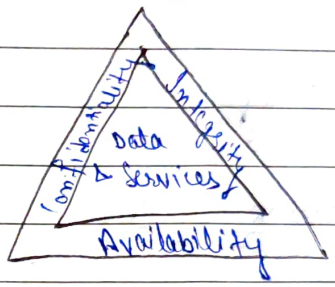
Page No.
Date :

3rd

Computer Security:

The protection afforded to an automated information system in order to attain the applicable objectives of preserving confidentiality, integrity & availability of information system resources.

CIA Triad:



Additional → Authenticity
↳ Accountability

- Level of impact of security breach.

* Low * Medium * High
↳ ∞ Everything is gone.

- Confidentiality (Eg: Account info).
- Integrity (Eg: Patient's info).
- Availability (Eg: Authentication service, google etc.)

- **Threat**

A potential for violation of security, which exists when there is a circumstance, capability, action that could breach security & cause harm.

३ॐ

Page No.
Date :

The OSI Security Architecture

Security Attack Security mechanism Security Service

Security Attack:

- Action that compromises the security of an individual or an organisation.

Types

(i) **Passive attack** :- Attempts to learn or make use of information from the system.
eg. unauthorised reading of messages.
(eavesdropping the conversation.)

- Does not affect system resources.
- Goal :-> Obtain information that is being transmitted

Types
(i) Release of message contents
(ii) Traffic Analysis.

(ii) **Active attack**:-

- Active attacks involve some modification of the data stream or the creation of a false stream.
- Subdivided into four categories.
(i) Masquerade.
(ii) Replay.
(iii) Modification of messages.
(iv) Denial of Service (DoS).

Page No.
Date :

② Security Service :-

The processing or communication service that is provided by a system to give a specific kind of protection to system resources, security services implement security policies and are implemented by security mechanism.

* Authentication.
  - Peer entity authentication
  - Data origin authentication.

* Access Control
* Data confidentiality eg encryption
* Data Integrity.
* Non repudiation.

③ Security Mechanism :-

→ Specific Security mechanism :-

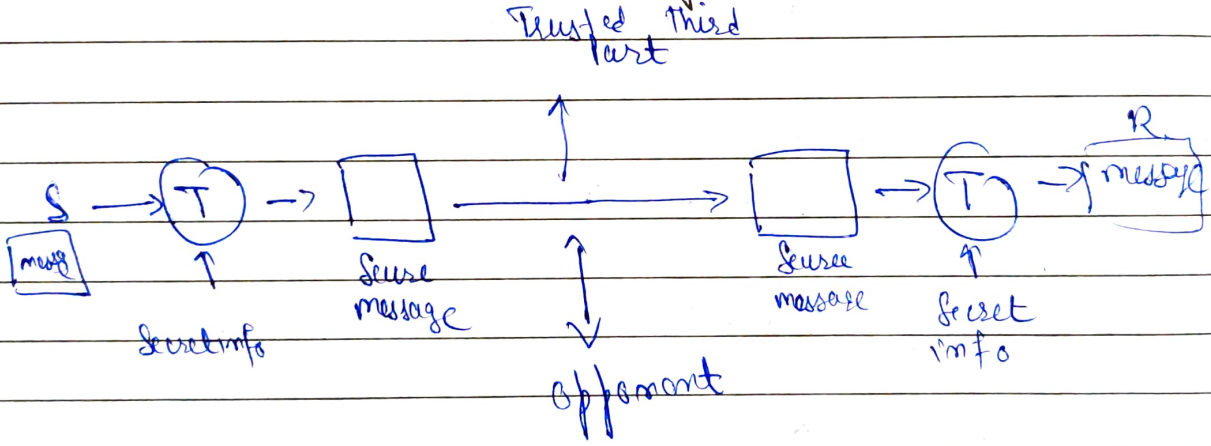
* Encipherment
* Digital Signature.
* Access control
* Data Integrity
* Authentication Exchange.
* Traffic Padding.
* Routing control.
* Notarization

Page No.
Date :

(iii) Pervasive Security mechanism

- Trusted functionality.
- Security label.
- Event Detection.
- Security Audit trail
- Security recovery.

Model for network Security Model



Four major tasks

⇒ Design an algorithm.
⇒ Generate the secret information.
⇒ Develops methods for distribution & sharing of information.
⇒ Specify a protocol.

Page No.
Date :

Cryptography

Plain text → Encryption Algo → Transmitted Ciphertext → Decryption Algo → Plain text

→ the art or science encompassing the principles & methods of transforming an intelligible message into one that is unintelligible & then retransforming that message back to its original form.

Types of cryptography

- Symmetric Cryptography (Private key cryptography)
- Asymmetric Cryptography (Public key cryptography)

- Encryption scheme
  - ↳ Unconditionally secure
  - ↳ Computationally secure.

- Cryptanalysis :

Cryptanalytic attacks → Based on the info known to cryptanalyst.

- most difficult : Cipher text only.

- Types of cryptanalytic attacks

(i) Cipher text only.
(ii) Known Plain text
(iii) Chosen Plain text
(iv) Chosen Cipher text
(v) Chosen Text

Page No.
Date :

Brute force attack

- Try every possible key.
- Until an intelligible translation of the ciphertext into plaintext is obtained.
- Guessing.
- Exhaustive key search.
- Software tools that can perform brute-force attack.

