devices is working by hitting “**ping fb.com**” and it is working.

b) IP address of my Phone



Command for dropping all the packets

sudo iptables -P INPUT DROP && sudo iptables -P OUTPUT DROP

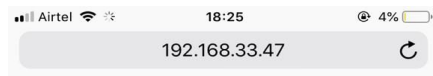
```
liitd@NameNode: /var/www/html
liitd@NameNode:/var/www/html$ sudo gedit index.html
** (gedit:6927): WARNING **: Set document metadata failed: Setting attribute metadata::gedit-position not supported
liitd@NameNode:/var/www/html$ sudo iptables -P INPUT DROP
liitd@NameNode:/var/www/html$ sudo iptables -P OUTPUT DROP
liitd@NameNode:/var/www/html$ sudo iptables -A INPUT -s 192.168.169.134 -j ACCEPT
liitd@NameNode:/var/www/html$ sudo iptables -A OUTPUT -d 192.168.169.134 -j ACCEPT
liitd@NameNode:/var/www/html$
```

Hosted the HTML webpage on apache2 on the Lab pc with IP address - 192.168.33.47

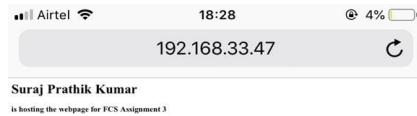
After Dropping all the packets i could not get the webpage on my phone or laptop
After that i allowed my IP address (phone) .

Command - **sudo iptables -A INPUT -s 192.168.169.134 -j ACCEPT**
sudo iptables -A OUTPUT -d 192.168.168.134 -j ACCEPT

Before Allowing



After Allowing



2. a) **File.txt** has been uploaded along with the document.

IP - 192.168.33.47

Subnet mask for lab B519- 255.255.240.0/20

subnet ID - 192.168.33.47/20.

ssh run on PORT - 22

Command - **nmap -sS 192.168.33.47/20 -p 22 > file.txt**

b) **Complete File** has been uploaded with pdf

Command - **nmap -O 192.168.43.110/20 > File_OSFingerprint.txt**

```
surajprathikkumar — -bash — 80x24
Last login: Thu Nov 15 21:26:44 on ttys000
[Suraj-P-Kumar:~ surajprathikkumar$ nmap -O 192.168.43.110/20
TCP/IP fingerprinting (for OS scan) requires root privileges.
QUITTING!
[Suraj-P-Kumar:~ surajprathikkumar$ sudo !!
sudo nmap -O 192.168.43.110/20
[Password:
Starting Nmap 7.70 ( https://nmap.org ) at 2018-11-15 21:58 IST
Nmap scan report for 192.168.32.2
Host is up (0.0049s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
3333/tcp   open  dec-notes
Aggressive OS guesses: Linux 3.12 (95%), Linux 3.13 (95%), Linux 3.16 (95%), Lin
ux 3.2 - 4.9 (95%), Linux 4.8 (95%), Linux 4.4 (95%), Linux 4.9 (95%), Linux 3.1
8 (95%), Linux 3.8 - 3.11 (95%), Linux 4.2 (95%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 2 hops

Nmap scan report for 192.168.32.3
Host is up (0.0049s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
```

Stats - 168 out of which only 6 were found to be windows and rest linux users
thereby proving these stats wrong of 70% windows and 20% Linux

