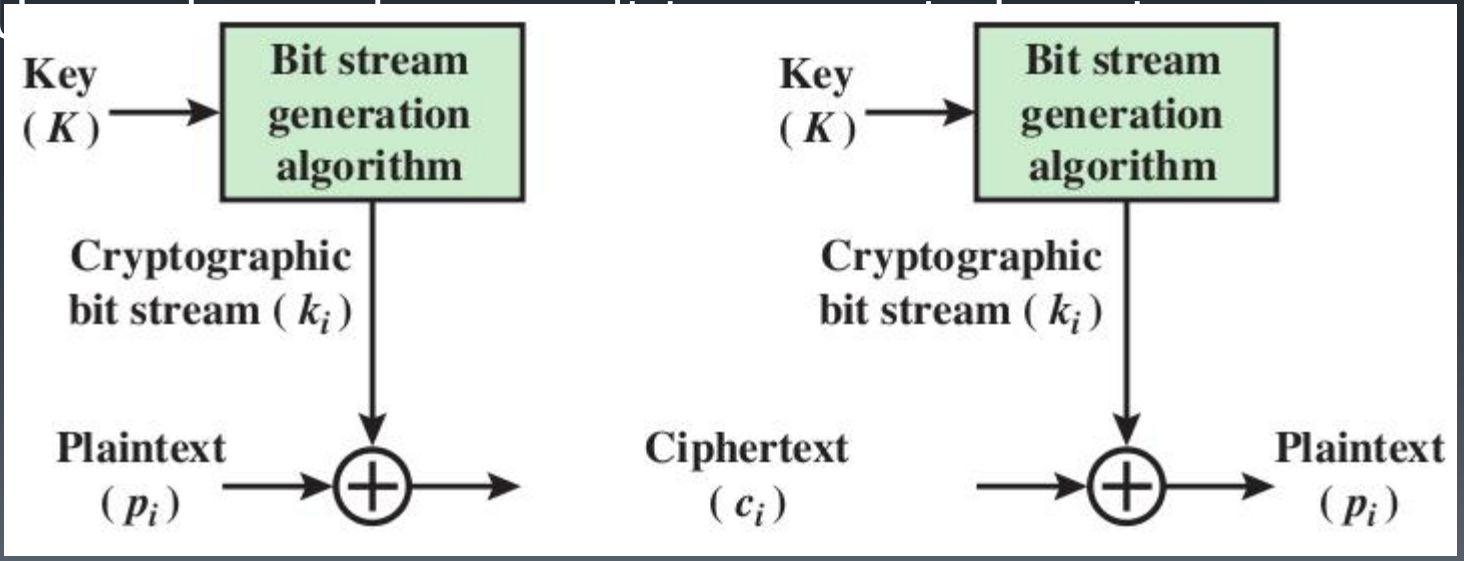
Chapter 2

Block Cipher Primitives: Confusion and Diffusion

- Claude Shannon: There are two primitive operations with which strong encryption algorithms can be built:
  - Confusion: An encryption operation where the relationship between key and ciphertext is obscured. Example : Substitution
  - Diffusion: An encryption operation where the influence of one plaintext symbol is spread over many ciphertext symbols with the goal of hiding statistical properties of the plaintext. Example : Bit Permutation

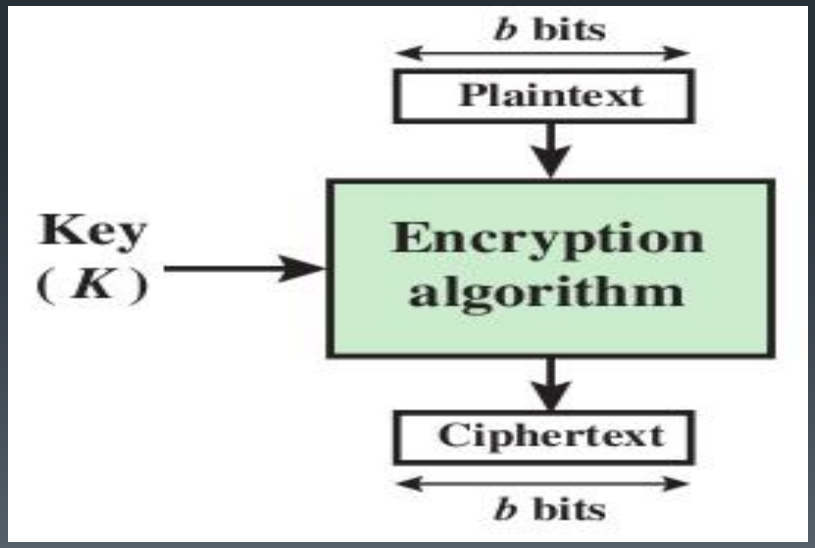
Stream Ciphers

- Encrypts a digital data stream one bit or one byte at a time
- Key (K) used as input to bit-stream generator algorithm
- Algorithm generates cryptographic bit stream (ki) used to encrypt plaintext
- U

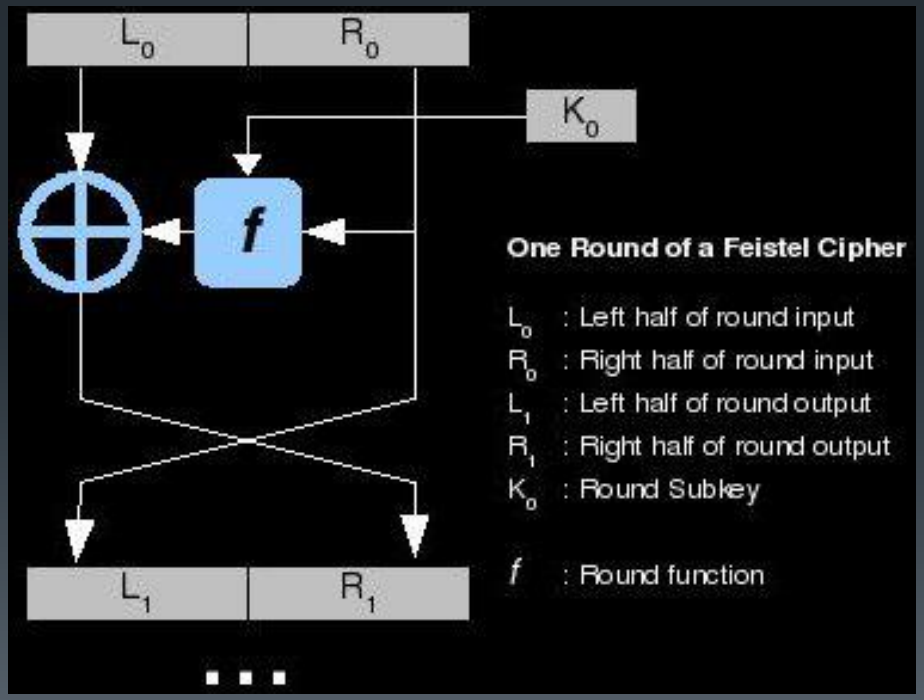


Block Ciphers

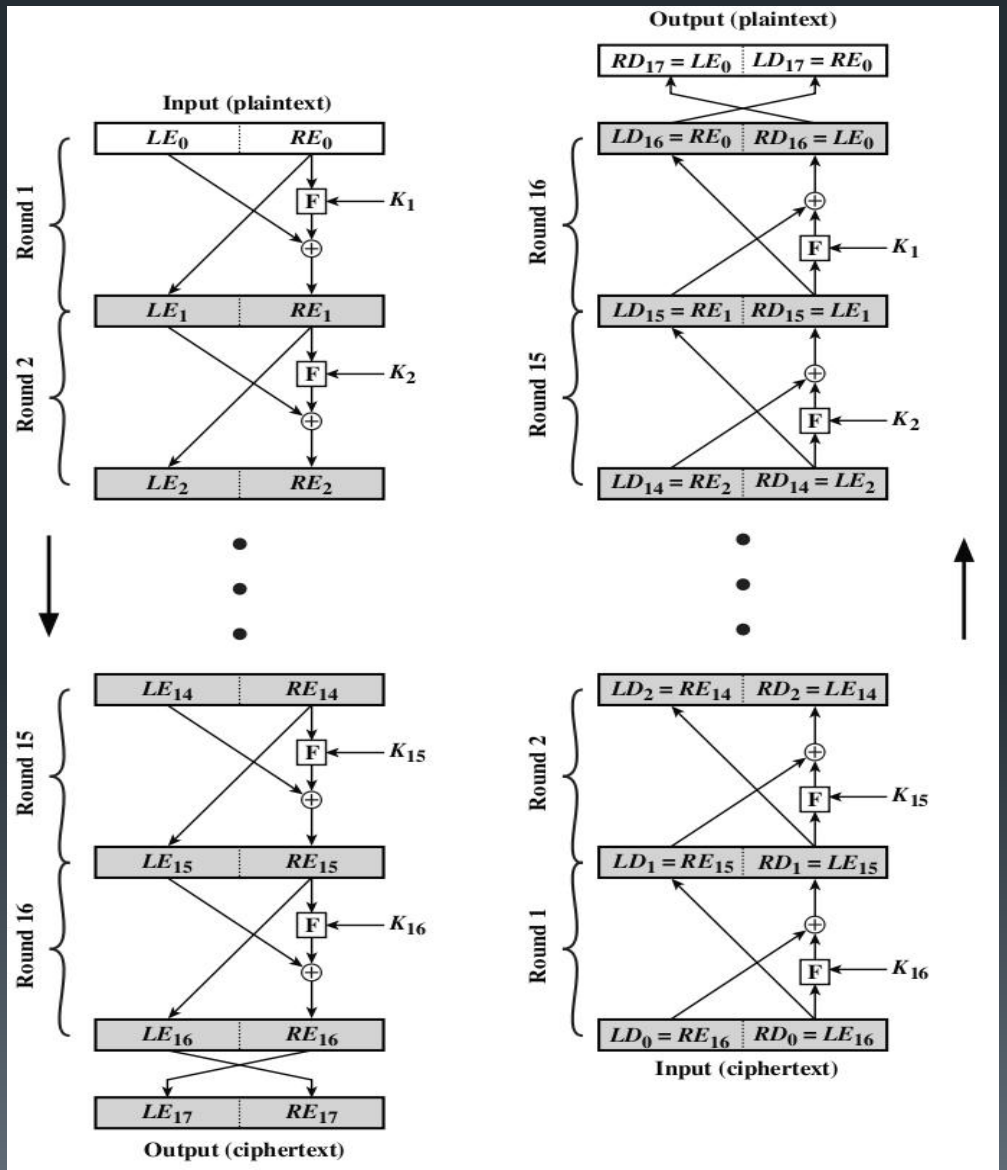
- Encrypt a block of plaintext as a whole to produce same sized ciphertext
- Typical block sizes are 64 or 128 bits
- Modes of operation used to apply block ciphers to larger plaintexts