Completely automated public turing test to tell computers & humans apart.
(CAPTCHA)

- Classical Encryption techniques

1. Substitution Techniques :-

* Letters are replaced by other letters or symbols

a → m, b → x, x → z, g → A

Plain text : bag
Cipher text : xmA

2. Transposition Techniques :

* Applying some sort of permutation on the plaintext letters.

Page No.
Date :

Plaintext: NOTE
Ciphertext: ONTE, TNUE, RONT..etc.

Substitution
Caesar Ciphers
Mono alphabetic Ciphers
Play fair Ciphers
Hill Ciphers
Poly alphabetic Ciphers
One-Time Pad

Transposition
Rail fence
Row Column Transposition

Caesar Ciphers

→ Letters are replaced by other letters or symbols
⇒ Replacing each letter of the alphabet with the letter standing three places further down the alphabet.

Algorithm:

For each plaintext letter 'p', substitute the ciphertext 'C'

$$C = E(p,k) \bmod 26 = (p+k) \bmod 26$$

$$p = D(C,k) \bmod 26 = (C-k) \bmod 26$$

Pros: 1. Simple.
2. Easy to implement.

Cons: 1. The encryption & decryption algorithms are known
2. There are only 25 keys to try. (Vulnerable to brute force)
3. The language of the plaintext is known & easily recognizable.

Page No.
Date :

Monoalphabetic Ciphers :-

→ The 'cipher' line can be any permutation of the 26 alphabetic characters.

⇒ This would seem to eliminate brute force techniques for cryptanalysis.

→ A single cipher alphabet (mapping from plain alphabet to cipher alphabet) is used per message.

⇒ English language :- Nature of plain text is known.

Pros

1. Better security than caesar cipher.

Cons

1. Monoalphabetic ciphers are easy to break because they reflect the frequency data of the original alphabet.
2. Prone to guessing attack using the english letter freqiency of occurence of letters
3. A counter measure is to provide multiple substitutes, known as homophones for a single letter.

Page No.
Date :

Playfair Cipher

→ Also Playfair square or Wheatstone-Playfair cipher.
→ Manual symmetric encryption technique.
→ Multiple letter encryption cipher.
→ Diagrams.
→ 5×5 matrix constructed using a keyword. (Ex: monarchy)

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Rules
(i) Diagram.
(ii) Repeating letters - filler letter.
(iii) Same column (↓) wrap around.
(iv) Same row (→) wrap around.
(v) Rectangle (⇆) Swap

Ex P: attack
D: at ta ck

P: balloon
D: ba ll oo n
ba lx lo on
↳ filler character
↳ also called bogus

Page No.
Date :

Ex1 attack
Diagrams: at ta ck

at	ta	ck
RS	SR	DE

Ex2 mosque

mo	sq	ue
ON	TS	ML

→ Hill Cipher

⇒ Multi-letter cipher
→ Encrypts a group of letters: diagraph, trigraph or polygraph.

*The Hill Algorithm:-
*This can be expressed as

$$C = E(K, P) = P \times K \bmod 26$$
$$P = D(K, C) = C K^{-1} \bmod 26$$
$$= P \times K \times K^{-1} \bmod 26$$

$$(C_1\ C_2\ C_3) = (P_1\ P_2\ P_3) \begin{pmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{pmatrix} \bmod 26$$

$$C_1 = (P_1 K_{11} + P_2 K_{21} + P_3 K_{31}) \bmod 26$$
$$C_2 = (P_1 K_{12} + P_2 K_{22} + P_3 K_{32}) \bmod 26$$
$$C_3 = (P_1 K_{13} + P_2 K_{23} + P_3 K_{33}) \bmod 26$$

Page No.
Date :

Ex: Encrypt "pay more money" using Hill cipher with key.

$$\begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}$$

Sol:

p	a	y	m	o	r	e	m	o	n	e	y
15	0	24	12	14	17	4	12	14	13	4	24

Key = 3×3 matrix

PT = pay mor eno ney

Encrypting: pay

$$C_1C_2C_3 = [15\ 0\ 24] \begin{bmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{bmatrix} \bmod 26$$

$$= [15\times17 + 24\times2 \quad 15\times17 + 24\times2 \quad 15\times5 + 24\times19] \bmod 26$$

$$= [303\ \ 303\ \ 531] \bmod 26$$

$$= [17\ \ 17\ \ 11]$$

$$= [R\ R\ L]$$

$$(C_1C_2C_3) = [12\ \ 14\ \ 17) \begin{bmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{bmatrix} \bmod 26$$

$$= [532\ \ 490\ \ 677] \bmod 26$$

$$= [12\ \ 22\ \ 1]$$

Final cipher = RRLMUBXASPDH

Page No.
Date :

Polyalphabetic cipher

→ To improve the simple monoalphabetic cipher
→ Polyalphabetic substitution ciphers.

Common features

1. A set of related monoalphabetic substitution is used.
2. A key determines which particular rule is chosen for a given transformation.

Vigenere Cipher

* It consists of the 26 Caesar ciphers with shifts of 0 through 25

Encryption process:

$$C_i = (P_i + K_{i \bmod m}) \bmod 26$$

Decryption

$$P_i = (C_i - K_{i \bmod m}) \bmod 26$$

Eg
key: deceptive deceptive deceptive
P text: wearediscoveredsaveyourself
ciphertext: ZICVTWQNGKZEIIGASXSTSLVVWA

Auto key system:
↳ The periodic nature of the keyword can be eliminated by using a non-repeating keyword that is as long as the message itself

Page No.
Date :

→ Vigenere proposed autokey system, in which a keyword is concatinated with the plaintext itself to provide a running key.