3. Install openvpn

Commands - `sudo apt-get install openvpn easy-rsa`

Setup ca Directory

```
make-cadir ~/openvpn-ca
nano vars
```

Then you create your certificate

Build the Certificate Authority

```
cd ~/openvpn-ca
source vars
```

Create the server Key

```
./build-key-server server
```

Generate a Client Certificate and Key Pa

`sudo nano /etc/sysctl.conf`

```
root@NameNode: /var/www/html
Location: https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh [following]
--2018-11-15 18:47:17-- https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.156.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.156.133|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh [following]
--2018-11-15 18:47:17-- https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.156.133
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.156.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 14739 (14K) [text/plain]
Saving to: 'openvpn-install.sh'

openvpn-install.sh 100%[=====] 14.39K --.-KB/s in 0.05s

2018-11-15 18:47:18 (312 KB/s) - 'openvpn-install.sh' saved [14739/14739]

Looks like OpenVPN is already installed.

What do you want to do?
 1) Add a new user
 2) Revoke an existing user
 3) Remove OpenVPN
 4) Exit
Select an option [1-4]: 1

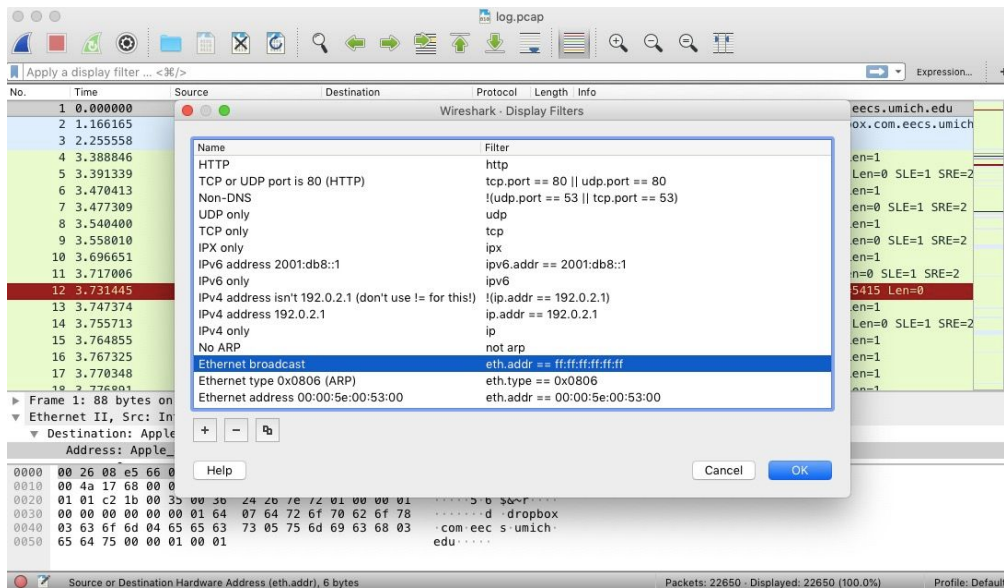
Tell me a name for the client certificate.
Please, use one word only, no special characters.
Client name: Suraj
Generating a 2048 bit RSA private key
.....+++
..+++
writing new private key to '/etc/openvpn/easy-rsa/pki/private/Suraj.key.VNXoiwsSLz'
-----
rand: Use -help for summary.
Using configuration from ./openssl-1.0.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName      :ASN.1 12:'Suraj'
Certificate is to be certified until Nov 12 13:17:35 2028 GMT (3650 days)

Write out database with 1 new entries
Data Base Updated

Client Suraj added, configuration is available at: /root/Suraj.ovpn
root@NameNode:/var/www/html# ifconfig
docker0  Link encap:Ethernet  HWaddr 02:42:69:4e:6c:aa
          inet addr:172.17.0.1  Bcast:172.17.255.255  Mask:255.255.0.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
```


Question 5

1. a) The ethernet broadcast is `eth.addr == ff:ff:ff:ff:ff:ff`



Associated IP and Mac Addresses are :

The top screenshot shows a Wireshark capture of network traffic. The packet list includes:

No.	Time	Source	Destination	Protocol	Length	Info
73	7.540196	10.0.2.2	255.255.255.255	DB-LS...	247	Dropbox LAN sync Discovery Protocol
74	7.540704	10.0.2.2	10.0.2.255	DB-LS...	247	Dropbox LAN sync Discovery Protocol
88	10.083645	10.0.2.1	10.0.2.255	DB-LS...	247	Dropbox LAN sync Discovery Protocol
89	10.083890	Apple_e5:66:07	Broadcast	ARP	42	Who has 169.254.255.255? Tell 10.0.2.1
126	15.948971	IntelCor_50:f0:a6	Broadcast	ARP	42	Who has 10.0.2.1? Tell 10.0.2.3
160	19.213874	10.0.2.3	10.0.2.255	DB-LS...	202	Dropbox LAN sync Discovery Protocol
4314	37.559905	10.0.2.2	255.255.255.255	DB-LS...	247	Dropbox LAN sync Discovery Protocol
4315	37.560201	10.0.2.2	10.0.2.255	DB-LS...	247	Dropbox LAN sync Discovery Protocol
4506	40.131870	10.0.2.1	10.0.2.255	DB-LS...	247	Dropbox LAN sync Discovery Protocol
4527	41.013503	10.0.2.3	10.0.2.255	NBNS	92	Name query NB WPAD<00>
4543	41.765177	10.0.2.3	10.0.2.255	NBNS	92	Name query NB WPAD<00>
4544	42.517178	10.0.2.3	10.0.2.255	NBNS	92	Name query NB WPAD<00>
4590	43.519020	10.0.2.1	10.0.2.255	NBNS	92	Name query NB <01><02>__MSBROWSE__<02><01>
4591	43.547107	10.0.2.1	10.0.2.255	NBNS	92	Name query NB MYGROUP<1d>
4592	43.547490	10.0.2.1	10.0.2.255	BROWS...	216	Get Backup List Request
4593	43.547732	10.0.2.1	10.0.2.255	BROWS...	216	Get Backup List Request

