Modes of Operation

- Encryption with Block Ciphers: Modes of Operation
  - Electronic Code Book mode (ECB)
  - Cipher Block Chaining mode (CBC)
  - Output Feedback mode (OFB)
  - Cipher Feedback mode (CFB)
  - Counter mode (CTR)
  - Galois Counter Mode (GCM)
- Exhaustive Key Search Revisited
- Increasing the Security of Block Ciphers

Block Ciphers

- A block cipher is much more than just an encryption algorithm, it can be used ...
  - to build different types of block-based encryption schemes
  - to realize stream ciphers
  - to construct hash functions
  - to make message authentication codes
  - to build key establishment protocols
  - to make a pseudo-random number generator
  - ...

- The security of block ciphers also can be increased by
  - key whitening
  - multiple encryption

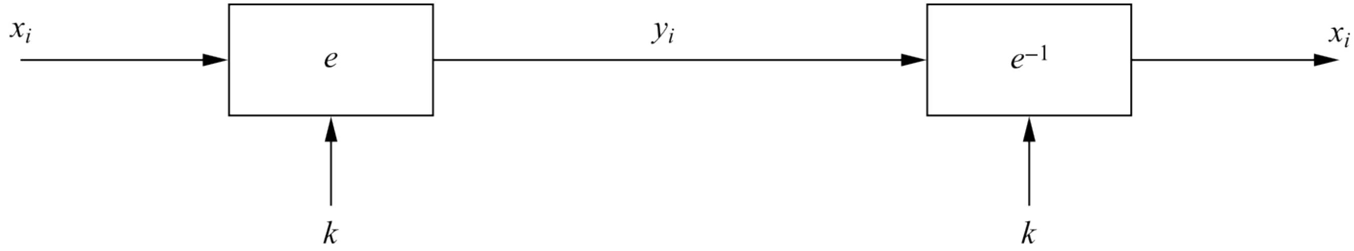
Encryption with Block Ciphers

- There are several ways of encrypting long plaintexts, e.g., an e-mail or a computer file, with a block cipher ("modes of operation")
  - Electronic Code Book mode (ECB)
  - Cipher Block Chaining mode (CBC)
  - Output Feedback mode (OFB)
  - Cipher Feedback mode (CFB)
  - Counter mode (CTR)
  - Galois Counter Mode (GCM)

- All of the 6 modes have one goal:
  - In addition to confidentiality, they provide authenticity and integrity:
    - Is the message really coming from the original sender? (authenticity)
    - Was the ciphertext altered during transmission? (integrity)

Electronic Code Book mode (ECB)

- $e_k(x_i)$ denote the encryption of a b-bit plaintext block x_i with key k
- $e_k^{-1}(y_i)$ denote the decryption of b-bit ciphertext block y_i with key k
- Messages which exceed b bits are partitioned into b-bit blocks
- **Each Block is encrypted separately**



Encryption: $y_i = e_k(x_i), \ \ i \geq 1$

Decryption: $x_i = e_k^{-1}(y_i) = e_k^{-1}(e_k(x_i)), \ \ i \geq 1$

ECB: advantages/disadvantages

- Advantages
  - no block synchronization between sender and receiver is required
  - bit errors caused by noisy channels only affect the corresponding block but not succeeding blocks
  - Block cipher operating can be parallelized
    - advantage for high-speed implementations

- Disadvantages
  - ECB encrypts highly deterministically
    - identical plaintexts result in identical ciphertexts
    - an attacker recognizes if the same message has been sent twice
    - plaintext blocks are encrypted independently of previous blocks
      - an attacker may reorder ciphertext blocks which results in valid plaintext

