Chapter 5

Morden Symmetric key cipher

The treditional symmetric key ciphers are "character oriented ciphers". But withe the advent of the computer we need "bit oriented ciphers". Because the information to be Encrypted is not just a text It can also consist of number, graphics, cuudio & video

In addition when the character is treated at bit level, each character is replaced by 8 bits. which means No. of symbols becomes 8 times larger.

Mixing a larger number of symbols increases Security.

~~Morden~~ Modern Block cipher :- A symmetric-key modern Block cipher encrypts an n-bit block of plaintext or decrypt an n-bit block of ciphertext.

The encryption & decryption algorithm uses a k-bit key.

If the message has fewer then n-bits, padding must be added to make it an n-bit block, if the message has more than n bits, It should be divided into n-bit blocks. & Padding is added to the last block.

Common values for n :- 64, 128, 256 or 512.

Substitution or Transposition:-

A modern block cipher can be designed to act as substitution cipher or a transposition cipher. This is the same idea as is used in traditional ciphers. the only difference is that the symbols to be substituted or transposed are bits instead of character.

If the cipher is designed as a substitution cipher a 1-bit or a 0-bit in plaintext is replaced with either a 0 or a 1.
This means that the Plaintext & ciphertext has different no. of 1's.

If the cipher is designed as a transposition cipher, the bits are only reordered.
This means that the plaintext and ciphertext has same no. of 1's.

The inherent characteristics of transposition makes the cipher vulnerable to Brute-force-attack.

[To resistant a brute-force-attack, a mordon block cipher needs to be designed as a substitution cipher]

A full size key for transposition cipher the key is $\lceil \log \lfloor n \rceil \rceil$ bit long

&

for ~~Full~~ Substitution cipher the full size key has $\lceil \log (2^n) \rceil$ bits.

#Partial Size key :- Actual cipher cannot use full-size keys because the size of the key become so large especially for a Substitution block cipher.

Example :- a common Substitution cipher is "DES" which uses a 64 Bit block cipher..

If the desigener of DES used a full-Size key. the key would be $\log_2(2^{64}!) \approx 2^{70}$ bits. But the size of key used in DES only 56 bits which is very small fraction of full size key.

This means DES uses 2^{56} mapping out of $2^{(2^{70})}$ possible mappings.

Keyless cipher :- Although a keyless cipher is practically useless by itself, keyless ciphers are used as components of keyed cipher.

Keyless transposition cipher :- A keyless transposition cipher can be thought as

Prewired transposition cipher when it is implemented in Hard ware.

or

can be thought as fixed key transposition when it is implemented is software.

The keyless transposition cipher are called "P-boxes" used as a building blocks of Modern Block cipher.

Keyless Substitution cipher :- The keyless (or fixed key) Substitution cipher can be thought of as Predetermined Mapping from input to output.

The keyless substitution cipher, called "S-boxes" used as a building blocks of ~~mode~~ modern block cipher.

Components of a Modern Block cipher :-

Modern block ciphers normally are "keyed substitution cipher"

However the modern block ciphers are not designed as a single unit.

A mordern block cipher is made of a combination of → transposition unit (P-boxes)
→ Substitution unit (S-boxes)
→ Diffusion and confusion
→ & some other units.

Transposition unit (P-boxes) (Permutation-boxes) :-

~~we have~~ It is a tradisional transposition cipher for characters. Now it is used for transposition of bits.

There are 3 types of P-boxes in modern block ciphers :-

① ~~Straight~~ P-boxes ⇒ n inputs, n outputs
② Expansition P-boxes ⇒ n inputs, >n outputs
③ compression P-boxes. ⇒ n inputs, <n outputs

P-boxes are normally keyless, which means that the mapping is predetermined.

If the P-boxes are implemented in hardware, it is prewired. If ~~It~~ is implemented in software, a "permutation table" shows the rule of mapping.

In Permutation table, the entries in the table are the inputs and the positions of the entries are the output.

Example of Permutation table:-

7	5	9	2	8
3	6	4	10	1

This table show that the first entery come from 7th position.

the 10th is entry came from first position & soon.

Compression P-Boxes :- It is a P-box with n-inputs & m outputs where m<n. Some of inputs are blocked & do not reach the output.

It is keyless. use permutation table that show a rules for transposing a bits.

~~the~~ The Permutation table has m entries which has 1 to n entries with some missing values.

Example of permutation table:-

32x24 P-Boxes (compresion)

01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	06	10	11	12	30	31	32

Compression P-boxes are used when we need to Permute bits and the same time decrease the number of bits for the next stage.

Expansion P-Boxes :- An expansion P-box is a P-box with n inputs & m outputs where m>n. Some of the inputs are connected to more than one input.
It is also keyless. The Permutation table shows the rules for transposing bits. In the Permutation table the m-n entries are ~~duplicated~~ repeated.

Example :-

12 x 16 Permutation table (expansion)

01	09	10	11	12	01	02	03	03	04	05	06	07	08	09	12

→ 1 3 9 12 are repeated character

It is used when we need to permute the bits & the same time increase the number of bits for the next stage.

Invertibility :- A Straight P-box is invertible. This means that we can use a Straight P-box in the encryption cipher & its inverse in the decryption cipher.

The compression & expansion P-boxes have no inverse.

S-boxes :- ~~The~~ ~~inf~~ It is type of substitution cipher.

