Group Number 25

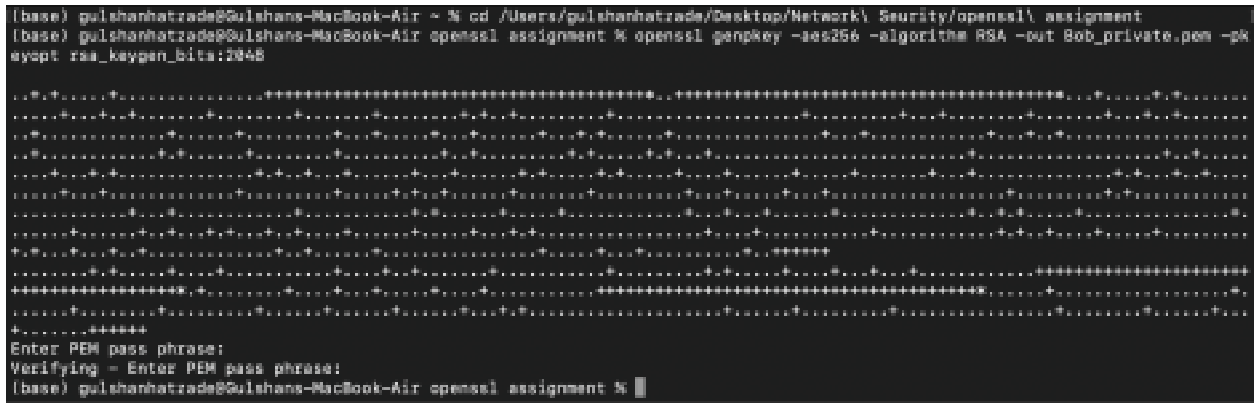
Alice - Anurag Sarva CS24MTECH14003
Bob - Gulshan Hatzade CS24MTECH14006

Part A: Authentication Using RSA

Step 1: Generate RSA Key Pairs

<u>Bob</u>

Command : openssl genpkey -aes256 -algorithm RSA -out Bob_private.pem -pkeyopt rsa_keygen_bits:2048

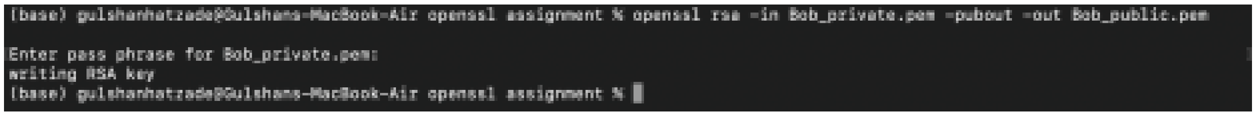


```
(base) gulshanhatzade@Gulshans-MacBook-Air ~ % cd /Users/gulshanhatzade/Desktop/Network\ Seurity/openssl\ assignment
(base) gulshanhatzade@Gulshans-MacBook-Air openssl assignment % openssl genpkey -aes256 -algorithm RSA -out Bob_private.pem -pk
eyopt rsa_keygen_bits:2048

..+.+.....+...............+++++++++++++++++++++++++++++++++++++++*..+++++++++++++++++++++++++++++++++++++++++*...+.....+.+.......
.....+...+..+.......+........+.......+........+.+..+.........+..................+.........+...+........+.......+...+..+.......
..+.............+......+.........+...+.....+...+.......+...+.+......+................+...+............+...+..+.................
..+...........+.+......+........+..........+..+.........+.+.....+.+...+......................+.....................+..+.....
....+...+.+..............+.+..+...+......+...+......+.+.....+.+.....+....+......+......+.....+.......+...+.............+.+...+..+....
.....+...+.............+.........+.....+.+..+......+.......+.........+...+.....+...+...................+.........+.+..........
...........+...+...........+..........+.+......+.....+...........+...+...+......+..................+..+.+.....+.............+.
......+......+..+...+.+...+..+....+......+.....+...+.+..................+....+...........+.............+.+..+....+.....+.......
+.+...+...+..+..............+..+......+................+.....+...+..........+..++++++
........+.+.....+....+..........+....+..+.......+..........+.........+.+.....+....+...+...+...........+++++++++++++++++++++
++++++++++++++++*.+.........+....+...+.....+....+...........+++++++++++++++++++++++++++++++++++++++++*......+.................+.
......+........+.........+......+......+......+...+.+...................+......+.........+................+........+......+...
+.......++++++
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
(base) gulshanhatzade@Gulshans-MacBook-Air openssl assignment %
```

This command generates a **private key** for Bob using the **RSA algorithm** with **2048-bit encryption** and **AES-256 protection** for extra security.

