National Chung Cheng University
Department of Information Management

Block Ciphers and the Data Encryption Standard

Pei-Ju (Julian) Lee

National Chung Cheng University
Information Security
pjlee@mis.ccu.edu.tw

Fall, 2016

Symmetric Cipher

- Stream cipher
- Block cipher
  - Data Encryption Standard (DES)

Stream Cipher

Encrypts a digital data stream one bit or one byte at a time

Examples:

- Autokeyed Vigenère cipher
- Vernam cipher

In the ideal case a one-time pad version of the Vernam cipher would be used, in which the keystream is as long as the plaintext bit stream

If the cryptographic keystream is random, then this cipher is unbreakable by any means other than acquiring the keystream

- Keystream must be provided to both users in advance via some independent and secure channel
- This introduces insurmountable logistical problems if the intended data traffic is very large

For practical reasons the bit-stream generator must be implemented as an algorithmic procedure so that the cryptographic bit stream can be produced by both users

It must be computationally impractical to predict future portions of the bit stream based on previous portions of the bit stream

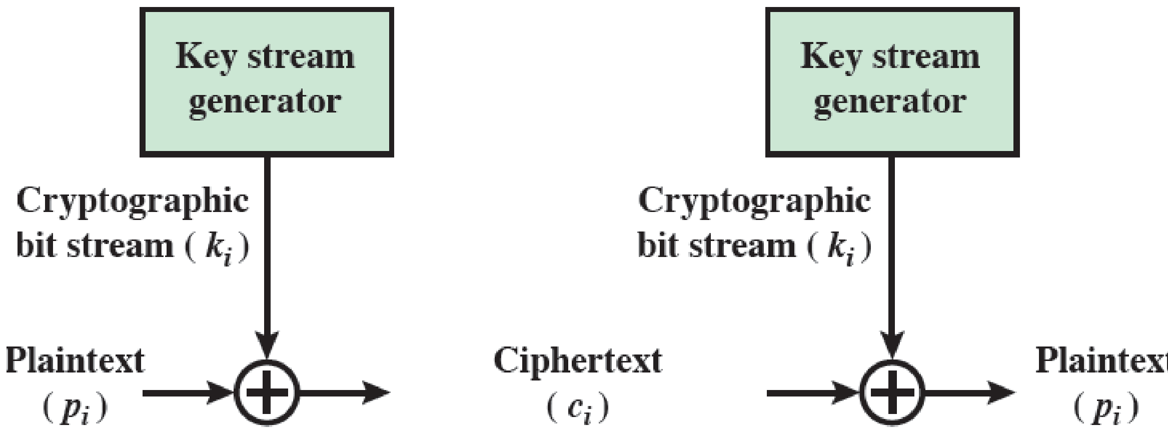
The two users need only share the generating key and each can produce the keystream

Recall: Vernam Cipher

p_i = ith binary digit of plaintext
k_i = ith binary digit of key
c_i = ith binary digit of ciphertext
$\oplus$ = exclusive-or (XOR) operation

$$c_i = p_i \oplus k_i$$

$$p_i = c_i \oplus k_i$$



- Vernam cipher works on binary data (bits)
- The essence of this technique is the means of construction of the **key**

Block Cipher

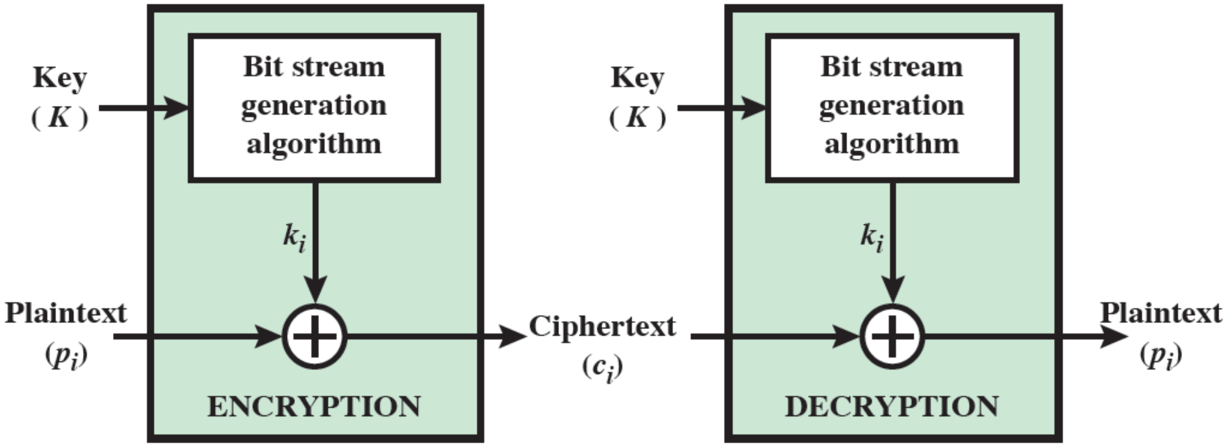
A block of plaintext is treated as a whole and used to produce a ciphertext block of equal length

Typically a block size of 64 or 128 bits is used

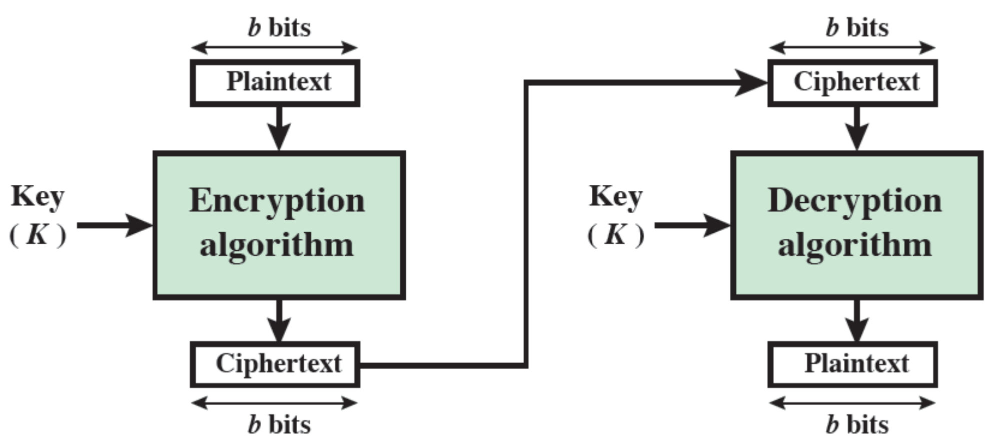
As with a stream cipher, the two users share a symmetric encryption key

The majority of network-based symmetric cryptographic applications make use of block ciphers

Symmetric Encryption



(a) Stream Cipher Using Algorithmic Bit Stream Generator



(b) Block Cipher

Block Cypher

- *n* bits plaintext block to produce *n* bits ciphertext block
  - Possible different plaintext block: 2^n
  - It has to produce a unique ciphertext block for the encryption to be **reversible** (i.e., for decryption to be possible) or **nonsingular**

