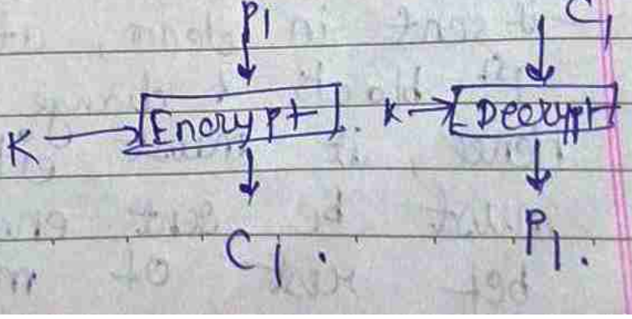
Module 2

* modes of Data Operation :-

* Block Cipher :- is a type of symmetric encryption algorithm that processes data in fixed-size blocks or chunks.
Symmetric encryp: - use same key for both encryption and decryption.
However, real world data does not always neatly fit into these fixed-size blocks, & you need to process data streams or data that isn't a perfect multiple of block size. Hence, diff modes of operation are used:-

1] Electronic CodeBook Mode (ECB):- Simplest mode. Each block of plaintext is encrypted seperately into ciphertext using the same key.
Pros:- Simple & fast. Since each block is processed seperately / independently, its easy to parallelize.
Cons:- Identical plaintext blocks are encrypted into identical ciphertext blocks, which can reveal patterns in data. This makes ECB unsuitable for many applications.

$C_i = DES_{K_i}(P_i)$ ← Plain text (original)
↑
Ciphertext
(Encrypted)

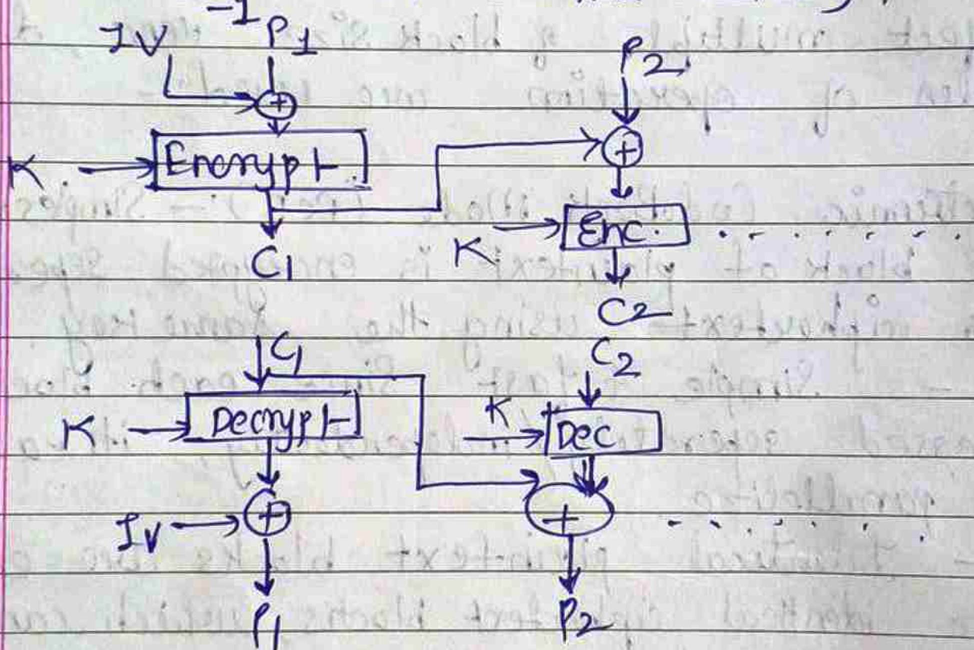


2] Cipher Block Chaining :— (CBC)

- message is broken into blocks.
- Blocks are linked together in encryption operation.
- Each block of plaintext is XoRed with the previous ciphertext block before being encrypted. 1st block is XoRed with an initial vector.

$$C_i = DES_{K1}(P_i \text{ XOR } C_{i-1})$$

C_{-1} = IV (initial vector).



Adv.- Better than ECB bec identical plaintext blocks will result in diff ciphertext blocks

Dis :- 1) Cannot be parallelized.

2) need initial vector (IV) which must be known to sender & receiver

- if sent in clear, attacker can change bits of 1st block & change IV to compensate. Hence, it must either be a fixed value or must be sent encrypted in ECB mode bef rest of message.

