CNS - U-2 (Chapter-1)

Mathematics of Asymmetric Key Cryptography ①

objectives

→ To review the concept of algebraic structures
→ To define and give some examples of groups.
→ To define and give some examples of rings
→ To " " " of fields

Algebic structures

→ we already discussed some set of Numbers, such as Z, Z_n, Z_n^*, Z_p and Z_p^*.

Cryptography requires set of Integers and specific operations that are defined for those sets

→ The combination of sets and the operations that are applied to the elements is called An algebraic structure.

→ 3- Algebraic structures.
→ groups
→ rings and
→ fields

Groups.

→ A Group(G) is a set of elements with a binary operation "·" that satisfies four properties.

→ A Commutative group, also called an abelian group, is in which the operator satisfies the four properties

→ Closure of a and b are elements of G, then $c = a \cdot b$ is also an element of G.

This means that the result of Applying the operation on any two elements in the set is another element in the set.

→ Associativity

If a, b and c are elements of G, then

$$(a \cdot b) \cdot c = a \cdot (b \cdot c).$$

→ Commutativity

For all a and b in G, we have

$$a \cdot b = b \cdot a.$$

→ Existence of Identity

For all in G, there exist an element e, called the Identity element, such that

$$e \cdot a = a \cdot e = a.$$

Existence of Inverse: For each a in G, there exists an element a', called the inverse of a, such that

$$a \cdot a' = a' \cdot a = e$$

Ex: Let us define a set $G = \{a, b, c, d\}$, $\cdot$ and the operation is

·	a	b	c	d
a	a	b	c	d
b	b	c	d	a
c	c	d	a	b
d	d	a	b	c

Sol

→ closure is satisfied. Applying the operation on any pair of elements result in another elements in the set.

→ Associativity is also satisfied

Ex: $(a \cdot b) \cdot c = a \cdot (b \cdot c)$

$b \cdot c = a \cdot d$

$d = d$ ∴

→ Commutative.

$a \cdot b = b \cdot a$

$b = b$ ∴

→ ∴

→ Finite Group: A group is called a finite group if the set has a finite Number of elements; otherwise, it is an Infinite groups

→ order of a groups

The order of a group, $|G|$, is the Number of elements in the group.
If the group is not finite, its order is Infinite; if the group is finite, the order is finite.

Subgroups

A subset H of a group G is a subgroup of G if H itself is a group with respect to the operation on G.

In otherwords; if $G=<S, \cdot>$ is a group, $H=<T, \cdot>$ is a group under the same operation, and T is a Nonempty subset of S, then H is a subgroup of G.

→ If a and b are Members of both groups, Then $c=a \cdot b$ is also a Member of both groups.

→ the group share the same Identity element.

→ If a is a member of both groups, the inverse of a is also a Member of both groups

→ The group made of the Identity element of G, $H = \langle \{e\}, \cdot \rangle$, is a subgroup of G.

→ Each group is a subgroup of itself.

Ex:

is the Group $H = \langle Z_{10}, + \rangle$ a subgroup of the group $G = \langle Z_{12}, + \rangle$?

Sol. The answer is NO.. Although H is a subset of G, the operations defined for these two groups are different.

Cyclic subgroups.

If a subgroup of a group can be generated using the power of an element, the subgroup is called the cyclic subgroup

$a^n \rightarrow a \cdot a \cdot a \cdot \ldots \cdot a$ (n times)

$a^0 = e$

Note

Ex. Four cyclic groups can be made from the group $G = \langle Z_6, + \rangle$. They are $H_1 = \langle \{0\}, + \rangle$, $H_2 = \langle \{0, 2, 4\}, + \rangle$, $H_3 = \langle \{0, 3\}, + \rangle$, and $H_4 = G$.

→ When the operation is addition, a^n means multiply n by a.. The operation is addition Modulo 6.

Ex. The following show how we find the elements in cyclic group

a) The cyclic subgroup generated from 0 is H_1 which has only one element, 0

$0^0 \bmod 6 = 0$.

b) The cyclic subgroup generated from 1 is H_4, which is G itself?

~~{0, 1, 2, 3}~~

$1^0 \bmod 6 = 0$

$1^1 \bmod 6 = 1$

$1^2 \bmod 6 = (1+1) \bmod 6 = 2$

$1^3 \bmod 6 = (1+1+1) \bmod 6$

$= 3 \bmod 6 = 3$

$1^4 \bmod 6 = (1+1+1+1) \bmod 6$

$= 4 \bmod 6 = 4$

$1^5 \bmod 6 = 5 \bmod 6 = 5$.

(the process will be repeated)

c) The cyclic subgroup generated from 2 is H_2, which has three elements: 0, 2, & 4.

$2^0 \bmod 6 = 0$

$2^1 \bmod 6 = 2$

$2^2 \bmod 6 = 4$ (The process will be repeated)

d) The cyclic subgroup generated from 3 is H_3, which has two elements: 0 & 3.

$3^0 \bmod 6 = 0$

$3^1 \bmod 6 = 3$ (Stop: the process will be repeat

(e) The cyclic subgroup generated from 4 is H_2:

$4^0 \bmod 6 = 0$

$4^1 \bmod 6 = 4$

$4^2 \bmod 6 = (4+4) \bmod 6$ (The process will be repeated)

$= 2$

$4^3 \bmod 6$

$\Rightarrow 12 \bmod 6$

$= 0$

(f) The cyclic subgroup generated from 5 is H_1, which is G itself.

$5^0 \bmod 6 = 0$

$5^1 \bmod 6 = 5$

$5^2 \bmod 6 = 4$

$5^3 \bmod 6 = 3$

$5^4 \bmod 6 = 2$

$5^5 \bmod 6 = 1$

* Ex.

→ Three cyclic subgroups can be made from the group $G = \langle Z_{10}^*, \times \rangle$. G has only four elements. 1, 3, 7 and 9. The cyclic subgroups are $H_1 = \langle \{1\}, \times \rangle$, $H_2 = \langle \{1, 9\}, \times \rangle$, and $H_3 = G$. The following show how we find the elements of these sub-groups

(a) The cyclic subgroup generated from 1 is H_1 The subgroup has only one element, the Identity element.

$1^0 \bmod 10 = 1$. (The process will be repeated)

(b) The cyclic subgroup generated from 3 is H_3, which is G itself.

$3^0 \bmod 10 = 1$

$3^1 \bmod 10 = 3$

$3^2 \bmod 10 = 9$

$3^3 \bmod 10 = 7$. (The process will be repeated)

(c) The cyclic subgroup generated from 7 is H_3, which is G itself.

$7^0 \bmod 10 = 1$

$7^1 \bmod 10 = 7$

$7^2 \bmod 10 = 9$

$7^3 \bmod 10 = 3$ (The process will be repeated)

(d) The cyclic subgroup generated from 9 is H_2. The subgroup has only 2-elements

$9^0 \bmod 10 = 1$

$9^1 \bmod 10 = 9$

(The process will be repeated)

Ring

→ A Ring, denoted as $R = \langle \{\dots\}, \cdot, \square \rangle$, is an algebraic structure with 2-operations. The first operation must satisfy all five properties. The 2nd operation must satisfy only the first two.

→ Distributivity means that for all a, b and c elements of R, we have

$$a \square (b \cdot c) = (a \square b) \cdot (a \square c) \text{ and}$$

$$(a \cdot b) \square c = (a \square c) \cdot (b \square c)$$

→ $\{a, b, c\}$ set. $\cdot \square$ Ring operations

Field.

A field, denoted by $F = \langle \{\dots\}, \cdot, \square \rangle$ is a commutative ring in which the 2nd operation satisfies all five properties.

Distribution of $\square$ over $\cdot$

$\cdot$	$\square$
1. Closure	1. Closure
2. Associativity	2. Associativity
3 Commutativity	3. Commutativity
4 Existence of Identity	4. Existence of Inverse
5. ... Inverse	5. Existence of Inverse
$\forall a, b, c \in$ Set	$\cdot \square$ operations

Galois field GF(P) fields

GF(P) fields, when $n=1$, we have GF(P) field. This field can be the set $Z_p, \{0, 1, \ldots, P-1\}$, with 2 – arithmetic operations.

Ex.

A very common field in this category is GF(2) with set $\{0,1\}$ and two operations, add and mul:

GF(2)

$\{0,1\}$ $+ \times$

+	0	1
0	0	1
1	1	0

Addetion

×	0	1
0	0	0
1	0	1

Multiplicate

a	0	0	a	0	1
-a	1	1	a⁻¹	~~2~~	~~1~~

Inverse

Ex We can define GF(5) on the set Z_5 (5 is a prime) with addetion & multiplication

GF(5)

$\{0,1,2,3,4\}$ $+ \times$

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

Add

X	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
-2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

MUl

a	0	1	2	3	4
a⁻¹	0	4	3	2	1

a	0	1	2	3	4
a⁻¹	-1	3	2	4	

GF(p^n) fields

Inaddelia to GF(p) fields; we are also Interested in GF(p^m) fields in Cryptography however, the set Z_i, Z_n, Z_p^* and Z_p, which we have used so far with operations such as Addetia, substraction & mul...

GF(2^n) fields

→ we have 4-operations (Add, Sub, Mul and div).

→ when we work with computer, the the Integers are stored in the Computer as n-bit words. in which n is usually 8, 16, 32, 64...

→ This means that the range of Integers is 0 to $2^n - 1$.

we have 2-choices if we want to use a field.

(1) we can use GF(p) with the set Z_p, where p is the largest prime Number less than 2^n. Although this scheme works, it is Inefficient because we can't use the Integers from p to $2^n - 1$.

Ex of n=4, the largest prime Number less than 2^4 is 13... This means we cant use Integers 13, 14, 15.

→ we can work in $GF(2^n)$ and uses a set of 2^n elements. The elements in this set are n-bit words.

Ex if n=3, the set is

{000, 001, 010, 011, 100, 101, 110, 111}

Polynomials

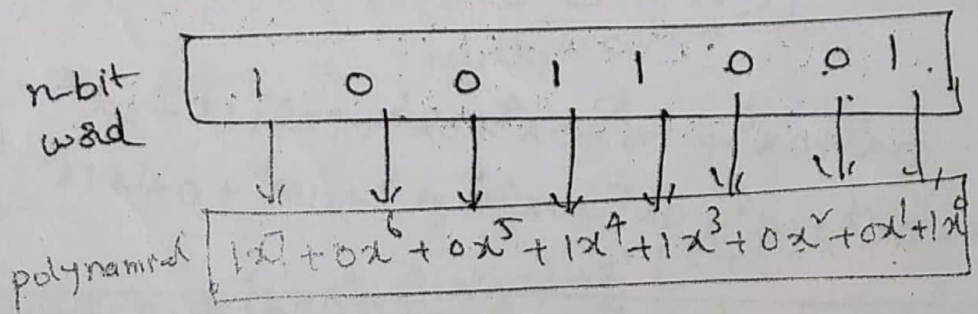
→ A polynomial of degree n-1 is an expression of the form

$$f(x) = a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \cdots + a_1x^1 + a_0x^0$$

where x^i is called i^{th} term and a_i is called coefficient of the i^{th} term.

Ex.

The following shows how we can represent the 8-bit word (10011001) using a polynomial.



First simplification

$$1x^7+1x^4+1x^3+1x^0$$

Second simplification

$$x^7+x^4+x^3+1$$

Ex

To find the 8-bit word related to the polynomial x^5+x^2+x, we first supply the omitted terms. Since n=8; it means the polynomial is of degree 7. The expanded polynomial is

$$0x^7+0x^6+1x^5+0x^4+0x^3+1x^2+1x^1+0x^0$$

This is related to the 8-bit word

00100110

Addition

Let us do $(x^5+x^2+x) \oplus (x^3+x^2+1)$ in $GF(2^8)$

$$0x^7+0x^6+1x^5+0x^4+0x^3+1x^2+1x^1+0x^0$$
$$\oplus\ 0x^7+0x^6+0x^5+0x^4+1x^3+1x^2+0x^1+1x^0$$

$$0x^7+0x^6+1x^5+0x^4+1x^3+0x^2+1x^1+1x^0.$$

$\rightarrow$ $x^5+x^3+x+1.$

Additive Identity

The additive identity in a polynomial is a zero polynomial because adding a polynomial with itself results in a zero polynomial

Additive Inverse

The additive inverse of a polynomial with coefficients in GF(2) is the polynomial itself.

Multiplication : Multiplication in polynomials is the multiplication of each term of the first polynomial with each term of the second

Ex Find the result of $(x^5+x^4+x) \otimes (x^7+x^4+x^3+x^2+x)$ in GF(2^8) with Irreducible polynamial $(x^8+x^4+x^3+x+1)$.. Note that we use the Symbol $\otimes$ to show the Multiplication of 2- polynamial

Sol

$$P_1 \otimes P_2 = x^5(x^7+x^4+x^3+x^2+x) + x^4(x^7+x^4+x^3+x^2+x) + x(x^7+x^4+x^3+x^2+x)$$

$$P_1 \otimes P_2 = x^{12}+x^9+x^8+x^7+x^6+x^5+x^4+x^3+x^8+x^5+x^4+x^3+x^2$$

$P_1 \otimes P_2$

$= (x^{12}+x^7+x^2) \bmod (x^8+x^4+x^3+x+1)$

$\Rightarrow x^5+x^3+x^2+x+1$

Multiplicative Identity.

The multiplicative Identity is always 1.

Ex In GF(2^8), the multiplicative Inverse of the bit pattern

00000001.

$= x^4 + x^3 + 1 - (x^8 + x^6 + x^5 + x^3 + x^2 + x)$

$= x^4 + x^3 + 1 - x^8 + x^6 + x^5 + x^3 + x^2 + x$

$= x^8 + x^4 + x^3 + x + 1$

Ex Find the result of Multiplying $P_1=(x^5+x^2+x)$ by $P_2=(x^7+x^4+x^3+x^2+x)$ in GF(2^8) with Irreducible polynomial $(x^8+x^4+x^3+x+1)$ ~~using the alg~~.

Sol

We first find the partial result of multiplying x^0, x^1, x^2, x^3, x^4 and x^5 by P_2. Note that only three terms are Needed..

Powers	operation	New Result	Reduction
$x^0 \otimes P_2$		$x^7+x^4+x^3+x^2+x$	No
$x^1 \otimes P_2$	$x \otimes (x^7+x^4+x^3+x^2+x)$	x^5+x^2+x+1	yes
$x^2 \otimes P_2$	$x \otimes (x^5+x^2+x+1)$	$x^6+x^3+x^2+x$	No
$x^3 \otimes P_2$	$x \otimes (x^6+x^3+x^2+x)$	$x^7+x^4+x^3+x^2$	No
$x^4 \otimes P_2$	$x \otimes (x^7+x^4+x^3+x^2)$	x^5+x+1	yes
$x^5 \otimes P_2$	$x \otimes (x^5+x+1)$	x^6+x^2+x	No

$x^8+x^4+x^3+x+1 \overline{)\, x^8+x^5+x^4+x^3+x^2}\ (1$

$x^8+x^4+x^3+x+1$

x^5+x^2+x+1

$x \otimes P_2$

$x^2 \otimes P_2 \Rightarrow x \times (x^5+x^2+x+1)$

$\Rightarrow x^6+x^3+x^2+x$

$x^3 \otimes P_2 = x(x^6+x^3+x^2+x)$

$= x^7+x^4+x^3+x^2$

$x^4 \otimes P_2 = x(x^7+x^4+x^3+x^2$

$= x^8+x^5+x^4+x^3$

$x^5 \otimes P_2 = x(x^5+x^2+x)$

$\Rightarrow x^6+x^3+x^2$

$x^8+x^4+x^3+x+1 \;\big)\; x^8+x^5+x^4+x^3+1$

x^8+x^4+x+1

---

x^5+x+1

$x^5 \otimes P_2 = x(x^5+x+1)$

$= x^6+x^2+x$

$P_1 \times P_2 =$

$x^1 \Rightarrow x^5+x^2+x+1$

$x^2 \Rightarrow x^6+x^3+x^2+x$

$x^5 \Rightarrow x^6+x^2+x$

$\rightarrow$

x^5+x^2+x+1

$x^6+x^3+x^2+x$

x^6+x^2+1

$x^5+x^3+x^2+x+1$

U-2 (Chapter-2)

Introduction to Modern Symmetric-Key Ciphers

* objectives are

→ To distinguish b/w tradetional and Modern Symmetric-Key Ciphers

→ To Introduce Modern block ciphers

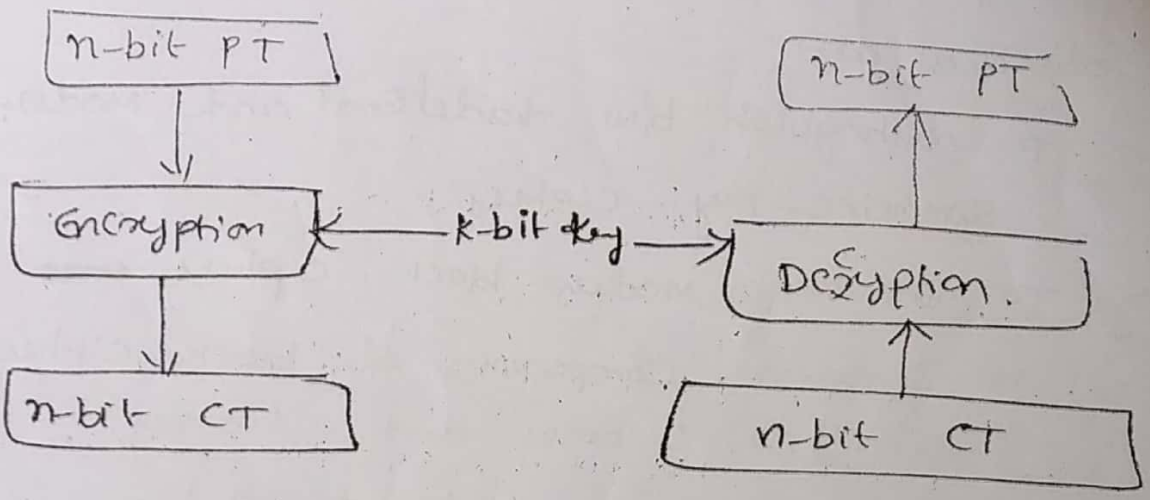
→ To Introduce Components of block ciphers such as D-boxes and S-boxes

↔ Fiestel and Non-Fiestel Ciphers.