Substitution Attack on ECB

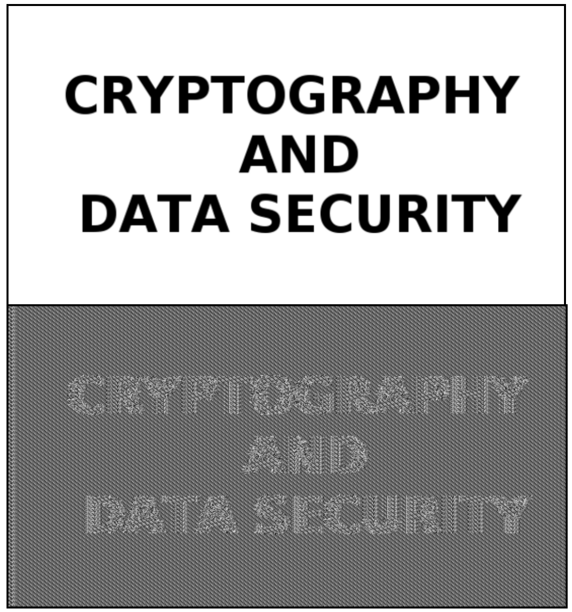
- Once a particular plaintext to ciphertext block mapping $x_i \rightarrow y_i$ is known, a sequence of ciphertext blocks can easily be manipulated
- Suppose an *electronic bank transfer*

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

  - the encryption key between the two banks does not change too frequently
  - The attacker sends $1.00 transfers from his account at bank A to his account at bank B repeatedly
    - He can check for ciphertext blocks that repeat, and he stores blocks 1,3 and 4 of these transfers
  - He now simply replaces block 4 of other transfers with the block 4 that he stored before
    - *all transfers* from some account of bank A to some account of bank B are redirected to go into the attacker's B account!

Example of encrypting bitmaps in ECB mode

- Identical plaintexts are mapped to identical ciphertexts



- Statistical properties in the plaintext are preserved in the ciphertext

Cipher Block Chaining mode (CBC)

- There are two main ideas behind the CBC mode:
  - The encryption of all blocks are "chained together"
    - ciphertext y_i depends not only on block x_i but on all previous plaintext blocks as well
  - The encryption is randomized by using an initialization vector (IV)

***Encryption* (*first block*)**: $y_1 = e_k(x_1 \oplus \mathrm{IV})$

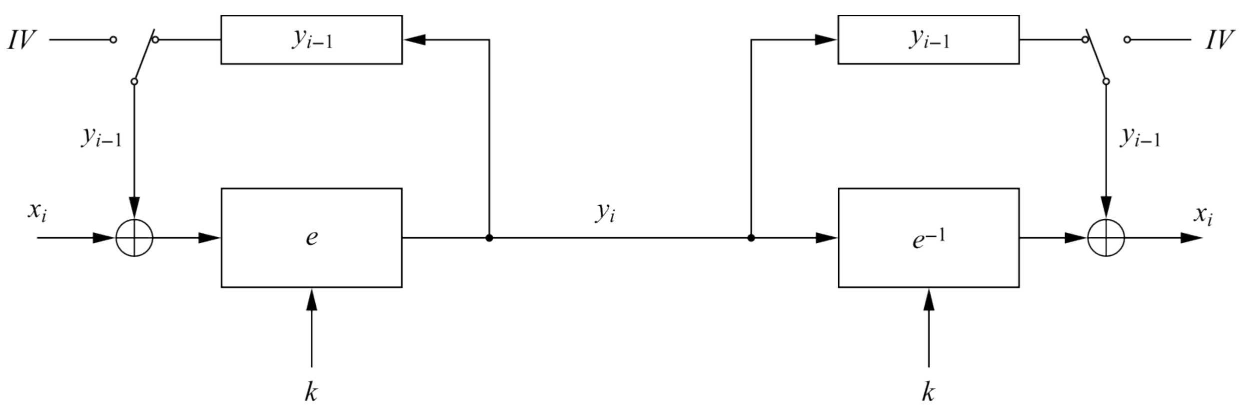
***Encryption* (*general block*)**: $y_i = e_k(x_i \oplus y_{i-1}), \quad i \geq 2$

***Decryption* (*first block*)**: $x_1 = e_k^{-1}(y_1) \oplus \mathrm{IV}$

***Decryption* (*general block*)** : $x_i = e_k^{-1}(y_i) \oplus y_{i-1}, \quad i \geq 2$

Cipher Block Chaining mode (CBC)

- For the first plaintext block x_1 there is no previous ciphertext
  - an IV is added to the first plaintext to make each CBC encryption nondeterministic
  - the first ciphertext y_1 depends on plaintext x_1 and the IV
- The second ciphertext y_2 depends on the IV, x_1 *and* x_2
- The third ciphertext y_3 depends on the IV and x_1, x_2 *and* x_3, and so on

