Network Security and Cryptography
Symmetric-key cryptography

Lecture 4: Block cipher modes

Mark Ryan

Block cipher modes

Block ciphers (like DES and AES) can be used directly only for a single block of plaintext.

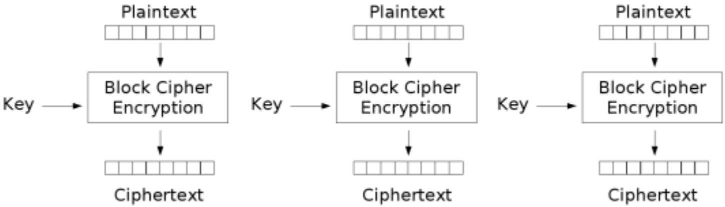
However, typically we want to encrypt plaintext that is much longer than a single block.

How can we use a block cipher to do that securely?

ECB

Simplest way: Apply the encryption block by block

This is called *Electronic Codebook mode*, ECB.

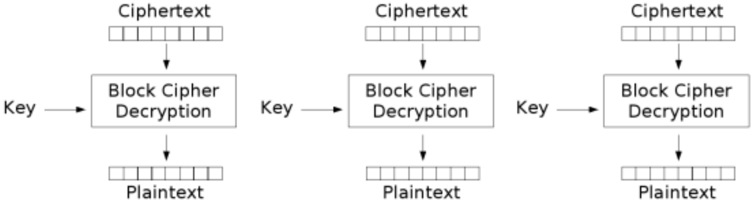


Electronic Codebook (ECB) mode encryption

Source: Wikipedia

ECB decryption

Decryption just does the operations in reverse, and uses the decrypt function of the block cipher.



Electronic Codebook (ECB) mode decryption

Source: Wikipedia

ECB is not secure

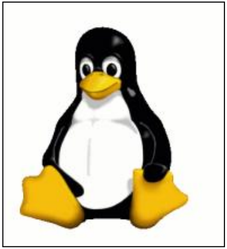
Example 1. Suppose Alice sends Bob and Charlie the same message. The adversary sees the same ciphertext going to each of them. The adversary doesn't know what it says, but s/he knows Alice send the same message to each of them.

Example 2. Suppose Alice sends Bob a long message (say 1000 blocks; with 64bit blocks that would be about 8K of text). Then the next day sends him the same message except it's different from block 900 onwards. Then with ECB the ciphertext is the same from block 900 onwards as well. The attacker doesn't know what the messages say, but knows they are identical except for the last 10 percent.

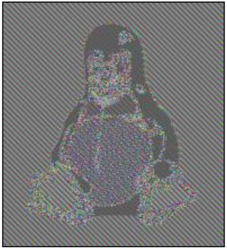
So some information has been leaked each time.

Visual example

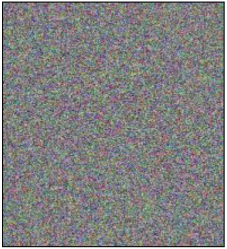
Suppose we encrypt a photo, stored in “raw” or raster format (i.e. bytes giving the RGB values for each pixel in turn).



Plain text



Ciphertext using ECB



Ciphertext using a secure mode

Source: Wikipedia

A good block mode should have the properties:

1. [Security (confidentiality)] Identical plaintexts shouldn't produce identical ciphertexts; and
   - Identical blocks within a plaintext shouldn't produce identical ciphertext blocks

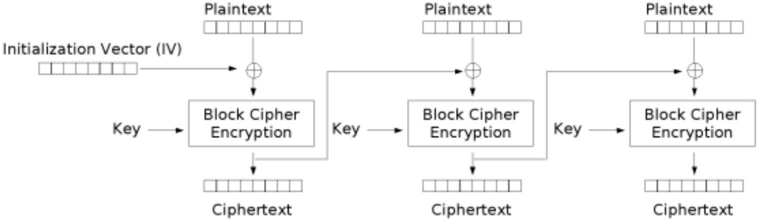
Other properties that might be nice too:

2. [Security (integrity)] There should be protection against deletion or insertion of blocks
3. [Recovery] Ciphertext transmission errors should affect only the the block containing the error
4. [Efficiency] It should be efficient (e.g., parallelisable)

ECB fails properties 1 and 2. It satisfies 3 and 4, but they are not as important as 1 and 2.

