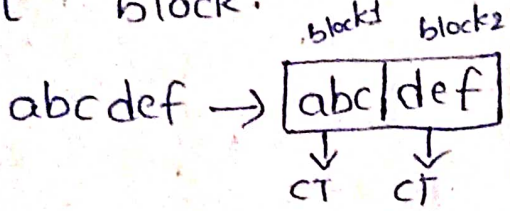
Unit-II

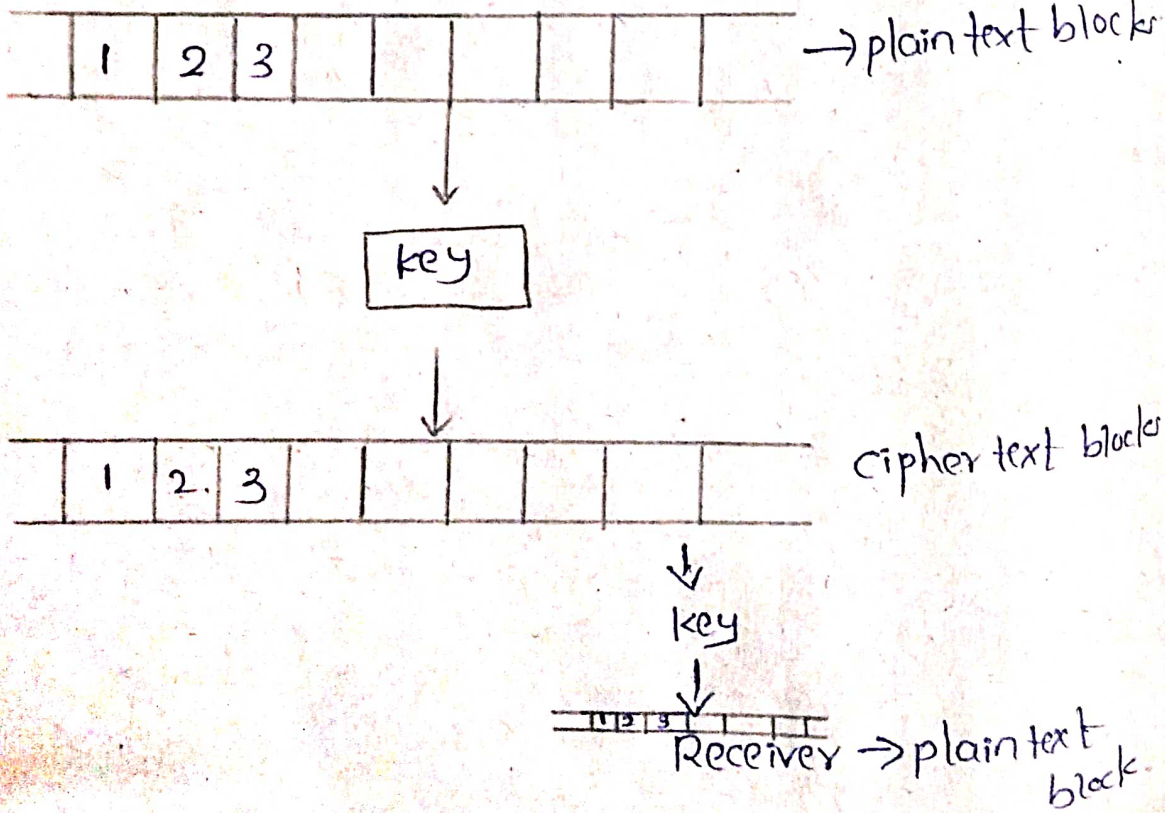
Symmetric key ciphers

Block cipher principles:-

Plain text is divided into no. of blocks. Each individual block it will generate a individual cipher text block.



* Each block can have a size of (40,56,64,128, 256 bits)
* plain text blocksize is equal to cipher text block size
* By using key, the plain text blocks are converted into cipher text block.



Block cipher principles:-

The design principles of Block cipher

1) Number of Rounds
2) Design of function
3) key schedule Algorithm

1) Number of Rounds:-

* Depending on the Algorithm, Each and every algorithm will have several rounds. (Based on)
* Based on the more no. of rounds, that much harder to the hacker.
* 10R Algorithm is easy to hack when compare to 20R Algorithm.
* so, No. of Rounds should be more.

2) Design of function F:-

* You should design a function F which will be very much complicated to understand.
* If the function is very much complicated to understand, that much more time hacker will take to decode the data.
* you have to take Non-linear functions, because linear functions are easy.

3) key schedule Algorithm:-

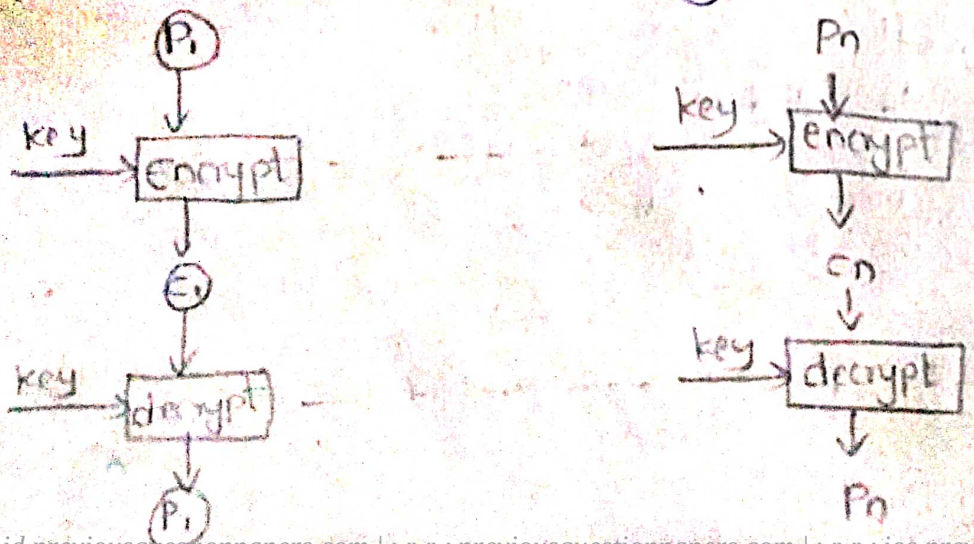
* we should be very careful when we are generating a key, because key is very important.
* Even though a minor change in key, there will be lot of changes in cipher text.

*The modes of operation of Block cipher.

1) ECB - Electronic code book
2) CBC - cipher block chaining
3) CFB - cipher feed back
4) OFB - output feed back
5) CTR - counter.

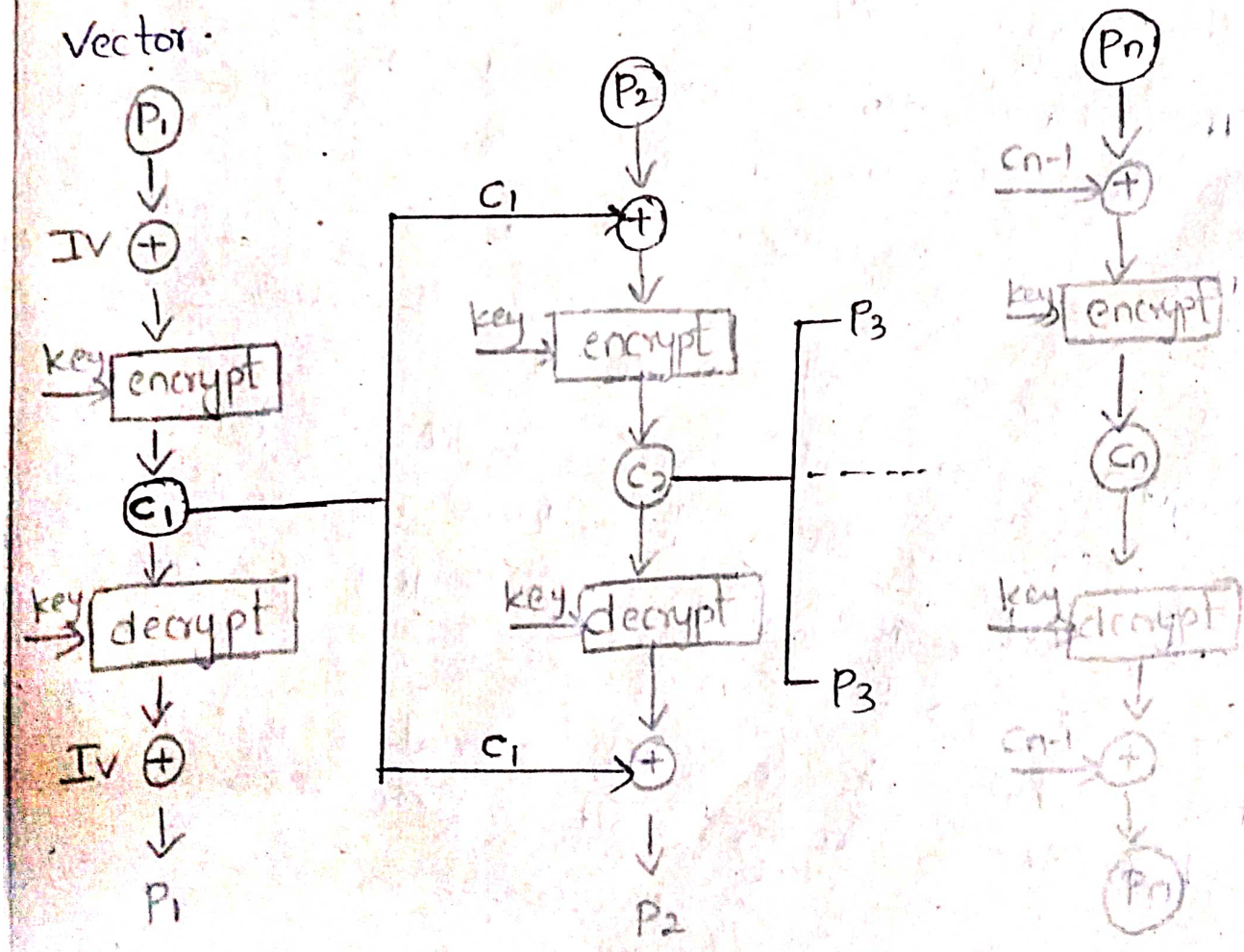
1) Electronic code book (ECB).