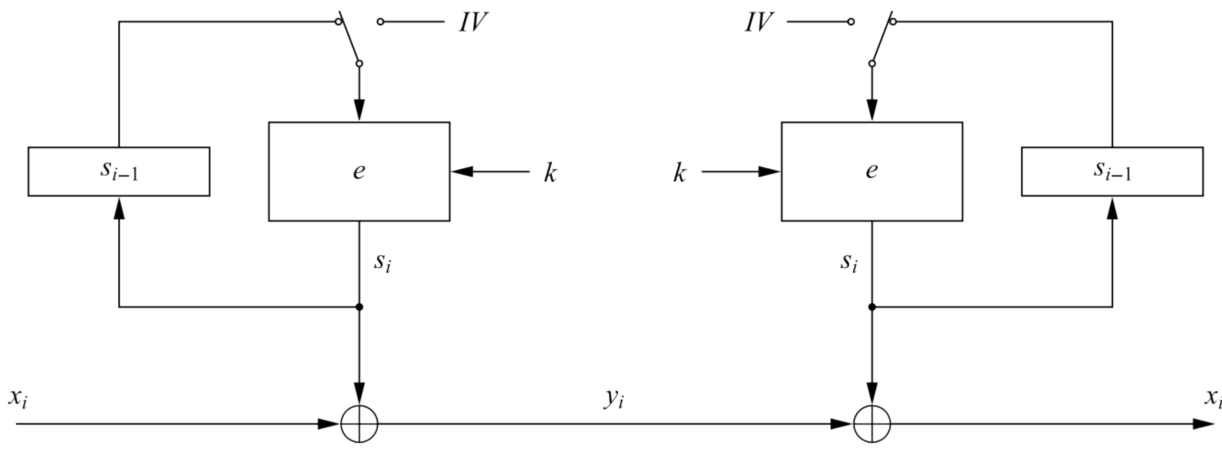


Substitution Attack on CBC

- Suppose the last example (*electronic bank transfer*)
- If the IV is properly chosen for every wire transfer, the attack will not work at all
- If the IV is kept the same for several transfers, the attacker would recognize the transfers from his account at bank A to back B
- If we choose a new IV every time we encrypt, the CBC mode becomes a probabilistic encryption scheme, i.e., two encryptions of the same plaintext look entirely different
- It is not needed to keep the IV *secret*!
- Typically, the IV should be a non-secret nonce (value used only once)

Output Feedback mode (OFB)

- It is used to build a *synchronous* **stream cipher** from a block cipher
- The key stream is not generated bitwise but instead in a blockwise fashion
- The output of the cipher gives us key stream bits S_i with which we can encrypt plaintext bits using the XOR operation



***Encryption* (*first block*)**: $s_1 = e_k(\text{IV})$ *and* $y_1 = s_1 \oplus x_1$

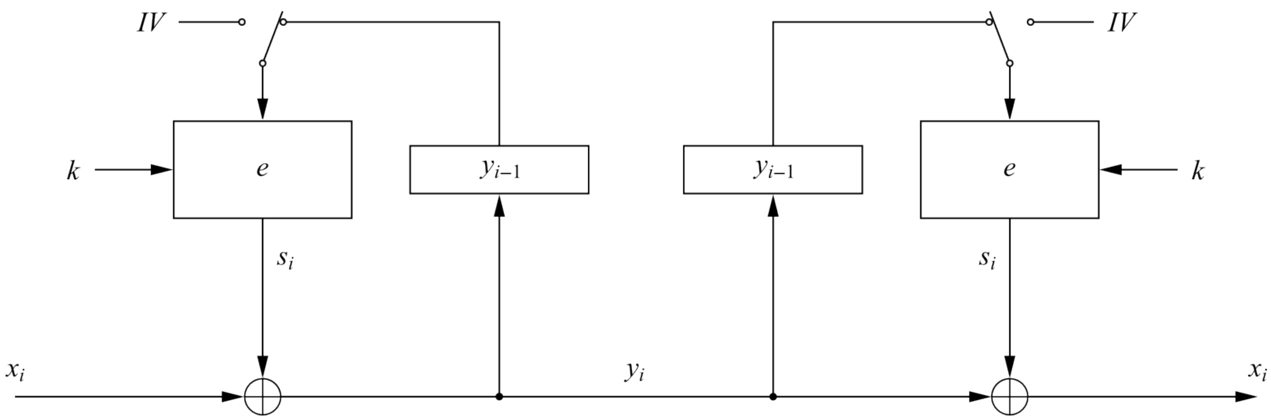
***Encryption* (*general block*)**: $s_i = e_k(s_{i-1})$ *and* $y_i = s_i \oplus x_i$, $i \geq 2$

