Task-3: Corrupting and Decrypting ECB and OFB Ciphertexts

The goal of this task was to understand how bit corruption affects encrypted data under two different AES modes ECB and OFB. At first I created a plaintext file and I performed this by intentionally corrupting a specific byte in each ciphertext file and then decrypting both files to observe the results.

Method:

1. **Corrupting the Ciphertexts:**
   i. I first wrote a PowerShell script to corrupt the 30th byte of each ciphertext file (`enc_ecb.bin` and `enc_ofb.bin`).
   ii. The script read the original binary data, flipped one bit of the 30th byte using the XOR operation, and saved the modified version as a new file (`enc_ecb_corrupted.bin` and `enc_ofb_corrupted.bin`).
   iii. Both the original and corrupted files were checked using MD5 hashes to confirm that they were indeed different.

The same process was repeated for the OFB ciphertext.

2. **Decrypting the Files:**
   I used the OpenSSL command-line tool to decrypt both the original and corrupted files.
   The decryption commands were as follows:

   For ECB mode:

   ```
   openssl enc -d -aes-128-ecb -in
   "C:\Users\USER\Desktop\Lab\Task3\enc_ecb_corrupted.bin" `
   -out "C:\Users\USER\Desktop\Lab\Task3\dec_ecb.txt" `
   -K "00112233445566778899AABBCCDDEEFF"
   ```

   For OFB mode:

   ```
   openssl enc -d -aes-128-ofb -in
   "C:\Users\USER\Desktop\Lab\Task3\enc_ofb_corrupted.bin" `
   -out "C:\Users\USER\Desktop\Lab\Task3\dec_ofb.txt" `
   -K "00112233445566778899AABBCCDDEEFF" `
   -iv "0102030405060708090A0B0C0D0E0F10"
   ```

   The decrypted outputs were saved as text files so I could easily compare the readable contents.

Observation:

1. In ECB mode, altering a single byte in the ciphertext caused corruption in an entire 16-byte block of the decrypted text. This confirmed that ECB encrypts each block independently, so a single bit error affects the whole block but not others.

2. In OFB mode, only the specific byte corresponding to the flipped bit was corrupted in the decrypted output. The rest of the text remained intact. This demonstrated that OFB behaves like a stream cipher. Each bit of ciphertext depends only on the keystream and not on previous blocks.