The input to an S-box could be an n bit word, but the output can be an m-bit word. where m & n are not necessarily the same. Also S-boxes can be keyed or keyless. a modern block cipher uses keyless S-boxes.

Linear vs Non-linear S-boxes :- In an S-box with n inputs & m outputs

inputs are $(x_0, x_1, \ldots x_n)$

& outputs are $(y_0, y_1, \ldots y_m)$

the relationship b/w the Input & output are

$$y_1 = f_1(x_0, x_1, \ldots x_n)$$
$$y_2 = f_2(x_0, x_1, \ldots x_n)$$
$$y_3 = f_3(x_0, x_1, \ldots x_n)$$
$$y_4 \quad \vdots$$
$$y_m = f_m(x_0, x_1, \ldots x_n)$$

In a linear-S-Box, the above relation can be expressed as

$$y_1 = a_{11} x_1 \oplus a_{12} x_2 \oplus \ldots \oplus a_{1n} x_n$$
$$y_2 = a_{21} x_1 \oplus a_{22} x_2 \oplus \ldots \oplus a_{2n} x_n$$
$$\vdots$$
$$y_m = a_{m1} x_1 \oplus a_{m2} x_2 \oplus \ldots a_{mn} x_n$$

{ a(i,j) = 0 if the corrosponding bit is not used else it is 1 }

In Non linear S-Box we cannot have this relationship

Example :-

$$y_1 = x_1 \oplus x_2 \oplus x_3$$
$$y_2 = x_1$$
} linear S-box

$$y_1 = (x_1)^3 + x_2$$
$$y_2 = (x_1)^2 + x_1 x_2 + x_3$$
} Non linear S-box.

Example of S-box ~~with~~ of size 3x2

leftmost bit

	00	01	10	11
0	00	10	01	11
1	10	00	11	01

← Rightmost bit

input 010 → output 01
input 101 → output 00

Invertibility :- A S-box may or may not be invertible. In an invertible S-box the number of input bit should be the same as the number of output bits.

3bit

	00	01	10	11
0	011	101	111	100
1	000	010	001	110

↓ 3bit

Encryption table

3bit

	00	01	10	11
0	100	110	101	000
1	011	001	111	010

↓ 3bit

Decryption table.

input in left box is 001, the output is 101
The input 101 in right table creates the output 001. which shows the two tables are inverse of each other.

Exclusive or :- It is an Important component of modern block ciphers.

Properties that Exclusive or follows:-

1. Closure
2. Associativity
3. commutative
4. Identity Element (Property used in feistel cipher)
5. Inverse (used in feistel cipher)

Compliment :- It is a unary operation.

we are interested in this result

$$\bar{x} \oplus x = 1$$

&

$$x \oplus 1 = \bar{x}$$

Inverse :- The exclusive or operation is self invertible.

$$y = x \oplus k \quad \text{then} \quad x = y \oplus k$$

Circular Shift :- Another component used in modern block ciphers is circular shift operation. Shifting can be to the left or to the right.

The number of positions to be shifted can be used as a key.

Invertibility :- Both circular left & Right shift operations are inverse of each other.

If one is used in the encryption then the other one is used for decryption.

Swap :- The swap operation is a special case of the circular shift operation where $k = n/2$ or shifting from middle.

It is also self invertible.

Split & Combine :- Two other operations found in some block ciphers are split & combine.

The split operation normally splits an n-bit word in the middle, creating two equal-length words.

The combine operation normally concatinate two equal length word to create n-bit word.

These two operations are inverse of each other.

Diffusion and confusion :-

Diffusion :- "The idea of diffusion is to hide the relationship between the ciphertext and the plaintext."

Diffusion implies that each symbol in the ciphertext is dependent on some or all symbols in the plaintext.

In other words, If a single symbol in the plaintext is changed, several or all symbols in the ciphertext will also be changed

confusion :- The idea of the confusion is to hide the relationship blw the ciphertext & the key.