→ The tradetional symmetric-key ciphers are Character-diented Ciphers, with the advent of computers, we need bit-diented ciphers. This is because the information (Numbers and letters) is convenient to convert these types of data into a stream of bits.

→ A Symmetric-key block cipher encrypt an n-bit block of PT & decrypt an n-bit block of CT.

→ The Encryption & Decryption alg uses a K-bit key. The decryption alg must be the inverse of the Encryption alg, and the both operation must use the same secret key.



→ If the msg is fewer than n bits, padding must be added to make it an n-bit block; if the msg has more than n-bits, it should be divided into n-bit blocks and the appropriate padding must be added to the last block if necessary

→ The common values of n are 64, 128, 256 and 512 bits

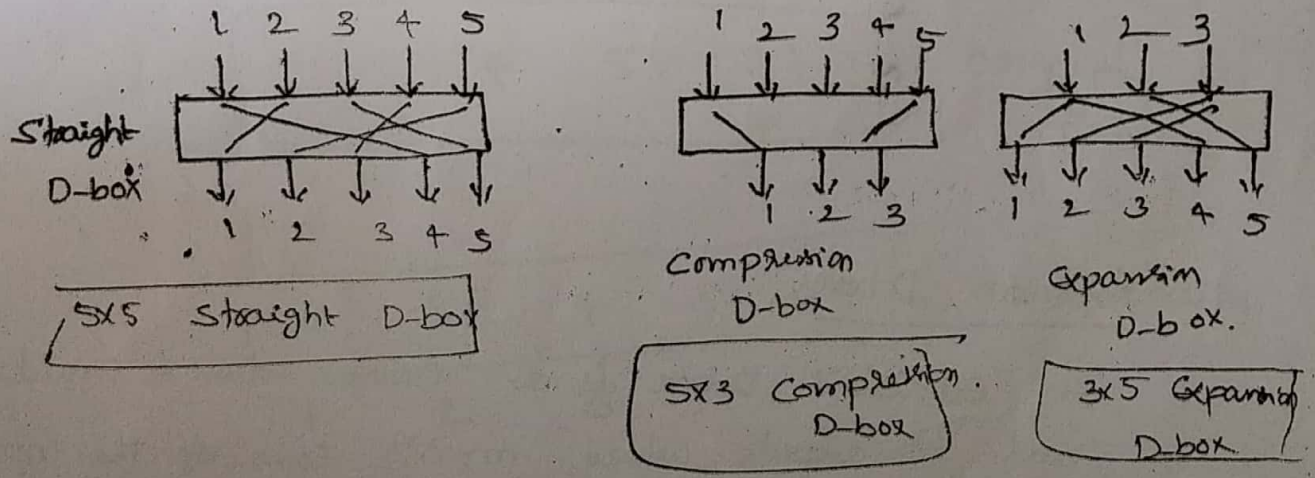
Components of a Modern Block Cipher

A Modern block cipher is made of a combination of transposition units for diffusion (called D-box), substitution units (called S-boxes).

D-box

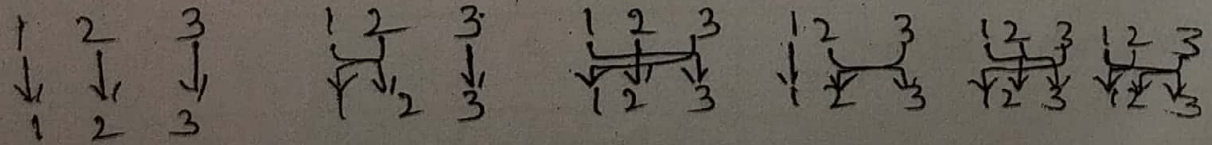
→ There are three types of D-boxes in modern block ciphers.

- Straight D-boxes
- Expansion D-boxes
- Compression D-boxes.



Straight D-boxes :-

A straight D-box with n inputs and n outputs is a permutation. There are n! possible mappings.



→ D-boxes are Normally keyless, which means that the mapping is predetermined.

Compression D-boxes:

It is a D-box with n inputs & m outputs where $m < n$. Some of the inputs are blocked and do not reach the output.

→ We need to know that a table for a compression D-box has m entries; but the content of each entry is from 1 to n with some missing values.

Expansion D-boxes:

An expansion D-box is a D-box with n inputs and m outputs where $m > n$. Some of the input are connected to more than one input.

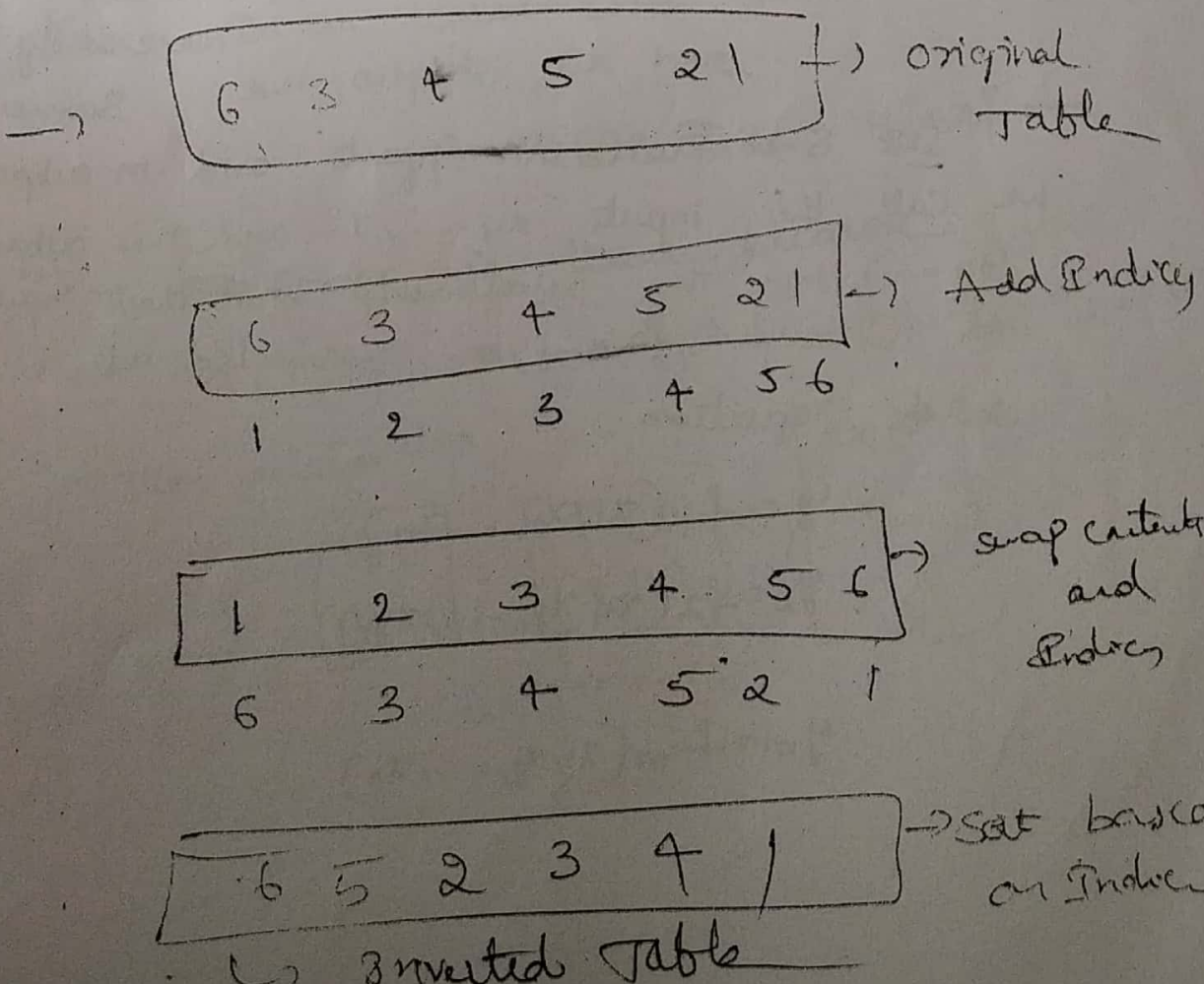
→ We need to know that a table for an expansion D-box has m entries, but m-n of the entries are repeated.

Invertibility

→ A straight D-box is Invertable. This means that we can use a straight D-box in the Encryption cipher and its Inverse in the Decryption.

Steps

→ 1. original table
→ 2. Add Indices.
→ 3. swap Contents and Indices
→ 4. Sort Based on Indices
→ 5. Inverted Table.



→ There is No Inverse for compression and expansion D-boxes.

S-Boxes..

An S-box can have a different No. of inputs and outputs. In other words, the input to an S-box could be an n-bit word, but the output can be m-bit word, where m and n are Not Necessarily the Same...

There are 2 S-Boxes.

1. Linear S-boxes
2. Nonlinear S-boxes

→ In S-box with n inputs and m outputs we call the inputs $x_1, \ldots, x_n$ and the outputs $y_1, \ldots, y_m$. The relationship b/w the inputs and the outputs can be represented as a set of Equation.

$$y_1 = f_1(x_1, x_2, \ldots x_n)$$

$$y_2 = f_2(x_1, x_2, \ldots, x_n)$$

$$\ldots\ldots\ldots$$

$$y_m = f_m(x_1, x_2, \ldots, x_n)$$

In a linear S-box, the above relations can be expressed as

$$y_1 = a_{1,1}x_1 \oplus a_{1,2}x_1 \oplus \cdots \oplus a_{1,n}x_n$$

$$y_2 = a_{2,1}x_1 \oplus a_{2,2}x_2 \oplus \cdots \oplus a_{2,n}x_n$$

$$\vdots$$

$$y_m = a_{m,1}x_1 \oplus a_{m,2}x_1 \oplus \cdots \oplus a_{m,n}x_n.$$

In a Non-linear S-box, we can't have the above relations for every output.

Ex: In an S-box with 3-three inputs and two outputs, we have

$$y_1 = x_1 \oplus x_2 \oplus x_3 \qquad y_2 = x_1$$

1) The S-box is linear because

$$a_{1,1} = a_{1,2} = a_{1,3} = a_{2,1} = 1 \quad \text{and} \quad a_{2,2} = a_{2,3} = 0.$$

The relationship

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Ex In a S-BOX with three inputs and two outputs, we have

$y_1 = x_1 x_2 \qquad y_2 = x_1 + x_2 x_3$

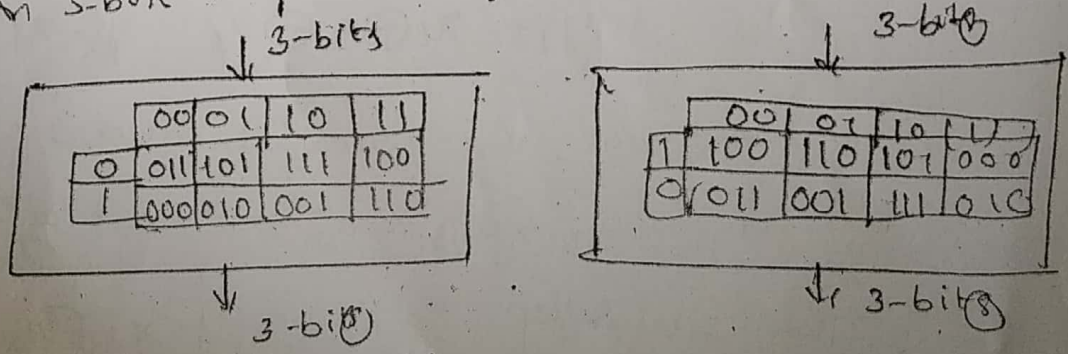
Sol The S-box is Non linear because there is No linear relationship b/w the inputs and outputs.

Invertability

In S-boxes are substitution ciphers in which the relationship b/w input and output is defined as a table.

An S-box may or may not be Invertible

Ex



In each table, the leftmost bit of the input defines the row & the next two bits defines the column.

Ex if the input to the left box 001, the o/p is 101.

The input 101 in the right table creates the output 001, which show that the I-O ... Invertible ...

Exclusive-or

The Most Important component in block ciphers is the exclusive-or operation.

properties

The five properties of the exclusive-or operation in the $GF(2^n)$ field makes this operation ~~possible~~.

1. Closure: This property guarantees that the result of exclusive-oring two n-bit words is another n-bit word.

2. Associativity: This property allows us to use more than one exclusive-or operator in any order.

$$x \oplus (y \oplus z) \leftrightarrow (x \oplus y) \oplus z$$

3. Commutativity: This property allows us to swap the inputs without affecting the output.

$$x \oplus y \leftrightarrow y \oplus x$$

4. Existence of Identity: The Identity element for the exclusive-or operation is an n-bit word that consist of all 0's, a (00...0). This implies that Exclusive

$$x \oplus (00\ldots0) = x$$

5. Existance of Identity.

On $GF(2^n)$ field, each word is the additive Inverse of itself.

$$x \oplus x = (00\ldots0)$$

Complement

The complement operation is a unary operation that flips each bit in a word.

A 0-bit is changed to a 1-bit; a 1-bit is changed to a 0-bit.

If $\bar{x}$ is the complement of x, then the following two relations hold.

$$x \oplus \bar{x} = (11\ldots1) \text{ and } x \oplus (11\ldots1) = \bar{x}$$

Circular shift:

→ Another component found in some Modern block ciphers is the circular shift operation.

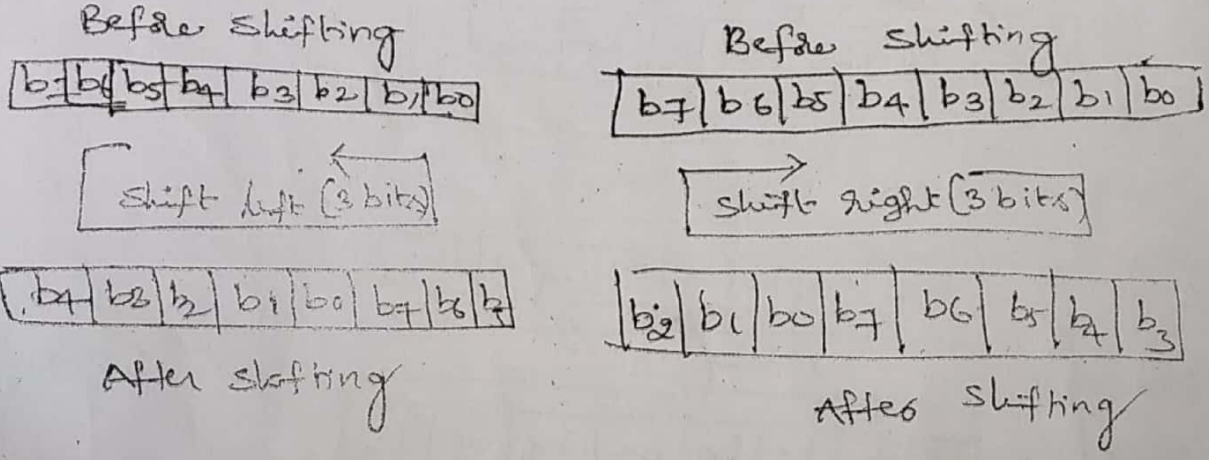
→ Shifting can be to the left or to the right

→ The circular left-shift operation shifts each bit in an n-bit word k positions to the left; the leftmost k bits are removed from left and become the rightmost bits.

→ The circular right-shift operation shifts each bit in an n-bit word k positions to the right; the rightmost k bits are

removed from the right and become the leftmost bits.

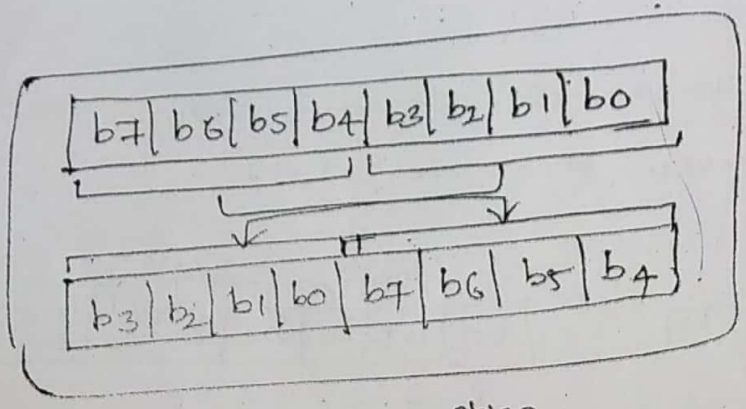
The following shows both left and right operations in the case where $n=8$ and $k=3$.



