INFORMATION SECURITY

UNIT - 1

Symmetric Cipher Model

* Ingrediants of a symmetric encryption scheme:

- Plaintext - original intelligible message or data that is fed into the algorithm as input
- Encryption algorithm - Performs various substitutions & transformations on the plaintext.
- Secret key - It is also input to the encryption algorithm. The algorithm produces a different output depending on the specific key being used at that time. The exact transformations & substitutions performed by the algo. depend on the key.
- Ciphertext - The scrambled message produced as output. It depends on the plaintext & the secret key. The ciphertext, as it stands, is unintelligible. * impossible to understand

- **Decryption algorithm** – This is essentially the encryption algorithm run in reverse. It uses the ciphertext & the secret key to produce the plaintext.

Requirements for secure use of conventional encryption

① The opponent should be unable to decrypt ciphertext or discover the key even if he or she is in possession of a no. of ciphertexts with the plaintext that produced each of them.

② Sender & Receiver must have obtained copies of the secret key in a secure fashion & must keep the key secure.

Symmetric Cryptosystem

Plaintext ⟶ $X = [X_1, X_2, \cdots X_M]$

(The M elements of X are letters in some finite alphabet.)

Key ⟶ $K = [K_1, K_2, \cdots, K_J]$

E ⟶ Encryption algo.

Sender: $Y = E(K, X)$, Y is produced by using encryption algo E as a function of the plaintext X, with the specific function determined by K.

Receiver: $X = D(K, Y)$ ↓ Decryption algo

Cryptographic systems characterization is based on:

① Type of operation used for transforming plaintext to ciphertext

⟶ **Substitution** – Each element in the plaintext is mapped onto another element

⟶ **Transposition** – Elements in the plaintext are rearranged.

The fundamental requirement is that all operations be reversible. Most systems, called product systems, involve multiple stages of substitutions & transpositions.

② Number of keys used

→ Symmetric / Single-key / Secret-key / conventional encryption – Both sender & receiver use the same key.

→ Asymmetric / Two-key / public-key – Sender & Receiver use different keys

③ The way in which the plaintext is processed

→ Block cipher – Processes the input 1 block of elements at a time, producing an output block for each input block.

→ Stream cipher – Processes the input elements continuously, producing output one element at a time, as it goes along

Approaches to attacking a conventional encryption scheme:

* Cryptanalysis : This type of attack exploits the characteristics of the algo. to attempt to deduce a specific plaintext or the key being used.

* Brute-force attack : The attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained.

Types of cryptanalytic attacks (based on the amount of info. known to the cryptanalyst)

(All types include encryption algo & cipher text)

	Type	Info
1.	Ciphertext only	–
2.	Known plaintext	1 or more plaintext – ciphertext pairs
3.	Chosen plaintext	Plaintext msg chosen by the cryptanalyst, & its ciphertext
4.	Chosen ciphertext	Ciphertext " ", & its decrypted plaintext
5.	Chosen Text	(chosen plaintext + chosen ciphertext)

* Computationally secure - An encryption scheme is said to be " " if either of the following 2 criteria are met:

① The cost of breaking the cipher exceeds the value of the encrypted information

② The time required to break the cipher exceeds the useful lifetime of the info.

* Unconditionally secure - " " "
if the ciphertext generated by the scheme does not contain enough info. to determine uniquely the corresponding plaintext.

* SUBSTITUTION TECHNIQUES

① Caesar Cipher

Involves replacing each letter of the alphabet with the letter standing a particular number (shift) of places further down the alphabet

plain: a b c d e f g h i j k l m n o p q r
cipher: D E F G H I J K L M N O P Q R

plain: s t u v w x y z
cipher: S T U V W X Y Z

plain: A B C
cipher: A B C

Example (shift = 3):

plain: meet me later
cipher: PHHW PH ODWHU

∴ GENERAL Caesars algorithm:

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

Decryption algorithm:

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

* Reasons you can use a brute-force cryptanalysis:

1. Encryp. & decryp. algorithms are known
2. There are only 25 keys to try
3. The language of the plaintext is known & easily recognizable.

② Monoalphabetic Ciphers

The cipher line can be any permutation of the 26 alphabetic characters, which means there are 26! possible keys.

→ Brute-fore techniques for cryptanalysis are ~~etc.~~ eliminated.

→ Another line of attack:

If the cryptanalyst knows the nature of the plaintext (eg. noncompressed English text), the analyst can exploit the regularities of the language.

Example → letters 'E' & 'T' are the most frequent letters in ~~E~~English text.
∴ The 2 most frequent letters in the ciphertext can be assumed to be 'e' & 't'

* Frequency of two-letter combinations, known as digrams.

Example → The most common digram is 'th'.

③ Playfair Cipher

- Multiple-letter encryption cipher
- treats digrams in the plaintext as single units & translates these units into ciphertext digrams.
- 5 × 5 matrix, constructed using a keyword.
- Fill the letters of the keyword (minus duplicates) from left to right & top to bottom.
- Fill the remainder of the matrix with the remaining letters in alphabetic order.
- letters I & J count as one letter.

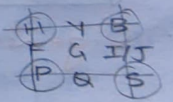
Example: keyword → MONARCHY

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

- ba ll oo n ✗
  ba lx lo on ✓
- (same row, right)
  ar → RM
- (same column, below)
  mu → CM
- normally, (row + column)
  hs → BP
  ea → IM (or JM)

- n ✗
  nx ✓



④ Hill Cipher

The Hill Algo : This algo takes m successive plaintext letters & substitutes them for m ciphertext letters.

$$C = E(K, P) = PK \bmod 26$$

Example, m = 3

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

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

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

⑤ Polyalphabetic Ciphers

* Using different monoalphabetic substitutions as one proceeds through the plaintext.

→ Vigenère cipher

Vigenère tabeau:

Key ↓ \ Plaintext →	a	b	c	d	...	z
a	A	B	C	D	...	A
b	B	C	D	E	...	B
c	C	D	E	F		
⋮	⋮	⋮	⋮	⋮		⋮
z	Z	A	B	C		Y

Example

key : deceptive

key: d e c e p t i v e d e c e p t
plaintext: w e a r e d i s c o v e r e d
ciphertext: Z I C V T W Q N G R Z G V T W

→ Vernam Cipher

- choosing a keyword that is as long as the plaintext & has no statistical relationship to it.
- works on binary data (bits) rather than letters.

$$C_i = P_i \oplus K_i$$

Decryption:

$$P_i = C_i \oplus K_i$$

⑥ One-Time Pad

- Using a random key that is as long as the message, so that the key need not be repeated.
- The key is used to encrypt & decrypt a single message, & then is discarded.
- Exhibits perfect secrecy.