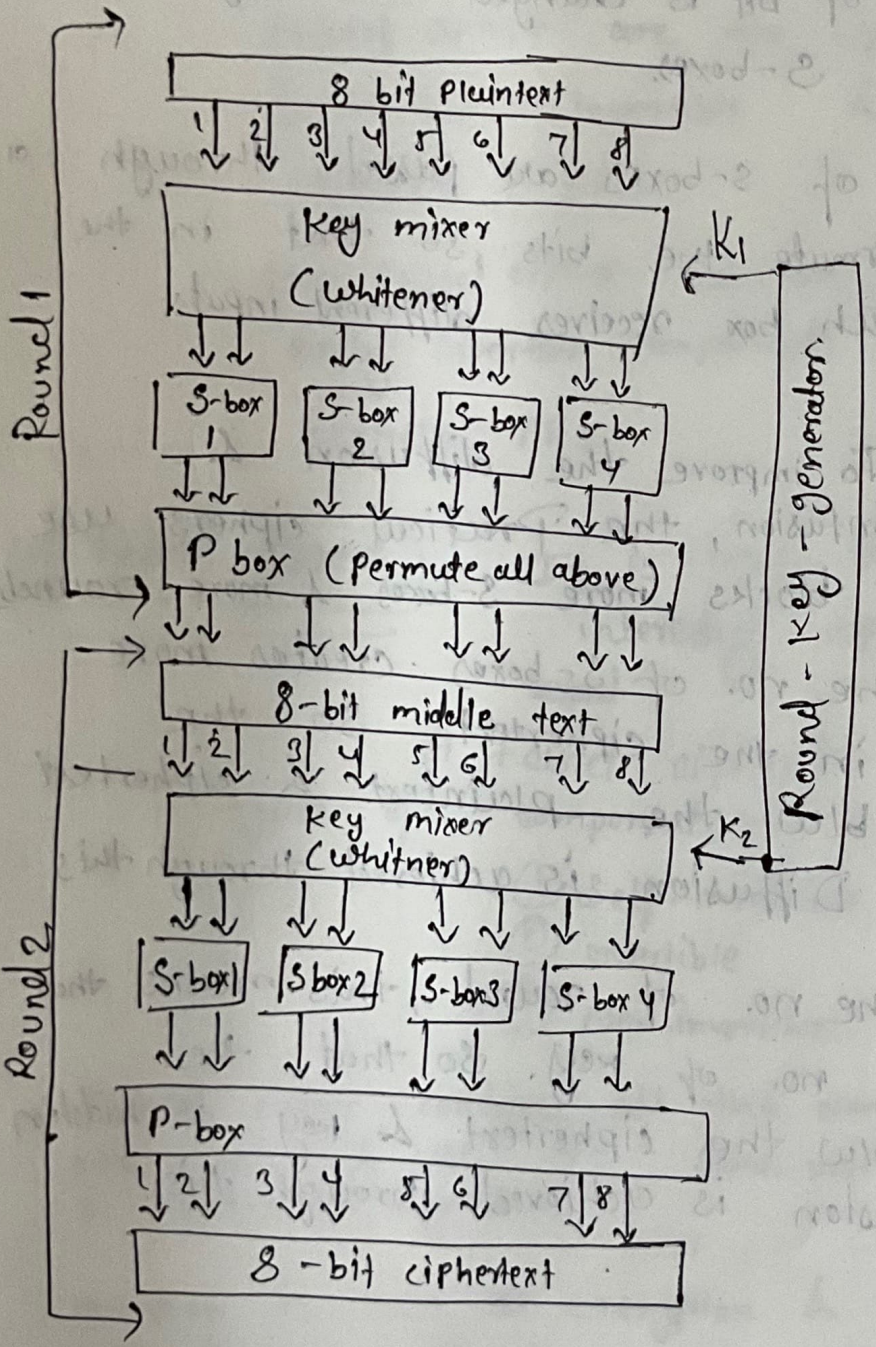
Product ciphers :- A Product cipher is a complex cipher combining Substitution, Permutation & other component. ~~discussed in~~ P

Rounds :- Diffusion and confusion can be achieved using interated Product ciphers where each iteration is a combination of S-boxes, P-boxes and other components. Each iteration is refferred to as a "round."

The block cipher uses a "key Scheduler" or "key generator" that creates a different key for each round. In N-round cipher, the plaintext is

encrypted N times to create ciphertext. & then the ~~plaintext~~ ciphertext is decrypted N times to create the plaintext.

Below is the illustration of Product cipher with 2 rounds (iteration)



(a) The 8-bit text is mixed with the key to hide bits (whiten).
This is done by exclusive-oring the 8-bit word with 8-bit key.

(b) The output of whitner is devided into 4 groups of 2 bit & fed into s-boxes.
The value of bit is changed according to the Structure of S-boxes.

(c) The outputs of s-boxes are passed through a P-box to permute the bits, so that in the next round each box receives different inputs.

Practical ciphers :→ To improve the diffusion & confusion, the Practical ciphers use larger data blocks, more S-boxes & more rounds.

Increasing the no. of S-boxes creates more randomness in the ciphertext so the relationship blw the plaintext & ciphertext is hidden. so Diffusion is achieved through this

Increasing the no. of round, this means that increasing the no. of key. so that the relationship blw the ciphertext & key is hidden So the confusion is achieved through this.

Two classes of Product cipher :-

Morden Block ciphers are all Product ciphers, but they are divided into two classes :-

1. feistel cipher
2. Non feistal ciphers

feistel cipher are the ciphers those uses Both the invertible & noninvertible components.

Example :- DES (Data Encryption Standard)

Non feistel cipher are the ciphers those uses only invertible components.

Example :- AES (Advanced Encryption standard)

feistel ciphers :-

It is very intelligent and interesting cipher. ~~that has been used~~

A "feistel cipher" can have three types of components :-

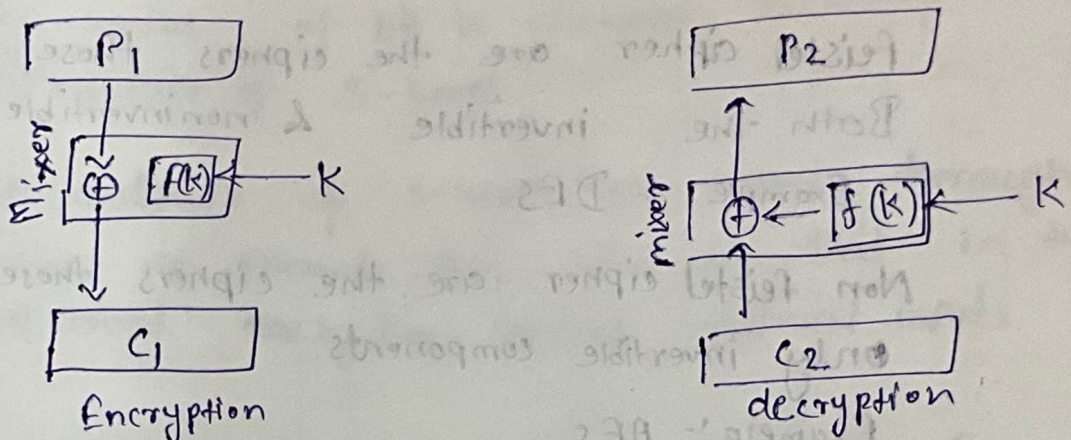
1. Self-invertible
2. invertible.
3. Non-invertible.

feistel cipher combines all the Non-invertible components in one unit & use the same unit in the encryption & decryption algorithm.

