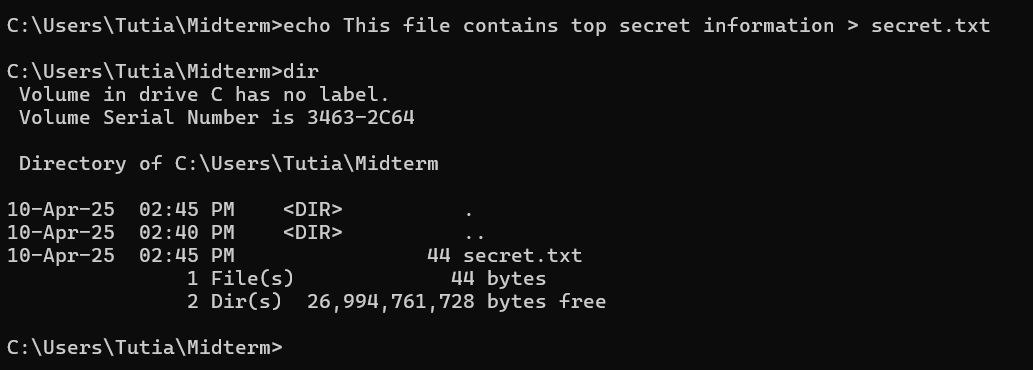
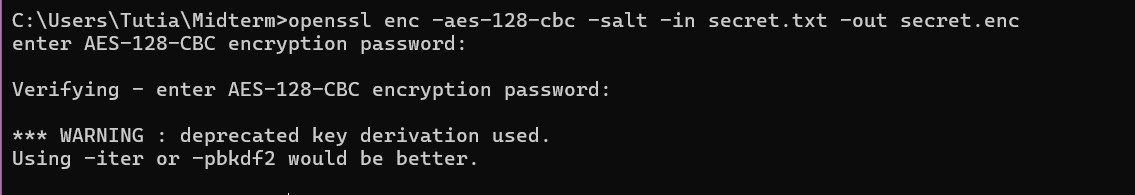
**Applied Cryptography – Midterm Lab Exam**

**Task 1A: Encrypt a file using AES-128-CBC (2.5 pts)**

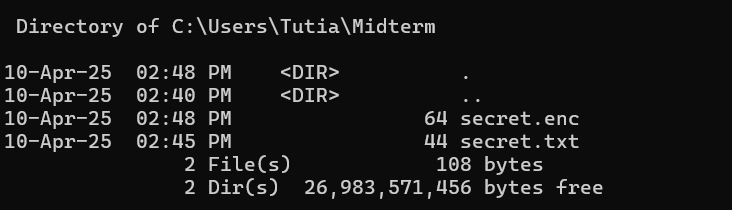
1. Create a text file: secret.txt with the line: This file contains top secret information.



1. Use OpenSSL to encrypt it with a passphrase of your choice.



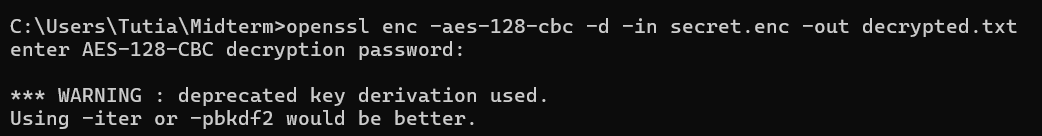
1. Save the encrypted file as secret.enc.



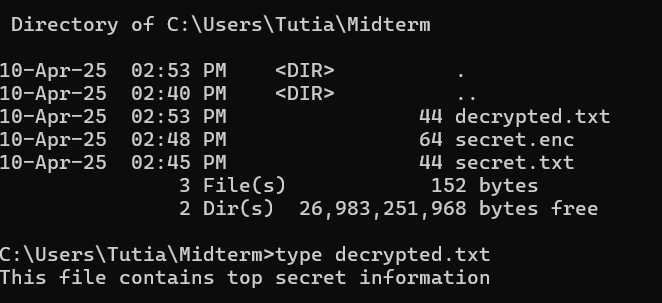
1. Document the commands used.**Commands and results are shown in the screenshots.**