* TRANSPOSITION TECHNIQUES

↓

Peforming some sort of permutation on the plaintext letters.

Rail fence technique

Plaintext is written down as a sequence of diagnols & then read off as a sequence of rows.

Example: To encipher "meet me after the toga party".
Rail fence depth of 2

m e m a f t e r t h e t o g a P a r t y
e t e e t e h o a a r y

Encrypted message:
MEMATRHTGPRYETEFETEOAAT

A more complex scheme:

- write the message row by row, & read it off column by column, but permute the order of the columns.
- Key → order of the columns

Example:

Key → 4312567

Key: 4 3 1 2 5 6 7
Plaintext: a t t a c k p
o s t p o n e
d u n t i l t
w o a m x y z

Ciphertext: TTNAAPTMTSUOAODWCOIX
KNLYPETZ

ROTOR MACHINES

* Multiple stages on encryption can produce an algo that is significantly more difficult to cryptanalyze.
* Before DES, the most important application of this principle was a class of systems known as rotor machines.
* The machine consists of a set of independently rotating cylinders through which electrical pulses can flow.
* Each cylinder has 26 input pins & 26 output pins, with internal wiring that connects each i/p pin to a unique o/p pin.

STEGANOGRAPHY

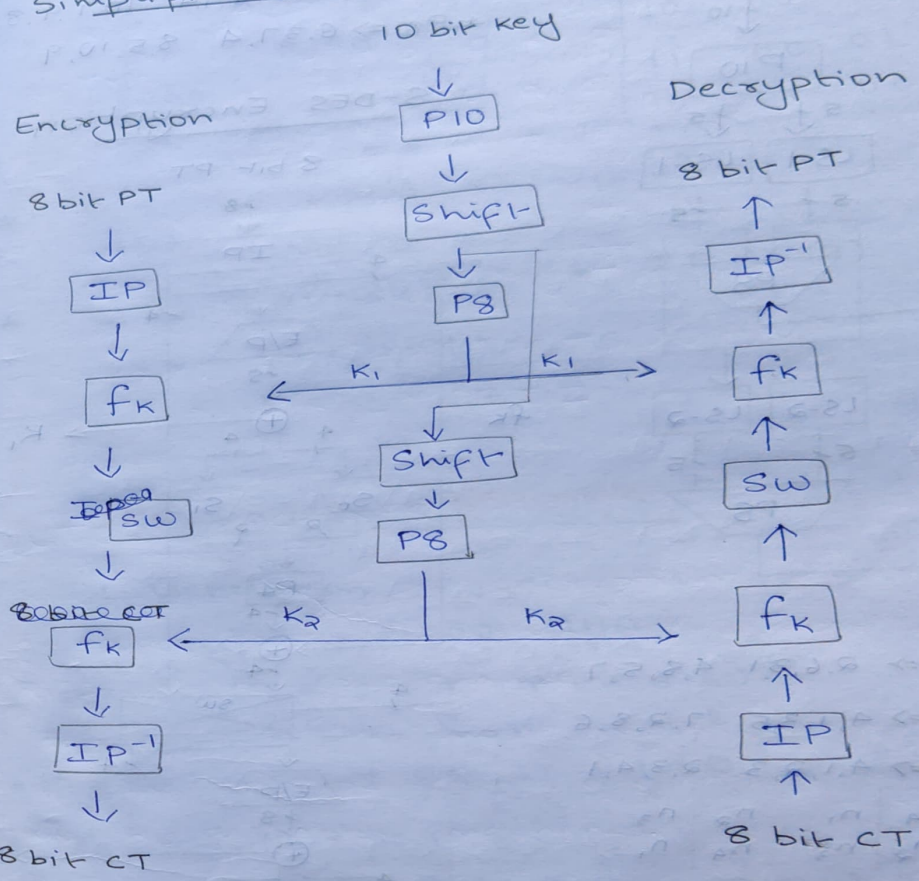
* This conceals the existence of the message whereas cryptography renders the message unintelligible.

Some techniques:

- Character marking - selected letters of the text are over-written in pencil. The marks aren't usually visible unless the paper is held against bright light.
- Invisible ink - Substances used for writing that leave no visible trace until heat or some chemical is applied to the paper.
- Pin punctures - pin punctures on selected letters that are ordinarily not visible unless the paper is held up in front of a light.
- Typewriter correction ribbon - Used b/w lines typed w/ a black ribbon, the results of typing w/ the correction tape that are visible only under a strong light.