Question is How encryption & Decryption algorithms are invers each other if each has Non-invertible components.

The answer is that we can cancel out the effect using XOR operation.

The effects of a noninvertible component in the encryption algorithm can be canceled in the decryption algorithm. If we use Exclusive or operation.



We can Prove that Encryption & Decryption are inverse of each other.

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

Decryption $P_2 = C_2 \oplus f(k)$

$P_2 = C_1 \oplus f(k)$ $\{ \because C_1 = C_2$

$P_2 = (P_1 \oplus f(k)) \oplus f(k)$

$P_2 = P_1 \oplus f(k) \oplus f(k)$

$P_2 = P_1 \oplus 0$

$\boxed{P_2 = P_1}$

the Mixer has Non-Invetible component But the Mixer itself is self invertible.

Example :- Plaintext = 0111

key = 101

$f(k) = k^2$ (Non invertible) ~~Take first &~~

Encryption $C = P \oplus f(k)$

$C = 0111 \oplus f(101)$

$C = 0111 \oplus 1001$

$C = 1110$

Decryption : $P = C \oplus f(k)$

$P = 1110 \oplus f(101)$

Example :- Plaintext = 0111

key = 11

$f(k) = k^2$ (Non invertible)

Encryption $C = P \oplus f(k)$

$C = 0111 \oplus f(11)$

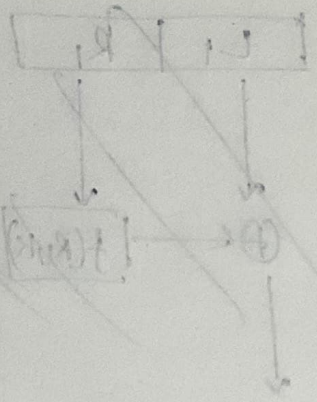
$C = 0111 \oplus 1001$

$C = 1110$

Decryption $P = 1110 \oplus f(11)$

$= 1110 \oplus 1001$

$P = 0111$



Improvement :- with some improvment in the input of the function, we came nearer to the feistel cipher.

In this Improvment we pass some part of Plaintext in the function during Encryption. & the Part of ciphertext during the Decryption.

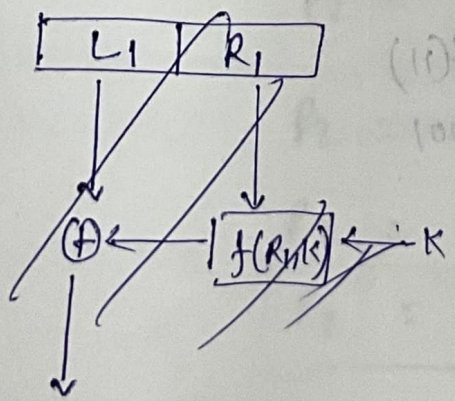
So our function become more complex By using some keyless component & some keyed component.

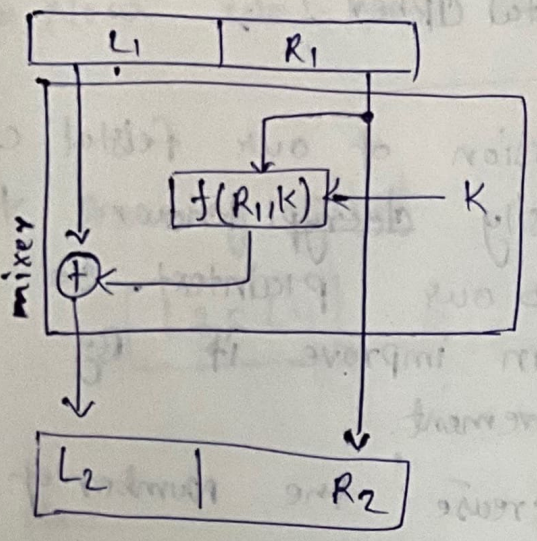
To achieve this goal ; divide the Plaintext & the ciphertext into 2 equal length blocks called Left Block & Right Block.

The right Block Be the input to the function & the left Block be the exclusive-ored with the function output.

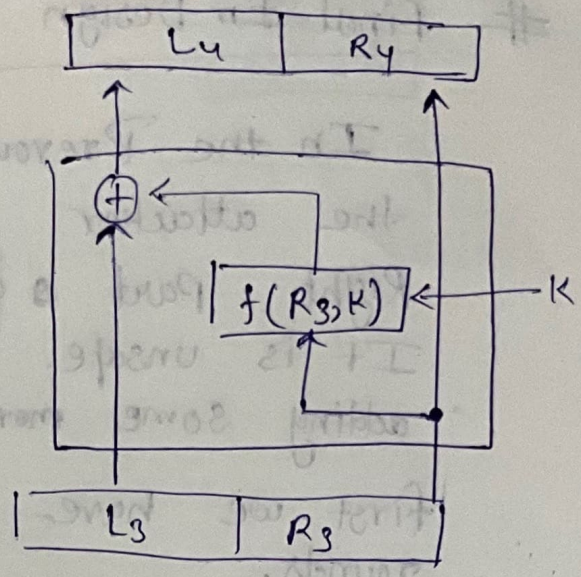
The input to the function must be the same in Both the Encryption & Decryption.