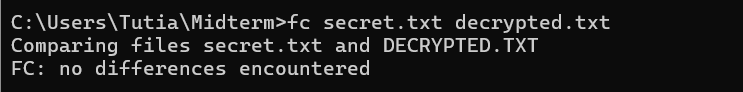
**Task 1B: Decrypt secret.enc (2.5 pts)**

1. Decrypt the file.



1. Show that it matches the original.





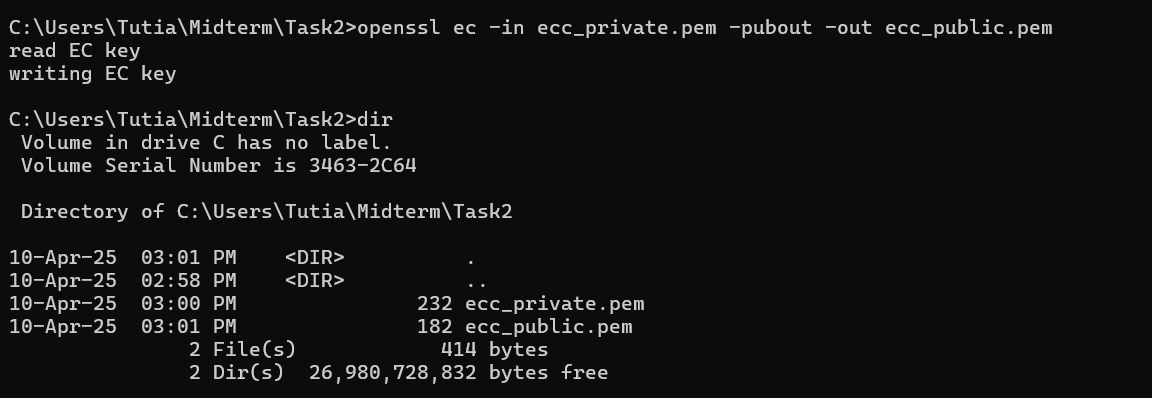
**All the listed files above is uploaded on Github Task1**

**Task 2A: Generate ECC keys (1.5 pts)**

1. Use the prime256v1 curve.



2. Save private and public keys.

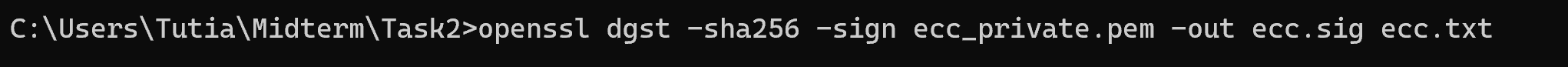


**Task 2B: Sign and verify a message (3 pts)**

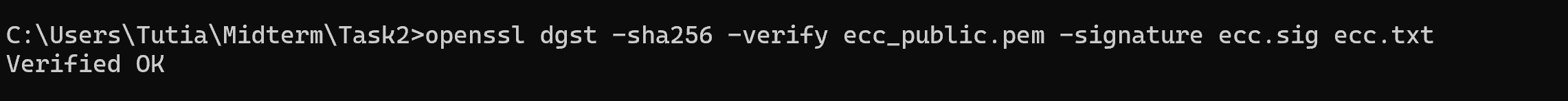
1. Create ecc.txt with Elliptic Curves are efficient.



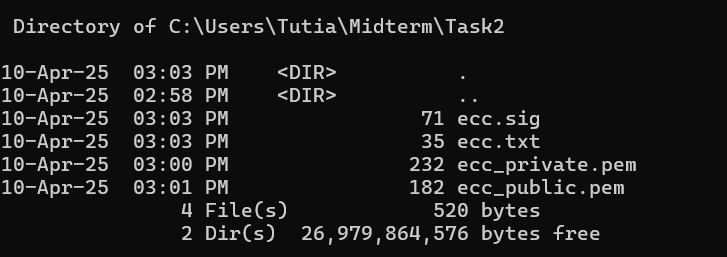
1. Sign it with your private key.



