Cryptography

Hemant Ghayvat
Department of Computer Science and Media Technology
hemant.ghayvat@lnu.se

Stream and Block Ciphers

- Both uses symmetric encryption key
- Stream Cipher: It encrypts a digital data stream one bit or 1 byte at a time
- Block Ciphers: In this a block of plain text is treated as a whole and used to produce the ciphertext of equal length, Typically a block size of 64 or 128 bits.

Block Ciphers

Block Cipher

- Plaintext and ciphertext consists of fixed sized blocks
- Ciphertext obtained from plaintext by iterating a round function
- Input to round function consists of key and the output of previous round
- Usually implementation friendly. Gives a high throughput


IBM
IBM

Feistel Cipher (1973 IBM)

- Feistel cipher refers to splitting the plaintext into two equal parts
- Split plaintext block into left and right halves: Plaintext = (L_0,R_0)
- These two halves of the data pass through n rounds of processing and then combine to produce the ciphertext block
- For each round i=1,2,...,n, compute

  $L_i = R_{i-1}$

  $R_i = L_{i-1} \oplus F(R_{i-1},K_i)$

  where F is round function and K_i is subkey (*refer next slide for it*)
- Ciphertext = (L_n,R_n)

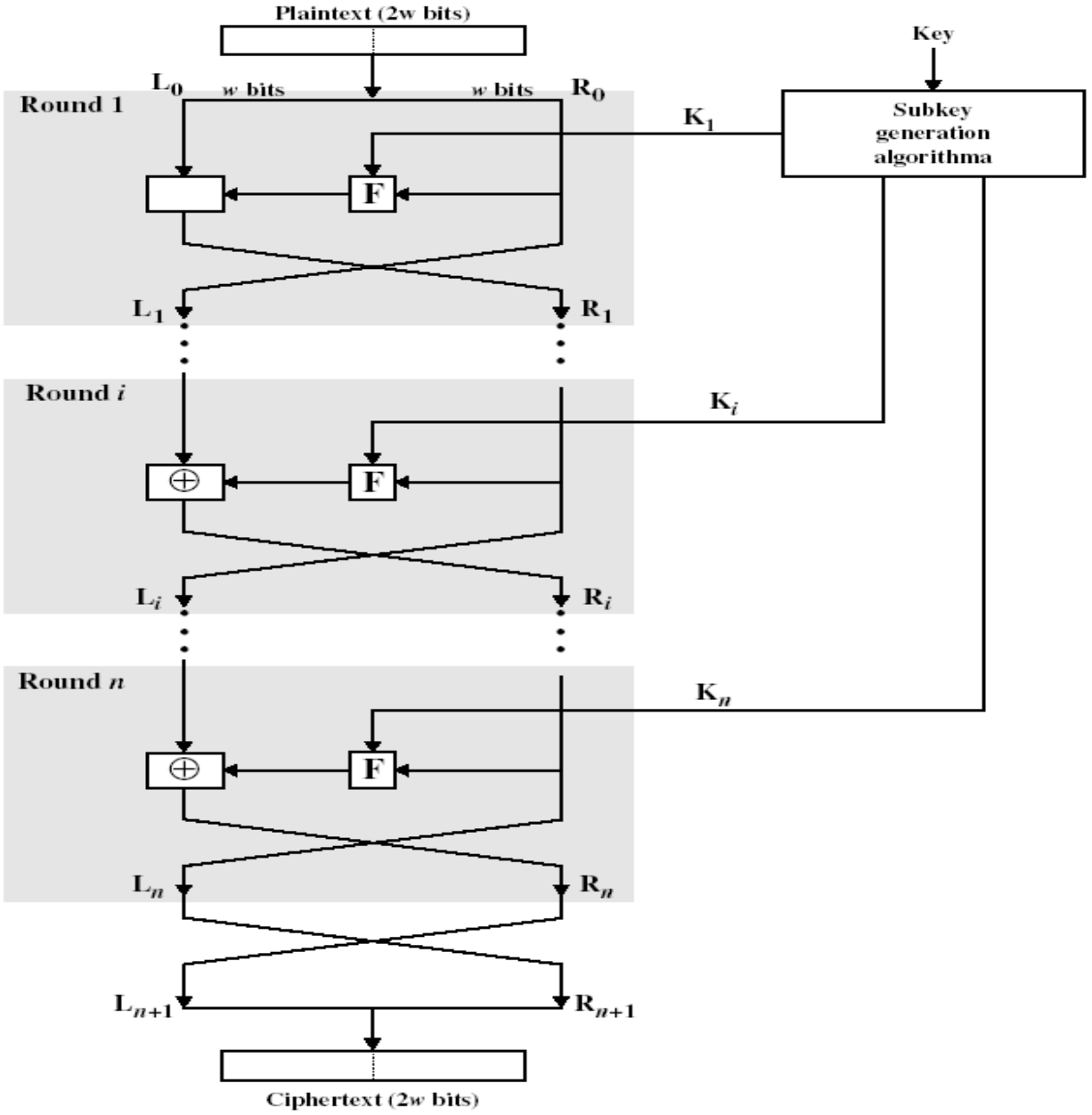
Feistel Cipher: Sub Key

- On the right half we apply a function and in the function, we use a subkey generated from the master key (main key)
- A substitution is performed on the left half of the data. This is done by applying a round function to the right half of the data followed by the XOR of the output of that function and the left half of the data.
- That’s count the first round.
- All rounds have the same structure
- All conventional block encryption algorithms including data encryption standard (DES) are based on Feistel Cipher Structure.

Feistel Cipher

- Decryption: Ciphertext = (L_n,R_n)
- For each round i=n,n–1,…,1, compute
  $R_{i-1} = L_i$
  $L_{i-1} = R_i \oplus F(R_{i-1},K_i)$
  where F is round function and K_i is subkey
- Plaintext = (L_0,R_0)
- Formula “works” for any function F
- But only secure for certain functions F

Classical Feistel Network



Design Features of Feistel Network

- **Block Size:** (larger block means greater security).
- **Key Size**:56-128 bits.
- **Number of Rounds**: a single round offers inadequate security; a typical size is 16 rounds.
- **Sub-key Generation Algorithms**: greater complexity should lead to a greater difficulty of cryptanalysis.
- **Round function:** Again, greater complexity generally means greater resistance to cryptanalysis.

Block Ciphers in Practice

Data Encryption Standard (DES)

- Developed by IBM and adopted by NIST in 1977
- 64-bit blocks and 56-bit keys
- Small key space makes exhaustive search attack feasible since late 90s

Triple DES (3DES)

- Nested application of DES with three different keys KA, KB, and KC
- Effective key length is 168 bits, making exhaustive search attacks unfeasible
- $C = E_{KC}(D_{KB}(E_{KA}(P)))$; $P = D_{KA}(E_{KB}(D_{KC}(C)))$
- Equivalent to DES when KA=KB=KC (backward compatible)

Advanced Encryption Standard (AES)

- Selected by NIST in 2001 through open international competition and public discussion
- 128-bit blocks and several possible key lengths: 128, 192 and 256 bits
- Exhaustive search attack not currently possible
- AES-256 is the symmetric encryption algorithm of choice

Data Encryption Standard (DES)

