

Ques) For the given eqⁿ $y_1 = x_1 \oplus x_2 \oplus x_3$
 $y_2 = x_1$. Find whether the S-box is linear or non linear. Represent the relationship using matrices.

→ The above set of eqⁿ's represent linear S-box.

$$y_1 = a_{1,1} x_1 \oplus a_{1,2} x_2 \oplus a_{1,3} x_3$$

$$y_2 = a_{2,1} x_1 \oplus 1 \quad 0 \quad 0 \quad \oplus \quad 0$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 00 & 1 & 01 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

Ques) In an S-box with 3 inputs and 2 output we have the eqⁿ's as

$$y_1 = x_1 x_2$$

$$y_2 = x_1 + x_2 x_3$$

Identify the type of S-box & show justification

→ The above set of eqⁿ represent non-linear S-box because it does not have any relationship between input and outputs.

Ques) S box may or may not be invertible

Justify your answer

For an invertible S-box the number of input bits should be equal with number of output bits.

Following shows an example:

Eg. I/O relation of 2×2 S-box is shown in the following table. Show Design S-box table.

leftmost	0	1	\rightarrow Rightmost
0	01	11	
1	10	00	

Soln :-

{ For the above table, the leftmost bit of the given input defines the row and the remaining rightmost bits define the column }

The above table can be represented as :

I/P \Rightarrow 00 01 10 11

O/P \Rightarrow 01 11 10 00

swap
content
and
indices

1	3	2	0
0	1	2	3

3	0	2	1
---	---	---	---

I/P \Rightarrow 00 01 10 11
 O/P \Rightarrow 11 01 00 10 01

Invertible S box :

0	11	00
1	10	01

Ques] Find the invertible S box for given values

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

I/P \Rightarrow 000 = 001 010 011 100 101 110 111
 O/P \Rightarrow 011 101 : 111 100 000 010 001 110

(3) 5 7 4 0 2 1 6

0 2 3 4 5 6 7

6 5 3 3 1 7 2

4 7 4

3 5 7 - 4x 0 2 1 6

$x \oplus y \leftrightarrow f \oplus x \leftarrow$

- Ques) Write short note on Block method
- ① Transposition P-box / D-box
 - ② S box \rightarrow linear \rightarrow non linear
 - ③ Ex-OR operation [Other component]

iii)

	01	10	00	11	00	11	00	11
10	0	00	01	10	11	00	11	00
0	110	001	010	011	000	110	011	101
1	100	111	101	0000	010	101	111	010

ii)

0	00	10
1	11	01

$GF(2^n)$: Galois Finite field

↳ Finite field

→ Finite no. of elements

~~Pg 160~~ It means finite range

For eg. no. of bits = 64 Range : 2^{64}

X-OR Operation uses $GF(2^n)$

Properties :

- ① Closure
- ② Associativity
- ③ Commutativity
- ④ Existence of identity
- ⑤ Existence of inverse

To avoid carry & borrow use

X-OR

$$④ \Rightarrow x \oplus (00 \dots 0) = x$$

$$⑤ \Rightarrow x \oplus x = (0000 \dots 0)$$

$$③ \Rightarrow x \oplus y \leftrightarrow y \oplus x$$

Other Component

- X-OR → Circular shift (L & R)
- Complement → Swap
- Inverse → Split & combine [62]

Invertibility [Circular Shift]

A circular left shift opr. is inverse of circular right shift.

Ques) a) Show the result of 3 bit circular ~~left~~^{right} shift on word $(10011011)_2$

$$\begin{array}{l} \rightarrow \\ \text{Result: } \begin{array}{c} 10011011 \\ \cancel{100} \\ 00000000 \\ \text{Original: } (10011011) \\ \text{Shifted: } (0001101101 \\ \quad \quad \quad 01110011) \end{array} \end{array}$$

b) Show the result of 3 bit circular ~~left~~^{right} shift on word resulting from Part (a)

$$\begin{array}{l} \rightarrow \\ \text{Result: } \begin{array}{c} 1011001101 \\ 0100111011 \\ 1101101110 \\ 11011100 \\ \text{Original: } (1011001101 \\ \quad \quad \quad 0100111011 \\ \quad \quad \quad 1101101110 \\ \quad \quad \quad 11011100) \end{array} \end{array}$$

c) Compare the result of Part b with original word in Part a



(Q8.1) ~~gates~~ solution

90-X-4

ques 2

Date _____
Page _____

Sundays & Thursdays

Answer

- Que 16] a) Swap the word $(10011011)_2$
b) Swap the word resulting from Part a
c) Compare the result of ~~the~~ Part a and Part b to show that swapping is a self invertible operation

Que 17] Find result of following operation

a) $(01001101) \oplus (01001101)$

b) $(01001101) \oplus (101101010)$

c) $(01001101) \oplus (00000000)$

d) $(01001101) \oplus (11111111)$

~~01001101~~

→ a) 01001101

$\oplus \underline{01001101}$
 000000000

b) 001001101

$\oplus \underline{1101110010}$

0111101111011

001110111

c) 01001101

$\oplus \underline{000000000}$

Ques) A 6×2 S-box is X-Orred the odd no. of bit to get the left bit of O/p. Similarly perform X-OR operation on the even numbered bits to get the right bit of the O/p. Perform the mentioned on the following i/p string.

① 110010

② 101101

$$\rightarrow \begin{array}{ccccccc} 5 & 4 & 3 & 2 & 1 & 0 & \text{odd bits} \\ 1 & 1 & 0 & 0 & 1 & 0 & \\ \end{array}$$

$$= 1 \oplus 0 \oplus 1 = 0$$

even bits

$$1 \oplus 1 \oplus 0 = 1$$

$$0 \oplus 0$$

01

② 101101

$$\text{odd bits, } 1 \oplus 1 \oplus 0 = 1$$

$$\text{even bits, } 0 \oplus 1 \oplus 1 = 0$$

Ques) The leftmost bit of 4×3 S-box rotates the other 3 bits. If the leftmost bit is 0 then other bits are rotated to right 1 bit. Similarly, if leftmost bit is 1 then the 3 other bits are rotated to the left 1 bit perform the above and generate the output for given input

① 1011

② 0110

010

0

Product Cipher

- It is generally implemented in modern block cipher.
- It is the product of substitution unit and transposition unit using other component.
- Benefit of creating Product cipher is to apply on Modern Block cipher.

Properties :

- ① Diffusion \Rightarrow Hides the relationship between CT & PT.
- ② Confusion \Rightarrow Confuse the relationship between key & CT.

NOTES,

- Product Cipher is a combination of substitution, transposition, other component.

- It was designed by Shannon to generate complex Modern Block cipher
- To build / design product cipher Shannon uses two imp properties.

① Diffusion : It hides the relationship between CT & PT

② Confusion : This property confuses the attacker to find relation between key & CT.

The diffusion and confusion properties are applied based on the number of rounds, i.e. number of iteration.

n -bit PT \Rightarrow n -bit key } depends
 ↓ on
 n -bit CT rounds.

MT : Middle text

PT It is generated at intermediate
MT between PT & CT.
CT

Product Cipher consist 3 phases :

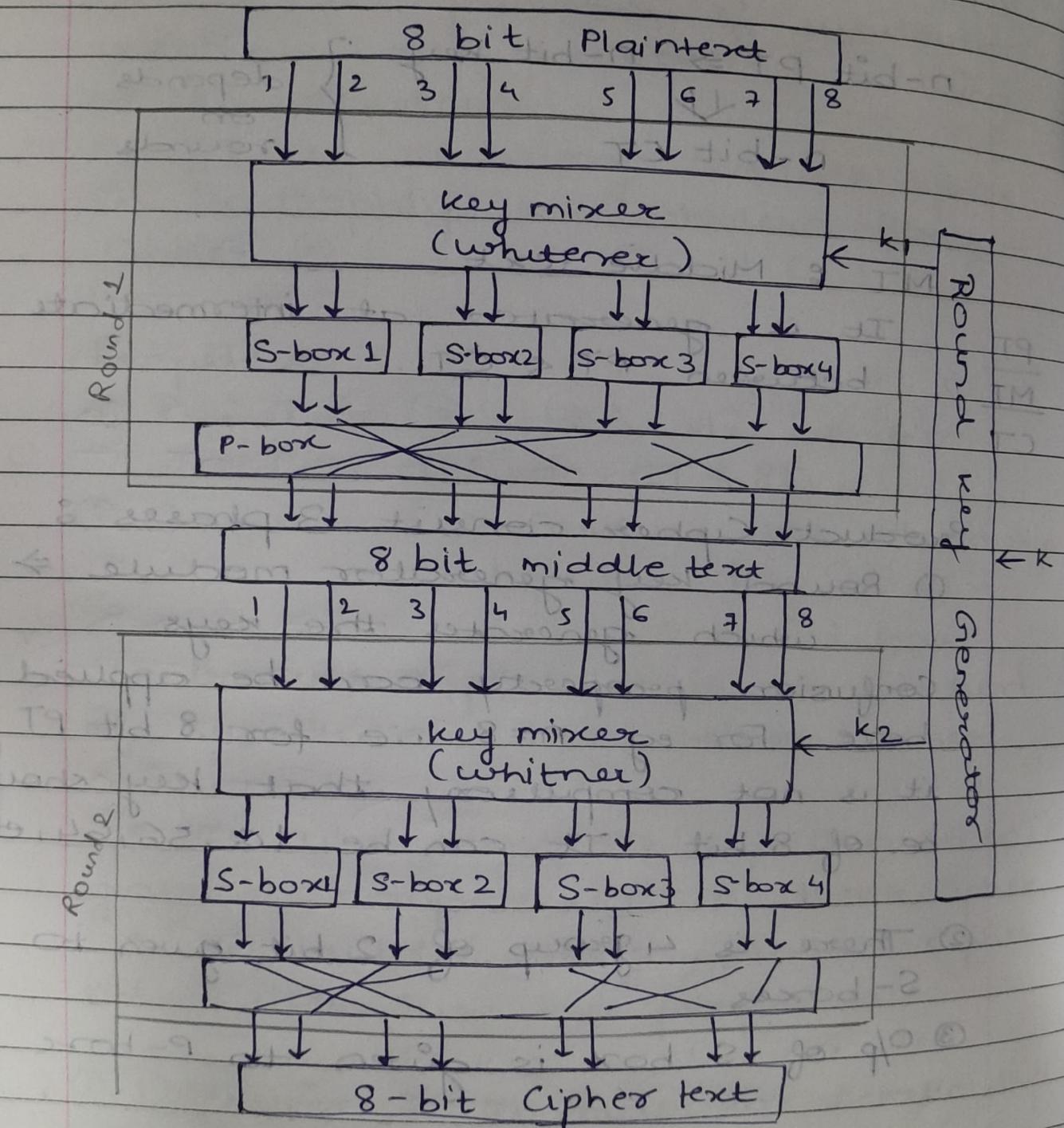
① Round key generator module \Rightarrow which generates the keys