* The plain text is divided into no. of blocks, and encrypt the plain text with the help of the key and we get cipher text. and again by using the key, you can decrypt the cipher text into plain text
* The process will continue upto P_n (plain text n)
* The properties are:-
  i) Block size is equal to 64 bits.
  ii) key is same in everywhere.
  iii) The size of PT and CT should be same
* P is divided into $P_1, P_2, P_3 \ldots P_n$ at the last you will combine the $P_1, P_2, P_3 \ldots P_n$ to get the orginal P.
* This is suitable for only short messages.



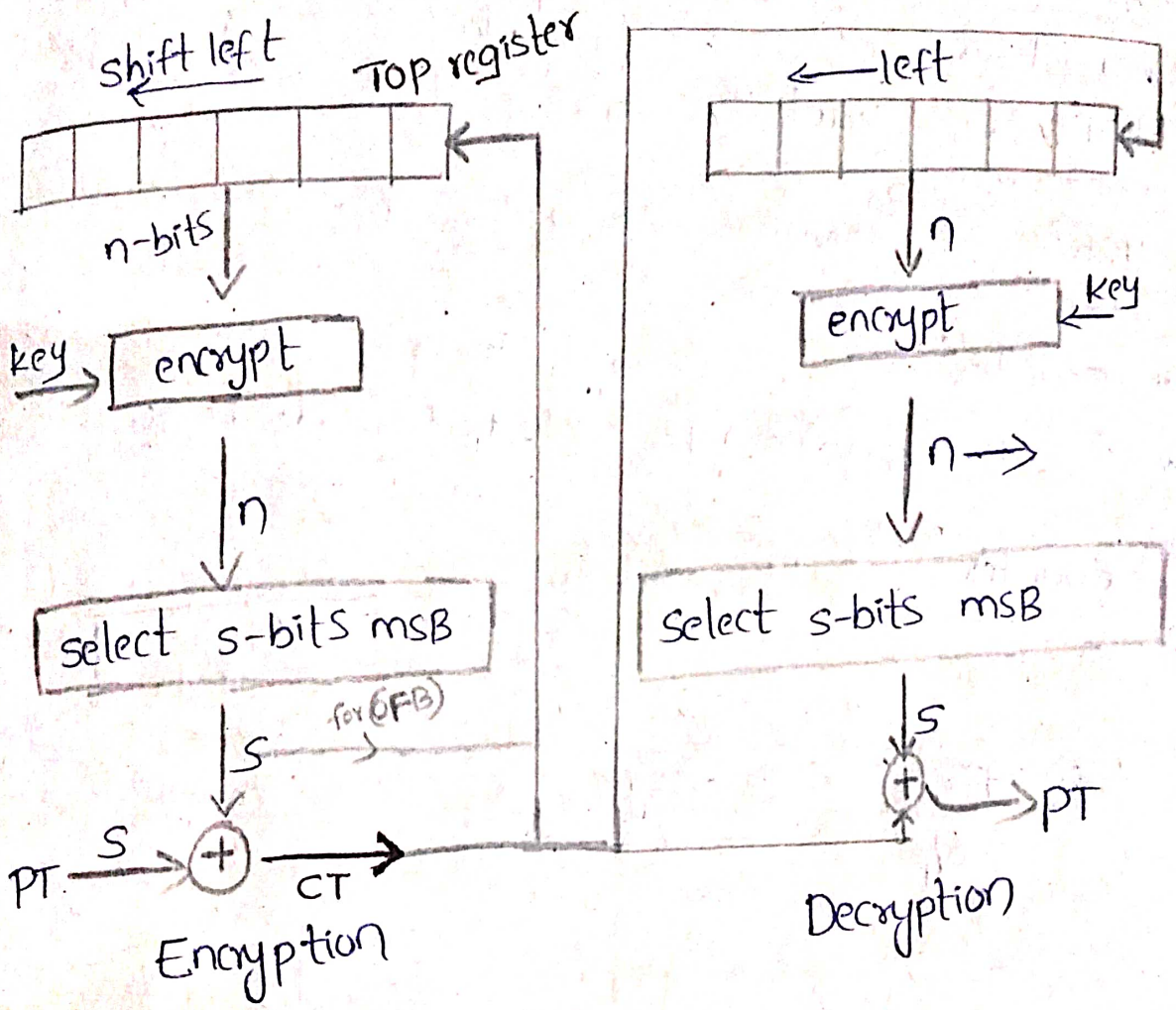
2) cipher block chaining (CBC):-

* The output of the XOR operation blw the plain text and initialization vector, is enters as a input.
* With the help of key and input, the encryption process happens.
* After encryption, the cipher text will get.
* you will decrypt the ~~key~~ ciphertext with the help of key, again that will be XOR with the initialization vector then we get the plain text.
* For the next step, we are using previous cipher text as a initialization vector.
* This process will continue, upto Pn.
* For Pn, the Cn-1 will act as a initialization Vector.



3) cipher feedback (CFB):

* In this we have shift register, the size of the shift register is n bits. (or) TOP (or) TOP
* The size of plaint text is s bits.
* we will encrypt the n bits of TOP register with the help of key then we will get the n bits of cipher text.
* Actually the size of plain text is s bits, so, from that s bits you have to select MSB (most significant bits)
* From the n bits of cipher text, you have to select s-bits as (MSB)
* The s value should be $1<s<n$
* Those s bits are XOR with the plain text. then you will get final cipher text.
* The final cipher text will be given to TOP register as a feedback.
* The top register will shifting towards the left side. because in order to accomidate the cipher text inside it.
* Again you will have n bits, n bits will encrypted by using key. Then you will get cipher text, in that cipher text you have to select S-bits as msB.
* The s bits do the XOR operation with the cipher text to get the plain text.



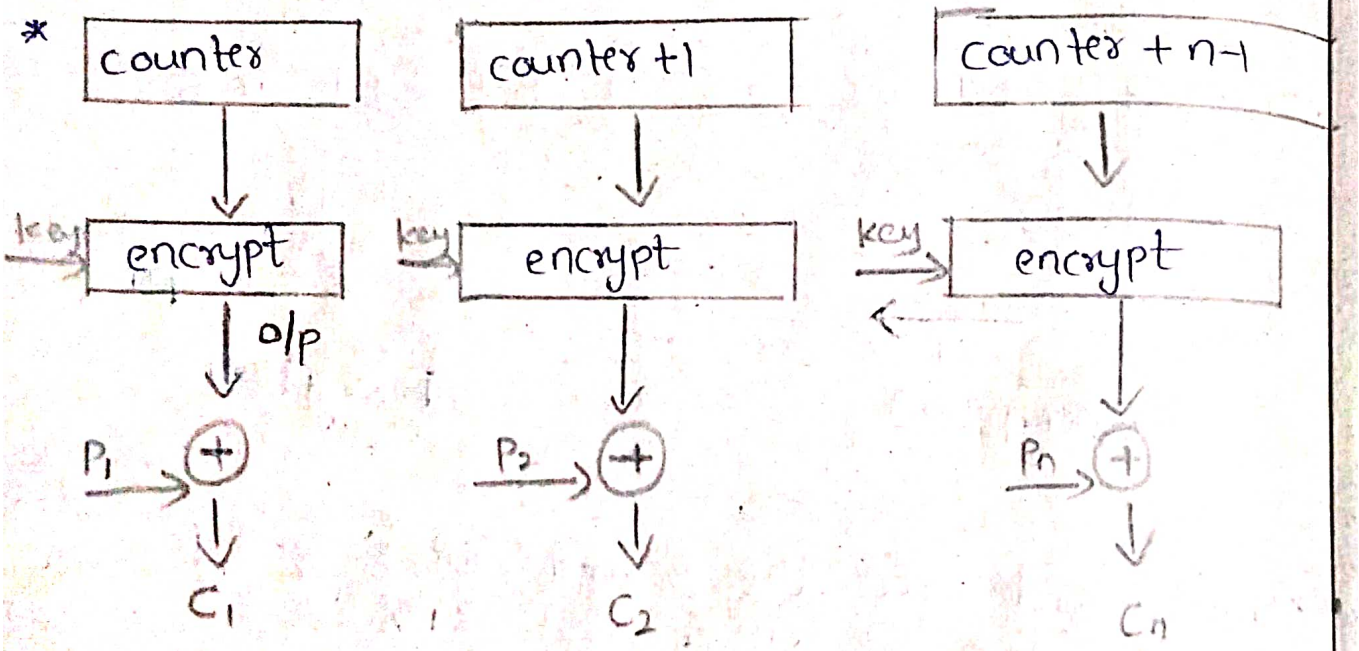
* Both sides, we use encrypt function only
* Top register is filled with IV.