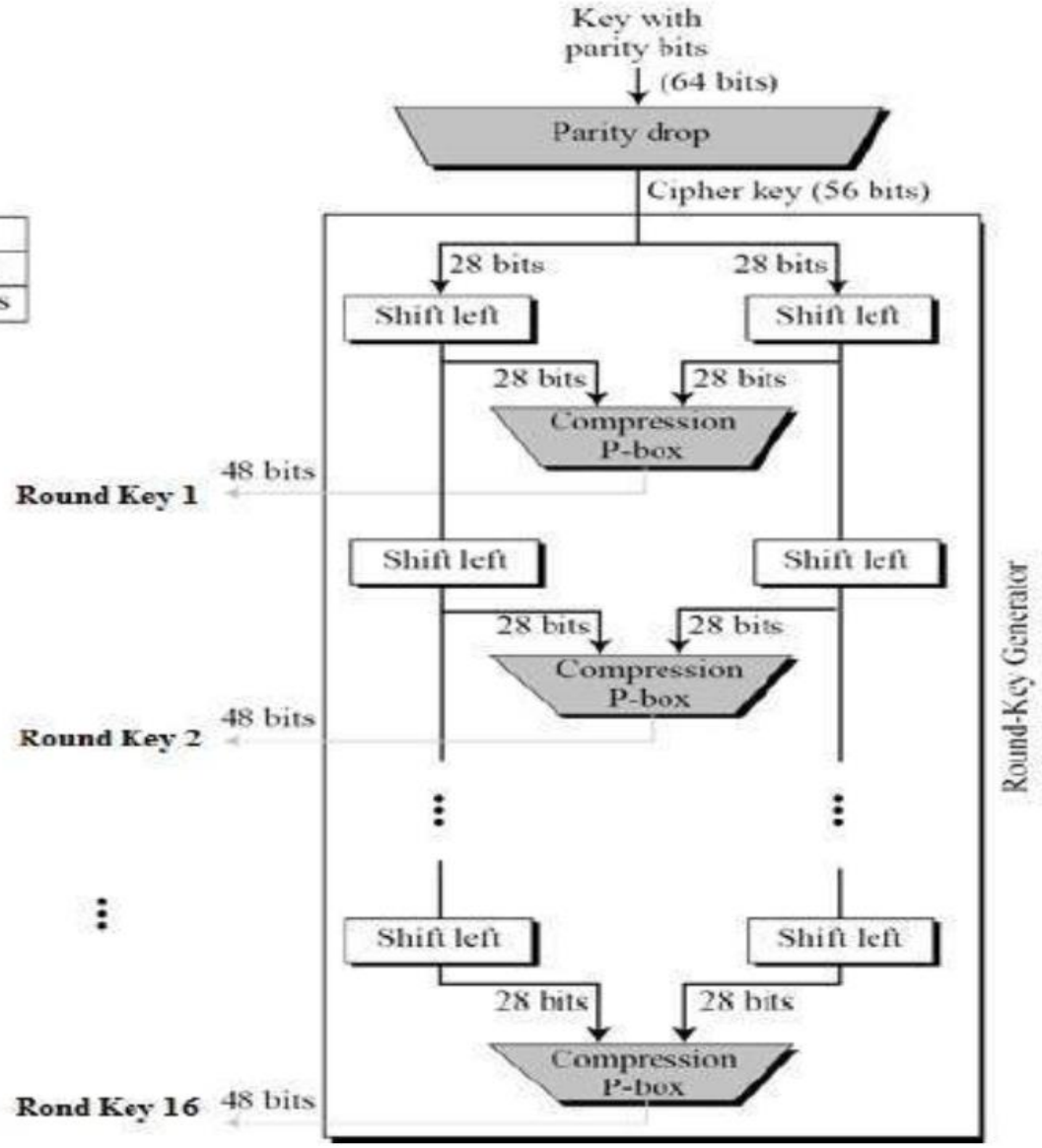
- Block Cipher
- Symmetric cipher (same Keys for the encryption and decryption)
- 64 bit plaintext block
- 16 fiestel round

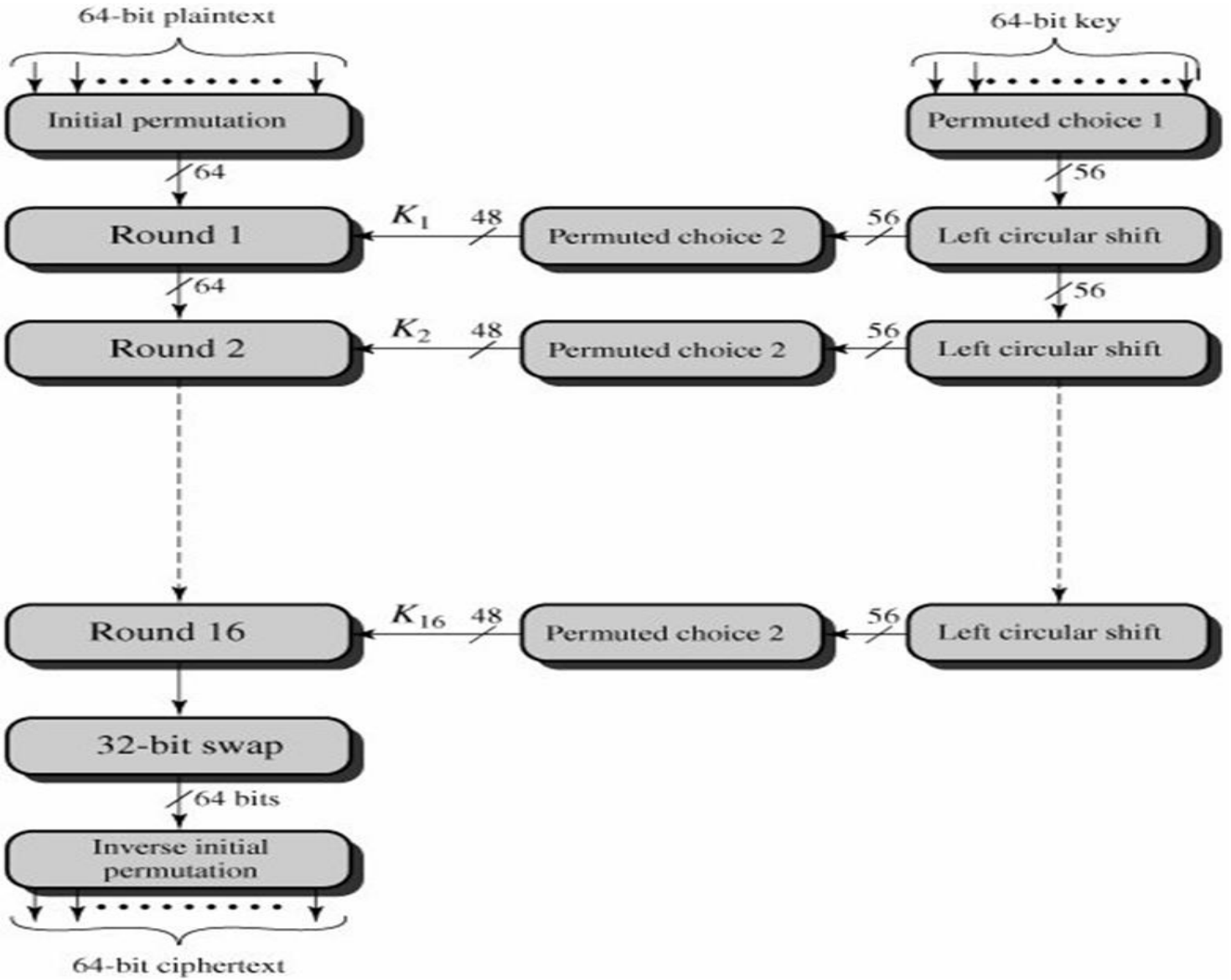
- Steps in DES:

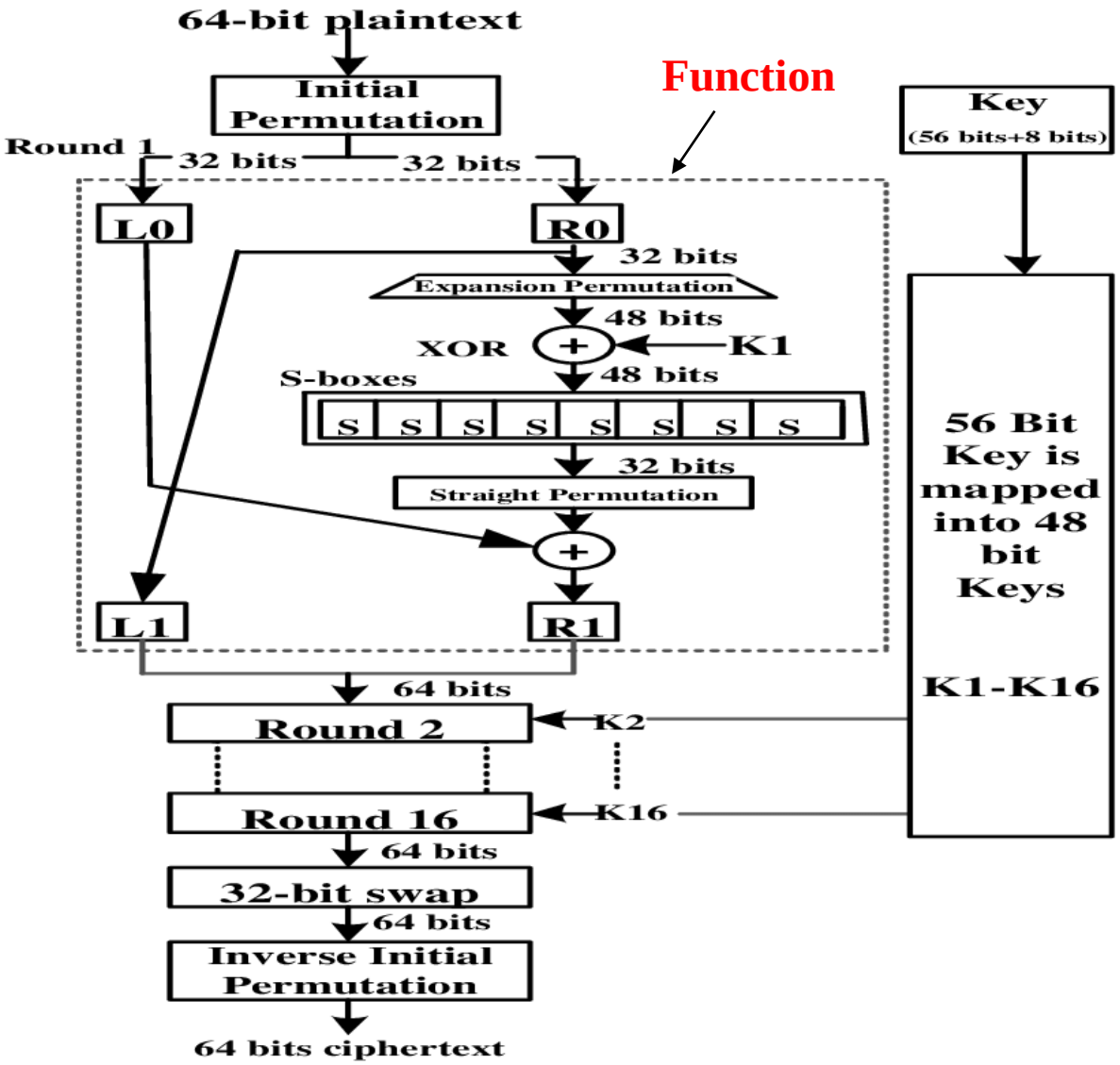
I. Initial Permutation
II. 16 Feistel rounds
III. Swapping or left right swap
IV. Final permutation or inverse initial permutation