Command : openssl rsa -in Bob_private.pem -pubout -out Bob_public.pem



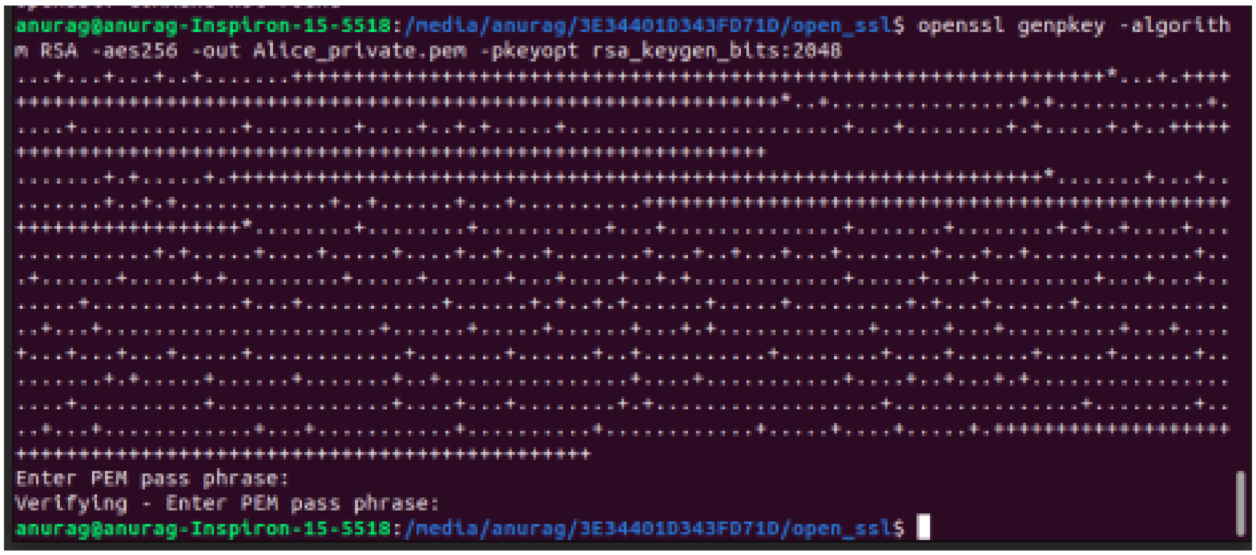
```
(base) gulshanhatzade@Gulshans-MacBook-Air openssl assignment % openssl rsa -in Bob_private.pem -pubout -out Bob_public.pem

Enter pass phrase for Bob_private.pem:
writing RSA key
(base) gulshanhatzade@Gulshans-MacBook-Air openssl assignment %
```

Extracts the **public key** from Bob's private key so it can be shared.

<u>Alice</u>

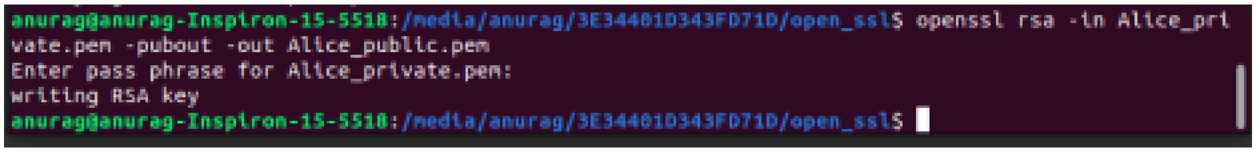
Command : openssl genpkey -aes256 -algorithm RSA -out Alice_private.pem -pkeyopt rsa_keygen_bits:2048



```
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/open_ssl$ openssl genpkey -algorith
m RSA -aes256 -out Alice_private.pem -pkeyopt rsa_keygen_bits:2048
...+...+...+..+.......+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*...+.++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*..+...............+.+............+.
....+.............+........+....+..+.+.....+.......................+...+........+.+.....+.+..+++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
.......+.+.....+.+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.......+...+..
.......+..+.+............+..+......+...+..........+++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++++*........+........+...........+...+...............+.......+.........+.+..+....+...
...........+.+.....+....+.....+....+..+...+.......+...+..+...+...+.......+...+..+.............+..
.+......+.....+.+..........+.....+......+...+....+..+.+..............+.....+...+..........+...+...+..
.....+............+...+...........+.......+.+..+.+......+.....+..........+.+...+.......+..........
..+...+.....................+.......+.....+......+...+.+.............+.....+...+.........+...+....
+...+...+...+.....+............+........+......+..+............+.........+....+......+.....+.....+..
.......+.+.....+.......+.......+..+...........+....+..............+....+..+...+.+...............
....+..........+..............+....+...+........+.+...................+...............+.........+..
..+...+............+...+...........+...........+.............+.....+....+.....+.+++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/open_ssl$
```

Same as Bob, but for Alice—**generating a private key**.