Ex
Key: deceptive wearediscoveredsav
Ptxt: wearediscoveredsaveyourself

One-time Pad

→ Random key is that is as long as message.
→ The key need not be repeated
→ In addition, the key is used to encrypt & decrypt a single message & then is discarded.
→ Each new message requires a new key of the same length as the new message.

→ Such a scheme, is known as one-time pad, is unbreakable.

Two fundamental problems

→ The practical problem of making large quantities of random keys.
→ Even more daunting is the problem of key distribution & protection
→ That's why used primarily for low-bandwidth channels requiring very high security

Perfect Secrecy

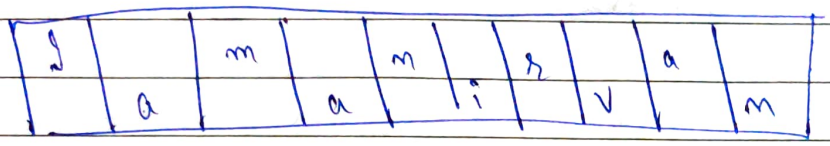
→ The one-time pad is the only cryptosystem that exhibits what is refferred to as perfect secrecy.

Page No.
Date :

• Rail Fence Technique

⇒ The plaintext is written down as a sequence of diagonals & then read off as a sequence of rows.

Ex 'I am Aniruem' Depth = 2



⇒ imnsavaaivm

Steganography

→ Conceal the existence of message.
→ Hiding the message.
→ Not an encryption scheme.
→ Cryptography renders the message unintelligible to outsiders by various transformations of text.

Ex Example: Simply encrypt correct reading exactly twice.

* Character marking.
* Invisible inks.
* Pin punctures.
* Typewriter color ribbon

Page No.
Date :

Drawbacks

→ Lot of overhead.
→ Once the system is discovered, it becomes virtually worth less.

LSB Steganography

↳ Least Significant Bit
↳ 10101011 → LSB

Watermarking is about establishing identity of information to prevent unauthorise use.
↳ They are imperceptible.
↳ They are inseperable from the works they are embedded in.
↳ They remain embedded in the work even during transformations.

Stream Cipher
↳ one byte (128 bit) is encrypted at a time.
↳ key of size 8-bit will be generated with pseudorandom bit generator.

Encryption
↳ Plaintext & keystream produces cipher text.
↳ The plaintext will undergo XOR operation bit by bit & produces the cipher text.

Page No.
Date :

Ex

Ptxt:	10011001	10011001
K :	11000011	11000011
Ctext:	~~01001010~~ 01011010	01011010

Decryption

- Cipher text & keystream gives the plain text with same key.
- Undergo XOR.

Ctext:	01011010
K :	11000011
Ptext:	10011001

Block Cipher

Page No.
Date :

Diffie Hellman key exchange algorithm

⇒ Symmetric key cryptography - the problem of key distribution key exchange solution is not fully practical possible.

→ This problem is called as key distribution or key exchange problem.

→ It is inherently linked with the symmetric key cryptography.

→ It allows 2 parties to share keys.

→ This algorithm can be used only for key agreements, but not for encryption or decryption.

→ It is based on mathematical principles.

Algorithm

1. Firstly Alice & Bob agree upon 2 large numbers - n & q.
   The 2 numbers need not to be secret & can be shared publicly.
2. Alice chooses another large number (random) X (private to her) & calculate A such that: $A = q^X \bmod n$.
3. Alice sends this to Bob. (X)
4. Bob chooses another large number (private to him) & calculate B such that: $B = q^Y \bmod n$.
5. Bob sends this to Alice.

Page No.
Date :

6. Alice now computes his private key as follows:

$$K_1 = B^x \bmod n.$$

7. Bob computes his secret key K_2 as:

$$K_2 = A^y \bmod n.$$

(8) $K_1 = K_2$ (key exchange complete)

Man in the middle attack $n=11, g=7$

Alice	Tom (Hacker)	Bob
$x=3$	$x=8$, $y=6$	$y=9$
$A^{(a)} = g^x \bmod n$	$A = g^x \bmod n$	$B^{(b)} = g^y \bmod n$
$= 7^3 \bmod 11 = 2$	$= 7^8 \bmod 11$	$= 7^9 \bmod 11 = 8$
	$= 9$	$B = 7^6 \bmod 11$
$B^{(t)} = 4$		$= 4$
	$A(a) = 2$	$B(b) = 8$
$K_1 = B^x \bmod n$	$K_2 = A(a)^y \bmod n$	$K_1 = B(b)^x \bmod n$
$= 4^3 \bmod 11$	$= 2^6 \bmod 11$	$K_1 = 8^8 \bmod 11$
$= 9$	$K_2 = 9$	$K_1 = 5$

$A^{(t)} = 9$

$K_2 = A^y \bmod n$

$= 9^9 \bmod 11$

$= 5$

Page No.
Date :

Digital Signature :

→ A digital signature is equivalent to handwritten signature.

→ It is an electronic verification of the sender.

A digital signature 3 purposes

1) **Authentication :**
A digital signature gives the reciever reason to believe the message was created and sent by the claimed sender.

2) **Non-repudiation :**
With digital signature, the sender cannot deny having sent the message later on.

3) **Integrity :**
A digital signature ensures that the message was not altered in transit.

Page No.
Date :

Basics of Channel & Code word

→ Every channel has an upper limit on the rate at which information can be transmitted reliably through the channel.