Shifting

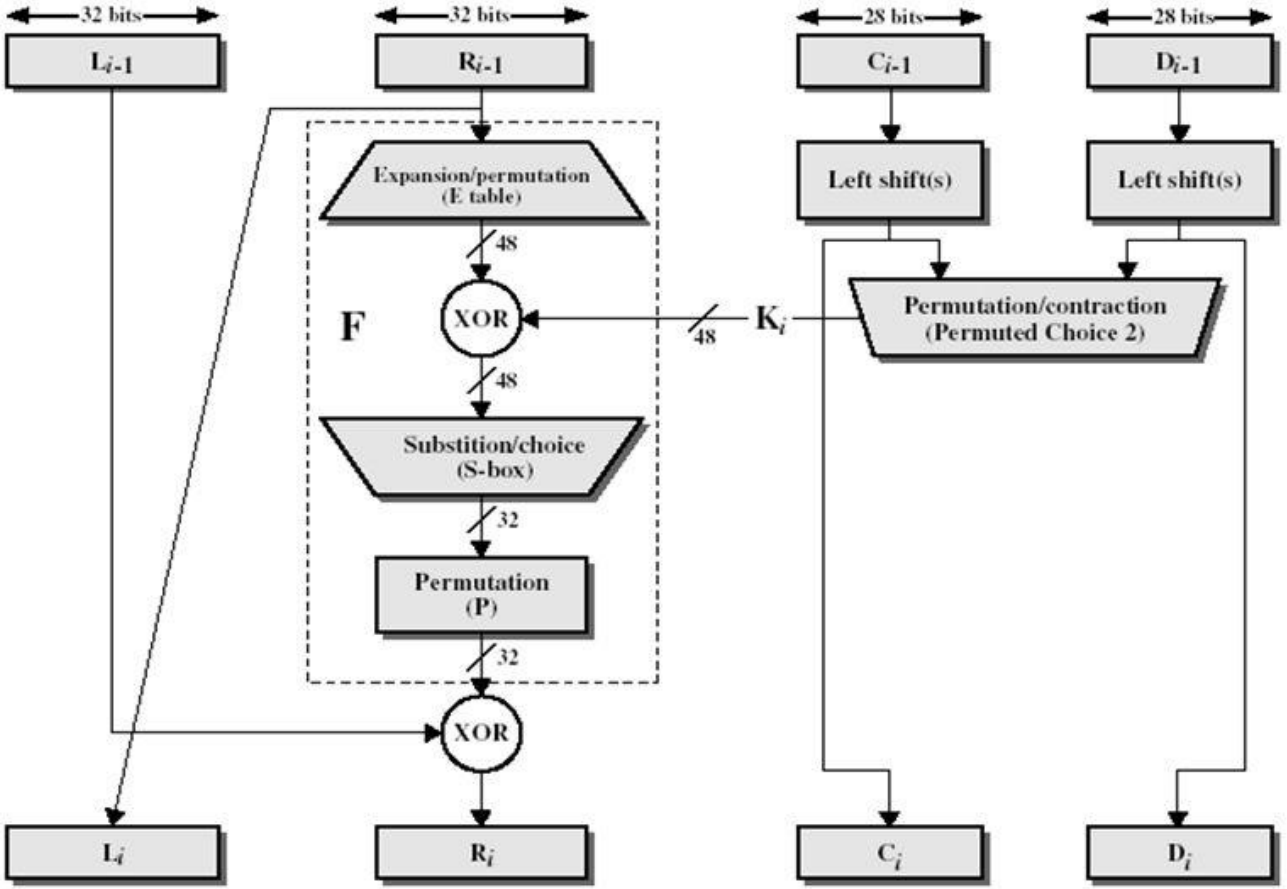
Rounds	Shift
1, 2, 9, 16	one bit
Others	two bits




64-bit plaintext
64-bit key
Initial permutation
Permuted choice 1
64
56
Round 1
K_1
48
Permuted choice 2
56
Left circular shift
64
56
Round 2
K_2
48
Permuted choice 2
56
Left circular shift
Round 16
K_{16}
48
Permuted choice 2
56
Left circular shift
32-bit swap
64 bits
Inverse initial permutation
64-bit ciphertext


64-bit plaintext
Initial Permutation
Round 1
32 bits
32 bits
L0
R0
Function
Key
(56 bits+8 bits)
32 bits
Expansion Permutation
48 bits
XOR
K1
S-boxes
48 bits
S S S S S S S S
32 bits
Straight Permutation
L1
R1
56 Bit Key is mapped into 48 bit Keys
K1-K16
64 bits
Round 2
K2
Round 16
K16
64 bits
32-bit swap
64 bits
Inverse Initial Permutation
64 bits ciphertext

Single Iteration of DES Algorithm



DES Analysis

The DES satisfies both the desired properties of block cipher. These two properties make cipher very strong.

- **Avalanche effect** − A small change in plaintext results in the very great change in the ciphertext.
- **Completeness** − Each bit of ciphertext depends on many bits of plaintext.