1. Verify it using your public key.



**All the listed files below is uploaded on Github Task2**



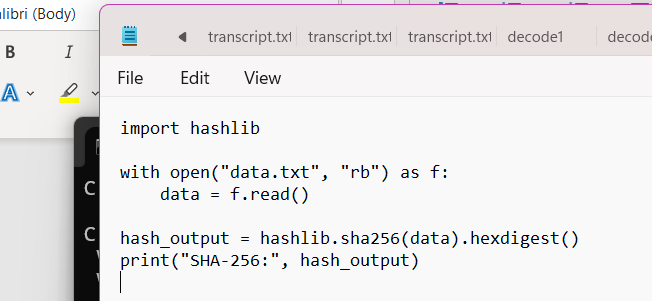
**Task 3A: SHA-256 Hash (2 pts)**

1. Create data.txt with: Never trust, always verify.



1. Hash it using Python or CLI.







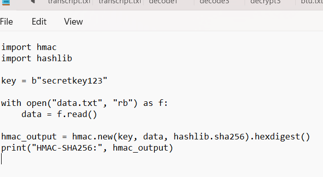
1. Submit hash output and code/command.



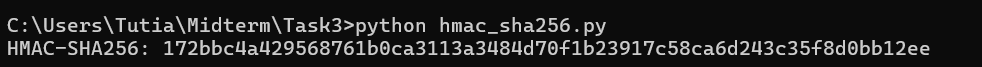
**Task 3B: HMAC using SHA-256 (2 pts)**

1. Use the key: secretkey123



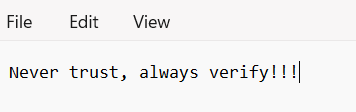


1. Create an HMAC for data.txt.

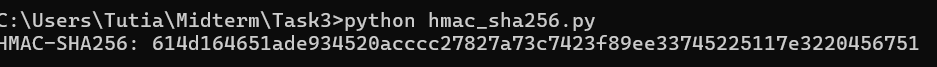


**Task 3C: Integrity Check (2 pts)**

1. Change one letter in data.txt.

 **added !!!**

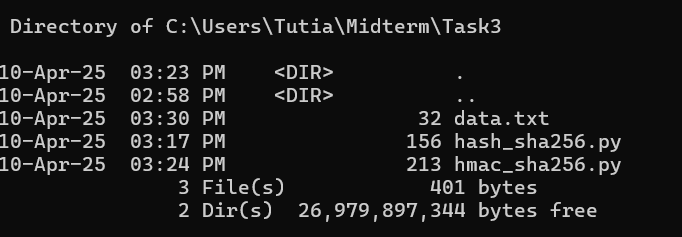
1. Recompute HMAC.



1. Explain what happens and why HMAC is important.

**Changing one character in the file results in a completely different HMAC. This shows that any modification and is crucial for verifying the integrity and authenticity of data.**

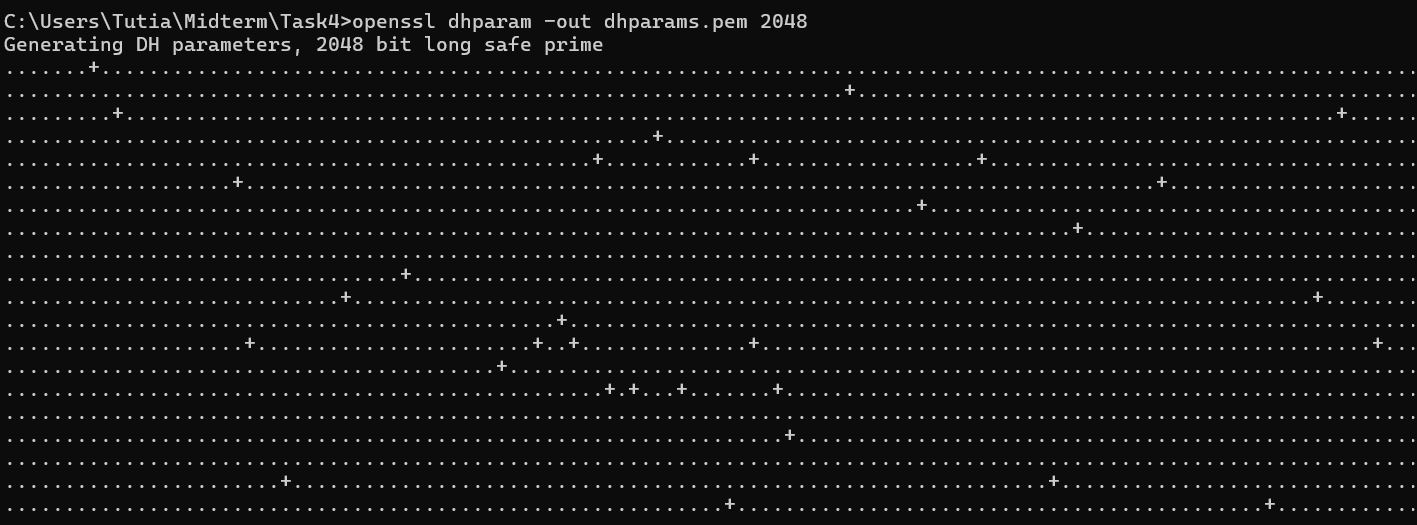
**All listed documents is uploaded on Github Task 3**