This means that the right section in Encryption & the right section in Decryption must be same.





Encryption



Decryption

Here $L_3 = L_2$ & $R_2 = R_3$ (Not changed during transmission)

$$R_4 = R_3 = R_2 = R_1$$

$$L_4 = L_3 \oplus f(R_3, K)$$

$$L_4 = L_2 \oplus f(R_2, K)$$

$$L_4 = \left(L_1 \oplus f(R_1, K)\right) \oplus f(R_2, K)$$

$$L_4 = L_1 \oplus f(R_1, K) \oplus f(R_1, K)$$

$$L_4 = L_1 \oplus 0$$

$$L_4 = L_1$$

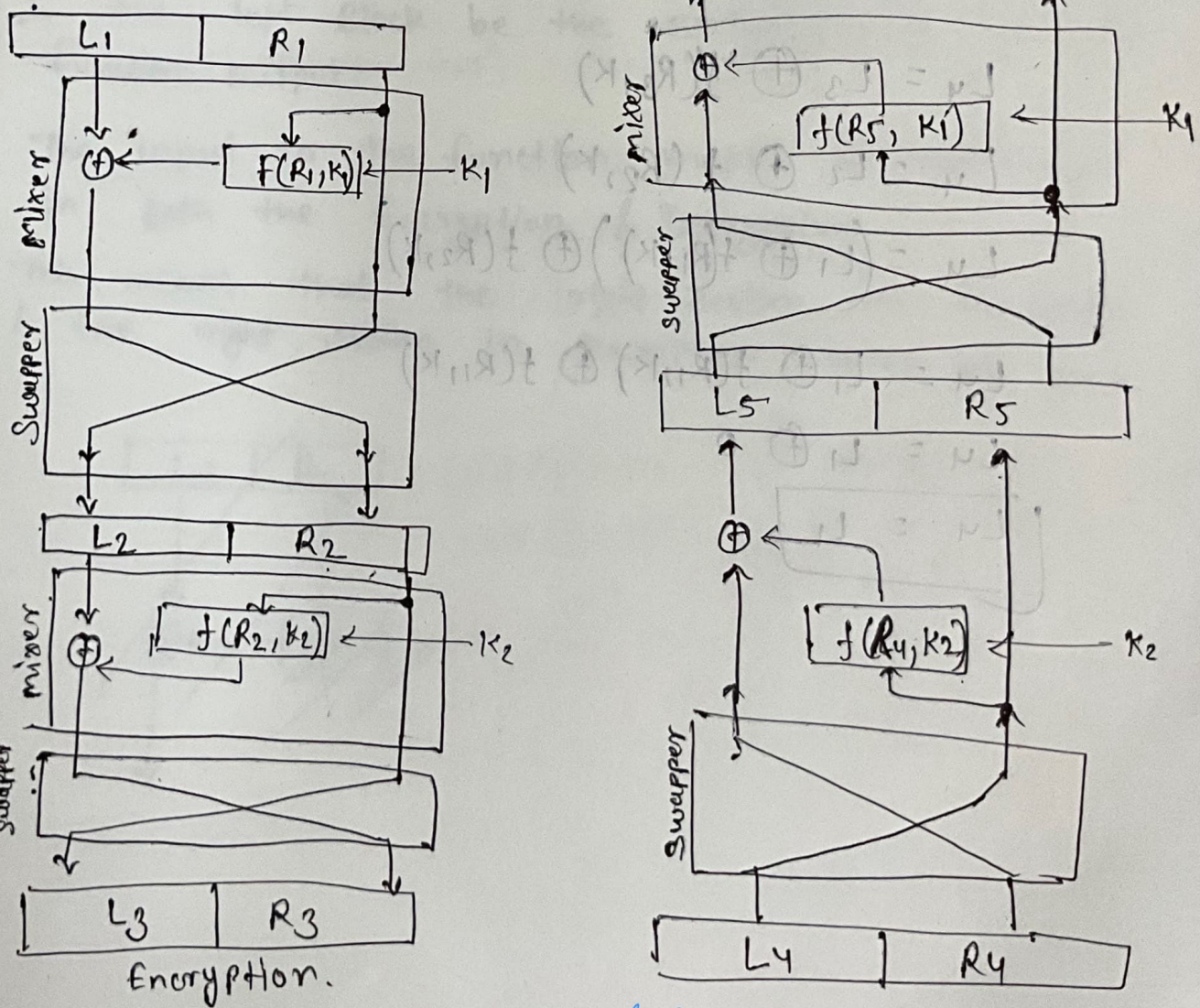
Final ~~Design~~ Design of feistal cipher :-

In the Prevous version of our feistal cipher the attacker can easily ~~decrypt~~ known the Right Part of ~~our~~ our plaintext. So, It is unsafe. we can improve it By adding some more improvement..

first we have to increase the Number of rounds.

Second we add a new element to each rounel a "swapper".