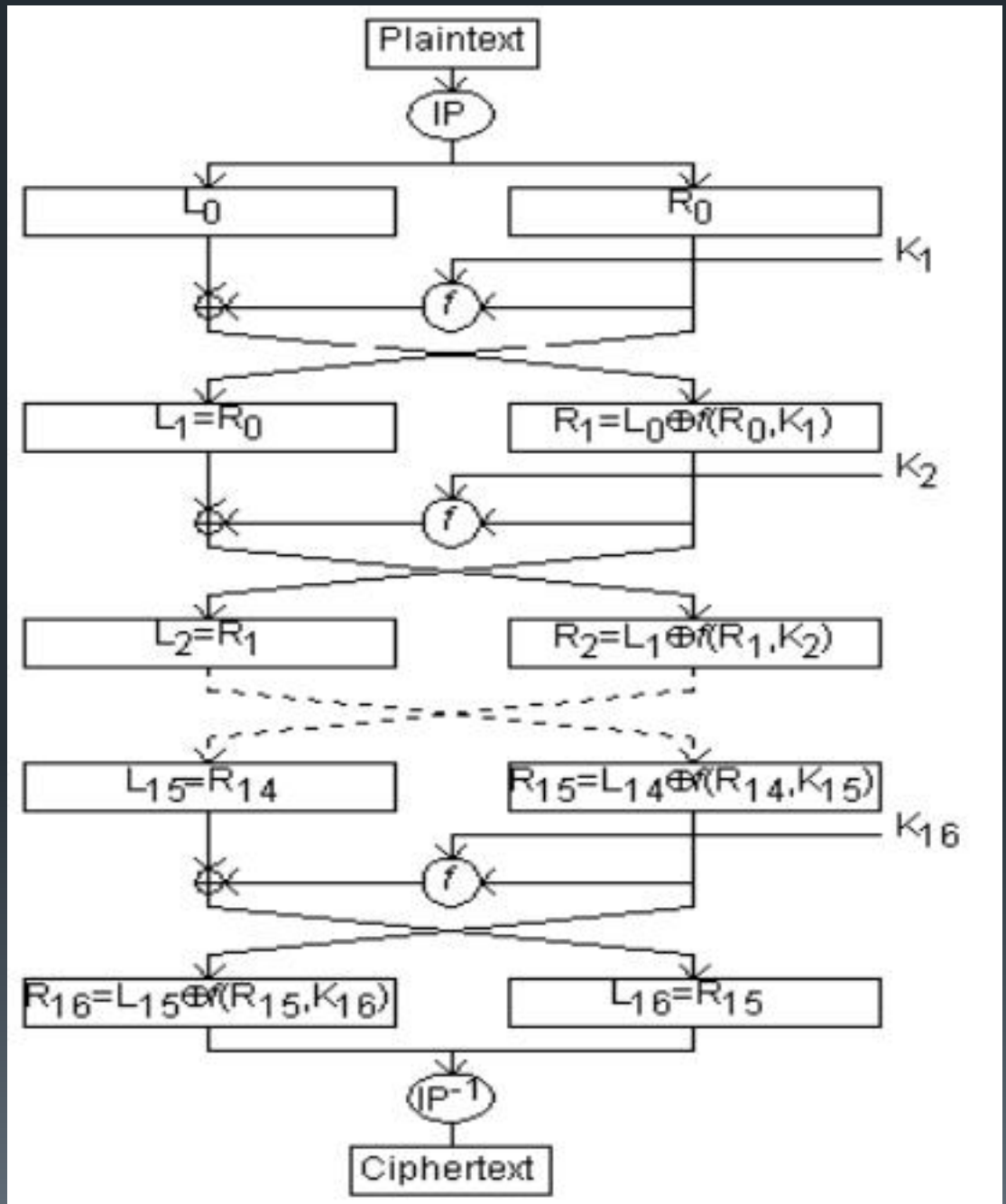
Fiestel Network



Feistel Encryption and Decryption



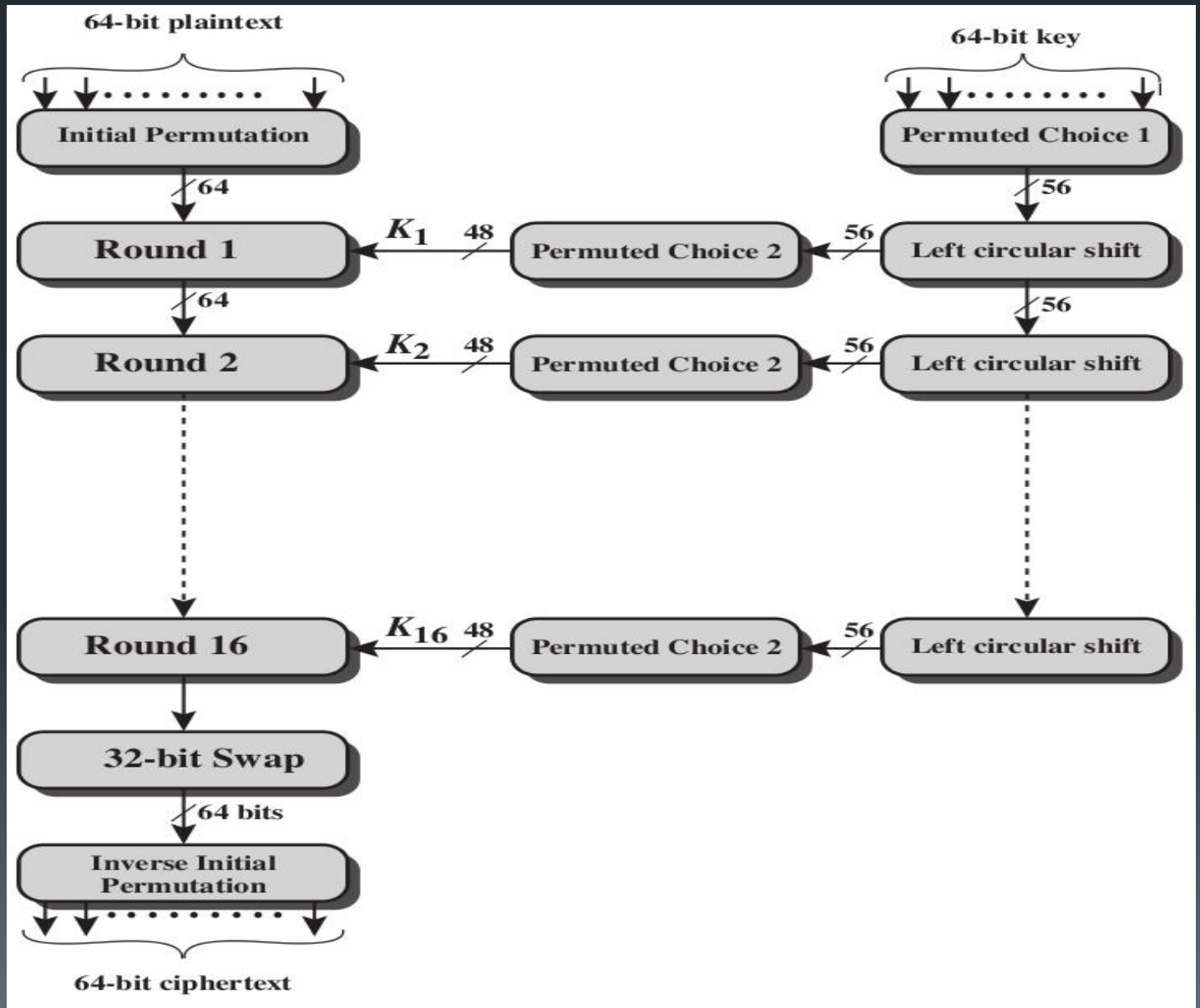
DES


Plaintext
IP
L_0
R_0
K_1
f
$L_1=R_0$
$R_1=L_0 \oplus f(R_0,K_1)$
K_2
f
$L_2=R_1$
$R_2=L_1 \oplus f(R_1,K_2)$
$L_{15}=R_{14}$
$R_{15}=L_{14} \oplus f(R_{14},K_{15})$
K_{16}
f
$R_{16}=L_{15} \oplus f(R_{15},K_{16})$
$L_{16}=R_{15}$
IP^{-1}
Ciphertext

DES

- Data Encryption Standard (DES) encrypts **blocks of size 64 bit.**
- Developed by **IBM based on the cipher *Lucifer* *under influence of the National*** *Security Agency (NSA), the design criteria for DES have not been published*
- **Standardized 1977 by the National Bureau of Standards (NBS)** today called *National Institute of Standards and Technology (NIST)*
- Most popular **block cipher for most of the last 30 years.**
- By far best studied symmetric algorithm.
- Nowadays considered insecure due to the small **key length of 56 bit.**
- **But: 3DES yields very secure cipher, still widely used today.**

General DES Encryption Algorithm



DES Round Structure

- uses two 32-bit L & R halves
- as for any Feistel cipher can describe as:

  $L_i = R_{i-1}$

  $R_i = L_{i-1} \quad \mathrm{F}(R_{i-1}, K_i)$
- F takes 32-bit R half and 48-bit subkey:
  - expands R to 48-bits using perm E
  - adds to subkey using XOR
  - passes through 8 S-boxes to get 32-bit result
  - finally permutes using 32-bit perm P

Permutation Tables for DES

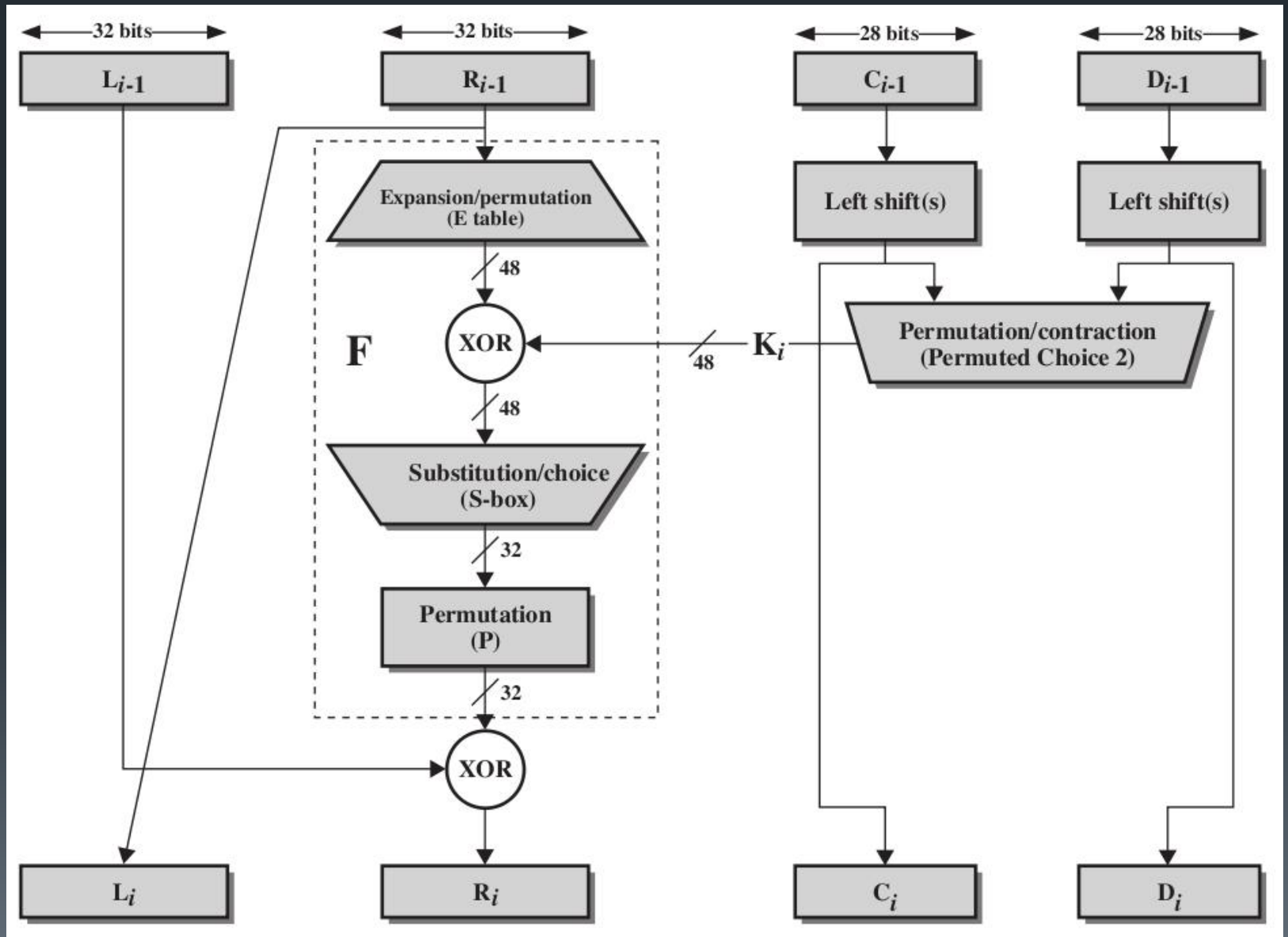
(a) Initial Permutation (IP)

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

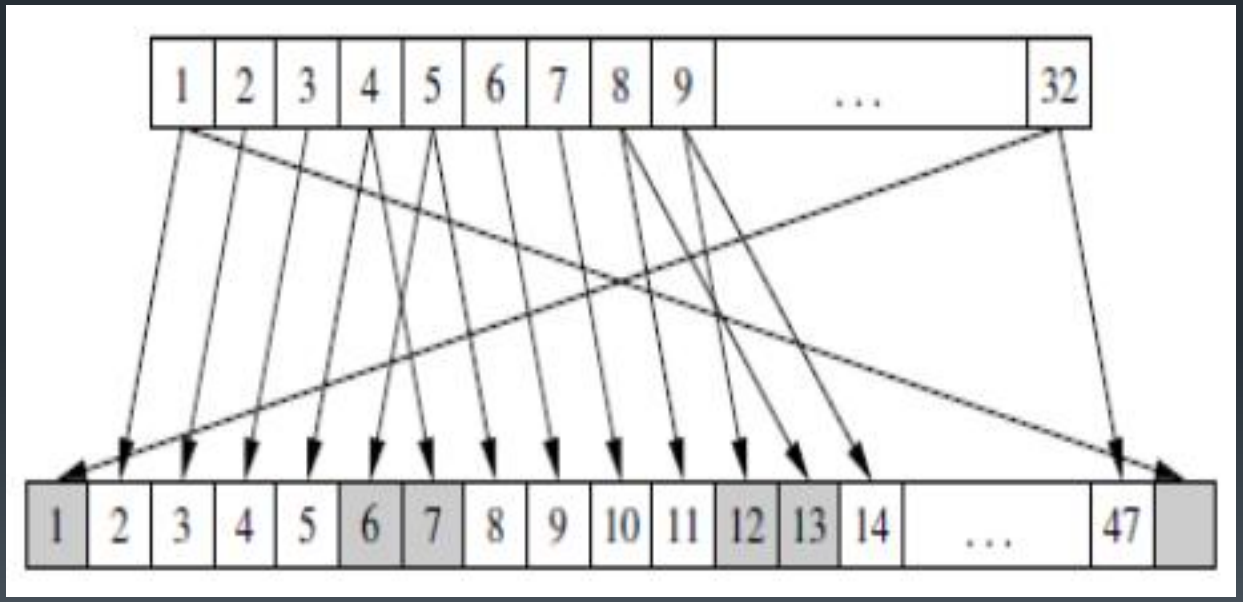
(b) Inverse Initial Permutation (IP^{-1})

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Single Round of DES Algorithm

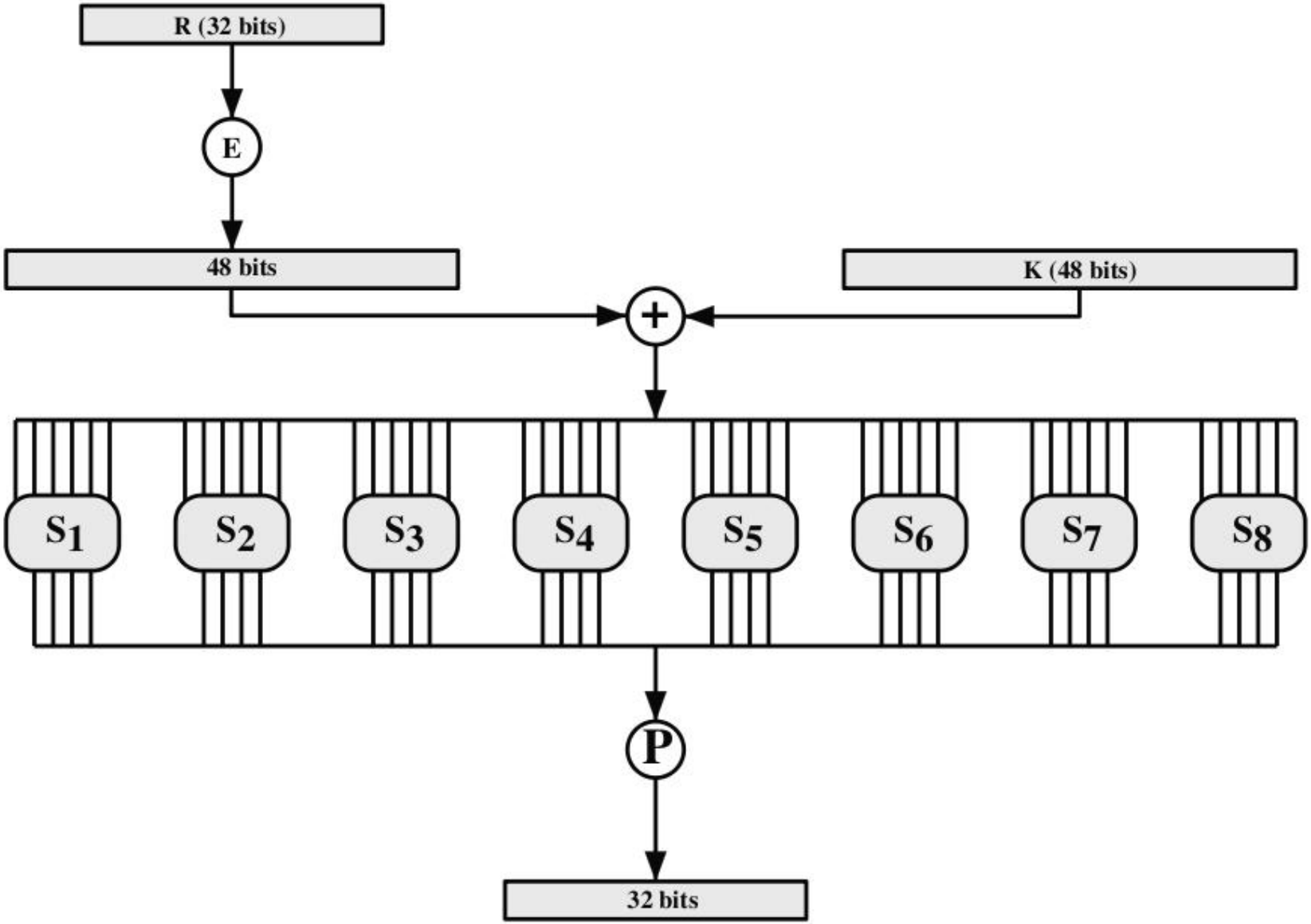


Expansion

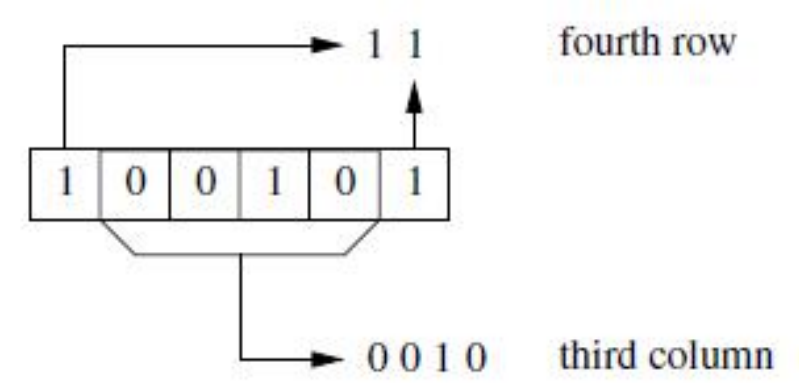


E					
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Calculation of F(R,K)



48 Bits to 32 Bits



S_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	01	10	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

Dentition 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

DES Key Schedule Calculation

(a) Input Key

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	52	53	54	55	56
57	58	59	60	61	62	63	64

(b) Permuted Choice One (PC-1)