* Message Padding :-
At end of message, if last block is short than blocksize of cipher, pad it either with known nondata value (Eg nulls) or pad last block along with count of pad size.

Eg :- 3 data only bytes & size of 8 :-

$$[\, b_1\ b_2\ b_3\ \underbrace{0\ 0\ 0\ 0\ 5}_{pad}\,]$$

5 → pad count

3] Cipher Feedback (CFB) mode :-

*• Stream cipher modes → Though not block cipher modes, stream cipher encrypt data bit or byte at a time. They are designed to handle data that may not fit nearly into block sizes and can be useful for appl^n requiring continuous data encryption.

In CFB, message is treated as a stream of bits.

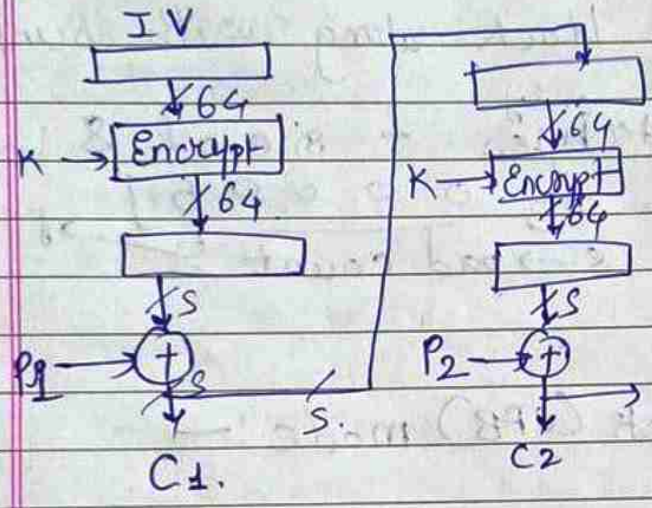
Initializ^n :- (IV) is used to start encryp. process. IV is encrypted with block cipher to produce initial "ciphertext block" or keystream block.
feedback :- Then for each subsequent bit/byte of plaintext, prev ciphertext is used as i/p to block cipher to generate next keystream.

then :- Plaintext is XORed with keystream Block.

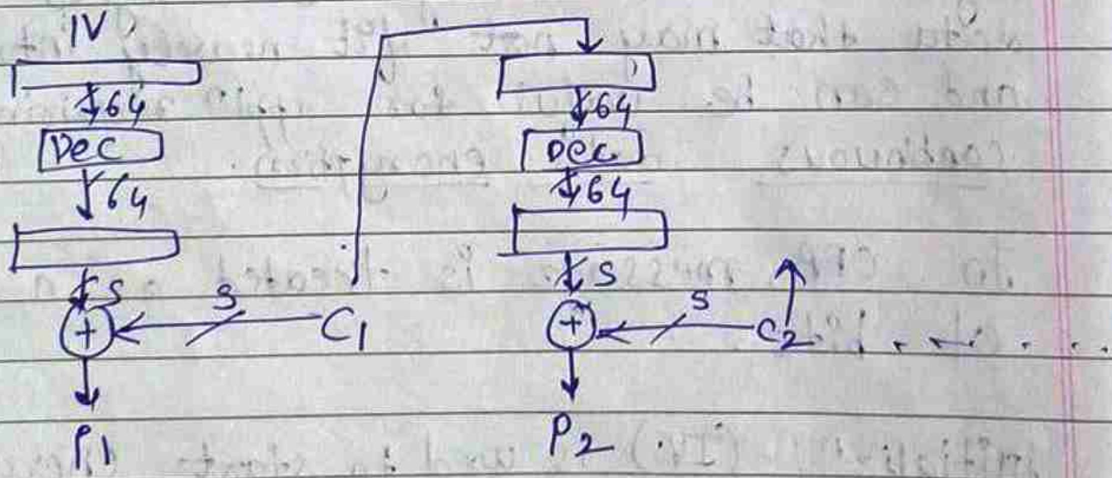
Standard allows any no. of bits. (1, 8, 64, etc).
CFB-1, CFB-8, CFB-64, etc.

$$C_i = P_i \text{ XOR } D_{ES_{K1}}(C_{i-1}).$$

$$C_{-1} = IV.$$



Decrypt : —



Adv: — Flexible: Suited when data arrives in bits or bytes. Errors in ciphertext only affect the corresponding bit/byte of plaintext, minimizing impact of errors. But if not manage properly, may propagate to several blocks.
disadv: — to produce keystream, CFB must wait for encryp of prev block to complete. ∴ Non-parrallelizable.

