FIS Lab 1: Crypto Basics

Name: Mohamad Nour Shahin

Group number: B22-CBS-01

Questions to Answer

- Upload all relevant program/scripts created to Moodle.
- Show all relevant input, commands executed, and output in the form of screenshots.
- Use tunnelshell for the ICMP/DNS format in steganography. You can run it on 2 virtual machines, you can also pair with someone else and use their computer as the victim or the attacker.

Task 1

Write a program that implements **one** of the following algorithms for both encryption and decryption.

- Substitution Cipher
- Transposition Cipher
- Rotor Machines or Simple XOR

Hint
http://www.crypto-it.net/eng/simple/index.html

Solution:

I choosed the Substitution Cipher, and the code below represnet it.

Code:

Encryption and Decryption Program

```
#include <bits/stdc++.h>
using namespace std;

// Declare global strings for language, plain text, cipher text, and
decrypted cipher
string language, plain_text, cipher_text, decrypt_cipher;

int main()
{
    // Declare a key for shifting letters
    int key;

    // Declare maps to store encryption and decryption mappings
    map<char, char> dictionary_lang, dictionary_cipher;
```

```
    // Prompt user to enter the language (alphabet) used for encryption
    cout << "Enter your language letters please:\n";
    cin >> language;

    // Prompt user to enter a key for shifting letters
    cout << "Enter the key you want to use in hashing:\n";
    cin >> key;

    // Ignore leftover newline character in the input buffer
    cin.ignore();

    // Create the encryption dictionary by shifting letters by the key
value
    for (int i = 0; i < language.length(); i++)
    {
        dictionary_lang[language[i]] = language[(i + key) %
language.length()];
    }

    // Prompt user to enter the plain text to be encrypted
    cout << "Enter your plain text please:\n";
    getline(cin, plain_text);

    // Encrypt the plain text using the encryption dictionary
    for (char &c : plain_text)
    {
        if (dictionary_lang.find(c) != dictionary_lang.end())
        {
            cipher_text += dictionary_lang[c]; // Replace with encrypted
character
        }
        else
        {
            cipher_text += c; // Leave non-language characters unchanged
        }
    }

    // Output the cipher text (encrypted text)
    cout << "Your cipher text is: " << cipher_text << "\n";

    // Create the decryption dictionary by shifting letters backwards by
the key value
    for (int i = 0; i < language.length(); i++)
    {
        dictionary_cipher[language[i]] = language[(i - key +
language.length()) % language.length()];
    }

    // Decrypt the cipher text using the decryption dictionary
    for (char &c : cipher_text)
    {
        if (dictionary_cipher.find(c) != dictionary_cipher.end())
        {
```

```
            decrypt_cipher += dictionary_cipher[c]; // Replace with
decrypted character
        }
        else
        {
            decrypt_cipher += c; // Leave non-language characters unchanged
        }
    }

    // Output the recovered plain text (decrypted text)
    cout << "Recovered plain text : " << decrypt_cipher << "\n";
    return 0;
}
```

```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ g++ SubstitutionCipher.cpp
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ ./a.out
Enter your lanuage letters please:
abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ
Enter the key you want to use it in hashing:
4
Enter your plain text please:
I am studying Data Encryption
Your cipher text is: M eq wxyhCmrk Hexe IrgvCtxmsr
Recovered plain text : I am studying Data Encryption
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$
```

Task 2

Hide data (e.g text) in another data file format. The data should be hidden in the following formats

- Video
- Audio
- ICMP/DNS

Solution:

1. Video: I will use third party tool from github JavDomGom/videostego.

- clone the repo and build it:

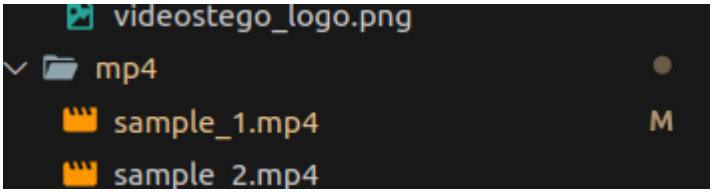
```
git clone https://github.com/JavDomGom/videostego.git
cd videostego
make build
```

```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ git clone https://github.com/JavDomGom/videostego.git
Cloning into 'videostego'...
remote: Enumerating objects: 62, done.
remote: Counting objects: 100% (62/62), done.
remote: Compressing objects: 100% (51/51), done.
remote: Total 62 (delta 14), reused 55 (delta 10), pack-reused 0 (from 0)
Receiving objects: 100% (62/62), 3.20 MiB | 2.15 MiB/s, done.
Resolving deltas: 100% (14/14), done.
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ cd videostego
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1/videostego$ make build
2024/08/31 12:05:59 [build] Building videostego binary ...
gcc -Wall -g -o videostego src/*.c
2024/08/31 12:05:59 [build] Done!
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1/videostego$
```