- Problem with DES

➢ Broken in 1998 by Electronic Frontier Foundation

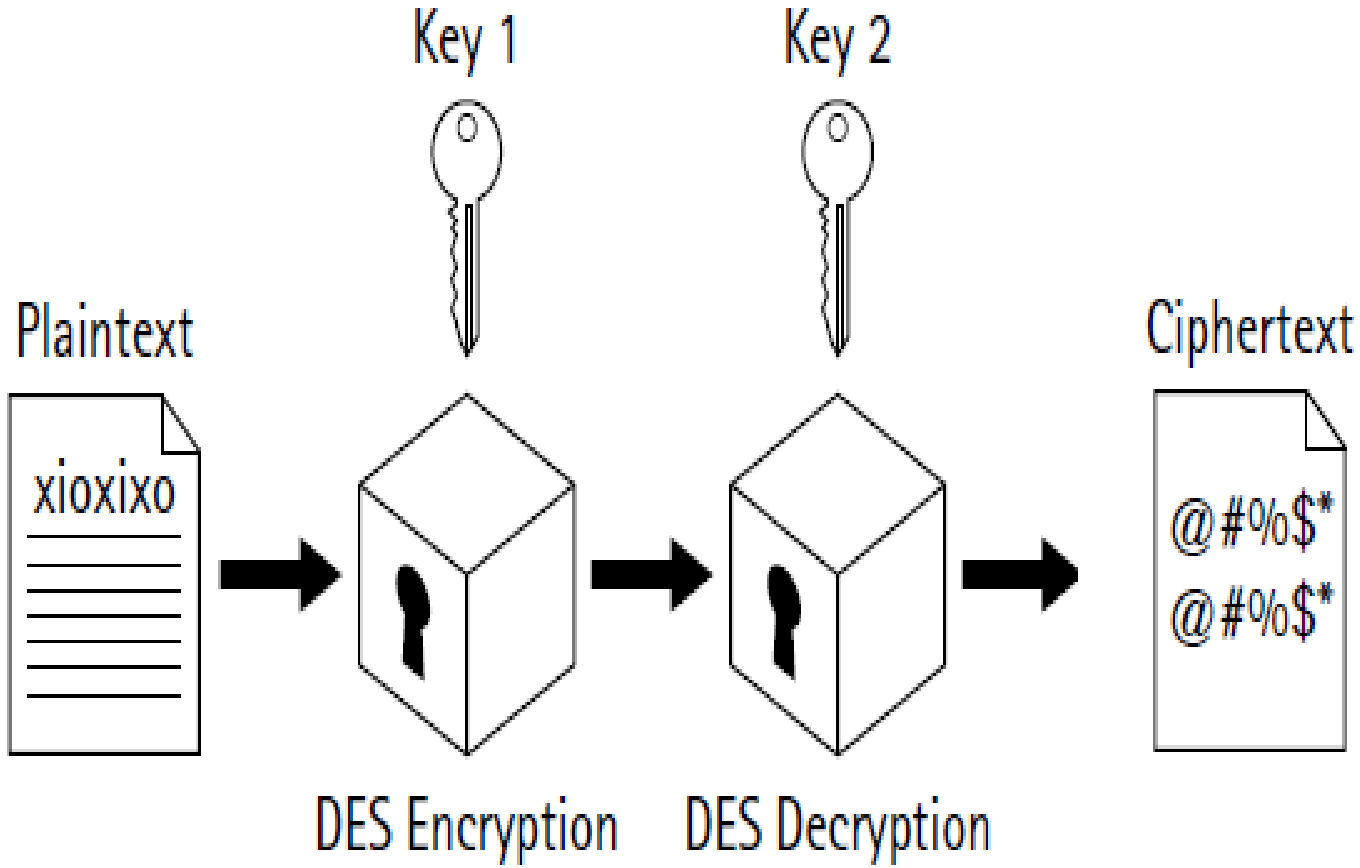
➢ Used special purpose machine - $250,000 [a]Took less than three days

Double DES

- DES uses a 56-bit key, this raised concerns about brute force attacks.
- Double uses two keys, K1 and K2
- Perform DES on the plaintext using K1 to get encrypt text.
- Again, perform DES on the encrypt text using K2.
- The final output is the encryption of the encrypted text.
- his leads to a 2x56=112 bit key, so it is more secure than DES. Is it?
- Double DES has a 112-bit key, and ciphers blocks of 64 bits.

$$p \rightarrow E(k_1, \quad p) \rightarrow E(k_2, \quad E(k_1, \quad p)) = C$$

Double DES


Key 1
Key 2
Plaintext
xioxixo
Ciphertext
@#%$*
@#%$*
DES Encryption
DES Decryption

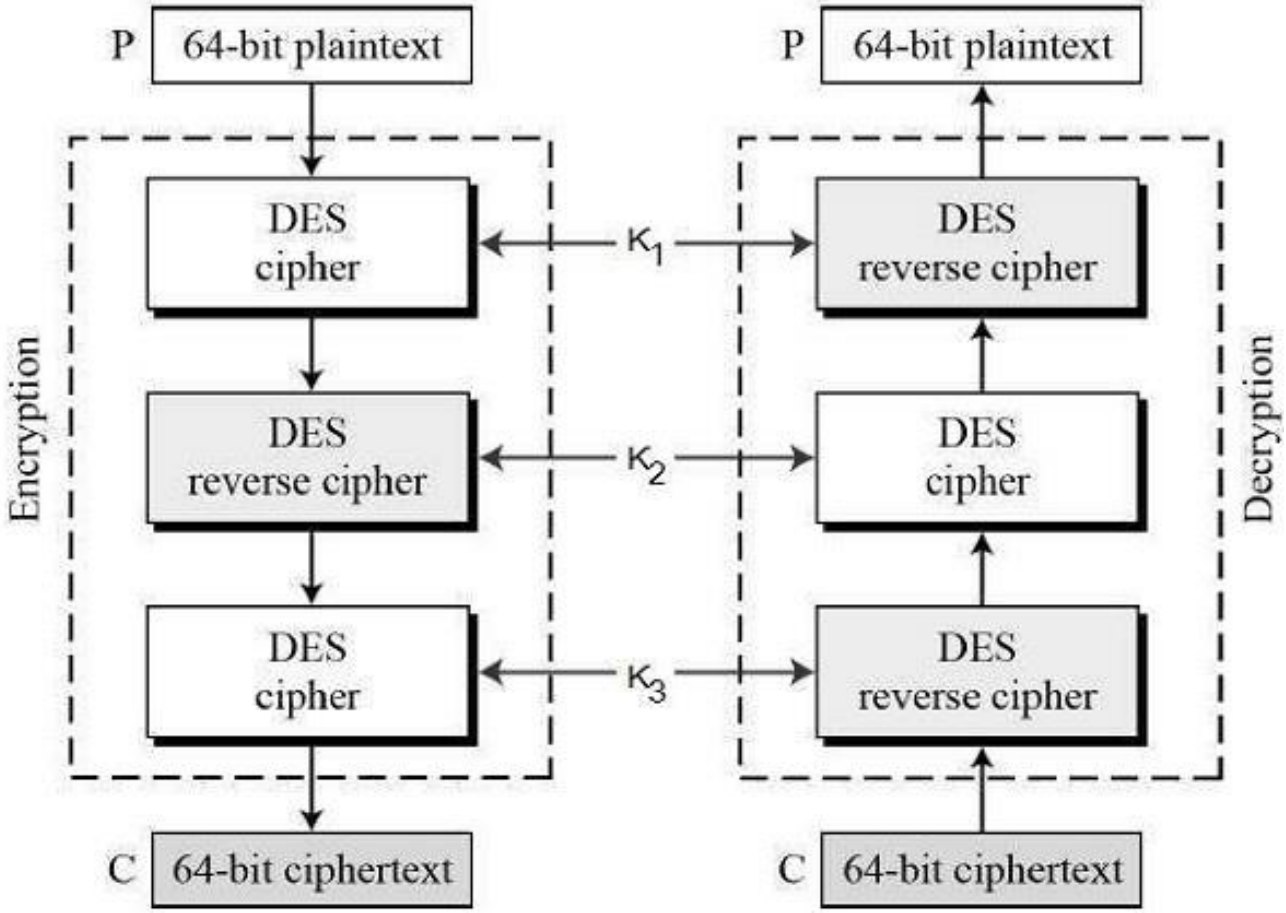
Meet-in-the-Middle Attack (MIM Attack)

- To improve the security of a block cipher, one might get the (naive) idea to simply use two independent keys to encrypt the data twice.
- In fact, an exhaustive search of all possible combinations of keys would take 2^{2n} attempts (if each key K1, K2 is n bits long), compared to the 2^{n} attempts required for searching a single key.
- The attacker can first compute E_{K1}(P) for all possible keys K1 and store the results in memory (in a lookup table).
- Afterwards he can decrypt the ciphertext by computing D_{K2}(C) for each K2.
- Any matches between these two resulting sets are likely to reveal the correct keys. (To speed up the comparison, the E_{K1}(P) set is stored in an in memory lookup table, then each D_{K2}(C) can be matched against the values in the lookup table to find the candidate keys.)
- Once the matches are discovered, they can be verified with a second testset of Plaintext and Ciphertext.