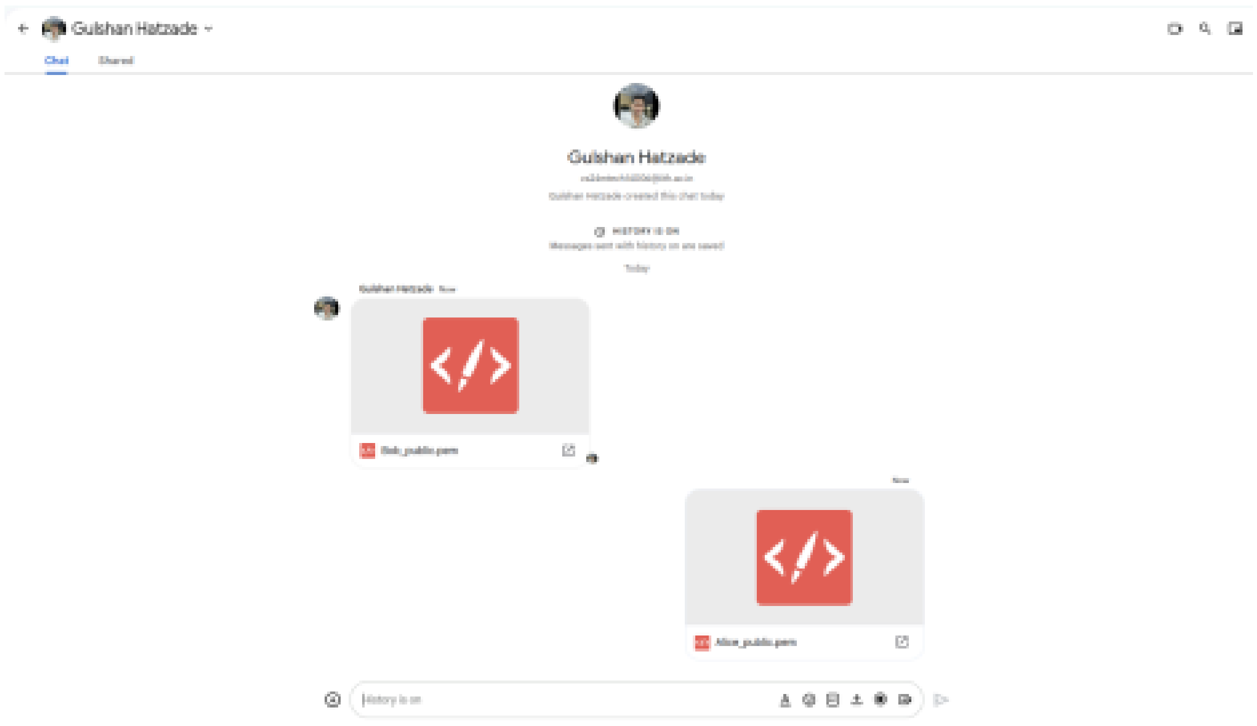
Command : openssl rsa -in Alice_private.pem -pubout -out Alice_public.pem



```
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/open_ssl$ openssl rsa -in Alice_pri
vate.pem -pubout -out Alice_public.pem
Enter pass phrase for Alice_private.pem:
writing RSA key
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/open_ssl$
```

This generates **Bob_private.pem**, **Bob_public.pem**, **Alice_private.pem**, and **Alice_public.pem**.

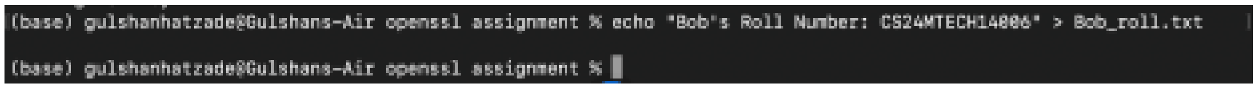
Step 2: Share Public Keys



Bob and Alice securely exchange their **public keys** by Google Chat.

Step 3: Create a Text File Containing Roll Number

Bob

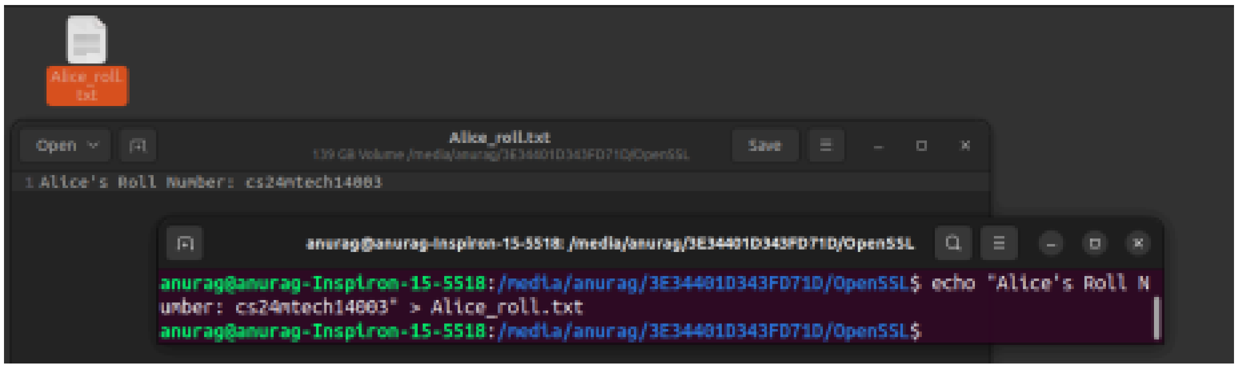


```
(base) gulshanhatzade@Gulshans-Air openssl assignment % echo "Bob's Roll Number: CS24MTECH14006" > Bob_roll.txt

(base) gulshanhatzade@Gulshans-Air openssl assignment %
```

Creates a simple text file containing Bob's **roll number**

Alice


Alice_roll.txt
Open
Alice_roll.txt
Save
anurag@anurag-Inspiron-15-5518: /media/anurag/3E34401D343FD71D/OpenSSL

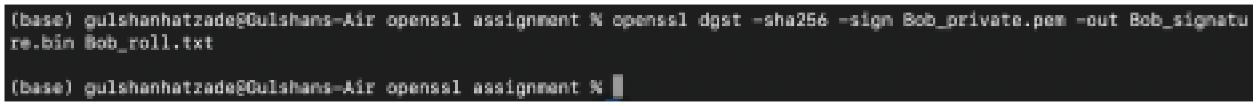
```
1 Alice's Roll Number: cs24mtech14003
```

```
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/OpenSSL$ echo "Alice's Roll N
umber: cs24mtech14003" > Alice_roll.txt
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/OpenSSL$
```

Same process, but for Alice.