4) output feedback mode : (OFB)

* Same as cipher feedback mode.
* But instead of cipher text, output is given as feedback.

5) Counter mode (CTR):-

* Instead of taking plain text directly, we ~~take a count~~ are giving it, in the form of counter.
* Taking a counter and giving plain text in the form of counter.
* counter size is equal to plain text size.
* It is similar to Electonic code book.



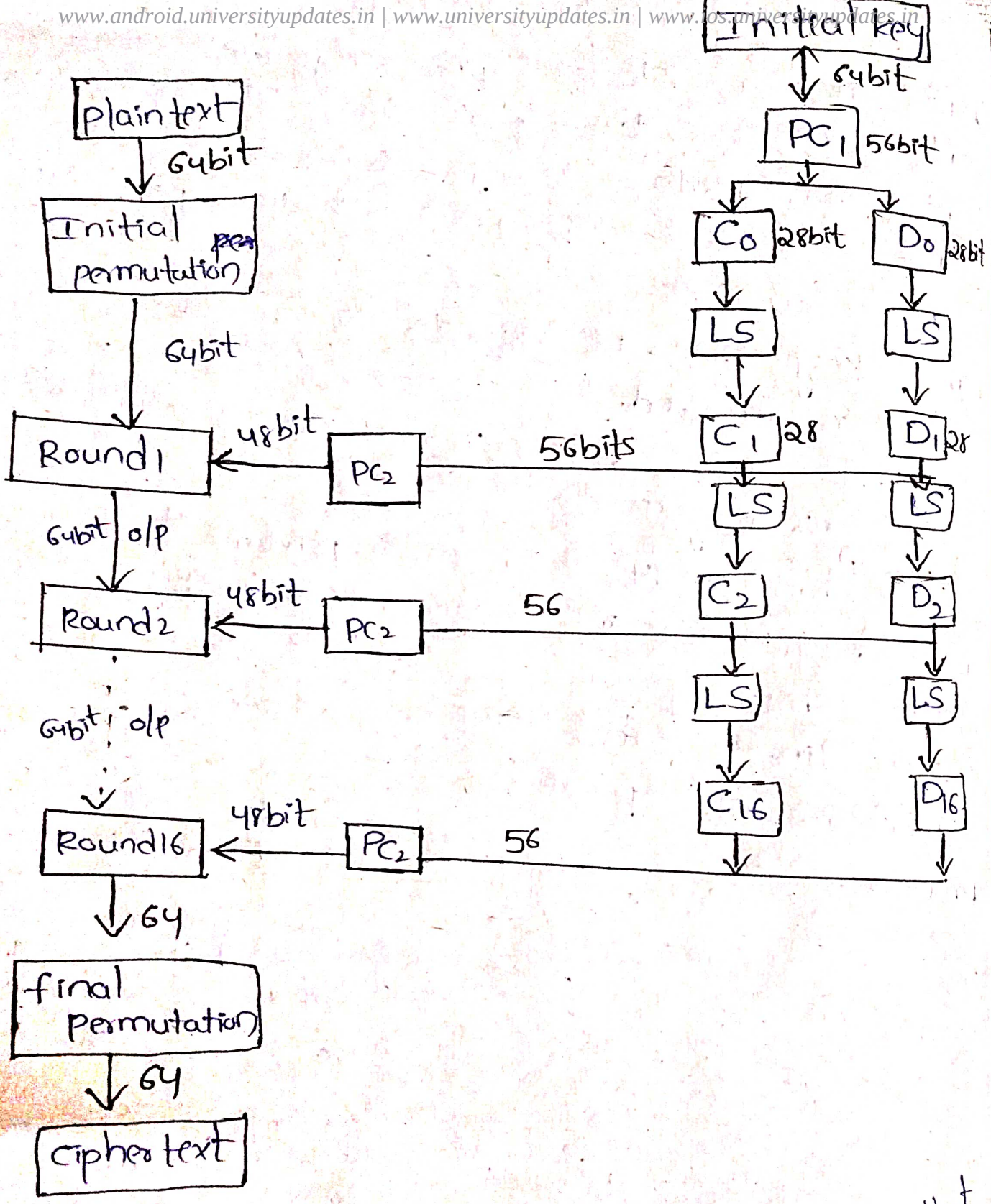
* The counter will encrypted with the help of key, and the XOR operation will done blw the o/p and plain text (P_1) then you will get the cipher text (C_1). Do Reverse proces to get plain text
* For each next step, the counter will be added or incremented with 1
* For nth step, the counter is counter+n-1
* This proress will continue..

2) Data Encryption Standard (DES):-

* It is a block cipher algorithm.
* It converts the plain text into cipher text.
* It has total of 16 Rounds.
* The text size of plain text and cipher text is 64 bits.
* The key size is 48 bits. The remaining 16 bits are removed.
* 8 bits are removed for parity and 8 bits for rearrangement.
* In each round, (4) steps are performed.
  i) Dividing bits into two parts - 32 bits for each
  ii) Bit shuffling
  iii) Non linear substitutions
  iv) Exclusive OR operations.

Process:-

~~Initial~~ key



*In PC1, initially, 64 bits are there. In that 8 parity bits are to be removed from every 8th position.

$64 = (8 \times 8)$, i.e., 56 bits

$64 - 8 = 56$

* Then apply left circular shift after dividing 56 bits into 2 parts: C_0 and D_0, each having 28 bits.
* D_1 and C_1 are obtained as Result.

left circular shift:-

* Move the bits based on Round number.
* For Rounds 1,2,9,16 - 1 bit shift
  other Rounds - 2 bit shift
* Here in PC2, C_1 and D_1 are combined to form 56bits again. permutted chocie 2 is applied,
* 56 bits are rearranged, permutted and in that 48 bits are selected.
* For Round 1, 48 bits are the key.

Round 1:- i/p - 64bit + 48 bit key
o/p - 64bit

Round 2:- i/p - 64bit + 48bit key
o/p - 64bit

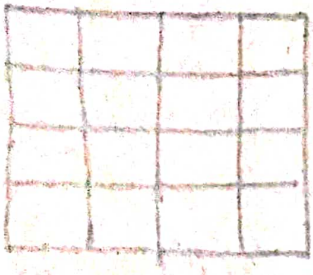
Round 16:- i/p - 64bit + 48bit key
o/p - 64bit

* At the last, the cipher text is 64bit.

AES (Advanced encryption Standard)

* It is a block cipher Algorithm.
* It has i/p array, stateArray and a key array

Input Array:-



each cell = 1 byte / 8 bits
Total cells = 16 cells
16 x 8 = 128 bits
4 words (32 each) = 128 bits.

PT is represented in the i/p Array.

State Array:-

0th word | 1st word | 2nd word | 3rd word

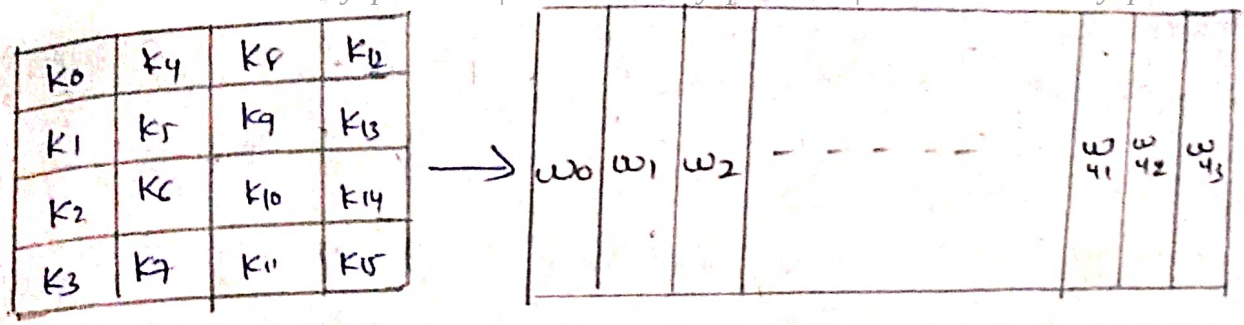
$S_{0,0}$ (0th bit of 0th word)	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$ → 0th bit of 3rd word.
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$

* It used to store intermediate states (results) within the Rounds.
* The result is stored in the form of four words.