Reversible Mapping

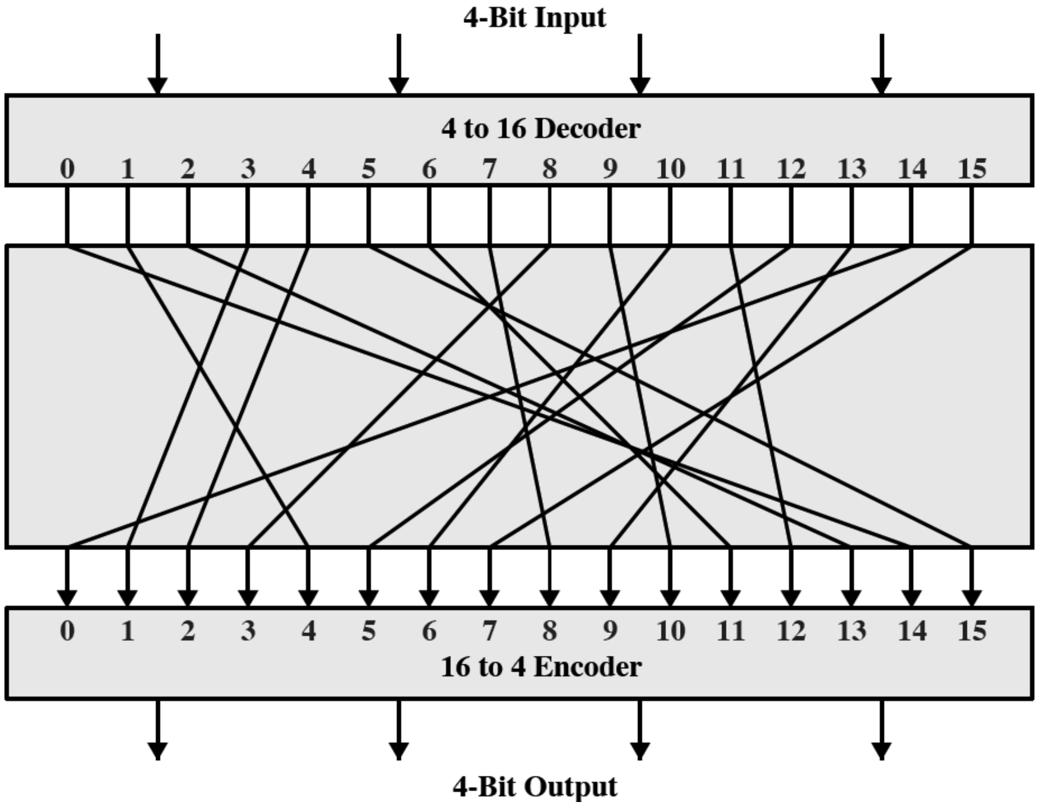
Plaintext	Ciphertext
00	11
01	10
10	00
11	01

Irreversible Mapping

Plaintext	Ciphertext
00	11
01	10
10	01
11	01

n-bit-*n*-bit Block Substitution (*n*=4)

- Example:
  4-bit input produce one of 16 (=2^4) possible input states, which is mapped by the substitution cipher into a unique one of 16 possible output states, each of which is represented by 4 ciphertext bits



Encryption and Decryption Mappings

- Reversible mapping
  - Problem: For small block size, the system is equivalent to a classical substitution cipher
    =>vulnerable to a statistical analysis.
    If n is sufficient large=>this type of cryptanalysis is infeasible.

- An arbitrary reversible substitution cipher (the ideal block cipher)(n is sufficient large) for a large block size is not practical
  - key length: (4bits)*(16 rows)=64 bits
  - For an *n*-bit ideal block cipher, the length of the key defined in this fashion is $n*(2^n)$ bits

Plaintext	Ciphertext
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

Ciphertext	Plaintext
0000	1110
0001	0011
0010	0100
0011	1000
0100	0001
0101	1100
0110	1010
0111	1111
1000	0111
1001	1101
1010	1001
1011	0110
1100	1011
1101	0010
1110	0000
1111	0101

General Block substitution example

- Consider the difficulties:
  - The mapping itself constitutes the key
  - Larger *n* to make cryptanalysis infeasible
  - Easy realization
  - One solution: use linear equations for the mapping
    - Ex. n=4 Fesitel cipher

x_i: four binary digits of the *P* block,
y_i: four binary digits of the *C* block,
k_{ij}: binary coefficients(mod 2)
n^2: size of the key

$$
\begin{aligned}
y_1 &= k_{11}x_1 + k_{12}x_2 + k_{13}x_3 + k_{14}x_4 \\
y_2 &= k_{21}x_1 + k_{22}x_2 + k_{23}x_3 + k_{24}x_4 \\
y_3 &= k_{31}x_1 + k_{32}x_2 + k_{33}x_3 + k_{34}x_4 \\
y_4 &= k_{41}x_1 + k_{42}x_2 + k_{43}x_3 + k_{44}x_4
\end{aligned}
$$

Problem: vulnerable by an attacker who is aware of the structure of this algorithm

Feistel Cipher

- Feistel: approximate the ideal block cipher by utilizing the concept of a product cipher, which is the execution of two or more simple ciphers in sequence
- Proposed the use of a cipher that alternates substitutions and permutation

Substitutions	• Each plaintext element or group of elements is uniquely replaced by a corresponding ciphertext element or group of elements
Permutation	• No elements are added or deleted or replaced in the sequence, rather the order in which the elements appear in the sequence is changed

- Is a practical application of a proposal by Claude Shannon to develop a product cipher that alternates confusion and diffusion functions

- Feistel's is a practical application of a proposal by Claude Shannon to develop a product cipher that alternates confusion and diffusion functions

Diffusion and Confusion

- Shannon's concern was to thwart cryptanalysis based on statistical analysis (for any cryptographic system)
  - In a readable language, the frequency distribution may be known
  - If these statistics are reflected in *C*
  - The encryption key may be deduced

Diffusion

- The statistical structure of the plaintext is dissipated into long-range statistics of the ciphertext
- This is achieved by having each plaintext digit affect the value of many ciphertext digits(Example)

Confusion

- Seeks to make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible
- Even if the attacker can get some handle on the statistics of the ciphertext, the way in which the key was used to produce that ciphertext is so complex as to make it difficult to deduce the key

Example of Diffusion

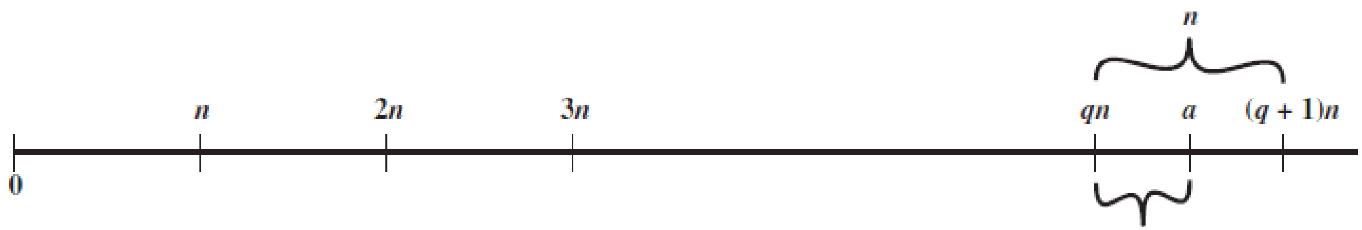
- Having each plaintext digit affect the value of many ciphertext digits
- Ex. *M*: plaintext, *Y*: ciphertext

$$y_n = \left(\sum_{i=1}^{k} m_{n+i}\right) \bmod 26$$

Y	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
M	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51
	…	…	…																							

Modular arithmetic (CH. 4)

- If *a* is an integer and *n* is a positive integer, we define *a mod n* to be the remainder when *a* is divided by *n*. We get an integer quotient *q* and an integer remainder/residue *r*. The integer *n* is called the *modulus*. $\lfloor x \rfloor$ is the largest integer less than or equal to *x*.