The bottom screenshot shows the same Wireshark capture with a summary table at the bottom:

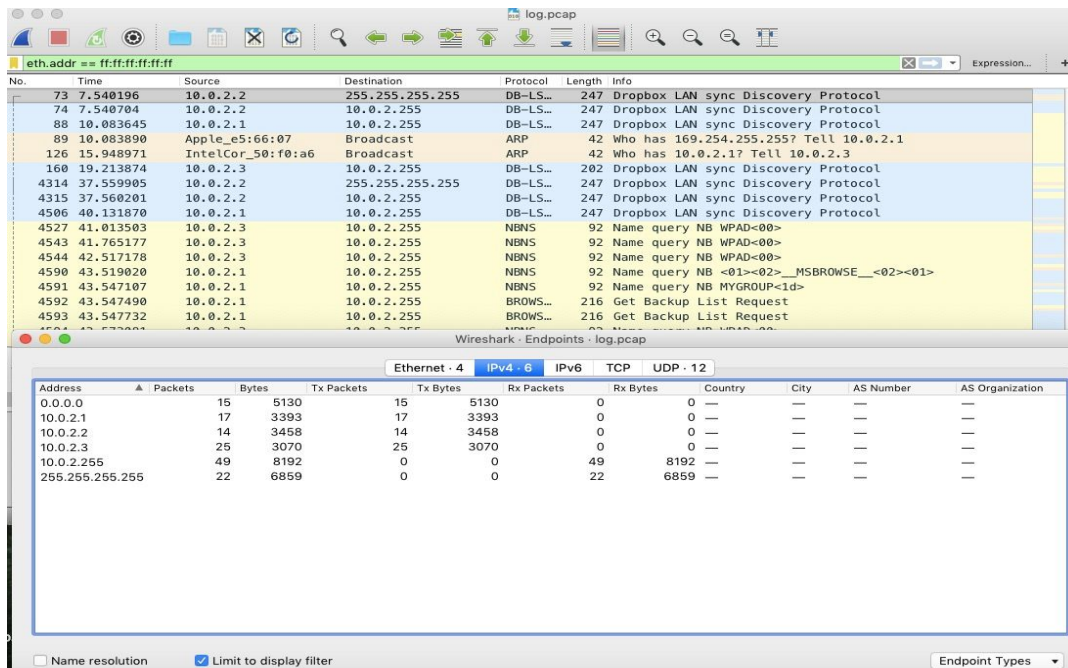
Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
0.0.0.0	15	5130	15	5130	0	0
10.0.2.1	17	3393	17	3393	0	0
10.0.2.2	14	3458	14	3458	0	0
10.0.2.3	25	3070	25	3070	0	0
10.0.2.255	49	8192	0	0	49	8192
255.255.255.255	22	6859	0	0	22	6859

The bottom screenshot also shows a summary table for the MAC addresses:

Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes
00:26:08:e5:66:07	40	8859	40	8859	0	0
04:0c:ce:d8:0f:fa	14	3458	14	3458	0	0
8c:a9:82:50:f0:a6	28	3196	28	3196	0	0
ff:ff:ff:ff:ff:ff	82	15 k	0	0	82	15 k

b) Since the number of host is 3 as seen in the picture. This seems like a home network or of a small institute. The websites visited frequently include social

networking sites like facebook and some educational sites.