The effect of swapper in encryption is canceled out By the swapper in decryption.



Note:- In Every round key is diff

Now we Have to Prove that $L_1 = L_6$ & $R_1 = R_6$

Assuming that

$L_3 = L_4$ & $R_3 = R_4$

Now

$$L_5 = R_4 \oplus f(L_4, K_2)$$
$$= R_3 \oplus f(L_3, K_2)$$
$$= R_3 \oplus f(R_2, K_2)$$
$$= (L_2 \oplus f(R_2, K_2)) \oplus f(R_2, K_2)$$
$$= L_2 \oplus f(R_2, K_2) \oplus f(R_2, K_2)$$
$$\Rightarrow L_2 \oplus 0$$

$$\boxed{L_5 = L_2}$$

$$\boxed{R_5 = L_4 = L_3 = R_2}$$

Now

$$L_6 = R_5 \oplus f(R_5, K_1)$$
$$L_6 = R_2 \oplus f(L_2, K_1)$$
$$L_6 = L_1 \oplus f(R_1, K_1) \oplus f(L_2, K_1)$$
$$L_6 = L_1 \oplus f(R_1, K_1) \oplus f(R_1, K_1)$$
$$L_6 = L_1 \oplus 0$$

$$\boxed{L_6 = L_1}$$

Non-feistal ciphers:-

A non-feistal cipher uses only invertible components.

A component in the Encryption cipher has the corresponding component in the decryption cipher.

S-boxes must equal no. of inputs & outputs.

No compression or expension P-boxes are allowed. Because they are Not Invertible.

No need to devide plaintext into two halves.

The components of non-feistel cipher are:-

1. Exclusive or operation
2. 2×2 S-boxes
3. Straight P-boxes.

Because each component are invertible, it can be shown that each round is invertible.

Modern Stream Ciphers :-

Stream ciphers are faster than block ciphers. The hardware implementation of a stream cipher is also easier.

Stream ciphers are also more immune to the corruption of bits during transmission.

The main issue in modern stream ciphers is how to generate the key stream $|K = K_n, \ldots K_2 K_1$.

Modern Stream ciphers are divided into two broad categories

1. Synchronous
2. Nonsynchronous.

1. Synchronous Stream Cipher :- In a synchronous stream cipher, the key is independent of the plaintext or ciphertext.

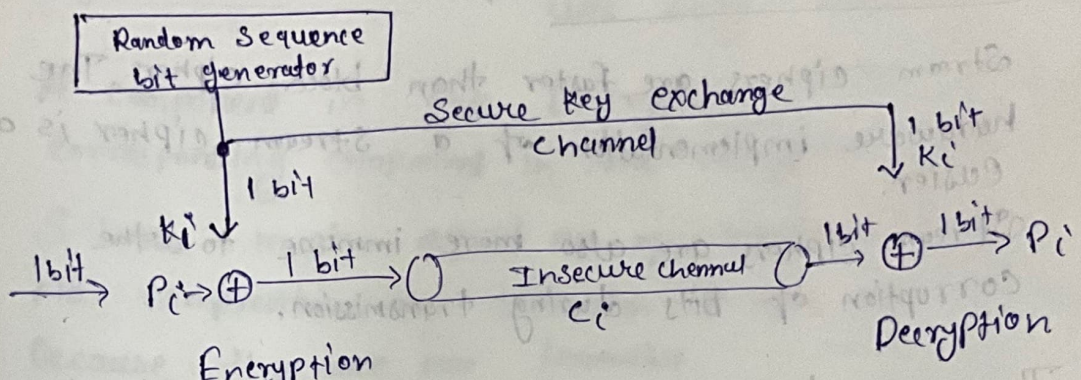
In other words the key stream is generated with no relationship b/w key bits & the Plaintext or ciphertext bits.

Example :-

1st One-Time-Pad :- The simplest & the most secure type of Synchronous Stream cipher is called the one-time-pad. In this cipher we use key stream which is randomly choosen during the Encryption every time.

The encryption & decryption algorithm simply using a Exclusive-or operation

Note :- The XOR operation is used to Encrypt one bit at a time. Not the whole text.



Example :- What is the pattern of the cipher in one-time-pad in following cases :-

case 1 :- Plaintext with all 0's

Because $0 \oplus K_i = K_i$ so the ciphertext is same as the keystream

case2 :- Plaintext with all 1's

Because $1 \oplus K_i = \bar{K_i}$ so the ciphertext is complement of keystream.

case3 :- Paintext with alternative 0's & 1's

In this case the bits in the ciphertext is either the same or compliment

case 4 :- Plaintext with random bits :-

The ciphertext is also a random ~~no~~ stream.

