

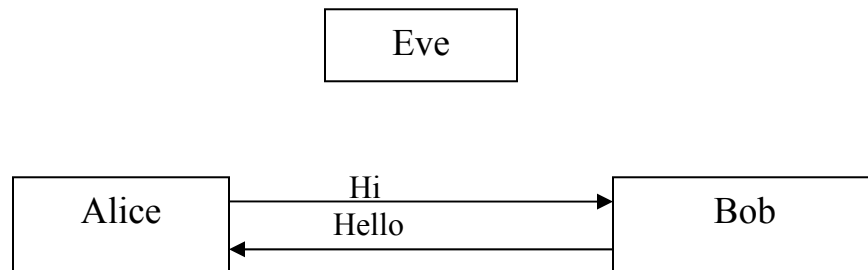
Lecture Notes for 02/16/06

SYMMETRIC KEY CRYPTOGRAPHY

(Green part is taken from: http://en.wikipedia.org/wiki/Data_Encryption_Standard)

Symmetric Key Cryptography:

Typically there are two parties involved in an encryption system. For example, let us have the following scenario of two people sending messages amongst themselves:



Assume that Alice and Bob want to send messages without a third party knowing. Assume that Eve is listening and she wants to read the messages that go between Alice and Bob and if she succeeds she would be violating the *confidentiality* goal of security.

Why Encryption?

So, what is the main reason for having an encryption system? It is to preserve confidentiality. If the messages that are sent between Alice and Bob are in clear text, then Eve can easily read them. Hence, for security, the messages sent should be encrypted.

Intuition:

Encryption should give us the same secrecy as sealed envelopes. If Bob places a letter in an envelope and he seals the envelope and sends it to Alice, then, Eve would not be able to read the letter in the sealed envelope.

It should be noted that encryption does not hide communicating but it hides what is being communicated. Hence, with our example, if encryption is implemented, Eve might know that a message is being sent to Alice, however, she would not be able to read it. The purpose of encryption is to be able to send messages secretly, even if a third party knows that a message is being sent and he knows some side info about the people who are sending messages.

Types of Symmetric Key Cryptography:

1) One-Time Pad:

One time pad has the following components and parameters:

- M = Message of length “ l ”
- K = Key of length “ l ”
- For the Key, each bit = 0 with probability = $\frac{1}{2}$. So, in other words each bit for the key is chosen randomly.

How can we encrypt?

- $C = M \oplus K$.
- C stands for Cipher Text.
- K is a string of bits.
- The Cipher Text can be obtained by performing the “XOR” operation between M and K .

Applying this to our previous example, we see that for preserving secrecy, both Alice and Bob must know K for the encrypted message, but, Eve must not know K . In order to come up with the Key, both Alice and Bob have to think of and agree on a set of bits that are arranged randomly. So, if Bob wants to say Hi, he would change “Hi” to ASCII and then XOR the bits with the Key bits and then send the cipher text.

An important thing is to note that in this system, each bit of key is used only once. This brings us to the following theorem:

Theorem:

A One-Time Pad is secure against a computationally unbound third party. So, no matter how long the third party can sit computing, he/she cannot read the message.

One-Time Pad Is Not Practical. Why?

- Key must be as huge as message.
- Cannot reuse bits of the Key.

What Happens If We Reuse The Key Bits?

Suppose we send two messages M_1 and M_2 , such that:

$$C_1 = M_1 \oplus K \text{ and}$$

$$C_2 = M_2 \oplus K.$$

What would happen?
Let us try computing the following:

$$\begin{aligned} C1 \oplus C2 \\ &= M1 \oplus K \oplus M2 \oplus K \\ &= M1 \oplus M2. \end{aligned}$$

So, now if Eve knows some side information, for example, that M1 and M2 are in English, then she can recover M1 and M2.

Hence in a One-Time encryption system, the Key can **NEVER** be reused.