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

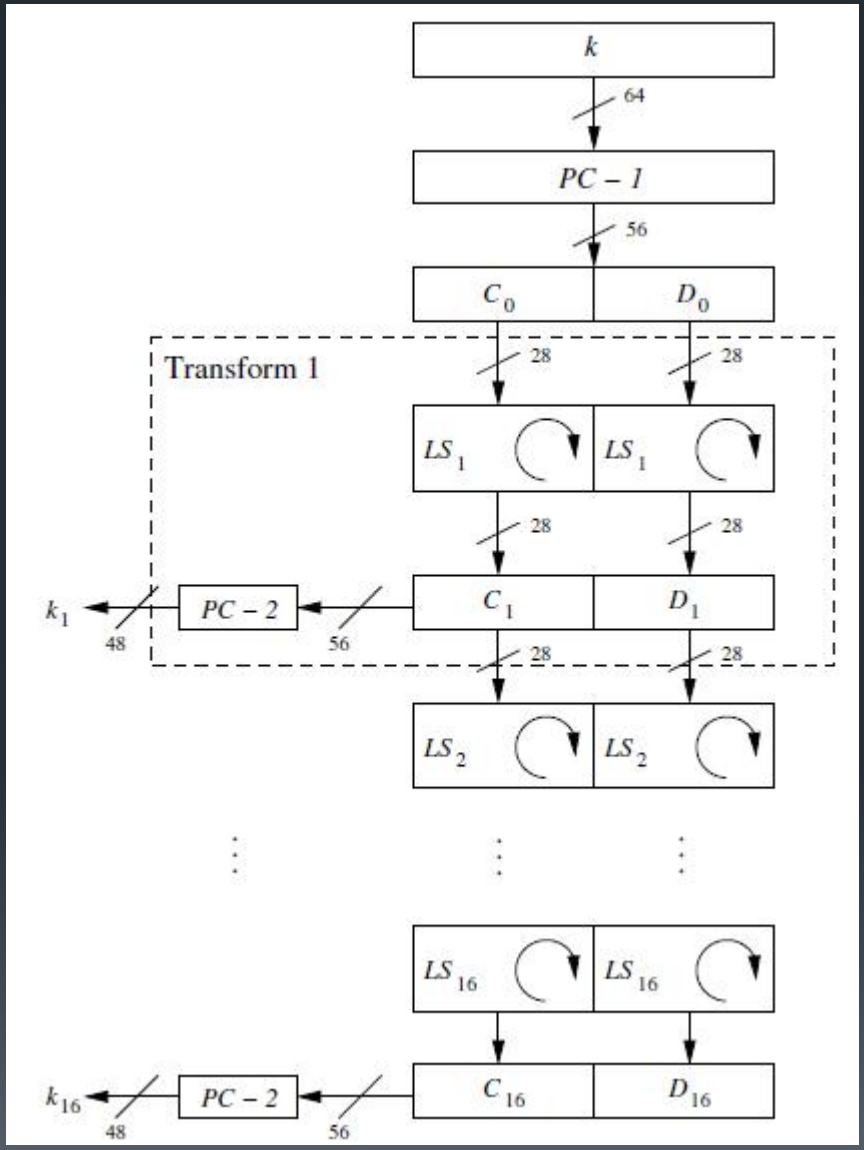
(c) Permuted Choice Two (PC-2)

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

(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

Key Schedule for DES Encryption



Key Size

- Although 64 bit initial key, only 56 bits used in encryption (other 8 for parity check)
- $2^{56} = 7.2 \times 10^{16}$
- 1977: estimated cost $US20m to build machine to break in 10 days.
- 1998: EFF built machine for $US250k to break in 3 days
- Today: 56 bits considered too short to withstand brute force attack
- 3DES uses 128-bit keys

DES Example

- Plaintext: 02468aceeca86420
- Key: 0f1571c947d9e859
- Ciphertext: da02ce3a89ecac3b

DES Example

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

Change in plaintext

Round		δ
	02468aceeca86420 12468aceeca86420	1
1	3cf03c0fbad22845 3cf03c0fbad32845	1
2	bad2284599e9b723 bad3284539a9b7a3	5
3	99e9b7230bae3b9e 39a9b7a3171cb8b3	18
4	0bae3b9e42415649 171cb8b3ccaca55e	34
5	4241564918b3fa41 ccaca55ed16c3653	37
6	18b3fa419616fe23 d16c3653cf402c68	33
7	9616fe2367117cf2 cf402c682b2cefbc	32
8	67117cf2c11bfc09 2b2cefbc99f91153	33

Round		δ
9	c11bfc09887fbc6c 99f911532eed7d94	32
10	887fbc6c600f7e8b 2eed7d94d0f23094	34
11	600f7e8bf596506e d0f23094455da9c4	37
12	f596506e738538b8 455da9c47f6e3cf3	31
13	738538b8c6a62c4e 7f6e3cf34bc1a8d9	29
14	c6a62c4e56b0bd75 4bc1a8d91e07d409	33
15	56b0bd7575e8fd8f 1e07d4091ce2e6dc	31
16	75e8fd8f25896490 1ce2e6dc365e5f59	32
IP^{-1}	da02ce3a89ecac3b 057cde97d7683f2a	32

Change in key

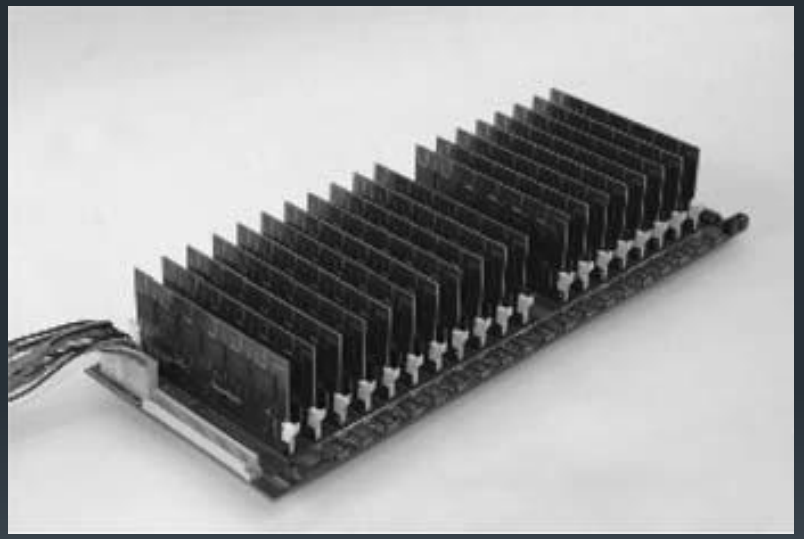
- Original key is 0f1571c947d9e859 , it is changed to 1f1571c947d9e859

Round		δ
	02468aceeca86420 02468aceeca86420	0
1	3cf03c0fbad22845 3cf03c0f9ad628c5	3
2	bad2284599e9b723 9ad628c59939136b	11
3	99e9b7230bae3b9e 9939136b768067b7	25
4	0bae3b9e42415649 768067b75a8807c5	29
5	4241564918b3fa41 5a8807c5488dbe94	26
6	18b3fa419616fe23 488dbe94aba7fe53	26
7	9616fe2367117cf2 aba7fe53177d21e4	27
8	67117cf2c11bfc09 177d21e4548f1de4	32

Round		δ
9	c11bfc09887fbc6c 548f1de471f64dfd	34
10	887fbc6c600f7e8b 71f64dfd4279876c	36
11	600f7e8bf596506e 4279876c399fdc0d	32
12	f596506e738538b8 399fdc0d6d208dbb	28
13	738538b8c6a62c4e 6d208dbbb9bdeeaa	33
14	c6a62c4e56b0bd75 b9bdeeaad2c3a56f	30
15	56b0bd7575e8fd8f d2c3a56f2765c1fb	33
16	75e8fd8f25896490 2765c1fb01263dc4	30
IP^{-1}	da02ce3a89ecac3b ee92b50606b62b0b	30

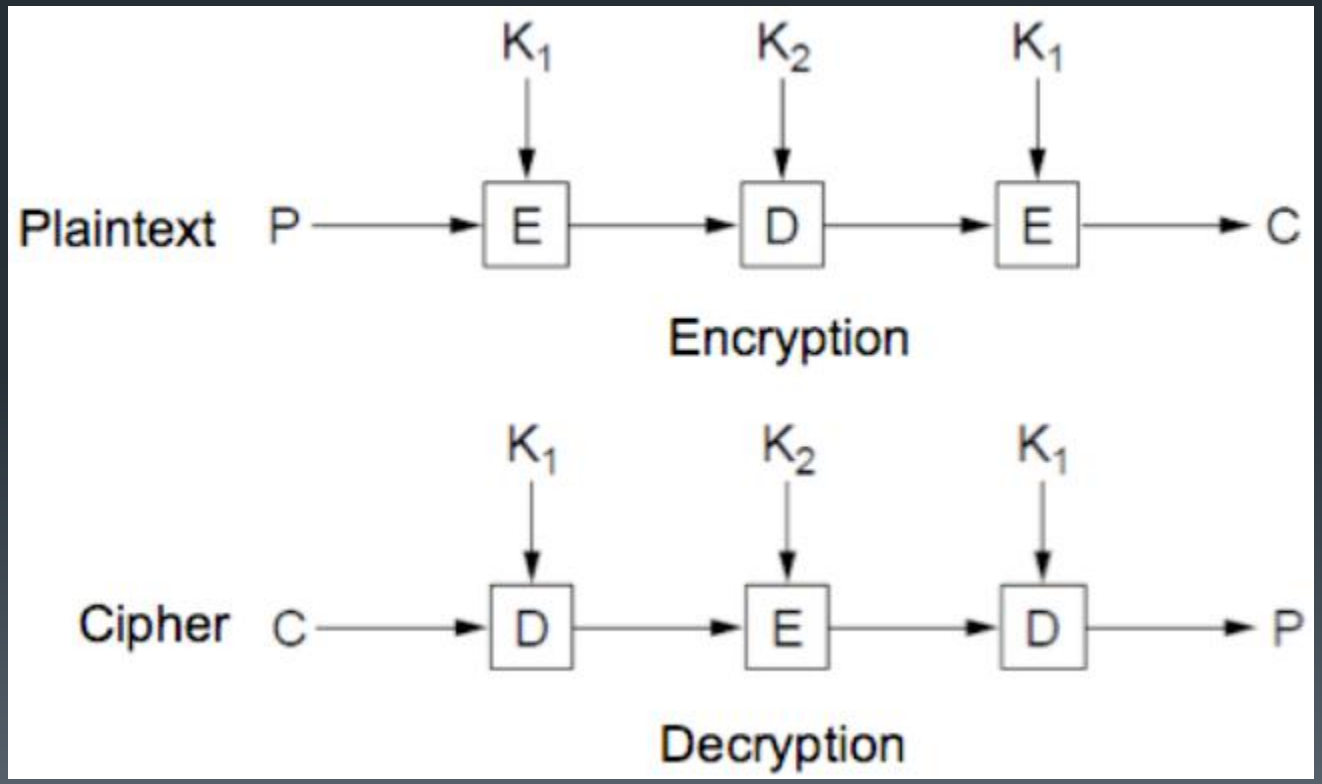
DES

Deep Crack — the hardware exhaustive key-search machine that broke DES in 1998



COPACOBANA —A cost-optimized parallel code breaker (2001)

3 DES


K1
K2
K1
Plaintext
P
E
D
E
C
Encryption
K1
K2
K1
Cipher
C
D
E
D
P
Decryption

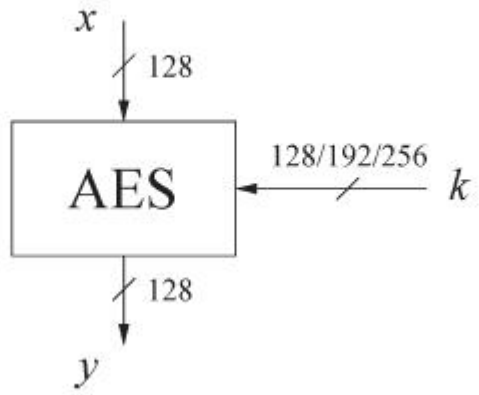
AES (Advanced Encryption Standard

- AES is the most widely used symmetric cipher today
- The algorithm for AES was chosen by the US *National Institute of Standards and Technology (NIST) in a multi-year selection process*
- The requirements for all AES candidate submissions were:
  - Block cipher with **128-bit block size**
  - **Three supported key lengths: 128, 192 and 256 bit**
  - **Efficiency in software and hardware**

AES

- The need for a new block cipher announced by NIST in January, 1997
- 15 candidates algorithms accepted in August, 1998
- 5 finalists announced in August, 1999:
  - *Mars – IBM Corporation*
  - *RC6 – RSA Laboratories*
  - *Rijndael – J. Daemen & V. Rijmen*
  - *Serpent – Eli Biham et al.*
  - *Twofish – B. Schneier et al.*
- In October 2000, *Rijndael was chosen as the AES*
- AES was formally approved as a US federal standard in November 2001

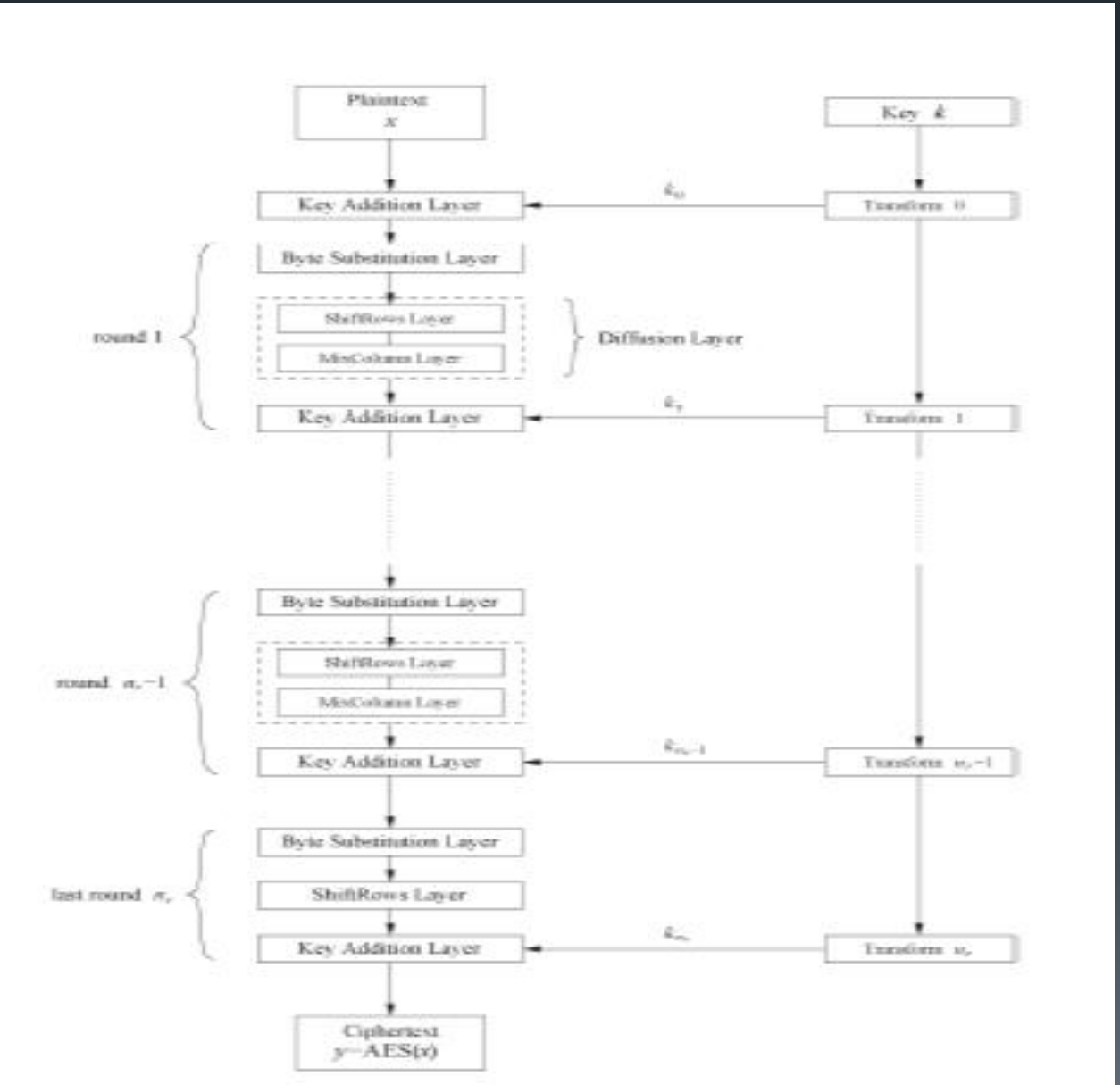
AES: Overview



The number of rounds depends on the chosen key length:

Key length (bits)	Number of rounds
128	10
192	12
256	14

AES: Overview



AES

- Each round consists of 4 layers
  - Byte Substitution (confusion)
  - ShiftRow (defusion)
  - MixColumn , except last round (defusion)
  - Key addition
- Key Whitening

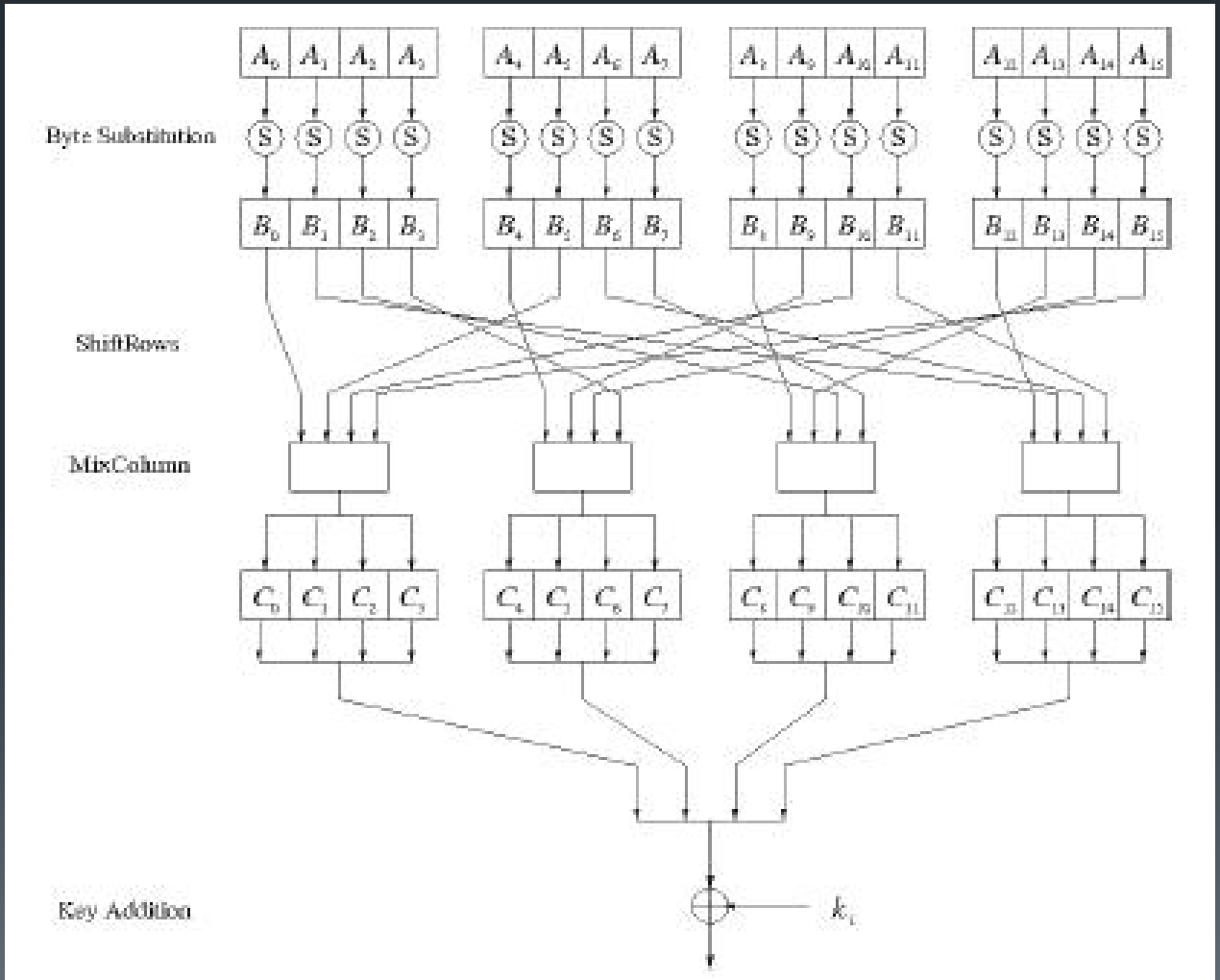
Internal Structure of AES

- AES is a byte-oriented cipher
- The state *A (i.e., the 128-bit data path) can be arranged in a 4x4 matrix:* with *A0,..., A15 denoting the 16-byte input of AES*

A_0	A_4	A_8	A_{12}
A_1	A_5	A_9	A_{13}
A_2	A_6	A_{10}	A_{14}
A_3	A_7	A_{11}	A_{15}

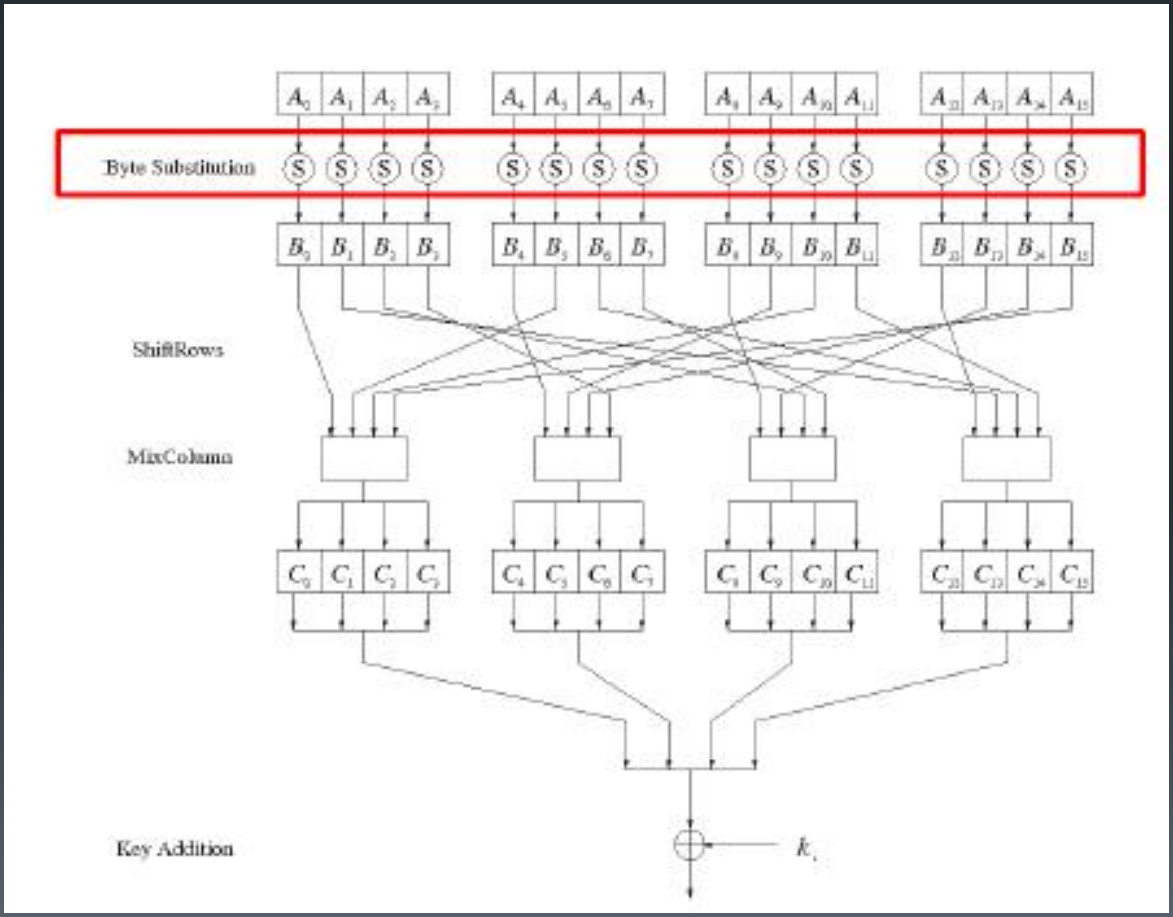
Internal Structure of AES

- Round function for rounds 1,2,…,*nr-1:*



Byte Substitution Layer

- The Byte Substitution layer consists of 16 similar **S-Boxes**

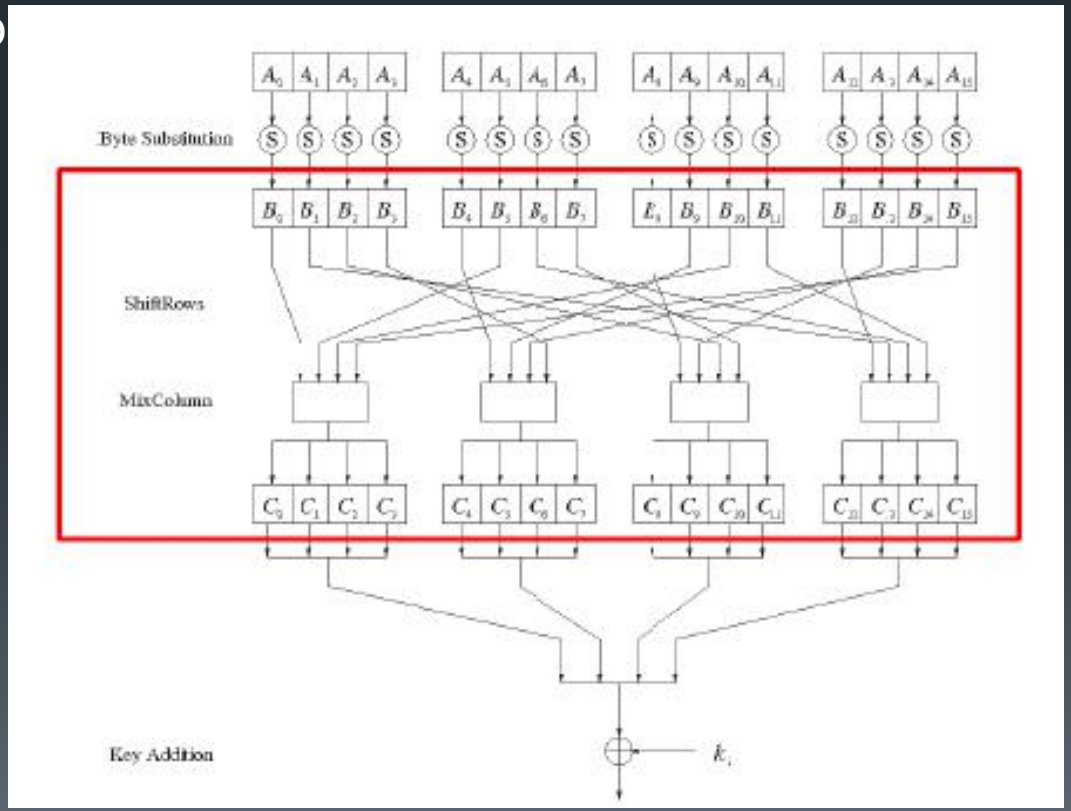


S Box

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Diffusion Layer

- Provides diffusion over all input state bits
- Consists of two sublayers:
  - ShiftRows Sublayer: Permutation of the data on a byte level
  - MixColumn Sublayer: Matrix operation which combines ("mixes") b