Key Array:-

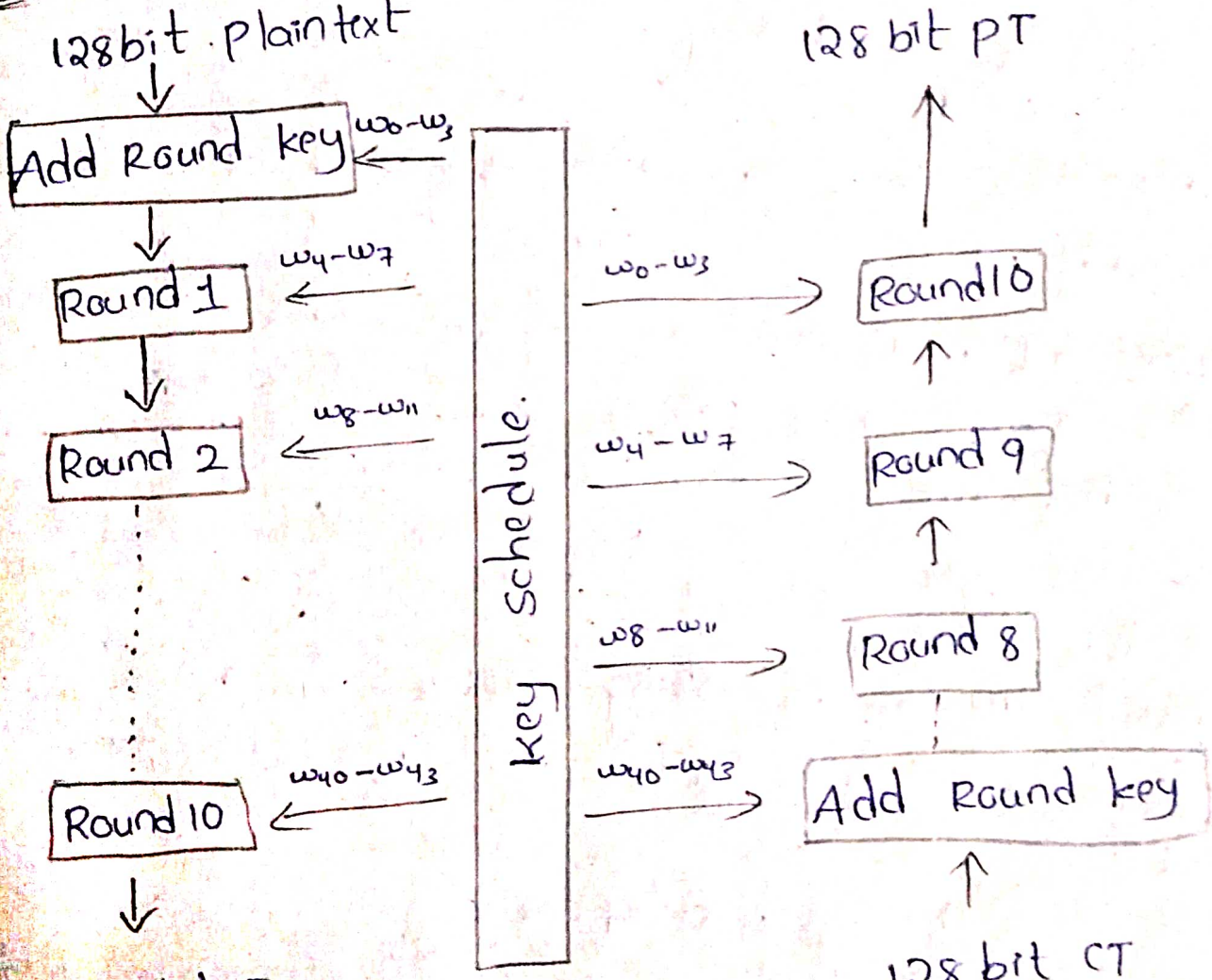
* Actually we have 4th words. They are expanded into 44 words.
* In AES algorithm, there are 10 Rounds.
* Each round = 4 words

∴ 10 Rounds x 4 words = 40 + 4 (for Add Round key)
= 44 words.



4 words ⟶ 44 words.

AES encryption and decryption



* The encryption and decryption starts from Add Round key.

* Each process (encryption & decryption) starts from Round 1 and ends with Round 10.
* Cole have no-of Rounds = 10 (for encryption & decryption)

* In each Round we have four steps.
  1) substitute Bytes
  2) shift Rows (LCS)
  3) Mix columns – Not in Round ⑩
  4) Add Round key

↓

In this XOR operation performed b/w the PT and key.

* 128 bit plain text is sending into the Add Round key along with the words w_0, w_1, w_2 & w_3. Total four words.
* For each and every round we have four words.
* Total 44 words along with add round key.
* Then we will get the 128 bit cipher text.
* This is the process of encryption.
* In decryption process the 128 bit cipher text is sending into add Round key along with the words $w_{40}, w_{41}, w_{42}, w_{43}$.
* This process will continue util the 128 bit plain text is obtained.

1) Blowfish:-

- It is a block cipher Algorithm
- It is a symmetric key cryptography.
- The input size is 64 bits.
- The key size is variable length key i.e., you can take any bit from 32 to 448 bits.
- so, it is more secure.

Properties:-

- It is very fast.
- It takes less memory
- It is simple to understand and implement
- It is more secure

* Blowfish Algorithm has 2 parts:
  1) key generation
  2) Data encryption.

* key generation:-

* keys are stored in an array
  $k_1, k_2, k_3 \ldots k_n \; [1 \le n \le 14]$
* Length of each block is 32 bits
  $(32 \times 14 = 448 \text{ bits})$

2) Initialise an array (P)
   $P_1, P_2, P_3 \cdots P_{18}$
   length of each word is 32 bits.

3) Initialise S-boxes (4) (Substitution boxes).

$S_1 \rightarrow S_0, S_1, \ldots S_{255}$

$S_2 \Rightarrow S_0, S_1, \ldots S_{255}$

$S_3 \Rightarrow$ "

$S_4 \Rightarrow$ "

4) Initialise each element of p-array and S-boxes with hexadecimal values.

5. XOR operations are performed

$P_1 = P_1 \text{ XOR } k_1$

$P_2 = P_2 \text{ XOR } k_2$

$\vdots$

$P_{14} = P_{14} \text{ XOR } k_{14}$ (bcoz only 14 keys)

$P_{15} = P_{15} \text{ XOR } k_{\cancel{1}}$

$\vdots$

$P_{18} = P_{18} \text{ XOR } k_4$

6. Take 64 bit PT (Initially all bits are 0)

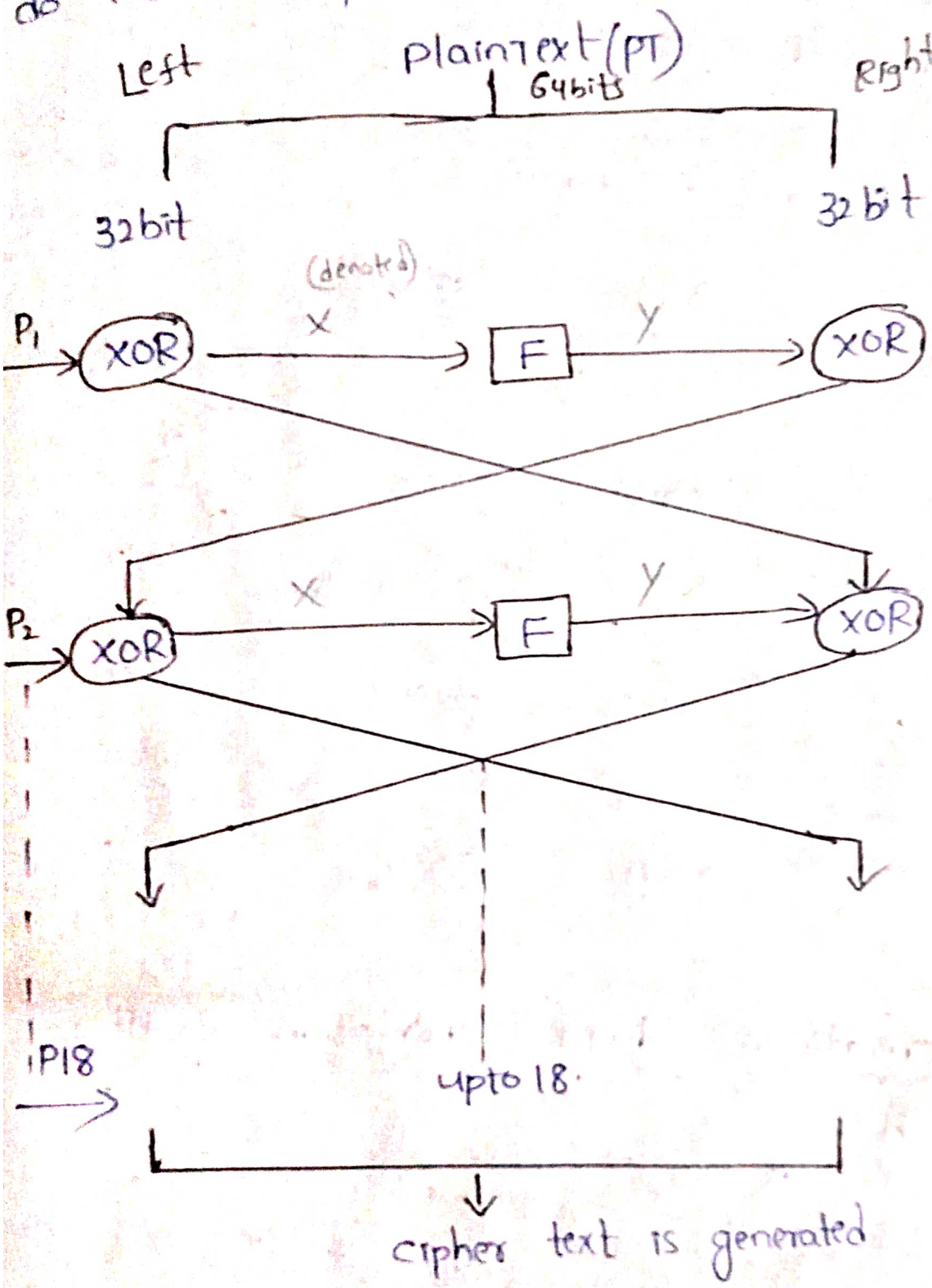
(0,0,0 .... 0)

Subkey is generated.