Confusion property can be applied here. For eg. $n=8$, i.e. for 8 bit PT it is not compulsory that key should be of 8 bit. It can be 16, 32, 64, etc.

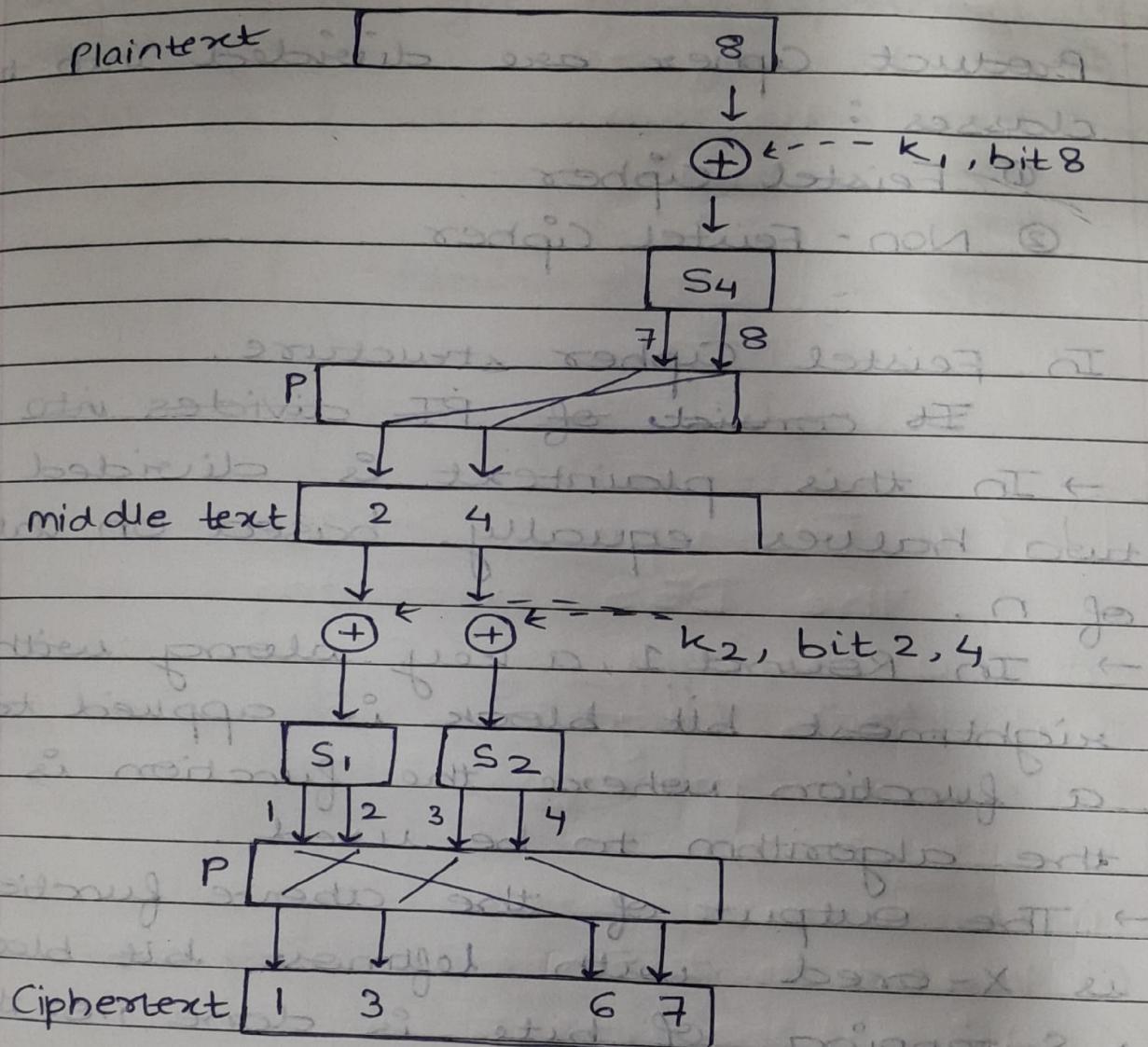
② There is 4 group of 2 bit given to S-boxes.

③ O/p of S box is given to P-box

Following figure shows product cipher of 2 rounds.



Following figures shows diffusion and confusion in block cipher



In the given figure, the 8th bit of PT is X-ored with 8th bit of key k_1 . The X-ored value is given as input to 1×2 box, so we get 2 output 7 & 8 bit. The output of S box is given as input to P box. It gives middle text 2, 4. It is O/p of Round 1.

Summary : The 8th bit of PT affects 1, 3, 6, 7 bit of CT

Classes of Product Cipher :

Product Cipher are divided into two classes :

① Feistel cipher

② Non-Feistel cipher

In Feistel Cipher structure,

It consists of PT divides into

→ In this plaintext is divided into two halves equally based on value of n .

→ In Round 1, a key along with rightmost bit block is applied to a function where the function is the algorithm to be used.

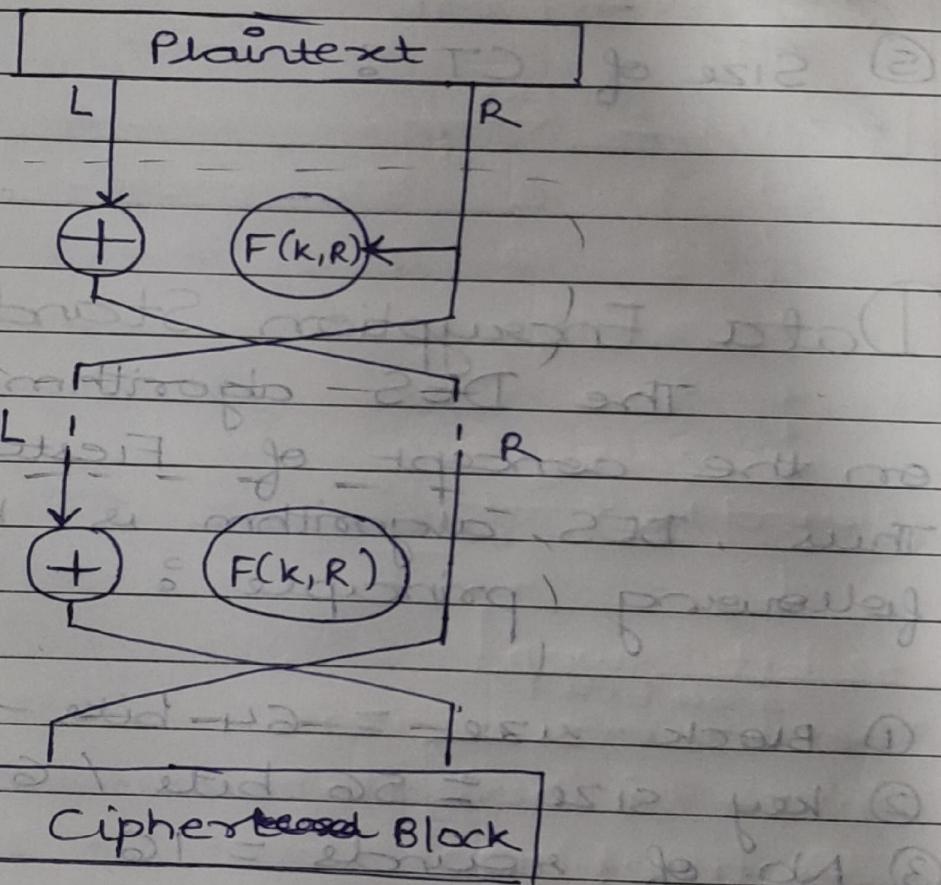
→ The output of the above function is X-ored with leftmost bit block.