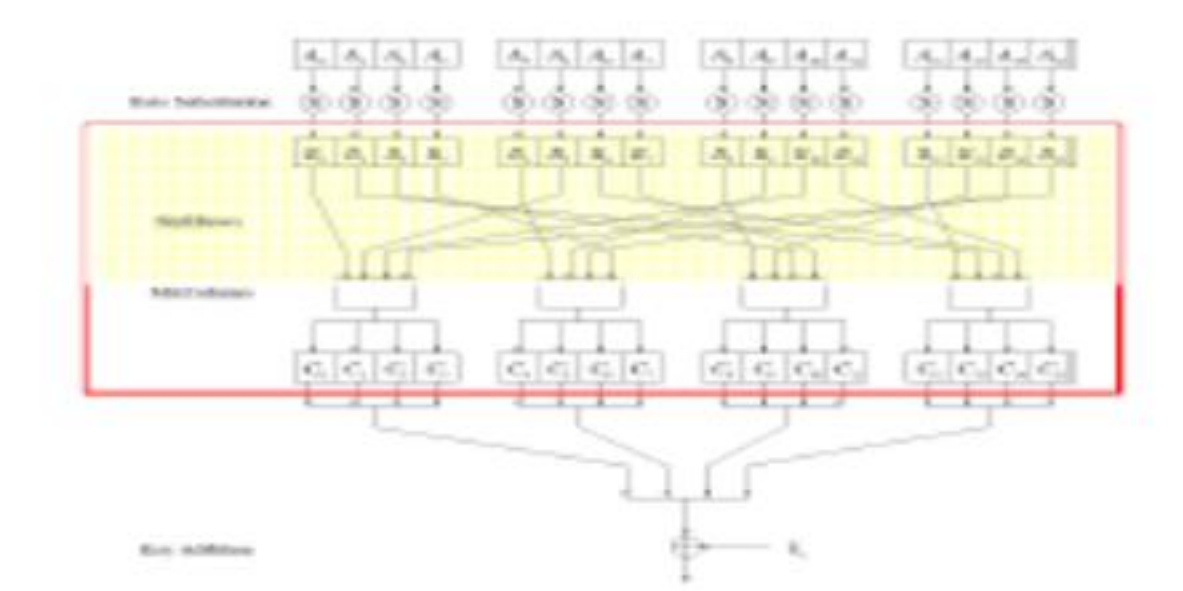
Shift Row

Input matrix

B_0	B_4	B_8	B_{12}
B_1	B_5	B_9	B_{13}
B_2	B_6	B_{10}	B_{14}
B_3	B_7	B_{11}	B_{15}

Output matrix

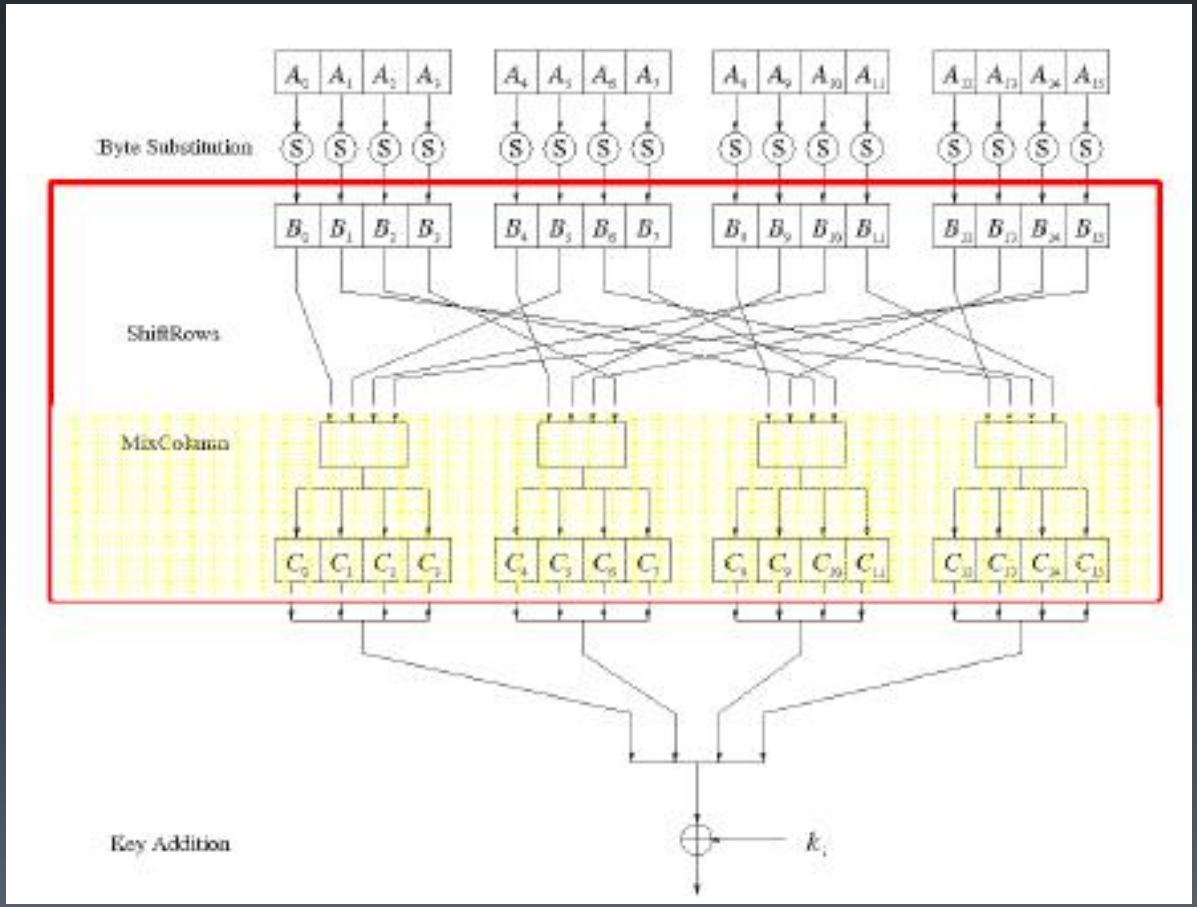
B_0	B_4	B_8	B_{12}	no shift
B_5	B_9	B_{13}	B_1	← one position left shift
B_{10}	B_{14}	B_2	B_6	← two positions left shift
B_{15}	B_3	B_7	B_{11}	← three positions left shift



MixColumn Sublayer

- Linear transformation which mixes each column of the state matrix
- Each 4-byte column is considered as a vector and multiplied x,

$$\begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} B_0 \\ B_5 \\ B_{10} \\ B_{15} \end{pmatrix}$$



Key Addition Layer

- Inputs:
  - 16-byte state matrix *C*
  - 16-byte subkey *ki*
- Output: ⊕ *ki*
- The subkeys are generated in the key schedule

AES

- AES is a modern block cipher which supports three key lengths of 128, 192 and 256 bit. It provides excellent long-term security against brute-force attacks.
- AES has been studied intensively since the late 1990s and no attacks have been found that are better than brute-force.
- AES is not based on Feistel networks. Its basic operations use Galois field arithmetic and provide strong diffusion and confusion.
- AES is part of numerous open standards such as IPsec or TLS, in addition to being the mandatory encryption algorithm for US government applications. It seems likely that the cipher will be the dominant encryption algorithm for many years to come.
- AES is efficient in software and hardware.

Decryption

- **Inv MixColumn layer:**
- • To reverse the MixColumn operation, each column of the state matrix *C* must be multiplied with the **inverse of the 4x4 matrix, e.g.**

$$\begin{pmatrix} B_0 \\ B_1 \\ B_2 \\ B_3 \end{pmatrix} = \begin{pmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{pmatrix} \cdot \begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix}$$

- where 09, 0*B, 0D and 0E are given in hexadecimal notation*

Decryption

- **Inv ShiftRows layer:**
- All rows of the state matrix *B are shifted to the opposite direction:*

Input matrix

B_0	B_4	B_8	B_{12}
B_1	B_5	B_9	B_{13}
B_2	B_6	B_{10}	B_{14}
B_3	B_7	B_{11}	B_{15}

Output matrix

B_0	B_4	B_8	B_{12}	no shift
B_{13}	B_1	B_5	B_9	→ one position right shift
B_{10}	B_{14}	B_2	B_6	→ two positions right shift
B_7	B_{11}	B_{15}	B_3	→ three positions right shift

Decryption

- **Inv Byte Substitution layer:**
- Since the S-Box is bijective, it is possible to construct an inverse, such that
  - $Ai = S^{-1}(Bi)$
- The inverse S-Box is used for decryption. It is usually realized as a lookup table

AES Decryption

- AES is not based on a Feistel network
- All layers must be inverted for decryption:
- MixColumn layer → **Inv MixColumn layer**
- ShiftRows layer→ **Inv ShiftRows layer**
- Byte Substitution layer → **Inv Byte Substitution layer**

International Data Encryption Algorithm (IDEA)

- IDEA is perceived as one of the strongest cryptographic algorithms but still not as popular as DES because of two reasons
  - It is patented unlike DES therefore, must be licensed before it can be used in commercial applications
  - DES has a long history and track record compared to IDEA
- IDEA is a block cipher designed by Xuejia Lai and James L. Massey in 1991
- It entirely avoids the use of any lookup tables or S-boxes

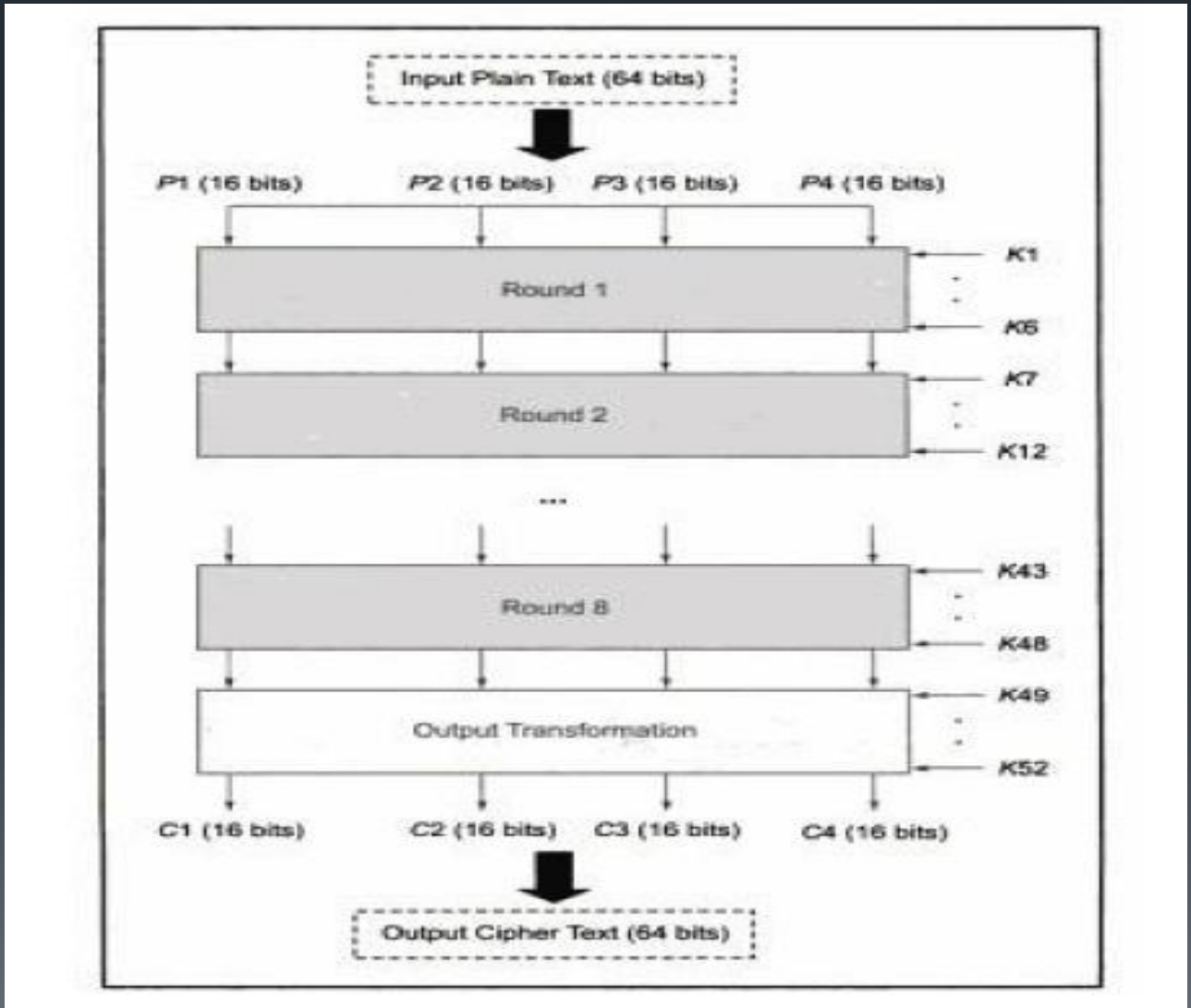
IDEA

- IDEA operates with 64-bit plaintext and cipher text blocks and is controlled by a 128-bit key
- Completely avoid substitution boxes and table lookups used in the block ciphers
- The algorithm structure has been chosen such that when different key sub-blocks are used, the encryption process is identical to the decryption process

Working of IDEA

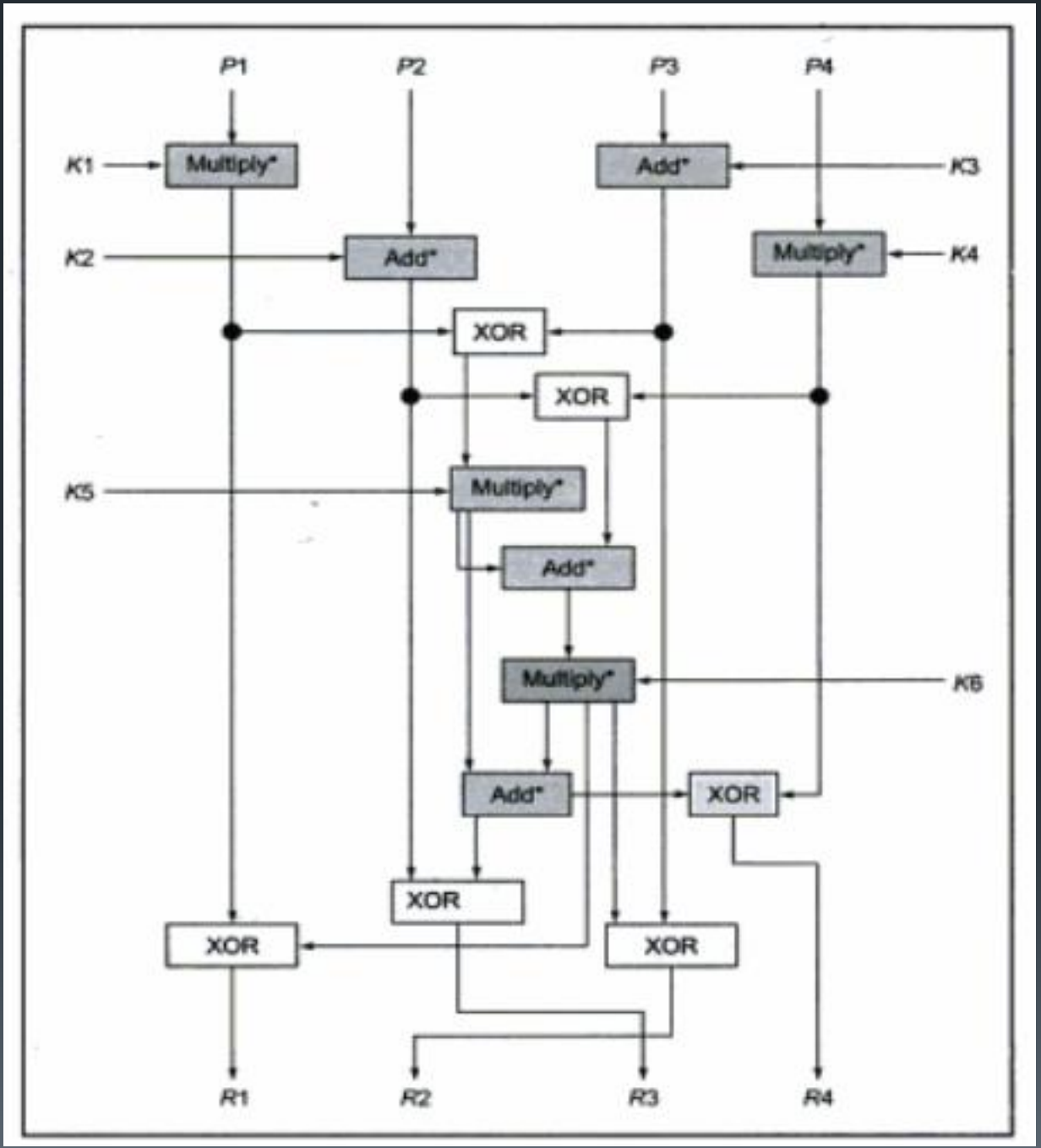
- The 64-bit input plaintext block P is divided into 4 portions, each of 16 bits, i.e. (P1 to P4).
- Thus, P1 to P4 are the inputs to the first round of the algorithm. There are *eight* such rounds.
- The key consist of *128 bits.*
- In each round, six sub keys are generated from the original key. Each of the sub-keys is of *16 bits*.
- These six sub-keys are applied to the four input blocks P1 to P4.
- Thus for the first round we will have the six keys K1 to K6. Similarly for the eighth round we will have keys K43 to K48.
- The final steps consist of an output transformation, which uses just four sub-keys (K49 to K52).
- The final output produced is the output produced by the output transformation step, which is four blocks of cipher text named C1 to C4 (each of 16 bits). These are combined to form the final 64 bit cipher text block.

Working of IDEA



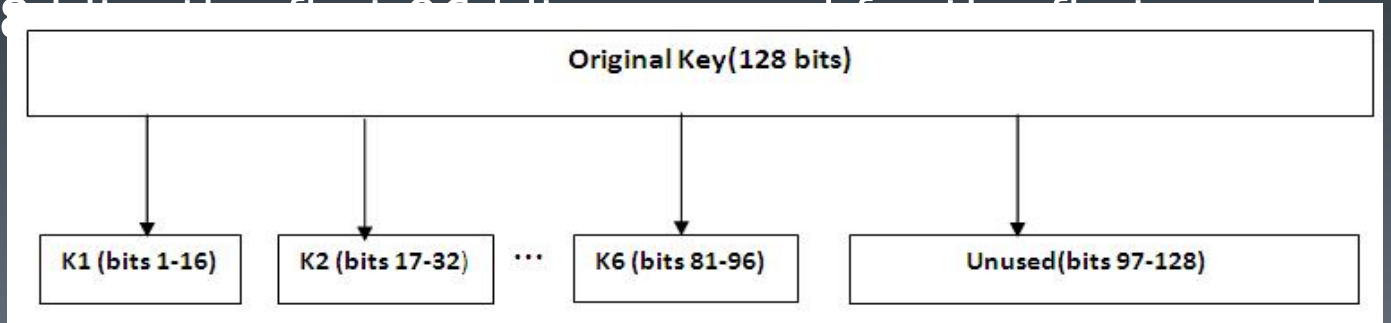
Rounds

- There are eight rounds in IDEA and each round involves a series of operations on the four data blocks using six keys
  - Step-1: Multiply* P1 and K1.
  - Step-2: Add* P2 and K2.
  - Step-3: Add* P3 and K3.
  - Step-4: Multiply* P4 and K4.
  - Step-5: XOR the results of step-1 and step-3.
  - Step-6: XOR the results of step-2 and step-4.
  - Step-7: Multiply* the results of step-5 with K5.
  - Step-8: Add* the results of step-6 and step-7.
  - Step-9: Multiply* the results of step-8 with K6.
  - Step-10: Add* the results of step-7 and step-9.
  - Step-11: XOR the results of step-1 and step-9.
  - Step-12: XOR the results of step-3 and step-9.
  - Step-13: XOR the results of step-2 and step-10.
  - Step-14: XOR the results of step-4 and step-10.