- I will use one of the video samples to hide the data that I want to hide it:

```
./videostego -f ./mp4/sample_1.mp4  -w -m "Hello, my name is Mohamad Nour"
```

```
2024/08/31 12:05:59 [build] Done!
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1/videostego$ ./videostego -f ./mp4/sample_1.mp4  -w -m "Hello, my name is Mohamad Nour"
```



- to check the hidden message inside the video i will use the command:

```
./videostego -f ./mp4/sample_1.mp4 -r
```

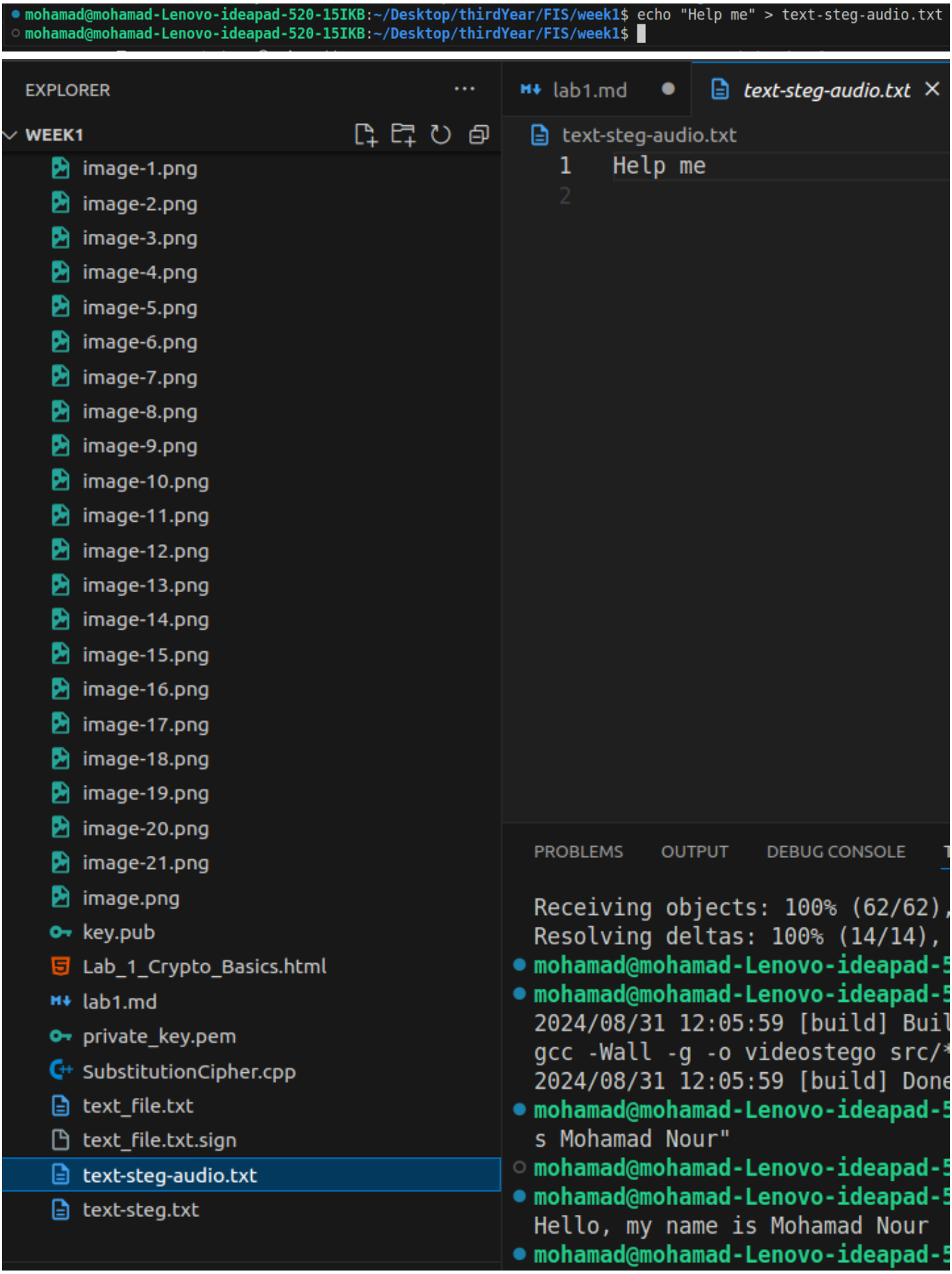
```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1/videostego$ ./videostego -f ./mp4/sample_1.mp4 -r
Hello, my name is Mohamad Nour
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1/videostego$
```