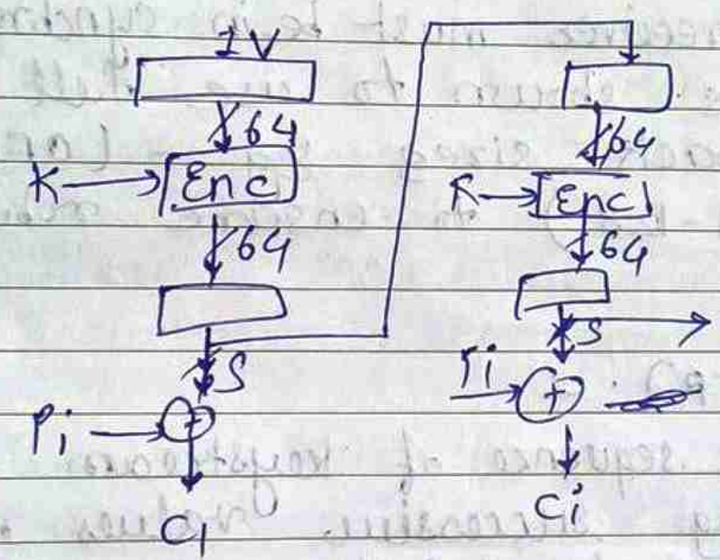
4] Output feedback Mode (OFB):-

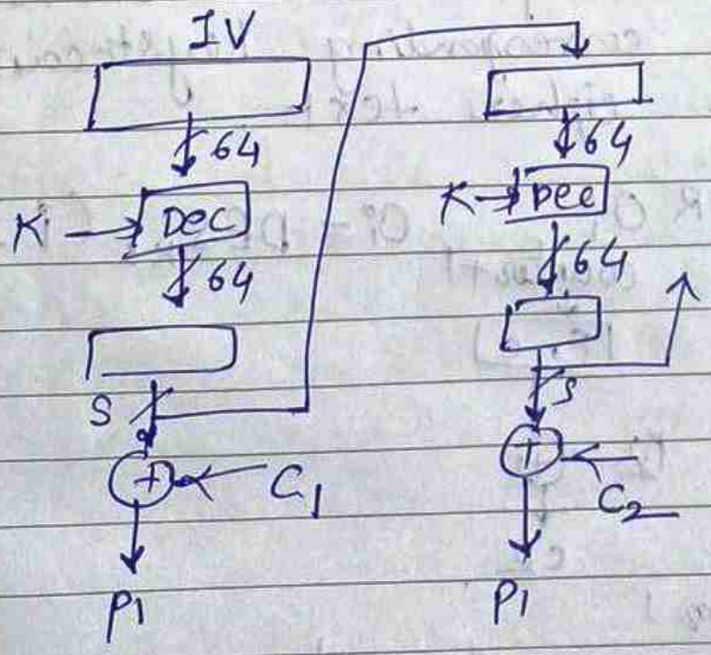
- Stream of bits
- output of cipher is added to message o/p is then feedback. This feedback mechanism creatis a keystream that is XORed with the plaintext to produce ciphertext.

$$C_i = P_i \text{ XOR } O_i$$

$$O_i = DES_{K_1}(O_{i-1})$$

$$O_{-1} = IV.$$





Adv:-

1) No error prop:- OFB doesnot propagate errors. If a bit in ciphertext is corrupted, only the corresponding bit in plaintext is affected, leaving subsequent bits unaffected

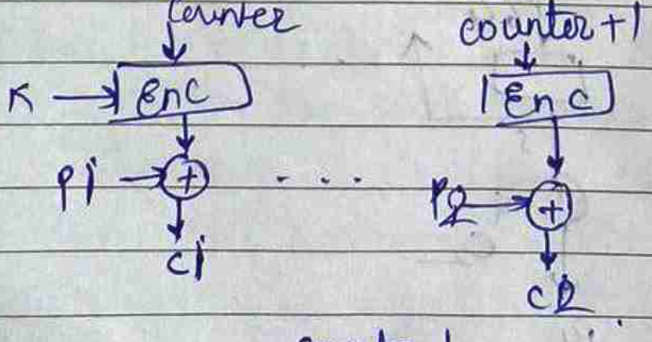
Dis adv:- increased vulnerability to message stream modification.