P1
P2
P3
P4
K1
Multiply*
Add*
K3
K2
Add*
Multiply*
K4
XOR
XOR
K5
Multiply*
Add*
Multiply*
K6
Add*
XOR
XOR
XOR
XOR
R1
R2
R3
R4

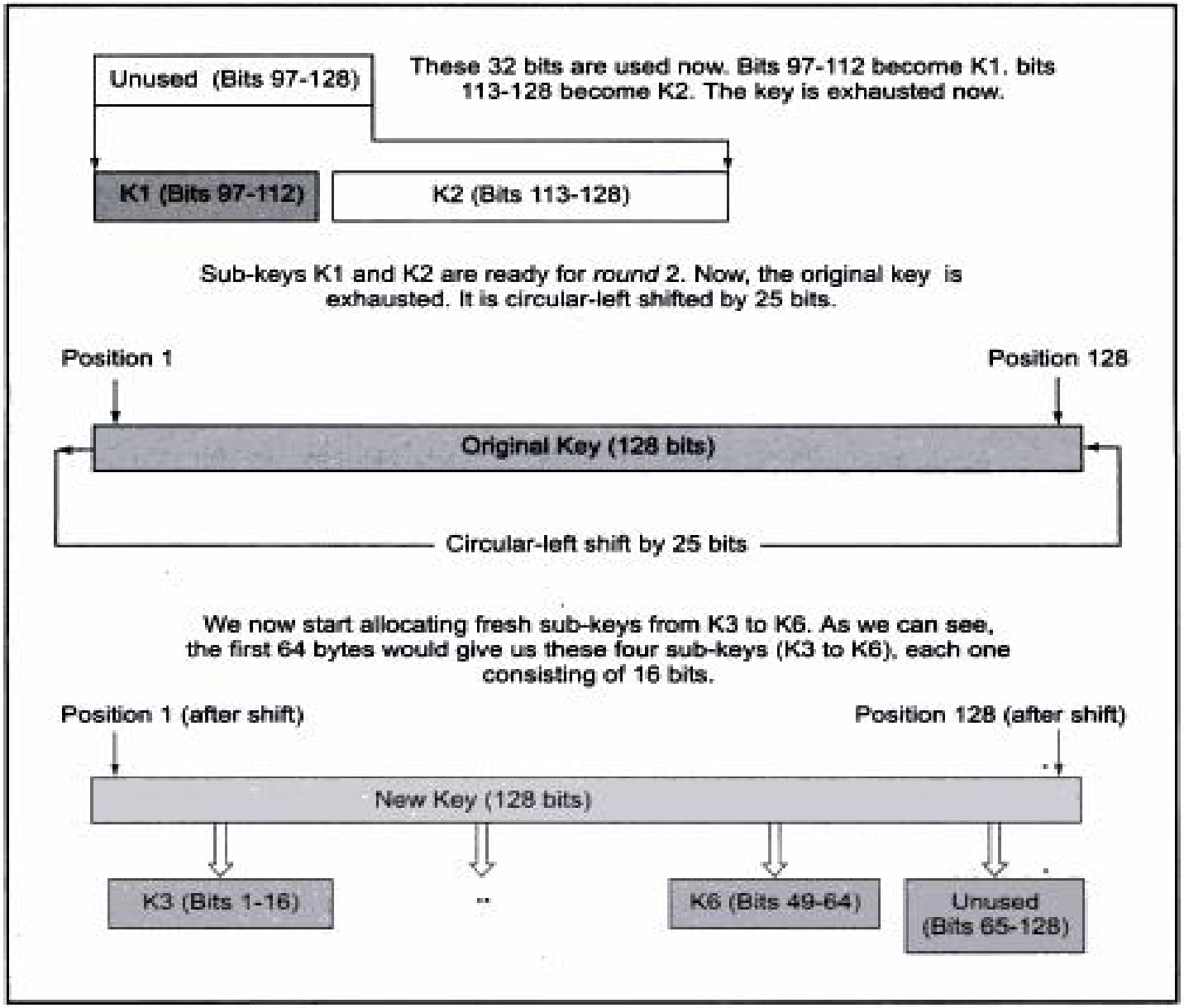
Sub key generation for a round

- As mentioned earlier, each of the eight rounds make use of six sub-keys (so, 8*6 = 48 sub-keys are required for the round) and the final output transformation uses four sub-keys (making a total of 48 + 4= 52 sub-keys overall).
- These 52 sub-keys are generated from an input key of 128 bits.
- The initial key consists of 128 bits, from which 6 sub-keys K1 to K6 are generated for the first round.
- Since K1 to K6 consists of 16 bit each, out of original 128



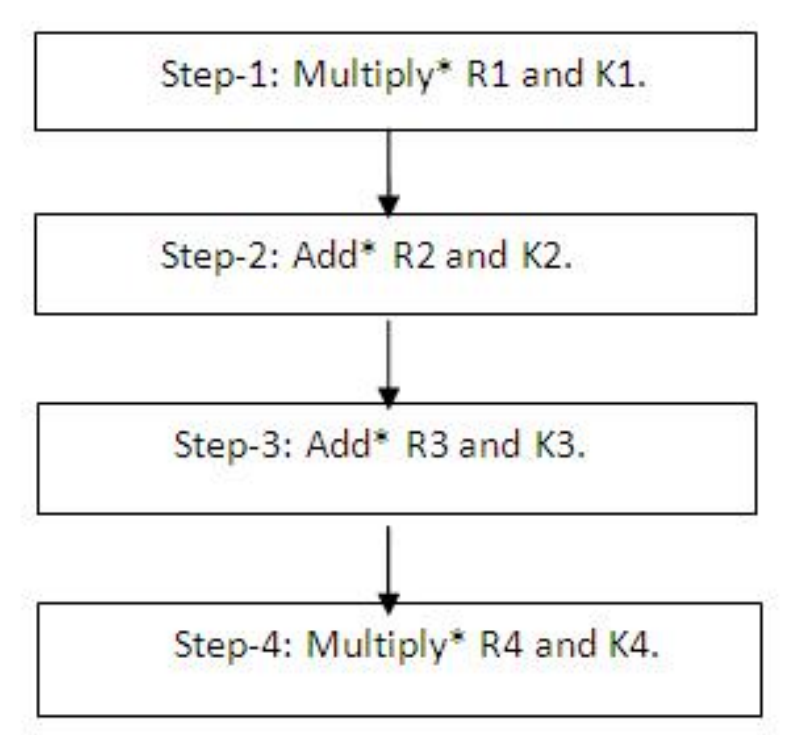
Second Round

- In the second round, firstly, the 32 unused bits (i.e. bits 97-128) of the first round are used.
- As each round requires 6 sub-keys K1 to K6, each of 16 bits, making a total of 96 bits.
- Thus, for the second round we still require (96-32 = 64) more bits.
- However, all the 128 bit of the original key are exhausted.
- For remaining 64 bits IDEA employs the technique of ***key shifting.***
- At this stage, the original key is *shifted left circularly* by 25 bits.


Unused (Bits 97-128)
These 32 bits are used now. Bits 97-112 become K1, bits 113-128 become K2. The key is exhausted now.
K1 (Bits 97-112)
K2 (Bits 113-128)
Sub-keys K1 and K2 are ready for round 2. Now, the original key is exhausted. It is circular-left shifted by 25 bits.
Position 1
Position 128
Original Key (128 bits)
Circular-left shift by 25 bits
We now start allocating fresh sub-keys from K3 to K6. As we can see, the first 64 bytes would give us these four sub-keys (K3 to K6), each one consisting of 16 bits.
Position 1 (after shift)
Position 128 (after shift)
New Key (128 bits)
K3 (Bits 1-16)
..
K6 (Bits 49-64)
Unused (Bits 65-128)

Output Transformation

- The **Output Transformation** is the one time operation. It takes place at the end of the 8th round.



Sub-Key Generation for the Output Transformation

- The process for the sub-key generation for the output transformation is exactly similar to sub-key generation process for the eight rounds.
- At the end of the eighth round, the key was exhausted. Hence, the key is again shifted by 25 bits.
- Post this shift operation, the first 64 bit of the key are taken, and are called as sub-keys K1 to K4 for the final output transformation.

IDEA Decryption & Strength

- The decryption process is exactly the same as encryption process.
- There are some alterations in the generation and pattern of sub-keys.
- The decryption sub-keys are actually inverse of encryption sub-keys
- IDEA uses 128-bit key, which is double than key size of DES. Thus, to break into IDEA, 2^{128} encryption operations are required

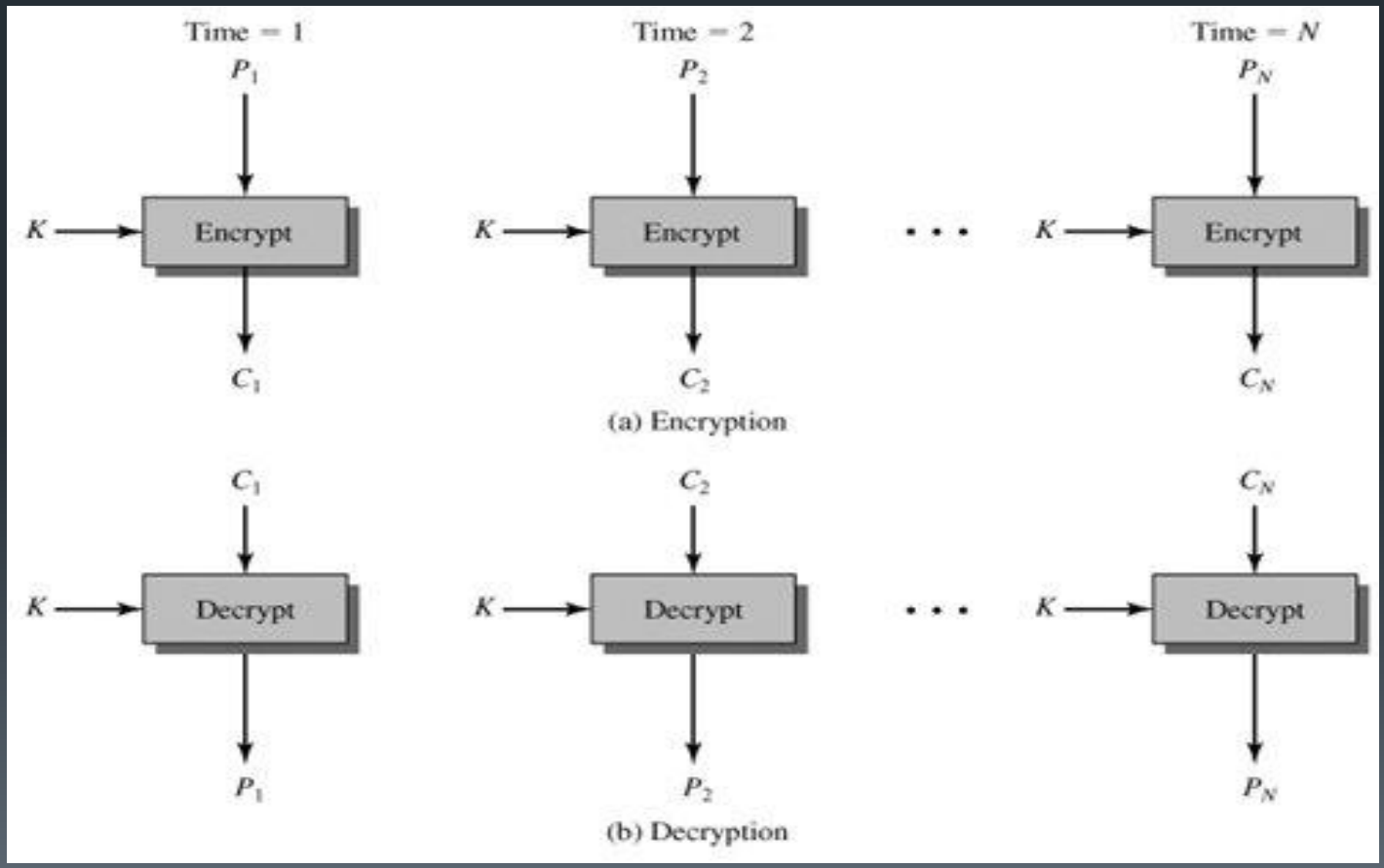
Block Cipher Modes of Operation

- A block cipher algorithm is a basic building block for providing data security. To apply a block cipher in a variety of applications, four "modes of operation" have been defined by NIST.
- Electronic Codebook (ECB)
- Cipher Block Chaining (CBC)
- Counter (CTR)

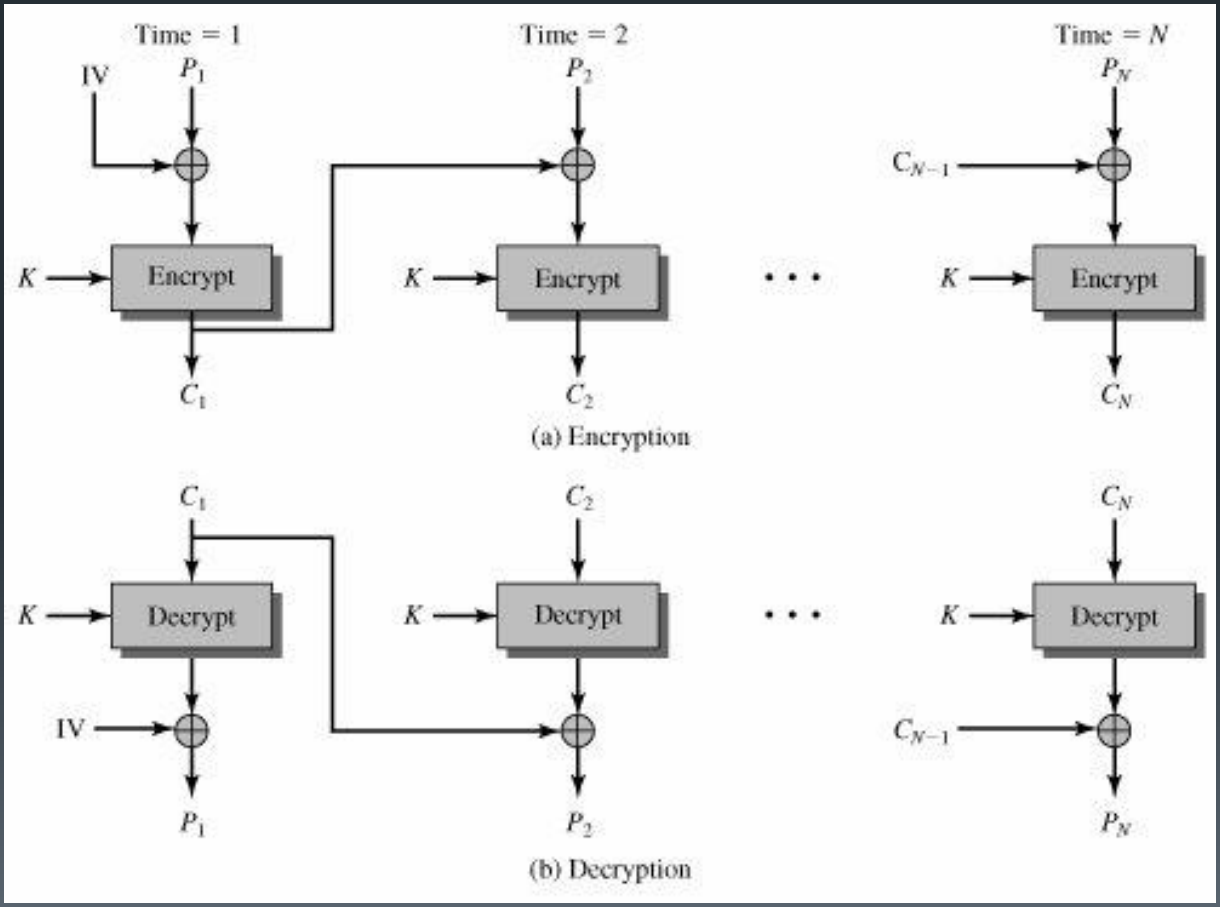
Electronic Codebook (ECB)

- The simplest mode is the electronic codebook (ECB) mode, in which plaintext is handled one block at a time and each block of plaintext is encrypted using the same key
- The term *codebook is* used because, for a given key, there is a unique ciphertext for every *b-bit block of plaintext.*
- *Therefore,* we can imagine a gigantic codebook in which there is an entry for every possible *b-bit plaintext pattern* showing its corresponding cipher text.