(a) General relationship

- Ex.
  - 11 mod 7 = 4
  - 4 mod 7 = 4
  - 74 mod 7 = 4

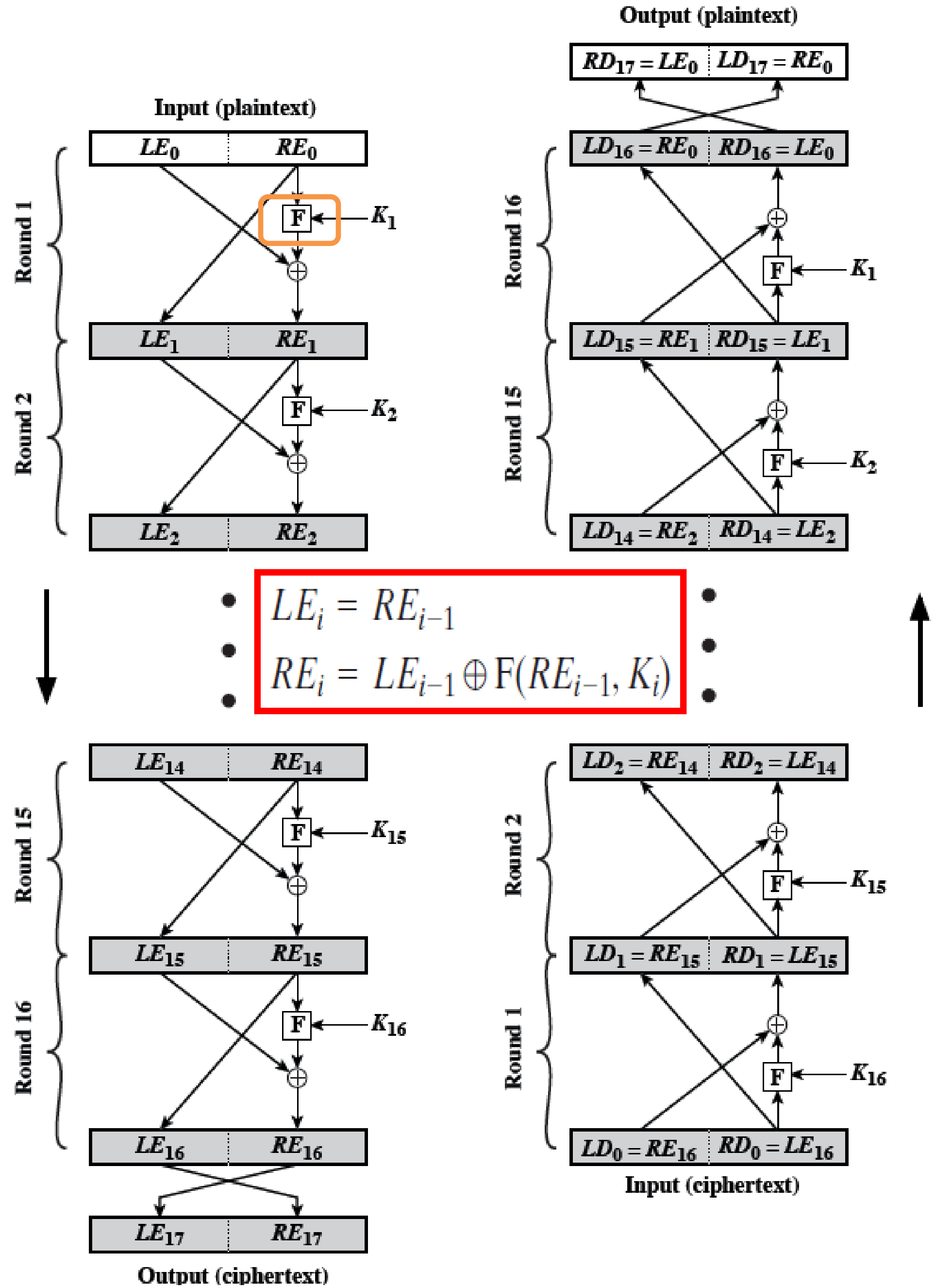
$$\boxed{a = qn + r \qquad 0 \leq r < n;\ q = \lfloor a/n \rfloor}$$

r: remainder/residue
q: quotient

Feistel Cipher Structure

- A *P* block of length *2w* bits (divided into L_0, R_0)and a key *K* -> pass through *n* rounds of round function (F) processing ->combine to *C* block
- Subkeys K_i are different from *K*
- Shannon: Substitution-permutation network (SPN)

- A substitution is performed on the left half of the data. This is done by applying a round function *F* to the right half of the data and then taking the exclusive-OR of the output of that function and the left half of the data. The round function has the same general structure for each round but is parameterized by the round subkey K_i .
- Permutation is performed of the two halves of the data



Feistel Decryption Algorithm

$LE_i \| RE_i$ (LE_i concatenated with RE_i)

the $(16 - i)$th decryption round is $RE_i \| LE_i$ or, equivalently, $LD_{16-i} \| RD_{16-i}$.

$$\begin{aligned} LD_1 &= RD_0 = LE_{16} = RE_{15} \\ RD_1 &= LD_0 \oplus \mathrm{F}(RD_0, K_{16}) \\ &= RE_{16} \oplus \mathrm{F}(RE_{15}, K_{16}) \\ &= [LE_{15} \oplus \mathrm{F}(RE_{15}, K_{16})] \oplus \mathrm{F}(RE_{15}, K_{16}) \end{aligned}$$

$$[A \oplus B] \oplus C = A \oplus [B \oplus C]$$

$$D \oplus D = 0$$

$$E \oplus 0 = E$$

$$LD_1 = RE_{15} \text{ and } RD_1 = LE_{15}$$

$$RE_{15} \| LE_{15}$$

For the ith iteration of the encryption algorithm,

$$\begin{aligned} LE_i &= RE_{i-1} \\ RE_i &= LE_{i-1} \oplus \mathrm{F}(RE_{i-1}, K_i) \end{aligned}$$

Rearranging terms:

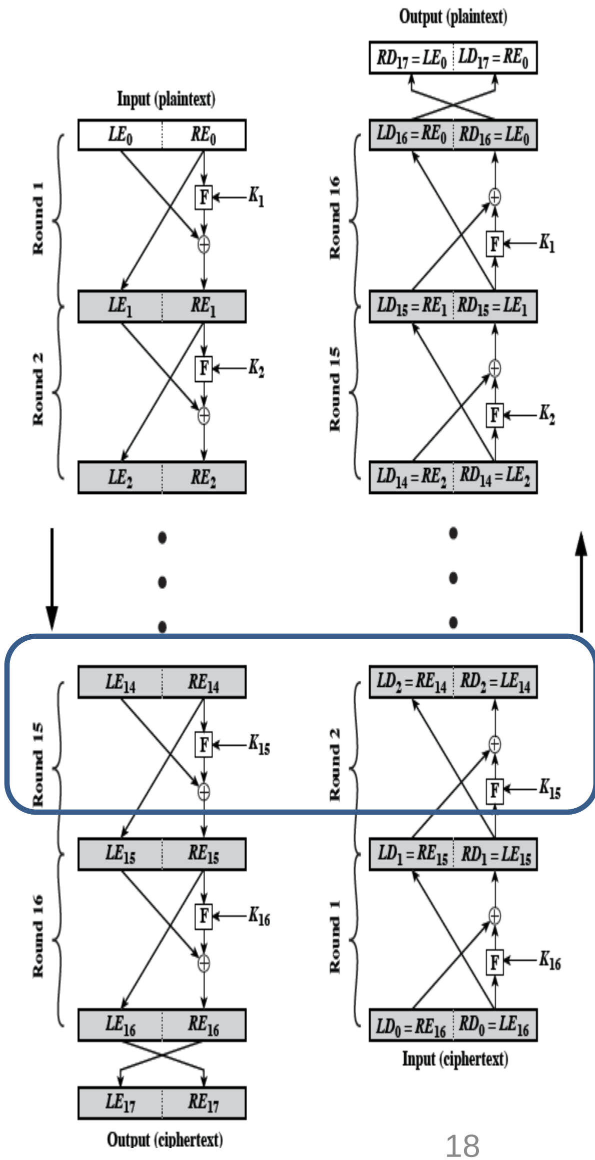
$$\begin{aligned} RE_{i-1} &= LE_i \\ LE_{i-1} &= RE_i \oplus \mathrm{F}(RE_{i-1}, K_i) = RE_i \oplus \mathrm{F}(LE_i, K_i) \end{aligned}$$

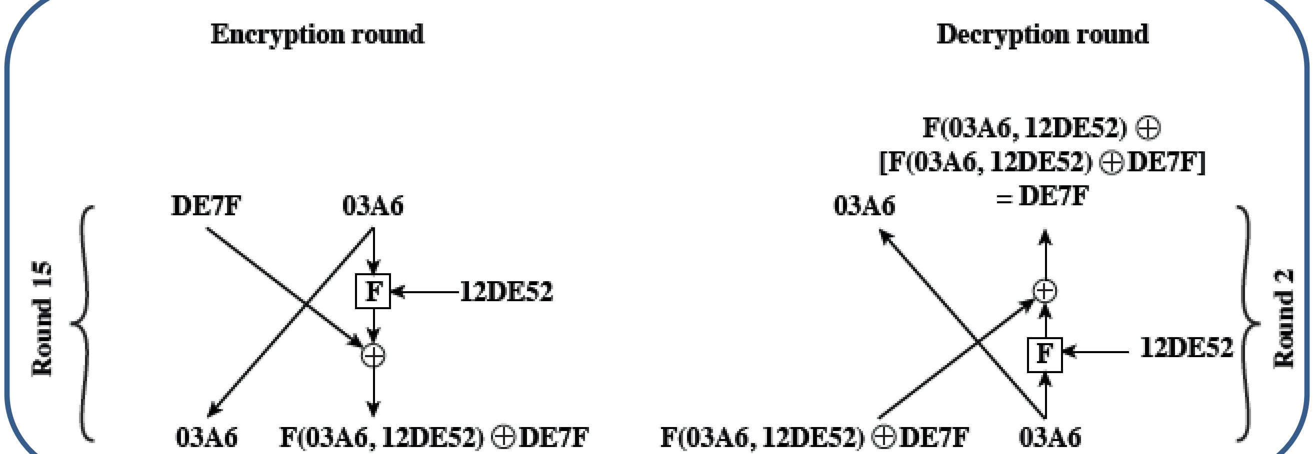
Feistel Cipher Design Depends on

- Block size
  - Larger block sizes mean greater security but reduced encryption/decryption speed for a given algorithm
- Key size
  - Larger key size means greater security but may decrease encryption/decryption speeds
- Number of rounds
  - The essence of the Feistel cipher is that a single round offers inadequate security but that multiple rounds offer increasing security
- Subkey generation algorithm
  - Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis
- Round function F
  - Greater complexity generally means greater resistance to cryptanalysis
- Fast software encryption/decryption
  - In many cases, encrypting is embedded in applications or utility functions in such a way as to preclude a hardware implementation; accordingly, the speed of execution of the algorithm becomes a concern
- Ease of analysis
  - If the algorithm can be concisely and clearly explained, it is easier to analyze that algorithm for cryptanalytic vulnerabilities and therefore develop a higher level of assurance as to its strength

Feistel Example

- The 15th round of encryption, corresponding to the 2nd round of decryption
- Block size: 32 bits (two 16 bits halves)
- Key size: 24 bits
- At the end of 14th encryption: DE7F03A6 (hexadecimal). LE_{14}=DE7F, RE_{14}=03A6, K_{15}=12DE52
- At the end of 1st decryption



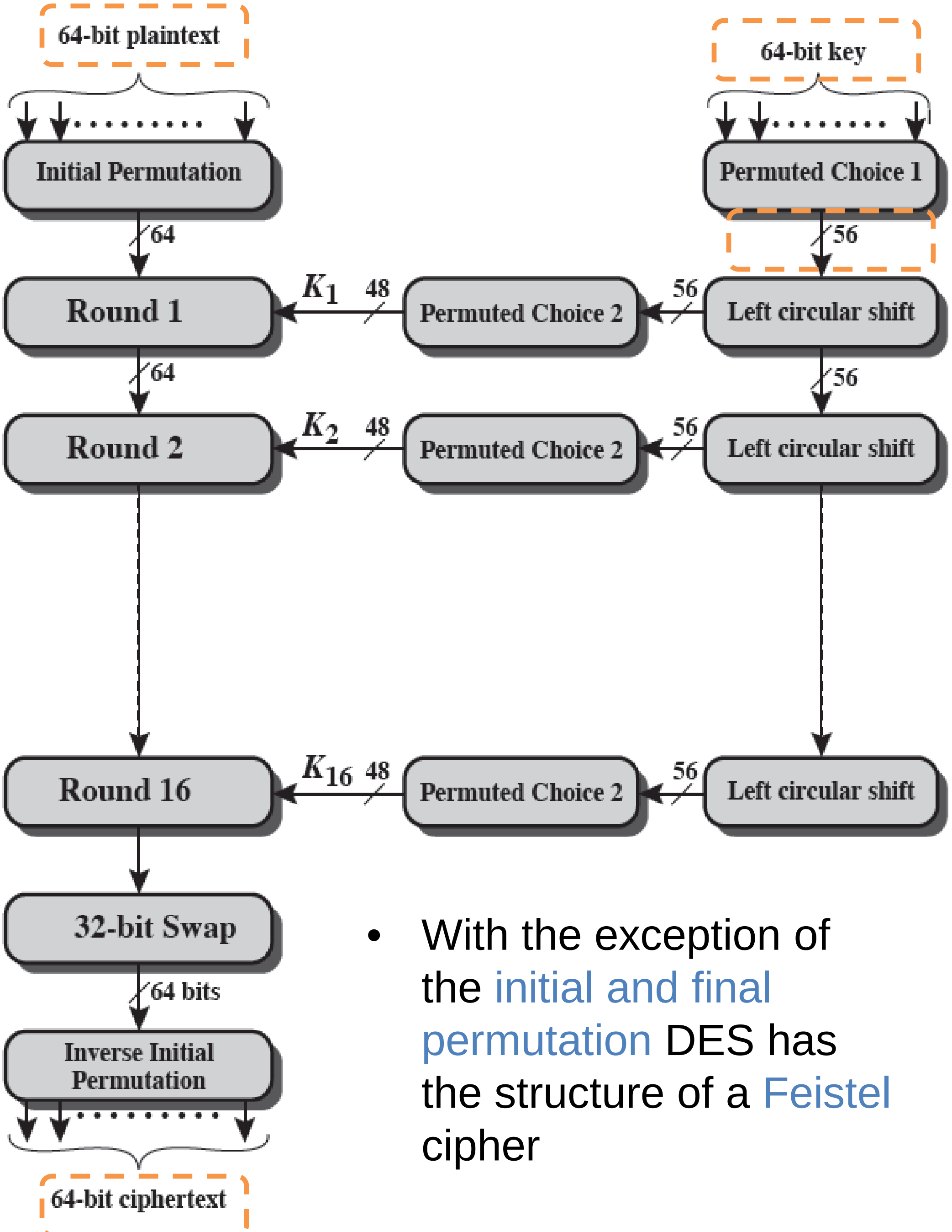


Data Encryption Standard (DES)