2) BLOCK CIPHER:

A block cipher has the following components and parameters:

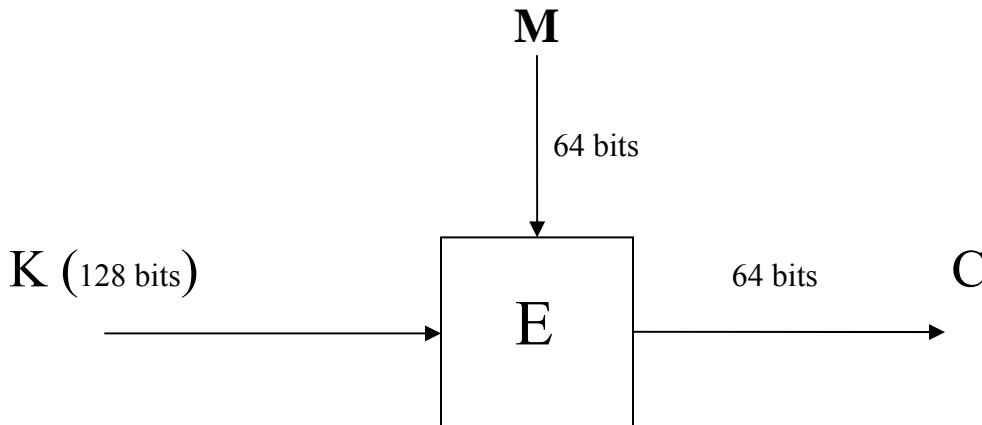
- A function: $E(K, M) = C$.
(A function that takes as its parameters a Message 'M', and a Key 'K' and gives back a Cipher Text, 'C'.)
- The function should be such that given 'K', it should be easy to compute $E(K, M)$.
- Moreover, given the Key and Cipher Text, it should be easy to compute the Message. So symbolically, the following should be easy to compute:

$$M = E^{-1}(K, C).$$

- Block Ciphers work on blocks.
- The function E takes a k-bit Key and n-bit Message and gives back an n-bit cipher text. So, symbolically,

$$E: \{0,1\}^k \times \{0,1\}^n \rightarrow \{0,1\}^n$$

The following is a simple figure that summarizes the encryption method:



The above figure shows a cipher block encryption that takes a 64 bit Message and a 128 bit key, and gives a 64 bit cipher text.

The following are examples of block ciphers:

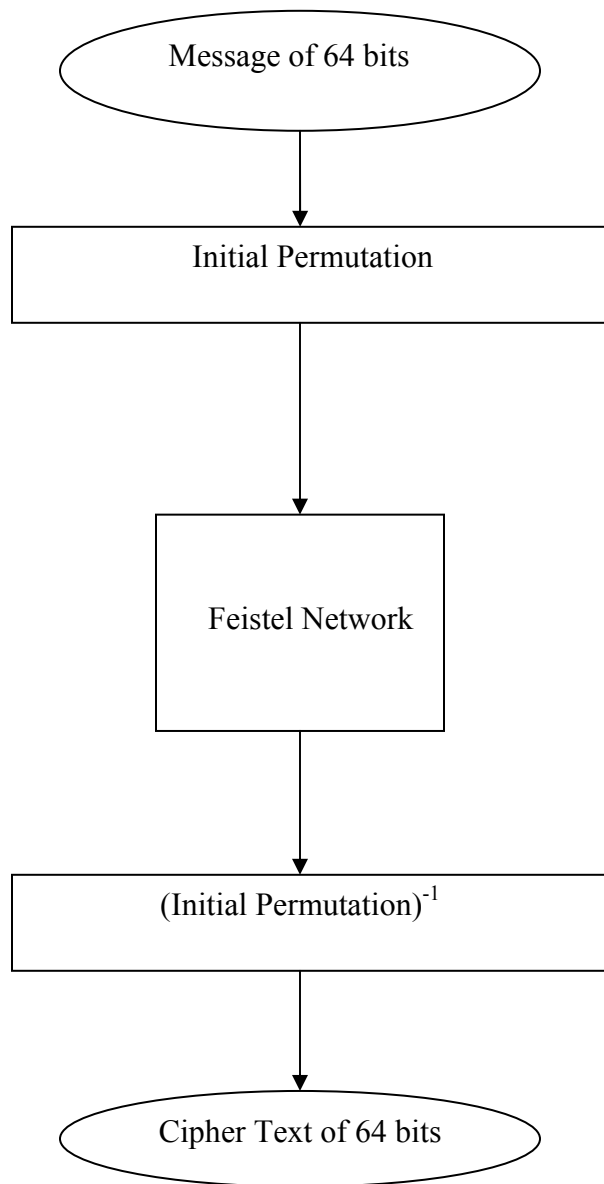
3) DES (Data Encryption Standard):