Triple DES

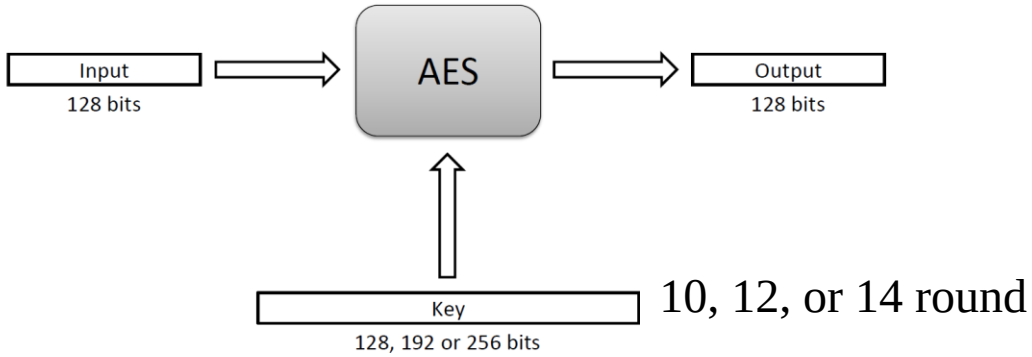
- 3DES was developed in 1999 by IBM – by a team led by Walter Tuchman. 3DES prevents a meet-in-the-middle attack. 3DES has a 168-bit key and ciphers blocks of 64 bits
- The plain text block is first encrypted with k1, then encrypted with k2 and finally with the k3
- All three keys would be different from each other
- Its three times slower than DES

Triple DES


P
64-bit plaintext
DES
cipher
DES
reverse cipher
DES
cipher
C
64-bit ciphertext
Encryption
K1
K2
K3
P
64-bit plaintext
DES
reverse cipher
DES
cipher
DES
reverse cipher
C
64-bit ciphertext
Decryption

The Advanced Encryption Standard (AES)

- Clear a replacement for DES was needed
  - ➢ have theoretical attacks that can break it
  - ➢ have demonstrated exhaustive key search attacks
- can use Triple-DES – but slow, has small blocks
- US NIST issued call for ciphers in 1997
- 15 candidates accepted in June 98
- 5 were shortlisted in Aug-99
- Rijndael was selected as the AES in Oct-2000 Nov-2001



The AES Cipher - Rijndael

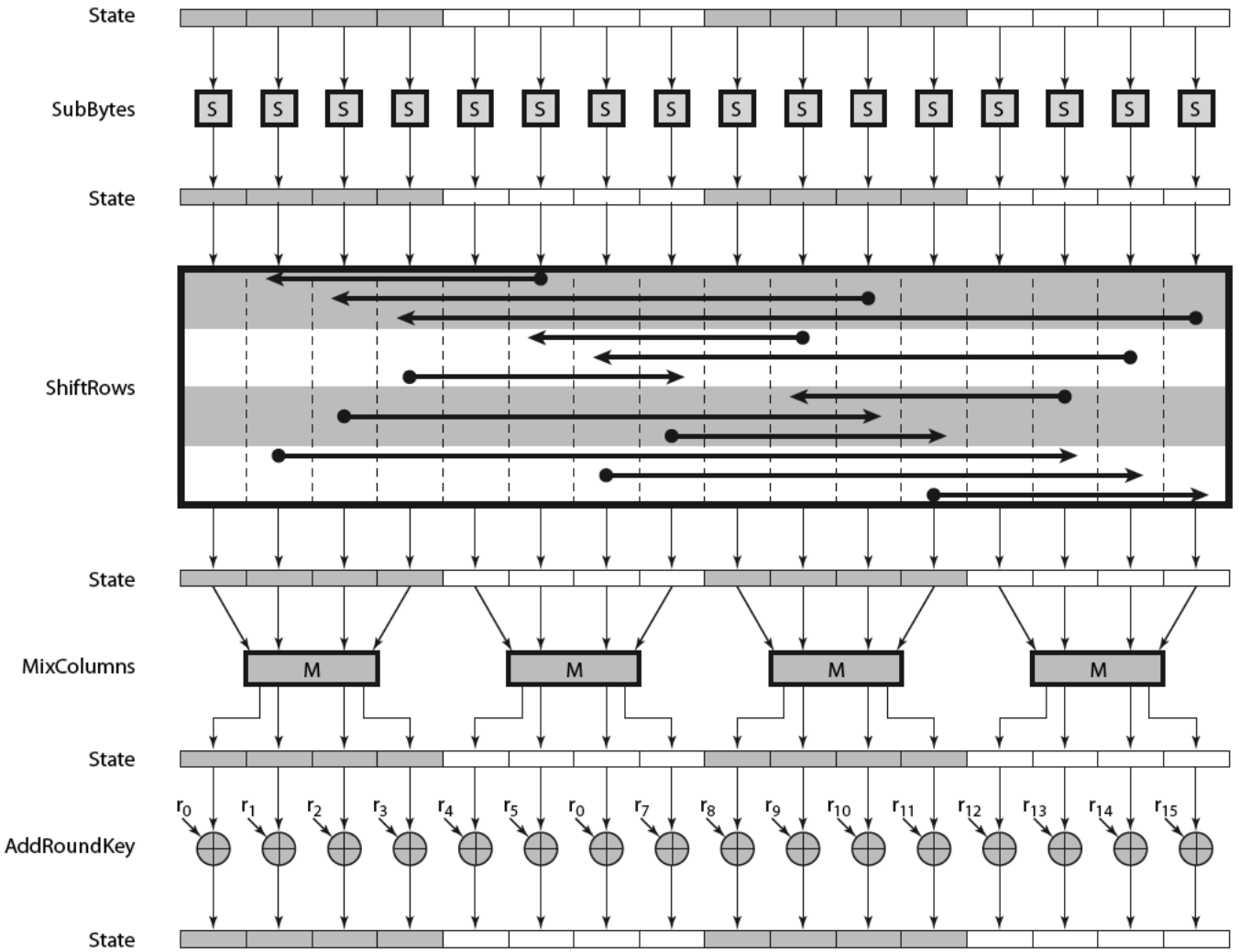
- has 128/192/256 bit keys, 128 bit data
- an **iterative** rather than **Feistel** cipher
  - processes data as block of 4 columns of each 4 bytes
  - operates on entire data block in every round
- designed to have:
  - resistance against known attacks
  - speed and code compactness on many CPUs
  - design simplicity

AES Rounds

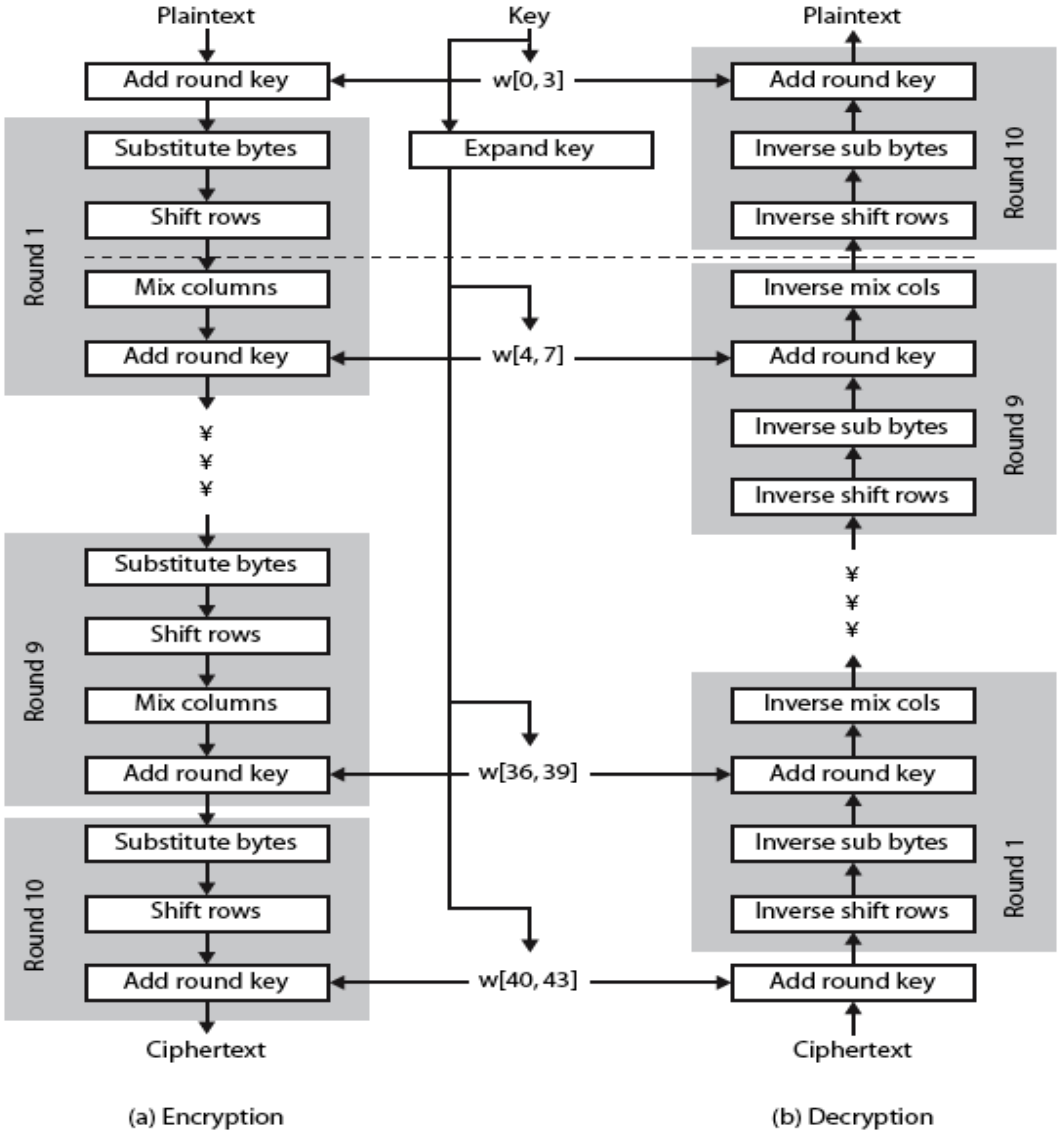
Each round is built from four basic steps:

1. SubBytes step: an S-box substitution step
2. ShiftRows step: a permutation step
3. MixColumns step: a matrix multiplication step
4. AddRoundKey step: an XOR step with a round key derived from the 128/192/256-bit encryption key

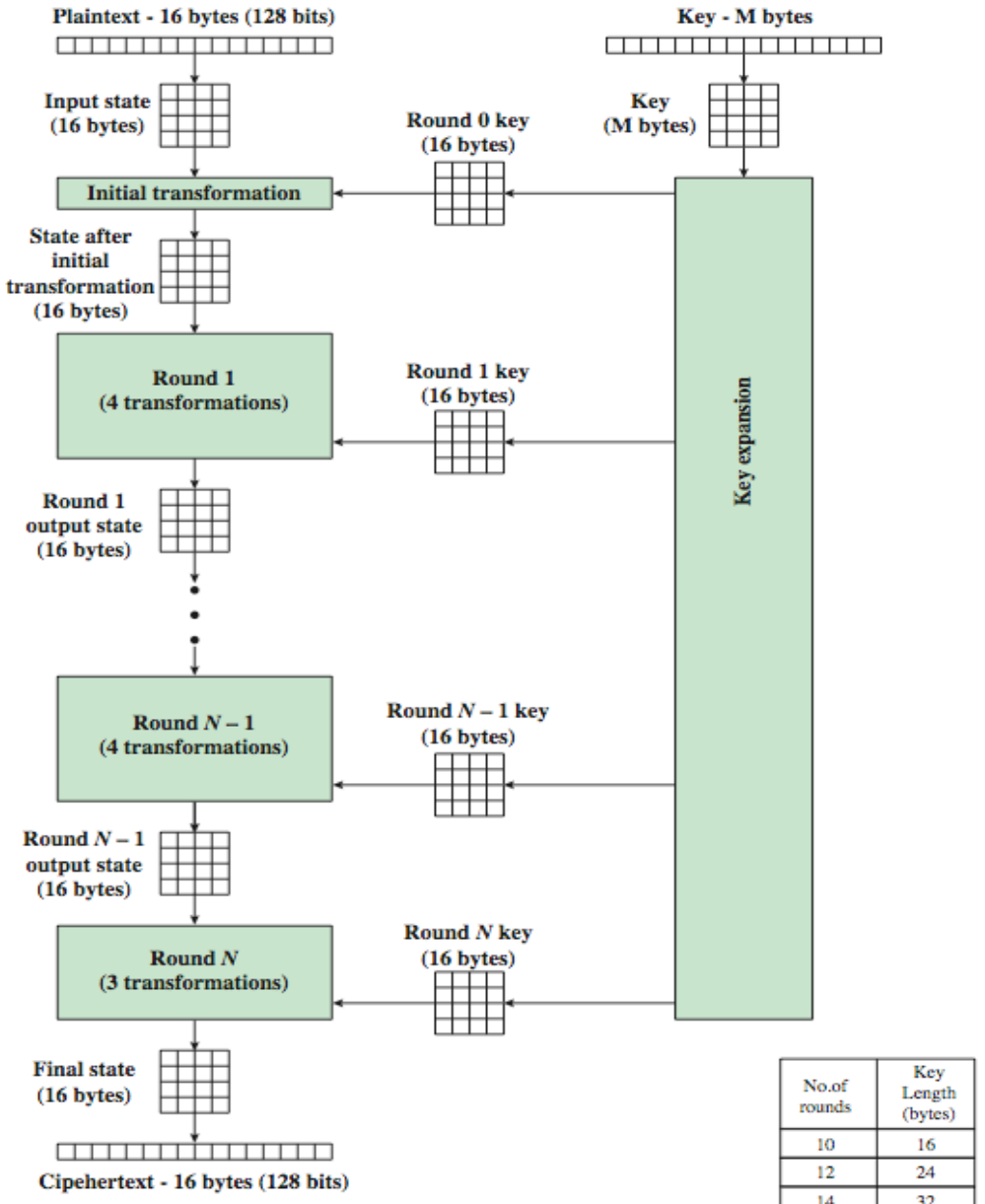
AES Round


State
SubBytes
S
State
ShiftRows
State
MixColumns
M
State
AddRoundKey
r0 r1 r2 r3 r4 r5 r0 r7 r8 r9 r10 r11 r12 r13 r14 r15
State

AES Structure

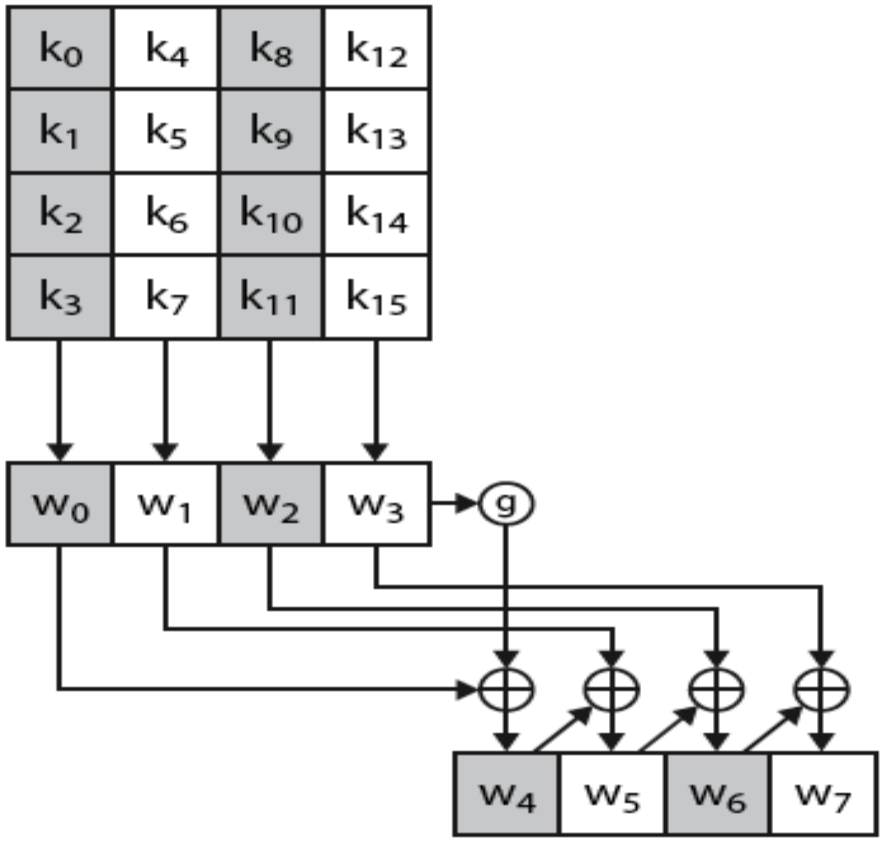


AES Encryption Process



No.of rounds	Key Length (bytes)
10	16
12	24
14	32

AES Key Expansion



Comparison table different symmetric encryption algorithms

Symmetric Key Algorithm	Structure	Key Size (bits)	No of Rounds	Block Size (bits)	Security	Speed
DES	Feistel	56	16	64	Already Broken	Slow
3 DES	Feistel	112, 168	48	64	Adequate	Very Slow
AES	Substitution/ Transposition	128, 192, 256	10, 12, 14	128	Excellent	Fast
Blowfish	Feistel	32-448	16	64	Excellent	Fast

How to use a block cipher?

- Block ciphers encrypt fixed-size blocks
  - e.g. DES encrypts 64-bit blocks
- We need some way to encrypt a message of arbitrary length
  - e.g. a message of 1000 bytes
- NIST defines several ways to do it
  - called **modes of operation**

Modes of Operation

- Block ciphers encrypt fixed size blocks
  - eg. DES encrypts 64-bit blocks, with 56-bit key

- Need way to use in practise, given usually have arbitrary amount of information to encrypt
  - Partition message into separate block for ciphering

- A mode of operation describes the process of encrypting each of these blocks under a single key

Some modes may use randomized addition input value

Five Modes of Operation

- Electronic codebook mode (ECB)
- Cipher block chaining mode (CBC) – most popular
- Cipher feedback mode (CFB)

Electronic Code Book (ECB)

The plaintext is broken into blocks, P_1, P_2, P_3, ...