Step 4: Generate Digital Signatures

Bob

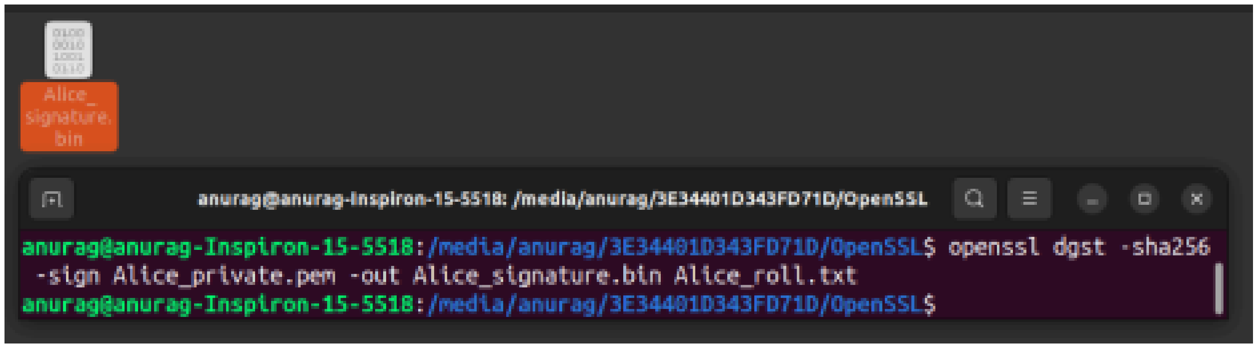


```
(base) gulshanhatzade@Gulshans-Air openssl assignment % openssl dgst -sha256 -sign Bob_private.pem -out Bob_signatu
re.bin Bob_roll.txt

(base) gulshanhatzade@Gulshans-Air openssl assignment %
```

Signs Bob’s roll number file using his **private key** and **SHA-256 hashing**.