→ Swapping of bits is done i.e. the leftmost bit ^{block} now becomes right most bit block and similarly the right most bit block becomes left most bit block.

Thus, round 1 gets completed in above 4 steps. Thus the step along with round key value becomes the input for next round.

Thus after the number of rounds completed depending on the type of algorithm we get final ciphertext block.

Following figure shows the above process.



Block Cipher designed principle are based on the following:

- ① Block Size : length of the PT is equal to length of CT (depends on Algo)
- ② key size : The given key size is known as master key size, depends on algorithm.

③ Number of rounds: Conversion from PT to CT

④ Number of subkeys count: We need to generate subkey from master key

⑤ Size of CT:

8-10M

Data Encryption Standard Algo:

The DES algorithm is based on the concept of Feistel structure. Thus, DES algorithm is based on following principle:

① Block size = 64 bits

② Key size = 56 bits / 64 bits

③ No. of rounds = 16

④ No. of subkey = 16 where each subkey size = 48 bits

⑤ Size of Ciphertext = 64 bits

- Initial permutation & reordering of bits
- 64 bits is given to round 1

Round 1 have ip2 i/p : 512 bits

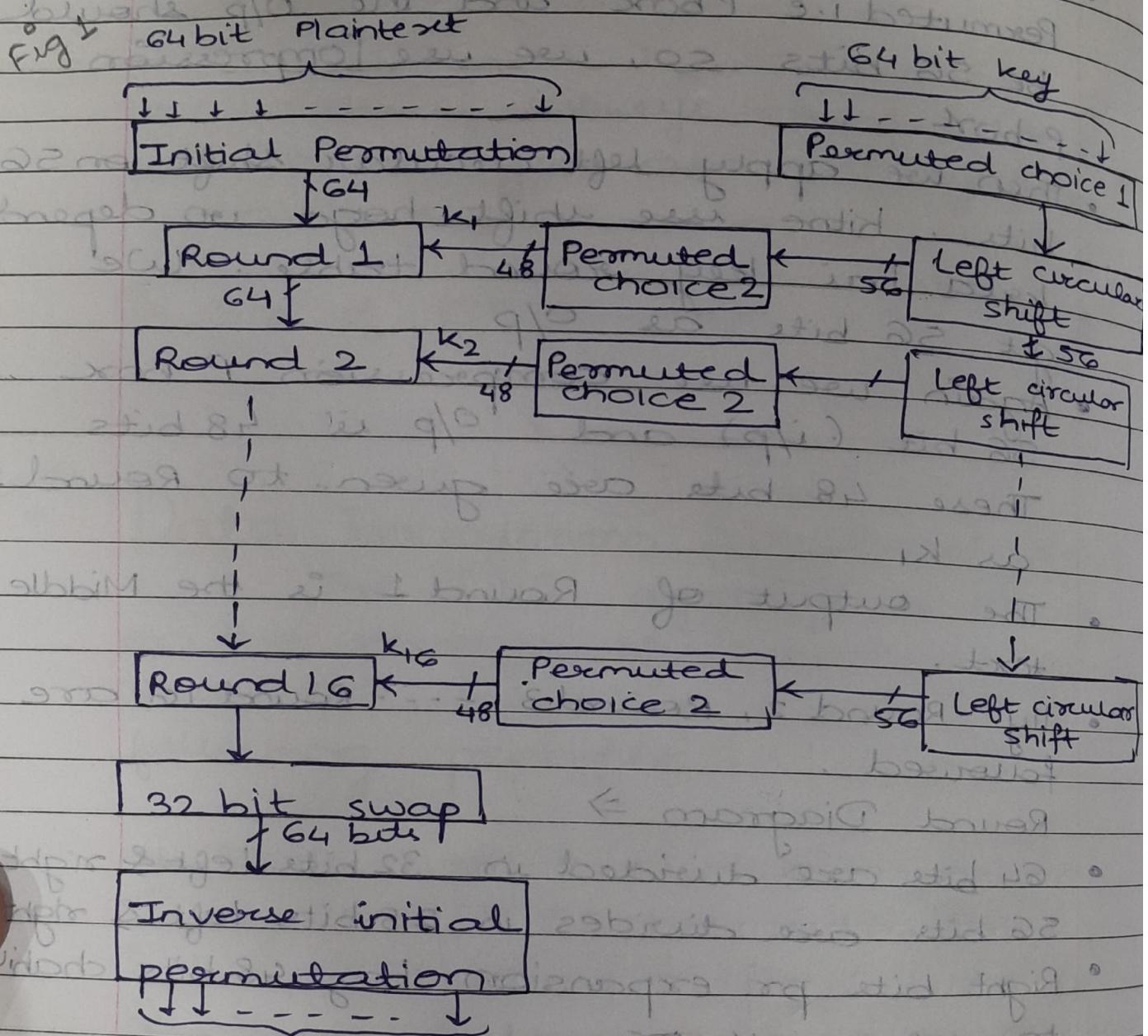
- 64 bit reordered: PT
- K_1

- We have 64 bit key which is given to Permutated i.e P box and the O/p should be 56 bits so, we use compression P-box
- Then we apply left circular shift on 56 bits. Kitne shift hoga to depend keta hai key round table. We get 56 bits as O/p.
- Again we use compression P box 56 bit (i/p) and O/p is 48 bits These 48 bits are given to Round 1 as k_1
- The output of Round 1 is the Middle-text.
- By Round 2, Round 3, ... Round 16 are followed.

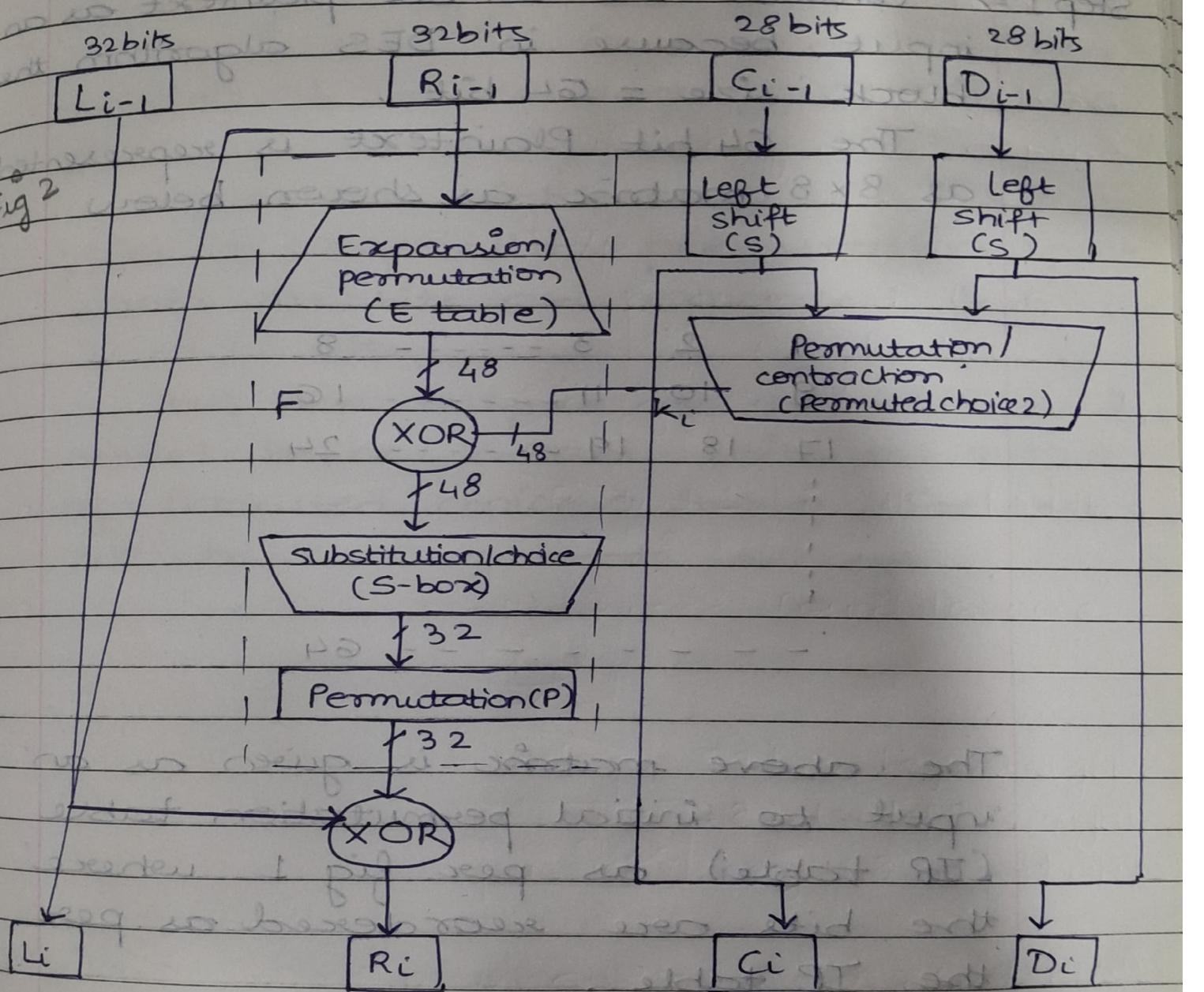
Round Diagram \Rightarrow

- 64 bits are divided in 32 bits left & right.
- 56 bits are divides in 28 bits left & right.
- Right bits for expansion bcz 48 bits chahiye to perform X-OR.
- X-OR ka o/p is given to S box (32 bits)
- Then permutation i.e reordering (32 bits)
- Then XOR and swapping.
- In case of key, we perform left shift bcz we are doing encryption and do right shift in case of decryption

Block diagram of DES algorithm.