DES was invented by IBM and NSA in the 70's.

The following is a simple description of DES:

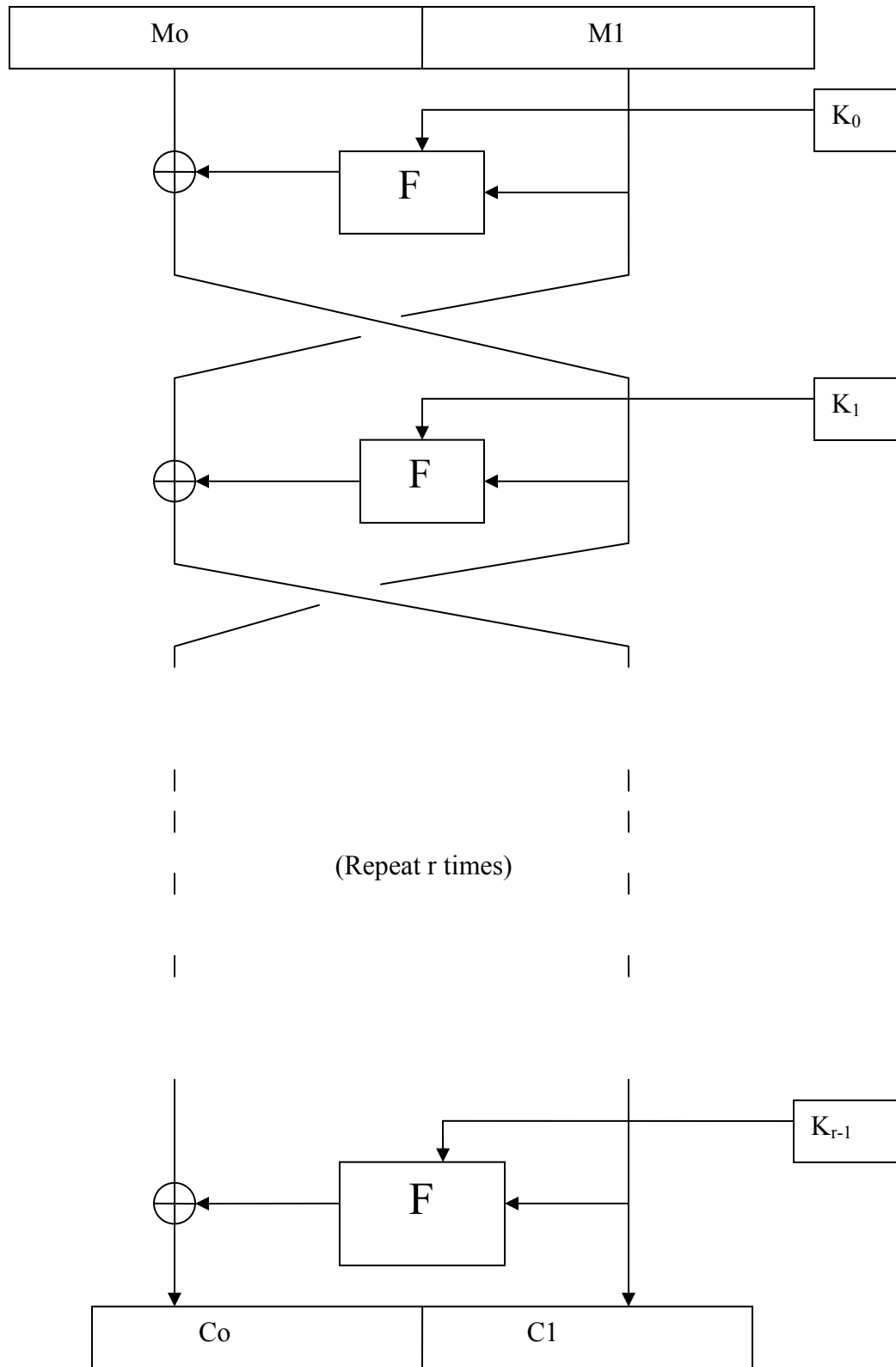
- The Key should be 56 bits.
- Block size should be 64 bits.
- DES is an example of a Feistel Network.
- Given the Key and Message it is easy to compute the cipher text. Moreover, the inverse is also true, i.e., given a cipher text and the key it is easy to compute the message. However, for people who don't know the key, it is hard for them to get the message from the cipher text, or vice versa.
- The message undergoes an initial permutation before entering the Feistel Network.
- The Cipher Text that is computed by the Feistel Network undergoes a final permutation which is the inverse of the initial permutation.