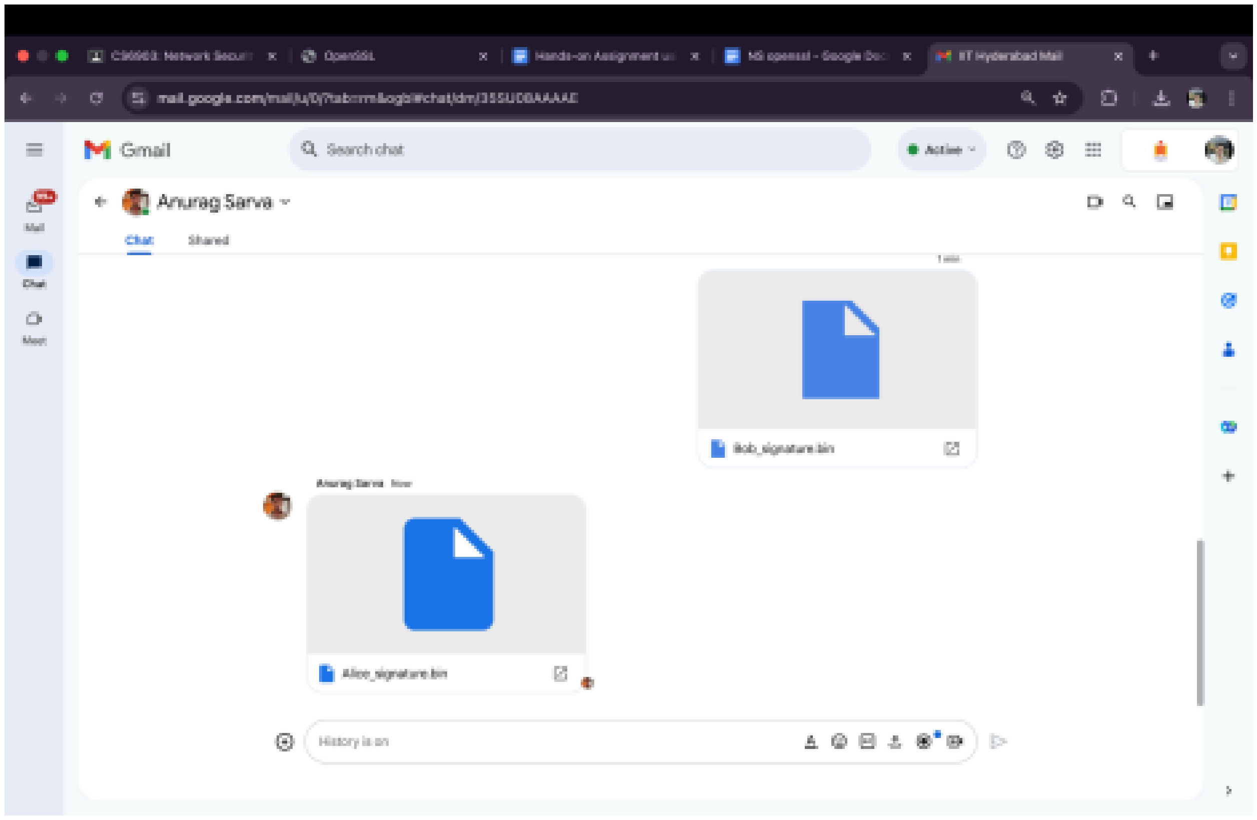
Alice



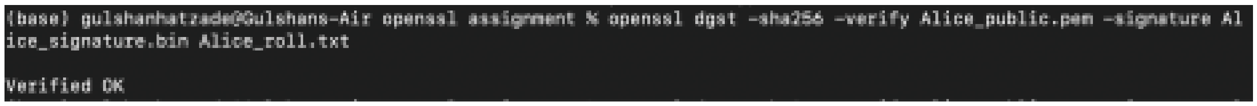
```
anurag@anurag-Inspiron-15-5518: /media/anurag/3E34401D343FD71D/OpenSSL
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/OpenSSL$ openssl dgst -sha256
 -sign Alice_private.pem -out Alice_signature.bin Alice_roll.txt
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/OpenSSL$
```

Alice does the same thing for her file—creating her **digital signature**.

Step 5: Exchange and Verify Digital Signatures



Bob Verifies Alice’s Signature

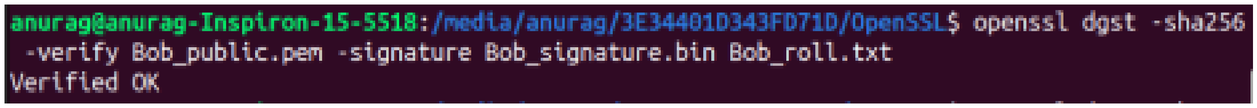


```
(base) gulshanhatzade@Gulshans-Air openssl assignment % openssl dgst -sha256 -verify Alice_public.pem -signature Al
ice_signature.bin Alice_roll.txt

Verified OK
```

Bob checks if **Alice's signature is valid** by using her **public key**. If everything is fine, it should show:

Alice Verifies Bob’s Signature

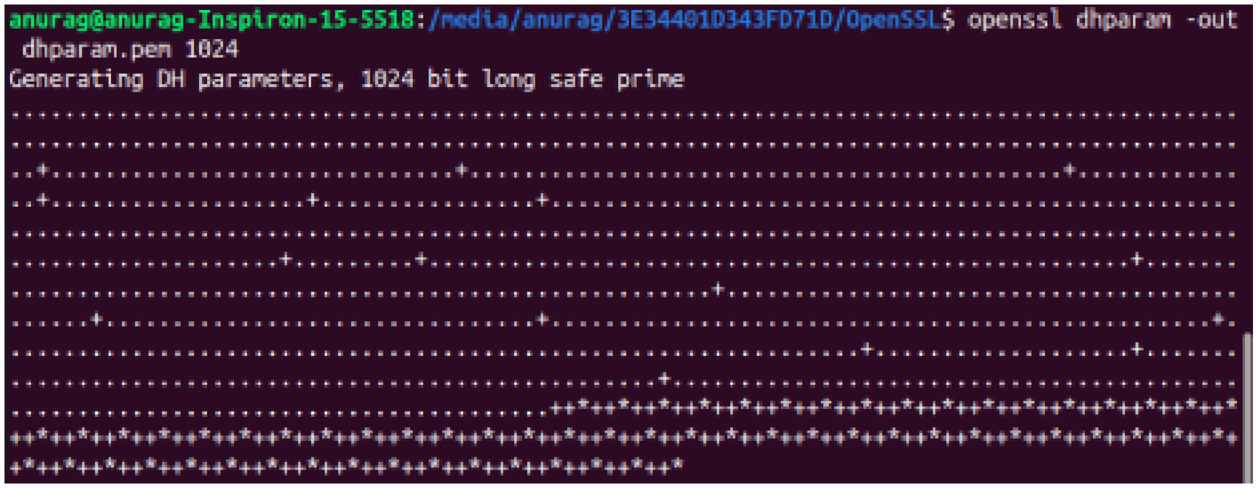


```
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/OpenSSL$ openssl dgst -sha256
 -verify Bob_public.pem -signature Bob_signature.bin Bob_roll.txt
Verified OK
```

