Kai Kovacik
V00880027
Sept. 19, 2018

LAB 1

1. Operating modes determine how multiple blocks of ciphertext will be “stitched” together. ECB simply concatenates block after block of ciphertext. CBC, on the other hand, uses the last encrypted block to blockchain the encryption, this helps avoid repetition in the ciphertext even when encrypting repeating plaintext. Since CBC handles repetition and ECB does not, If you used the same encryption algorithm and key to encrypt the same plaintext with both CBC and ECB, the resulting ciphertext would be different.
2. ECB is fundamentally flawed when it comes to encrypting images. Because ECB has no mechanism for handling repetition (x always encrypts to y), images, which almost always repeat colours and patterns will end up maintaining their general-legibility even after encryption. Since CBC handles repetition in the input subject, the removal of any patterns results in an image that is completely illegible.
3. Padding is used to ensure that any amount of input data (to be encrypted) can be divided into equal blocks for encryption. There are many different padding conventions, however, since both ECB and CBC are for block ciphers, they both work with the same padding schemes.
4. Since padding is used specifically so that a cipher can break input data into equal parts and both CBC and CFB are for block ciphers, they work with the same padding schemes.
5. If a single bit were corrupted in a ciphertext before decryption,
    a. Since ECB works block by block, only the affected block would be illegible. The rest would decrypt fine.
    b. Since CBC decryption works by decrypting the current block of ciphertext then XOR operating on the result using the ciphertext of the previous block, the block that is affected would be illegible and the adjacent block that uses the previous, corrupted one, would contain exactly one corrupted bit. Everything else would be legible.
    c. Since CFB decryption works by XOR operating the current ciphertext block with the previous block encrypted again, the block of ciphertext containing the corrupted bit would decrypt into plaintext with exactly one corrupt bit. Every following block in the blockchain would be completely illegible.
    d. Since OFB decryption works by XOR operating the current ciphertext block with an incrementally encrypted IV (the first block encrypts once, second encrypts again, third again… etc.), the block of ciphertext with the corrupt bit, would decrypt into a block of plaintext with exactly one corrupt bit. Everything else would be legible.
6. Since CFB uses a XOR operation with the IV and the plaintext prior to encryption, a XOR operation needs to be executed on the decrypted text using the known IV. This (if the decryption was correct) will yield the correct plaintext. It could be argued that because of this, CFB would take more computing power and is, therefore, more secure, but in practice, XOR operations are very fast and as so, both CFB and ECB are of the same security for sub-1-block encryptions.

7. Since modes that require an IV, chain previous encryption work to further scramble the current block, the first block needs some additional initial data (an initialization vector). Without an IV, the encryption is likely to contain repetition, which is naturally insecure, likewise, if an IV doesn't change it could be evaluated by an attacker, hence, voiding the additional security that chaining modes provide. If you kept your IV the same and changed the key every time you encrypted something, although that would indeed be very secure, it would be logistically unsound. This would be like changing your front door lock every day. Anybody who wanted to get in would have to frequently change their key. Additionally, you would need some way to memorize which key was used to encrypt a certain saved message, which would get more and more impossible as you were to encrypt more and more.
8. The attack conducted in Lab 1 was successful because the OFB algorithm being used had the same initialization vector for both separately-encrypted messages. Since OFB allows you to produce the IV ciphertext by XOR operating the plaintext and ciphertext of a message and since the IV was the same for both messages, that IV ciphertext for the first message could be used to decrypt the second message. An easy way to avoid this kind of KPA, is to **change the IV**.