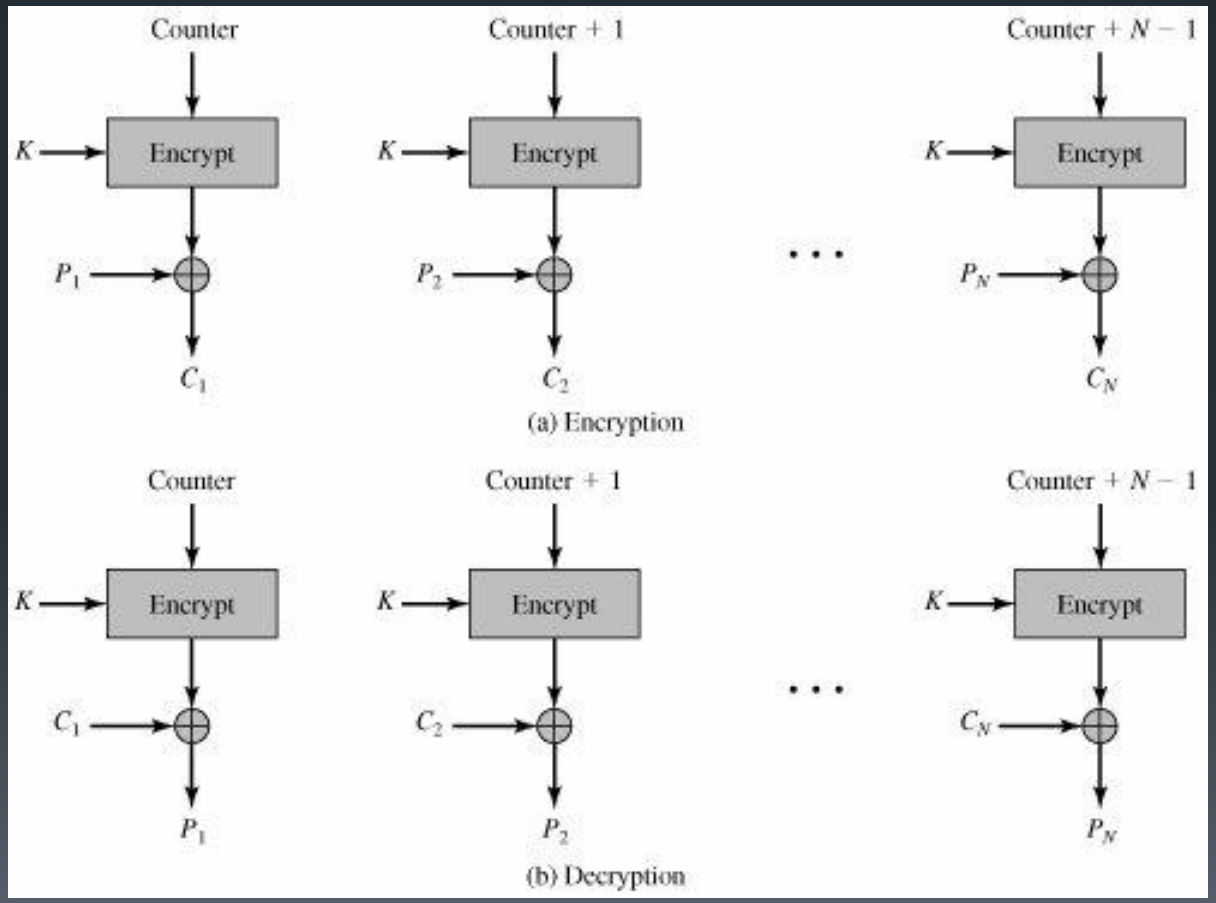
Electronic Codebook (ECB)



Cipher Block Chaining Mode

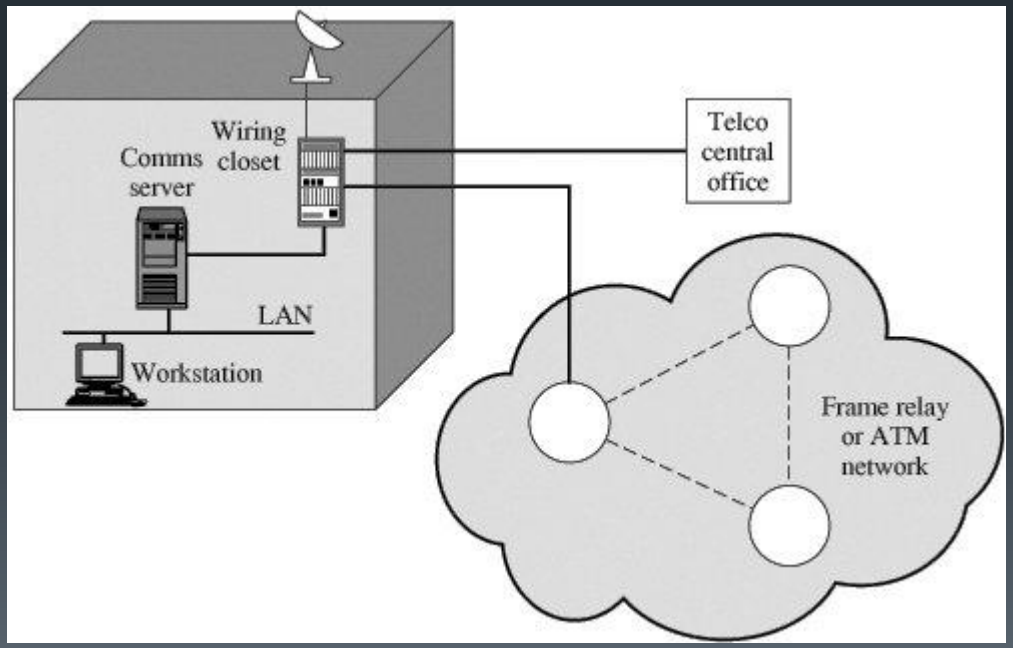


Counter Mode

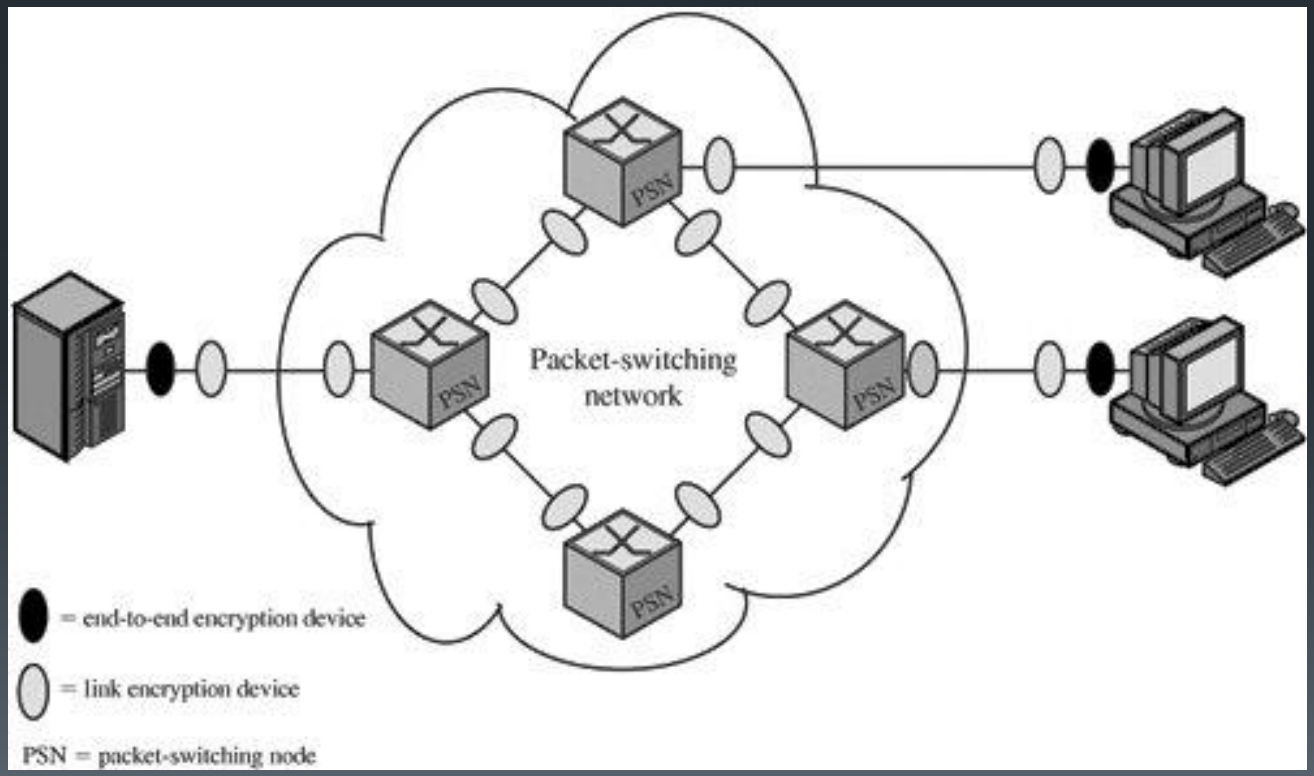


Confidentiality Using Symmetric Encryption

- Placement of Encryption Function



Link versus End-to-End Encryption



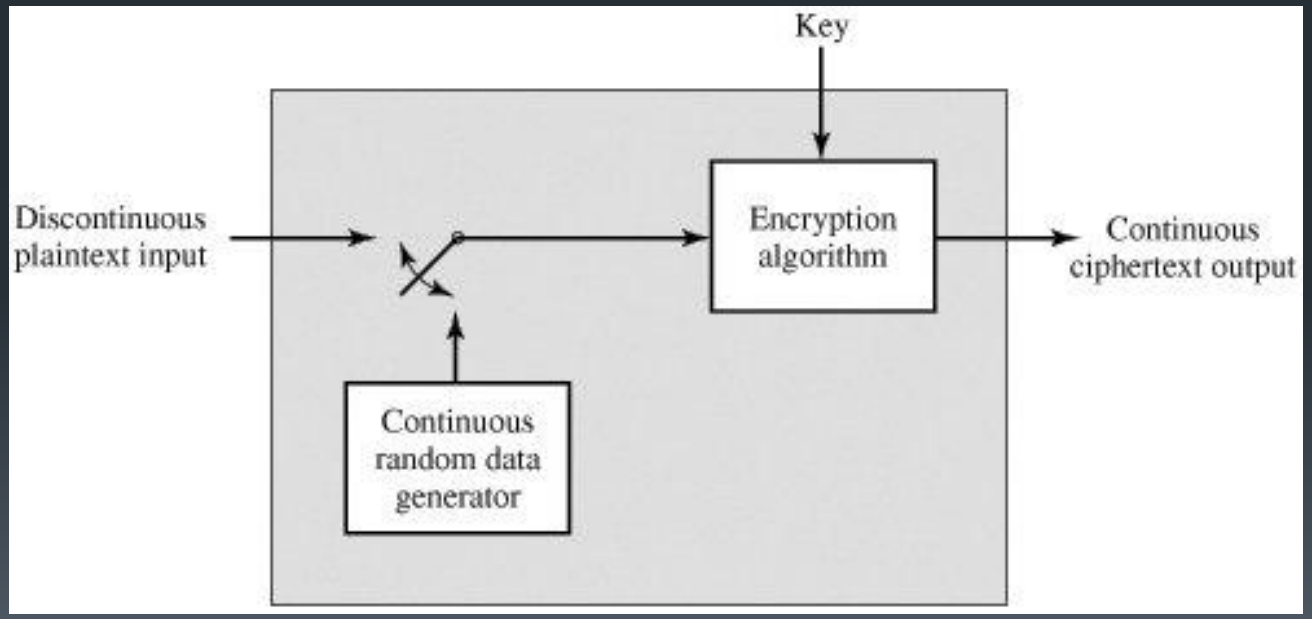
Logical Placement of End-to-End Encryption Function

- With link encryption, the encryption function is performed at a low level of the communications hierarchy. In terms of the Open Systems Interconnection (OSI) model, link encryption occurs at either the physical or link layers.
- For end-to-end encryption, several choices are possible for the logical placement of the encryption function

Traffic Confidentiality

- The following types of information that can be derived from a traffic analysis attack:
  - Identities of partners
  - How frequently the partners are communicating
  - Message pattern, message length, or quantity of messages that suggest important information is being exchanged
  - The events that correlate with special conversations between particular partners

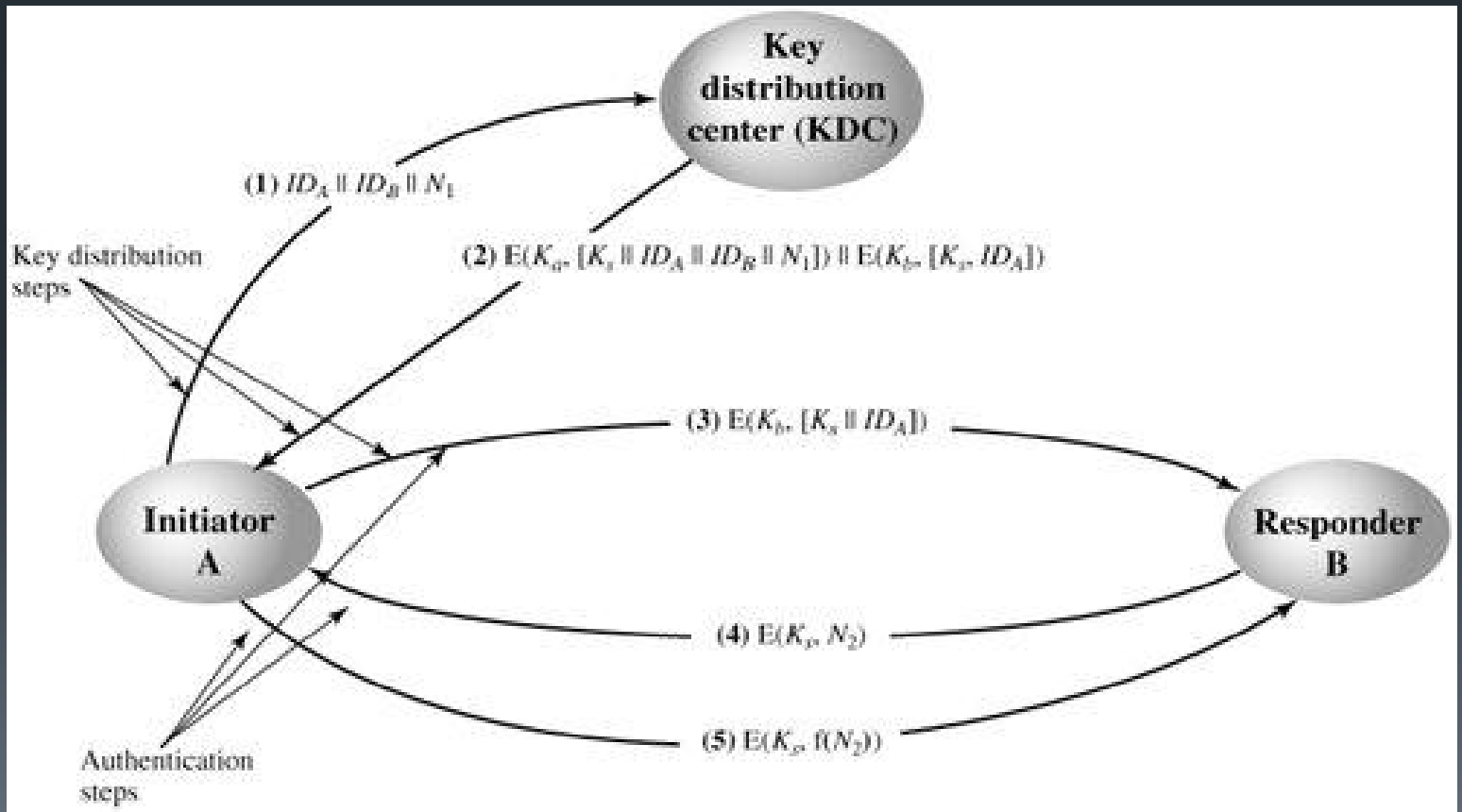
Traffic-Padding Encryption Device



Key Distribution

- The strength of any cryptographic system rests with the *key distribution technique, a term that refers to the means of* delivering a key to two parties who wish to exchange data, without allowing others to see the key.
- For two parties A and B, key distribution can be achieved in a number of ways, as follows:
  - A can select a key and physically deliver it to B.
  - A third party can select the key and physically deliver it to A and B.
  - If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.
  - If A and B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to A and B.