Each block is encrypted independently:

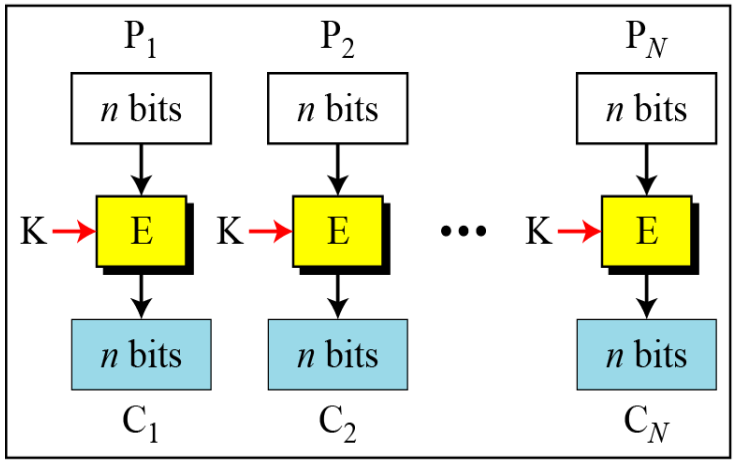
$$C_i = E_K(P_i)$$

For a given key, this mode behaves like we have a gigantic codebook, in which each plaintext block has an entry, hence the name Electronic Code Book

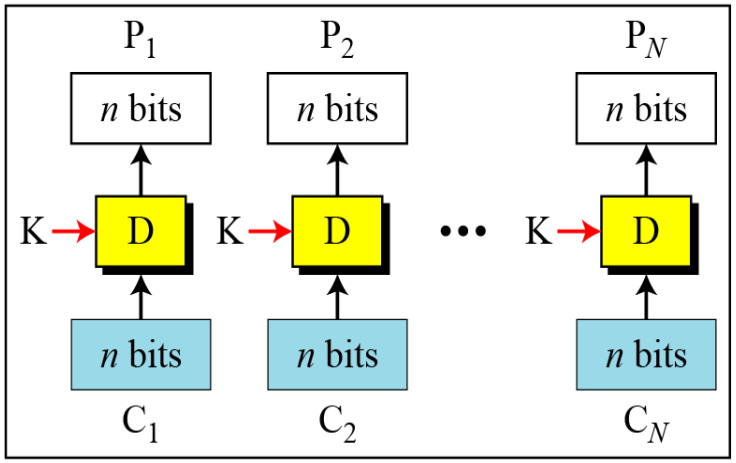
ECB Scheme

Encryption: $C_i = E_K (P_i)$ Decryption: $P_i = D_K (C_i)$

E: Encryption D: Decryption
P_i: Plaintext block i C_i: Ciphertext block i
K: Secret key



Encryption



Decryption

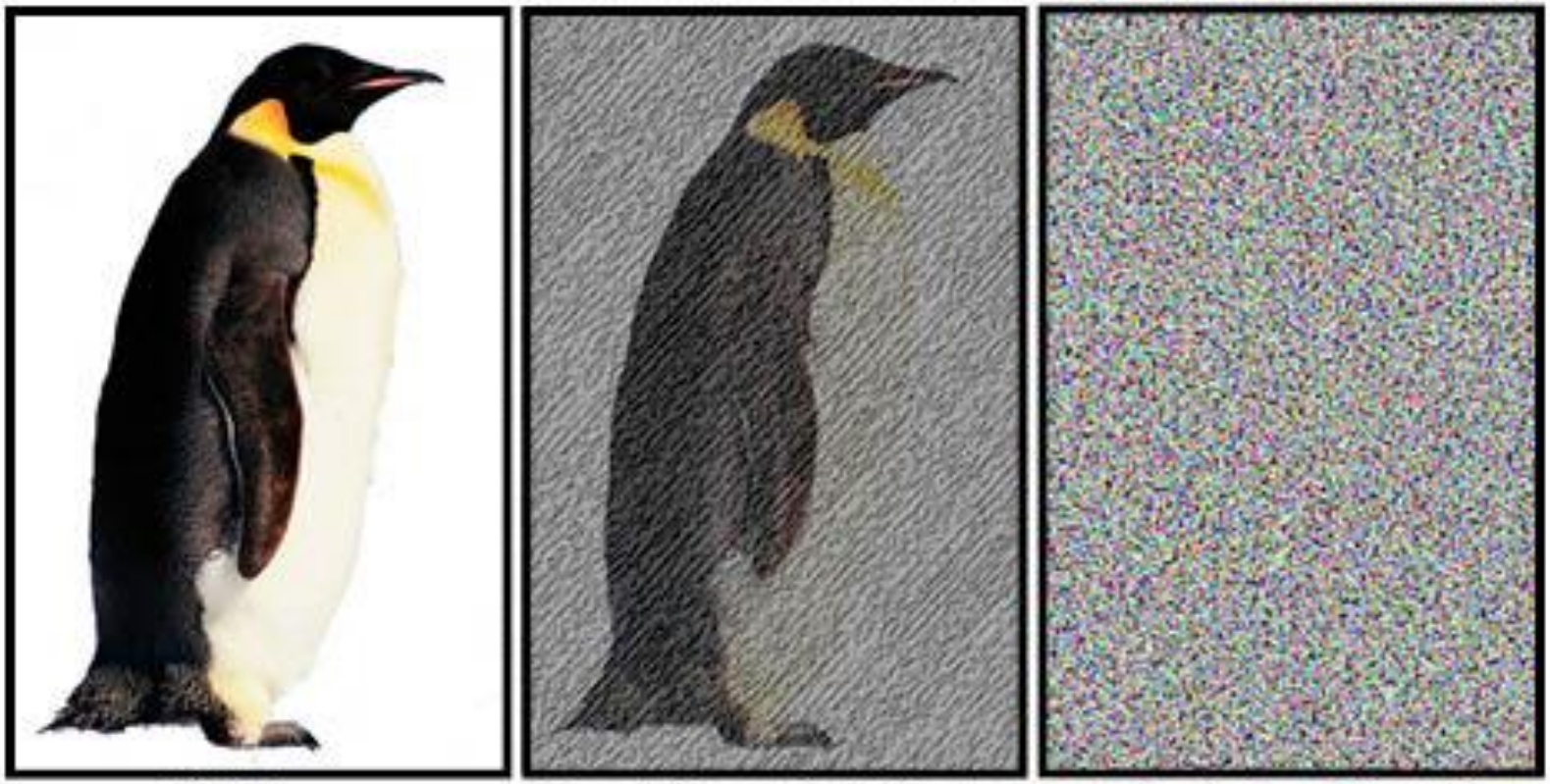
Remarks on ECB

- Strength: it's simple.

- Weakness:
  - Repetitive information contained in the plaintext may show in the ciphertext, if aligned with blocks.
  - If the same message (e.g., an SSN) is encrypted (with the same key) and sent twice, their ciphertexts are the same.

- Typical application: secure transmission of short pieces of information (e.g. a temporary encryption key)

Example of ECB



Original Image

Encrypted image using ECB mode

Modes other than ECB result in pseudorandomness

Cipher Block Chaining (CBC)

- Solve security deficiencies in ECB
  - Repeated same plaintext block result different ciphertext block

- Each previous cipher blocks is chained to be input with current plaintext block, hence name

- Use Initial Vector (IV) to start process

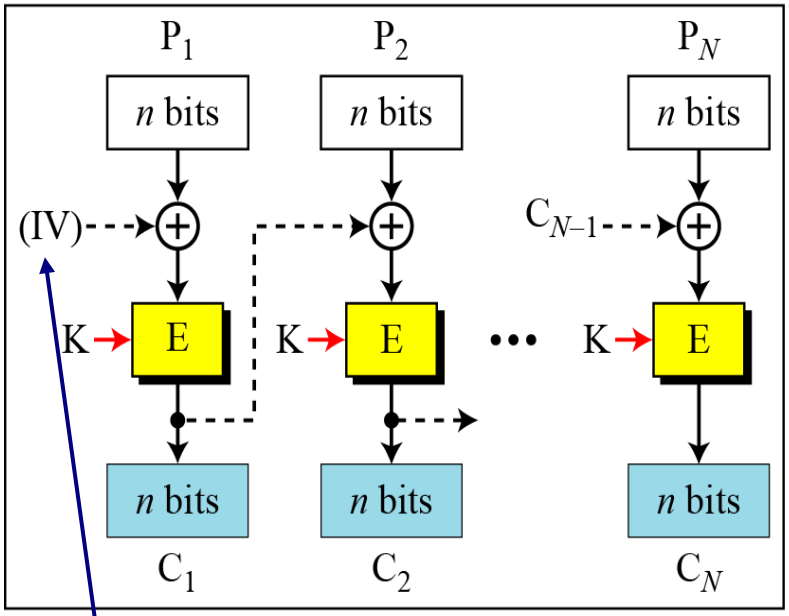
$$C_i = E_K (P_i \text{ XOR } C_{i-1})$$
$$C_0 = IV$$