Step 1 :- firstly we convert our text into bits

Step 2 :- Random sequence bit generator generate key of same length as text in bits

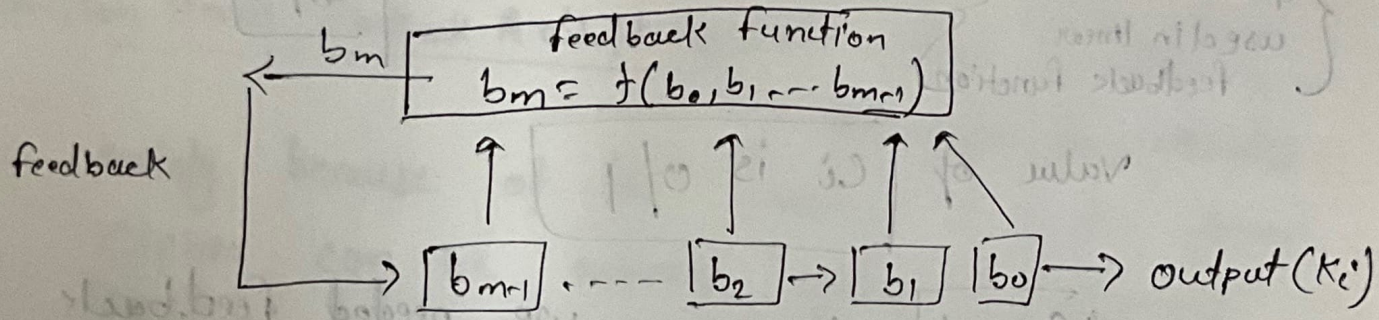
Step 3 :- we perform XOR operation with text bits and key so we got cipher

To Decrypt

we again perform XOR operation of Random sequence bit generated previously (key) with cipher to get text

Feedback Shift Register :-

A feedback shift register is made of a "Shift register" & a feedback function.



Components of shift register :-

① It contains a sequence of m cells, b_0 to b_{m-1} where each cell holds a single bit.

The cells are initialized to m-bit word. called the initial value or seed.

when we need one output bit then every cell is shifted one cell right.

which means each cell give its value to its right cell & take value from its left cell.

The rightmost cell b_0 gives the value as output (K_i).

The leftmost cell b_{m-1} receives its value from a feedback function.

The output of the feedback function is b_m.

Linear feedback shift register :-

In linear feedback shift register b_m is a linear function of $b_0, b_1, \ldots b_{m-1}$

$$b_m = c_{m-1} b_{m-1} \cdots c_2 b_2 + c_1 b_1 + c_0 b_0$$

{ + & ⊕ is used in linear feedback function }

$c_0 \neq 0$

Value of c_i is 0/1

$c_0 = 1$ because we needed feedback from the output.

The addition operation is also exclusive-or operation.

$$b_m = c_{m-1} b_{m-1} \oplus \cdots \oplus c_2 b_2 \oplus c_1 b_1 \oplus c_0 b_0$$

Example :- Create a lenear feedback shift register with 5 cells in which

$$b_5 = b_4 \oplus b_2 \oplus b_0$$

If $c_i = 0$ the b_i has no role

So $c_4 = c_2 = c_0 = 1$ (Involve in calculation)

$c_3 = c_1 = 0$ (No role in calculation)

Random Bits :- x_1, x_2

here we take 2 because we want only 0 or 1 and when we take 2 it give 0

Choose :- $x_1 = x_2 = 1$

$x_3 = x_1 + x_2 = 1+1 \Rightarrow 2 \pmod 2 \Rightarrow 0$

$x_4 = x_3 + x_2 = 0+1 \Rightarrow 1 \pmod 2 \Rightarrow 1$

$x_5 = x_4 + x_3 = 1+0 \Rightarrow 1 \pmod 2 \Rightarrow 1$

$x_{n+2} = x_n + x_{n+1} \pmod 2$

1, 1, 0, 1, 1, ---------- random bits

Recurrence Relation

In general we choose a length l and then coff $c_0, c_1, \dots c_{l-1} \in \{0, 1\}$, and form the recurrence relation

$$x_{n+l} \equiv c_0 x_n + c_1 x_{n+1} + \dots + c_{l-1} x_{n+l-1} \pmod 2$$

Example :- $x_{n+2} \equiv x_n + x_{n+1} \pmod 2$

$l = 2$; $c_0 = 1$ $c_1 = 1$

Example :- $x_{n+5} \equiv x_n + x_{n+3} \pmod 2$

$l = 5$; $c_0 = 1$, $c_1 = 0$ $c_2 = 0$, $c_3 = 1$, $c_4 = 0$

here we have length 5 so

$x_1 = 0$, $x_2 = 1$, $x_3 = 0$, $x_4 = 1$, $x_5 = 1$

$x_{n+5} \Rightarrow x_6 \Rightarrow x_{1+5}$ $[n = 1]$

$x_6 = x_1 + x_4 \equiv 0+1 \pmod 2 \equiv 1 \pmod 2$

$x_7 = x_2 + x_5 \equiv 1+1 \pmod 2 \equiv 0 \pmod 2$

$x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9 \dots$

0, 1, 0, 1, 1, 1, 0, 1, ---------

Q Suppose Sumit knows the length $l = 3$ and the known random bits sequence

0, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, ------

then he can find the corresponding congrence used for generating these random no's

Sol^n :-

$$x_{n+3} \equiv c_0 x_n + c_1 x_{n+1} + c_2 x_{n+2} \pmod 2$$

$n = 1 \quad 1 \equiv x_4 \equiv c_0 x_1 + c_1 x_2 + c_2 x_3 \Rightarrow c_0\,0 + c_1(1) + c_3(1) \bmod 2$

$n = 2 \quad 0 \equiv x_5 \equiv c_0 x_2 + c_1 x_3 + c_2 x_4 \Rightarrow c_0(1) + c_1(1) + c_2(1) \bmod 2$