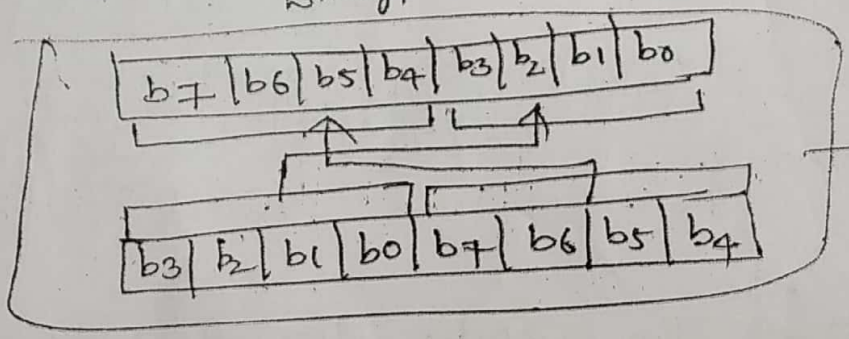
Invertibility: The circular left-shift operation is the inverse of the circular right-shift operation. If one is used in the encryption cipher, the other can be used in the decryption cipher.

Swap

The swap operation is a special case of the circular shift operation where $k = n/2$. This means, this operation is valid only if n is even number because left-shifting $n/2$ bits is the same as right-shifting $n/2$.



Encryption



→ Decryption.

Product ciphers.

→ Shannon Introduced the concept of a product cipher. A product cipher is a complex cipher combining substitution, permutation.

Diffusion and confusion.

→ The Idea of diffusion is to hide the relationship b/w the ciphertext and the plaintext

→ Diffusion Implies that each symbol (character) bi in the ciphertext is dependent on some of all symbols in the plain text.

→ The idea of confusion is to hide the relationship b/w the ciphertext and key.. If a single bit in the key is changed, most (if) all bits in the ciphertext will also be changed.

Rounds

→ Diffusion and confusion can be achieved using Iterated product cipher where each Iteration is a combination of S-boxes, D-boxes and other components

→ The block cipher uses a key schedule (or) key generator that creates different keys for each round from the cipher key.

→ In an N-round, the PT is encrypted N times to create the CT; the CT is decrypted N times creates the PT.

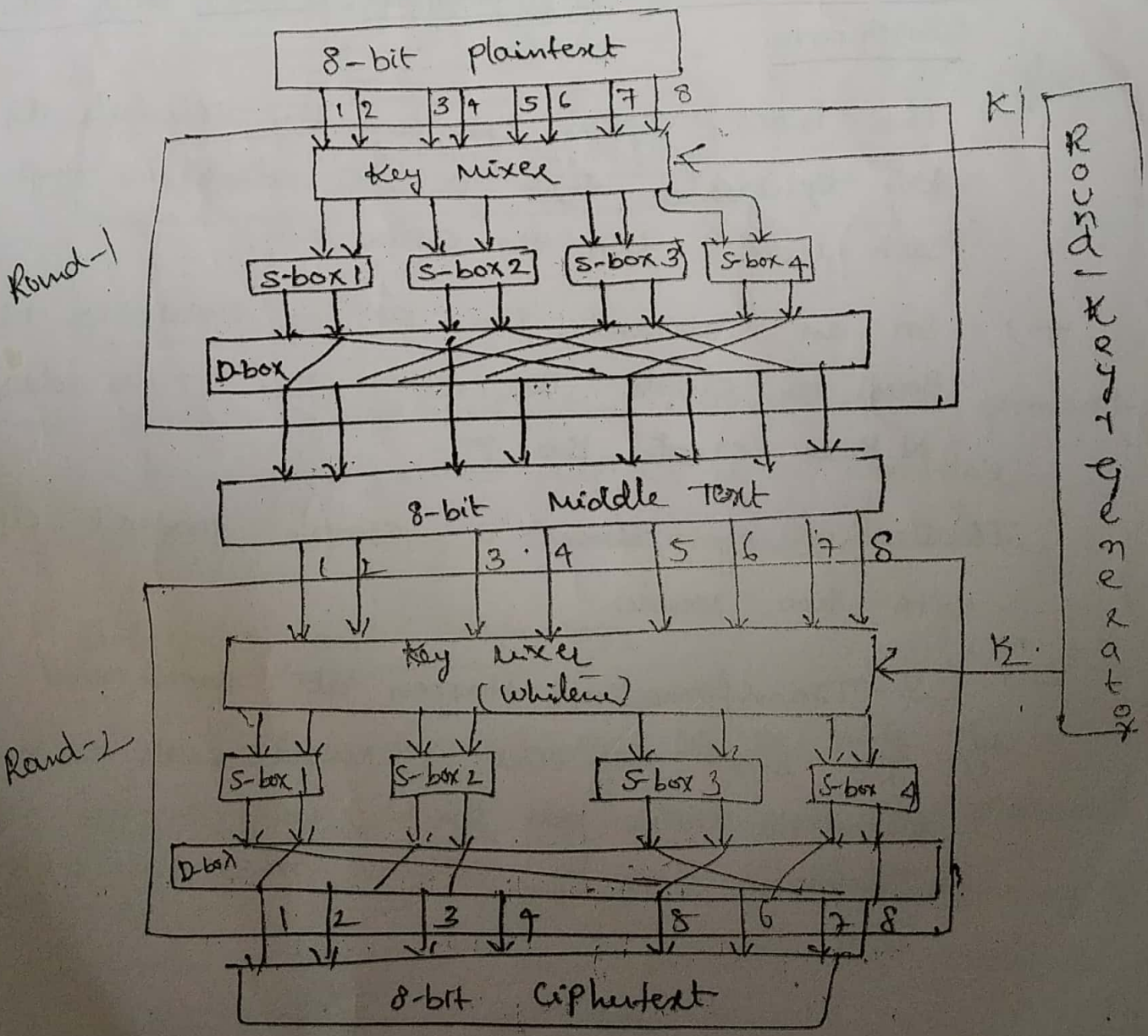
The following shows a simple product cipher with two rounds.

→ 3-Transformations Happen at each round.

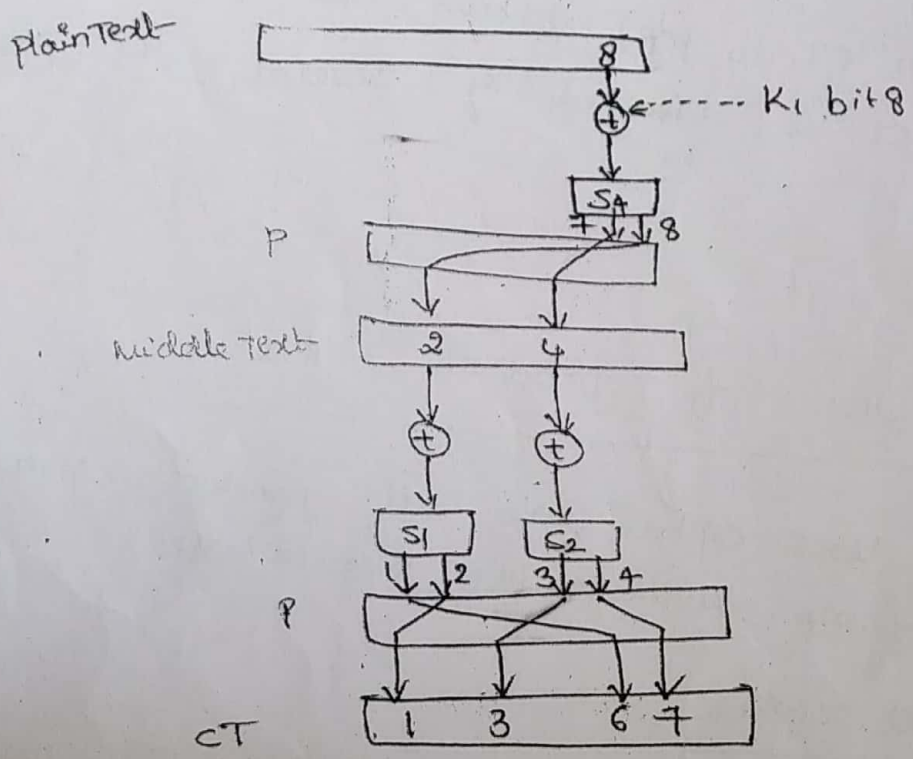
1) The 8-bit text is mixed with the key to whiten the text (Hide the bits using the key). This is normally done by exclusive-oring the 8-bit word with the 8-bit key.

2) The output of the whitener are organized into four 2-bit groups and are fed into four S-boxes. The values of bits are changed based on the structure of the S-boxes in this transformation

3) The o/p of S-boxes are passed through a D-box to transpose the bits so that in the next round each box receives diff inputs.



The following shows how changing single-bit in the PT affects many bit(s) in the CT.



→ In the first round, bit 8, after being Exclusive-ored with the corresponding bit of K1, affects two bits (bit 7 & 8) through S-box 4. Bit 7 is permuted and becomes bit 2; bit 8 is permuted and becomes bit 4. After the first round, bit 8 has affected bit 2 and 4.

→ In the 2nd round, bit 2, after being Exclusive-Ored with the corresponding bit of K2, affects two bits (bits 1 and 2) through S-box 1.

Bit 1 is permuted and becomes bit 6, bit 2 is permuted and becomes bit 1. Bit 4, after being exclusive-Ored with the corresponding bit in K2, affects bit 3 and 4. Bit 3 remains the same; bit 4 is permuted and becomes bit 7. After the second round, bit 8 has affected bits 1, 3, 6 and 7.

→ Going through these steps in the other direction (from CT to PT) shows that each bit in the CT is affected by several bits in the PT.

Two classes of product cipher

Modern block ciphers are all product ciphers, but they are divided into two classes.

Feistel cipher

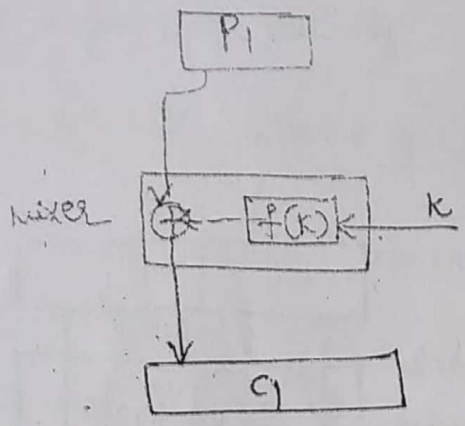
→ Feistel is a cipher that has been used for decades.

→ A Feistel cipher can have 3- types of components

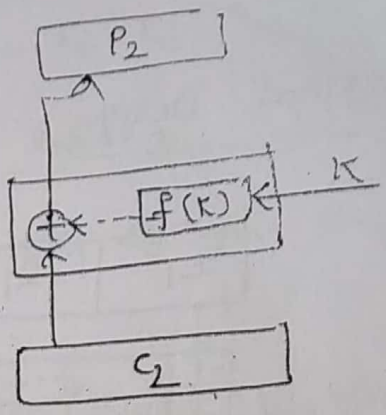
- → self-Invertible
- → Invertible
- → Non-Invertible

→ A Feistel cipher combines all Noninvertible elements in a unit and uses the same unit in the Encryption and decryption Alg.

Let us see how we can use the same Noninvertible component in the Encryption and decryption alg.



Encryption.



Decryption.

→ In the Encryption, A Non-Invertible function, f(k), accepts the key as the input. The o/p of this component is Exclusive-ored with the PT. The result becomes the CT.

→ We call the combination of the function and the Exclusive-or operation the Mixer.

Encryption: $C_1 = P_1 \oplus f(k)$.

Decryption: $P_2 = C_2 \oplus f(k) = C_1 \oplus f(k) = P_1 \oplus f(k) \oplus f(k) =$

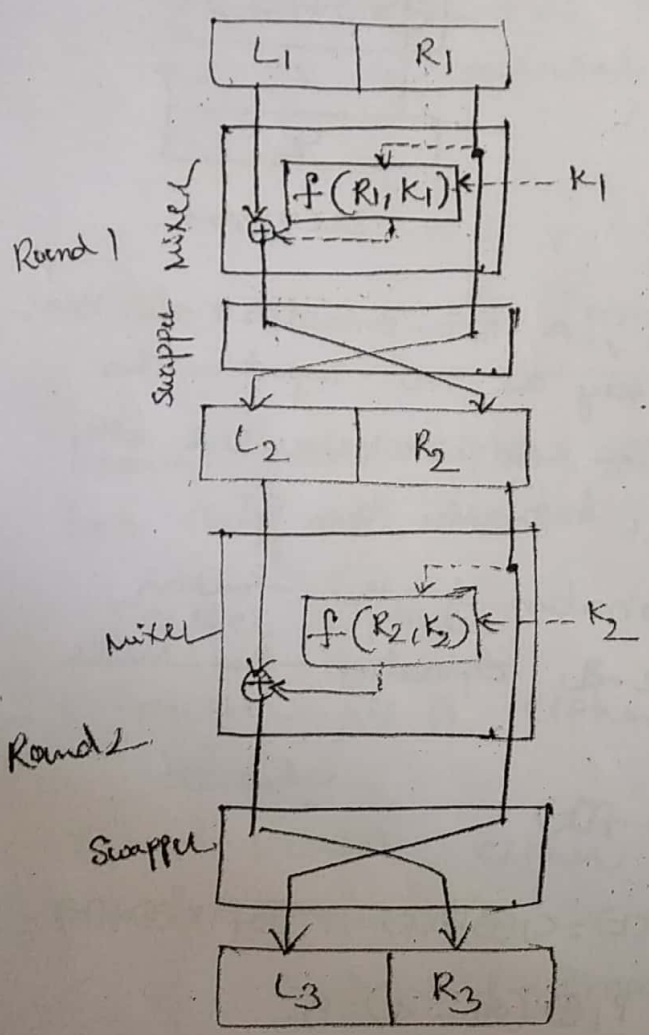
$P_1 \oplus (00\ldots.0) = P_1$.

Improvement.

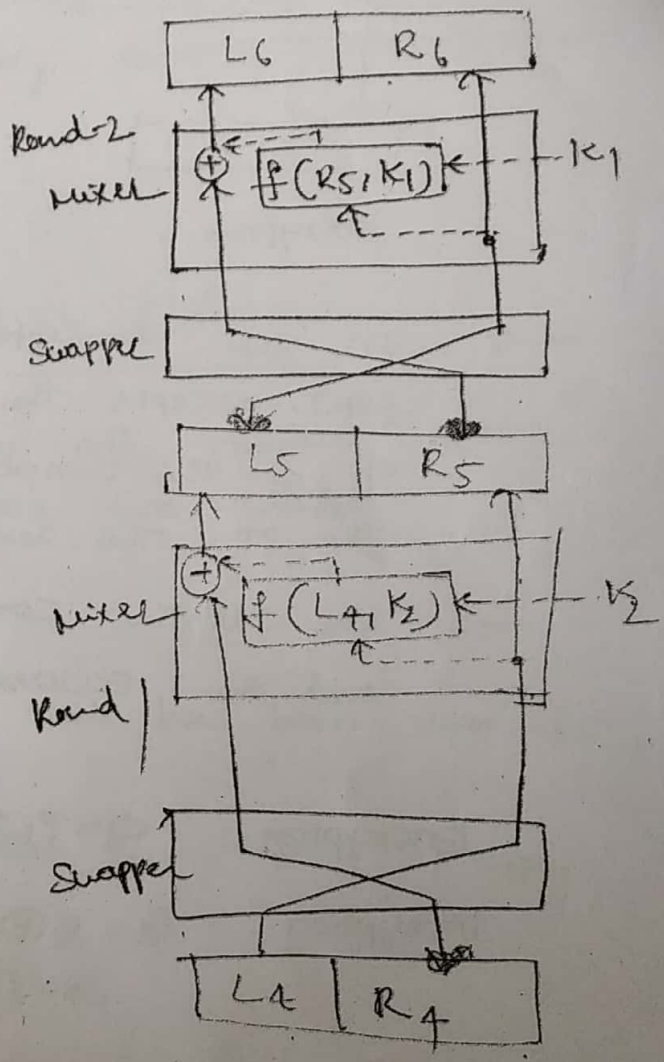
To achieve this goal, divide the PT and CT into two Equal-length blocks, left and right.

We call left block(L) and right Block(R)

Final Design



Encryption



Decryption

There are 2-round keys, K_1 and K_2. The keys are used in reverse order in the encryption and decryption.

Because the two mixers are Inverse of each other, and the swappers are Inverse of each other, it should be clear that the encryption and decryption ciphers are Inverse of each other.

Let us see if $L_6 = L_1$ and $R_6 = R_1$

Let us Assume $\Rightarrow L_4 = L_3$ and $R_4 = R_3$

we first prove the Equality ~~of~~ for the Middle text.

$$L_5 = R_4 \oplus f(L_4, K_2) = R_3 \oplus f(R_2, K_2) = L_2 \oplus f(R_2, K_2) \oplus f(R_2, K_2) = L_2$$

$$R_5 = L_4 = L_3 = R_2.$$

Then it is Easy to prove that the Equality holds for two plaintext blocks

$$L_6 = R_5 \oplus f(R_5, K_1) = R_2 \oplus f(L_2, K_1) = L_1 \oplus f(R_1, K_1) \oplus f(R_1, K_1) = L_1.$$

$$R_6 = L_5 = L_2 = R_1.$$

Attacks on Block cipher

→ Attacks on tradelional ciphers can also be used on Modern Block ciphers..