Following fig shows DES single round function process:



DES Encryption process:

Step 1 \Rightarrow Consider the 64 bit plaintext as an input because in DES algorithm the block size = 64 bit.

The 64 bit Plaintext is represented as 8×8 matrix as shown below

1	2	3	-----	8
9	10	11	-----	16
17	18	19	-----	24
1	1	1	-----	8H
1	1	1	-----	initialization (Round-0)
-----	-----	-----	-----	64
-----	-----	-----	-----	(9) initialisation

The above matrix is given as an input to initial permutation table (IP table) as per fig 1 where the bits are reordered as per the IP table.

According to the IP table, the first bit position of the IP table represented by 58th bit of PT.

11y second bit position of IP table is reordered with 50th bit of PT and so on.

In this way 64 bits are reordered

After IP it is given to the round 1, according to fig 2.

Thus, the PT is now divided into two equal halves each of 32 bit block size.

Now, the right half 32 bite is given to expansion permutation table where the bits are expanded from 32 to 48 bits.

Thus the output of the EP table is needed to be X-ored with the output of Permutated choice 2 table. For this, we have to generate subkeys.

* Blowfish Algorithm :

- It is an symmetric key algorithm
- The size of the plaintext is 64 bits
It uses the concept of Feistel structure
- It has variable key size as compare to DES (56 bits) (32 to 448 bit)
- Blowfish was developed by Bruce Schneier in 1993
- Total no. of rounds = 16 to convert the plaintext to ciphertext.
- It is an alternative DES algorithm
- It is divided into 2 parts :
 - ① Generation of subkeys
 - ② Data Encryption / Decryption process

① GENERATION OF SUBKEYS :

→ As the key size is variable that is of the size (32 bits to 448 bit) it is divided into 14 subkeys i.e. $k_1, k_2 \dots k_{14}$ ($1 \leq n \leq 14$) each of 32 bit size.

→ Blowfish introduces the concept of p-arrays and S-boxes. Thus there are total 18 p-arrays numbered from p_1 to p_{18} or P_0 to P_{17} followed by 43 boxes of size 8×32 .

→ Each S-box has total 256 entries that is (0-255) which is defined as ~~skew~~

$S_{1,0} \quad S_{1,1} \dots \quad S_{1,255}$
 $S_{2,0} \quad S_{2,1} \dots \quad S_{2,255}$

0th entry ↪

↪ 255th entry

→ Each S-box has 8 bit input & 32 bit output thus in 8 bit input 1st, 7th, 8th bit represents the row number.

→ whereas (2,3,4,5,6) bit represents the column number. The intersection of these gives skew values.

→ Initialize the

In Fig 2,

32 bits are given as input
i.e. 8 bit to each S box (4-S boxes)

→ The function F in Fig A is as follows
as shown in Fig B.

- ① Divide the 32 bit left block into four 8 bit subblock i.e. 8 bit S-box named as a, b, c, d.
- ② compute $F[a, b, c, d] = ((S_{1,a} + S_{2,b}) \text{ x-or } S_{3,c}) + S_{4,d}$

Assume, $a = 10, b = 95, c = 37, d = 191$

then $F(a, b, c, d) = ((S_{1,10} + S_{2,95}) \text{ x-or } S_{3,37}) + S_{4,191}$

$S_{1,10}$: 1st S box ki 10th value

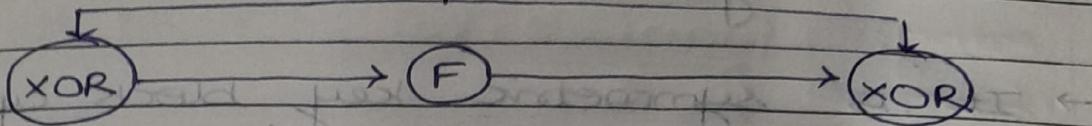
$S_{2,95}$: 2nd S box ki 95th entry

[Each S-box has total 256 entries]

Following figure shows the Blowfish Decryption process :

Atul kate ch3 Pg 123-130

ciphertext



Ques) Justify how to generate the subkey values using Blowfish Encryption algorithm

subkey : P array] and S array

Initial values $\rightarrow P_1 - P_{18}$ } 32 bit
 $S_1 - S_4$ } fp of π

Refer ch 3 atul kahte pt. cd, e

* AES algorithm

- It is symmetric key block cipher method
- Developed by NIST (National Institute of Standards and Technology) in Dec 2001
- It's a non-Fiestel structure
- Block size of PT and CT is 128 bits
- No. of rounds : 10/12/14 bits

Fig 7.1 Fig 1. General design of AES encryption cipher

→ 128 bit PT is given to pre-round transformation [also called preprocessing where add round key operation is performed. Here $\underline{128 \text{ PT}}$ X-OR $\underline{k_0}$
 Round key = 128.

NR = Number of rounds

Imp

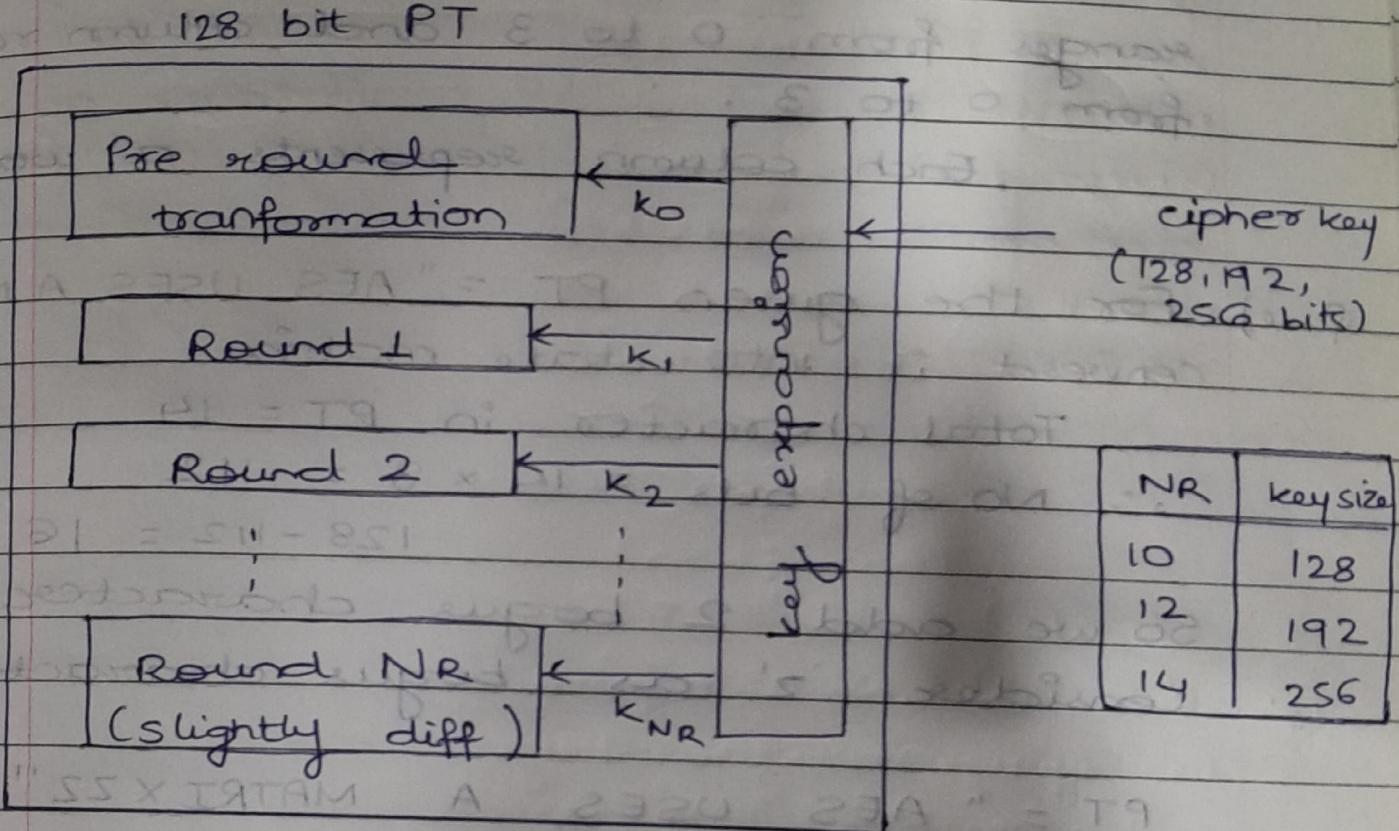
→ The number of round key generated by key expansion algo. is always one more than the number of rounds.