CBC

Cipher block chaining adds a random “initialisation vector” (IV) to randomise the first block, and then uses each block to randomise the next block.



Cipher Block Chaining (CBC) mode encryption

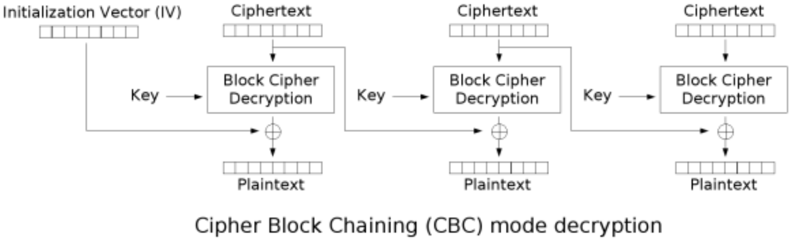
Source: Wikipedia

The IV should be randomly chosen for each encryption. Note that you must store the IV with the ciphertext, otherwise it's not possible to decrypt. Thus, the IV is *random*, but not *secret*.

Note that, since the ciphertext includes the IV, it is one block longer than the plaintext. So we have a small expansion in size.

Figure out which of properties 1, 2, 3, 4 hold for CBC.

CBC decryption:

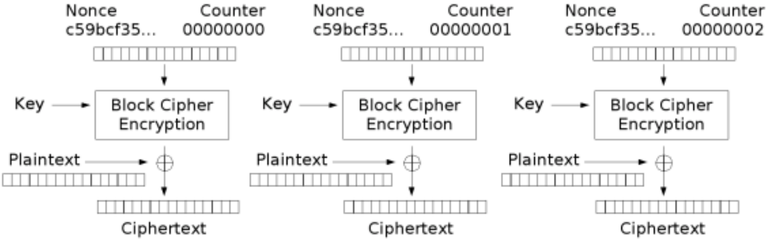


Source: Wikipedia

Counter mode (CTR)

In counter mode, we don't chain the blocks together, but still we aim to make sure that identical plaintext blocks have different ciphertext blocks.

To encrypt a message, we chose a random nonce, and then set up a counter which is incremented for each block.



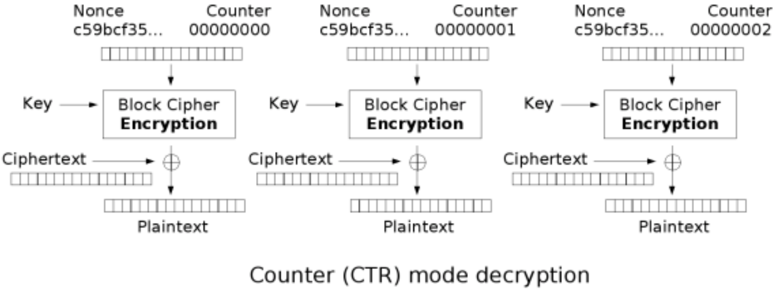
Counter (CTR) mode encryption

Source: Wikipedia

The nonce must be stored with or transmitted with the ciphertext blocks. Thus, similarly to the IV in CBC, in CTR mode the nonce must be chosen at random, but is not secret. Like CBC, CTR also increases the size of the ciphtertext by one block.

Figure out which of properties 1, 2, 3, 4 hold for CTR.

CTR mode decryption



Source: Wikipedia

Block modes: summary

1. ECB is not secure and should never be used.
2. CBC is secure as far as confidentiality is concerned. However, its chaining method means you can't parallelise it.
3. CTR is secure as far as confidentiality is concerned, and can be parallised. Interestingly, it only uses the block cipher (e.g., DES or AES) in encryption mode, even for decryptions.