There also 2-types of attacks on Block ciphers

→ differential cryptanalysis

→ linear cryptanalysis

Differential cryptanalysis

→ This is a choosen plaintext attack; Eve can somehow access Alices computer, submitting choosen plaintext and obtaining corresponding CT.

→ The goal is to find Alice's Cipher key.

Algorithm Analysis

→ Before Eve uses the choosen-plaintext attack, she needs to analyze the Encryption alg in order to collect some information about PT-CT relations. Obviously, Eve does not know the Cipher-key. However, some ciphers have weakness in their structure that can allow Eve to find a relation b/w the PT diffrences and CT differences leaking Information about the key

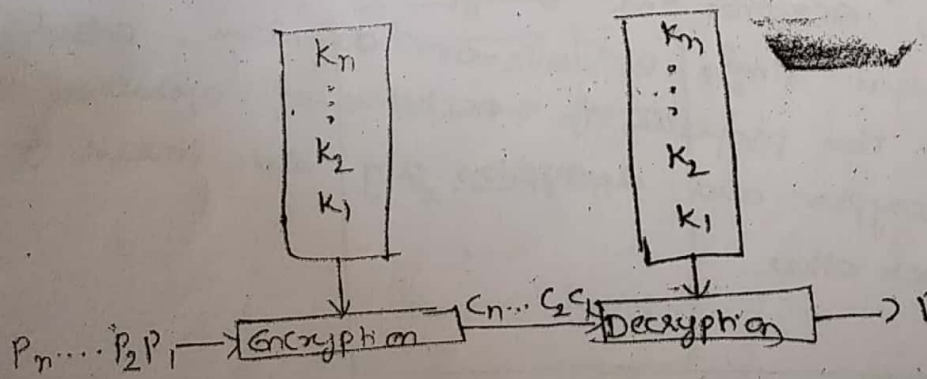
Linear Cryptanalysis

Modern Stream Ciphers

In a Modern Stream ciphers, encryption and decryption are done r bits at a time. we have a plaintext bit stream $P = P_n \ldots P_2 P_1$, a ciphertext bit stream $C = C_n \ldots C_2 C_1$, and a key bit stream $K = K_n \ldots K_2 K_1$, in which P_i, C_i, and K_i are r-bit words.

Encryption $C_i = E(K_i, P_i)$

Decryption $P_i = D(K_i, C_i)$



→ Two types of Modern Stream ciphers

1. Synchronous Stream Cipher
2. Asynchronous Stream Cipher.

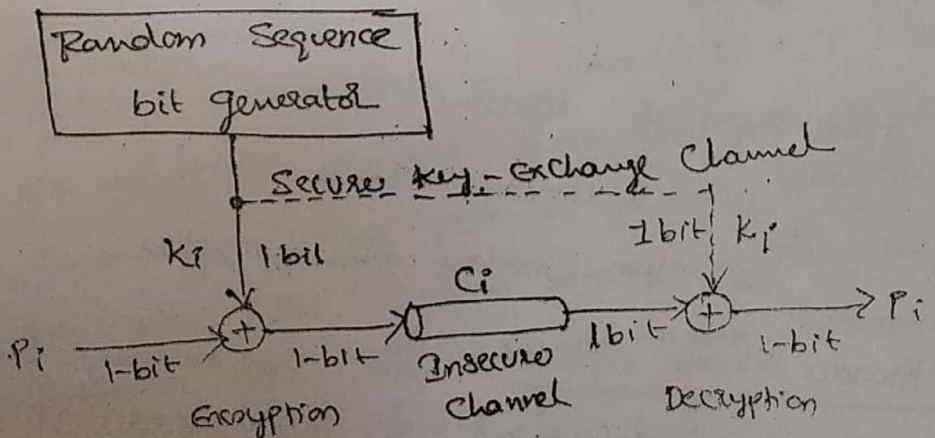
Synchronous Stream Cipher: In Synchronous the key stream is Independent of the PT (&) CT stream. The key stream is Generated and used with NO relationship b/w key bits and the PT (&) CT bits

Asynchronous Stream ciphers:- In Asynchronous, each key in the key stream depends on previous PT & CT.

One-Time pad:- The simplest and the most secure type of synchronous stream cipher is called the one-time pad, which was Invented and patented by Gilbert Vernam.

→ A one-time pad cipher uses a key stream that is randomly chosen for each encipherment.

→ The encryption and decryption algorithms each uses a single exclusive-or operation. Based on the properties of exclusive-or operation the encryption and decryption alg are inverse of each other.



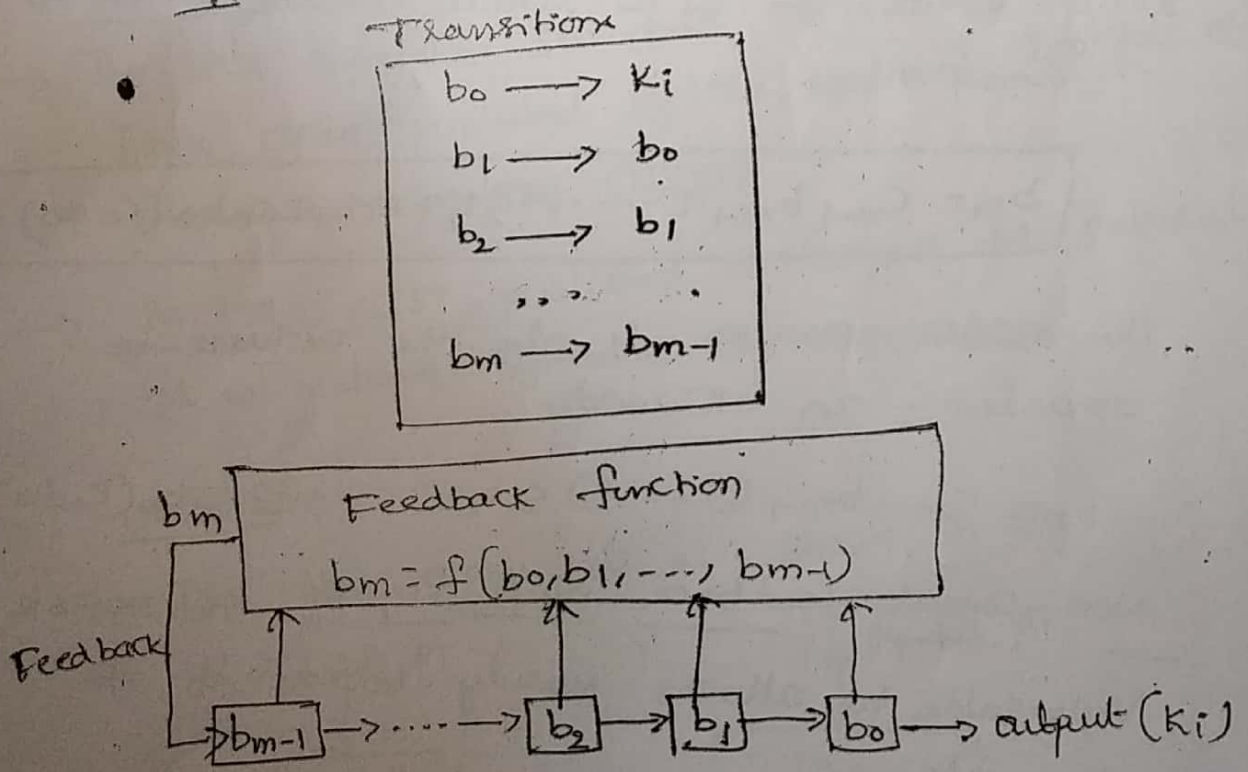
→ The one-time pad is an Ideal cipher. It is perfect. There is no way that an adversary can guess the key & the plaintext and ciphertext statistics.

→ There is No relationship b/w the plaintext and ciphertext, either.

Feedback Shift Register: one compromise to the one-time-pad is the feedback shift register (FSR).

→ An FSR can be Implemented in either S/w & H/w, but the hardware Implementation is easier to discuss.

A feedback shift register is made of a shift register and a feedback function, as shown in fig.



→ The shift register is a sequence of m cells, b_0 to b_{m-1}, where each cell holds a single bit. The cells are Initialized to an m-bit word, called the Initial value of the seed.

→ Whenever an output bit is Need, Every bit is shifted one cell to the right, which Means that Each cell give its value to the cell to its right and receives the Value of the cell to its left.

→ The rightmost cell, b_0, gives its value as output (k_i); the left most cell, b_{m-1}, receives its value from the feedback function.

Linear Feedback Shift Register.

In a LFSR, b_m is a linear function of $b_0, b_1, \ldots b_{m-1}$.

$$b_m = c_{m-1} b_{m-1} + \cdots + c_2 b_2 + c_1 b_1 + c_0 b_0 \quad (c_0 \neq 0)$$

The addition operation is also the exclusive-or operation; In other words,

$$b_m = c_{m-1} b_{m-1} \oplus \cdots \oplus c_2 b_2 \oplus c_1 b_1 \oplus c_0 b_0 \quad (c_0 \neq 0)$$

Non-linear feedback Shift Register: The NLFSR is Vulnerable to attacks mainly because of its linearity.

An NLFSR has the same structure as an LFSR Except that the b_m is the nonlinear function of $b_0, b_1, \ldots b_m$.

Ex: In a 4-bit NLFSR, the relation can be as shown below where AND Means bit-wise and operation, OR Means bit-wise or operation.

U-2 (Chapter-3)

Data Encryption Standard.

Objectives.

→ To review a short history of DES
→ To define Basic Structure of DES.
→ To describe the details of building elements of DES.
→ To describe the round keys generation process.
→ To Analyze DES.

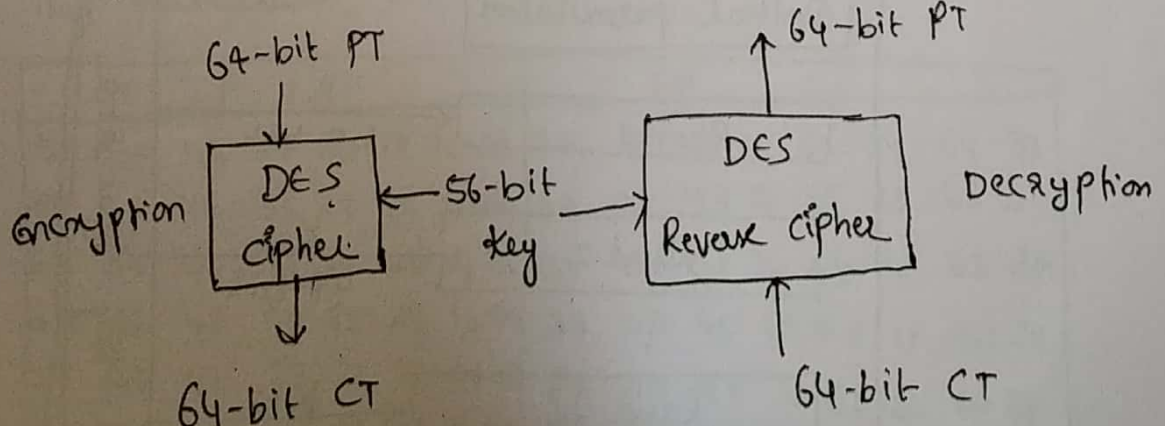
Introduction.

→ The Data Encryption Standard is a symmetric-key block cipher published by the National Institute of Standard Technology (NIST).

→ In 1973, NIST published a request for proposal for a National Symmetric-key Cryptosystem.

Overview:

DES block cipher, is as shown in fig.



→ At the encryption site, DES takes a 64-bit PT and creates a 64-bit CT; at the decryption site, DES takes a 64-bit CT and creates a 64-bit block of PT.

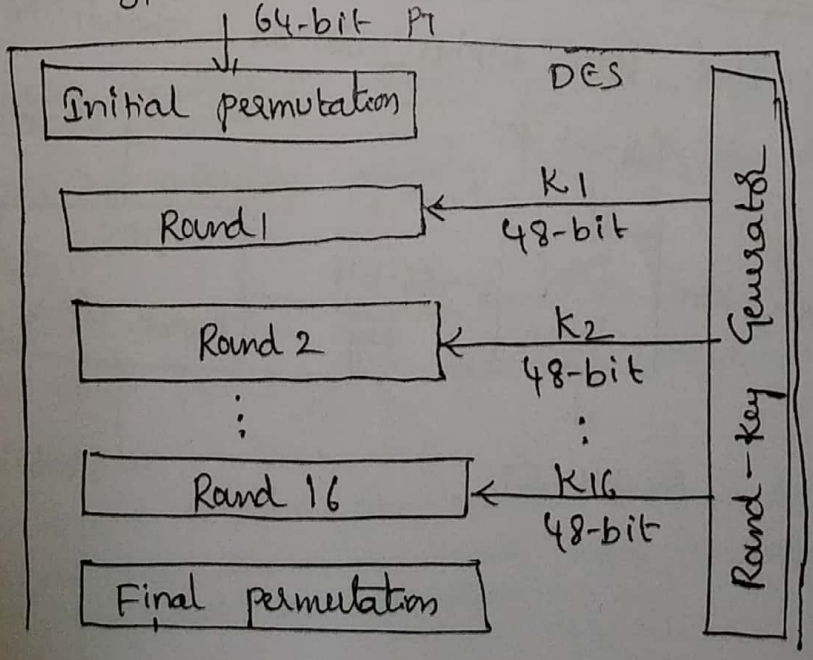
→ The same 56-bit cipher key is used for both encryption and Decryption.

DES Structure

The encryption process is made of two permutations (P-boxes), which we call Initial permutation and Final permutation, and sixteen Feisel Rounds.

→ Each round uses a different 48-bit Round Key generated from the cipher key.

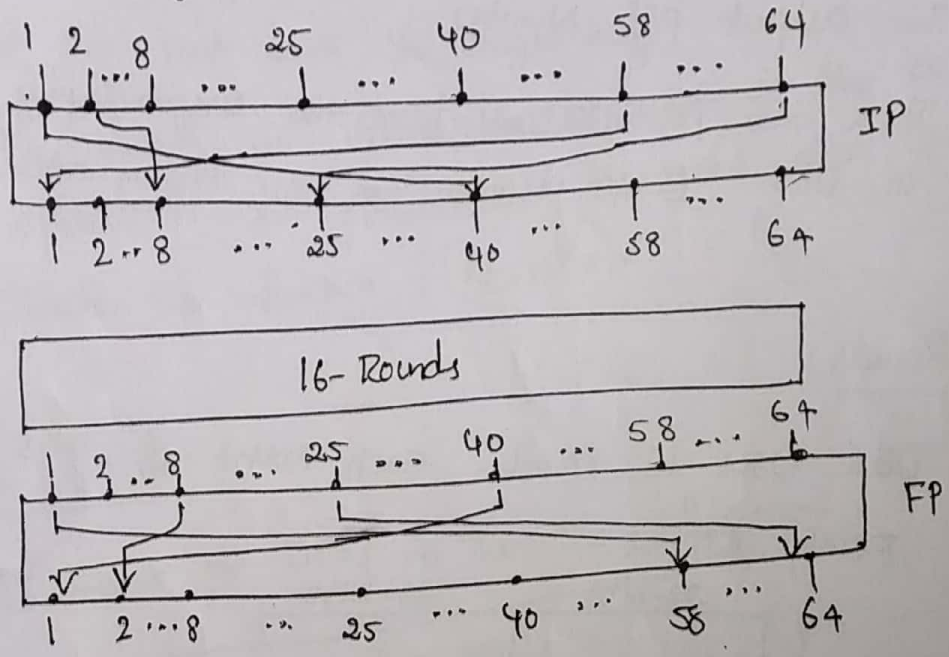
The following shows the elements of DES cipher at the encryption site.



64-bit CT

Initial and Final permutations

The following shows the Initial and final permutations.



→ Each of these permutation takes a 64-bit Input and permutes them according to a predefined rule. We have shown only a few input pins and the corresponding output pins. These permutations are keyless straight permutations that are the Inverse of each other.