In other words, we have

$$\text{No. of round keys} = \text{Nr} + 1$$

We refer round keys as $k_0, k_1, k_2, \dots, k_{\text{Nr}}$

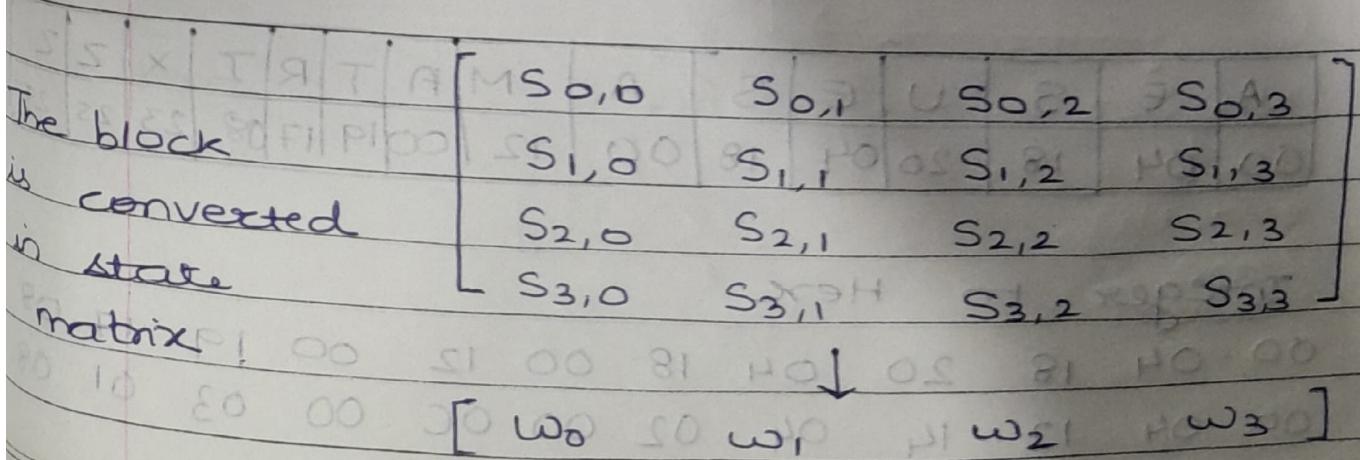
Fig. General design of AES encryption



Data units used in AES

There are 5 data units at work

- ⑤ bit
- ① byte = 8 bits
- ② word = 4 bytes
- ③ block = 16 bytes = 128 bits
- ④ State matrix = 4 words



Define state matrix (2m)

It is a 4×4 matrix where row ranges from 0 to 3 and column ranges from 0 to 3.

Each column represents a word

Ques) For the given PT = "AES USES A MATRIX"
convert it into state matrix.

Total character in PT = 14

$$\therefore \text{No of bits} = 14 \times 8 = 112$$

$$128 - 112 = 16$$

So we add 2 bogus characters

Consider 'z' as a bogus character

PT = "AES USES A MATRIXZZ"

$$16 \times 8 = 128 \text{ bits}$$

char to int values

hexdec

ASCII → 00 to 25

hexdec

A	E	S	U	E	S	A	M	A	T	R	I	X	Z	Z
00	04	18	20	04	18	00	12	00	19	17	08	23	25	25

Integer to Hex

00 04 18 20 04 18 00 12 00 19 17 08
00 04 12 14 04 02 00 0C 00 03 01 08

23 25 25
07 09 09

state matrix :

00	04	00	08
04	02	03	07
12	00	01	09
14	06	08	09

que] Let the PT = "NET THE R THIS IS AN EXERCISE". Convert in state matrix

$$16 \times 8 = 128 \text{ bits}$$

TH I S I S A N E X E R C I S E

19 7 8 18 8 18 0

is writing on the information. The words

is writing

as per

original

bit

1 result



bit



out

bit

original

bit

out

bit

Unit : 3 Assymmetric key

In symmetric key cryptography, the secret key must be shared between the two users whereas in case of asymmetric key cryptography, the key is generated and kept as a personal secret.

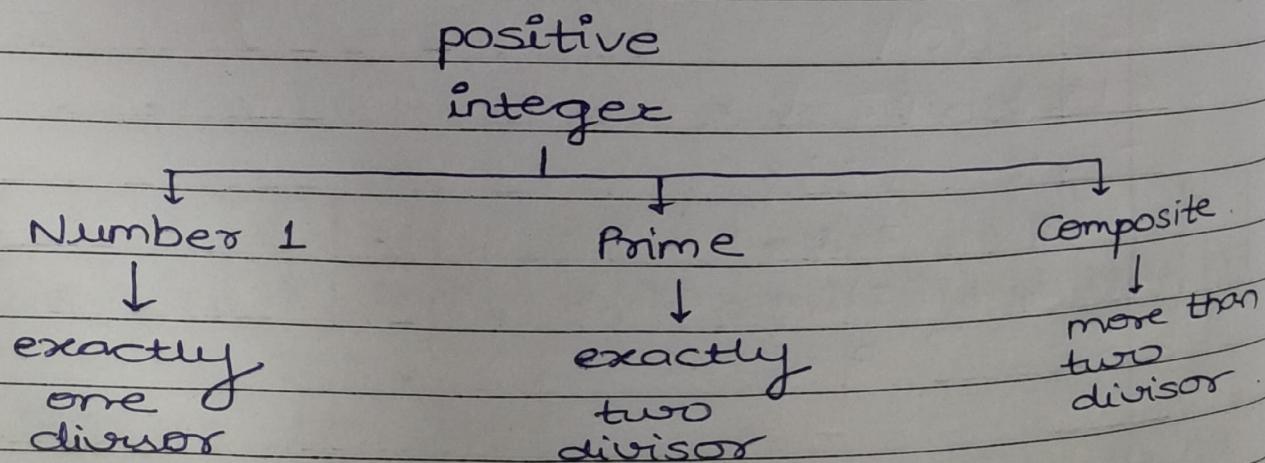
Assymmetric

public key
(encryption) private key
(decryption)

It is the responsibility of the receiver to generate the keys.

The public key is distributed by the receiver and private key is kept by the receiver.

Chapter 9 : Mathematics for assymmetric key



To check if number is prime?

→ check number is divisible by all primes less than \sqrt{n}

For eg 97

$\sqrt{97} \Rightarrow 9$ (we will consider only floor value)
 $\{ 2, 3, 5, 7 \}$ check these.

→ Euler's Phi-Function

Denoted by $\phi(n)$ which is also called as Euler's totient function plays a very important role in cryptography

The function finds the number of integers that are both smaller than n and relatively prime to n .

The following helps to find the value of $\phi(n)$:

- ① $\phi(1) = 0$
- ② $\phi(p) = p - 1$ if p is a prime
- ③ $\phi(m \times n) = \phi(m) \times \phi(n)$ if m and n are relatively prime
- ④ $\phi(p^e) = p^e - p^{e-1}$ if p is a prime

We can combine the above four rules to find values of $\phi(n)$. For example, if n can be factored as

$$n = p_1^{e_1} \times p_2^{e_2} \times \dots \times p_k^{e_k}$$

then we combine the third and the forth rule to find

$$\phi(n) = (p_1^{e_1} - p_1^{e_1-1}) \times (p_2^{e_2} - p_2^{e_2-1}) \times \dots \times (p_k^{e_k} - p_k^{e_k-1})$$

$$\textcircled{1} \quad \phi(13) = 13 - 1 = 12 \quad [\text{Using } 2^{\text{nd}} \text{ property}]$$

$$\textcircled{2} \quad \phi(10) = \phi(2 \times 5)$$

By using property $\textcircled{3}$ where m and n are relative coprime

$$\phi(2) \times \phi(5) = 1 \times 4 = 4$$

$$\textcircled{3} \quad \phi(240)$$

$$240 = 2^4 \times 3 \times 5$$

Using property $\textcircled{4}$

$$(2^4 - 2^3) \cdot \phi(3) \times \phi(5)$$

$$16 - 8 \quad \phi(8) \times \phi(3) \times \phi(5)$$

$$(2^3 - 2^2) \times \phi(3) \times (5)$$

$$(2^2 - 2^1) \times \phi(3) \times \phi(5)$$

$$\phi(1) \times \phi(3) \times \phi(5)$$

$$1 \times 2 \times 4 = 8$$

que) Find values of $\phi(529)$, $\phi(532)$, $\phi(580)$, $\phi(500)$, $\phi(5101)$

① $\phi(529)$

② $\phi(532) = 52 - 2^5$

③ $\phi(580) =$

$$\begin{aligned}\rightarrow \phi(529) &= (23^2) \times (23 - 1) \\ &= 23^2 - 23^1 \times (23 - 1) \\ &= 529 - 23 \times (8 - 1) \\ &= 506\end{aligned}$$

2015
2016

memorT still strong

in play training has a good fit
with practice has much breakdown
memorT is memory and working set
is used

is activeV test

is good to test episodic memory with
activeV and it is hard to remember
and it is difficult to remember
1-9 from 0-9. $\{9 from 1 = 1-9\}$

is activeV because

not remember memory traces for
1-9 if test episodic V. so no inhibition
and repetition in is hard, may be

Que) ① $\phi(29) \Rightarrow 29-1 = 28$

② $\phi(32) \Rightarrow \phi(2^5)$

$$= 2^5 - 2^4 = 32 - 16 \\ = 16$$

③ $\phi(80) \Rightarrow 2^4 \times 5$

$$(2^4 - 2^3) \times \phi(5)$$

$$(16 - 8) \times 4$$

$$8 \times 4$$

$$= 32$$

2	80
2	40
2	20
2	10
	5

④ $\phi(100) \Rightarrow 2^2 \times 5^2$

2	100
2	50
	25

* Fermat's Little Theorem

It plays a very important role in number theory and cryptography. We introduce two versions of theorem here :