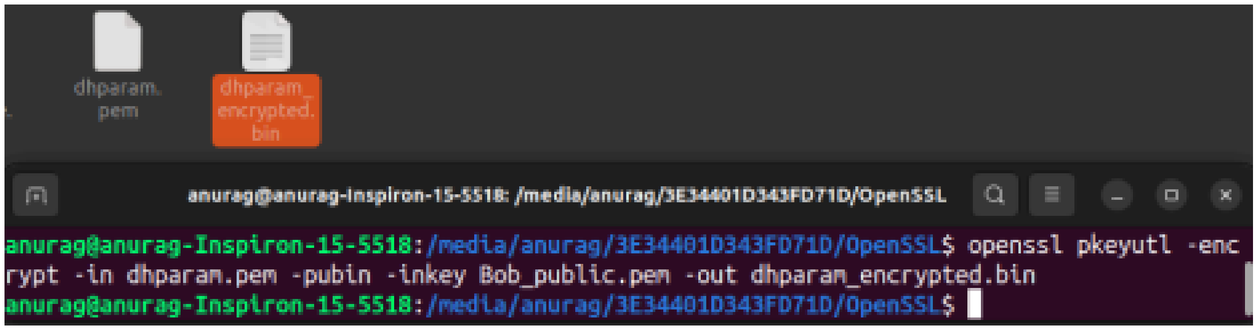
Alice does the same to confirm Bob’s signature.

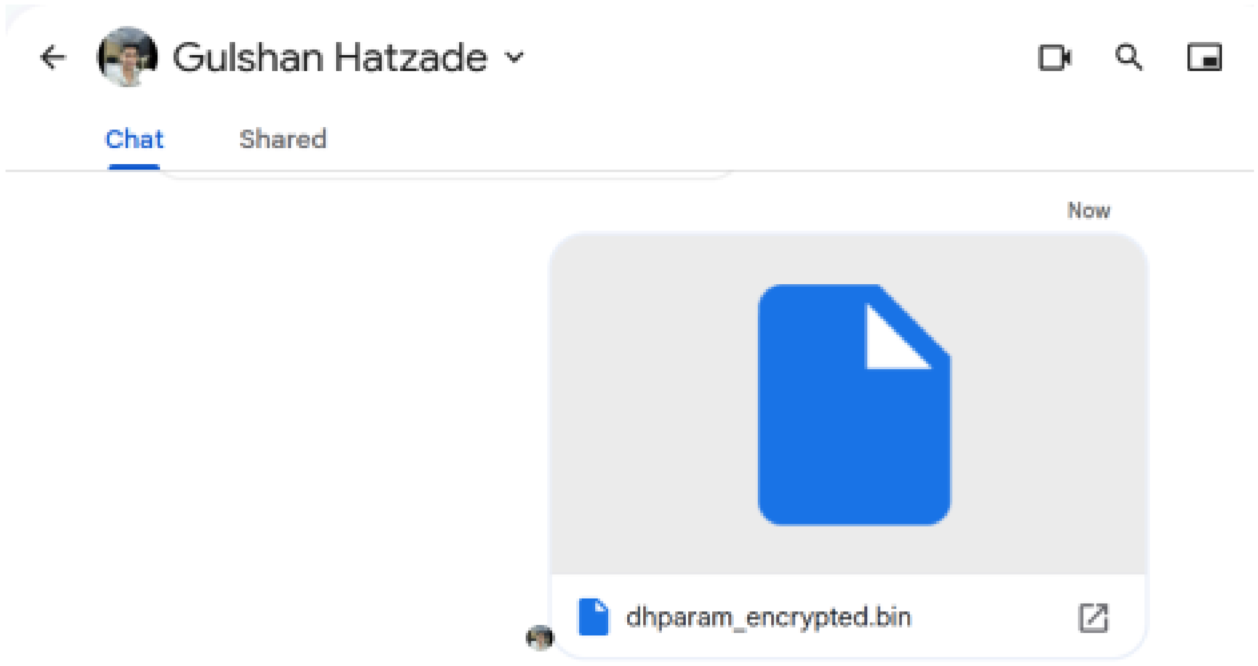
Part B: Key Exchange Using Diffie-Hellman (DFH)

Step 1: Generate DFH Parameters (Alice)


```
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/OpenSSL$ openssl dhparam -out
 dhparam.pem 1024
Generating DH parameters, 1024 bit long safe prime
..............................................................................
..............................................................................
..+...........................+.......................................+...........
..+...................+.................+.......................................
..............................................................................
.................+........+.............................................+......
.............................................+...................................
.....+............................+...........................................+.
......................................................+.................+......
..........................................+...................................
..................................++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*
++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*++*
+*++*++*++*++*++*++*++*++*++*++*++*++*++*++*
```
Alice creates **Diffie-Hellman parameters** (a shared structure needed for key exchange).

Step 2: Encrypt DFH Parameters with Bob’s Public Key


dhparam.
pem
dhparam_
encrypted.
bin
anurag@anurag-Inspiron-15-5518: /media/anurag/3E34401D343FD71D/OpenSSL
```
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/OpenSSL$ openssl pkeyutl -enc
rypt -in dhparam.pem -pubin -inkey Bob_public.pem -out dhparam_encrypted.bin
anurag@anurag-Inspiron-15-5518:/media/anurag/3E34401D343FD71D/OpenSSL$
```
Alice **encrypts** `dhparam.pem` using **Bob’s public key** before sending it to him.