- must never reuse same sequence (Key + IV).
- sender & receiver must be in synchronization
- Research has shown to use full block feedback sizes Eg:- (OFB-64 or OFB-128) to ensure robust security.

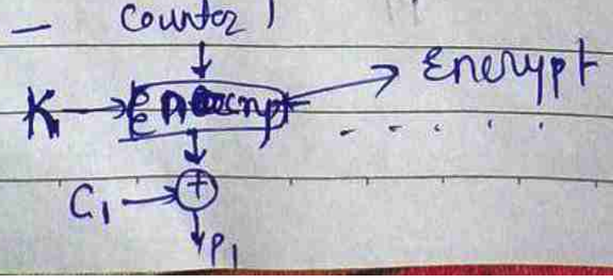
5] Counter (CTR):-

Generates a sequence of keystream blocks by encrypting successive values of a counter. Each plaintext block is then XORed with corresponding keystream blocks to produce cipher text.

$C_i = P_i \text{ XOR } O_i \quad O_i = DES_{K1}(i)$.



Dec:-



Adv:—

1) Efficiency: Unlike other chaining modes can be done in parallel.
2) Due to: parallel execution, processors that support parallel features, such as aggressive pipelining, large no. of registers, etc can be utilized effectively
3) Execution of CTR does not depend on I/p of P or C, so with sufficient memory & security, preprocessing can be used to prepare o/p of encryption boxes that feed into XOR functions.
4) Random access → ith block of P or C can be processed in random fashion
5) More secure
6) Requires only encryption algo. & not decryp.

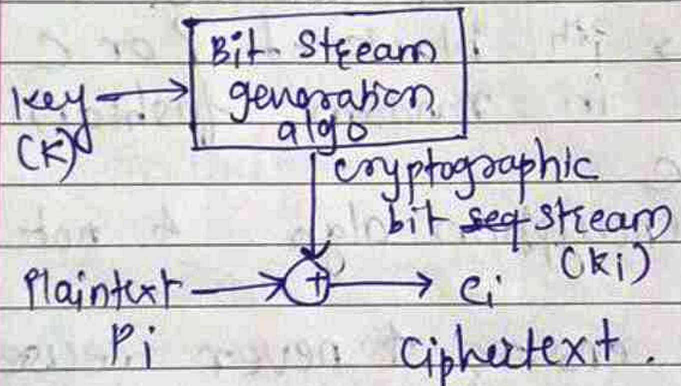
Disadv:— Must ensure & never reuse key/counter values, otherwise could break.

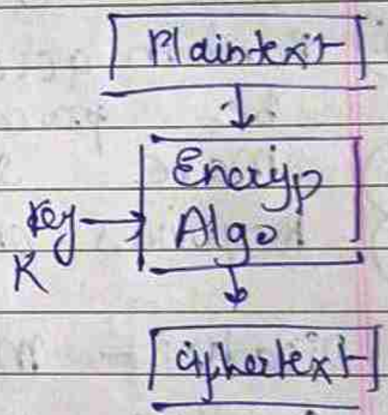
Security	Mode	CIA?	Encrypt Parallel	Decryp Parallel	Random access
low	ECB	CIA/CAI	yes.	yes	yes
High	CBC	CAI	No	yes	no
high	CFB	ACI	No	No	no
High	OFB	AIC/ACI	No	NO	no.
medium	CTR.	CIA/CAI	yes	yes	yes

Notes on CIA:-

- Confidentiality :- All modes prioritize confidentiality by design.
- Integrity :- Some modes implicitly provide integrity by chaining methods (Eg CBC, CFB)
- Availability :- ECB + CTR are more aligned with availability because of their parallelism and random access features.

Stream Cipher:- Block cipher


Bit Stream generation algo
key (K)
cryptographic bit ~~seq~~ stream (ki)
Plaintext
Pi
ci
Ciphertext.



- Most symmetric block ciphers indeed use a Feistel network structure. Key idea is that same structure can be used for both encryption & decryption.

- Claude Shannon's Substitution - Permutation (S-P) networks.