The following is a figure that shows how DES is structured and how it operates:



Now, in the Feistel Network, the main 64-bit message is split in two 32-bit blocks and they are processed, (in a cross-crossing manner,) with the help of Feistel, or “F”, Functions. The *F-function* scrambles half a block together with some of the key. The output from the F-function is then combined with the other half of the block by using the XOR function, and the halves are swapped before the next round.

The following is a picture of what happens inside a Feistel Network:



Question) Did the permutations before and after the Feistel Network make the breaking of the code any harder?

Answer) No, These permutations are pointless in terms of making the encrypted message harder to break. The permutation is not secret, and hence, in terms of security it is a waste of time.

Successful Attacks On DES:

- DES is vulnerable to an attack called “Differential Cryptanalysis.”
- The number of bits required for the key is known. It is 56. So, we need to try 2^{56} different keys to break the system.
- With a super computer this can be done within a day’s limit.
- EFF built a DES cracker that takes less than 24 hours. (Just need one cipher text to break the system.)

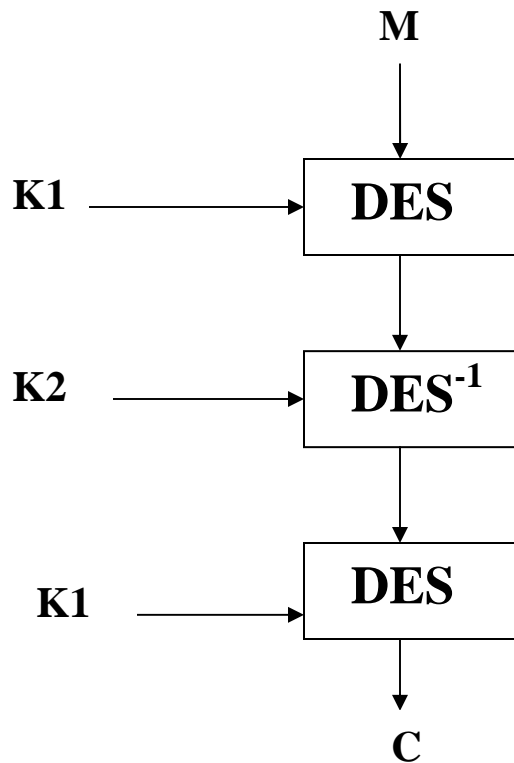
Conclusion:

DONOT use DES as an encryption system!

Most Block Ciphers are not completely safe nowadays. In order to extend the life of a DES people came up with the following slightly better system:

4) 3DES (Triple Data Encryption Standard):

This system uses more key bits. The following is a picture that illustrates how 3DES works. We have a 112 bit key for the following system:



(If we would have had K1 instead of K2 then we would have had a DES system)

Possible Attacks on This System:

This system is vulnerable to plain-text and chosen-text attacks. Hence, this is not that great either.