Data encryption:-

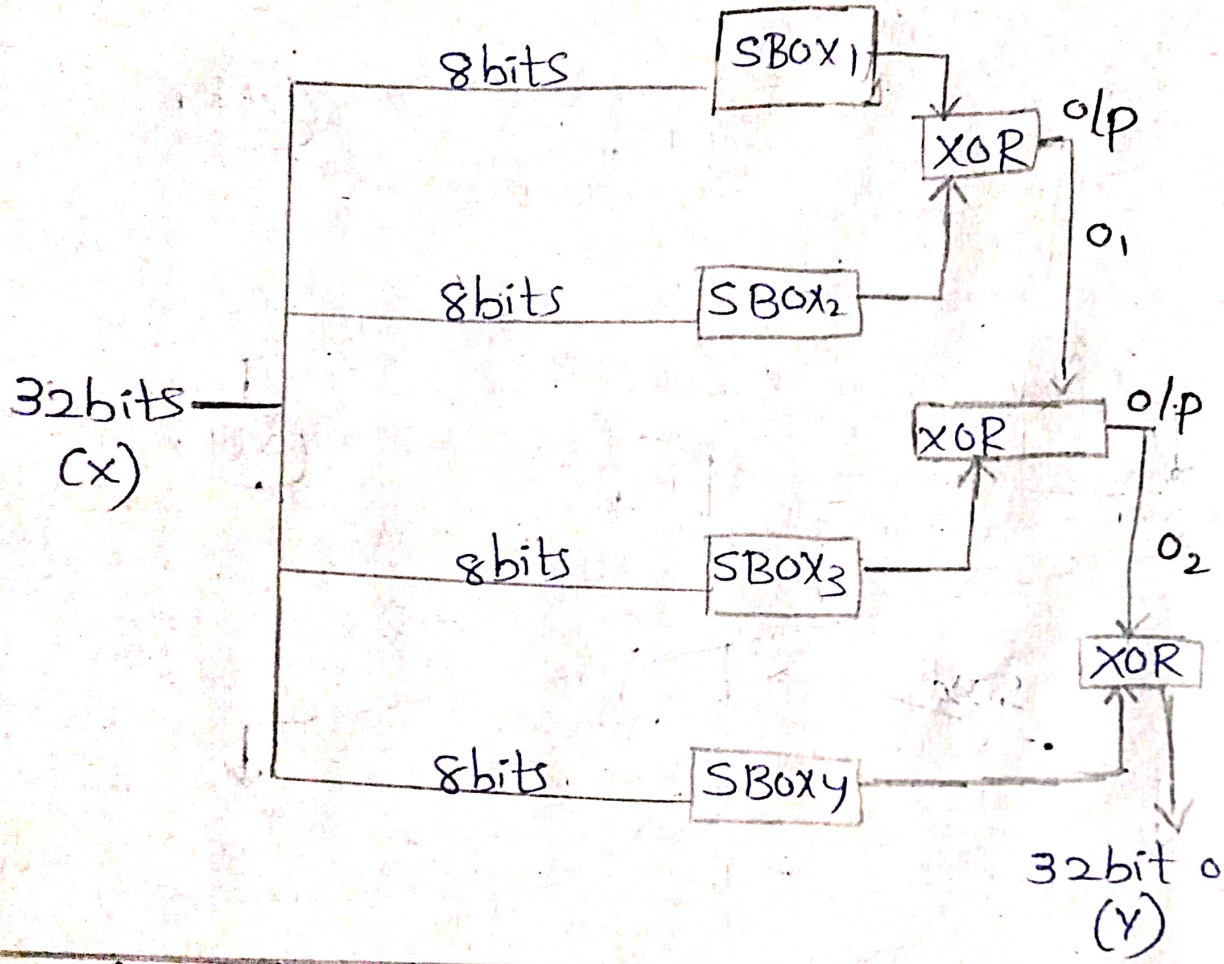
* Divide the plain text into two parts.
* Later the left 32 bit do the XOR operation with P_1. Then we get the output X.
* This o/p will sent into a function and Do function, then we get the output Y.

* The output of 32 bit of right side will do the XOR operation.



* The o/p of XOR operation of left side will send to Right side.
* The o/p of XOR operation of Right side will send to left side.

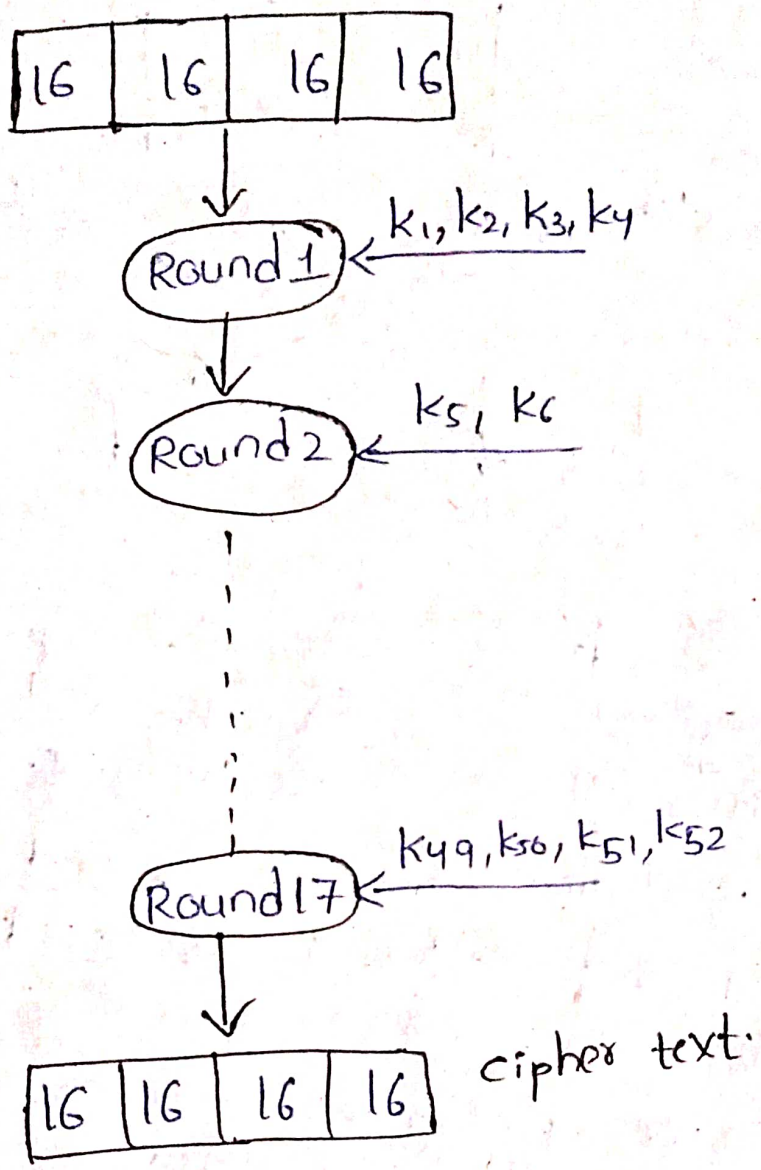
*This process continue upto P18.
*After that we have merge the 32 bits of left side and right side to get the cipher text of 64 bits.
*The diagrametic representation of function is:-



5) International data encryption Algorithm (IDEA):-

*It is a block cipher Algorithm
*symmetric key crypotography.
*It follows feistel ciphers (dividing PT into two parts).
*The input size is 64 bits i.e., (16,16,16,16)
*The key size is 128bits and divided into 52 sub keys.
*We have 17 no. of Rounds.