- Substitution :- this operation replaces bits/grp of bits with other bits according to a substitution table. It introduces confusion by making relationship betn plaintext + ciphertext less direct.

- Permutation (P-Box): This operation rearranges bits of data. It introduces diffusion, spreading the influence of individual plaintext bits across many ciphertext bits.

Combining this 2 approaches, can give us secure encryption algorithm.

- diffusion :- dissipates statistical structure of plaintext over bulk of ciphertext.
- Confusion - makes relationship betn ciphertext + key as complex as possible.

* Feistel Cipher :- Built on Shannon's ideas, Feistel ciphers use a specific structure that processes data in rounds with substitutions + peremutations.

- Block partitioning: I/p block of data is divided into 2 halves, left + right (L + R).
- Round functn: In each round :-

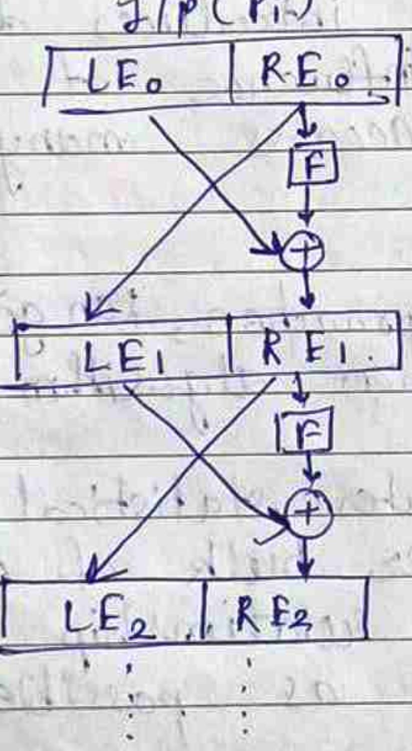
1. Substitution: functn F is applied to R along with round subkey. (Confusion).

Page No.
Date.

2. XOR operⁿ : result of f is XORed with L.
this is diffusion.

3. Swap: halves are swapped, so modified left half becomes right half for next round, & vice versa.

Fiestal structure is invertible, meaning same algo can be used for both enc & dec (just in reverse order).



- Fiestel Cipher Design Elements:—

1] Block size → size of data block that cipher processes at once. size ↑ → security ↑ → performance ↓

2] Key size: length of encryption key. Large key provides more security, but req more computation.

3] No. of Rounds: no. of iterations algo performs. more rounds enhance security but may affect performance.

Page No.
Date.

4] Subkey Generation Algo: used to generate round keys from main key

5] Round function: It must be designed to provide strong confusion and diffusion.

6] fast Software Encryp/Decryp :- should perform efficiently in s/w implementations.

7] Ease of analysis: The design should be robust against attacks & should be well understood to ensure its security.

** Just for clarity:—

- A fiestel cipher is a design pattern for block ciphers.
- A block cipher (like DES or AES) uses specific algo's to encrypt data in fixed-size blocks.
- modes of operation (like ECB, CBC) determine how to apply block ciphers to data of varying lengths & provide additional functionalities.

* DES Algorithm :—

DES (Data encryption Standard) algorithm is a widely used symmetric-key block cipher that encrypts data in 64-bit blocks using a 56-bit key.

Block size: 64 bits
Key size: 56 bits (+ 8 parity bits for total of 64)
Rounds: 16 rounds of encryption.
Steps:—

1. Initial Permutation: (IP)
IP is defined by a fixed permutation table. Permutation reorders bits of plaintext block. doesn't involve any encryption. Eg: If IP specifies that 1st bit of o/p comes from 58th bit of i/p, then 1st bit is taken from 58th bit & plaintext. Even bits to left half, odd bits to right half.

2. Key scheduling :- 56 bit Key is used to generate 16 subkeys
Key is permuted first. then divided into 2 halves. Each half is shifted left by certain numbers. then each block is combined again.

3. Round function:—
Split 64 bit data block into 2 halves (L&R)
- functn F is applied to right half & subkey for round:
Expands R to 48 bits using expansion permutation, & is XORed with 48 bit round key.