- Issued in 1977 by the National Bureau of Standards (now NIST) as Federal Information Processing Standard 46
- Was the most widely used encryption scheme until the introduction of the Advanced Encryption Standard (AES) in 2001
- Algorithm itself is referred to as the Data Encryption Algorithm (DEA)
  - Data are encrypted in 64-bit blocks using a 56-bit key into a 64-bit output
  - The same steps, with the same key, are used to reverse the encryption

DES Encryption Algorithm

- Two inputs: the plaintext, the key
- Three phases of *P* processing
  - Initial permutation (IP): rearrange the bits to produce the permuted input
  - 16 rounds of the same function, which involves permutation and substitution
  - At the last round, swap the left and right halves to produce the preoutput. Then pass through a permutation [IP^{-1}] which is the inverse of the IP function, to produce the 64 bit *C*



- With the exception of the initial and final permutation DES has the structure of a Feistel cipher

Single round of DES

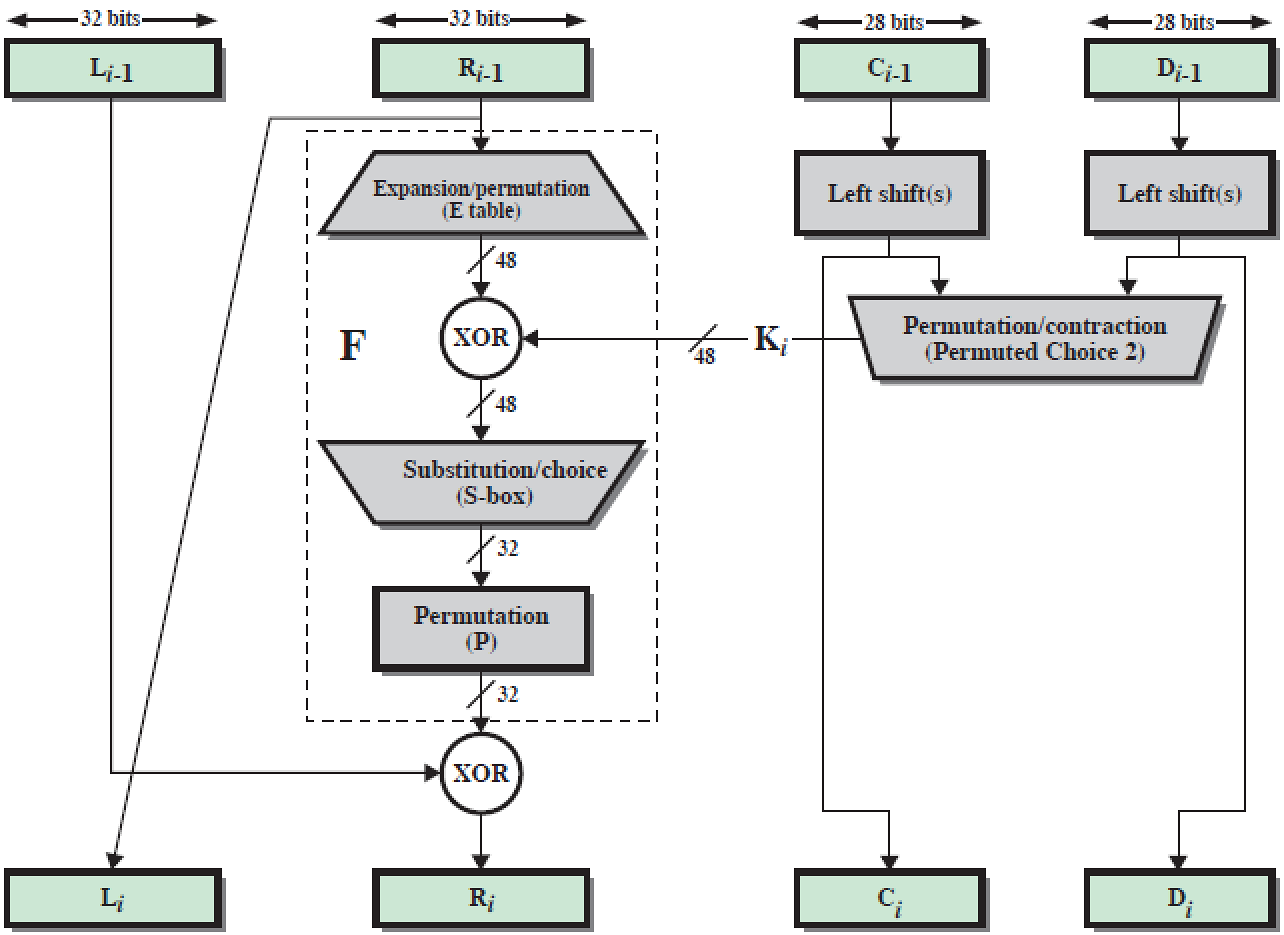


Figure S.2 Single Round of DES Algorithm

Explain later

DES Example

The original plaintext is hexadecimal palindrome

Plaintext:	02468aceeca86420
Key:	0f1571c947d9e859
Ciphertext:	da02ce3a89ecac3b

Combine the final row of left/right data after IP^{-1} form the ciphertext

Roud IP: 32-bit values of the left/right data after IP

Round	K_i	L_i	R_i
IP		5a005a00	3cf03c0f
1	1e030f03080d2930	3cf03c0f	bad22845
2	0a31293432242318	bad22845	99e9b723
3	23072318201d0c1d	99e9b723	0bae3b9e
4	05261d3824311a20	0bae3b9e	42415649
5	3325340136002c25	42415649	18b3fa41
6	123a2d0d04262a1c	18b3fa41	9616fe23
7	021f120b1c130611	9616fe23	67117cf2
8	1c10372a2832002b	67117cf2	c11bfc09
9	04292a380c341f03	c11bfc09	887fbc6c
10	2703212607280403	887fbc6c	600f7e8b
11	2826390c31261504	600f7e8b	f596506e
12	12071c241a0a0f08	f596506e	738538b8
13	300935393c0d100b	738538b8	c6a62c4e
14	311e09231321182a	c6a62c4e	56b0bd75
15	283d3e0227072528	56b0bd75	75e8fd8f
16	2921080b13143025	75e8fd8f	25896490
IP^{-1}		da02ce3a	89ecac3b