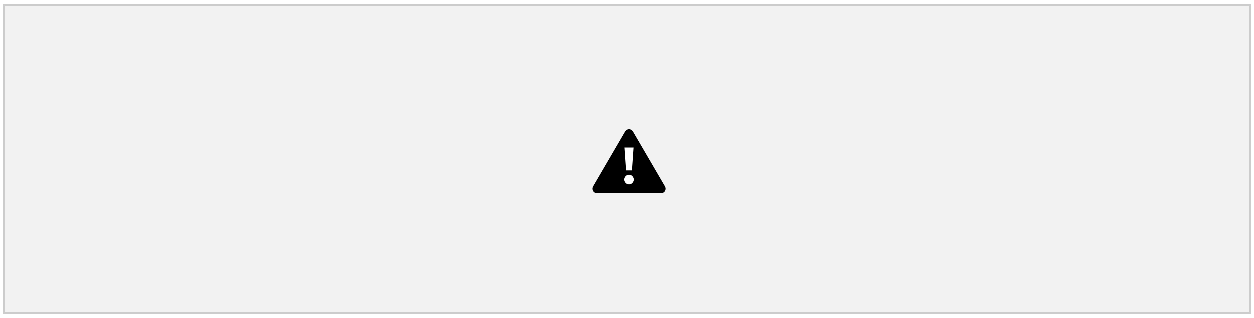
Step 3: Bob Decrypts the DFH Parameters

Bob **decrypts the received parameters** using his **private key**.

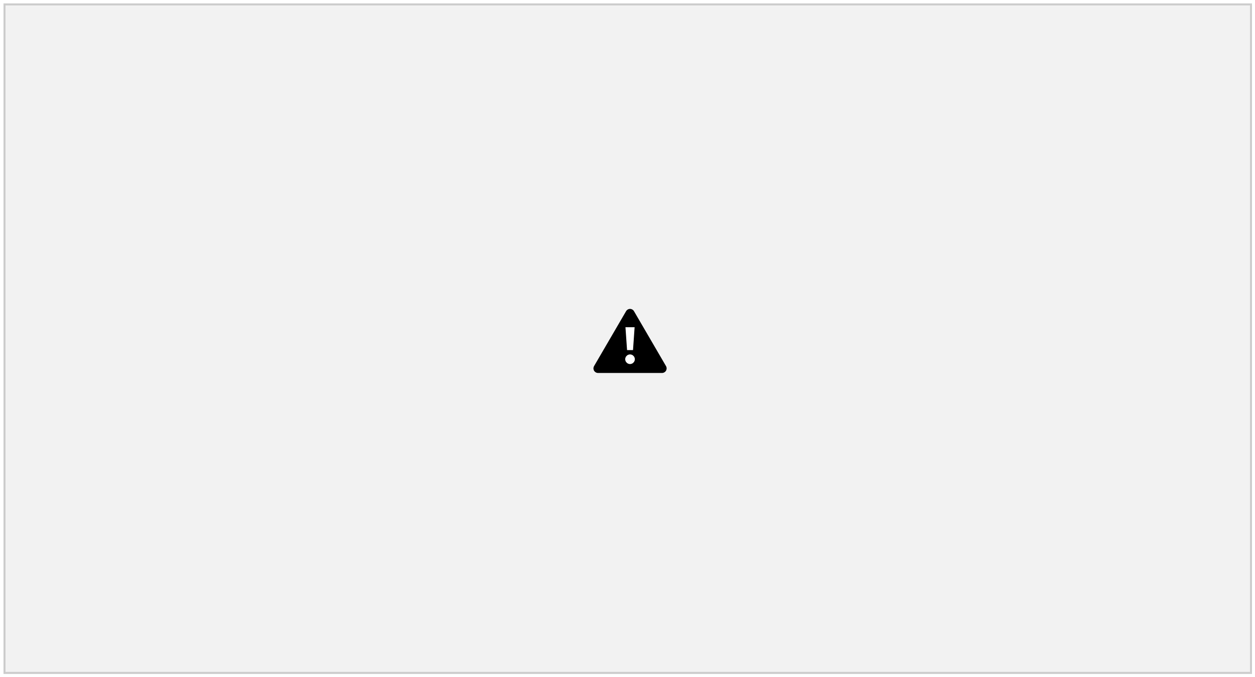
Step 4: Generate Diffie-Hellman Key Pairs

BOB

Bob generates **his private & public Diffie-Hellman keys**.
Alice


Alice does the same for her keys.

Step 5: Exchange and Compute the Shared Secret



Bob Computes Shared Secret

Bob calculates **the shared secret** using **his private key** and **Alice’s public key**.

Alice Computes Shared Secret

Alice does the same. The **shared secrets must match** for encryption to work.