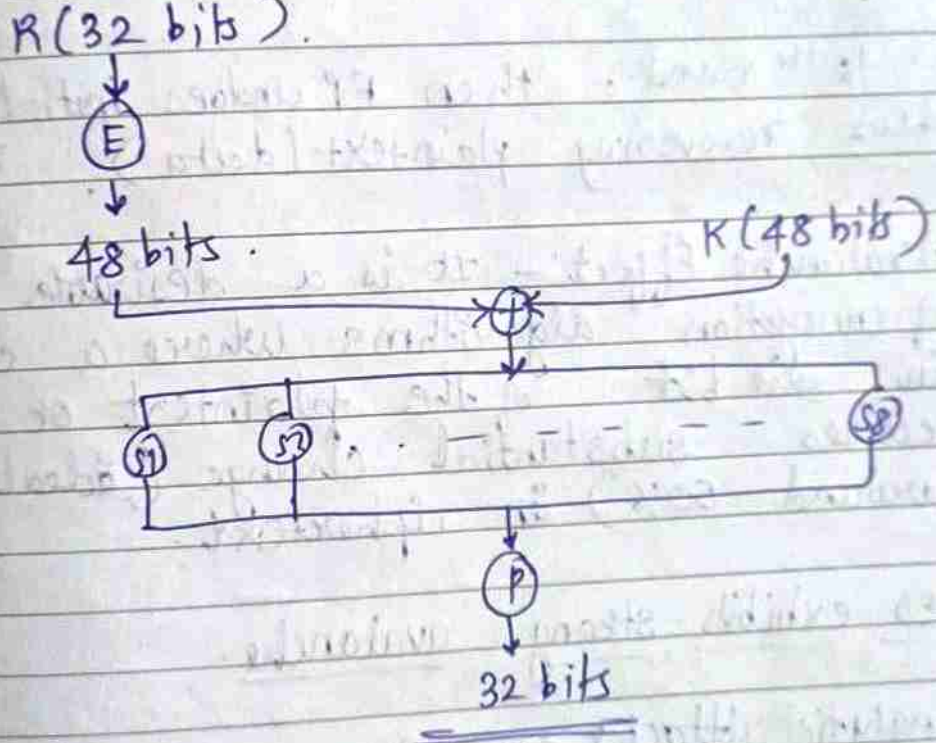
The result is divided into 8 blocks, each passed through substitution boxes (S-box) to produce o/p. 32 bit result is again permuted.

o/p of functn is XoRed with L & result becomes new right half for next round

$L_i = R_{i-1}$

$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$

DES round Structure



After all 16 rounds, final permutation is applied to concatenated L & R halves to produce cipher text.

- Substitution boxes: have 8 S-boxes which map 6 to 4 bits. Outer bits 1 & 6 (row bits) select one row of 4. inner bits 2-5 (col bits) are substituted.

Page No.
Date.

roco selection depends upon both data & Key ~~x~~ this is known as autoclaving (autokeying).

Decryption:-

- To decrypt, reverse the steps of encryption.
- with feistel design, do encryption steps again using subkeys in reverse order.
  - IP undoes final FP step of encryption
  - 1st round

  ⋮

  16th round. then FP undoes initial encryp. IP.
thus recovering plaintext (data).

- Avalanche Effect:- It is a desirable property of encryption algorithms where a change in one bit of the plaintext or key causes substantial change (ideally around 50%) in ciphertext.

DES exhibits strong avalanche.

* Analytic attacks on DES:-

1. Differential Cryptanalysis:- Analyzes how differences in plaintext pairs propagate through encryption process to effect diff in ciphertext pairs.
By compairing pairs of plaintext with known diff & their ciphertext, can deduce info about subkey.

2. Linear Cryptanalysis: ~~to~~ uses linear approximations to describe behavior of cipher. Goal is to find reln betn plaintext, ciphertext & subkeys looks for linear eqn's.

3. Related Key-attacks: attacker has access to ciphertexts encrypted with diff keys that are related in some specific way.

* Timing attacks:- Exploits variations in the time it takes to perform encryption operations based on i/p values. Corelates timing info with values of subkeys.

* Power Analysis Attack:- measures power consumption during encryption to infer info about secret key
* fault injection attack: introduces faults in encryption to obtain erraneous o/ps that can be analyzed to reveal info about key.

* Block cipher design principle :-
1. No. of rounds
2. function f.
3. S-Boxes: introduce nonlinearity & confusion. carefully designed to resist known cryptanalytic attacks. Strong resistance to defferential & linear cryptanalysis.
4. Key schedule to be able to secure against attacks that might exploit weak subkey generation process.