***Decryption* (*first block*)**: $s_1 = e_k(\text{IV})$ *and* $x_1 = s_1 \oplus y_1$

***Decryption* (*general block*)** : $s_i = e_k(s_{i-1})$ *and* $x_i = s_i \oplus y_i$, $i \geq 2$

Cipher Feedback mode (CFB)

- It uses a block cipher as a building block for an asynchronous **stream cipher** (similar to the OFB mode), more accurate name: "Ciphertext Feedback Mode"
- The key stream S_i is generated in a blockwise fashion and is also a function of the ciphertext
- As a result of the use of an IV, the CFB encryption is also nondeterministic

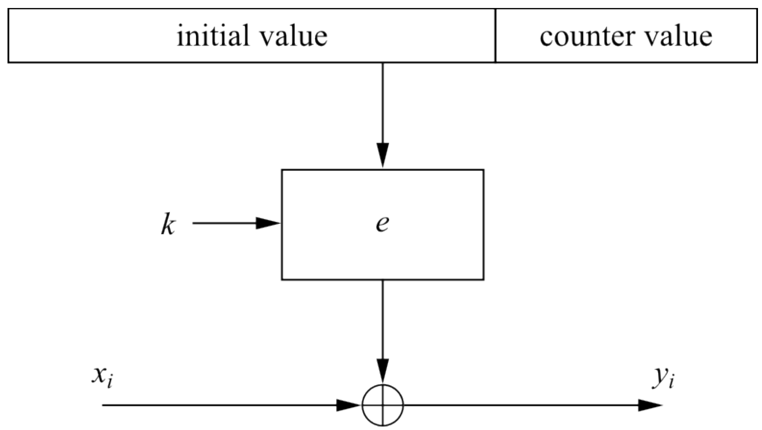


Encryption (first block): $y_1 = e_k(\mathrm{IV}) \oplus x_1$
Encryption (general block): $y_i = e_k(y_{i-1}) \oplus x_i, \quad i \geq 2$
Decryption (first block): $x_1 = e_k(\mathrm{IV}) \oplus y_1$
Decryption (general block) : $x_i = e_k(y_{i-1}) \oplus y_i, \quad i \geq 2$

- It can be used in situations where short plaintext blocks are to be encrypted

Counter mode (CTR)

- It uses a block cipher as a **stream cipher** (like the OFB and CFB modes)
- The key stream is computed in a blockwise fashion
- The input to the block cipher is a counter which assumes a different value every time the block cipher computes a new key stream block



- Unlike CFB and OFB modes, the CTR mode can be parallelized since the 2nd encryption can begin before the 1st one has finished
  - Desirable for high-speed implementations, e.g., in network routers

Encryption: $y_i = e_k(\mathrm{IV} \,||\, \mathrm{CTR}_i) \oplus x_i, \quad i \geq 1$

Decryption : $x_i = e_k(\mathrm{IV} \,||\, \mathrm{CTR}_i) \oplus y_i, \quad i \geq 1$

