Chapter 6

More about Block Ciphers

Further Reading:

Section 8.1 in [Sch93].

Note:

The following modes are applicable to all block ciphers $e_k(X)$.

6.1 Modes of Operation

6.1.1 Electronic Codebook Mode (ECB)

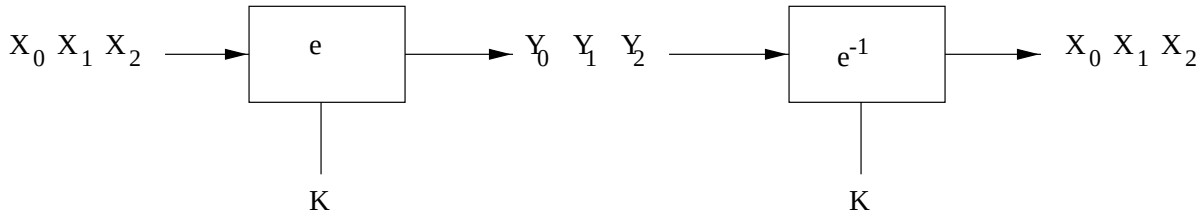

Figure 6.1: ECB model

General Description:

$e_k^{-1}(Y_i) = e_k^{-1}(e_k(X_i)) = X_i$; where the encryption can, for instance, be DES.

Problem:

This mode is susceptible to substitution attack because same X_i are mapped to same Y_i.

Example: Bank transfer.

Block #	1	2	3	4	5
	Sending Bank A	Sending Account #	Receiving Bank B	Receiving Account #	Amount $

Figure 6.2: ECB example

1. Tap encrypted line to bank B.
2. Send $1.00 transfer to own account at bank B repeatedly → block 4 can be identified and recorded.
3. Replace in all messages to bank B block 4.
4. Withdraw money and fly to Paraguay.

Note: This attack is possible only for single-block transmission.

6.1.2 Cipher Block Chaining Mode (CBC)

Beginning: $Y_0 = e_k(X_0 \oplus IV)$.

$$X_0 = IV \oplus e_k^{-1}(Y_0) = IV \oplus e_k^{-1}(e_k(X_0 \oplus IV)) = X_0.$$

Encryption: $Y_i = e_k(X_i \oplus Y_{i-1})$.

Decryption: $X_i = e_k^{-1}(Y_i) \oplus Y_{i-1}$.

Question: How does it work?

$X_i = e_k^{-1}(e_k(X_i \oplus Y_{i-1})) \oplus Y_{i-1}$.

$X_i = (X_i \oplus Y_{i-1}) \oplus Y_{i-1}$.

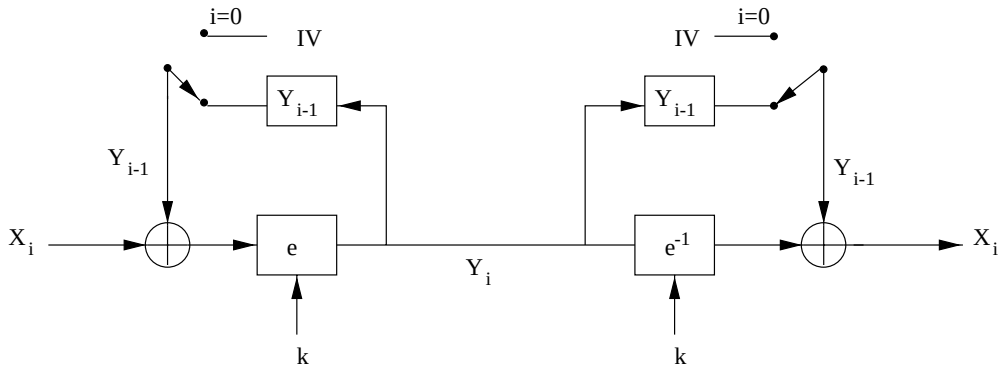

Figure 6.3: CBC model

$X_i = X_i$. *q.e.d.*

Remark: The Initial Vector (IV) can be transmitted initially in cleartext.

6.1.3 Cipher Feedback Mode (CFB)

Assumption: block cipher with b bits block width and message with block width l, $1 \leq l \leq b$.