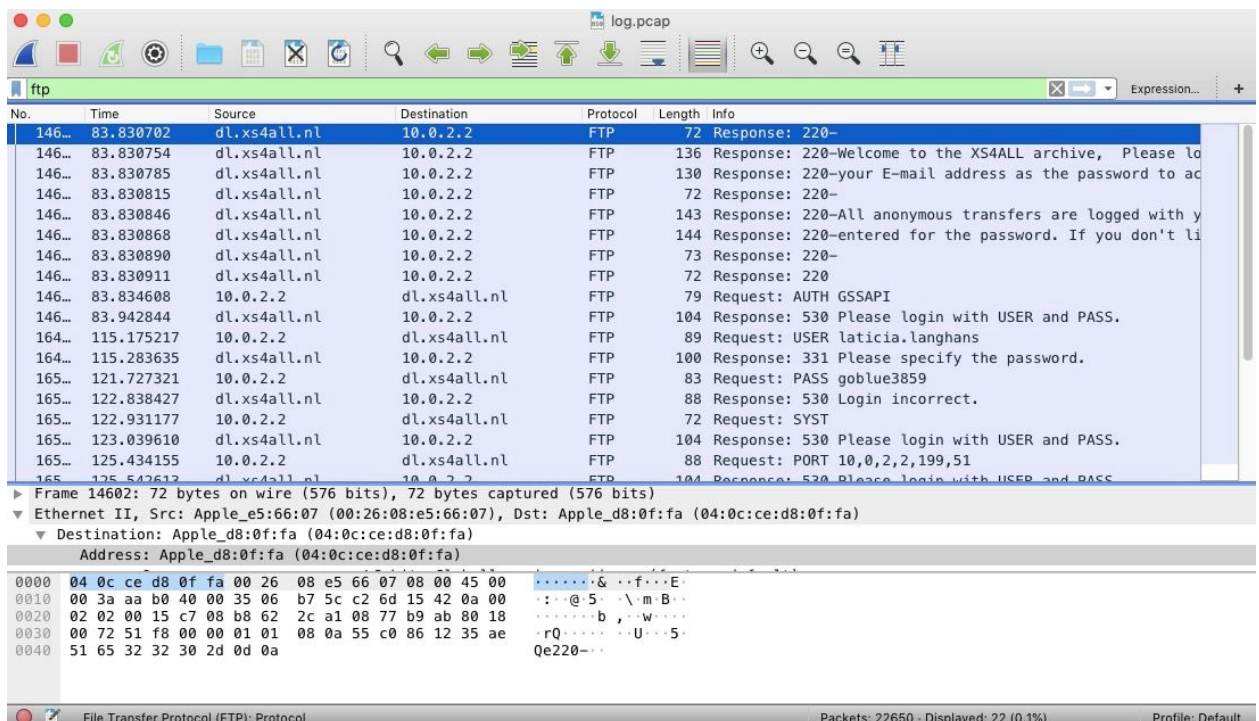


Wireshark packet capture showing network traffic. The top pane displays a list of packets with columns: No., Time, Source, Destination, Protocol, Length, Info. The bottom pane shows a summary table for Ethernet II, IPv4, TCP, and UDP.

Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes	Country	City	AS Number	AS Organization
0.0.0.0	15	5130	15	5130	0	0	—	—	—	—
10.0.2.1	17	3393	17	3393	0	0	—	—	—	—
10.0.2.2	14	3458	14	3458	0	0	—	—	—	—
10.0.2.3	25	3070	25	3070	0	0	—	—	—	—
10.0.2.255	49	8192	0	0	49	8192	—	—	—	—
255.255.255.255	22	6859	0	0	22	6859	—	—	—	—

c) The DNS host name is xs4all and 192.109.21.66

All the ftp requests are sent to this ip by the other hosts and they get a response like Login incorrect etc.



Wireshark packet capture showing FTP traffic. The top pane displays a list of packets with columns: No., Time, Source, Destination, Protocol, Length, Info. The bottom pane shows a detailed view of a packet, including the Ethernet II header and the FTP payload.