Uses: bulk data encryption, authentication

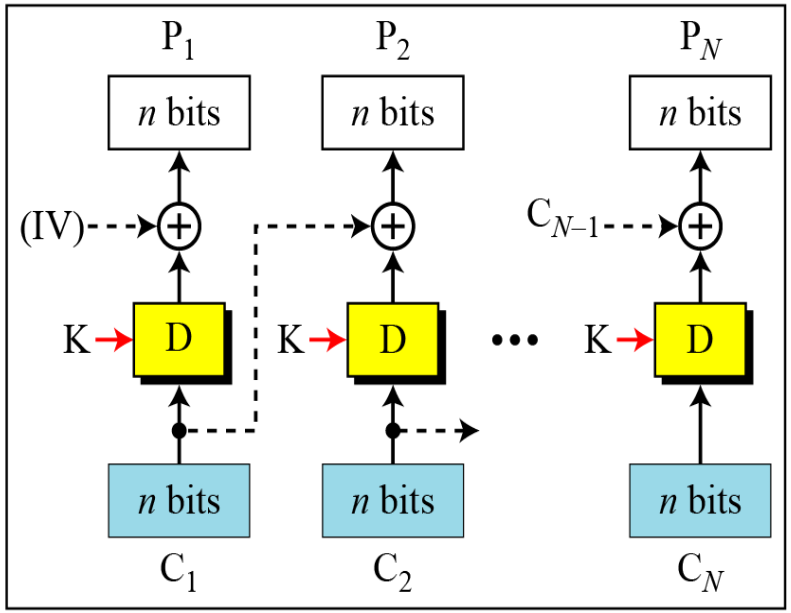
CBC scheme

E: Encryption
P_i: Plaintext block i
K: Secret key

D : Decryption
C_i : Ciphertext block i
IV: Initial vector (C_0)



Encryption



Decryption

Encryption:
$C_0 = IV$
$C_i = E_K (P_i \oplus C_{i-1})$

Decryption:
$C_0 = IV$
$P_i = D_K (C_i) \oplus C_{i-1}$

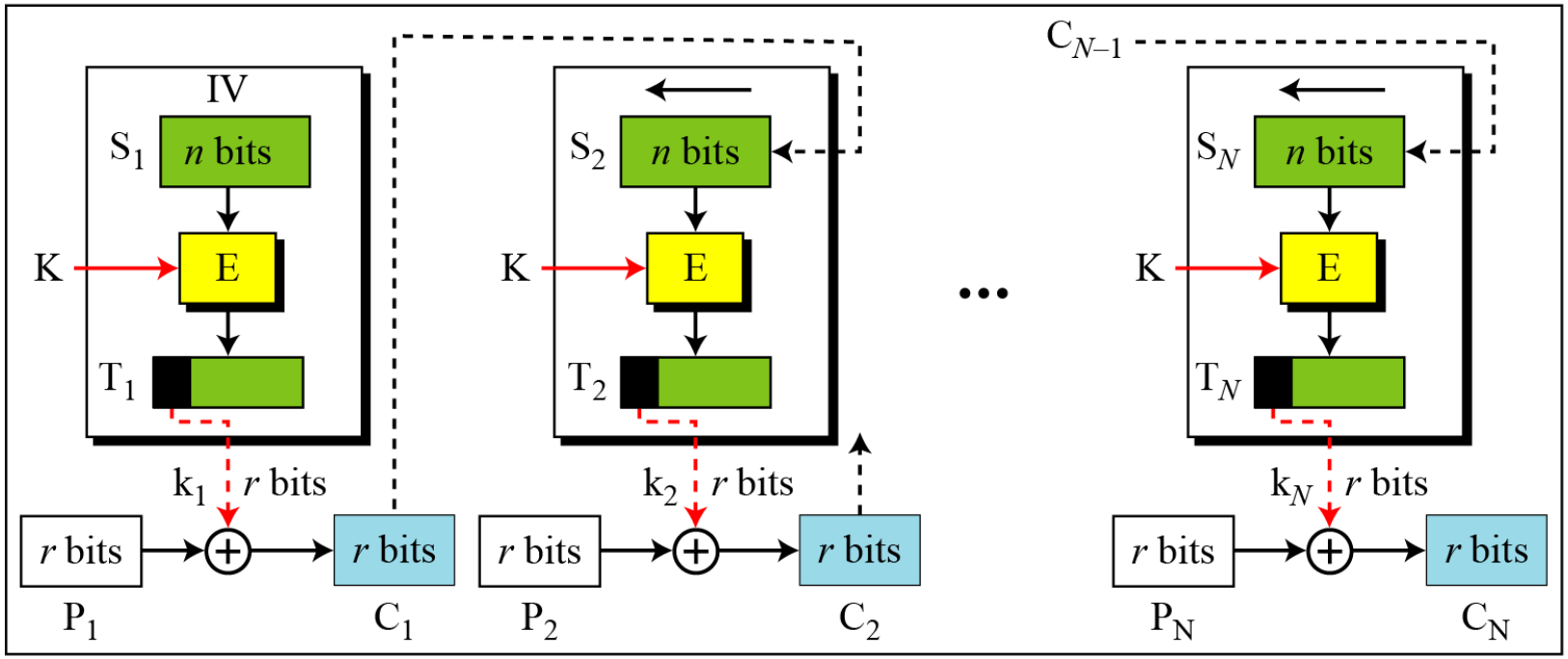
Cipher feedback mode (CFB) Scheme

Encryption: $C_i = P_i \oplus \text{SelectLeft}_r \{E_K [\text{ShiftLeft}_r (S_{i-1}) \mid C_{i-1})]\}$
Decryption: $P_i = C_i \oplus \text{SelectLeft}_r \{E_K [\text{ShiftLeft}_r (S_{i-1}) \mid C_{i-1})]\}$

E : Encryption
P_i: Plaintext block *i*
K: Secret key

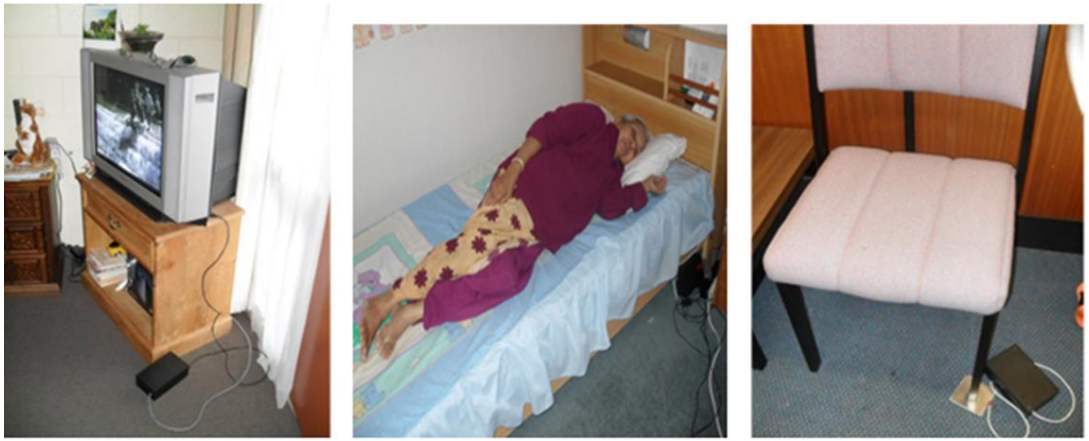
D : Decryption
C_i: Ciphertext block *i*
IV: Initial vector (S_1)

S_i: Shift register
T_i: Temporary register



Encryption

Example: Designing a Secure Smart Home



References

1. http://www.sfu.ca/~ljilja/ENSC427/News/Kurose_Ross/Chapter_8_V7.0_Accessible.pdf
2. Cryptography and Network Security: Principles and Practice 0133354695, 9780133354690.
3. A.K. Dewdney, The New Turning Omnibus, pp. 250-257, Henry Holt and Company, 2001.

Lnu.se