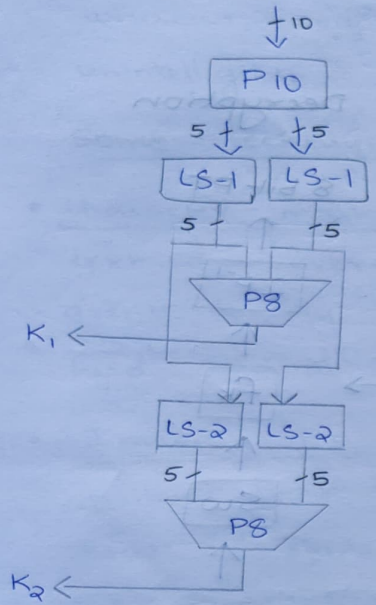
THE DATA ENCRYPTION

Simplified DES



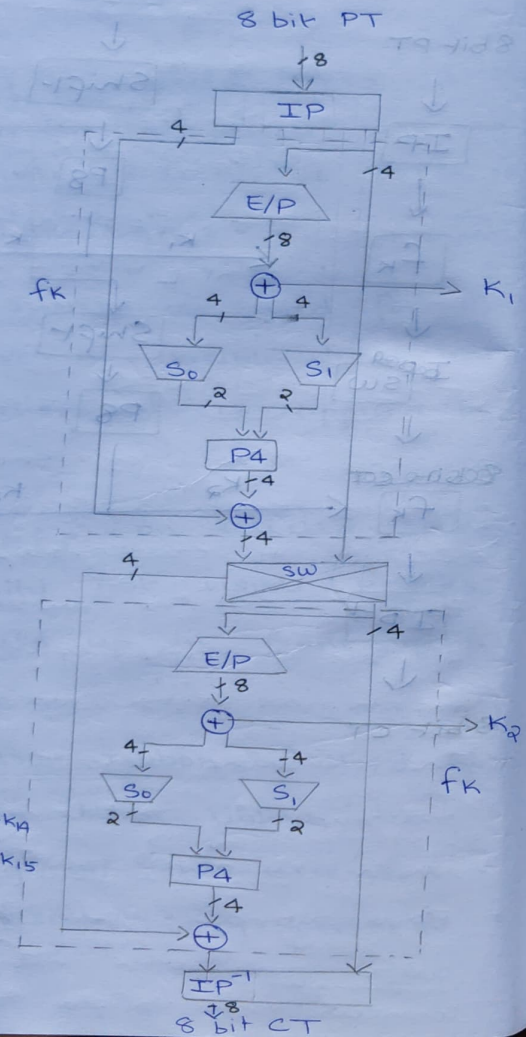
S-DES Key Generation

10 bit key



* P10 => 3,5,2,7,4 10,1,9,8,6
* P8 => 6,3,7,4 8,5,10,9

S-DES Encryption



* IP => 2,6,3,1 4,8,5,7
* IP^{-1} => 4,1,3,5 7,2,8,6
* E/P => 4,1,2,3 2,3,4,1

n_4	n_1	n_2	n_3
n_2	n_3	n_4	n_1

* EXOR with K_1

n_4+K_{11}	n_1+K_{12}	n_2+K_{13}	n_3+K_{14}
n_2+K_{15}	n_3+K_{16}	n_4+K_{17}	n_1+K_{15}

$P_{0,0}$	$P_{0,1}$	$P_{0,2}$	$P_{0,3}$
$P_{1,0}$	$P_{1,1}$	$P_{1,2}$	$P_{1,3}$

* E/P ⊕ K_1 -> S_0 (4b) S_1 (4b)

S_0 => P_{00} P_{03}
S_1 => P_{01} P_{02}

S_0, S_1 => Row : 1 & 4 (bit)
Col : 2 & 3 (bit)

S_0 =

01	00	11	10
11	10	01	00
00	10	01	11
11	01	11	10

S_1 =

00	01	10	11
10	00	01	11
11	00	01	00
10	01	00	11

* P_4 => 2,4,3,1
* S_0 S_1 (4b)
* P_4 ⊕ IP(MSB)
* IP (LSB)

Swap

S-DES sum : Example

① Key Generation

10 bit key -> 1 1 0 0 0 1 1 1 1 0
(1 2 3 4 5 6 7 8 9 10)

* P_{10} -> (3 5 2 7 4 10 1 9 8 6)
  0 0 1 1 0 0 1 1 1 1
* LS_1(1) -> 0 1 1 0 0 1 1 1 1 0
  (1 2 3 4 5 6 7 8 9 10)
* P_8 -> (6 3 7 4 8 5 10 9)
  1 1 1 0 1 0 0 1 -> K_1
* LS_2(2) -> 1 1 0 0 0 1 1 1 0 1 (once)
  (to LS_1) 1 0 0 0 1 1 1 0 1 1 (twice)
  (1 2 3 4 5 6 7 8 9 10)
* P_8 -> (6 3 7 4 8 5 10 9)
  1 0 1 0 0 1 1 1 -> K_2

② Encryption

Plaintext → 0 0 1 0 1 0 0 0
(1 2 3 4 5 6 7 8)

* IP → (2 6 3 1 4 8 5 7)
  0 0 1 0 0 0 1 0
  (1 2 | 3 4)

* E/P → (4 1 2 3 2 3 4 1)
  0 0 0 1 0 1 0 0

* E/P ⊕ K_1 → 0 0 0 1 0 1 0 0
  ⊕ 1 1 1 0 1 0 0 1
  1 1 1 1 1 1 0 1
  (S_0 S_1)

* S_0 → 1 1 1 1 → Row : 11 = 3
  Col : 11 = 3 } 10

  S_1 → 1 1 0 1 → Row : 11 = 3
  Col : 10 = 2 } 00

  ∴ $S_0 S_1$ → 1 0 0 0 (S-box)
  (1 2 3 4)

* P_4 → (2 4 3 1)
  0 0 0 1

* P_4 ⊕ IP (MSB) → 0 0 0 1
  ⊕ 0 0 1 0
  0 0 1 1

* IP (LSB) → 0 0 1 0

* swap → 0 0 1 0 0 0 1 1
  (1 2 | 3 4)

* E/P → (4 1 2 3 2 3 4 1)
  1 0 0 1 0 1 1 0

* E/P ⊕ K_2 → 1 0 0 1 0 1 1 0
  ⊕ 1 0 1 0 0 1 1 1
  0 0 1 1 0 0 0 1
  (S_0 S_1)

* S_0 → 0 0 1 1 → Row : 01 = 1
  Col : 01 = 1 } 10

  S_1 → 0 0 0 1 → Row : 01 = 1
  Col : 00 = 0 } 10

  ∴ $S_0 S_1$ → 1 0 1 0 (S-box)

* P_4 → (2 4 3 1)
  0 0 1 1

* P_4 ⊕ swap → 0 0 1 1
  ⊕ 0 0 1 0
  0 0 0 1
  (1 2 3 4)

* swap (LSB) → 0 0 1 1
  (5 6 7 8)

  &

* IP^{-1} → (4 1 3 5 7 2 8 6)
  1 0 0 0 1 0 1 0

Answer

DIFFERENTIAL CRYPTANALYSIS

* The rationale behind diff. cryp. is to observe the behavior of pairs of text blocks evolving along each round of the cipher, instead of observing the evolution of a single text block.

* Consider the original plaintext block m to consist of 2 halves m_0, m_1.
* Each round of DES maps the right-hand i/p into the left-hand o/p & sets the right-hand o/p to be a function of the left-hand i/p & the subkey for this round.