Avalanche Effect in DES

- **Change in Plaintext**

Round		δ	Round		δ
Original:	02468aceeca86420	1	9	c11bfc09887fbc6c	32
Changed:	12468aceeca86420			99f911532eed7d94	
1	3cf03c0fbad22845	1	10	887fbc6c600f7e8b	34
	3cf03c0fbad32845			2eed7d94d0f23094	
2	bad2284599e9b723	5	11	600f7e8bf596506e	37
	bad3284539a9b7a3			d0f23094455da9c4	
3	99e9b7230bae3b9e	18	12	f596506e738538b8	31
	39a9b7a3171cb8b3			455da9c47f6e3cf3	
4	0bae3b9e42415649	34	13	738538b8c6a62c4e	29
	171cb8b3ccaca55e			7f6e3cf34bc1a8d9	
5	4241564918b3fa41	37	14	c6a62c4e56b0bd75	33
	ccaca55ed16c3653			4bc1a8d91e07d409	
6	18b3fa419616fe23	33	15	56b0bd7575e8fd8f	31
	d16c3653cf402c68			1e07d4091ce2e6dc	
7	9616fe2367117cf2	32	16	75e8fd8f25896490	32
	cf402c682b2cefbc			1ce2e6dc365e5f59	
8	67117cf2c11bfc09	33	IP^{-1}	da02ce3a89ecac3b	32
	2b2cefbc99f91153			057cde97d7683f2a	

-The number of bit different

- Avalanche Effect : a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext.

Cont'd

- **Change in key**

- The original key, 0f1571c947d9e85
- The altered key, 1f1571c947d9e859

Round		δ	Round		δ
	02468aceeca86420 02468aceeca86420	0	9	c11bfc09887fbc6c 548f1de471f64dfd	34
1	3cf03c0fbad22845 3cf03c0f9ad628c5	3	10	887fbc6c600f7e8b 71f64dfd4279876c	36
2	bad2284599e9b723 9ad628c59939136b	11	11	600f7e8bf596506e 4279876c399fdc0d	32
3	99e9b7230bae3b9e 9939136b768067b7	25	12	f596506e738538b8 399fdc0d6d208dbb	28
4	0bae3b9e42415649 768067b75a8807c5	29	13	738538b8c6a62c4e 6d208dbbb9bdeeaa	33
5	4241564918b3fa41 5a8807c5488dbe94	26	14	c6a62c4e56b0bd75 b9bdeeaad2c3a56f	30
6	18b3fa419616fe23 488dbe94aba7fe53	26	15	56b0bd7575e8fd8f d2c3a56f2765c1fb	33
7	9616fe2367117cf2 aba7fe53177d21e4	27	16	75e8fd8f25896490 2765c1fb01263dc4	30
8	67117cf2c11bfc09 177d21e4548f1de4	32	IP^{-1}	da02ce3a89ecac3b ee92b50606b62b0b	30

Concerns of DES

- **1. The key size**
  - The use of 56-bit keys
    - With a key length of 56 bits, there are 2^{56} possible keys, which is approximately 7.2×10^{16} keys
      (Assuming that, on average, half the key space has to be searched, a single machine performing one DES encryption per microsecond would take more than a thousand years to break the cipher)(1 encryption per microsecond)
  - But:
    with current technology, a rate of 10^{13} encryption per second is reasonable
- AES, 3DES

Average Time Required for Exhaustive Key Search

Key size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 decryptions/s	Time Required at 10^{13} decryptions/s
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	2^{55} ns = 1.125 years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	2^{127} ns = 5.3×10^{21} years	5.3×10^{17} years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	2^{167} ns = 5.8×10^{33} years	5.8×10^{29} years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	2^{191} ns = 9.8×10^{40} years	9.8×10^{36} years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	2^{255} ns = 1.8×10^{60} years	1.8×10^{56} years
26 characters (permutation)	Monoalphabetic	$26! = 4 \times 10^{26}$	2×10^{26} ns = 6.3×10^{9} years	6.3×10^{6} years

Concerns cont'd

- **2. The nature of the DES algorithm**
  - S-boxes in the round function F(R, K)
  - Design criteria were not made public

- The substitution consists of a set of 8 S-boxes, each of which accepts 6 bits as input and produce 4 bits as output.

F be nonlinear

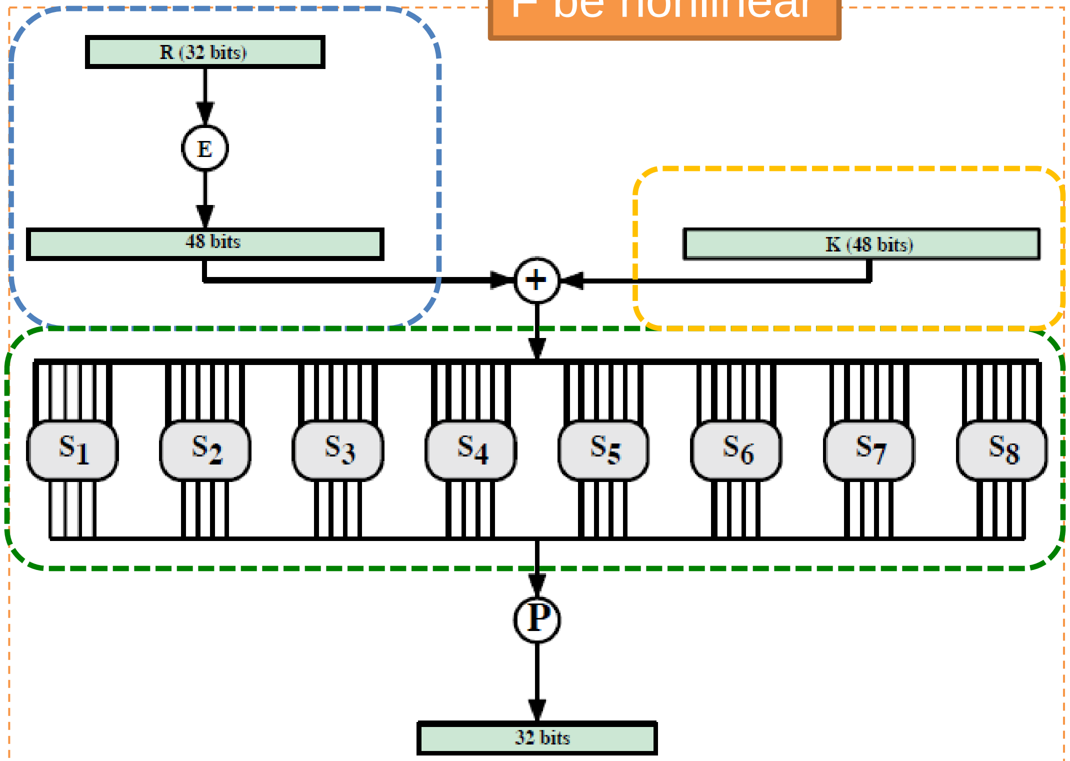


Figure S.3 Calculation of F(R, K)

Definition of DES S-Boxes

- The first and last bits of the input to box Si form a 2 bits binary number to select one of 4 rows
- The middle 4 bits select one of the 16 columns.
- The decimal value in the cell selected by the row and column is then converted to its 4 bits representation to produce the output.