First Version :

The first version says that if p is a prime and a is an integer such that p does not divide a , then

$$a^{p-1} \equiv 1 \pmod{p}$$

$$\text{i.e. } a^{p-1} \pmod{p} = 1$$

Second Version :

The second version removes the condition on a . It says that if p is a prime, and a is an integer then

format me ek to power wala p chota
nara change p se or equal but in que
it is greater ($12 > 11$) so we apply rules

$$a^p = a \text{ mod } p$$

(que) $6^{10} \text{ mod } 11$

$$6^{11-1} \text{ mod } 11 = 1$$

(2) $3^{12} \text{ mod } 11 \quad (3^{11} \times 3) \text{ mod } 11$

$$(a \times b) \text{ mod } n = ((a \text{ mod } n) \times (b \text{ mod } n)) \text{ mod } n$$

$$=((3^{11} \text{ mod } 11) \times (3 \text{ mod } 11)) \text{ mod } 11$$

$$= (3 \times 3) \text{ mod } 11$$

↳ using second version

$$= 9 \text{ mod } 11$$

$$= 9$$

* Application of Fermat's little theorem

It is helpful for the following :

① To find a solution for modular exponentiation if modulus is a prime number

② Used to find the solution for calculating of multiplicative inverses, if modulus is a prime number.

following equation shows the concept to find the multiplicative inverse as given below :

If p is a prime and a is an integer such that p does not divide a ($p \nmid a$), then

$$[a^k \bmod p = a^{p-1} \bmod p]$$

Ques) $18^{-1} \bmod 7$

$\rightarrow 8^5 \bmod 7$

$5 < 7$

power $< p$ then we can use Fast exponentiation (square and multiply method)

Represent x in binary

$$x = 5$$

$$5 = (101)_2$$

$$8^5 \bmod 7$$

$$\text{Let } x = s = (101)_2$$

$$= (8^4 \cdot 8^1) \bmod 7$$

$$= ((8^4 \bmod 7)(8^1 \bmod 7) \bmod 7)$$

$$= (1 \times 1) \bmod 7$$

$$= 1 \bmod 7$$

$$= 1$$

que) $8^{-1} \pmod{17}$

$8^{15} \pmod{17}$

Since $15 < 17$, we use fermat exponentiation (square and multiply method)

Let $x = 15 = (1111)_2$

Using $8^8 \pmod{17} = 1$

$8^4 \pmod{17} = 16$

$8^2 \pmod{17} = 13$

$8 \pmod{17} = 8$

$1 \times 16 \times 13 \times 8 = 1664$

$1664 \pmod{17} = 15$

We can solve this using EEA to check the ans

* Euler's Theorem

Pg 257

It can be thought of as a generalization of Fermat's little theorem. The modulus in the Fermat theorem is a prime, the modulus in Euler's theorem is an integer. We introduce two versions:

First Version :

It is similar to first version of Fermat's little theorem. If a and n are coprime, then $[a^{\phi(n)}] = 1 \pmod{n}$

Second Version :

It is similar to second version of Fermat's little theorem ; it removes the condition that a and n should be coprime. If $n = p \times q$, $a \in n$, and k is an integer then

$$a^{k \times \phi(n) + 1} \equiv a \pmod{n}$$

Thus the Euler's theorem is used to solve problem related to modular exponentiation and to find multiplicative inverse of the number.

Multiplicative Inverse : Euler's theorem can be used to find multiplicative inverses modulo a prime; Euler's theorem can be used to find multiplicative inverse modulo a composite. If n and a are coprime then

$$a^{-1} \pmod{n} = a^{\phi(n)-1} \pmod{n}$$

$$5^{-1} \pmod{346} \rightarrow 5^{345} \pmod{346}$$

① $20^{62} \pmod{77}$ $(20 \pmod{77})^{62} \pmod{77}$

$$\textcircled{1} \quad 20^{62} \pmod{77}$$

Using second version,

$$a = 20 \quad n = 77$$

$$\gcd(20, 77) = 1$$

$$\phi(77) = \phi(7 \times 11)$$

$$= \phi(7) \times \phi(11) \quad (\text{using 3rd prop})$$

$$= 6 \times 10$$

$$= 60$$

$$20^{[60 \cdot 2]} \pmod{77}$$

$$\Rightarrow [20^{60} \pmod{77}] \times [20^2 \pmod{77}]$$

$$\textcircled{2} \quad 6^{24} \pmod{35}$$

$$a = 6 \quad n = 35$$

$$\phi(35) = \phi(7 \times 5)$$

$$= \phi(7) \times \phi(5)$$

$$= 6 \times 4$$

$$= 24$$

Using first version,

$$6^{24} \pmod{35} = 1$$

$$\textcircled{3} \quad \text{Proof that } 11^{\phi(10)} \pmod{10} = 1$$

$$a = 11 \quad n = 10$$

$$\phi(n) = \phi(10) = \phi(5 \times 2)$$

$$= 4 \times 1$$

$$= 4$$

$$11^4 \pmod{10} \Rightarrow 1 \quad [\text{calculator}]$$

$$④ 7^{-1} \bmod 15$$

$$n = 15$$

$$\phi(n) = \phi(15) = \phi(3 \times 5) \\ = 2 \times 4 = 8$$

$$7^{8-1} \bmod 15$$

$$7^7 \bmod 15 \Rightarrow 13$$

Using square and binary function
of & by applying modular property

$$7^7 \bmod 15 = 13$$

Q

* Chinese Remainder Theorem

In cryptography we use the concept of congruence instead of equality thus the two integers are congruent if we use the congruent operator (\equiv) and add the phrase $(\bmod n)$ to the right side of the congruence to define the value of modulus that makes the relationship valid. So in general, congruence eqn is shown by

$$x \equiv a (\bmod n)$$

\mathbb{Z}
↓
contains all
-ve, 0, +ve
values

\mathbb{Z}_n
↓
contains
0 to $n-1$
values.

$-8 \bmod 10$
 $(10-8)$

The congruence operator is used as equality in cryptography, but both are not same because it shows many to one relationship between each other.

For eg.

$$2 \equiv 12 \pmod{10}$$

$$2 \equiv 22 \pmod{10}$$

$$2 \equiv 222 \pmod{10}$$

$$2 \equiv -8 \pmod{10}$$

$$2 \equiv 2 \pmod{10}$$

$$2 \equiv 2 \pmod{10}$$

The CRT is used to solve a set of congruent equation with one variable but different moduli, which are relative prime, as shown below

$$x \equiv a_1 \pmod{m_1}$$

$$x \equiv a_2 \pmod{m_2}$$

:

$$x \equiv a_k \pmod{m_k}$$

The CRT states that above eq has unique solution if moduli are relatively prime.

To solve the congruent set of equations there are various methods available like CRT, Quadratic Reduce method, Discrete logarithmic method.

Following shows the steps :

- ① Find $M = m_1 \times m_2 \times \dots \times m_k$. This is the common modulus
- ② Find $M_1 = M/m_1, M_2 = M/m_2, \dots, M_k = M/m_k$.
- ③ Find the multiplicative inverse of M_1, M_2, \dots, M_k using corresponding moduli (m_1, m_2, \dots, m_k). Call the inverses $M_1^{-1}, M_2^{-1}, M_3^{-1}, \dots, M_k^{-1}$
- ④ The solution to simultaneous eq's

$$x = (a_1 \times M_1 \times M_1^{-1} + a_2 \times M_2 \times M_2^{-1} + \dots + a_k \times M_k \times M_k^{-1}) \text{ mod } M$$

For eg $x \equiv 2 \pmod{3}$

$$x \equiv 3 \pmod{5}$$

$$x \equiv 2 \pmod{7}$$

$$\rightarrow m_1 = 3, m_2 = 5, m_3 = 7$$

According to CRT algo,

$$M = 3 \times 5 \times 7 = 3 \times 35 = 105$$

$$M_1 = \frac{105}{3} = 35$$

$$M_2 = \frac{105}{5} = 21$$

$$M_3 = \frac{105}{7} = 15$$

To find M_1^{-1} , M_2^{-1} , M_3^{-1}

$$M_1^{-1} = 35^{-1} \pmod{3} \quad (\text{Using Fermat})$$

$$= 35^1 \pmod{3}$$

$$1 < 3, \quad x = 1 = 0001$$

$$35 \pmod{3}$$

$$\text{Ans } \therefore 23$$

HW

Find value of x for given set of congruent eqⁿ.