→ The permutation rules for these P-boxes are shown

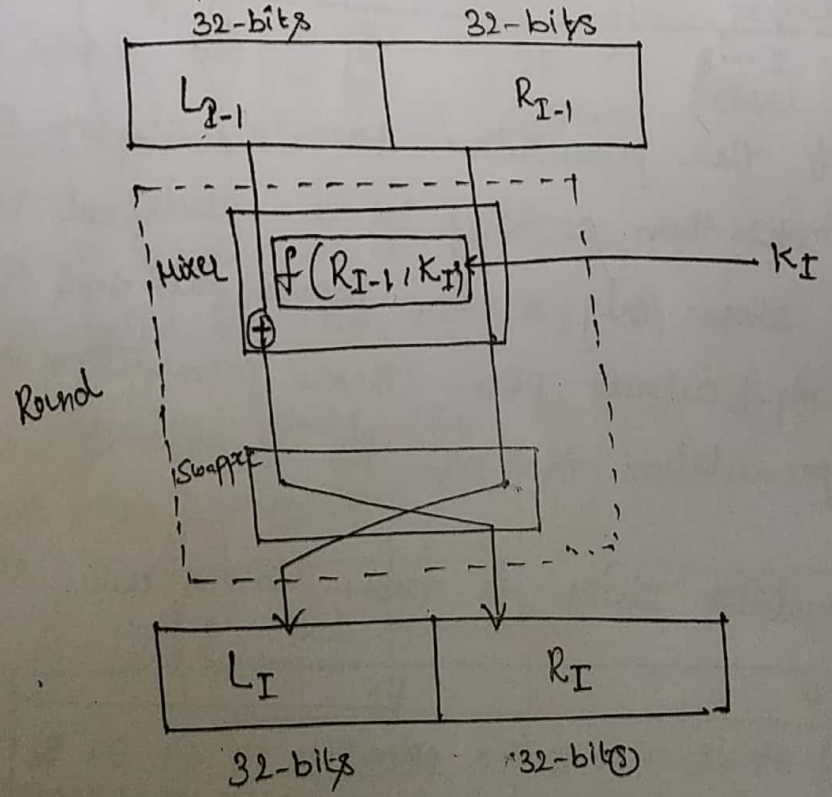
IP								FP							
58	50	42	34	26	18	10	02	40	08	48	16	56	24	64	32
60	52	44	36	28	20	12	04	39	07	47	15	55	23	63	31
62	54	46	38	30	22	14	06	38	06	46	14	54	22	62	30
64	56	48	40	32	24	16	08	37	05	45	13	53	21	61	29
57	49	41	33	25	17	09	01	36	04	44	12	52	20	60	28
59	51	43	35	27	19	11	03	35	03	43	11	51	19	59	27
61	53	45	37	29	21	13	05	34	02	42	10	50	18	58	26
63	55	47	39	31	23	15	07	33	01	41	09	49	17	57	25

→ Each side of the table can be thought of a 64-Element array., the values of each element defines the i/p pat Number, and the order of the element defines the output pat Number.

→ These two permutations have No Cryptography significance in DES. Both permutations are keyless and predetermined

Rounds

DES Uses 16 Rounds. Each round of DES is a Feistel Cipher.



→ The round takes L_{I-1} and R_{I-1} from previous round and creates L_I and R_I, which go to the Next round..

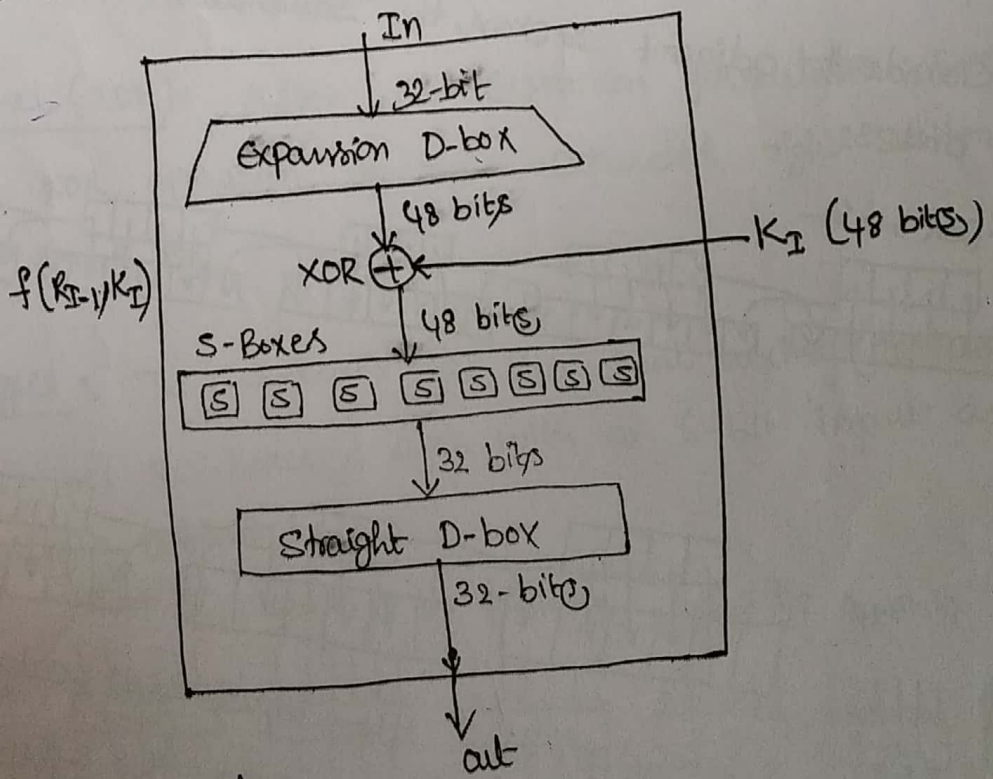
→ Each round has two cipher elements. (Mixer & swapper).

→ Each of these elements is invertible. The swapper is obviously Invertible. It swaps the left half of the text with the right half.

→ The Mixer is Invertible because of the XOR operation. All Invertible elements are collected Inside the function $f(R_{I-1}, K_I)$.

DES Function :

The Heart of DES is the DES function.



→ The DES function applies a 48-bit key to the rightmost 32-bits (R_{I-1}) to produce a 32-bit output.

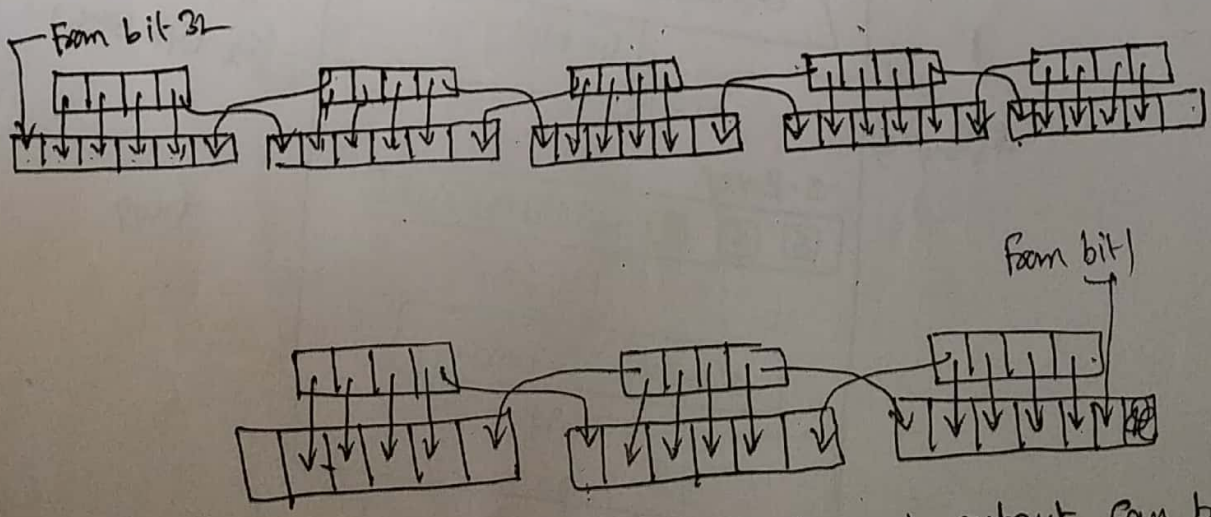
→ This function is made up of four sections: an Expansion D-box, a whitener (that adds key), a group of S-boxes and a Straight D-box.

Expansion D-box :

Since R_{I-1} is a 32-bit input and K_I is a 48-bit key, we first need to expand R_{I-1} to 48-bits. R_{I-1} is divided into 8 4-bit sections.

→ Each 4-bit section is then Expanded to 6-bits. This Expansion permutation follows a predetermined rule.

→ For each section, input bits 1, 2, 3 and 4 are copied to output bits 2, 3, 4 and 5 resp. output bit 1 comes from bit 4 of the previous section; output bit 6 comes from bit 1 of the Next section. If section 1 and 8 can be considered adjacent sections, the same rule applies to bit 1 and 32.



→ The relationship b/w the input and output can be defined mathematically, DES uses the following table to define this D-box.

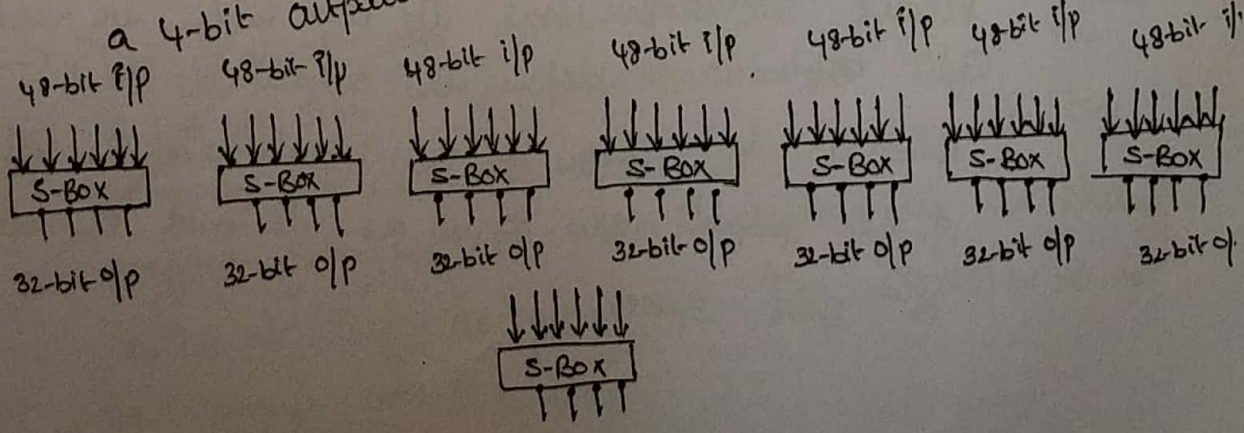
→ Note that the No. of output port is 48, but the value range is only 1 to 32. Some of the inputs go to More than one output.

Expansion D-box table

32	01	02	03	04	05
04	05	06	07	08	09
08	09	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	31	31	32	01

Whitener (XOR): After the Expansion permutation, DES uses the XOR operation on the expanded right section and the round key.

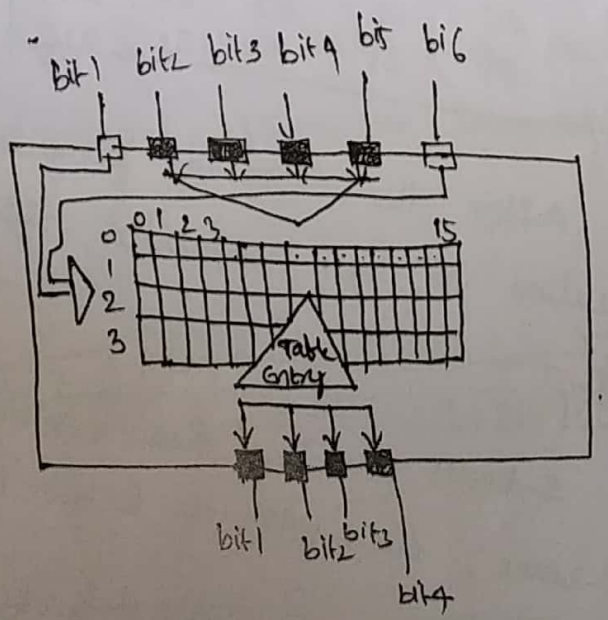
S-Boxes: The S-boxes do the real Mixing (Confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output.



→ The 48-bit data from the second operation is divided into eight 6-bit chunks, and each chunk is fed into a box. The result of each box is a 4-bit chunk; when these are combined the result is a 32-bit text.

→ The substitution in each box follows a pre-determined rule based on a 4-row by 16-column table.

→ The combination of bits 1 and 6 of the input defines one of four rows; the combination of bits 2 through 5 defines one of the sixteen columns.



→ Because each S-box has its own table, we need eight tables. to defines the output of these S-boxes. The values of the inputs and the values of the outputs are given as decimal numbers to save spaces.

Final permutation.

The last operation of DES operation function is a permutation with a 32-bit input and a 32-bit output.

Cipher and Reverse Cipher:

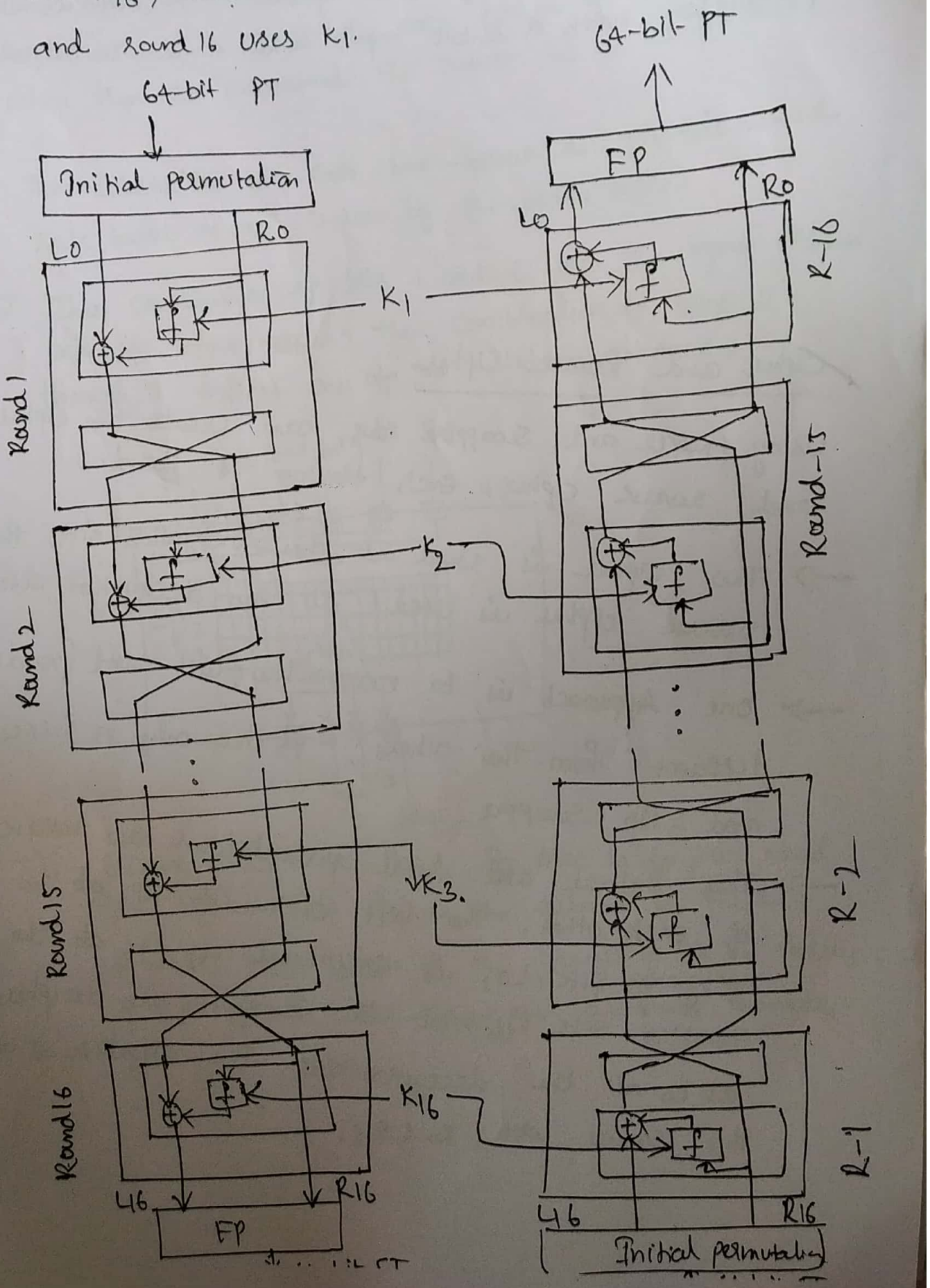
Using Mixer and Swapper we can create the cipher and reverse cipher, each having 16 Rounds.

→ The cipher is used at the encryption site; the reverse cipher is used at the decryption site.

→ One Approach is to make the last round (round 16) different from the others; it has only a Mixer and No Swapper.

→ The Initial and final permutations are inverse of each other. The left section of PT at the encryption site, L_0, is enciphered as L_{16} at the encryption site; L_{16}, at the decryption is deciphered as L_0 at the decryption site. The situation is the same with R_0 & R_{16}.

→ At the Encryption site, round 1 uses K_1 and round 16 uses K_{16}; at the Decryption site, round 1 uses K_{16} and round 16 uses K_1.