plain text = 64bits



for odd Rounds only 4 keys

for even Rounds only 2 keys

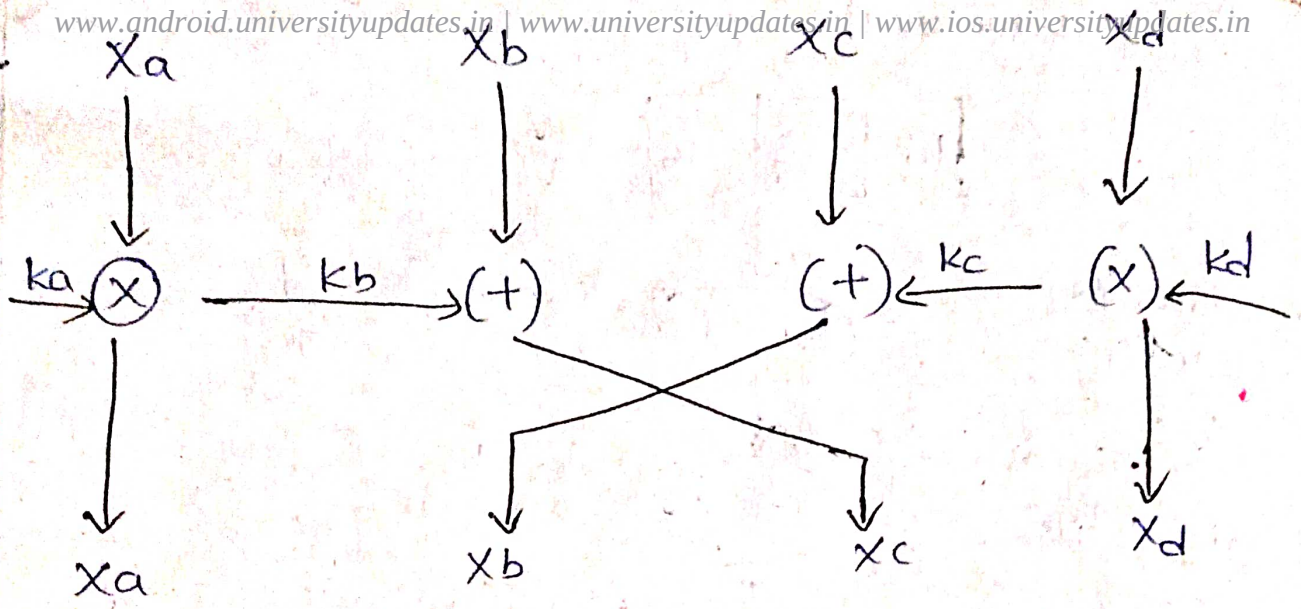
Rounds:-

odd Rounds:-

Input is divided in four parts

keys are four

16	16	16	16 → input
Xa	Xb	Xc	Xd
ka	kb	kc	kd → key



* Here Xa and ka are the inputs and new Xa is generated as o/p. As well as Xd and kd are inputs, the new Xd is generated as o/p.
* But in the cases of Xb and Xc, the o/p's of Xb, kb and Xc, kc are swapped each other. In order to ensure security.
* The outer parts will be same and the inner parts will be swapped.

Even Rounds:-

* We have 4 no. of parts but no. of keys are two only.

Xa Xb Xc Xd

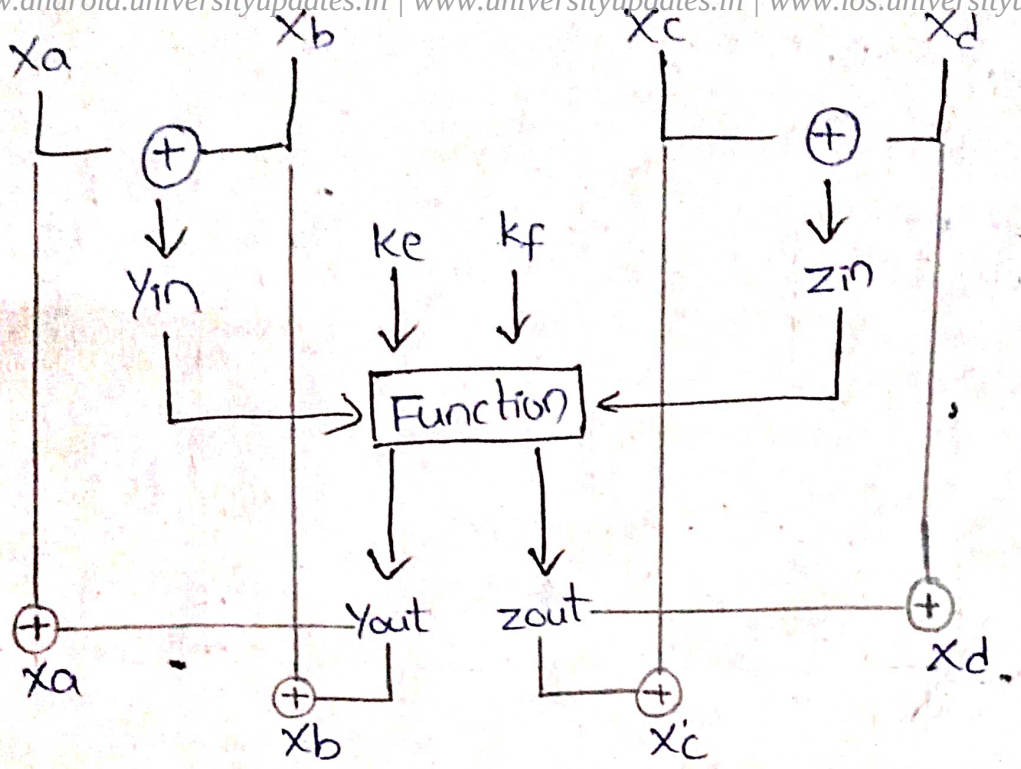
Ke, Kf

i/p = 4 but key = 2 so, Take (2) parameters and perform XOR operation.

$$Y_{in} = X_a \oplus X_b$$

$$Z_{in} = X_c \oplus X_d$$

Now i/p = 2 & key = 2



$X_a = X_a \oplus Y_{out}$

$X_b = X_b \oplus Y_{out}$

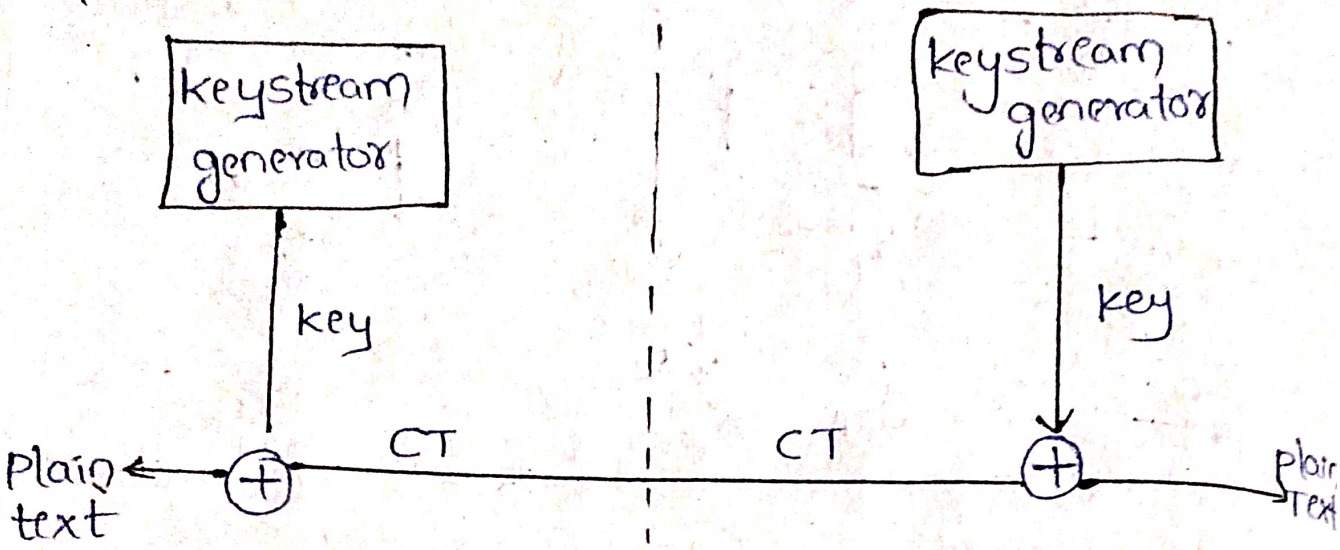
$X_c = X_c \oplus Z_{out}$

$X_d = X_d \oplus Z_{out}$

* X_a and X_b do XOR operation to get Y_{in}. And Y_{in} along ke are the inputs to do the function. After functioning we get the Yout as o/p.
* X_c and X_d do XOR operation to get zin. And zin along with kf are the inputs given to the function to generate the zout as o/p.
* Then Xa and Yout will do XOR operation to get the Xa as o/p.
* X_b and Yout will do XOR operation to get the X_b as o/p.
* X_c and Zout will do XOR operation to get the X_c as o/p.
* X_d and Zout will do XOR operation to get the X_d as o/p.

7] Stream ciphers:-

* plain text is divided into number of stream



* (Bitwise) Keys are generated from keystream generator
* Bitwise XOR operation is performed blw the key and plain text. As the result we get the CT.
* In decryption, with the help of key and CT, we will perform a bitwise XOR operation to get plain text.
* Bitwise XOR means it will consider each bit one by one.

Explanation:-

$m_1 \quad m_2 \quad m_3 \quad \cdots \quad m_i \rightarrow$ plain text

$\oplus \; k_1 \quad k_2 \quad k_3 \quad \cdots \quad k_i \rightarrow$ keys

---

$c_1 \quad c_2 \quad c_3 \quad \cdots \quad c_i \rightarrow$ cipher text

---

(cipher text $\oplus$ key $\Longrightarrow$ plain text) decryption

$C_1 \quad C_2 \quad C_3 \quad \ldots \quad C_l$

$k_1 \quad k_2 \quad k_3 \quad \ldots \quad k_q$

$\oplus$

$P_1 \quad P_2 \quad P_3 \quad \ldots \quad P_l$

Eg:-

PT : 1100

key : 1011

(bitwise XOR :- same = 0, diff = 1)

0111 → CT

CT : 0111

key : 1011

1100 → PT

8) RC4:-

*It is a stream cipher Algorithm.