→ The limitation of capacity of channel to transmit information is referred as the CHANNEL CAPACITY.

→ Due to channel noise there will be errors in transmitted bits at receiver side, & to correct error we use block codes.

→ A block code is a set words that has well defined mathematical property structure where each word is sequence of fixed no. of bits.

→ The words belonging to a ~~code~~ block code are called as CODE WORDS.

Basics of Block Code

Information bits = k

$$i = [i_1, i_2, i_3, \ldots, i_k]$$

Parity bits / Redundant bits = r

$$p = [p_1, p_2, p_3 \ldots p_r]$$

Here, total bits of code, $n = k + r$

$m = [i_1, i_2, i_3, \ldots i_k, p_1, p_2, p_3, \ldots p_r]$

→ (n, k) is block code representation.

→ A code word, whose information bits are kept together is Systematic.

→ A code word, whose information bits are not kept together is non-systematic.

Important parameters of block code:

- Total code words required as per n Block codes $= 2^n$
- " " " " " k info bits $= 2^k$

Total redundant code words required as r parity bits $= 2^n - 2^k$

Code rate $= \boxed{k/n = R}$

k information bits | r redundant bits (Parity bits)

Ex (4,3) block codes

even parity check

	3 info Bits (i)	Parity (p)	Code word
✓	0 0 0	0	0 0 0 0
✗	0 0 0	1	0 0 0 1
✗	0 0 1	0	0 0 1 0
✓	0 0 1	1	0 0 1 1
✗	0 1 0	0	0 1 0 0
✓	0 1 0	1	0 1 0 1
✓	0 1 1	0	0 1 1 0
✗	0 1 1	1	0 1 1 1
✗	1 0 0	0	1 0 0 0
✓	1 0 0	1	1 0 0 1

Page No.
Date :

1 0 1	0	1 0 1 0
1 0 1	1	1 0 1 1
1 1 0	0	1 1 0 0
1 1 0	1	1 1 0 1
1 1 1	0	1 1 1 0
1 1 1	1	1 1 1 1

- Info bits = x

$$i = [i_1, i_2, i_3, \ldots, i_x]$$

- Parity bits / Redundant bits = r

$$p = [p_1, p_2, p_3, \ldots p_r]$$

- Code word, $c = [I, p]$

$$c = [i_1, i_2, i_3 \ldots i_x, p_1, p_2 \ldots p_r]$$

Error code word

$$e = [e_1, e_2, e_3, \ldots e_n]$$

$e_j = 1$ means errors.
$e_j = 0$ means no errors.

Valid data = received code word + error code word

Page No.
Date :

Generator matrix to generate code words in linear code

- using a matrix to generate code words is a better approach.

$$[C] = [i][G]$$

C → code word, i → info word, G → generator matrix

- The generator matrix of an (n, k) linear code has 'k' rows & 'n' columns.
- Generator matrix for $(7,4)$ code is given by

$$[G] = [I : P]$$

$$= \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

Identity matrix | Parity matrix

Ex: Generate codeword for $i = (1110)$ with $(7,4)$ generator matrix code.

$$G = \left[\begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & \phi \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{array}\right]$$

Sol: message $[i] = [1110]$

Page No.
Date :

$$C = [i][y]$$

$$= \begin{bmatrix} 1 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}$$

→ Take mod 2 ↑ here.

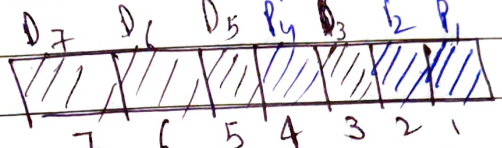
Page No.
Date :

Hamming Code

→ Given by R.W Hamming.
→ Easy to implement.
→ 7-bit hamming code is used commonly.

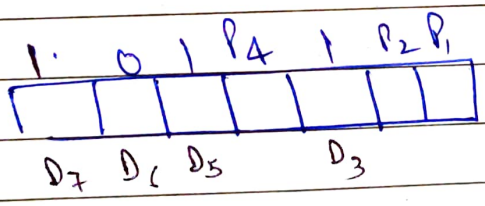
Data bits - 4
Parity bits - 3

2^n { where $n = 0, 1, \ldots n$ } → position of hamm [bit]



$P_1 \rightarrow D_3 D_5 D_7$
$P_2 \rightarrow D_3 D_6 D_7$
$P_4 \rightarrow D_5 D_6 D_7$

Ex 1011



$P_1 \rightarrow 1$ [1 1 1]
$P_2 \rightarrow 0$ [1 0 1]
$P_4 \rightarrow 0$ [1 0 1]

P_x ——— Tx

1 0 1 0 1 0 1

(1 0 1 0 1 0 1)

$P_1 = 1$
$P_2 = 0$
P_4

Page No.
Date :

Hamming code is the set of error-correcting codes that can be used to detect & correct errors that can occur when data is moved or stored from sender to the receiver.

Redundant bits

Redundant bits are extra bits that are generated & added to the information-carrying bits of data transfer to ensure no bits were lost during the data transfer.

$$2^r \geq m + r + 1$$

↳ r = redundant bits
m = data bit

Parity bits

A parity bit is a bit appended to a data of binary bits to ensure that the total number of 1's in the data is even or odd. They are used for error detection.

↳ Odd Parity bits.
↳ Even n bits

Page No.
Date :

General Algorithm of Hamming code :-

(i) Write the bit positions starting from 1 in binary form.

(ii) All the bits that are power of 2 are marked as parity bits. (1, 2, 4, 8 etc.)