* At each round, 1 new 32-bit block is created. If each new block is labelled m_i $(2 \le i \le 17)$, then the intermediate msg halves are related:

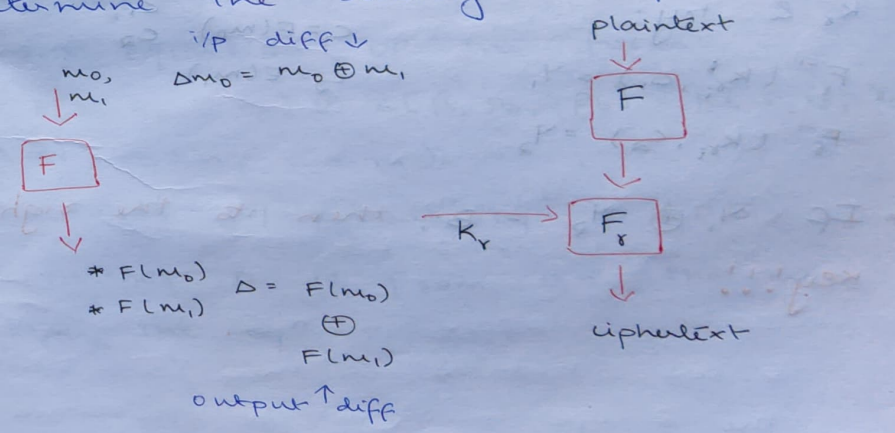
$$m_{i+1} = m_{i-1} \oplus f(m_i, K_i)$$

* we start w/ 2 msgs → m & m' w/ a known XOR difference

$\Delta m = m \oplus m'$, ∴ for intermediate msg halves $\Delta m_i = m_i \oplus m'_i$

$$\therefore \Delta m_{i+1} = m_{i+1} \oplus m'_{i+1}$$
$$= [m_{i-1} \oplus f(m_i, K_i)] \oplus [m'_{i-1} \oplus f(m'_i, K_i)]$$
$$= \Delta m_{i-1} \oplus [f(m_i, K_i) \oplus f(m'_i, K_i)]$$

* Suppose that many pairs of i/ps to f w/ the same difference yield the same o/p difference if the same subkey is used.
* Therefore, if we know Δm_{i-1} & Δm_i w/ high probability, then we know Δm_{i+1} w/ high ".
* If a number of such differences are determined, it is feasible to determine the subkey in f.



* Differential crypt. attack:

> We're toying to find the key! K_r

Differential Attack (chosen plaintext attack)

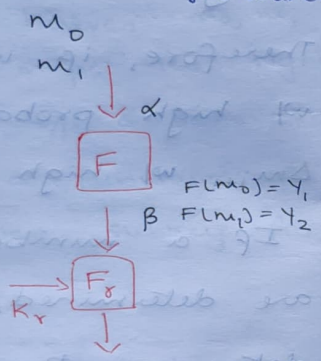
we know α → i/p difference

x → some plaintext

r → rounds

$m_0 = x$

$m_1 = x \oplus \alpha$



we know β → o/p diff

$Y_1 \oplus Y_2 = \beta$

$C_1 = E(m_0)$

$C_2 = E(m_1)$

* Assume a value for K_r & toy:

$F_r^{-1}(K_{r_i}, C_1) = Y_1$

$F_r^{-1}(K_{r_i}, C_2) = Y_2$

* If $Y_1 \oplus Y_2 = \beta$, then its the right key!!!

LINEAR CRYPTANALYSIS

* It is based on finding linear approximations to describe the transformation performed in DES.

* This method can find a DES key, given 2^{43} known plaintexts.

* For a cipher w/ n-bit plaintext & ciphertext blocks & an m-bit key, let the plaintext block be labeled $P[1], \dots P[n]$, the ciphertext block $C[1], \dots C[n]$, & the key $K[1], \dots K[m]$, then define

$$A[i, j, \dots k] = A[i] \oplus A[j] \oplus \dots \oplus A[k]$$

* The objective is to find an effective linear equation of the form:

$$P[\alpha_1, \alpha_2, \dots \alpha_a] \oplus C[\beta_1, \beta_2, \dots \beta_b] = K[\gamma_1, \gamma_2, \dots \gamma_c]$$

$(a \geq 1;\ b \leq n;\ c \leq m)$

that holds w/ probability $p \neq 0.5$.

The further p is from 0.5, the more effective the eqn.

BLOCK CIPHER DESIGN PRINCIPLES

DES Design Criteria

This criteria focused on the design of the S-boxes & on the P function that that takes the output of the S-boxes.

Criteria for the S-boxes:

→ No o/p bit of any S-box should be too close a linear function of the i/p bits.

→ Each row of an S-box should include all 16 possible o/p bit combinations.

→ If 2 i/ps to an S-box differ by/in exactly 1 bit, the o/ps must differ in at least 2 bits.

→ If " " differ in the 2 middle bits exactly, the o/ps must differ in at least 2 bits

→ If " " differ in their 1st 2 bits & are identical in their last 2 bits, the 2 o/ps must not be the same.

→ For any non-zero 6-bit difference b/w i/ps, no more than 8 of the 32 pairs of i/ps exhibiting that difference may result in the same o/p difference.

→ Similar to the above criterion, but for the case of 3 S-boxes.

Criteria for the permutation P

→ The 4 o/p bits from each S-box at round i are distributed so that 2 of them affect end bits & the other 2 affect "middle bits" of round (i+1).

→ The 4 o/p bits from each S-box affect 6 different S-boxes on the next round, & no 2 affect the same S-box

→ For 2 S-boxes j, k, if an o/p bit from S_j affects a middle bit of S_k on the next round, then an o/p bit from S_k cannot affect a middle bit of S_j.

Critical aspects of block cipher design

① NUMBER OF ROUNDS :

// The cryptographic strength of a Feistel cipher derives from 3 aspects of the design → No. of rounds, function F, & the key schedule algo. //

* The greater the no. of rounds, the more difficult it is to perform cryptanalysis, even for a relatively weak F.

* In general, the criterion should be that the no. of rounds is chosen so that known cryptanalytic efforts require greater effort than a simple brute-force key search attack.

② Design of Function F :

This function relies on the use of S-boxes.

Design Criteria for F

* This function F provides the element of confusion in a Feistel cipher.
* It must be difficult to "unscramble" the substitution performed by F.
* F has to be non linear.
* A change in 1 bit of the i/p should produce a change in many bits of the o/p (Avalanche effect).
* Strict Avalanche effect – Any o/p bit j of an S-box should change w/ probability 1/2 when any single i/p bit i is inverted for all i, j.
* Bit independence criterion (BIC) – O/p bits j & k should change independently when any single i/p bit i is inverted for all i, j, & k.