Procedure:-

1) Uses an array(S) - state vector of length 256 bits (0-255)

2) It has a key encoded with ASCII

3) It has a key array of length 256 (0-255)

*RC4 algorithm has three steps:-

1) key scheduling

2) key stream generation

3) Encryption & decryption

1) Key scheduling:-

*No. of iterations is equal to size of S-array.

initialize j=0

for i=0 to 255 do

$j = [j + S[i] + T(i)] \bmod 256$ (depends on size given the mod value)

swap(S[i], S[j]);

Here,

S[i] → state vector

T[i] → key array (temporary vector).

Example:-

S-array = [0 1 2 3 4 5 6 7] (assume)

key array = [1 2 3 6]

Plain text = [1 2 2 2]

Initialise T-array with key

T = [1 2 3 6 1 2 3 6]

i) j = 0

for i = 0 to 7

$j = [0 + 0 + 1] \bmod 8$

$j = 1 \bmod 8$

j = 1

swap S(0) and S(1)

S = [1 0 2 3 4 5 6 7]

ii) For i = 1

$j = (1 + 0 + 2) \bmod 8$

$j = 3 \bmod 8$

j = 3

swap S(1) & S(3)

S = [1 3 2 0 4 5 6 7]

iii) For i = 2

$j = [3 + 2 + 3] \bmod 8$

$j = 8 \bmod 8$

∴ j = 0

swap S(2) & S(0)

S = [2 3 1 0 4 5 6 7]

* we have to do like this upto 8th iteration.

Stream generation:-

* No. of iterations is equal to size of key (4)

```
i,j = 0;
while (true)
i = (i+1) mod 256;
j = (j+S[i]) mod 256;
swap (S(i), S(j));
t = (S[i] + S[j]) mod 256
k = S[t];
```

* Like this we will get the key array.
  for 1st iteration we get K(0)
  for 2nd iteration we get K(1)
  :
  " " " K(n)

* New key is obtained is used for encryption and decryption.

Encryption and decryption:-

Encryption :- PT XOR New key

* In this first we have to convert the PT and new key into binary.

PT = 1 2 2 2

Then,

PT :- 0001 0010 0010 0010

key :- ( ) ( ) ( ) ( )

CT :-

decryption:- CT XOR New key

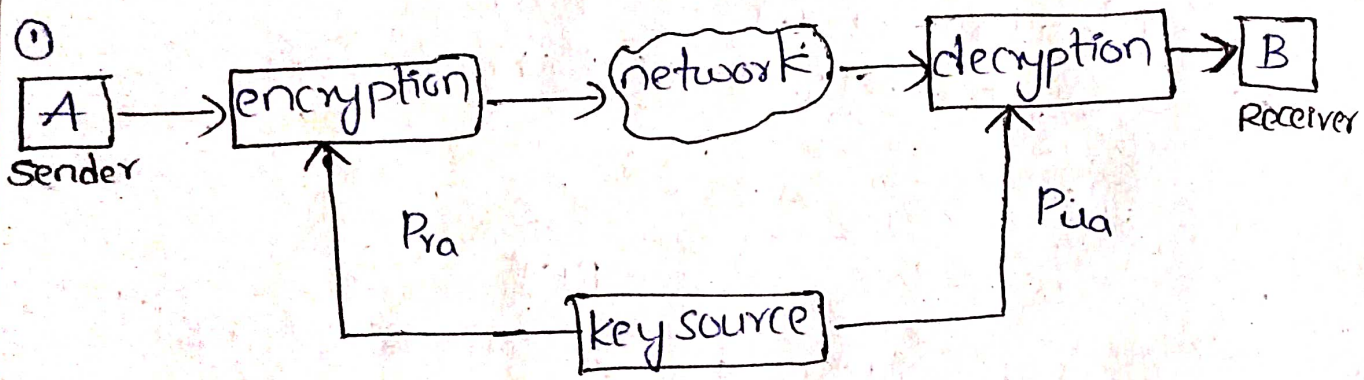
PART-B

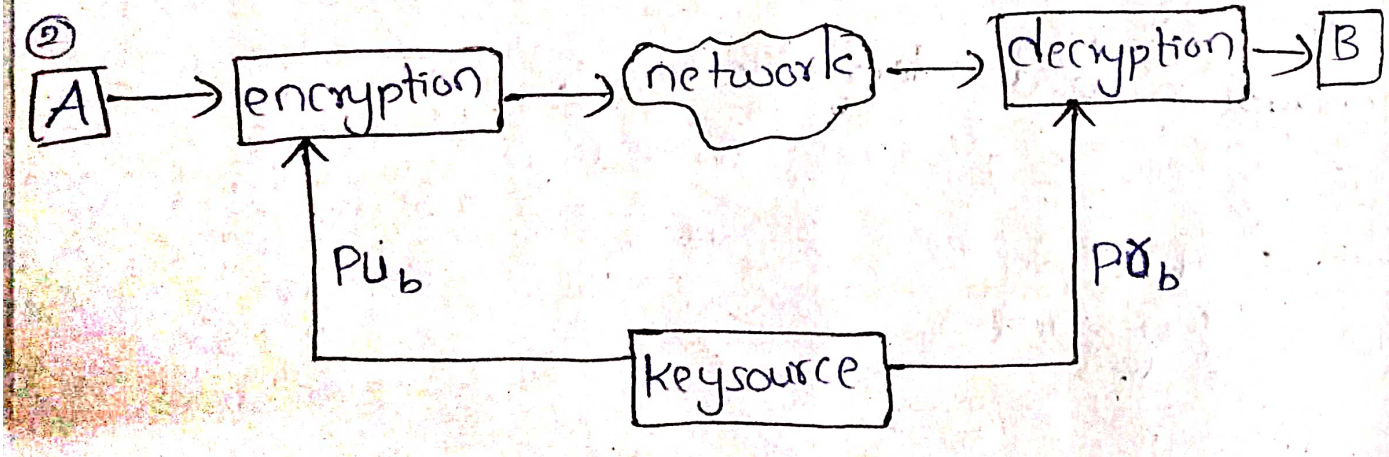
Asymmetric key ciphers:-

Principles of public key cryptosystems:-

* For Asymmetric key cryptography we have different key for encryption and decryption.
* There are two principles:
  1) Authentication
  2) confidentiality.

1) Authentication:-



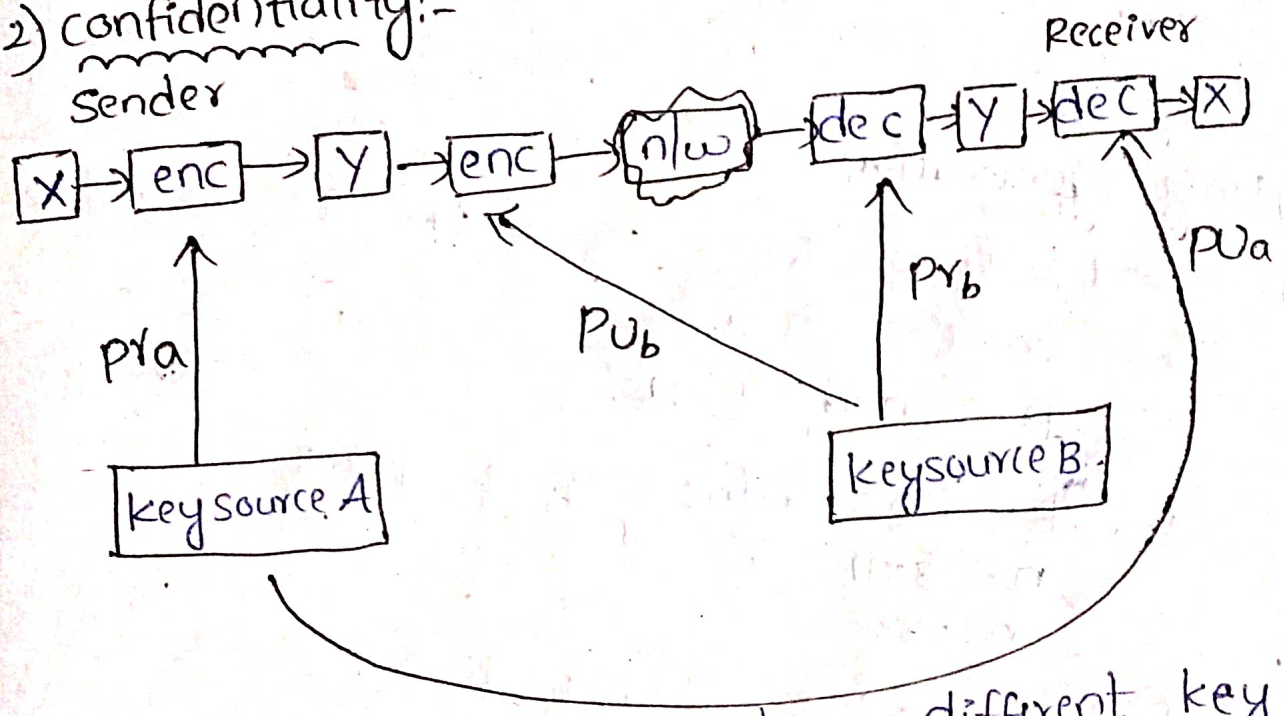


* Here the sender sends the PT and it will encrypted with the help of key which is named as private, taken from key source.
* Then the encrypted msg will enter into a network and come out of this network then to decrypt the msg we are using public A key from

the key source. At last the cipher text is converted into plain text which is received by the receiver.