---

2. Audio: I will use steghide from lab itself:

- Create a text file that contains the message you want to embed in the audio:

```
echo "Help me" > text-steg-audio.txt
```



- Embed the text in the audio:

```
steghide embed -cf file_example_WAV_1MG.wav -ef text-steg-audio.txt
```

```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ steghide embed -cf file_example_WAV_1MG.wav -ef text-steg-audio.txt
Enter passphrase:
Re-Enter passphrase:
embedding "text-steg-audio.txt" in "file_example_WAV_1MG.wav"... done
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$
```

- View the embedded text after deleting the txt file:

```
steghide extract -sf file_example_WAV_1MG.wav -xf tes.txt
```

```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ steghide extract -sf file_example_WAV_1MG.wav -xf tes.txt
Enter passphrase:
wrote extracted data to "tes.txt".
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$
```

```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ cat tes.txt
Help me
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$
```

3. ICMP: I did it with my friend Ali hamdan's computer as victim:

check the ips of each device:

- mine:

```
root@mohamad-Lenovo-ideapad-520-15IKB:/home/mohamad# ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:50:51:fb:6d  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp2s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 54:e1:ad:f7:05:56  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ham0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1404
        inet 25.11.58.91  netmask 255.0.0.0  broadcast 25.255.255.255
        inet6 2620:9b::190b:3a5b  prefixlen 96  scopeid 0x0<global>
        inet6 fe80::7879:19ff:fe0b:3a5b  prefixlen 64  scopeid 0x20<link>
        ether 7a:79:19:0b:3a:5b  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 105  bytes 16190 (16.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 8160  bytes 5164659 (5.1 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8160  bytes 5164659 (5.1 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.91.57.107  netmask 255.255.240.0  broadcast 10.91.63.255
        inet6 fe80::d37d:3e54:b529:b2ad  prefixlen 64  scopeid 0x20<link>
        ether 40:a3:cc:10:09:7d  txqueuelen 1000  (Ethernet)
        RX packets 12846  bytes 11455333 (11.4 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8868  bytes 2876419 (2.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

root@mohamad-Lenovo-ideapad-520-15IKB:/home/mohamad#
```

- Ali:

```
wlp7s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.91.55.205  netmask 255.255.240.0  broadcast 10.91.63.255
        inet6 fe80::154d:411b:6864:c594  prefixlen 64  scopeid 0x20<link>
        ether d0:53:49:c0:25:94  txqueuelen 1000  (Ethernet)
        RX packets 7176  bytes 7353141 (7.3 MB)
        RX errors 0  dropped 0  overruns 0  frame 2896
        TX packets 3771  bytes 935192 (935.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 18

ali@ali-workstation:~$ sudo ./tunneld -t icmp -m echo-reply,echo
```

- Run tunnelshell on the victim machine:

```
sudo ./tunneld -t icmp -m echo-reply,echo
```

- Run tunnelshell on the attacker's machine with:

```
sudo ./tunnel -t icmp -m echo-reply,echo 10.91.55.205
```