③ S-box Design

* We would like any change to the i/p vector to an S-box to result in random-looking changes to the o/p.
* The relationship should be nonlinear & difficult to approximate with linear functions.
* "Size" Larger S-boxes are more resistant to differential & linear cryptanalysis.
* Larger the S-box, the more difficult it is to design it properly.

Criteria

* S-box should satisfy both SAC & BIC.
* All linear combinations of S-box columns should be bent.
  Bent functions are a special class of Boolean functions that are highly nonlinear according to certain mathematical criteria.

* Gauranteed Avalanche (GA) - An S-box satisfies GA of order γ if, for a 1-bit i/p change, at least γ output bits change.

Methods of selecting S-box entries:

- Random - Use pseudorandom number generation or some table of random digits to generate the entries.
- Random w/ testing - Choose entries randomly, then test the results against various criteria, & throw away those that don't pass.
- Human-made - Manual approach w/ only simple maths to support it.
- Math-made - Generate acc. to mathematical principles.

④ Key Schedule Algo :

* At minimum, the key schedule should guarantee key/ciphertext Strict Avalanche Criterion & Bit Independence Criterion.

BLOCK CIPHER MODES OF OPERATION

① ELECTRONIC CODE BOOK

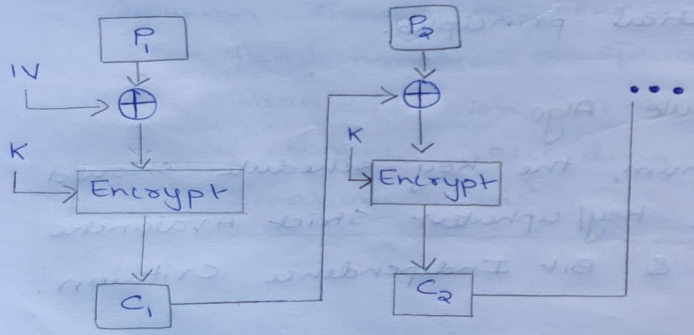
* The simplest mode, in which plaintext is handled one block at a time & each block of pt. is encrypted using the same key.

② CIPHER BLOCK CHAINING

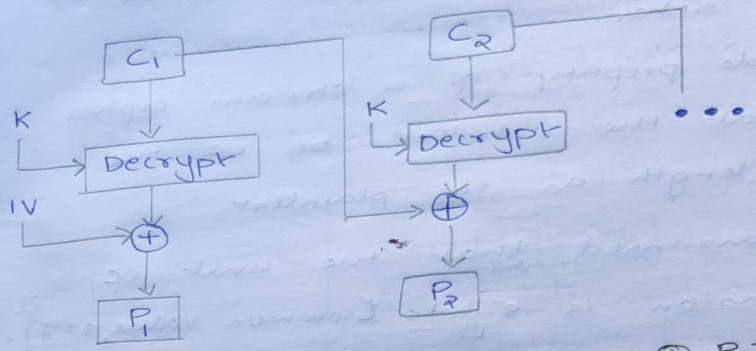
* The i/p to the encryption algo is the XOR of the current pt. block & the preceding ciphertext block; the same key is used for each block.
* The i/p to the encryption function for each pt. block bears no fixed relationship to the pt. block.

$$C_j = E(K, [C_{j-1} \oplus P_j])$$

Encryption



Decryption



$$D(K, C_j) = D(K, E[K, [C_{j-1} \oplus P_j]])$$

* To produce the 1st block of ciphertext, an initialization vector (IV) is XORed w/ the 1st block of plaintext.
* The IV is a data block that is the same size as the cipher block.

$C_1 = E(K, [P_1 \oplus IV])$	$P_1 = D(K, C_1) \oplus IV$
$C_j = E(K, [P_j \oplus C_{j-1}])$	$P_j = D(K, C_j) \oplus C_{j-1}$

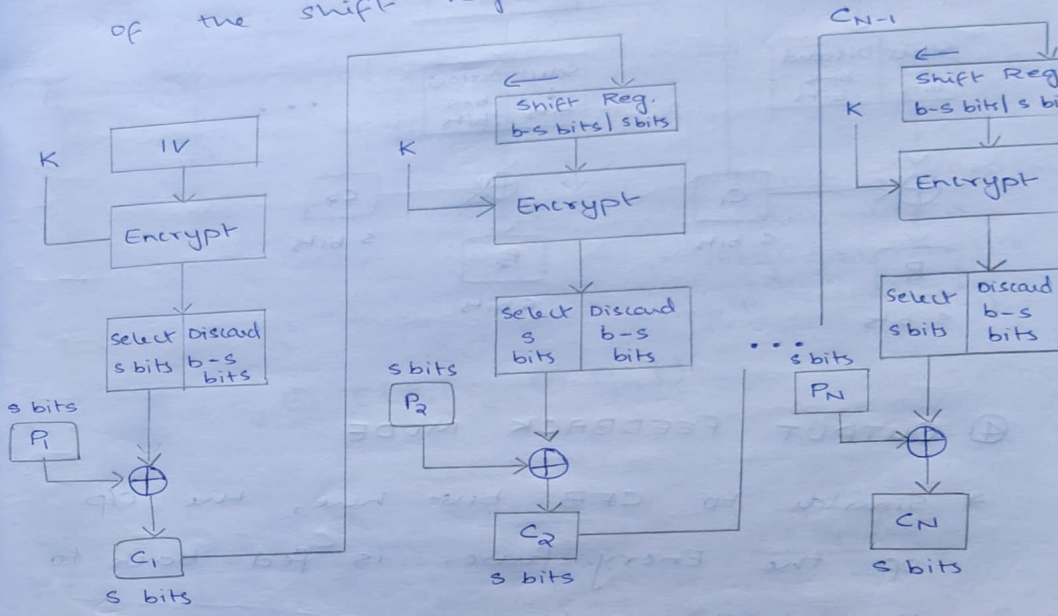
* The IV should be known to the sender & receiver.

③ CIPHER FEEDBACK MODE

* Stream cipher.
* Desirable property of a stream cipher is that the ciphertext be of the same length as the plaintext.
* It is assumed that the unit of transmission is s bits (common value, s=8).
* As w/ CBC, the units of plaintext are chained together, so that the ciphertext of any plaintext unit is a function of all the preceding plaintext.
* The plaintext is divided into segments of s bits.