Key Distribution Scenario



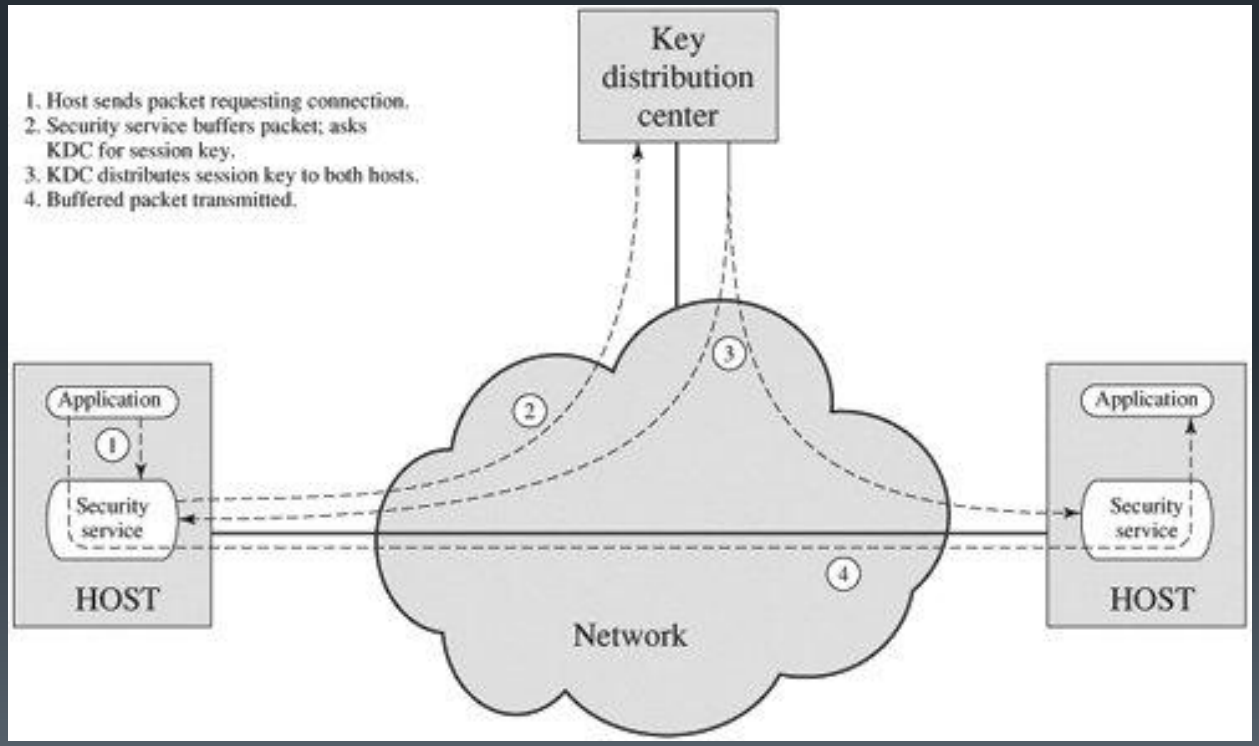
Hierarchical Key Control

- It is not necessary to limit the key distribution function to a single KDC.
- Indeed, for very large networks, it may not be practical to do so. As an alternative, a hierarchy of KDCs can be established.
- For example, there can be local KDCs, each responsible for a small domain of the overall internetwork, such as a single LAN or a single building

Session Key Lifetime

- The more frequently session keys are exchanged, the more secure they are, because the opponent has less ciphertext to work with for any given session key
- On the other hand, the distribution of session keys delays the start of any exchange and places a burden on network capacity
- For connection-oriented protocols, one obvious choice is to use the same session key for the length of time that the connection is open
- For a connectionless protocol , the most secure approach is to use a new session key for each exchange

A Transparent Key Control Scheme

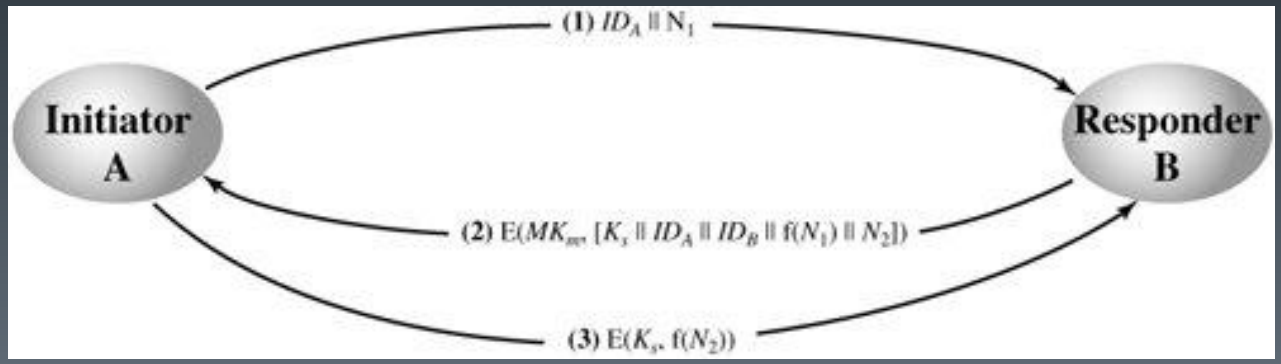


Decentralized Key Control

- The use of a key distribution center imposes the requirement that the KDC be trusted and be protected from subversion.
- This requirement can be avoided if key distribution is fully decentralized.

Decentralized Key Control

- A session key may be established with the following sequence of steps:
  - A issues a request to B for a session key and includes a nonce, *N1*
  - B responds with a message that is encrypted using the shared master key. The response includes the session key selected by B, an identifier of B, the value f(N_1), *and another nonce,* N_2.
  - Using the new session key, A returns f(N_2) *to B.*



Controlling Key Usage

- The concept of a key hierarchy and the use of automated key distribution techniques greatly reduce the number of keys that must be manually managed and distributed.
- It may also be desirable to impose some control on the way in which automatically distributed keys are used
  - Data-encrypting key, for general communication across a network
  - PIN-encrypting key, for personal identification numbers (PINs) used in electronic funds transfer and point-of-sale applications
  - File-encrypting key, for encrypting files stored in publicly

Random Number Generation

- The uses of random number are:
  - To prevent replay attacks
  - Session key generation
  - Generation of keys for RSA algorithm
- These applications give rise to two distinct and not necessarily compatible requirements for a sequence of random numbers: randomness and unpredictability

Randomness

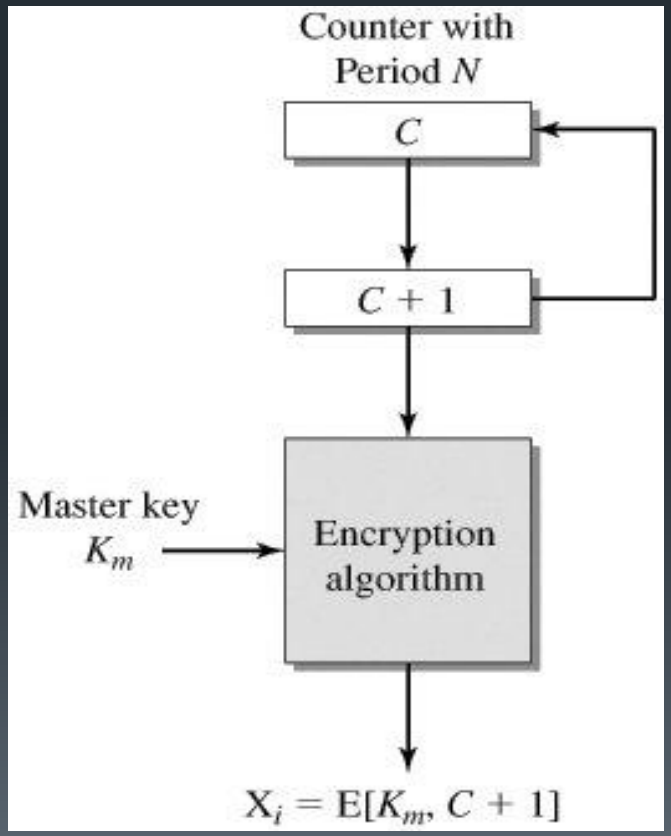
- The following two criteria are used to validate that a sequence of numbers is random:
  - Uniform distribution: The distribution of numbers in the sequence should be uniform; that is, the frequency of occurrence of each of the numbers should be approximately the same.
  - Independence: No one value in the sequence can be inferred from the others.

Unpredictability

- In applications such as reciprocal authentication and session key generation, the requirement is not so much that the sequence of numbers be statistically random but that the successive members of the sequence are unpredictable.
- With "true" random sequences, each number is statistically independent of other numbers in the sequence and therefore unpredictable

Cryptographically Generated Random Numbers

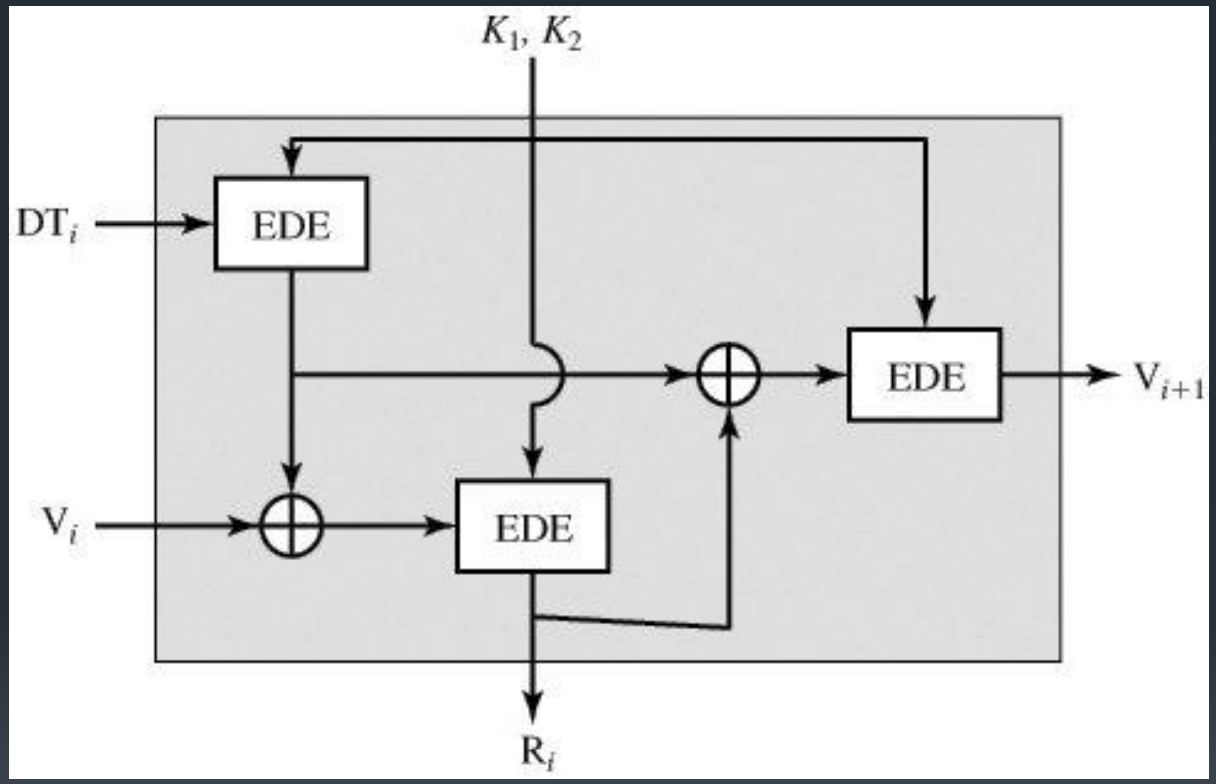
Cyclic Encryption



ANSI X9.17 PRNG

- One of the strongest PRNGs is specified in ANSI X9.17.
- Input: Two pseudorandom inputs drive the generator. One is a 64-bit representation of the current date and time, which is updated on each number generation. The other is a 64-bit seed value; this is initialized to some arbitrary value and is updated during the generation process
- Keys: The generator makes use of three triple DES encryption modules. All three make use of the same pair of 56-bit keys
- Output: The output consists of a 64-bit pseudorandom number and a 64-bit seed value.

ANSI X9.17 PRNG



$$R_i = \text{EDE}([K_1, K_2], [V_i \oplus \text{EDE}([K_1, K_2], DT_i)])$$

$$V_{i+1} = \text{EDE}([K_1, K_2], [R_i \oplus \text{EDE}([K_1, K_2], DT_i)])$$

Blum Blum Shub Generator

- A popular approach to generating secure pseudorandom number is known as the Blum, Blum, Shub (BBS) generator
- The procedure is as follows.
  - First, choose two large prime numbers, *p and q,* that both have a remainder of 3 when divided by 4. That is,
  - Let $n = p \times q$
  - Choose a random number *s, such that s is relatively prime to n*
  - Then the BBS generator produces a sequence of bits B*i* *according to the following* algorithm:

$$X_0 = s^2 \bmod n$$

for i = 1 **to** ∞

$$X_i = (X_{i1})^2 \bmod n$$

$$B_i = X_i \bmod 2$$

Example

i	X_i	B_i
0	20749	
1	143135	1
2	177671	1
3	97048	0
4	89992	0
5	174051	1
6	80649	1
7	45663	1
8	69442	0
9	186894	0
10	177046	0
11	137922	0
12	123175	1
13	8630	0
14	114386	0
15	14863	1
16	133015	1
17	106065	1
18	45870	0
19	137171	1
20	48060	0

n = 192649 = 383 x 503
and the seed s = 101355