Task 3

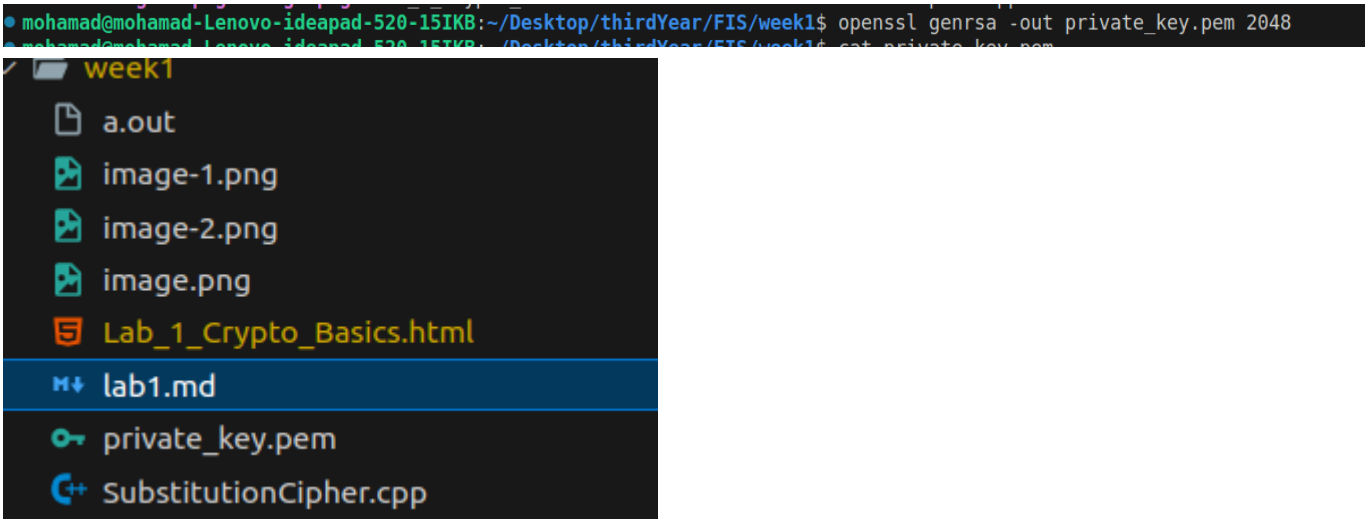
Generate a RSA keypair of key length 2048-bit using OpenSSL. Write your first name in a text file, sign, and verify the integrity of the text file. Your answer should include these:

- Generate the private key with OpenSSL
- Extract the public key from the private key
- Create a text file that includes your first name
- Verify the digital signature using OpenSSL digest (dgst)

Solution:

1. Generate the private key with OpenSSL using Command:

```
openssl genrsa -out private_key.pem 2048
```



2. view the private key file using Command:

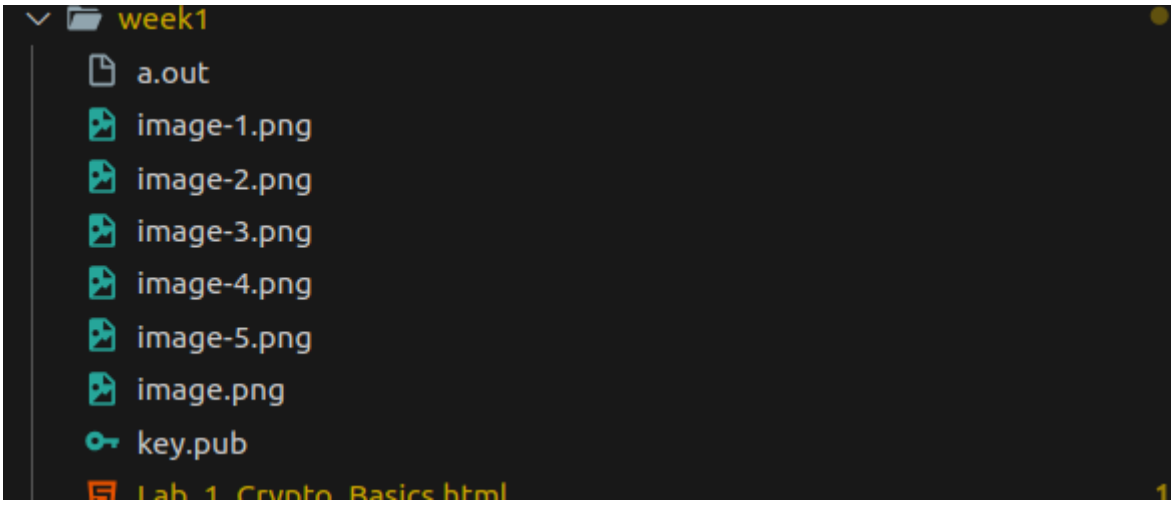
```
cat private_key.pem
```

```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ cat private_key.pem
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDtieH86kNfmZZo
V1tFZZxweOMmPs/sHxNSRfvi8fE0AlXht4p8JITzi0Qz+j3dTMJ2VMecJHjOXwW+
xoENnzpfa8MshooTUgciBHGB4473dlhg2VrSr80FK6XOvm6zEh8o7bFSF9cBN7I9
5FdzWnBDCj4kKZMyu2Y/BXuO2BAnQ28kwuAeMKRiZ+IEUz31avRWnbRg7MTpSHmv
pkGOYHJfW4XkdeunaqlSHM4LRD0hFwO3sBfN6qVASkES5ftg7cSpzCDhLuPLxnzn
FesxPYFmlnqwPhR6bXpeuwtJkhy31j1kY3sr0ofGq1yhIJL5fDg+LwNjzgluiu/a
CJLMRa7hAgMBAAECggEAB3kJvsDo7zJAANbUtH1ymzg2Xy7lMCa3HyS7/KMDZVE4
ffId/XNRqm7EbcUu2o4uZPYcVH1yPn3PSrXoe/L5LWTQvk8aO2redhPBNWahCghz
VdKosI2ey82YeQVVKrWEfO9jX3OIDE4CAb/z/1lGRFmnZFBZhORP3mYwAamq8m01
ZXhcULBk7lpLf2AWMAtxw+wQcNwZ4j+K5edss1kfbCYd6ShlfSuyK2fSezPrH/+T
tjSiJLjy+wB3M3BkJI6ol6OnAwo32JOvGZpEFgxs2RhvJiZj6nEpLb+SSjZ/63bf
E6twufII0ExNTFeb00FCk2DbTmHcYzqPARCoTmLz1QKBgQD90SRVzVtGHLNMSidq
JAw6JcQ3xnVzXjTmeAnkC5KIhZFVS/Re03pmNovkUla9yRnySIQW96yly8lrITyh
A/lxDM/y+SvKn9SwOmDwsNfLzhHn0oOBi8OqxeWUg23ob8OiWfFE1uWN55JngyuJ
qQI+y2fVOKYbaek5bqRkjlCtvQKBgQDvlOYeKSLlLe43dr0IOxTPUNABSNo/tos4
5JDGFlX/TDw6rmIeKAHZ/P26UZpL2yUtxWy2crYah4nX+TloJueUu4fZA8QJNe+q
rfcazs6UqHggb7Zn/dVJAw7i20ac5WraDNWO6FdC5u/IzYJJLxCr27T/g7JnYSxg
ot+mJnUd9QKBgCLp/PyMhgyr8AIDhU51bPUmssoVU5yuDcrSIlIrQVnkPaylCRVu
M70RrcaXiPUMT7RjQ0BBi4827Br/dG6VtF9Xqtc6bUqvF7uyHBmfyd9wFkh0TcCp
MB4yoLUAK0M49IlFpAmbIfKWMy5n/bou9G8c3ciXxUmqFPngfV6TsQdlAoGBAI3a
x/YLvVGdRHMqpvp8oPGvk7lu/WrEFuL00pO3Hewvo2nLXuUkIxSDK2hqcMWBWB4l
yxysBw7mOcPbGRaGEpCKCQGBrU0W1nVk0+7XjJGQvq/V9VbVgLfeS64ez0yOuhmp
h7k58JD4U/m9VOsoJuLbD+xKehOkAbamOW4TTz0RAoGBANdLnmxwI9qzje9IRnNV
sx9VbQ6RcoIHHssqyO/y1WQoI24hq+Onr8L1ZKEliA4490Beeh5XNaVw/Z7+7FkW
e4zt0RwkKr0aIovdgxbPmJMmm53/Ytz1Jq3PsQjYPXAgUERfk2LwgFzcnOr0mFTo
BR8Rl8G3cVlt96L/4kuDwaz0
-----END PRIVATE KEY-----
```

3. Extract the public key from the private key using Command:

```
openssl rsa -in private_key.pem -pubout > key.pub
```

```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ openssl rsa -in private_key.pem -pubout > key.pub
writing RSA key
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$
```