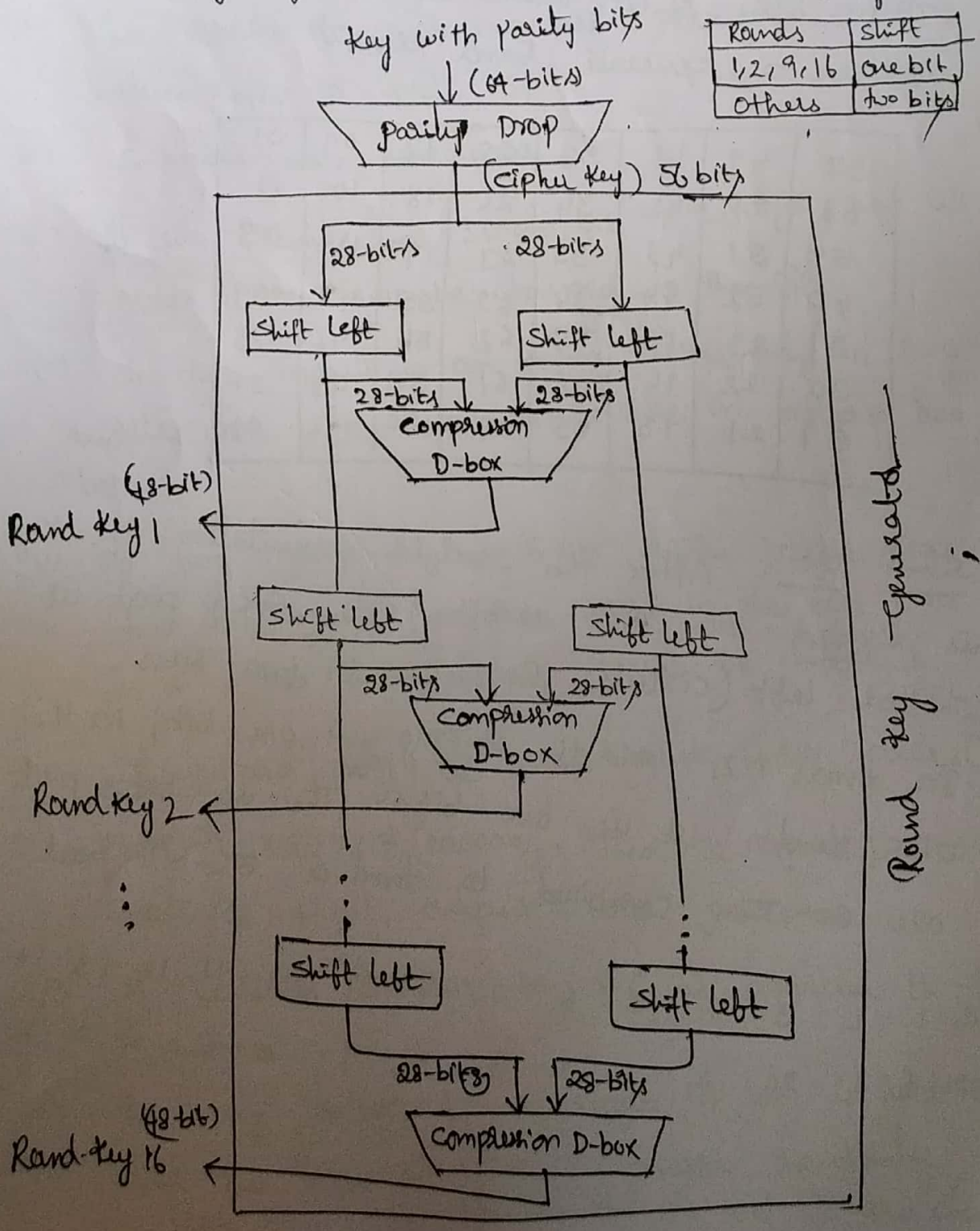


Key Generation:

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key. However, the cipher key is normally given as a 64-bit key in which 8 extra bits are the parity units, which are dropped before the actual key-generation process.

Shifting

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



Parity Drop :- The preprocess before key expansion is also Compression transposition step that we call parity bit drop. It drops the parity bits (8, 16, 24, 32, ... 64) from the 64-bit key and permutes the rest of the bits according to the below table. The remaining 56-bit value is the actual cipher key which is used to generate round keys.

57	49	41	33	25	17	09	01
58	50	42	34	26	18	10	02
59	51	43	35	27	19	11	03
60	52	44	36	63	55	47	39
31	23	15	07	62	54	46	38
30	22	14	06	61	53	45	37
29	21	13	05	28	20	12	04

Shift left :- After the straight permutation, the key is divided into two 28-bit parts. Each part is shifted left (circular shift) one or two bits. In rounds 1, 2, 9 and 16, shifting is one bit; in the other rounds, it is two bits. The two parts are then combined to form a 56-bit part.

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
BitShift	1	1	2	2	2	2	2	2	1	2	2	2	2	2	1

Design Criteria

→ The Design of DES was revealed by IBM in 1994;

S-Boxes

According to the revelation and some research, we can mention several properties of S-boxes.

(•) The Entries of each row are permutations of Values b/w 0 and 15.

(•) S-boxes are Nonlinear.

(•) If we change a single bit in the i/p, two or More bits will be changed in the o/p.

(•) If two inputs to an S-box differ only in two middle bits, the o/p must differ in atleast two bits.

(•) If two inputs to an S-box differ in the first two bits and are the same in the last two bits, the two outputs must be different.

(•) There are only 32 6-bit input-word pairs (x_i and x_j) in which $x_i \oplus x_j \neq (000000)_2$. These 32 input pairs create 32 4-bit output-word pairs.

(•) A Criterion similar to #6 is applied to three S-boxes.

(•) In any S-box, if a single i/p bit is held constant and the other bits are changed randomly, the Number of 0's and 1's are minimized.

Compression D-box

The Compression D-box changes the 58 bits to 48 bits which are used as a key for a round.

14	17	11	24	01	05	03	28
15	06	21	10	23	19	12	04
26	08	16	07	27	20	13	02
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

DES Analysis

Two desired properties of a block cipher are the avalanche effect and the completeness.

Avalanche Effect: Avalanche effect means a small change in the PT (or key) should create a significant change in the CT.

Completeness Effect: Completeness effect means that each bit of the CT needs to depends on many bits on the PT. The Diffusion and confusion produced by D-boxes and S-boxes in DES, show a very strong completeness effect.

D-Boxes.

:) Between two rows of S-boxes, there are one straight D-box and one Expansion D-box (32 to 48 bit). these two D-boxes together provide diffusion of bits.

The following criteria were Implemented in the design of D-boxes to achieve this goal:

1) Each S-box input comes from the output of a different S-box
2) No i/p to a given S-box comes from the o/p from the same box.
3) The four outputs from each S-box go to six different S-boxes.
4) No two output bits from an S-box go to the same S-box.
5) If we Number the eight S-boxes, $S_1, S_2, \ldots, S_8$.
   a) An o/p of S_{j-2} goes to one of the first two bits of S_j
   b) An o/p bit from S_{j-1} goes to one of the last two bits of S_j
   c) An o/p of S_{j+1} goes to one of the last two middle bits of S_j
6) For each S-box, the two output bits go to the first (&) last two bits of an S-box in the next round.

The other two output bits go to the middle bits of an S-box in the next round.

7) If an output bit from S_j goes to one of the middle bits in S_k, then an output bit from S_k can't go to the middle bit of S_j.

Number of Rounds: DES uses 16 Rounds of Feistel cipher. It has been proved that after eight rounds, each CT is a function of every plaintext bit and every key bit; the CT is thoroughly a random function of PT and CT.

DES weakness.

We have some weakness in DES.

→ weakness in cipher Design
→ weakness in cipher key.

weakness in cipher Design:

The following are some weakness that have been found in the Design of the cipher.

S-boxes At least three weaknesses are Mentioned in the literature for S-boxes.

1. In S-box 4, the last three output bits can be derived in the same way as the first output bit by complementing some of the input bits.
2. Two specifically chosen inputs to an S-box array can create the same output.
3. It is possible to obtain the same output in a single round by changing bits in only three neighboring S-boxes.

D-boxes : one weakness were found in the design of D-boxes

→ It is NOT clear why the designers of DES uses the IP and FP; they have No security benefits

→ In the expansion permutation, the first and fourth bits of every 4-bit series are repeated.

Weakness in the Cipher Key :

Several weaknesses have been found in the cipher key.

→ key size, weak keys, semi-weak keys, possible weak keys., key complement, key clustering

Key size:

→ The most serious weakness of DES is in its key size (56 bits). To do a brute-force attack on a given CT block, the adversary needs to check 2^{56} keys..

→ The DES with a cipher key of 56 bits is Not safe enough to be used comfortably.

Weak keys.

Four out of 2^{56} possible keys are called weak keys. A weak key is the one that, after parity drop operation, consists either of all 0s, all 1s, & half 0s and half 1s.

~~Semi-weak keys~~.

keys before parities drop (64 bits)				Actual key (56 bits)	
0101	0101	0101	0101	0000000	0000000
1F1F	1F1F	0E0E	0E0E	0000000	FFFFFFF
E0E0	E0E0	F1F1	F1F1	FFFFFFF	0000000
FEFE	FEFE	FEFE	FEFE	FFFFFFF	FFFFFFF

Semi-weak Keys:

There are six key pairs that are called semi-weak keys. These six pairs are shown in below.

First key in the pair	Second key in the pair
01FE 01FE 01FE 01FE	FE01 FE01 FE01 FE01
1FE0 1FE0 0EF1 0EF1	E01F E01F F10E F10E
01E0 01E1 01F1 01F1	E001 E001 F101 F101
1FFE 1FFE 0EFE 0EFE	FE1F FE1F FE0E FE0E
011F 011F 010E 010E	1F01 1F01 0E01 0E01
E0FE E0FE F1FE F1FE	FEE0 FEE0 FEF1 FEF1

→ A semi-weak key creates only two different round keys and each of them is repeated eight times.

Possible weak keys: There are totally 48 keys that are called possible weak keys. A possible weak key is a key that creates only four distinct round keys; in other words, the 16 Rand keys are divided into 4 groups and each group is made of four Equal Rand keys..

Key complement: In the key domain (2^{56}), definitely half of the keys are complement of the other half. A key complement can be made by inverting each bit in the key.

Key clustering: Key clustering refers to the situation in which two or more different keys can create the same CT from the same PT. Obviously, each pair of the semi-weak keys is a key cluster.

Security in DES

DES, as the first important block cipher, has gone through much scrutiny. Among the attempted attacks, three are of interest: brute-force, differential cryptanalysis, and linear cryptanalysis.

Brute-Force attack:

→ A brute force attack against a cipher consists of breaking a cipher by trying all possible keys. On statistically, if the keys were originally chosen randomly, the PT will become available after about half of the possible keys are tried.

Multiple DES - conventional Encryption Algorithms

2-DES and Meet in the Middle attack:

Consider a msg m, which is to be encrypted. The corresponding block cipher for one application of the DES applications is represented by E_K, where K is the corresponding DES key.

→ The output of 2-DES is $\boxed{C = E_{K_2}(E_{K_1}(m))}$.

→ To decrypt similarly, $\boxed{m = D_{K_1}(D_{K_2}(C))}$.

→ The 2DES should offer additional security, Equivalent to both K_1 and K_2. the cipher 2-DES obtained by the repeated Application of DES is Called, $\boxed{2\text{-DES} = DES \times DES}$.

→ This is called as product cipher obtained by the Composition of two ciphers.

Meet-in-the-Middle (MIM) Attack and 3-DES.

Consider the cipher 2-DES as defined. The PT and the CT of the cipher is $P = \{0,1\}^m$.

The key space of DES is $K = \{0,1\}^n$, the key size of the product cipher is expected to be $K_1 \times K_2$, where the key is represented as the ordered pair (K_1, K_2), where K_1 belongs to K_1 and K_2 belongs to K_2..

→ The attacker obtain 1 pairs of PT and CT: $(P_1, C_1), \ldots, (P_1, C_1)$. The key is say (K_1, K_2) but unknown to the attacker..

→ It is easy to prove that for all $1 \le i \le 1$, $DES_{K_1}(P_i) = DES^{-1}_{K_2}(C)$. There are in total 2^{2n} keys..

→ The probability of a key satisfying this Equation for a particular value of i is 2^{-m}, as that is the block size of the cipher.

→ The probability of the key satisfying the above Equation for all the 1 values of i, is 2^{-m1}

Thus the reduced key space which satisfies the above test is expected to be $2^{2n} \cdot 2^{-m1} = 2^{2n-m1}$.

Examples of Block ciphers Influenced by DES.

CAST Block cipher.

→ The CAST Block cipher is an Improvement of the DES block cipher;

→ The CAST alg has 64 bit block size and has a key of size 64 bits.

→ CAST is Based on the Feistel structure to Implement the substitution and permutation Network.

S-Boxes of CAST:

→ CAST uses S-Boxes of dimension $m \times n$ ($m < n$), The typical dimension of the S-Boxes of CAST is 8×32.

→ The principle behind the construction is as follows.

Choose n distinct binary bent functions of length 2^m, such that the linear combinations of these functions sum to Highly Non-linear, Boolean functions.

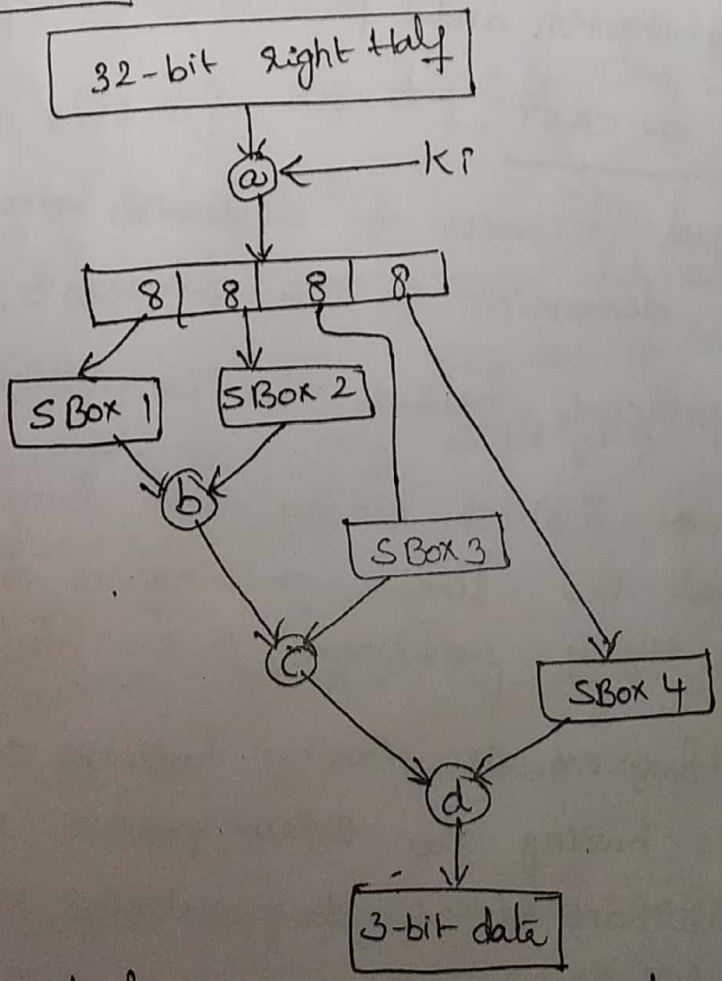
→ Bent functions are Boolean functions with even input variables having the highest possible non-linearity. The resultant function also satisfy Strict Avalanche Criteria (SAC).

- Encryption function: The plaintext block is divided into a left half and a right half. The Alg has 8 Rounds. Each round is essentially a Feistel Structure.

→ In each round the right half is combined with the round key using a function f and then XOR with the left half.

→ The new left half after the round is the same as the right half before the round. After 8 iterations of the rounds, the left & right half are concatenated to form the CT.

The Round function.



→ The Round function in CAST can be realized as follow. The 32 bit input can be combined with 32 bits of Round key through a function, denoted by "a".

The 32-bit data half is combined using operation 'a' and the 32-bit result is split into 8-bit pieces. Each piece is input into a 8x32 S-Box.

→ The output of S-Box 1 and 2 are combined using the operation "b", the 32 bit output is combined with the output of S-box 3, the output is combined in turn with the output of S-Box 4.

→ The combining functions are denoted in the figure by "c" and "d".

Key scheduling of CAST.

→ A key scheduling in CAST has 3 main components.

1. A key transformation step which converts the primary key (i/p key) to an Intermediate key.
2. A relatively simple bit-selection Alg mapping the primary key and the Intermediate key to a form, referred as partial key bits.
3. A set of key-schedule S-Boxes which are used to create subkeys from the partial key bits.

Let, the i/p key be denoted by

—> $KEY = K_1 K_2 K_3 K_4 K_5 K_6 K_7 K_8$, where K_i is the ith byte of the primary key.

—> The key transformation step generated the Intermediate key. $KEY' = K'_1 K'_2 K'_3 K'_4 K'_5 K'_6 K'_7 K'_8$ as follows.

$$K'_1 K'_2 K'_3 K'_4 = K_1 K_2 K_3 K_4 \oplus S_1[K_5] \oplus S_2[K_7]$$

$$K'_5 K'_6 K'_7 K'_8 = K_5 K_6 K_7 K_8 \oplus S_1[K'_2] \oplus S_2[K'_4]$$

Here S1 and S_2 are key-schedule S-Boxes of dimension 8x32.

Subsequently, there is a bit-selection step which operates as shown below.

$$K'_1 = K_1 K_2$$
$$K'_2 = K_3 K_4$$
$$K'_3 = K_5 K_6$$
$$K'_4 = K_7 K_8$$
$$K'_5 = K'_4 K'_3$$
$$K'_6 = K'_2 K'_1$$
$$K'_7 = K'_8 K'_7$$
$$K'_8 = K'_6 K'_5$$

The partial bits are used to obtain the subkeys, K_i. The subkey are 32-bits, and are obtain as follows.