No.	Time	Source	Destination	Protocol	Length	Info
146...	83.830702	dl.xs4all.nl	10.0.2.2	FTP	72	Response: 220-
146...	83.830754	dl.xs4all.nl	10.0.2.2	FTP	136	Response: 220-Welcome to the XS4ALL archive, Please lo
146...	83.830785	dl.xs4all.nl	10.0.2.2	FTP	130	Response: 220-your E-mail address as the password to ac
146...	83.830815	dl.xs4all.nl	10.0.2.2	FTP	72	Response: 220-
146...	83.830846	dl.xs4all.nl	10.0.2.2	FTP	143	Response: 220-All anonymous transfers are logged with y
146...	83.830868	dl.xs4all.nl	10.0.2.2	FTP	144	Response: 220-entered for the password. If you don't li
146...	83.830890	dl.xs4all.nl	10.0.2.2	FTP	73	Response: 220-
146...	83.830911	dl.xs4all.nl	10.0.2.2	FTP	72	Response: 220
146...	83.834608	dl.xs4all.nl	10.0.2.2	FTP	79	Request: AUTH GSSAPI
146...	83.942844	dl.xs4all.nl	10.0.2.2	FTP	104	Response: 530 Please login with USER and PASS.
164...	115.175217	dl.xs4all.nl	10.0.2.2	FTP	89	Request: USER laticia.langhans
164...	115.283635	dl.xs4all.nl	10.0.2.2	FTP	100	Response: 331 Please specify the password.
165...	121.727321	dl.xs4all.nl	10.0.2.2	FTP	83	Request: PASS goblue3859
165...	122.838427	dl.xs4all.nl	10.0.2.2	FTP	88	Response: 530 Login incorrect.
165...	122.931177	dl.xs4all.nl	10.0.2.2	FTP	72	Request: SYST
165...	123.039610	dl.xs4all.nl	10.0.2.2	FTP	104	Response: 530 Please login with USER and PASS.
165...	125.434155	dl.xs4all.nl	10.0.2.2	FTP	88	Request: PORT 10,0,2,2,199,51
165...	125.542612	dl.xs4all.nl	10.0.2.2	FTP	104	Response: 530 Please login with USER and PASS.

Frame 14602: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface 0
Ethernet II, Src: Apple_e5:66:07 (00:26:08:e5:66:07), Dst: Apple_d8:0f:fa (04:0c:ce:d8:0f:fa)
Destination: Apple_d8:0f:fa (04:0c:ce:d8:0f:fa)
Address: Apple_d8:0f:fa (04:0c:ce:d8:0f:fa)

Offset	Hex	ASCII
0000	04 0c ce d8 0f fa 00 26 08 e5 66 07 08 00 45 00&..f...E..
0010	00 3a aa b0 40 00 35 06 b7 5c c2 6d 15 42 0a 00	...@.5...m.B...
0020	02 02 00 15 c7 08 b8 62 2c a1 08 77 b9 ab 80 18b...w.....
0030	00 72 51 f8 00 00 01 01 08 0a 55 c0 86 12 35 ae	...rQ.....U...5..
0040	51 65 32 32 30 2d 0d 0a	Qe220-...

FTP - It is not the correct method to transfer packets as no encryption is there therefore anyone can intercept messages and This is also susceptible to man in the middle attack.

The more secure method is HTTPS and FTPS as it does encryption

d) Another HTTPS site other than facebook is bouazizi.torserver.net

No.	Time	Source	Destination	Protocol	Length	Info
143...	82.910678	10.0.2.3	bouazizi.torserver...	TCP	54	55586 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
143...	82.913496	10.0.2.3	bouazizi.torserver...	TLSv1	197	Client Hello
143...	82.913504	10.0.2.3	96.44.189.101.stat...	TCP	54	55587 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
143...	82.915146	10.0.2.2	10.0.2.1	DNS	73	Standard query 0xfaba A ftp.mirror.nl
143...	82.916229	10.0.2.3	96.44.189.101.stat...	TLSv1	194	Client Hello
143...	82.930525	www.google.com	10.0.2.3	HTTP	190	HTTP/1.1 304 Not Modified
143...	82.943759	bouazizi.torserver...	10.0.2.3	TCP	54	443 → 55586 [ACK] Seq=1 Ack=144 Win=7168 Len=0
143...	82.951207	96.44.189.101.stat...	10.0.2.3	TCP	54	443 → 55587 [ACK] Seq=1 Ack=141 Win=16384 Len=0
143...	82.952196	96.44.189.101.stat...	10.0.2.3	TLSv1	987	Server Hello, Certificate, Server Key Exchange, Server He
143...	82.962212	10.0.2.3	96.44.189.101.stat...	TLSv1	252	Client Key Exchange, Change Cipher Spec, Encrypted Handsh
143...	82.965082	bouazizi.torserver...	10.0.2.3	TLSv1	994	Server Hello, Certificate, Server Key Exchange, Server He
143...	82.975882	10.0.2.3	bouazizi.torserver...	TLSv1	252	Client Key Exchange, Change Cipher Spec, Encrypted Handsh
143...	82.995787	w40.b.cap-mii.net	10.0.2.2	TCP	66	80 → 50927 [FIN, ACK] Seq=989 Ack=866 Win=5244 Len=0 TSva
143...	82.996943	10.0.2.2	w40.b.cap-mii.net	TCP	66	50927 → 80 [ACK] Seq=866 Ack=990 Win=131072 Len=0 TSval=9
143...	82.999455	96.44.189.101.stat...	10.0.2.3	TLSv1	304	New Session Ticket, Change Cipher Spec, Encrypted Handsh
143...	83.004212	10.0.2.3	96.44.189.101.stat...	TLSv1	187	Encrypted Handshake Message
143...	83.004942	tor-exit3-readme.d...	10.0.2.3	TCP	66	35440 → 55588 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=
142	82.008200	10.0.2.2	tor-exit3-readme.d...	TCP	54	55588 → 25440 [ACK] Seq=1 Ack=1 Win=65536 Len=0