* Here the key source provided the different keys ie, privateA and publicA. but the same source.
* This is known as Authentication.

2) confidentiality:-



* In confidentiality we have different key sources (key source A & key source B).
* The keysource A will genitrate the public A & privateA keys which is provided to X (sender).
* The keysource B will generate the privateB, & public B keys which is provided to Y (receiver).
* In this, the privateA key of keysourceA is provided to sender and publicA key of keysourceA is provided to receiver.
* The privateB key of keysource B provided to Receiver and publicB key of keysource B is provided to sender.

2) RSA Algorithm:-

* Rivest - Shamir - Adleman (RSA)
* It is a asymmetric key algorithm and block cipher algorithm.
* There are three steps involved:-
  1) key generation
  2) Encryption
  3) Decryption.

1) key generation:-

1. Select two large prime numbers p and q for more security
   $p=3$ and $q=11$

2. calculate the value of $n = p * q$
   $n = 3 \times 11$
   $n = 33$

3. calculate $\phi(n) = (p-1)(q-1)$
   $\phi(n) = (3-1)(11-1)$
   $= 2 \times 10$
   $= 20$

4. choose the value of 'e' such that $(1 < e < \phi(n)$ and $\gcd(\phi(n), e) = 1$
   $(1 < e < 20)$ and $\gcd(20, e) = 1$
   $e = 7$ $\gcd(20, 7) = 1$

5. calculate $d = e^{-1} \bmod \phi(n)$
   $ed = 1 \bmod \phi(n)$
   $ed \bmod \phi(n) = 1$

   $ed \bmod \phi(n) = 1$
   $7 \times d \bmod 20 = 1$
   $7 \times 3 \bmod 20 \Longrightarrow 21 \bmod 20 = 1$ $\therefore$ **d = 3**

6) public key = {e,n} =
= {7,33}

7) private key = {d,n}
{3,33}

2) Encryption:-

The formula of encryption is:

$$C = m^e \bmod n$$

M = no. of digits in PT
C = cipher text
e = encryption
m should $m < n$ i.e., $m < 33$

$\Rightarrow$ Let $m = 31$

$C = (31)^7 \bmod 33$

$= 4$

$\therefore C = 4$

3) Decryption:-

$$m = c^d \bmod n$$

$\Rightarrow m = 4^3 \bmod 33$

$m = 64 \bmod 33$

$m = 31$

---

3) Elgamal cryptography:-

*It follows the principles of Asymmetric key cryptography.

*It has three steps:- 1) key generation
2) Encryption
3) decryption

1) key generation:-

1) select large prime number (P) **P = 11**

2) select a dec-key also called private key **d = 3**

3) select second part of encryption key (e_1) **$e_1 = 2$**

4) calculate third part of encryption key (e_2).

$$e_2 = e_1^d \bmod p$$

$$e_2 = (2)^3 \bmod 11$$

$$= 8 \bmod 11$$

$e_2 = 8$

5) public key = (e_1, e_2, P) and private key = d.

pub key = (2, 8, 11)

2) Encryption:-

1) select random integer (R)

R = 4

2) calculate $c_1 = e_1^R \bmod p$

$$c_1 = 2^4 \bmod 11$$

$$= 16 \bmod 11$$

$c_1 = 5$

3) calculate $c_2 = (PT \times e_2^R) \bmod P$

PT = Assume (7)

$$= (7 \times 8^4) \bmod 11$$

$$= 28672 \bmod 11$$

$c_2 = 6$

4) cipher text = (c_1, c_2)

$c_1, c_2 = (5, 6$

3) Decryption:-

i) $PT = \left[C_2 \times \left((C_1)^D\right)^{-1}\right] \bmod p$

$= 6 \times \left((5)^3\right)^{-1} \bmod 11$

$= 6(5^3)^{-1} \bmod 11$

$= 6(125)^{-1} \bmod 11$

$= 125 \times x \bmod 11 = 1$

If $x = 3$, $125 \times 3 \bmod 11 = 375 \bmod 11$

$= 1$

$\therefore \boxed{x = 3}$

$6 \times 3 \bmod 11 = 18 \bmod 11 = 7$

$\boxed{PT = 7}$

4) DIFFIE-Hellman key Exchange Algorithm:-

* It is a Asymmetric key cryptography.
* It is used to exchange keys blw sender and Receiver.
* It just exchange the key, didn't perform the encryption / decryption Algorithm.

Procedure:-

1) consider a prime number q

let $q = 7$.

2) select α such that $\alpha < q$ and α is primitive root of q.

primitive root?

$\alpha^1 \bmod q$

$\alpha^2 \bmod q$

$\alpha^3 \bmod q$

$\vdots$

$\alpha^{q-1} \bmod q$

Ex:- $\alpha = 3$ and $q = 7$

$3^1 \bmod 7 = 3$

$3^2 \bmod 7 = 2$ $= \{7, 3, 2, 6, 4, 5, 1\}$

$3^3 \bmod 7 = 6$ $\{1, 2, 3, 4, 5, 6, 7\}$

$3^4 \bmod 7 = 4$

$3^5 \bmod 7 = 5$ $\therefore$ 3 is primitive root of 7.

$3^6 \bmod 7 = 1$

3. Assume X_A (private key of A) and $X_A < q$

Calculate $Y_A = \alpha^{X_A} \bmod q$

Y_A = public key of A

Ex:- $q = 7$ and $\alpha = 5$ and let $X_A = 3$ (∴ X = private, Y = public)

$Y_A = (5)^3 \bmod 7 = 125 \bmod 7 = 6$

$\boxed{Y_A = 6}$

4. Assume X_B and $X_B < q$

Calculate $Y_B = \alpha^{X_B} \bmod q$

Let $\boxed{X_B = 4}$

$Y_B = (5)^4 \bmod 7 = 625 \bmod 7 = 2$

$\boxed{Y_B = 2}$

5. Calculate secret keys k_1 and k_2 for exchanging

$k_1 \rightarrow$ person A and

$k_2 \rightarrow$ person B

$k_1 = (Y_B)^{X_A} \bmod q$ $\quad k_2 = (Y_A)^{X_B} \bmod q$.

After calculating, if $k_1 = k_2$ then success

$k_1 = (2)^3 \bmod 7 = 8 \bmod 7 = 1 \longrightarrow \boxed{k_1 = 1}$

$k_2 = (6)^4 \bmod 7 = 1296 \bmod 7 = 1 \longrightarrow \boxed{k_2 = 1}$

$\boxed{k_1 = k_2}$ ∴ success.

keys exchanged successfully.

5) knapsack problem:-

* It was proposed by Hellman.
* It is a Asymmetric key cryptography.

Ex:-

weights = (1,6,8,15 and 24)

In general knapsack, we select weights to acheive a sum

* If we want sum = 30 then,
  we select 1, 6, 8, 15.

Let plain text

PT:- 1 0 0 1 1 1
wi:- 1 6 8 15 24

CT:- 1+15+24 = 40

1 1 0 1 0
1 6 8 15 24

1+6+15 = 22

∴ CT = 40 & 22

* key generation:-

1) public key (Hard knapsack)
2) private key (easy knapsack)

↓ we find private key first.

Ex:-

{1,2,4,9,20,40}

weights are always in increasing order

1. first, find private key (Assume)

D = {1,2,4,10,20,40} — pvt key.

select ② numbers "n" and "m"

m > sum of all no.s in sequence.

Sum = 47 ∴ let m = 110 (Select such that it is more than 70)

n - select so that it has no common factor with m

∴ let $\boxed{n = 31}$

Now (D * n) mod M ∀ elements in D

(1 × 31) mod 110 = 31

(2 × 31) mod 110 = 62

(4 × 31) mod 110 = 14

(10 × 31) mod 110 = 90

(20 × 31) mod 110 = 70

(40 × 31) mod 110 = 30

{31, 62, 14, 90, 70, 30} ⇓ public key.

* Encryption:-

Now, assume PT

let PT = 10010 | 111100 | 101110

divide into 6-6 parts (no. of elements in sequence = 6).

1st part ⇒ 100100 = 1×31 + 0×62 + 0×14 + 1×90 + 0×70 + 0×30

= 31 + 90

= 121.

2nd part ⇒ 111100 = 1×31 + 0×62 + 1×14 + 1×90 + 0×70 + 0×30

= 197.

3rd part ⇒ 101110 = 1×31 + 0×62 + 1×14 + 1×90 + 1×70 + 0×30

= 205

$\therefore CT = [121 \quad 197 \quad 205]$.

*Decryption:-

Calculate $n^{-1} = 31^{-1}$

$31x \bmod 110 = 1$ then we get $x = 71$

$(CT \times x) \bmod m$ from seq $D = \{1, 2, 4, 10, 20, 40\}$ - pvt key

$(121 \times 71) \bmod 110 = 11 = 100100$ $(1+10 = 11)$

$(197 \times 71) \bmod 110 = 17 = 111100$ $(1+2+4+10 = 17)$

$(205 \times 71) \bmod 110 = 35 = 101110$ $(1+4+10+20 = 35)$