- Ex. Input of S_1=**0**1100**1**
  => **01** row
  => 1100=12 column
  => Output=12=1100

Table S.2 Definition of DES S-Boxes

S_1	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

E Table

- E table: Expansion/permutation table

- The 32 bits input divided into small groups with size 4 bits
- Then the 4 bit data convert to 6 bits by repeating the characters on the edge to become 48 bits output

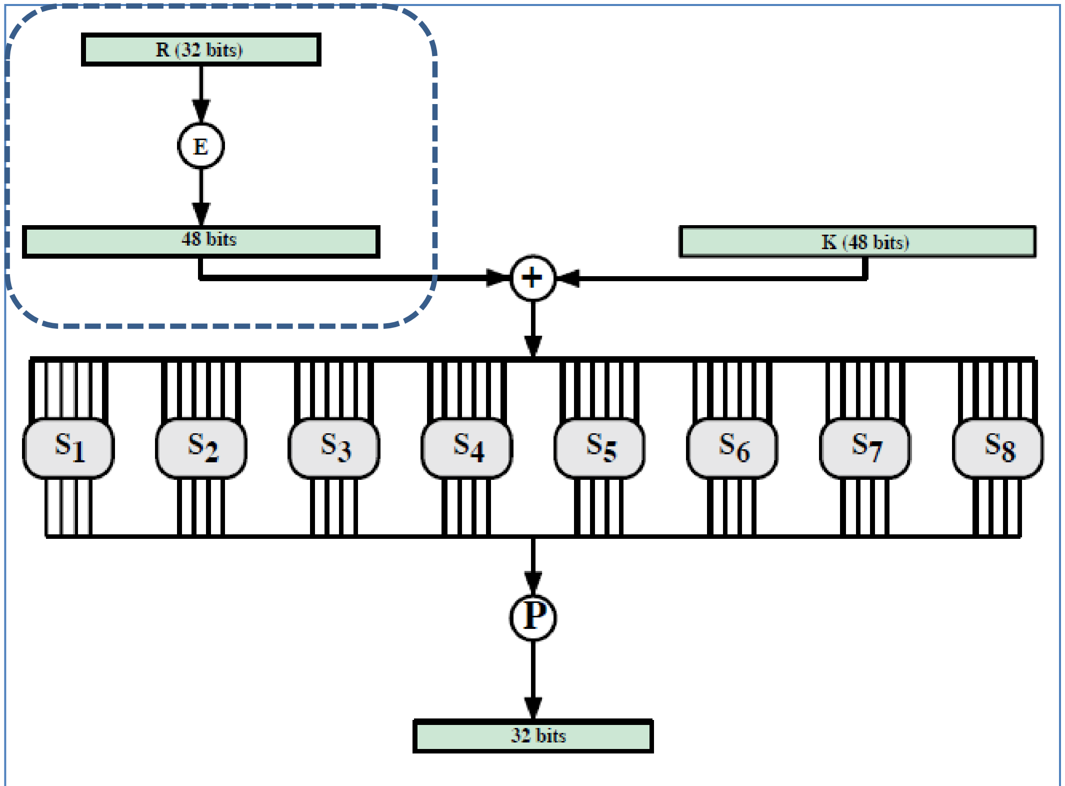


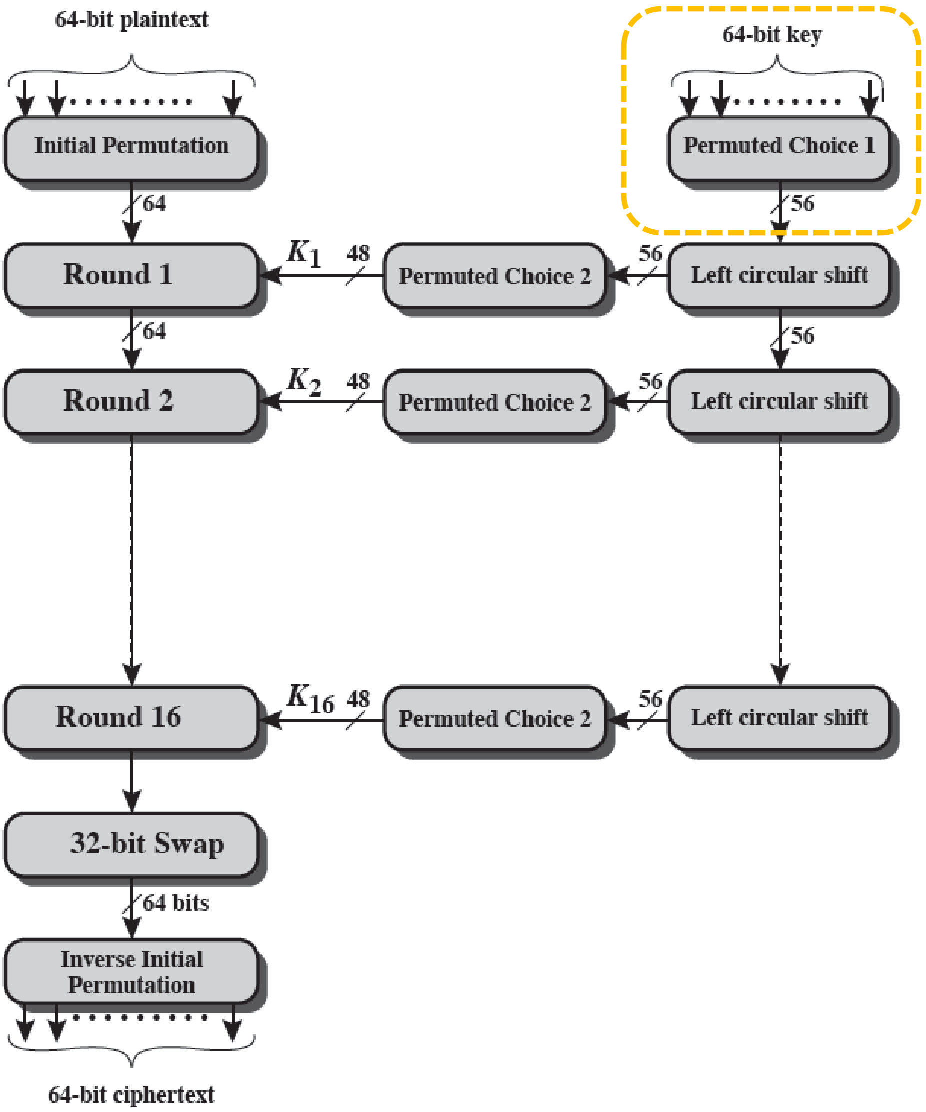
Figure S.3 Calculation of F(R, K)

- Example:
  32 bits input:
  …efgh ijkl mnop…

  becomes
  48 bits output:
  …defghi hijklm lmnopq…

Key Generation

- The 64 bits key is passed through a permutation function then generate Permuted Choice 1 for 56 bits output



Cont’d

- Then the 56 bits input is divided into half, Ci and Di, with size 28 bits
- Ci and Di Left Shift(s) one or two bit according to the number of round
- These two 28 bits blocks pass through another permutation Permuted Choice 2 and shrink to 48 bits