$n = 3 \quad 0 \equiv x_6 \equiv c_0 x_3 + c_1 x_4 + c_2 x_5 \Rightarrow c_0(1) + c_1(1) + c_2(0) \bmod 2$

$$0 + c_1 + c_3 = 1 \quad \text{---} (1)$$
$$c_0 + c_1 + c_2 = 0 \quad \text{---} (2)$$
$$c_0 + c_1 + 0 = 0 \quad \text{---} (3)$$

$$\begin{array}{r} 0 + c_1 + c_3 = 1 \\ c_0 + c_1 + c_2 = 0 \\ \hline -c_0 + 0 + c_3 - c_2 = 1 \end{array}$$

$$-c_0 - c_2 + c_3 = 1$$

$$\begin{array}{r} c_0 + c_1 + c_2 = 0 \\ c_0 + c_1 + 0 = 0 \\ \hline c_2 = 0 \end{array}$$

Put $c_2 = 0$

$$-c_0 + c_3 = 1$$

$$\begin{array}{r} 0 + c_1 + c_3 = 1 \\ -c_0 \quad\quad + c_3 = 1 \\ \hline c_0 - c_1 = 0 \end{array}$$

$$c_0 = 1 \quad c_2 = 0 \quad c_1 = 1$$

$$x_{n+3} = x_n + x_{n+1} \pmod 2$$

Problem 1 :- Linear feedback shift register machine uses the recurrence relation

$$x_{n+3} = x_n + x_{n+1} + x_{n+2} \pmod 2$$

and initial values $x_1 = x_2 = 1, x_3 = 0$

find $x_4, x_5, \ldots, x_{10}$

$$x_4 = x_{1+3} \Rightarrow x_1 + x_2 + x_3 \Rightarrow 1 + 1 + 0 = 2 \pmod 2$$
$$= 0$$

$$x_5 = x_{2+3} \Rightarrow x_2 + x_3 + x_4 \Rightarrow 1 + 0 + 0 = 1 \pmod 2$$

Problem 2 :- The sequence 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0 was produced by a recurrence relation

$$x_{n+3} \equiv c_0 x_n + c_1 x_{n+1} + c_2 x_{n+2} \pmod 2$$

Find c_0, c_1, c_2

$$1 \equiv x_4 = x_{1+3} = c_0 x_1 + c_1 x_2 + c_2 x_3 = c_0(0) + c_1(1) + c_2(1) \bmod 2$$

$$0 \equiv x_5 = x_{2+3} = c_0 x_2 + c_1 x_3 + c_2 x_4 = c_0(1) + c_1(1) + c_2(1) \bmod 2$$

$$1 \equiv x_6 = x_{3+3} = c_0 x_3 + c_1 x_4 + c_2 x_5 = c_0(1) + c_1(1) + c_2(0) \bmod 2$$

$$0 + c_1 + c_2 = 1$$
$$c_0 + c_1 + c_2 = 0$$
$$c_0 + c_1 + 0 = 1$$

$$0 + c_1 + c_2 = 1$$
$$c_0 + c_1 + c_2 = 0$$
$$- \quad - \quad -$$
$$-c_0 = 1$$
$$c_0 = -1 \bmod 2$$
$$c_0 = 1$$

$$0 + c_1 + c_2 = 1$$
$$c_0 + c_1 + 0 = 1$$
$$- \quad - \quad -$$
$$-c_0 + c_2 = 0$$
$$-(1) + c_2 = 0$$
$$c_2 = 1$$

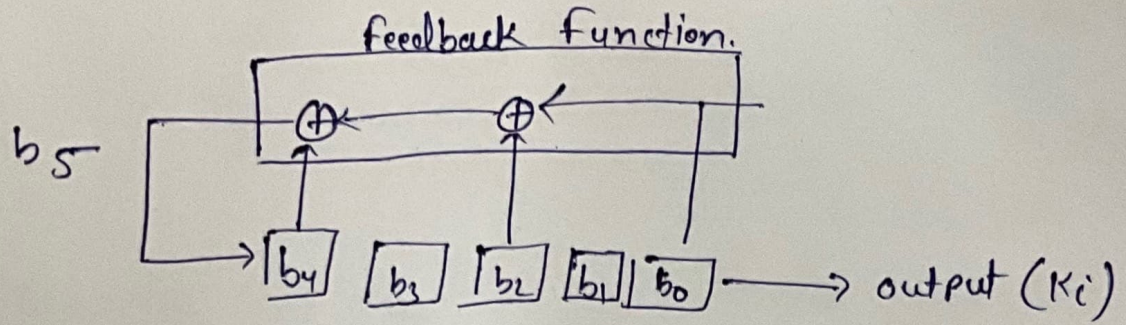
$$c_0 + c_1 + c_2 = 0$$

$$1 + c_1 + 1 = 0$$

$$c_1 = -2 \bmod 2$$

$$c_1 = 0$$

$$c_0 = 1 \;,\; c_1 = 0 \;,\; c_2 = 1$$



Non linear feedback Shift Register:- The LFSR is vulnerable to attacks mainly because of its linearity. A Better Stream Cipher can be achieved using a nonlinear feedback shift register (NLFSR).

All the things are same except that bm is non linear function of $b_0, b_1, b_2 \ldots b_{m-1}$.

bm is a combination of OR, AND & Compliment

Example.

$$b_4 = (b_3 \text{ AND } b_2) \text{ OR } (b_1 \text{ AND } \overline{b_0})$$

Nonsynchronous Stream Cipher:- In a nonsynchronous Stream cipher each key in the key stream depends on prevous plaintext or ciphertext.