Encryption

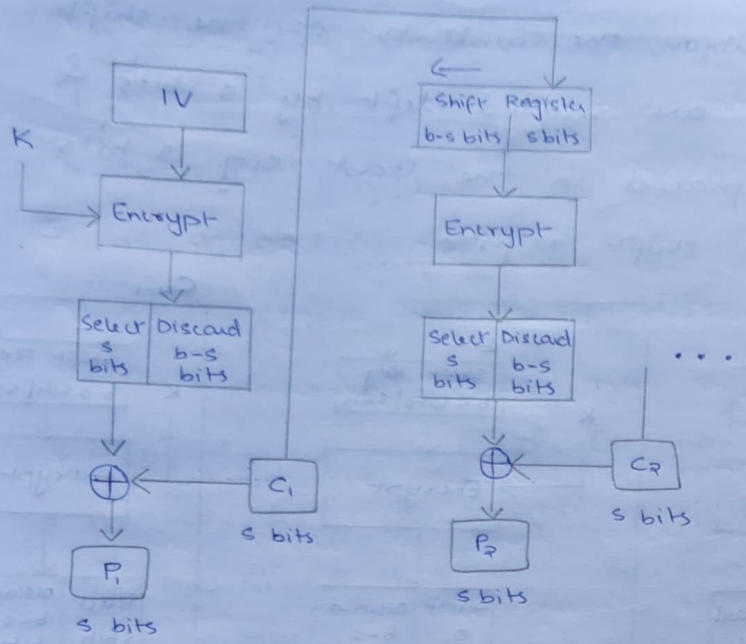
* The i/p to the encry. function is a b-bit shift register that is initialized to some IV.
* The most sign. s bits of the o/p of the EF are XORed w/ the 1st segment of plaintext P_1 to produce the 1st unit of ciphertext C_1.
* In addition, the contents of the shift register are shifted left by s bits & C_1 is placed in the least sig. s bits of the shift register.



Decryption

* Received ciphertext unit is XORed w/ the o/p of the encryption function.

$$C_1 = P_1 \oplus MSB_s[E(K, IV)]$$
$$P_1 = C_1 \oplus MSB_s[E(K, IV)]$$

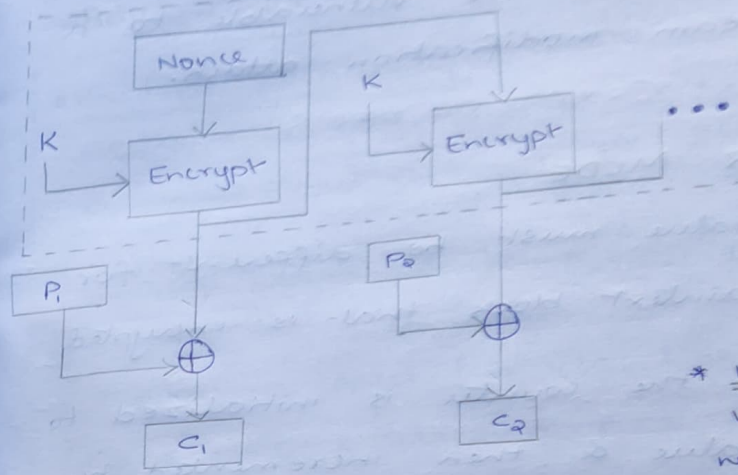


⊕ OUTPUT FEEDBACK MODE

* Similar to CFB, but here, the o/p of the Encry. func. is fed back to the shift register.
* Another difference → OFB mode operates on full blocks of plaintext & ciphertext, not on an s-bit subset.

Encryption

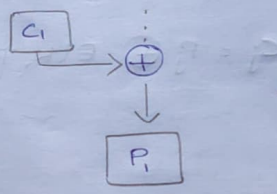
$$C_j = P_j \oplus E[K, [C_{j-1} \oplus P_{j-1}])$$



* Nonce → Random value w/ a negligible chance of repeating.

Decryption

$$P_j = C_j \oplus E(K, [C_{j-1} \oplus P_{j-1}])$$



(same as encryption)

(only 1 difference)

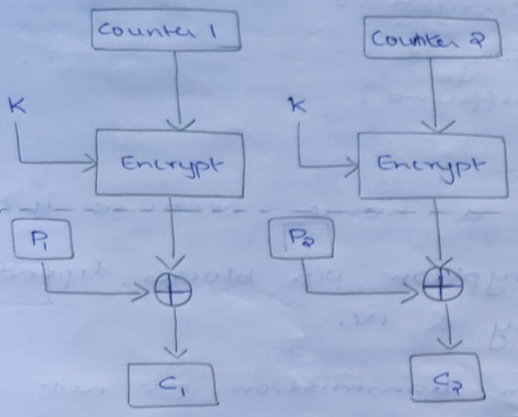
* The IV must be a nonce because the sequence of encryption o/p blocks, depends only on the key & IV.
* Adv : Bit errors in transmission do not propagate. (subsequent values won't be corrupted)

* Disadv : It is more vulnerable to a msg stream modification attack.

⑤ COUNTER MODE

* Counter value must be different for each plaintext block that is encrypted.
* Typically the counter is initialized to some value & then incremented by 1 for each subsequent block.

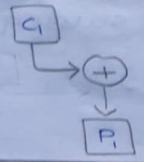
Encryption



$C_j = P_j \oplus E(K, T_j)$

Decryption

(Same) only diff →



$P_j = C_j \oplus E(K, T_j)$

Advantages of CTR mode:

- Hardware Efficiency - Unlike the 3 chaining modes, encr. here can be done in parallel on multiple blocks of pt. or ct.
- Software Efficiency - Because of opps for parallel execution, processors that support parallel features, like SIMD instr., aggressive pipelining, & a large no. of registers can be effectively utilized
- Preprocessing - The exec. of the Encry algo. doesn't depend on i/p of the pt. or ct. Therefore w/ sufficient memory & security, preproc. can be used to prepare the o/p of the Enc. boxes.
- Random Access - The ith block of pt. or ct. can be processed in random-access fashion
- Simplicity - This mode requires only the implementation of the Encry. algo & not the decry. algo.