Moreover, this system also proved to be very slow.

During the 90's a new block cipher known as AES was chosen as an encryption standard by the u.s. government.

5) Advanced Encryption Standard (AES):

- Another name for AES is Rijndael.
- This is a special design that uses Algebra, and it is fast and secure.

AES has the following parameters:

- The key modes are : 128-bit key mode, 192-bit key mode and 256-bit key mode.
- The block size is: 128 bits.

So the weakest version of an AES system would have a 128 bit key. Even this is very hard to break and would take probably thousands of years. Hence, AES is a fine cipher.

Now, all these ciphers described above are block ciphers. What should we do if we have to encrypt more than 1 block? In order to encrypt more than 1 block we have the following Modes of Operation:

Modes of Operation:

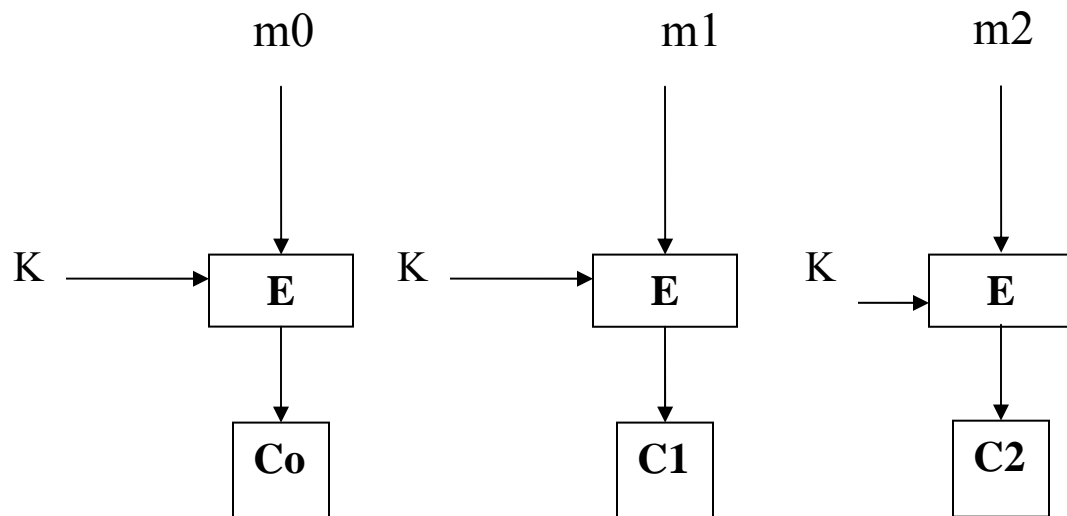
There are three modes of operation:

- ECB (Electronic Code Book.)
- CTR (Counter Mode.)
- CBC (Cipher Block Sharing.)

1) ECB:

- Also known as Electronic Code Book.
- It is like a dictionary. To encrypt you look up a word and see what it is translated to. To decrypt you go oppositely, i.e., you look up the cipher text and see what message it corresponds to.
- NOT secure.
- Does not work like an opaque envelope.
- Same message → same cipher text.
- Needs randomization.
- Not good to use.

The following is an illustration of ECB:

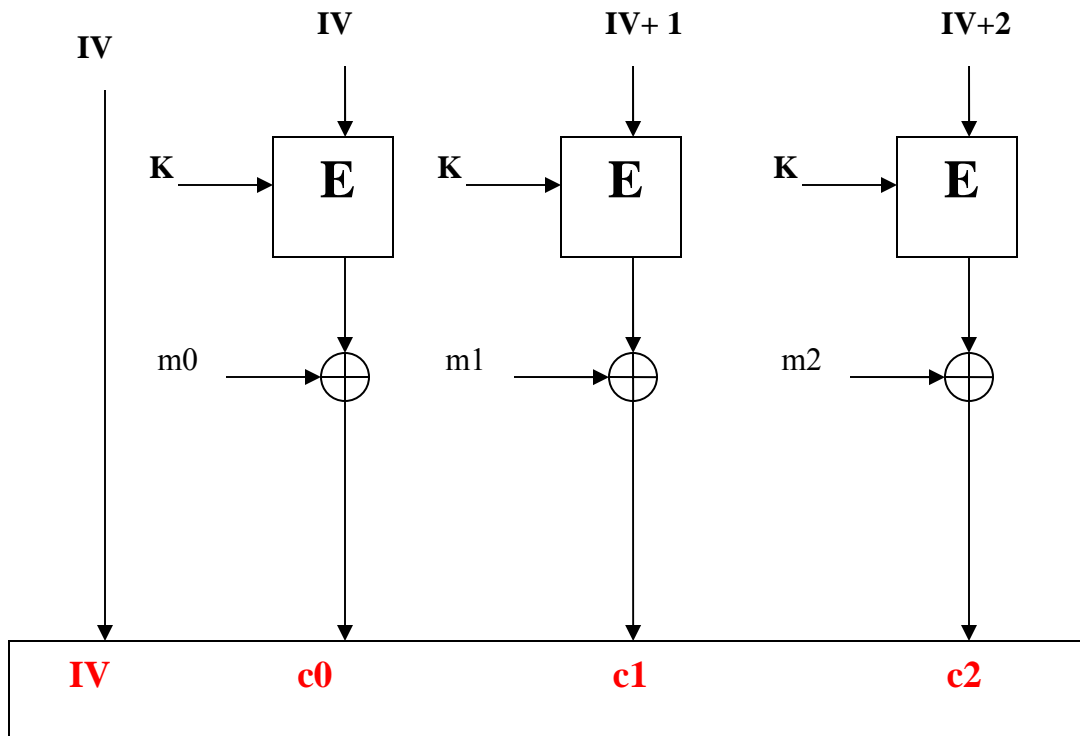


(The above is an illustration of ECB)

2) CTR:

- Also known as counter mode.
- Secure against chosen plain text attacks
- This system has an Initialization Vector “IV,” which is a number that is chosen randomly for each cipher text.
- If you use same IV twice then that destroys security, hence, the space of IV should be very large.
- CTR is similar to One-Time Padding.

Below is a picture that illustrates the CTR Mode of Operation:

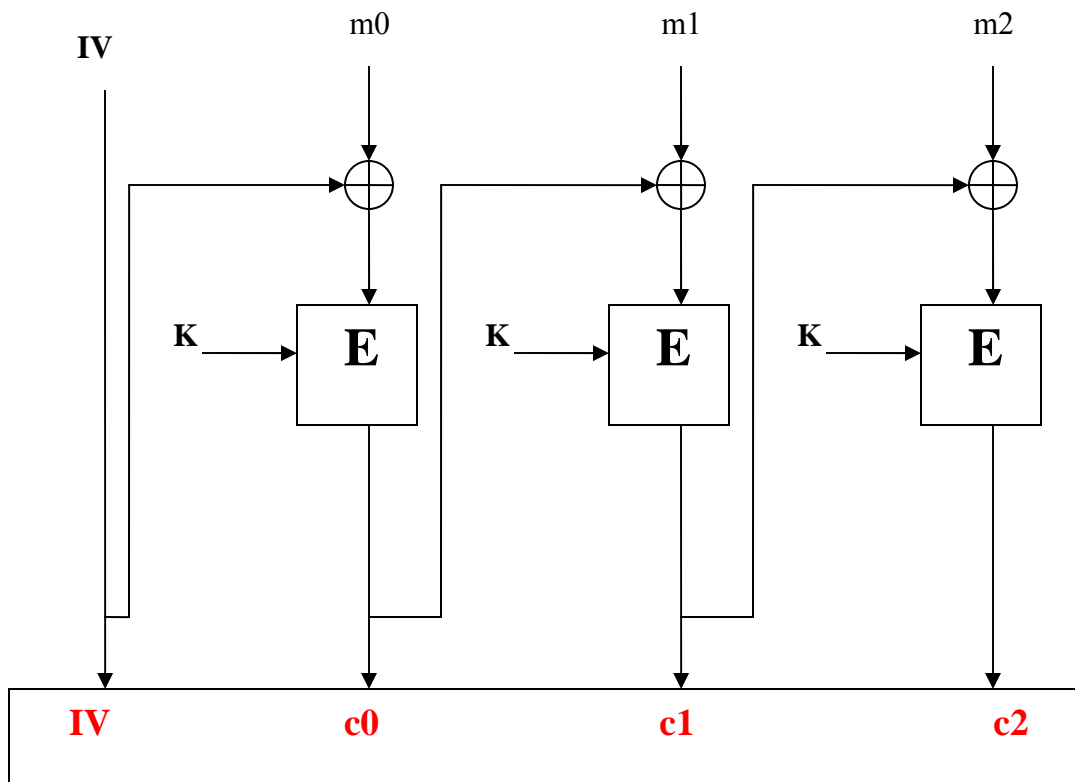


(The elements in red show the output that would be sent)

CBC:

- Abbreviation for Cipher Block Chaining.
- In this mode, each cipher block is chained with the next one.
- Secure against chosen plain text attacks.
- Most Common.

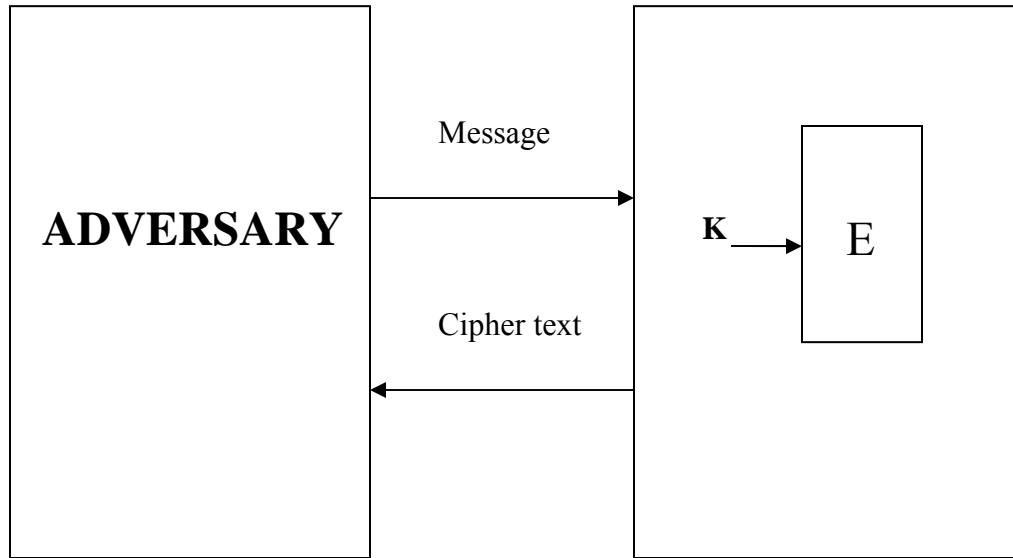
Below is a picture that illustrates the CBC Mode of Operation:



Attack Models Against Cryptosystems:

- Passive Adversary:
 - Weakest form of attack.
 - Observe Cipher texts and
 - Try to infer the plain texts or possibly the key.
- Known Plain Text Attack:
 - Attacker knows the plain text that is given to the cryptosystem, and knows the cipher text.
 - Tries to decrypt future messages.
 - Attacker cannot give inputs to the cryptosystem.
- Chosen Plain Text Attack:
 - Adversary can request cipher text corresponding to any plain text.
 - Then adversary tries to decrypt any future messages.
 - So, it can be thought of as, the adversary is given access to a black box, and adversary is allowed to send queries to the black box and get cipher texts as output from the black box.

The following is an illustration of this case:



(Chosen Plain Text Attack)

- Chosen Cipher Text Attack:
 - This is the most powerful mode of attack.
 - Adversary can request encryption/decryption of any plain text/cipher text.
 - Then adversary tries to decrypt any future messages.