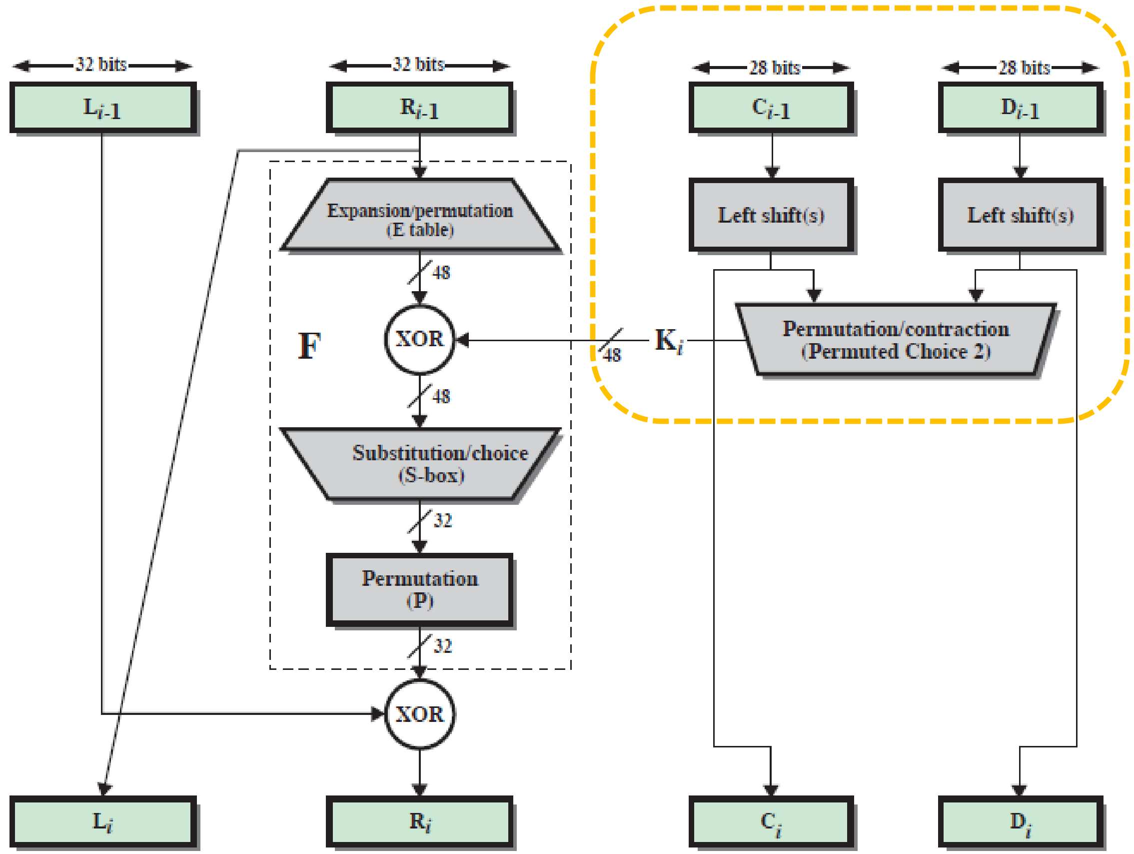


Figure S.2 Single Round of DES Algorithm

(d) Schedule of Left Shifts

Round Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits Rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

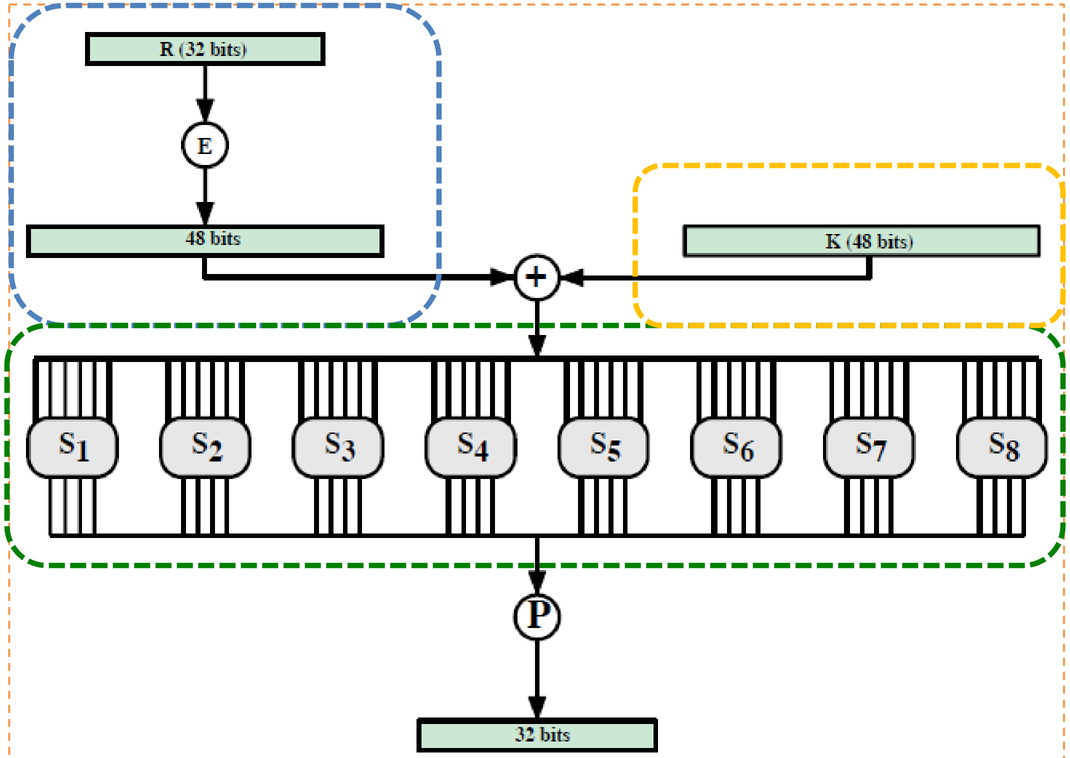


Figure S.3 Calculation of F(R, K)

Timing attacks

- One in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts
- Exploits the fact that an encryption or decryption algorithm often takes slightly different amounts of time on different inputs
- So far it appears unlikely that this technique will ever be successful against DES or more powerful symmetric ciphers such as triple DES and AES

Block Cipher Design Principles

- The cryptographic strength of a Feistel cipher derives from three aspects of the design:
  - Number of Rounds
  - Design of the function F
  - Key scheduling algorithm

1. Number of Rounds

The greater the number of rounds, the more difficult it is to perform cryptanalysis

In general, the criterion should be that the number of rounds is chosen so that known cryptanalytic efforts require greater effort than a simple brute-force key search attack

If DES had 15 or fewer rounds, differential cryptanalysis would require less effort than a brute-force key search

- The differential cryptanalysis attack requires $2^{55.1}$ operations, whereas brute force requires 2^{55}.
- The strength of any algorithm that satisfies the criterion can be judged solely on key length

2. Design of Function F

- The heart of a Feistel block cipher is the function F
- The more nonlinear F, the more difficult any type of cryptanalysis will be
- The SAC and BIC criteria appear to strengthen the effectiveness of the confusion function

Strict avalanche criterion (SAC)

States that any output bit j of an S-box should change with probability 1/2 when any single input bit i is inverted for all i , j

Bit independence criterion (BIC)

States that output bits j and k should change independently when any single input bit i is inverted for all i , j , and k

- Avalanche: a change in one bit of the input should produce a change in many bits of the output.

3. Key Schedule Algorithm

- With any Feistel block cipher, the key is used to generate one subkey for each round
- In general, we would like to select subkeys to maximize the difficulty of deducing individual subkeys and the difficulty of working back to the main key
- It is suggested that, at a minimum, the key schedule should guarantee key/ciphertext Strict Avalanche Criterion and Bit Independence Criterion