Step 6: Derive AES Key from Shared Secret

Bob

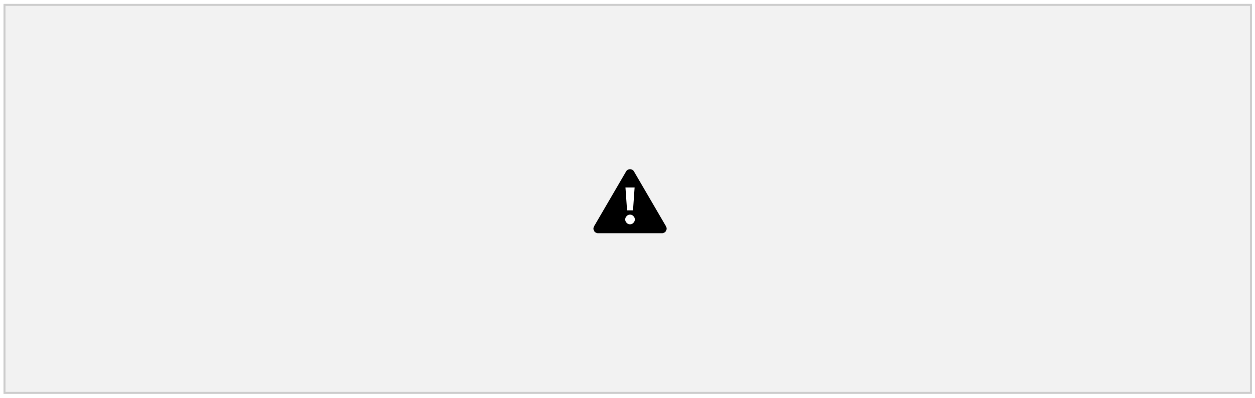
Bob **derives an AES key** from the shared secret using **SHA-256 hashing**.

Alice

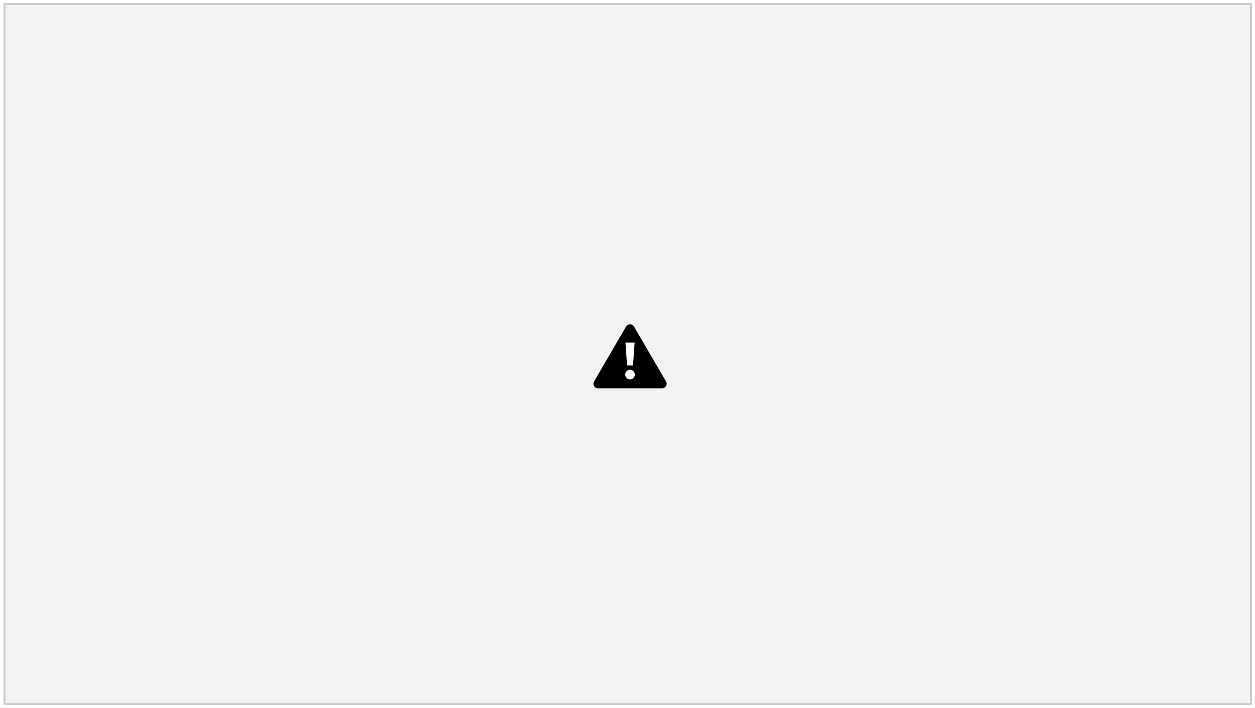
Alice does the same.

Part C: Secure File Sharing Using AES

Step 1: Alice Encrypts the File



Alice encrypts `file.txt` using the **AES key** she generated from **Diffie-Hellman shared secret**.



`file.enc` is the **encrypted file**, which Alice sends to Bob.

Step 2: Bob Decrypts the File

Bob **decrypts** `file.enc` using his **AES key**, restoring the original file.

Got the message

We got the message which matched with what was expected.

Anti-Plag Statement

We certify that this assignment/report is the result of our collaborative work, based on our collective study and research. All sources, including books, articles, software, datasets, reports, and communications, have been properly acknowledged. This work has not been previously submitted for assessment in any other course unless specific permission was granted by all involved instructors.

We also acknowledge the use of AI tools, such as LLMs (e.g., ChatGPT), for assistance in refining this assignment, if used. We have ensured that their usage complies with the academic integrity policies of this course. We pledge to uphold the principles of honesty, integrity, and responsibility at CSE@IITH.

Additionally, we understand our duty to report any violations of academic integrity by others if we become aware of them.

Names <Roll No.>: CS24MTECH14003, CS24MTECH14006

Date: 25/02/2025

Signatures: Anurag Sarva, Gulshan Hatzade