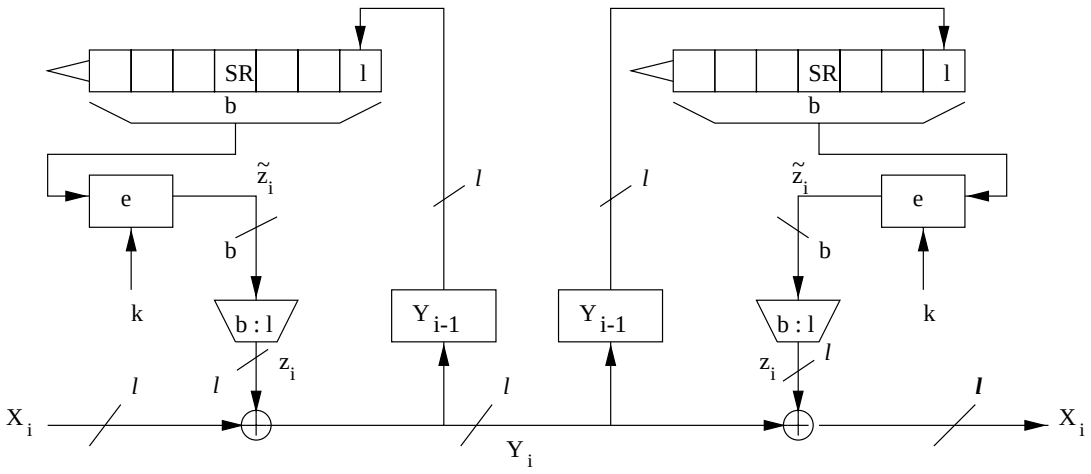

Figure 6.4: CFB model

Procedure:

1. Load shift register with initial value IV.
2. Encrypt $e_k(IV) = \tilde{z}_0$.
3. Take l leftmost bits: $\tilde{z}_0 \rightarrow z_0$.
4. Encrypt data: $Y_0 = X_0 \oplus z_0$.
5. Shift the shift register and load Y_0 into the rightmost SR position.
6. Go back to (2) substituting $e(IV)$ with $e(SR)$.

6.1.4 Counter Mode

Notes:

- Another mode which uses a block cipher as a pseudo-random generator.
- Counter Mode does not rely on previous ciphertext for encrypting the next block.
  $\Rightarrow$ well suited for parallel (hardware) implementation, with several encryption blocks working in parallel.
- Counter Mode stems from the Security Group of the ATM Forum, where high data rates required parallelization of the encryption process.

Description of Counter Mode:

1. An n-bit initial vector (IV) is loaded into a (maximum length) LFSR. The IV can be publically known, although a secret IV (i.e., the IV is considered part of the private key) turns the counter mode systems into a non-deterministic cipher which makes cryptoanalysis harder.
2. Encrypt block cipher input.

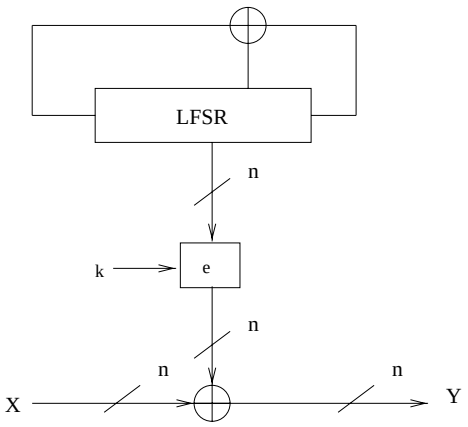

Figure 6.5: Counter Mode model

3. The block cipher output is considered a pseudorandom mask which is XORed with the plaintext.

4. The LFSR is clocked once (note: **all** input bits of the block cipher are shifted by one position).

5. Goto to Step 2.

Note that the period of a counter mode is $n \cdot 2^n$ which is very large for modern block ciphers, e.g., $128 \cdot 2^{128} = 2^{135}$ for AES algorithms.

6.2 Key Whitening

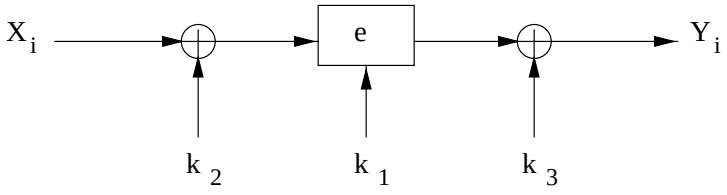

Figure 6.6: Whitening example

Encryption: $Y = e_{k_1,k_2,k_3}(X) = e_{k_1}(X \oplus k_2) \oplus k_3$.