4. View the public key file using command:

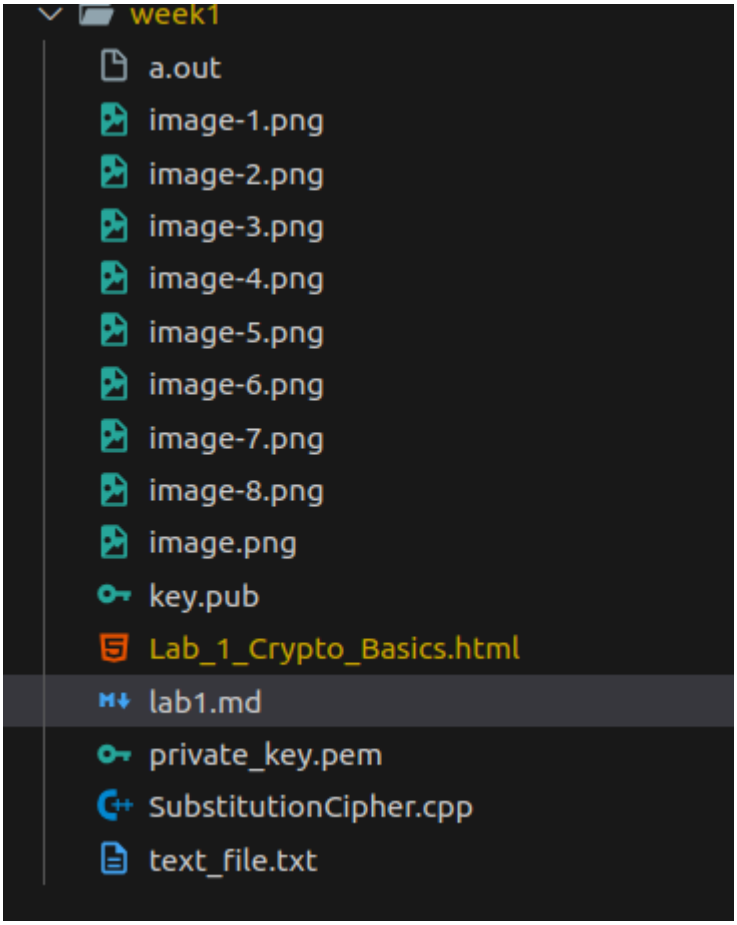
```
cat key.pub
```

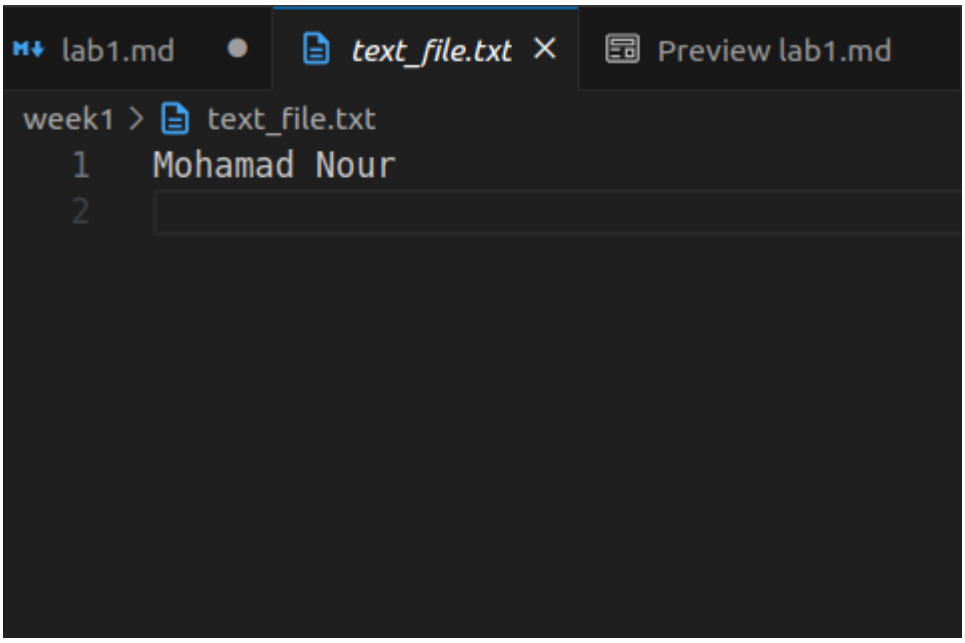
```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ cat key.pub
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA7Ynh/OpDX5mWaFdbRWWc
cHjjJj7P7B8TUkX74vHxNAJV4beKfCSE84tEM/o93UzCdlTHnCR4zl8FvsaBDZ86
X2vDLIaKE1IHIgRxgeOO93ZYYNla0q/NBSulzr5usxIfKO2xUhfXATeyPeRXc1pw
Qwo+JCmTMrtmPwV7jtgQJ0NvJMLgHjCkYmfiBFM99Wr0Vp20YOzE6Uh5r6ZBjmBy
X1uF5HXrp2qpUhzOC0Q9IRcDt7AXzeqlQEpBEuX7YO3Eqcwg4S7jy8Z85xXrMT2B
ZpZ6sD4Uem16XrsLSZIct9Y9ZGN7K9KHxqtcoSCS+Xw4Pi8DY84Jborv2giSzEWu
4QIDAQAB
-----END PUBLIC KEY-----
```