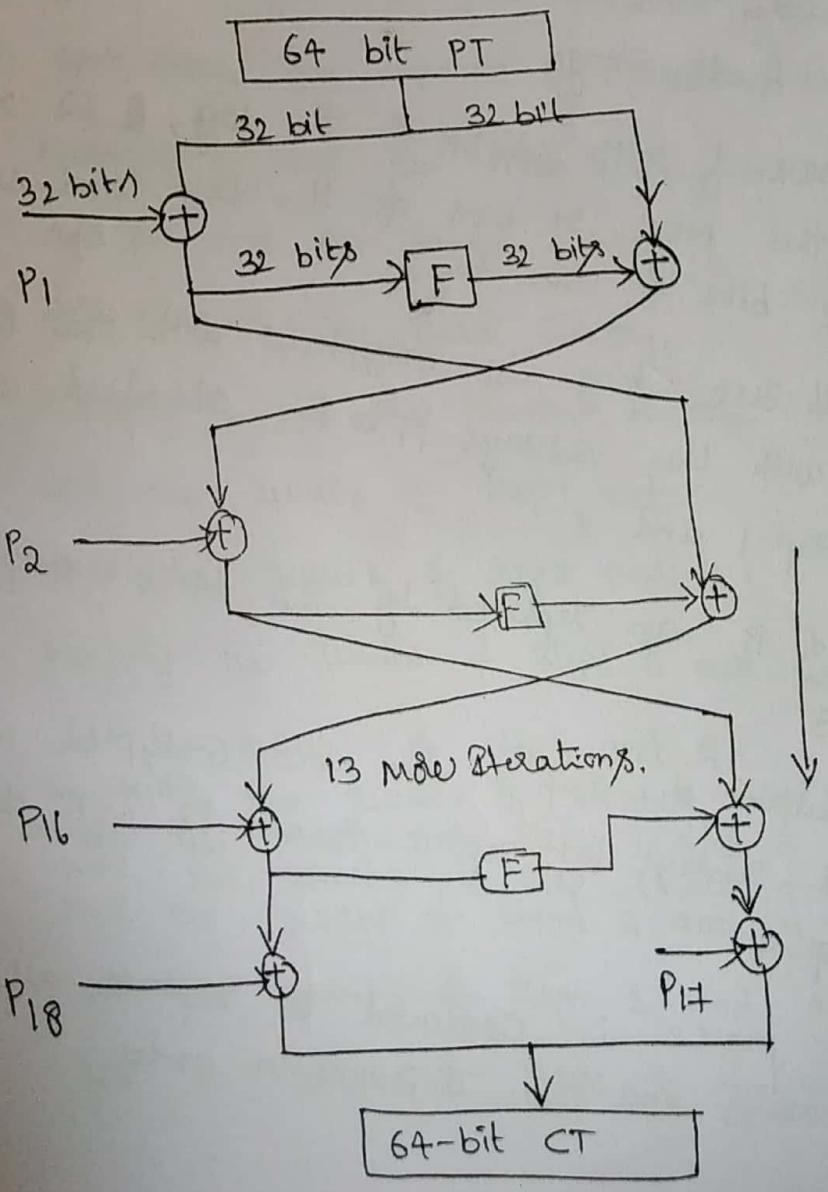
$$K_i = S_1(K'_{i,1}) \oplus S_2(K'_{i,2})$$

Blowfish: Blowfish is a 64-bit block cipher. Blowfish was designed for fast ciphering on 32-bit microprocessors.

Round structure:

This alg is based on the Feistel structure and has two important parts: → The Round structure
→ Key Expansion function.

→ There are 16 rounds, and each round are made of simple transformations which are iterated. Each round consist of a key-dependent permutation, and a key and data-dependent substitution.



→ All the operations are additions and XORs on 32-bit words, and lookups in 4 32-bit S-Boxes.

→ Blow fish has a P-array, $P_0, P_1, \ldots, P_{17}$ Each of which are 32 bit subkeys. There are 4 S-Boxes, Each of which maps an 8-bit Input to 32-bits.

[redacted].

Key Scheduling Alg:

The subkeys are computed using the following method:

1) The P-array and then the four S-Boxes are Initialized with a fixed string.

2) P_1 is XOR-ed with 32 bit of the key, P_2 is XOR-ed with the Next 32 bits of the key, and so on for all the bits of the key.

3) An all-zero string is Encrypted with the Blowfish alg, with the subkeys P_1 to P_{18} obtained so far in Step 1 and 2.

4) P_1 and P_2 are replaced by the 64-bit output of Step 3.

5) the output of Step 3 is Now Encrypted with the updated subkeys to replace P_3 and P_4 with the CT of Step 4.

6) This process is Continued to replace all the P-arrays and the S-Boxes in order.

IDEA: (International Data Encryption Alg)

IDEA is another block cipher. It operates on 64-bit data blocks and the key is 128 bit long.

→ The Design principle behind IDEA is the "mixing of arithmetical operations from different algebraic groups".

Round Transformation of IDEA:

→ The 64-bit data is divided into four 16 bit blocks: $X1, X2, X3, X4$. These four blocks are processed through Eight rounds and transformed by the above arithmetical operations among each other and with Six 16 bit subkeys.

→ In each round the sequence of operations is as follows.

1) (·) multiply X1 and the first subkey.
2) (·) Add X2 and the second subkey.
3) (·) Add X3 and the third subkey.
4) (·) multiply X_4 and the fourth subkey.
5) (·) XOR the results of step 1 and 3.
6) (·) XOR the results of step 2 and 4.
7) (·) multiply the results of steps 5 and with fifth subkey.
8) (·) Add the results of step 6 and 7.
9) (·) multiply the results of steps 8 with the sixth subkey.
10) (·) Add the results of steps 7 and 9.
11) (·) XOR the results of step 1 and 9.
12) (·) XOR the results of steps 3 and 9.
13) (·) XOR the results of steps 2 and 10
14) (·) XOR the results of steps 4 and 10.

→ The outputs of step 11, 12, 13 and 14 are stored in four words of 16 bits each, Namely Y_1, Y_2, Y_3 and Y_4.

→ The blocks Y_2 and Y_3 are swapped, and the resultant four blocks are the output of a round of IDEA.

Instead the last round has the following additional Transformation.

1. Multiply Y_1 and the first sub key.
2. Add Y_2 and the second subkey.
3. Add Y_3 and the third sub key.
4. Multiply Y_4 and the fourth sub key.

Finally, the CT is the concatenation of the blocks Y_1, Y_2, Y_3 and Y_4.

Key Scheduling of IDEA:

→ IDEA has a very simple key scheduling. It takes the 128 bit key and divides it into Eight 16 bit blocks

→ The first six blocks are used for the first round, while the remaining two are to be used for the second round.. Then the entire 128 bit key is given a rotation for 25 steps to the left and again divided int Eight blocks.. the first four blocks are used as the remaining subkeys for the second round, while the last four blocks are to be used for the third round..

→ The key is then again given a left shift by 25 bits, and the other subkeys are obtained. The process is continued till the End of the Alg.

U-2 (Chapter-4)

Advanced Encryption Standard (AES) ①

→ The AES is a symmetric-key block cipher published by the National Institute of standard and Technology in December 2001.

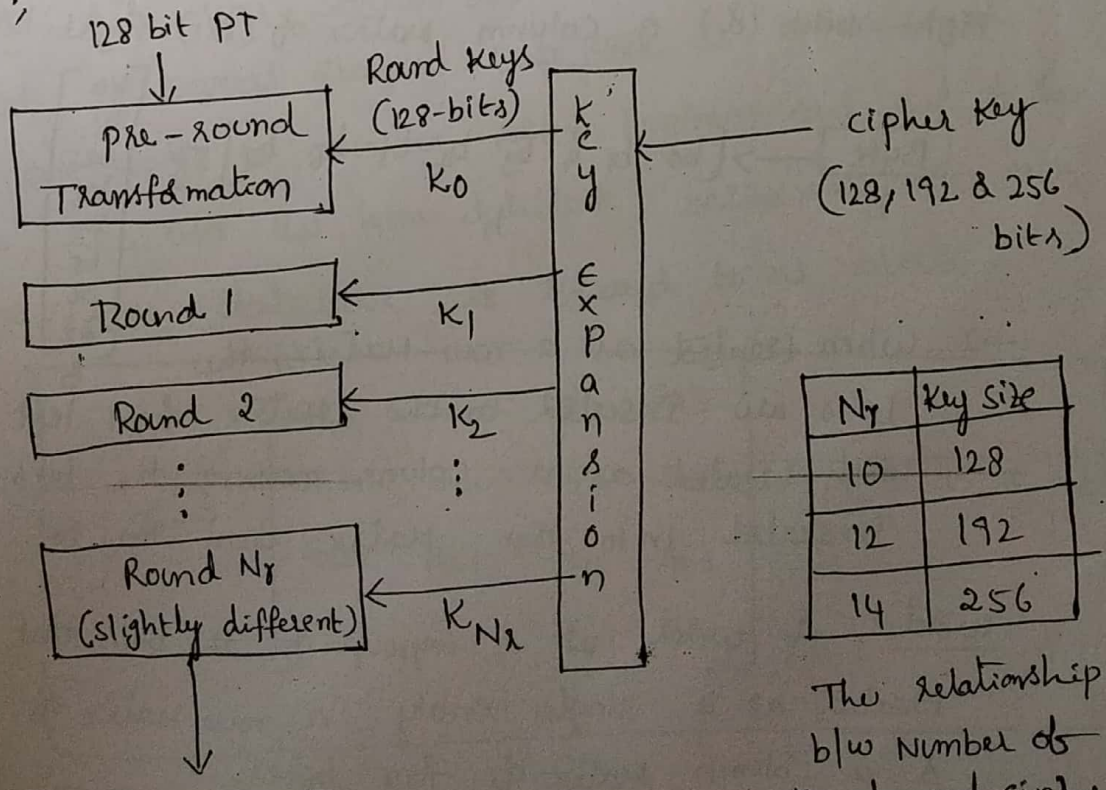
→ In 1997, NIST started looking for replacement for DES, which would be called the AES.

AES Rounds.

AES is a Non-Feistel cipher that encrypted and decrypts a data block of 128 bits. It uses 10, 12 & 14 rounds.

→ The key size, which can be 128, 192 & 256 bits, depends on the No. of rounds.

The following shows the general design for the Encryption Alg;



Nr	Key size
10	128
12	192
14	256

The relationship b/w Number of Rounds and cipher key size.

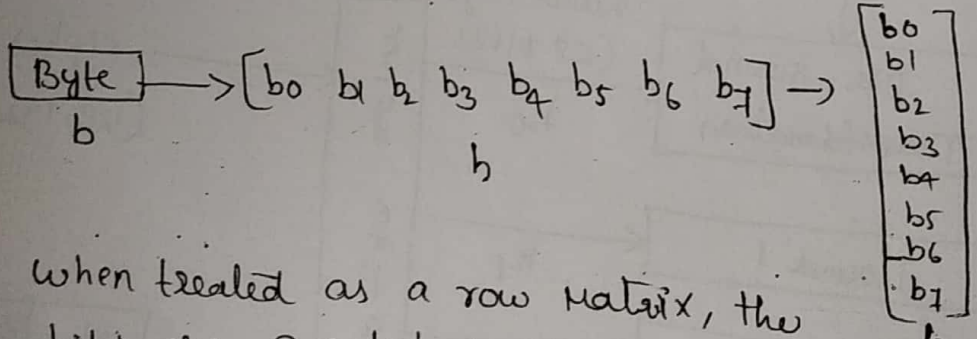
→ N_r, defines the number of rounds. The fig also shows the relationship b/w the No. of rounds and the key size, which means that we can have three different AES versions; they are referred as AES-128, AES-192 and AES-256.

Data Unit(s):

AES uses five units of measurements to refer to data: bit(s), bytes, words, blocks and state.

Bit: In AES, a bit is a binary digit with a value of 0 & 1. We use a lowercase letter to refer a bit.

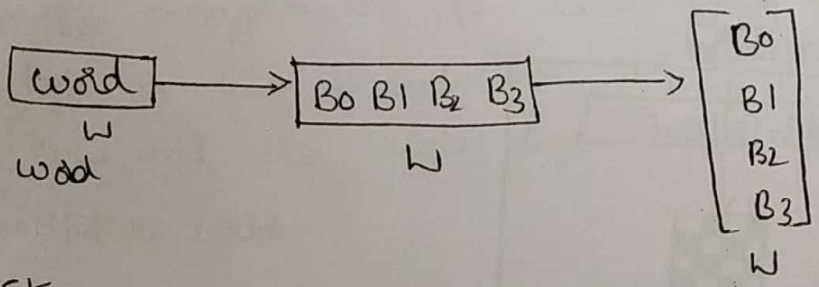
Byte: A byte is a group of eight bits that can be treated as a single entity, a row Matrix (1×8) of eight bits (&) a column Matrix of (8×1) eight bits.



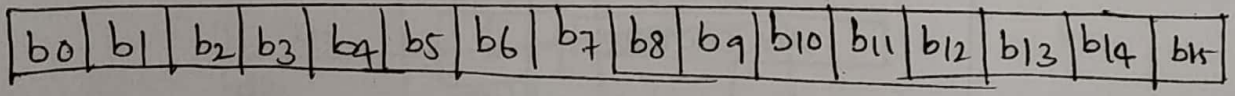
→ When treated as a row Matrix, the bits are inserted to the Matrix from left to right; when treated as a column matrix, the bits are inserted into the Matrix from top to bottom.

Word: A word is a group of 32 bits that can be treated as a single entity, a row Matrix of four bytes, & a column Matrix of four bytes.

→ when it is treated as a ~~single~~ row Matrix, the bytes are Inserted into ~~four bytes~~ Matrix from left to right; when it is Considered as Column Matrix, the bytes are Inserted into the Matrix from top to bottom.

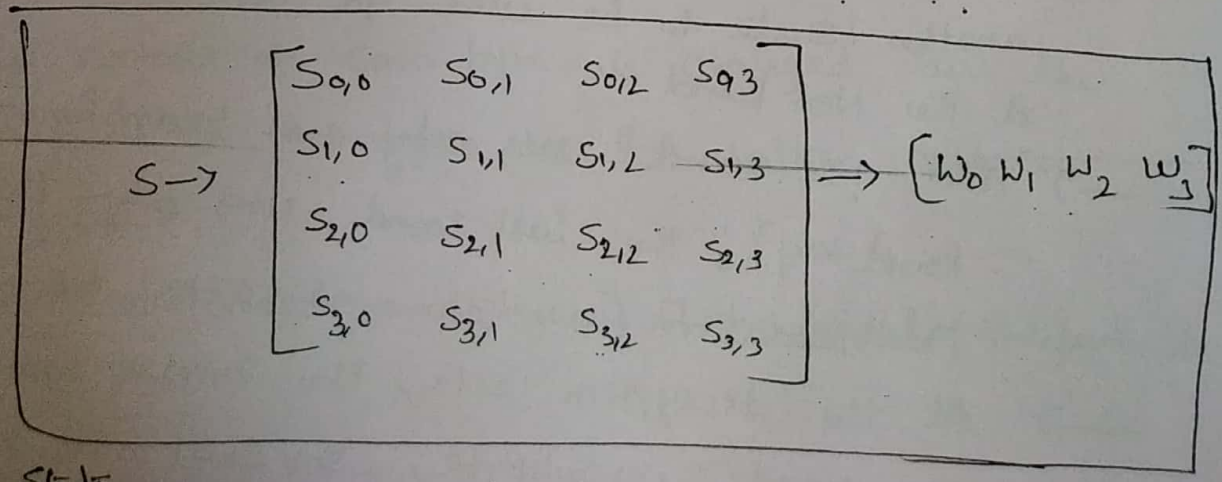


Block: AES Encrypt and decrypts data blocks. A block in AES is a group of 128-bits. However, a block can be represented as a row Matrix of 16 bytes.



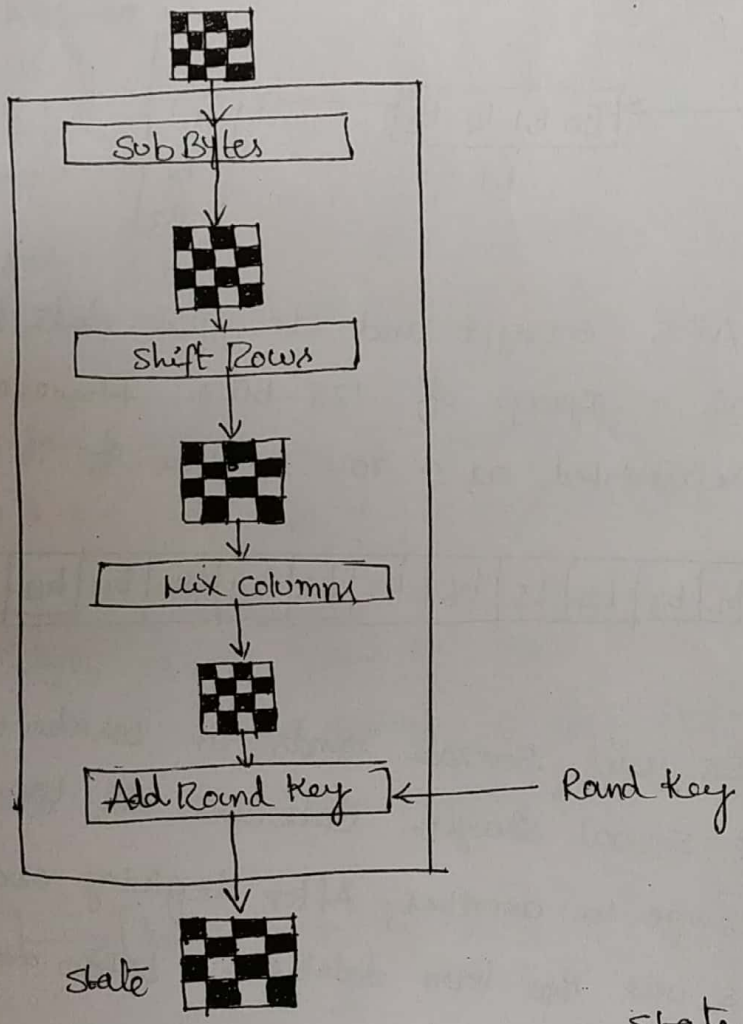
Block.