Decryption: $X = e_{k_1}^{-1}(Y \oplus k_3) \oplus k_2$.

popular example: DESX

6.3 Multiple Encryption

6.3.1 Double Encryption

Note: The keyspace of this encryption is $|k| = 2^k \cdot 2^k = 2^{2k}$.

However, using the meet-in-the-middle attack, the key search is reduced significantly.

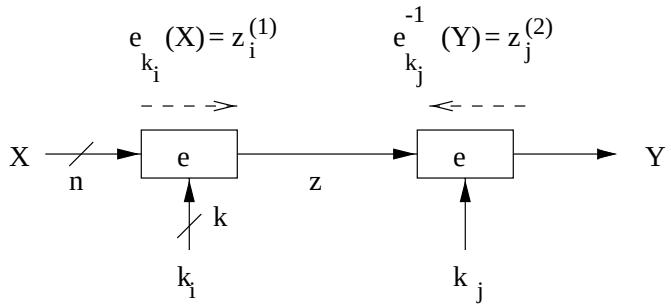

Figure 6.7: Double encryption and meet-in-the-middle attack

Meet in the middle attack:

Input $\rightarrow$ some pairs (x', y'), (x'', y''), ...

Idea $\rightarrow$ compute $z_i^{(1)} = e_{k_i}(x')$ and $z_j^{(2)} = e_{k_j}^{-1}(y')$.

Problem $\rightarrow$ to find a matching pair such that $z_i^{(1)} = z_j^{(2)}$.

Procedure:

1. Compute a look-up table for all $(z_i^{(1)}, k_i)$, $i = 1, 2, \ldots, 2^k$ and store it in memory. Number of entries in the table is 2^k with each entry being n bits wide.

2. Find matching $z_j^{(2)}$.

   (a) compute $e_{k_j}^{-1}(y') = z_j^{(2)}$

   (b) if $z_j^{(2)}$ is in the look-up table, i.e., if $z_i^{(1)} = z_j^{(2)}$, check a few other pairs $(x'', y''), (x''', y'''), \ldots$ for the current keys k_i and k_j

   (c) if k_i and k_j give matching encryptions stop; otherwise go back to (a) and try different key k_j.

Question: How many additional pairs $(x'', y''), (x''', y'''), \ldots$ should we test?

General system: l subsequent encryptions and t pairs $(x', y'), (x'', y''), \ldots$.

1. In the first step there are 2^{lk} possible key combinations for the mapping $E(x') = e(\cdots(e(e(x'))\cdots) = y'$ but only 2^n possible values for x' and y'. Hence, there are

$$\frac{2^{lk}}{2^n}$$

   mappings $E(x') = y'$. Note that only one mapping is done by the correct key!

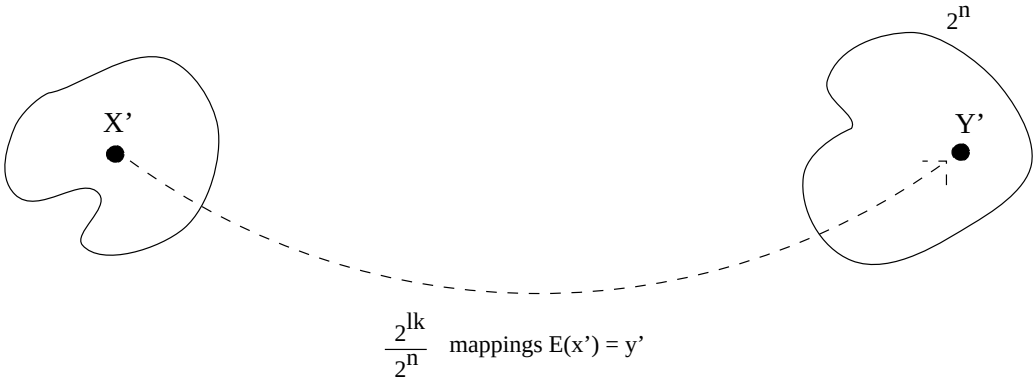

Figure 6.8: Number of mappings x' to y' under l-fold encryption

2. We use now a candidate key from step 1 and check whether $E(x'') = y''$. There are 2^n possible outcomes y for the mapping $E(x'')$. If a random key is used, the likelyhood that $E(x'') = y''$ is