5. Create a text file that includes your first name:

```
echo 'Mohamad Nour' > text_file.txt
```

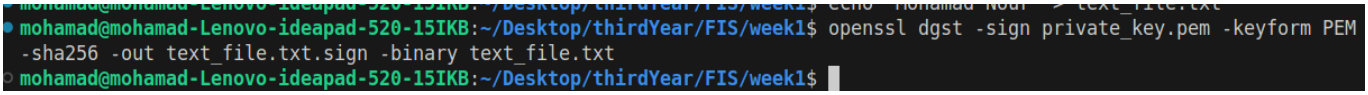
```
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$ echo 'Mohamad Nour' > text_file.txt
mohamad@mohamad-Lenovo-ideapad-520-15IKB:~/Desktop/thirdYear/FIS/week1$
```

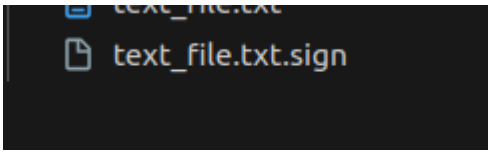




6. Sign the text file with OpenSSL digest (dgst) using command :

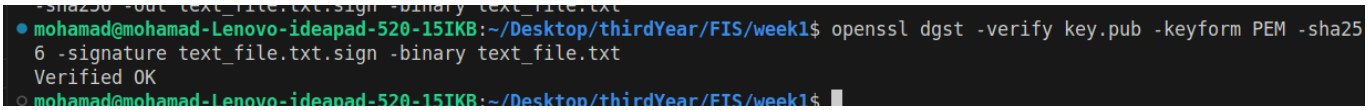
```
openssl dgst -sign private_key.pem -keyform PEM -sha256 -out
text_file.txt.sign -binary text_file.txt
```





7. Verify the digital signature using OpenSSL digest (dgst) using command:

```
openssl dgst -verify key.pub -keyform PEM -sha256 -signature
text_file.txt.sign -binary text_file.txt
```



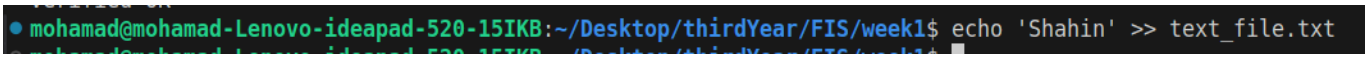
Task 4

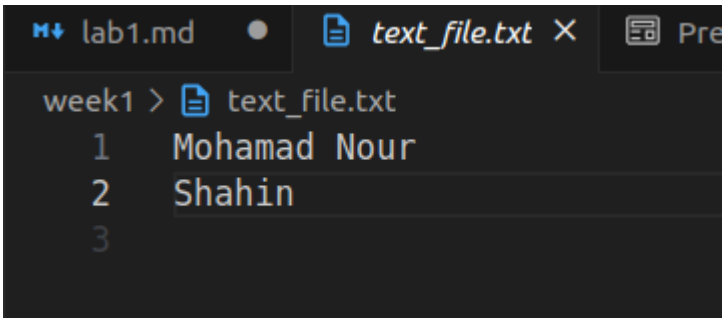
Add your last name to the text file from task 3. Now verify the text file by using the previous signature you created for your first name. Is the verification succesful?