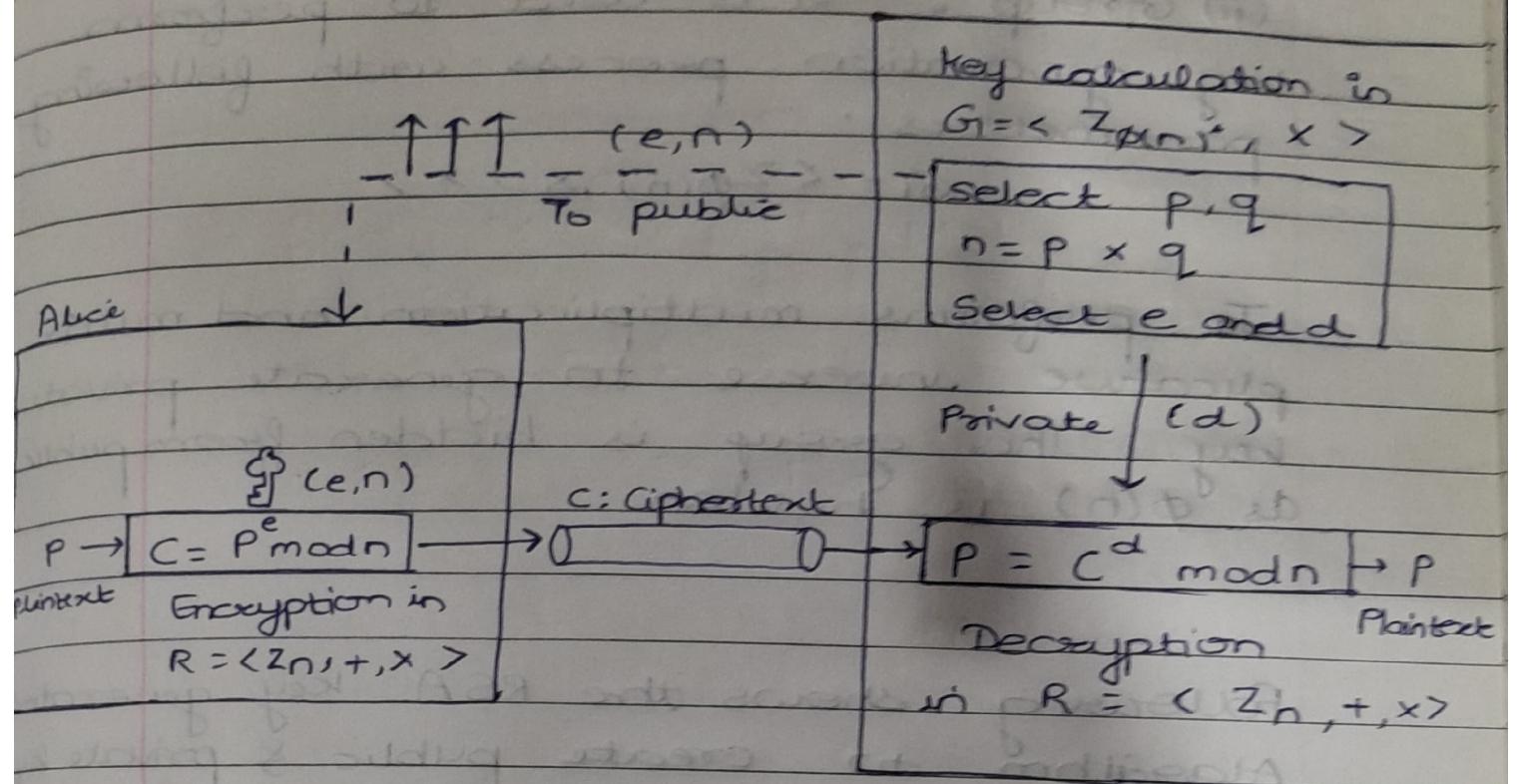
$$x \equiv 3 \pmod{7}$$

$$x \equiv 3 \pmod{13}$$

$$x \equiv 0 \pmod{12}$$

* RSA crypto system

- The most common public key algorithm is RSA cryptosystem which was invented by 3 inventors : Rivest, Shamir and Adleman.
- RSA uses two exponents 'e' & 'd' where 'e' is used to create Public key and 'd' is used to generate Public key.
- RSA uses modular exponentiation method to perform encryption and decryption.
- Assuming $PT = \text{Plaintext}$ and $CT = \text{Ciphertext}$, therefore encryption is performed using
$$CT = PT^e \mod n$$
- By, the decryption process uses $PT = CT^d \mod n$ to generate PT .
- The modulus 'n' should be a large number to improve security during key generation process.
- Following fig shows encryption, decryption and key generation process



From the above fig RSA uses two algebraic structures known as a ring and a group

(i) Ring : This structure is used to perform encryption and decryption with the following parameters

$$R = \langle \mathbb{Z}_n, +, \times \rangle$$

Thus the ring perform two arithmetic operation : Addition and multiplication

The ring is public as the modulus is public. Thus Alice can send message to Bob to perform encryption.

The reverse is used to perform the decryption at Bob side.

It is also known as
commutative ring

Home
Page

(ii) Group : G is used to perform key generation process with following parameters.

$$G = \langle \mathbb{Z}_{\phi(n)}, \times, \times \rangle$$

It performs multiplication and multiplicative inverse to generate private key. This group is hidden from public as $\phi(n)$ is hidden.

Following shows the RSA key generation algorithm to create public & private key. After key generation Bob announces the tuple e, n as the public key to be used by Alice for performing encryption. Similarly, Bob keeps the integer d as private key.

Imp

The recommended size for each prime number 'p' and 'q' is 512 bits i.e. 154 decimal digits and the size of modulus 'n' is 1024 bits i.e. 309 digits to make RSA algorithm more secure.

- RSA key Generation Algorithm :

RSA - key - Generation

{

Select two large prime $p \neq q$, such $p \neq q$

$$n \leftarrow p \times q$$

$$\phi(n) \leftarrow (p-1) \times (q-1)$$

Select e such that $1 < e < \phi(n)$ and e is coprime to $\phi(n)$

$$d \leftarrow e^{-1} \bmod \phi(n) \quad // d \text{ is inverse of } e \text{ modulo } \phi(n)$$

$$\text{Public-key} \leftarrow (e, n) \quad // \text{To be announced public}$$

$$\text{Private-key} \leftarrow d \quad // \text{To be kept secret}$$

return Public-key & Private-key

}

- RSA encryption :

RSA - Encryption (P, e, n)

{

$$C \leftarrow \text{Fast-exponentiation}(P, e, n)$$

return C

}

- RSA decryption :

RSA - Decryption (C, d, n)

{

$$P \leftarrow \text{Fast-exponentiation}(C, d, n)$$

return P

}

Unit 4 : Key Management

→ How to generate a session key and distribute it

two user key, b/w me,
for communication
the two have key
we have.

→ Symmetric key

↳ Alice $\xrightarrow{\text{key}_1}$ Bob
Bob $\xrightarrow{\text{key}_2}$ Alice

For n no. of user,

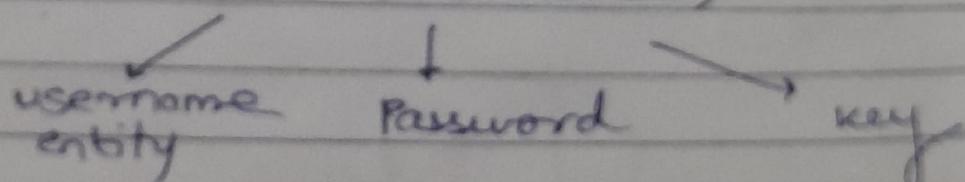
No. of session key = $n(n-1)$
for bidirectional

No. of session key = $\frac{n(n-1)}{2}$
for unidirectional

Key Distribution center .

Fig 15.1 for user

It is centralised server ~~use~~ which contain database
KDC \rightarrow db (database)



when Alice communicates with bob

① Alice send request to KDC

② KDC checks authentication

③ KDC sends request to bob

and in this way they communicate

④ If bob agrees session key is created between these two

Types of KDC

- ① flat multiple KDC (15.2)
- ② Hierarchical multiple KDC (15.3)

→ Write KDC approach ?

→ Summary of KDC .

→ Working of KDC ? *→ syllabus* *stay away*
4 not syllabus

• KDC uses Needham-Schroeder Protocol

There are chances of replay attack which is improved using Needham Protocol.

In this session ko log in and log out time diya jayega .

Session will be expired ,

i.e Session time out hogा .

i.e We will use time out function .

nonce → number's once .

↓
pseudo random no. which can be used only one time .

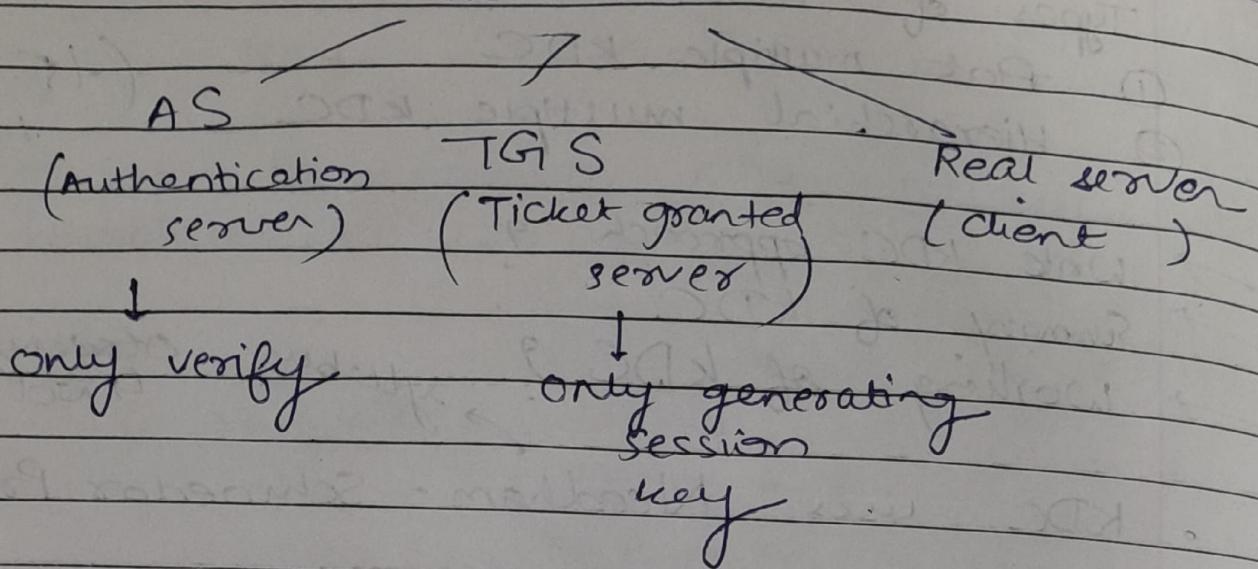
* Kerberos .

