Lab 3 - Task-2: AES ECB vs CBC on BMP Image

Objective

Encrypt a bitmap image using AES-128 in two modes (ECB and CBC), rebuild valid BMP files by keeping the first 54-byte header unchanged, and compare the visual results.

Theory

ECB (Electronic Codebook): encrypts each 16-byte block independently. Identical plaintext blocks produce identical ciphertext blocks, so image patterns can remain visible.

CBC (Cipher Block Chaining): XORs each plaintext block with the previous ciphertext block and uses an IV for the first block. This destroys repeating patterns and hides image structure.

Requirements

- Ubuntu/Linux terminal
- OpenSSL installed
- Image viewer (eog) to view BMP files
- Optional: xxd or GHex to inspect bytes

Procedure

```
1) Prepare a BMP image named pic_original.bmp (24-bit). Example using ImageMagick:
   convert -size 200x200 xc:red pic_original.bmp

2) Extract payload (skip 54-byte BMP header):
   dd if=pic_original.bmp of=payload.bin bs=1 skip=54

3) Set key and IV in hex (demo values):
   KEY=00112233445566778899aabbccddeeff
   IV=0102030405060708090a0b0c0d0e0f10

4) Ensure payload length is a multiple of 16 (AES block size):
   size=$(stat -c%s payload.bin); rem=$((size % 16))
   head -c $((size - rem)) payload.bin > payload_head.bin
   tail -c $rem payload.bin > payload_tail.bin

5) Encrypt in ECB and CBC (no padding on the aligned part):
   openssl enc -aes-128-ecb -e -in payload_head.bin -out tmp_ecb.bin -K $KEY -nosalt -nopad
   cat tmp_ecb.bin payload_tail.bin > payload_ecb.full

    openssl enc -aes-128-cbc -e -in payload_head.bin -out tmp_cbc.bin -K $KEY -iv $IV -nosalt
-nopad
   cat tmp_cbc.bin payload_tail.bin > payload_cbc.full

6) Rebuild BMP files by keeping original header and replacing only pixel payload:
   cp pic_original.bmp enc_ecb.bmp
```

Lab 3 - Task-2: AES ECB vs CBC on BMP Image

```
   dd if=payload_ecb.full of=enc_ecb.bmp bs=1 seek=54 conv=notrunc

   cp pic_original.bmp enc_cbc.bmp
   dd if=payload_cbc.full of=enc_cbc.bmp bs=1 seek=54 conv=notrunc

7) View results:
   eog enc_ecb.bmp &
   eog enc_cbc.bmp &
```

Observations

- enc_ecb.bmp: visible outlines or block-like patterns remain. This is expected in ECB.
- enc_cbc.bmp: appears noisy and random. This is expected in CBC.
- Headers are unchanged (first 54 bytes), only the pixel data is encrypted.

Verification

```
Header check (should be identical for first 54 bytes):
cmp -n 54 pic_original.bmp enc_ecb.bmp && echo Header identical
cmp -n 54 pic_original.bmp enc_cbc.bmp && echo Header identical
```

Result

- ECB mode leaks structure; patterns remain visible in the encrypted image.
- CBC mode hides structure; the encrypted image looks random.
- Both files are valid BMPs because the header is preserved.

Conclusion

ECB is not suitable for data with visible patterns, such as images. CBC eliminates visible patterns by chaining blocks with an IV, providing better confidentiality.

Files Produced

1) pic_original.bmp (original image)
2) payload.bin (pixel data only)
3) enc_ecb.bmp (AES-128-ECB encrypted image)
4) enc_cbc.bmp (AES-128-CBC encrypted image)

Screenshot Instruction

Attach screenshots of enc_ecb.bmp (pattern visible) and enc_cbc.bmp (random noise) side by side.