****

**Task 4A: Simulate DH Key Exchange (2 pts)**

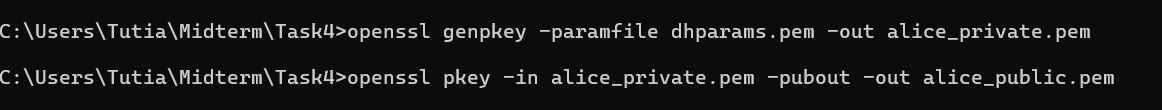
1. Use Python or OpenSSL CLI to simulate the Diffie-Hellman key exchange between

Alice and Bob.

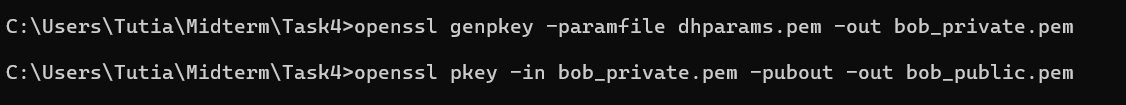


2. Show the following:

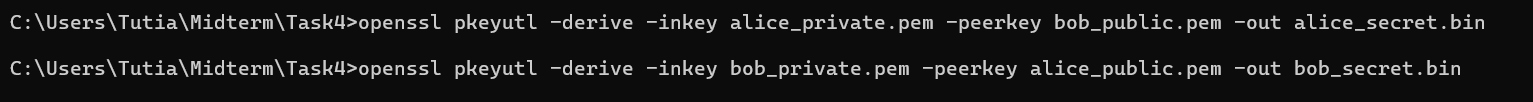
○ Alice’s public key



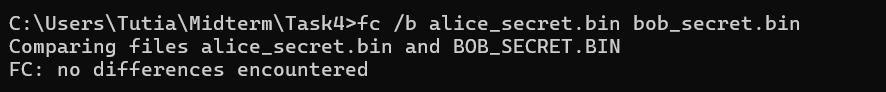
○ Bob’s public key



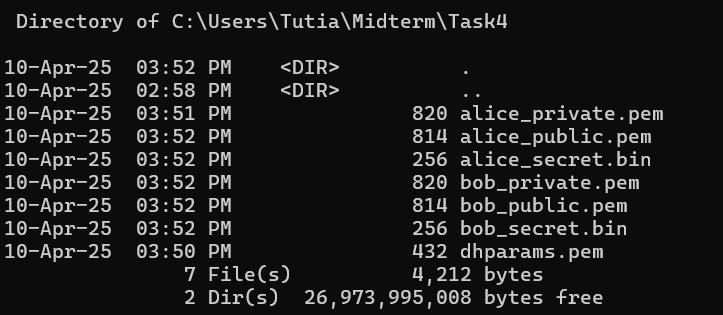
○ Shared secret key derived by both



1. Ensure that both shared keys are identical.



All documents will be uploaded to Github Task4.



**Task 4B: Real-Life Application (2.5 pts)**

1. In a short paragraph, explain where Diffie-Hellman is used in practice, such as:

○ TLS handshake

○ Secure messaging (e.g., Signal Protocol)

**Diffie-Hellman key exchange is commonly used in protocols like the TLS handshake to securely generate encryption keys for HTTPS connections. It’s also implemented in secure messaging apps such as Signal, which use variants like X3DH and the Double Ratchet algorithm to establish and maintain encrypted communication between users without sharing private keys directly.**

2. Mention why it’s important for secure communication.

**Diffie-Hellman is important because it enables two parties to create a shared secret over an untrusted network without transmitting the secret itself. This prevents eavesdroppers from accessing the encryption key, ensuring confidentiality, integrity, and forward secrecy in secure communications.**