Solution:

1. Add your last name to the text file from task 3 using Command:

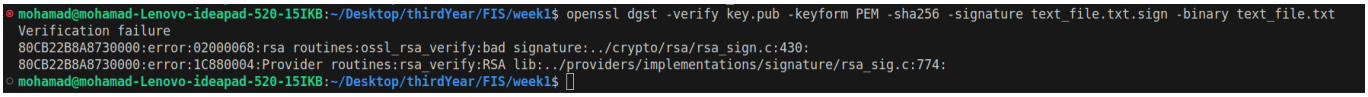
```
echo 'Shahin' >> text_file.txt
```





2. verify the text file by using the previous signature you created for your first name using command:

```
openssl dgst -verify key.pub -keyform PEM -sha256 -signature
text_file.txt.sign -binary text_file.txt
```



conclusion:

The verification was not successful because the signature was created for the first name only and not for the full name.

Task 5

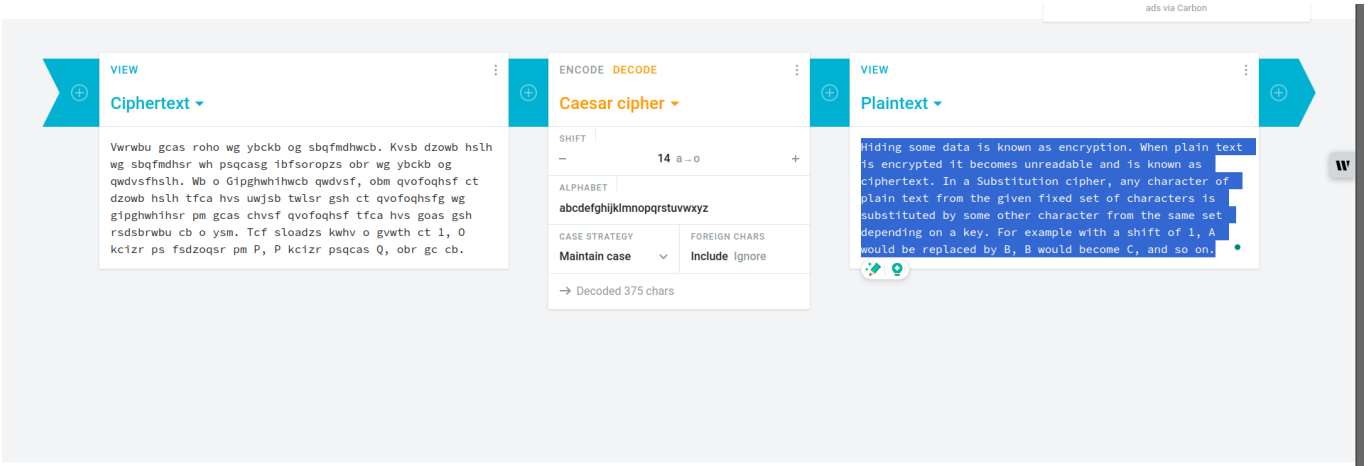
Decode the following ceasar cipher: **Decode the following Caesar cipher:**

> Vwrwbu gcas roho wg ybckb og sbqfmdhwcb. Kvsb dzowb hslh wg sbqfmdhsr wh psqcasg ibfsoropzs obr wg ybckb og qwdvsfhslh. Wb o Gipghwhihwcb qwdvsf, obm qvofoqhsf ct dzowb hslh tfca hvs uwjsb twlsr gsh ct qvofoqhsfg wg gipghwhihsr pm gcas chvsf qvofoqhsf tfca hvs goas gsh rsdsbrwbu cb o ysm. Tcf sloadzs kwhv o gvwth ct 1, O kcizr ps fsdzoqsr pm P, P kcizr psqcas Q, obr gc cb.

Solution:

The key used for shifting is 14:

Hiding some data is known as encryption. When plain text is encrypted it becomes unreadable and is known as ciphertext. In a Substitution cipher, any character of plain text from the given fixed set of characters is substituted by some other character from the same set depending on a key. For example with a shift of 1, A would be replaced by B, B would become C, and so on.



Caesar cipher: Encode and decode online

Referneces:

Task 3,4 Task 2 Task 1 Task 5