(iii) All the other bits are marked as parity bits.

(iv) Each data bit is included in a unique set of parity bits, as determined its bit position in binary form.

(v) Since we check for even parity set a parity bit to 1 if the total number of ones in the position it checks is ~~even~~ odd.

(vi) Set it to 0 if total ones is even.

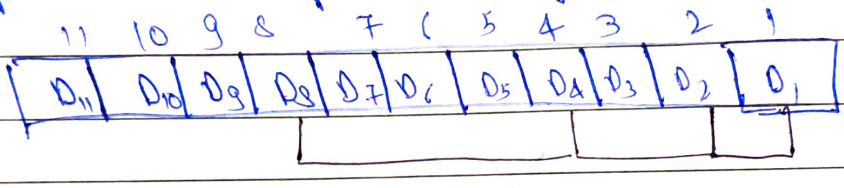
$R_1 \rightarrow 1, 3, 5, 7, 9, 11$
$R_2 \rightarrow 2, 3, 6, 7, 10, 11$
$R_3 \rightarrow 4, 5, 6, 7$
$R_4 \rightarrow 8, 9, 10, 11$

Position	R8	R4	R2	R1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

10	9	8	7	6
11	1	0	1	1

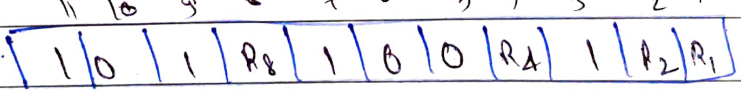
Determining the position of redundant bits -

These redundancy bits are placed which corresponds to power of 2.



Redundant bits

Suppose the data to be transmitted is 1011001, the bits will be placed as.



Determining the parity bits -

R_1 : bits 1, 3, 5, 7, 9

R_1, 1, 0, 1, 1, 1

$\therefore R_1 = 0$

R_2 : 2, 3, 6, 7, 10, 11

R_2, 1, 0, 1, 0, 1

$R_2 = 1$

R_4 : R_4, 5, 6, 7

R_4 0, 0, 1

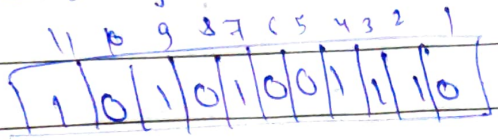
Page No.
Date :

R_8: $R_{8,9,10,11}$

R_8 1 0 1

$R_8 = 0$

∴ data transferred is:



Error detection & correction

Suppose in the above example 6th bit is changed from 0 to 1 during data trans—, then it gives new parity values in binary numbers.

Buffer overflow:

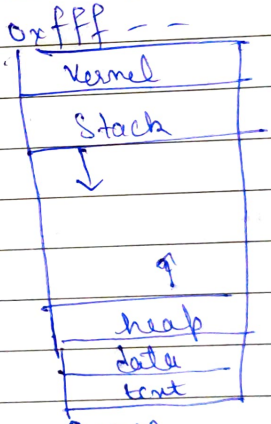
A buffer is a small memory allocation used when a program or a process is executed.

OR a buffer is a small memory allocation for an input.

Buffer overflow is a condition when a program writes more data to the memory than it is actually supposed to take.

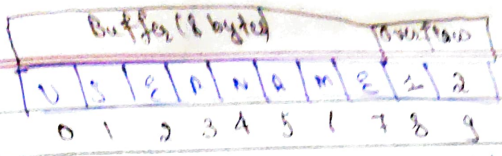
? What will happen to the extra input that went into buffer input?

? What if the buffer contains any malicious script / virus?



Attackers plants code that overflows buffer & corrupts the return address. Instead of returning to the appropriate calling procedure, the modified return address returns control to malicious code, located elsewhere in process memory.

→ Buffer overflow can challenge the integrity of the server or system.



Buffer overflow example

overflow

If the overwritten part in memory contains a pointer the attacker could replace that code with another pointer that points to an exploit payload. This can transfer control of the whole program over to the attacker's code.

Protection against buffer overflow

- **Address space randomisation :-**

Randomly rearranges the address space locations of key data areas of a process. Buffer overflow attacks generally rely on knowing the exact location of important executable code, randomisation of address spaces makes that nearly impossible.

- **Data execution prevention:-**

Marks certain areas of memory either executable or non-executable, preventing the code in a non-executable area from running.

Page No.
Date :

Integer overflow

↳ occurs when an attacker causes a value in the program to be large enough to overflow unexpectedly.

→ occurs when the result of an arithmetic operation such as multiplication or addition, exceeds the max size of int type used to solve it

maximum possible value a binary integer can hold $= 2^n - 1$
$= n$ = no of bits

→ Integer overflow can cause software vulnerabilities like heap overflow.

3.2 Types of Malware

Bots :-

Bots are created to perform a series of actions automatically. It is used to perform DDOS attacks to get access to servers. It can be used to auto render ads clicks on the ads to benifit its creator.

How antimalware catches it?

- Analyzes the behaviour of all programs, when it detects a repetitive action, it blocks that program.

eg:- CAPTCHA services are used in website.

Page No.
Date :

Root kits:

They are created to provide remote access to a computer w/o getting detected. With root kits hackers could remotely execute files, modify system settings, alter the software, steal data etc

How antimalware catches it?
It needs both signature & behavioural base detection methods. Catching the security loop holes & vulnerabilities with both latest updates removes the chances of Rootkits injection.