MIM :- encrypting plaintext with all possible keys + storing intermediate results. then decrypted with all keys + store result. When both match, it reveals key.

* Modified version of DES :-

1. Double DES :- apply DES algo twice with 2 diff keys.
   1. Encrypt plaintext using DES with 1st key (K1)
   2. Encrypt o/p from 1st using DES with second key (K2).

$$\begin{tikzcd} P \arrow[r] & \text{Encrypt} \arrow[r] & \text{Temp result} \arrow[r] & \text{Encrypt} \arrow[r] & \text{Cipher} \\ & K1 \arrow[u] & & K2 \arrow[u] & \end{tikzcd}$$

vulnerable to MIM (meet in middle attack). ~~encrypt + store~~ all possible ~~keys to~~

2. Triple DES (3 keys) :- apply DES algo 3 times using 3 keys.

$$\begin{tikzcd} P \arrow[r] & \text{Encrypt} \arrow[r] & \text{Cipher text 1} \arrow[r] & \text{Encrypt} \arrow[r] & \text{CT 2} \arrow[r] & \text{Encrypt} \arrow[r] & \text{final} \\ & K1 \arrow[u] & & K2 \arrow[u] & & K3 \arrow[u] & \end{tikzcd}$$

3. Triple DES with 2 keys :- (Replace K3 with K1)

$$\begin{tikzcd} P \arrow[r] & \text{Encrypt} \arrow[r] & \text{CT1} \arrow[r] & \text{Enc} \arrow[r] & \text{CT2} \arrow[r] & \text{Enc} \arrow[r] & \text{final} \\ & K1 \arrow[u] & & K2 \arrow[u] & & K1 \arrow[u] & \end{tikzcd}$$

Page No.
Date.

* AES (Advanced Encryption Standard):-

- AES is symmetric encryption algo, uses same key for both encryp. + decryp.
- Operates on block size of 128 bits (16 bytes).
- key sizes — 128 bits, or 192 bits, or 256 bits. these keys provide diff level of security and performance.
- an iterative rather than feistel cipher
  - processes data as block of 4 columns of 4 Bytes
  - operates on entire data block for every round
- designed to be :- ~~regi~~ resistant against known attacks
  - speed + code compactness on many CPUs
  - design simplicity.

128 bit key → 10 rounds, 192 → 12 rounds, 256 - 14 rounds
variable :- 9 rounds if both key + block are 128 bit long
- 11 if either of them is 192 bits long + neither is longer than that.
- 13 if either block / key is 256 bit long.

a] Key Expansion :- Starts by expanding key into a set of round keys. generates array of key schedule words from original key.
takes 128 bit key + expand into array of 44/52/60 words ~~[illegible]~~.
(size of key (subkey).

then loop creating words that depend on values in prev 1 + 4 places back.
- ~~1st word in 4 has rotate + S-box + XOR round constant on previous~~
- designed to resist known attack.
- design criteria involve
1] knowing part of / round key insufficient to find remaining

Page No.
Date.

2] invertible transformation
3] fast on wide range of CPUs
4] use round constants to break symmetry
5] diffuse key bits into round keys
6] enough non-linearity to make analysis
7] simplicity of description.

- Main Rounds :- Each round involves 4 steps:- (3 substitutions + 1 peremutation)

1. SubBytes :- Each byte in state is replaced with corresponding byte from predefined substitution table known as S-box. This provides non-linearity to encryption.

2. Shift Rows :- rows are shifted cyclically.
1st row remains unchanged
2nd row is shifted one byte to left
3rd row is shifted two byte to left
4th row is shifted three byte to left.
∴ Provides diffusion.

3. Mix Columns :- substitution that makes use of some arithmetic. Columns are mixed using linear transformations. Each col is transformed to ensure that o/p of one byte depends on values of multiple byte.
∴ using prime poly $m(x) = x^8 + x^4 + x^3 + x + 1$

4. AddRound Key :- Another round key is XORed with the state (data block).

* AES Decryption:-

AES decryption is not identical to encryption. but can define an equivalent inverse cipher with steps as for encryption.

- using inverse of each step
- with a different key schedule.

- works since result is unchanged when
- swap byte substitution & shift rows.
  swap mixed columns & ~~ro~~ add round key again.