▼ Frame 14369: 994 bytes on wire (7952 bits), 994 bytes captured (7952 bits)
 ▼ Ethernet II, Src: Apple_e5:66:07 (00:26:08:e5:66:07), Dst: IntelCor_50:f0:a6 (8c:a9:82:50:f0:a6)
 ▼ Destination: IntelCor_50:f0:a6 (8c:a9:82:50:f0:a6)
 Address: IntelCor_50:f0:a6 (8c:a9:82:50:f0:a6)

```

0000  8c a9 82 50 f0 a6 00 26 08 e5 66 07 08 00 45 00  ...P...&...f...E...
0010  03 d4 af 05 40 00 34 06 30 20 4a 78 0d 84 0a 00  ...@.4. 0 Jx...
0020  02 03 01 bb d9 22 24 a2 a8 21 ff 42 ec a4 50 18  ..."$...! B...P...
0030  00 07 a1 c8 00 00 16 03 01 00 35 02 00 00 31 03  ........5...1...
0040  01 50 72 59 67 b4 30 c4 30 31 8b 7a 27 1b c0 c2  -PrYg-0 01 z'...
0050  5b 47 70 ba 42 ee a9 d5 a9 4b af 4c 80 33 16 41  [Gp-B...K-L-3-A
0060  86 00 00 39 00 00 09 ff 01 00 01 00 00 23 00 00  ...9.....#...
  
```

Packets: 22650 - Disallowed: 22650 (100.0%) Profile: Default

Trust, Integrity and SEO is there with https.

It is better as it is not susceptible to man in the middle attack as it does encryption and provides proper security protocols.

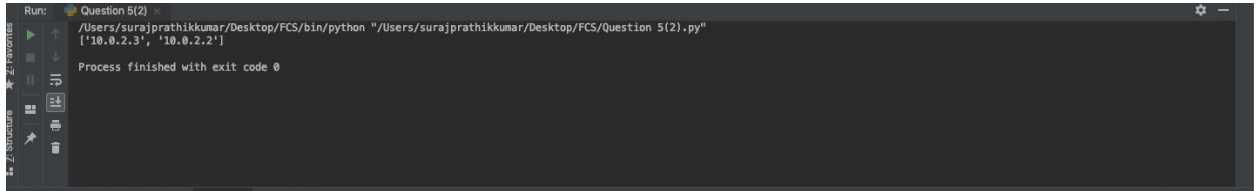
e) Even though logins are processed over HTTPS but the security property is violated in case of the facebook at the time of authentication of the user. As the user doesn't have to provide a valid proof of himself. NO CA assigned certificates to the user. A user can have multiple accounts and claim to be different in all of them. With only credentials fb can verify the user and if they are compromised no check is there.

2. Code has been uploaded with the pdf

To Run : install dpkt and run on python 3 and above

The eth data is extracted from the file to find IP Packets. This IP packets are filtered to get TCP packets. TCP packets with SYN packets with 2 way handshake are filtered and printed in the output.

10.0.2.3 and 10.0.2.2 are the malicious IP's



The screenshot shows a terminal window titled "Question 5(2)". The command executed is `/Users/surajprathikkumar/Desktop/FCS/bin/python "/Users/surajprathikkumar/Desktop/FCS/Question 5(2).py"`. The output of the script is `['10.0.2.3', '10.0.2.2']`. Below the output, it says "Process finished with exit code 0". The terminal window has a dark background and a light-colored border.