Trojan Horse

It enters the users computer as a normal file or program & performs malicious tasks set by its maker. It can give remote access to hackers or steal info.

How antimalware catches it?

Uses Heuristic Analysis & Sandbox to detect the trojan. It analysis the behaviour of the program.

VIRUS :-

Virus is a type of malware that copies itself and spread to other computers It spreads itself by attaching itself with other files

How antimalware catches it?

Primarily uses signature based detection For complex viruses behavioural analysis is used.

Worms:

Similar to viruses, while virus need user actions to spread, the worms don't require that. It can spread on its own & it is designed to overload the webservers by consuming the bandwidth.

How antimalware catches it?

It needs to have a network monitor feature to catch it since worms spread through network usually.

Page No.
Date :

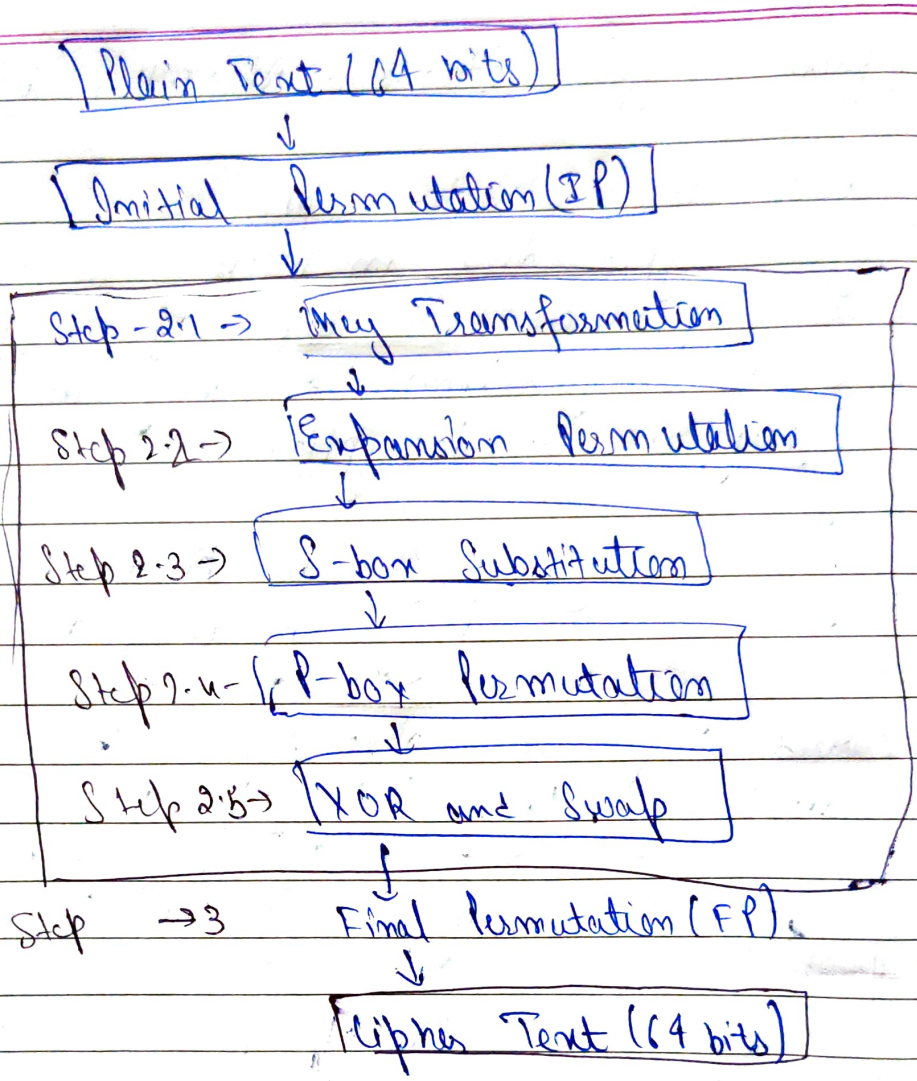
Data Encryption Standard Algorithm (DES)

→ DES is a block cipher.
→ It encrypts data in block size of 64 bits each.
→ It produces 64 bit of cipher text.
→ Same algorithm & key are used for encryption & decryption.

→ The key length is ~~64 bit~~ 56-bits. The key originally consists of 64 bits - however only 56 of these are actually used by the algorithm. Eight bits are used solely for parity check, & are these after discarded. Hence, the effective key length is 56 bits

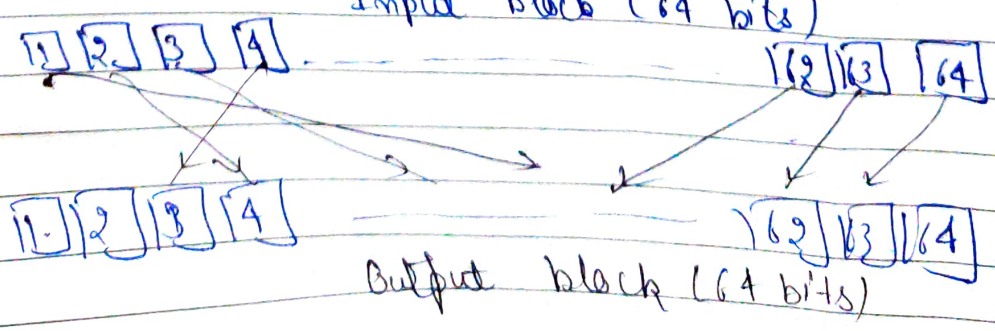
→ Consists of 16 steps each of which is called as round.
Each round performs the steps of substitution & transformation.