UNIT - 2

PUBLIC KEY CRYPTOSYSTEMS

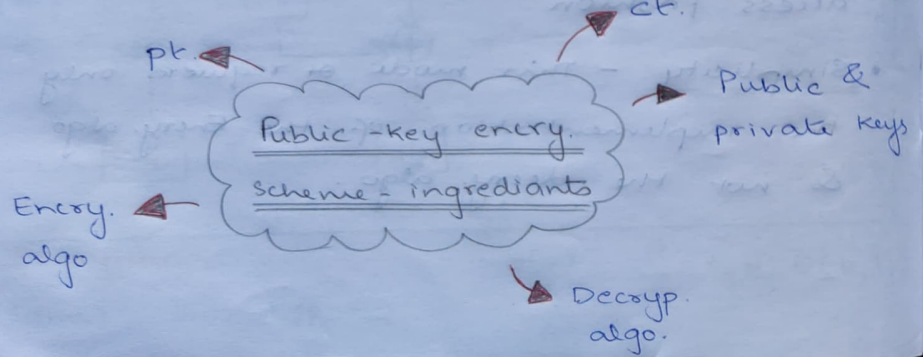
* Asymmetric algos rely on one key for encryption & a different but related key for decryption.

** Imp characteristic:

It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algo. & the encryption key.

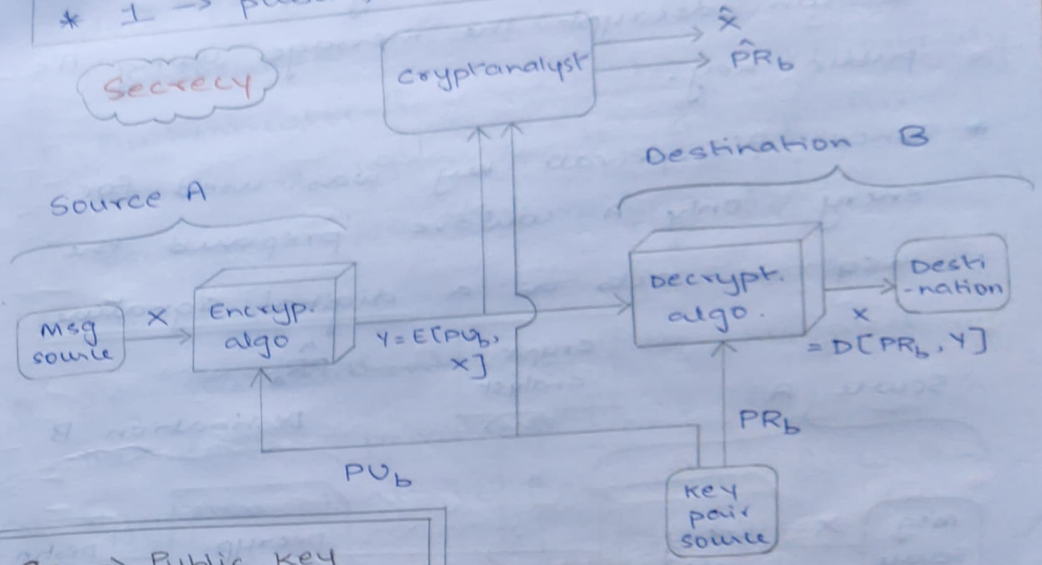
** Some algos (RSA) exhibit:

Either of the 2 related keys can be used for encryption, with the other used for decryption.



Steps:

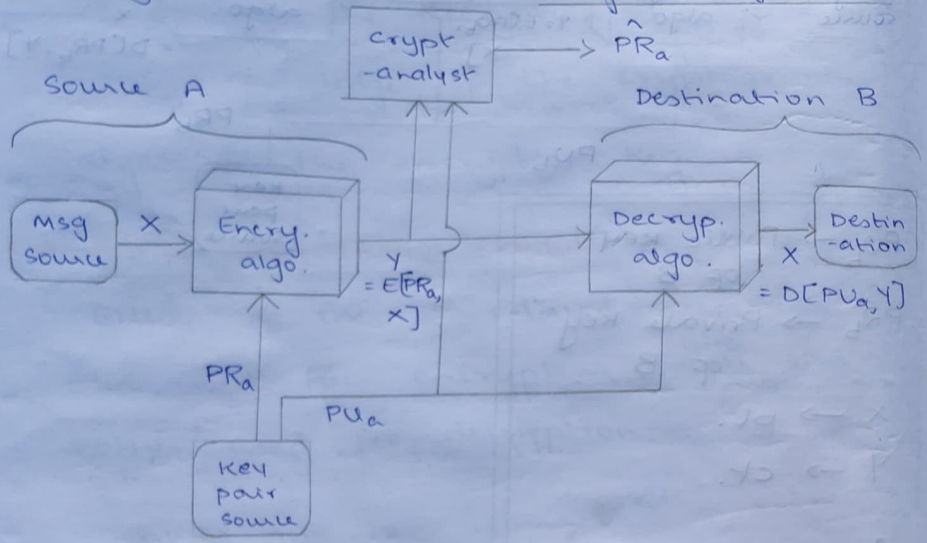
* Each user generates a pair of keys.
  1 -> encry. the other -> decry.
* 1 -> public, " " -> private



PU_b -> Public key of B
PR_b -> Private key of B
x -> pt.
Y -> ct.
$\hat{x}$ -> x estimated by the cryptan.
$\hat{PR}_b$ -> PR_b estimated " " "

Authentication

* A prepares a msg to B & encrypts it using A's private key.
* B can decrypt the message using A's public key.
* Since A's private key was used for encry., only A could've prepared the msg → acts as a digital signature