$$\frac{1}{2^n}$$

If we check additionally a third pair (x''', y''') under the same "random" key from step 1, the likelyhood that $E(x'') = y''$ *and* $E(x''') = y'''$ is

$$\frac{1}{2^{2n}}$$

If we check $t-1$ additional pairs $(x'', y''), (x''', y'''), \ldots (x^{(t)}, y^{(t)})$ the likelyhood that a random key fulfills $E(x'') = y''$, $E(x''') = y'''$, ... is

$$\frac{1}{2^{(t-1)n}}$$

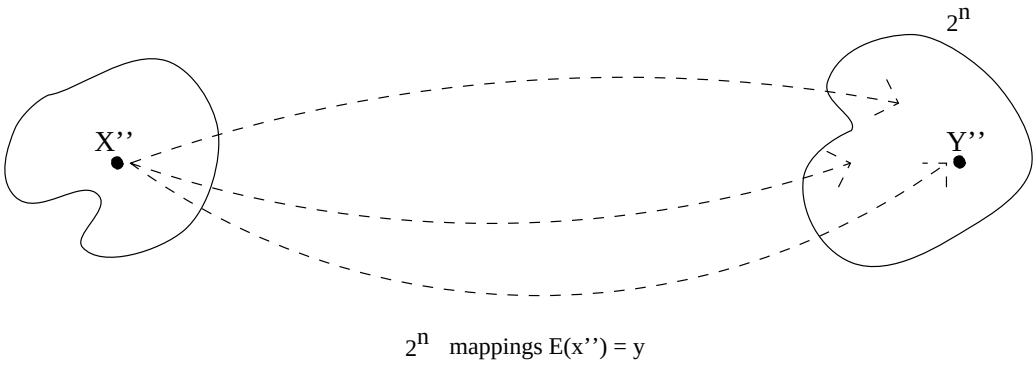

Figure 6.9: Number of mappings x'' to y

3. Since there are $\frac{2^{lk}}{2^n}$ candidate keys in step 1, the likelyhood that at least one of the candidate keys fulfills all $E(x'') = y''$, $E(x''') = y'''$, ... is

$$\frac{1}{2^{(t-1)n}} \frac{2^{lk}}{2^n} = 2^{lk-tn}$$

Example: Double encryption with DES. We use two pairs $(x', y'), (x'', y'')$. The likelyhood that an incorrect key pair k_i, k_j is picked is

$$2^{lk-tn} = 2^{112-128} = 2^{-16}$$

If we use three pairs $(x', y'), (x'', y''), (x''', y''')$, the likelyhood that an incorrect key pair k_i, k_j is picked is

$$2^{lk-tn} = 2^{112-192} = 2^{-80}$$

Computational complexity:

Brute force attack: 2^{2k}.

Meet in the middle attack: 2^k encryptions + 2^k decryptions = 2^{k+1} computations and 2^k memory locations.

6.3.2 Triple Encryption

Option 1:

$Y = e_{k_1}(e_{k_2}^{-1}(e_{k_1}(X)))$; if $k_1 = k_2 \rightarrow Y = e_{k_1}(X)$.

Option 2:

$Y = e_{k_3}(e_{k_2}(e_{k_1}(X)))$; where $|k| \approx 2^{2k}$

Option 2 should be preferred.

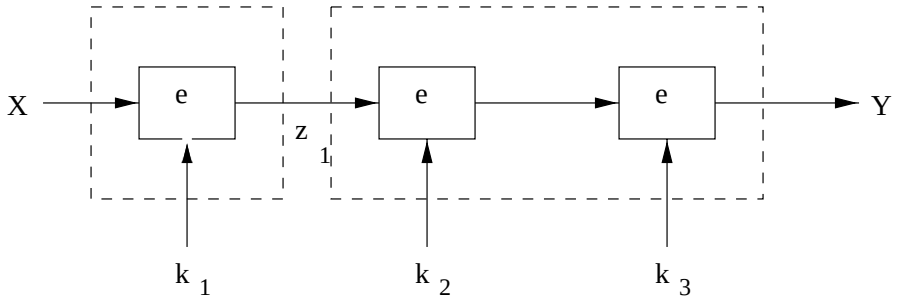

Figure 6.10: Triple encryption example

Note:

Meet in the middle attack can be used in a similar way by storing z_i results in memory. The computational complexity of this approach is $2^k \cdot 2^k = 2^{2k}$.