→ It is based on 2 - fundamental attributes.
1. Substitution - also called as confusion.
2. Transposition - also called as diffusion.



Step 1 : Initial Permutation

- It happens only once, before first round
- After IP is done, the resulting 64 bit permuted block is divided into two 32-blocks ; (RPT & LPT).
- 16 rounds are performed on these 2 blocks.



Page No.
Date :

16 - Rounds (1 Round working)

S-2.1 Key Transformation

→ For each round, a 56-bit is ~~as~~ available

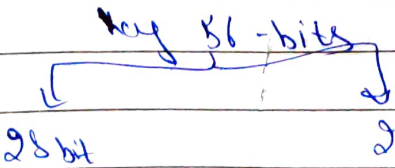
→ From this 56-bit key, a different 48-bit subkey is generated during each round using a process called key transformation.

→ The 56-bits is divided into two halves each of 28 bits.

→ These halves are circularly shifted left by one or two positions depending on the round.

→ After an appropriate shift, 48 of the 56 bits are selected.

→ The key transformation process involves permutation as well as selection of a 48-bit subset of the original 56-bit key, it is called as **Compression Permutation.**



~~0 + 0 = 1~~
1 0 1 0
1 0 1 0 1

Page No.

Date :

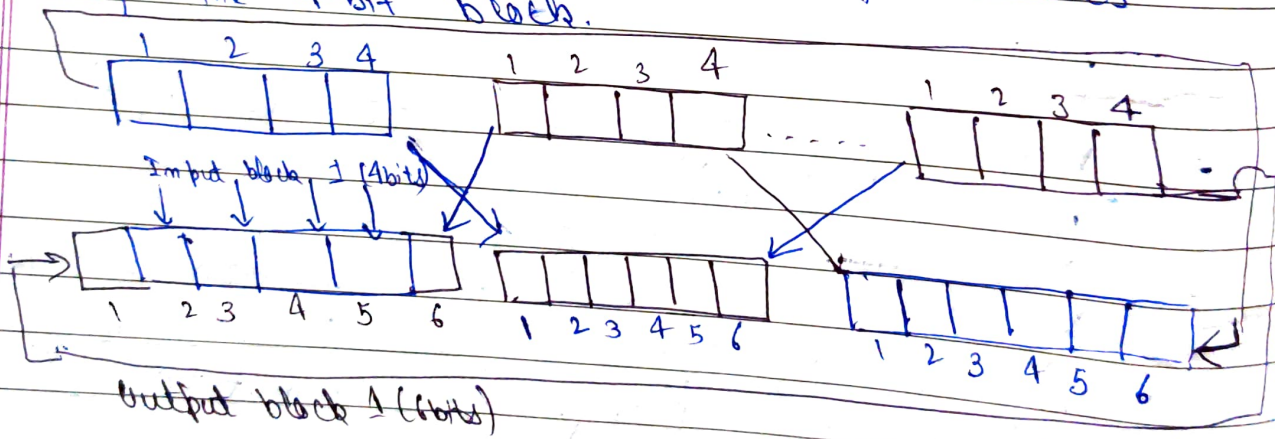
S-2-2 Expansion Permutation

⇒ During expansion permutation, the RPT is expanded from 32-bits to 48 bits.

⇒ Besides increasing the bit size from 32 to 48, the bits are permuted as well, hence the name expansion permutation.

⇒ The 32-bit is divided into 8 blocks, each consisting of 4 bits.

⇒ Next each 4 bit block is then expanded to 6-bit block. They are actually the repeated first & the fourth bits of the 4-bit block.



Key Transformation (Compress key from 56 bits to 48 bits)

↓

48-bit key ⟶ XOR ⟵ 48-bit RPT

Expansion Permutation (Expand RPT from 32 bits to 48 bits)

↓

↓

S-Box Substitution

Page No.
Date :

§ 2.3 S-Box Substitution

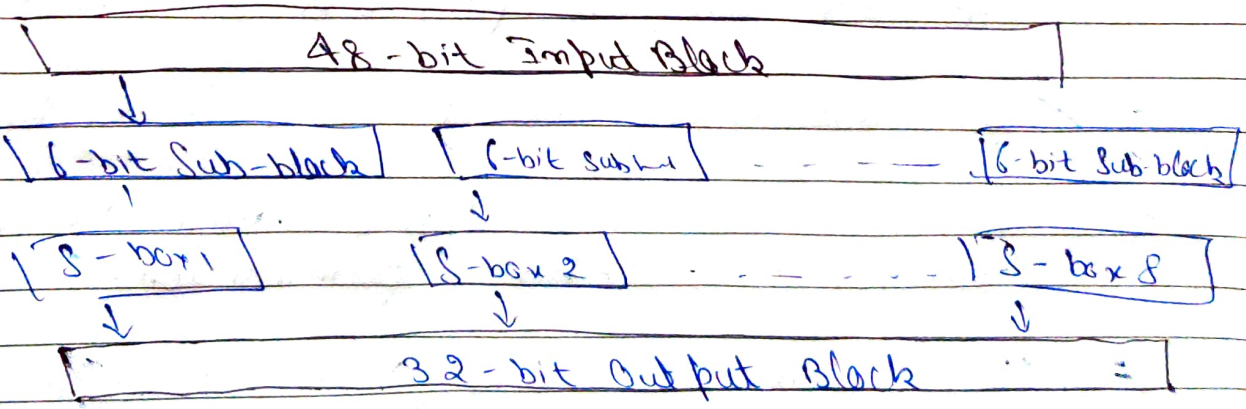
→ It is a process that accepts the 48-bit input from XOR operation involving the compressed key & expanded RPT, & produces the a 32 bit output using the substitution technique.