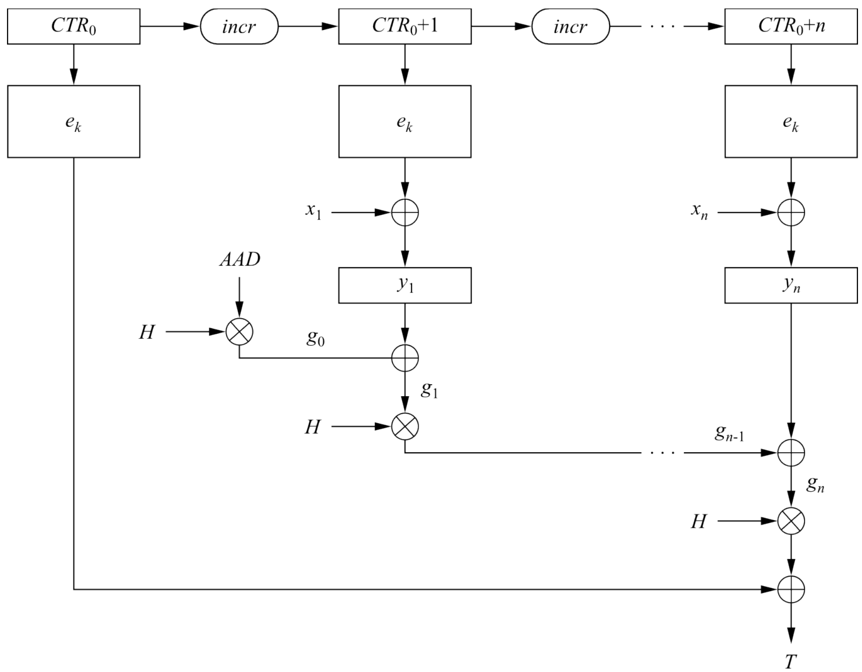
Galois Counter Mode (GCM)

- It also computes a *message authentication code* (MAC), i.e., a cryptographic checksum is computed for a message
- By making use of GCM, two additional services are provided:
  - Message Authentication
    - the receiver can make sure that the message was really created by the original sender
  - Message Integrity
    - the receiver can make sure that nobody tampered with the ciphertext during transmission

Galois Counter Mode (GCM)

- For encryption
  - An initial counter is derived from an IV and a serial number
  - The initial counter value is incremented then encrypted and XORed with the first plaintext block
  - For subsequent plaintexts, the counter is incremented and then encrypted
- For authentication
  - A chained Galois field multiplication is performed
  - For every plaintext an intermediate authentication parameter g_i is derived
    - g_i is computed as the XOR of the current ciphertext and the last g_{i-1}, and multiplied by the constant H
      - H is generated by encryption of the zero input with the block cipher
  - All multiplications are in the 128-bit Galois field $GF(2^{128})$

Galois Counter Mode (GCM)



Encryption:

a. Derive a counter value CTR_0 from the IV and compute $\mathrm{CTR}_1 = \mathrm{CTR}_0 + 1$

b. Compute ciphertext: $y_\mathrm{i} = e_k(\mathrm{CTR}_\mathrm{i}) \oplus x_i, \quad i \geq 1$

Authentication:

a. Generate authentication subkey $H = e_k(0)$

b. Compute $g_0 = AAD \times H$ (Galois field multiplication)

c. Compute $g_i = (g_{i-1} \oplus y_i) \times H,\ 1 \leq i \leq n$ (Galois field multiplication)

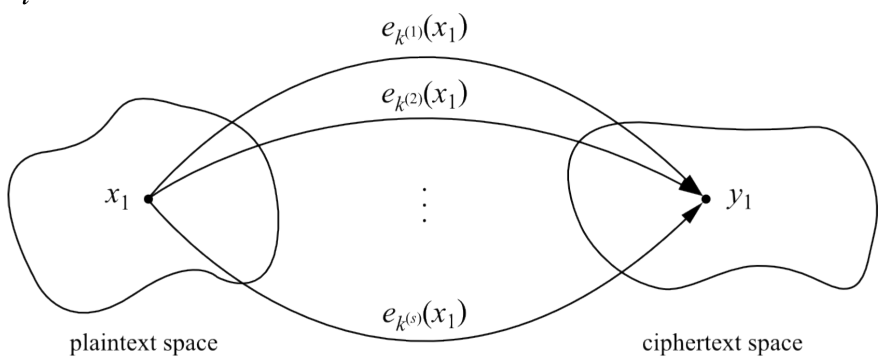
d. Final authentication tag: $T = (g_n \times H) \oplus e_\mathrm{k}(\mathrm{CTR}_0)$