It is network authentication protocol which is used in KDC .

Version available are 4 & 5 .

→ Write a short note on kerberos

Kerberos



These were two responsibilities of KDS which are distributed in two servers.

→ Write a short note on Kerberos and explain with example. 10 marks
(2 diagrams)

Kerberos version 5

Symmetric key KDC

- ① Using KDC (using Kerberos)
- ② without using KDC (using Diffie-Hellman key agreement protocol)

A session key can be created using two ways

* Diffie Hellman method Fig 15.9.

The values of p and g are public.

It means that we are using it through internet.

As shown in the above figure let the two parties choose two random numbers ' p ' and ' g ' where ' p ' is a large prime number of the order 1024 bits i.e maximum 300 decimal digit.

Similarly, the second number ' g ' is a generator of the order $p-1$ in the group $\langle \mathbb{Z}_p^*, \times \rangle$ \times : multiplicative inverse

These two values are public values and can be send through the internet. Following are the steps to be followed

- ① Alice choose a large random number x such that $0 \leq x \leq p-1$ and calculates $R_1 = g^x \text{ mod } p$.
- ② Bob chooses another large random number y such that $0 \leq y \leq p-1$ and calculates $R_2 = g^y \text{ mod } p$.
- ③ Alice sends R_1 to Bob. Note that Alice does not send the value of x ; she sends only R_1 .

④ Bob sends R_2 to Alice. Again, note that Bob does not send the value of y , he sends only R_2

⑤ Alice calculates $K = (R_2)^x \bmod p$

Bob calculates $K = (R_1)^y \bmod p$

from the above steps K is symmetric key for the session.

$$\begin{aligned} K &= (g^x \bmod p)^y \bmod p \\ &= (g^y \bmod p)^x \bmod p \\ &= g^{xy} \bmod p \end{aligned}$$

Que) Let $P = 23$, $g = 7$, $x = 3$, $y = 6$

use Diffie Hellman key exchange algorithm and get value of session key K

$$\begin{aligned} R_1 &= g^x \bmod p \\ &= 7^3 \bmod 23 \\ &= 21 \end{aligned}$$

$$\begin{aligned} R_2 &= g^y \bmod p \\ &= 7^6 \bmod 23 \\ &= 4 \end{aligned}$$

$$\begin{aligned} K &= (R_2)^x \bmod p \\ &= (4)^3 \bmod 23 \\ &= 18 \end{aligned}$$

Assignment Que

- ① In Diffie - Hellman Protocol, $g = 7, p = 23$, $x = 3, y = 5$
- (A) what is value of symmetric key k
 - (B) what is value of R_1 and R_2
- ② In the Diffie Hellman protocol what happens if x and y have same value i.e Alice and Bob choose same number.
- ③ Are R_1 and R_2 the same. Do the session keys calculated by Alice and Bob have the same values. Use an example to prove your claim.

Ans 1 $\Rightarrow g = 7, p = 23, x = 3, y = 5$

$$\begin{aligned} R_1 &= g^x \bmod p \\ &= 7^3 \bmod 23 \\ &= 21 \end{aligned}$$

$$\begin{aligned} R_2 &= g^y \bmod p \\ &= 7^5 \bmod 23 \\ &= 17 \end{aligned}$$

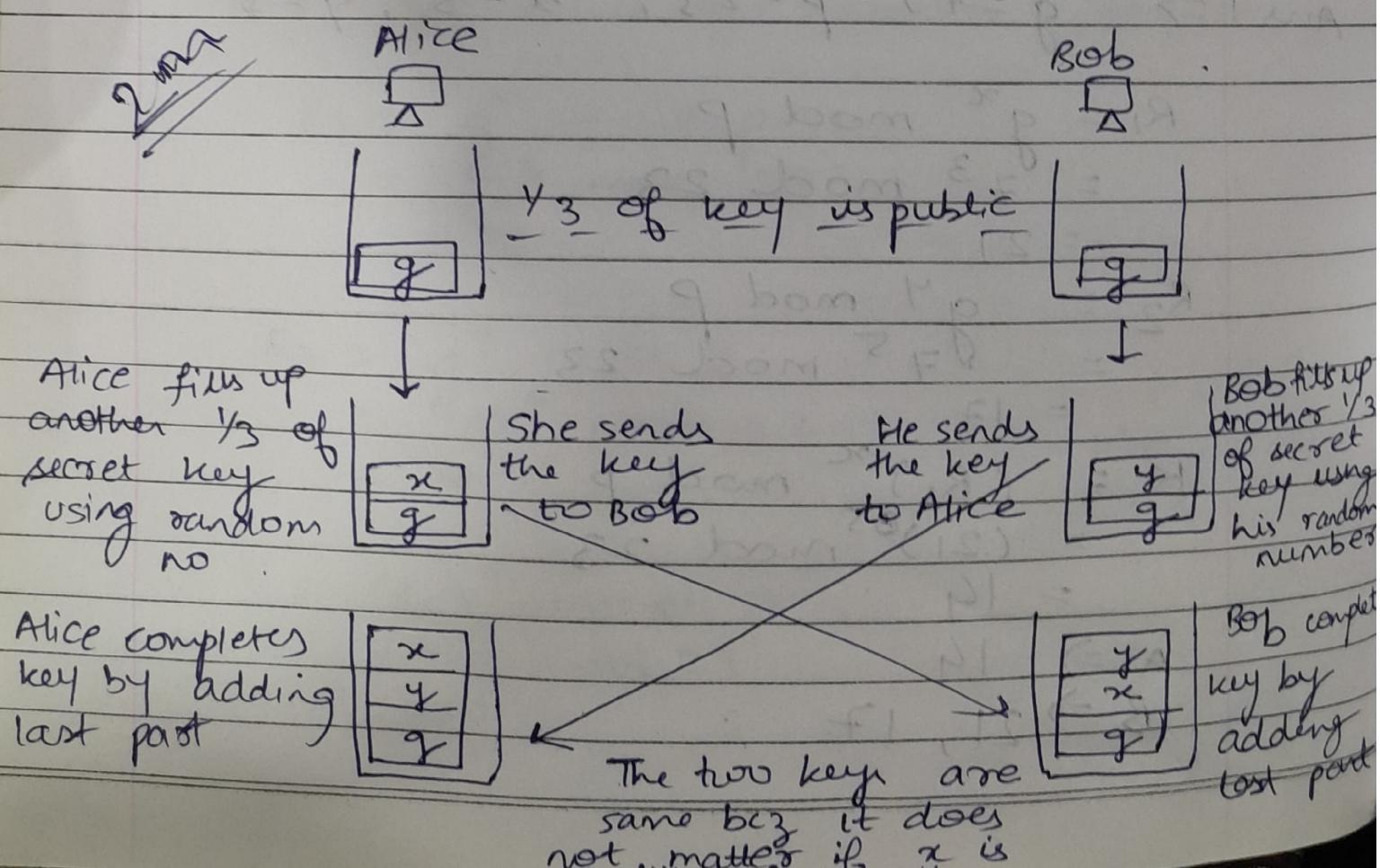
$$\begin{aligned} K &= (R_1)^x \bmod p \\ &= (21)^3 \bmod 23 \\ &= 14 \end{aligned}$$

$$A \Rightarrow 14$$

$$B \Rightarrow 21, 17$$

* Analysis of Diffie Hellman Algo.

As shown in the following figure the session key between Alice and Bob is made up of 3 parts g, x, y . The first part i.e 'g' is made public whereas the other two parts must be added by Alice and Bob. As the two keys are same, Alice cannot find the value 'y' used by Bob because the calculation is done in modulo p. Similarly Bob cannot find the value 'x' used by Alice for the reason same. Thus to know the values of x and y Discrete logarithm method is used.



* Public key Distribution X.509 certificate

- In public key cryptography, everyone has access to everyone's public key; public keys are available to the public.
- Public key like secret key or session key needs to be distributed publically.

① Public Announcement

→ Announcing a public key

② Trusted center

Brokerage

③ Controlled Trusted center.

④ Certification Authority

* X.509 certificate format

Fig 15.17

As each certificate have different formats the ITU has designed X.509 certificate format in a structured way accepted by internet. It is a well known protocol called as ASN.1 (Abstract syntax notation)

ITU : International telecommunication union

→ bytes wala diagram google se dekhna.

Date
Page

Following fig shows X.509 (1518 bytes) pg 482