State: AES uses several rounds in which each round is made of several stages. Data Block is transformed from one stage to another. After begining and end of the cipher, AES uses the term datablock; before and after each stage, the data block is refered to as state.



Structure of each round.

The following diagram shows the structure of each round at the Encryption site.



→ Each transformation takes a state and creates another state to be used for the next transformation & the next round.

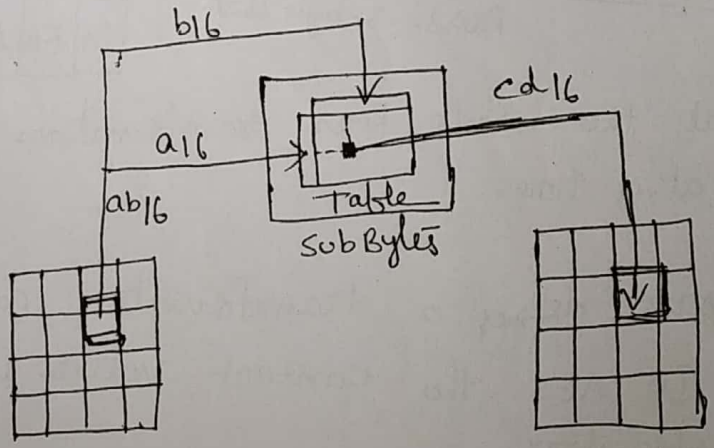
→ The pre-round uses only one transformation (Add Round key); the last round uses only three transformations (Mix Columns transformation is missing)

→ At the decryption site, the inverse transformations are used: InvSubByte, InvShiftrows, InvMixColumns and Add Round key.

Transformations: To provide security, AES uses four types of transformations:
Substitution, permutations, Mixing and key-adding.

SubBytes: The first transformation, SubBytes, is used at the encryption site. To substitute a byte, we Interpret the byte as two Hexadecimal digits. The left digit defines the row and the right digit defines the column of the Substitution table.

→ The two Hexadecimal digits at the junction of the row and the column are the New byte.

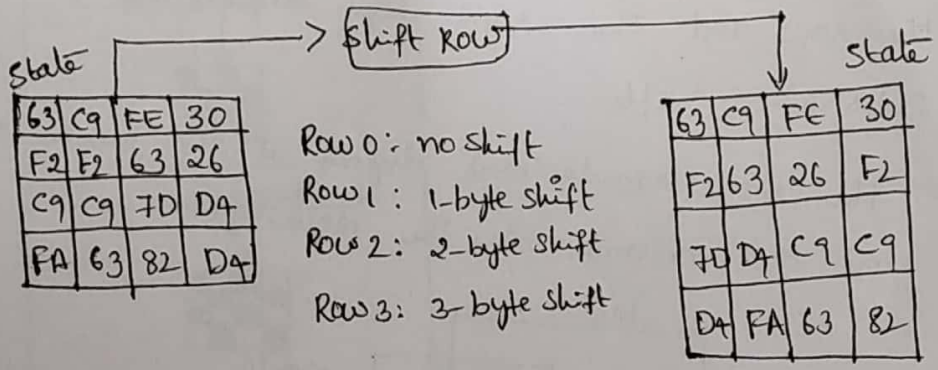


→ In the SubByte transformation, the state is treated as 4×4 Matrix of bytes. Transformation is done one byte at a time.

→ the contents of Each byte is changed, but the arrangement of the bytes in the Matrix remains the Same.

→ on the proces, Each byte is Transformed Independently

Shift Rows: In the encryption, the transformation is called shift Rows and the shifting is to the left. The No. of shift depends on the row Number (0,1,2,&3) of the State Matrix. This Means the row 0 is Not Shifted at all and the last row is shifted three bytes.



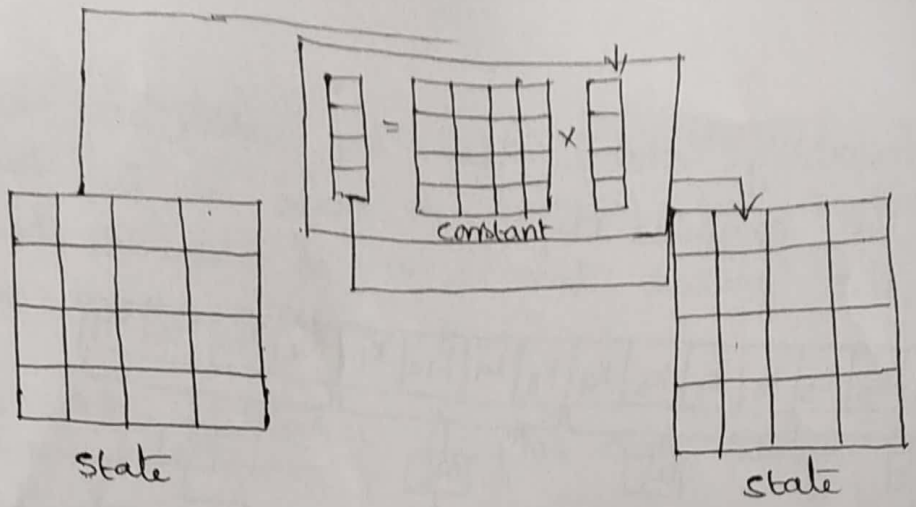
Note that the Shift Rows transformations operations one row at a time.

Mixing: AES defines a transformation, called Mix Columns. In AES the constant Matrices used for these transformations.

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

→ The Mix Columns transformation operates at the column level; it transforms each column of the State to a New column.

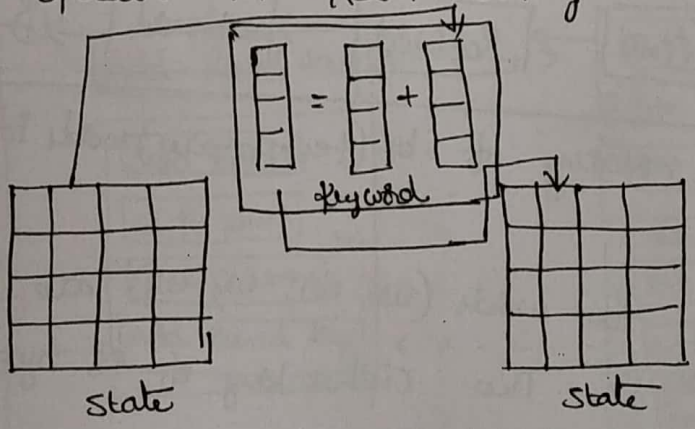
→ The transformation is actually the Matrix Multiplication of the State column by a constant Square Matrix.



Add Round Key: Add Round Key also proceeds one column at a time. It is similar to Mix Columns.

→ In Mix columns multiplies a constant square Matrix by each state column; Add Round Key adds a round key word with each state column Matrix.

→ The operation in Mix Columns is Matrix Multiplication the operation in Add Round Key is Matrix Addition

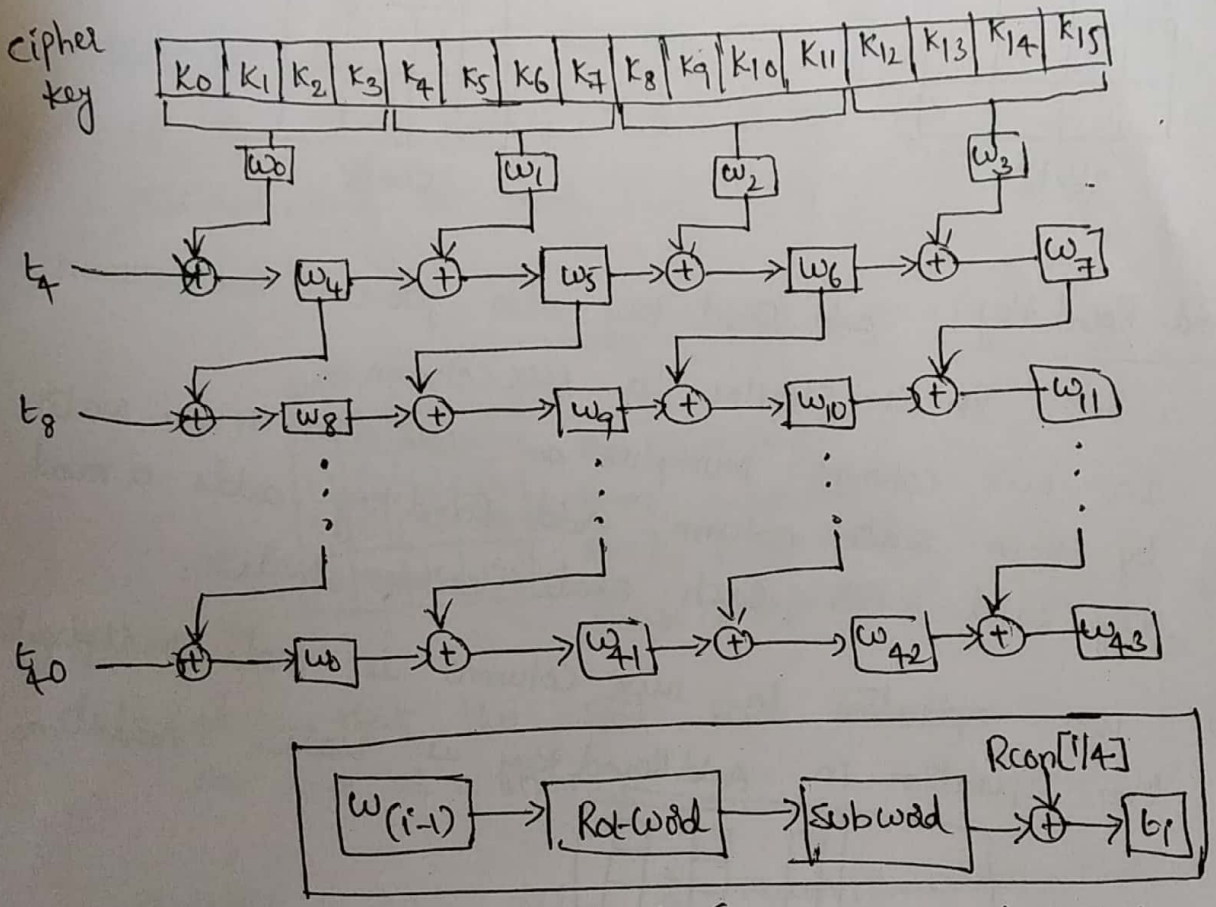


*** Key Expansion in AES

To Create a Round key for each round, AES uses the key Expansion process.

If the No. of rounds is Nr, the key Expansion Creates Nr+1 128-bit round keys from on single 128-bit Cipher Key.

The following shows how 44 words are made from the original key.



Making of t_i (temporary) words $i = 4N_k$.

The process is

→ The first four words (w_0, w_1, w_2, w_3) are made from the cipher key. The cipher key of 16 bytes (K_0 to K_{15}). The first four bytes (K_0 to K_3) become w_0; the next four bytes (K_4 to K_7) become w_1, and so on.

→ The rest of the words (w_i for $i = 4$ to 43) are made as follows.

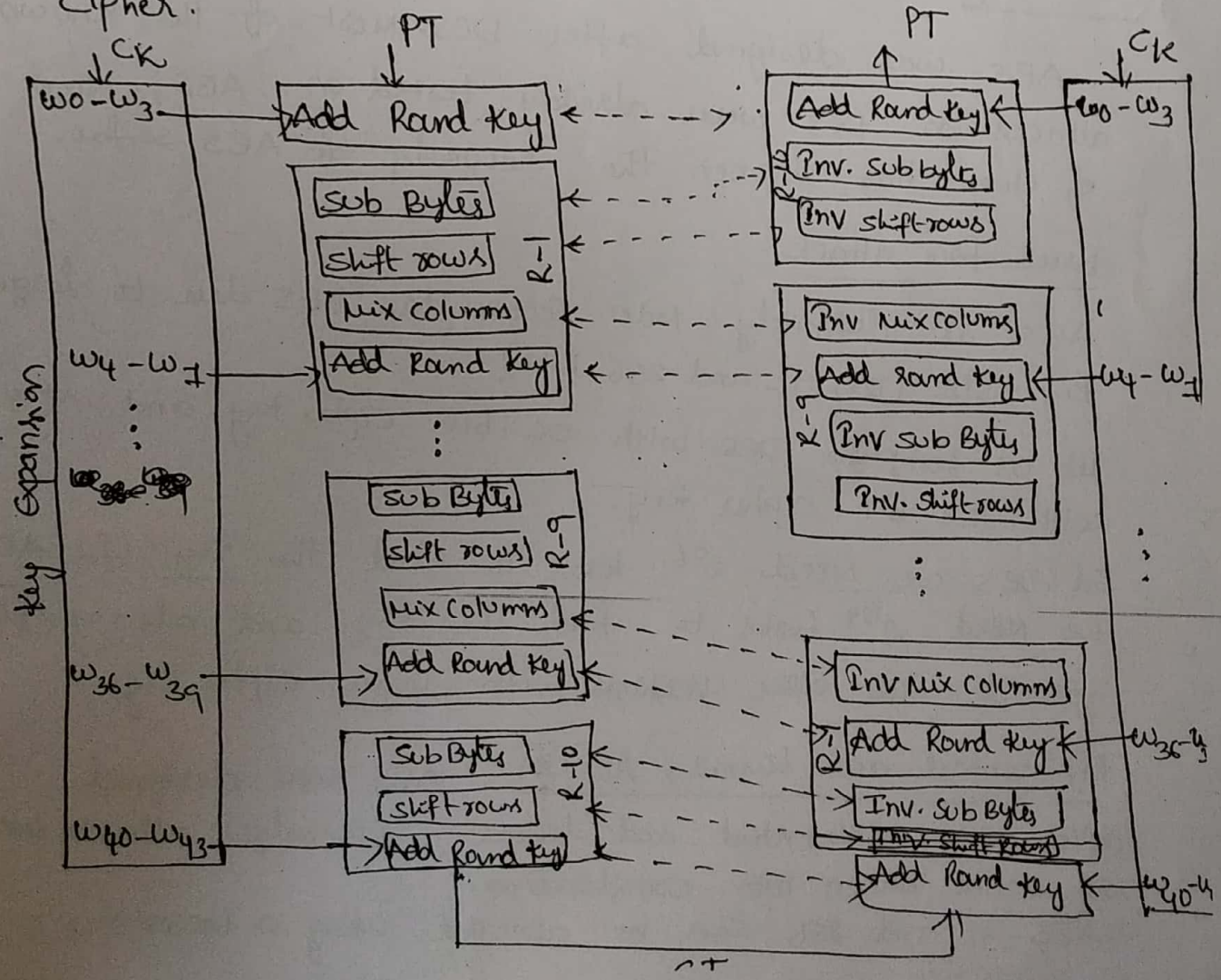
a) if $(i \bmod 4) \neq 0$, $w_i = w_{i-1} \oplus w_{i-4}$. this means each words is made from the one at the left and the one at the top.

b) if $(i \bmod 4) = 0$, $w_i = t \oplus w_{i-4}$. Here t, a temporary word, is the result of applying two routines, SubWord and RotWord, on w_{i-1} and XORing the result with a round constant, Rcon.

The AES Cipher: (Cipher and reverse cipher)

AES uses four types of transformations for encryption and decryption. In the standard, the encryption alg is referred to as the cipher and the decryption alg is referred as the Inverse cipher.

In the AES, the order of transformations in each round is Not the same as in the cipher and reverse cipher.



First, the order of subBytes and shiftRows is changed in the reverse cipher. Second, the order of MixColumns and AddRoundKey is changed in the reverse cipher.

Analysis of AES

Following is a brief review of the 3-characteristics of AES.

→ Security
→ Implementation
→ Simplicity and cost.

Security:

→ AES was designed after DES. Most of the known attacks on DES were already tested on AES;. None of them has broken the security of AES so far.

Brute-force Attack:

AES is definitely more secure than DES due to large-key size (128, 192 and 256 bits).

Let us compare DES with 56 bit cipher key and AES with 128-bit cipher key.

For DES we need 2^{56} tests to find the key; for AES we need 2^{128} tests to find the key. and also AES provide two other versions with longer cipher keys.

Differential and linear Attacks: AES was designed after DES. Differential and linear cryptanalysis attacks were no doubt taken into consideration.

→ AES-192 and 256 can be attacked using a technique

known as "related key cryptanalysis"..

Implementation:

AES can be Implemented in s/w, h/w and firmware. The Implementation can use table lookup process & routines that use a well-defined algebraic structure.

→ The transformation can be either byte-oriented (or) word oriented..

Simplicity and cost.

→ The Algs used in AES are so simple that they can be easily Implemented using cheap processors and a minimum amount of Memory.