Exhaustive Key Search Revisited

- A simple exhaustive search for a DES key knowing one pair (x_1, y_1):

$$DES_{k^{(i)}}(x_1) \stackrel{?}{=} y_1, \quad i = 0,1, \ldots, 2^{56}-1$$

- However, for most other block ciphers a key search is somewhat more complicated
- A brute-force attack can produce *false positive* results
  - keys k_i that are found are not the one used for the encryption



  - The likelihood of this is related to the relative size of the key space and the plaintext space
  - A brute-force attack is still *possible*, but several pairs of plaintext–ciphertext are needed

An Exhaustive Key Search Example

- Assume a cipher with a block width of 64 bit and a key size of 80 bit
- If we encrypt x_1 under all possible 2^{80} keys, we obtain 2^{80} ciphertexts
  - However, there exist only 2^{64} different ones
- If we run through all keys for a given plaintext–ciphertext pair, we find on average $2^{80}/2^{64} = 2^{16}$ keys that perform the mapping $e_k(x_1) = y_1$

> Given a block cipher with a key length of k bits and block size of n bits, as well as t plaintext–ciphertext pairs $(x_1, y_1), \ldots, (x_t, y_t)$, the expected number of *false* keys which encrypt all plaintexts to the corresponding ciphertexts is:
>
> $$2^{k-tn}$$

- In this example assuming two plaintext-ciphertext pairs, the likelihood is

$$2^{80-2\cdot 64} = 2^{-48}$$

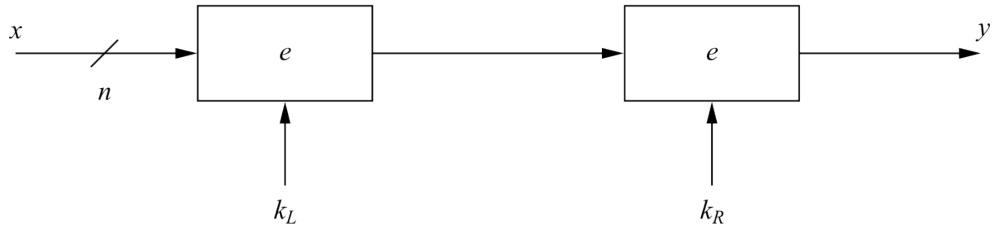
  - for almost all practical purposes two plaintext-ciphertext pairs are sufficient

Increasing the Security of Block Ciphers

- In some situations we wish to increase the security of block ciphers, e.g., if a cipher such as DES is available in hardware or software for legacy reasons in a given application
- Two approaches are possible
  - Multiple encryption
    - theoretically much more secure, but **sometimes** in practice increases the security very little
  - Key whitening

Double Encryption

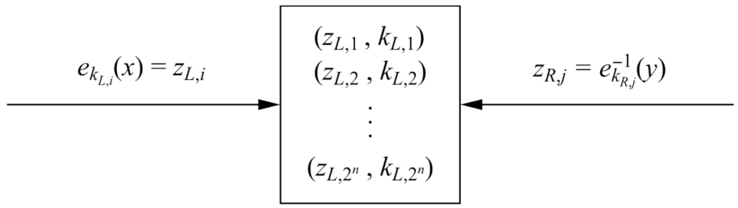
- A plaintext x is first encrypted with a key k_L, and the resulting ciphertext is encrypted again using a second key k_R



- Assuming a key length of k bits, an exhaustive key search would require $2^k \cdot 2^k = 2^{2k}$ encryptions or decryptions

Meet-in-the-Middle Attack

- A Meet-in-the-Middle attack requires $2^k + 2^k = 2^{k+1}$ operations!