→ The substitution is performed by 8 substitution boxes called as S-Boxes.

→ Each of the eight S-boxes has a 6-bit input & a 4-bit output.

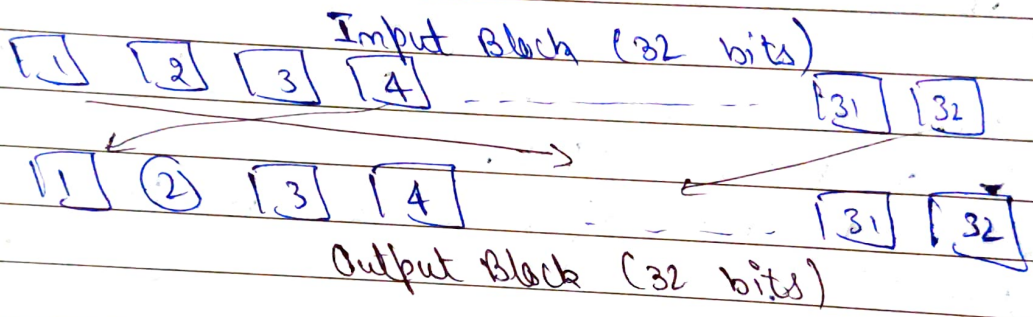
→ The 48-bit input block is divided into 8 sub-blocks & each block is given to a S-box.

→ The output of all the S-boxes are then combined to form a 32-bit block, which is given to the next stage of a round, the P-box permutation.



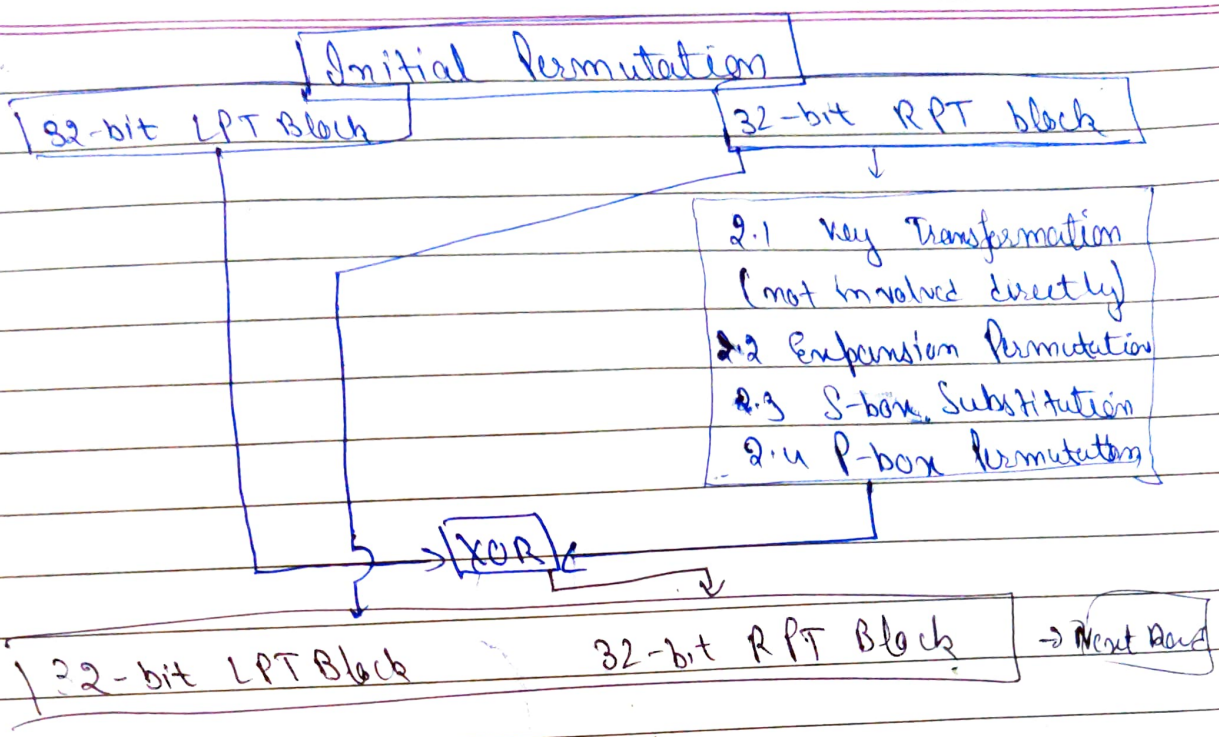
§ 2.4 P-box permutation

- The output of S-box ~~consist~~ consists of 32 bits, these are permuted using a P-box.
- This involves simple permutation
- This is called as P-box permutation.



§ 2.5 XOR & Swap

- All these operations are performed only on the 32-bit right half portion of the 64-bit original plain text (i.e. on the RPT)
- The left-half i.e LPT was untouched.
- The LPT is now XORed with the output produced by P-box permutation.
- The result of this XOR operation becomes the new left half (i.e LPT) in a process of swapping.



Step-3 Final Permutation (FP)

→ At the end of the 16 rounds, the final permutation is performed (only once).

→ This is a simple transposition.

→ The output of final permutation is the 64-bit encrypted block.