  - **Phase I**: for the given (x_1, y_1) the **left** encryption is brute-forced for all $k_{L,i}$, i=1,2, ..., 2^k and a lookup table with 2^k entry (each $n+k$ bits wide) is computed
    - the lookup table should be ordered by the result of the encryption ($z_{L,i}$)
  - **Phase II**: the **right** encryption is brute-forced (using decryption) and for each $z_{R,i}$ it is checked whether $z_{R,i}$ is equal to any $z_{L,i}$ value in the table of the first phase

- Computational Complexity

number of encryptions and decryptions = $2^k + 2^k = 2^{k+1}$ number of storage locations = 2^k

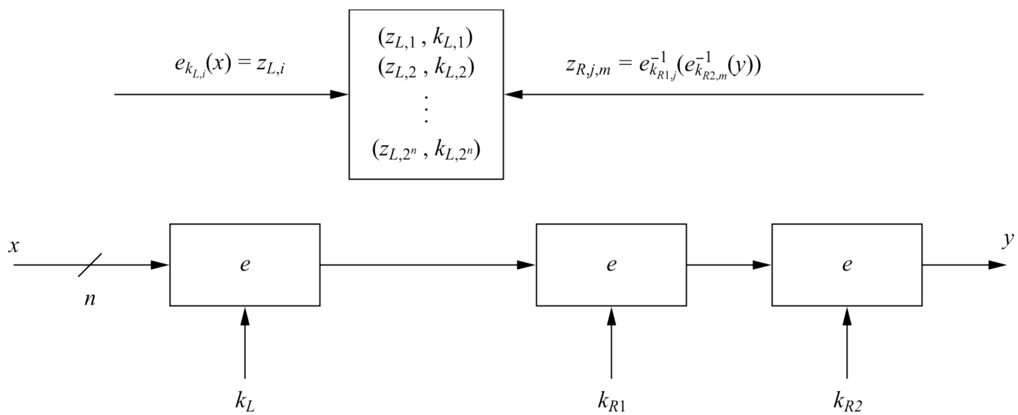
- **Double encryption is not much more secure then single encryption!**

Triple Encryption

- The encryption of a block three times $y = e_{k3}(e_{k2}(e_{k1}(x)))$
- In practice a variant scheme is often used EDE (encryption-decryption-encryption)

$$y = e_{k3}(e^{-1}_{k2}(e_{k1}(x)))$$

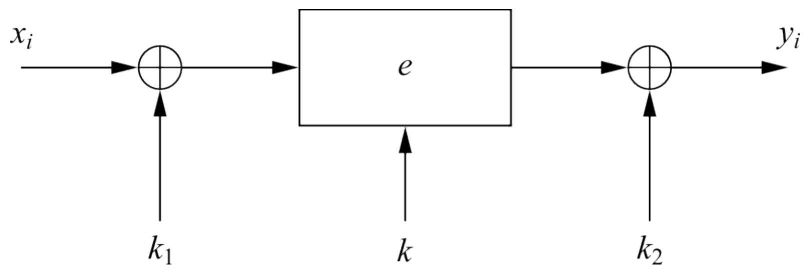
  - Advantage: choosing $k1=k2=k3$ performs single DES encryption
- Still we can perform a meet-in-the middle attack, and it reduces the *effective key length* of triple encryption from $3K$ to $2K$!
  - The attacker must run 2^{112} tests in the case of 3DES



- **Triple encryption effectively doubles the key length**

Key Whitening

- Makes block ciphers such as DES much more resistant against brute-force attacks
- In addition to the regular cipher key k, two whitening keys k_1 and k_2 are used to XOR-mask the plaintext and ciphertext



- It does not strengthen block ciphers against most analytical attacks such as linear and differential cryptanalysis
- It is not a "cure" for inherently weak ciphers
- The additional computational load is negligible
- Its main application is ciphers that are relatively strong against analytical attacks but possess too short a key space especially DES
  - a variant of DES which uses key whitening is called DESX

Lessons Learned

- There are many different ways to encrypt with a block cipher. Each mode of operation has some advantages and disadvantages
- Several modes turn a block cipher into a stream cipher
- There are modes that perform encryption together together with authentication, i.e., a cryptographic checksum protects against message manipulation
- The straightforward ECB mode has security weaknesses, independent of the underlying block cipher
- The counter mode allows parallelization of encryption and is thus suited for high speed implementations
- Double encryption with a given block cipher only marginally improves the resistance against brute-force attacks
- Triple encryption with a given block cipher roughly *doubles* the key length
- Triple DES (3DES) has an effective key length of 112 bits
- Key whitening enlarges the DES key length without much computational overhead.