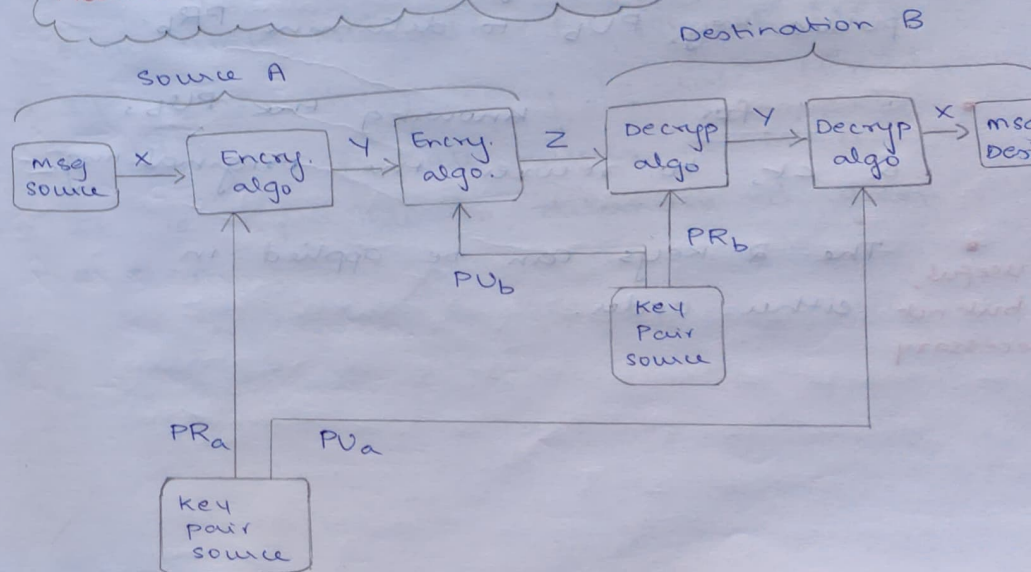


* Encrypting the entire msg, requires a great deal of storage.

∴ Efficient method → Encrypt a small block of bits that is a func. of the document.

* Authenticator ↑, must have the property that it is infeasible to change the document w/out changing the authenticator.
* If the authenticator is encrypted w/ the sender's private key, it serves as a signature that verifies origin & content.

Authentication & Secrecy



Requirements for Public-key Cryptography

- It is computationally easy for a party B to generate a pair (PU & PR)
- " " for A (sender), knowing the public key & the msg to be encrypted, to generate the corresponding ct.
- " " for the receiver B to decrypt the ct. using the private key
- It is computationally infeasible for an adversary, knowing the public key, PU_b to determine PR_b.
- " " for " ", knowing the PU_b, & ct, to recover the original msg.
- The 2 keys can be applied in either order.

Useful, but not necessary

THE RSA ALGORITHM

* It is a block cipher in which the pt. & ct. are integers b/w 0 & n-1 for some n. (1024 bits - typical size of n)

Encry. → $C = M^e \bmod n$

Decry. → $M = C^d \bmod n = M^{ed} \bmod n$

* Sender & Receiver need to know n.
* Sender knows e & only the receiver knows d.

$\therefore PU = \{e, n\}$

$PR = \{d, n\}$

Requirements:

→ It is possible to find values of e, d, n such that $M^{ed} \bmod n = M$ for all $M < n$

→ It is relatively easy to calc. $M^e \bmod n$ & $C^d \bmod n$ for all values of $M < n$

→ It is infeasible to determine d given e & n.

* $M^{ed} \bmod n = M$

holds only if e & d are multiplicative inverses modulo $\phi(n)$.

* For p & q prime, $\phi(n)$ → Euler totient function

$$\phi(pq) = (p-1)(q-1)$$

* Rel^{n}ship b/w e & d:

$$ed \bmod \phi(n) = 1$$

Ingrediants of the scheme	
P, q, 2 prime no:s	private, chosen
$n = pq$	public, calculated
e, w/ $\gcd(\phi(n), e) = 1$; $1 < e < \phi(n)$	public, chosen
$d = e^{-1} \pmod{\phi(n)}$	private, calculated

Example

1. Select 2 prime no's, $p = 17$, $q = 11$
2. Calc. $n = pq = 17 \times 11 = 187$
3. Calc. $\phi(n) = (p-1)(q-1) = 16 \times 10 = 160$
4. Select e such that e is relatively prime to $\phi(n) = 160$ & less than $\phi(n)$
   ∴ choose $e = 7$
5. Determine d, such that $de = 1 \pmod{160}$ & $d < 160$.

$$23 \times 7 = (1 * 160) + 1$$
$$161 = 161$$

$$d = \frac{1 + k\phi(n)}{e}$$

∴ PU = {7, 187}
PR = {23, 187}

$M = 88$

Encryption ⇒ $C = M^e \bmod n$

$C = 88^7 \bmod 187$

$88^7 \bmod 187 = [(88^4 \bmod 187) \times (88^2 \bmod 187) \times 88^1 \bmod 187)] \bmod 187$

$88^1 \bmod 187 = 88$
$88^2 \bmod 187 = 7744 \bmod 187 = 77$
$88^4 \bmod 187 = 59,969,536 \bmod 187 = 132$

∴ $88^7 \bmod 187 = (88 \times 77 \times 132) \bmod 187$

$C = 11$

Decryption ⇒ $M = C^d \bmod n$

$M = 11^{23} \bmod 187$

$11^{23} \bmod 187 = [(11^1 \bmod 187) \times (11^2 \bmod 187) \times (11^4 \bmod 187) \times (11^8 \bmod 187)] \bmod 187$

$11^1 \bmod 187 = 11$
$11^2 \bmod 187 = 121$
$11^4 \bmod 187 = 14641 \bmod 187 = 55$
$11^8 \bmod 187 = 214,358,881 \bmod 187 = 33$

∴ $11^{23} \bmod 187 = (11 \times 21 \times 55 \times 33 \times 33) \bmod 187$

∴ $M = 88$

Approaches to attacking the RSA algo.

- Brute Force - trying all possible private keys.
- Mathematical Attacks - several approaches, all equivalent in effort to factoring the pdt. of 2 primes.
- Timing attacks - Depend on the running time of the decry. algo.
- Chosen ciphertext attacks - Exploits properties of the RSA algo.

* Defense against brute force → use a large key space.

THE FACTORING PROBLEM

Approaches to attack RSA mathematically

* Factor n into its 2 prime factors. Then $\phi(n)$ can be calculated, & in turn $d \equiv e^{-1} \pmod{\phi(n)}$ is determined.
* Determine $\phi(n)$ directly. $d \equiv e^{-1} \pmod{\phi(n)}$ will/can be determined
* Determine d directly.

To avoid values of n that may be factored more easily, the algo's inventors suggested the following constraints on p & q:

* p & q should differ in length by only a few digits
* (p-1) & (q-1) should contain a large prime factor.
* gcd(p-1, q-1) should be small.

TIMING ATTACKS

* A snooper can determine a private key by keeping track of how long a comp. takes to decipher msgs.
* Suppose the target system uses a modular multiplication func. that is very fast in almost all cases, but in a few cases takes much more time than an entire avg. modular exponentiation.

* The attack proceeds bit by bit starting w/ the leftmost bit, b_k.
* Suppose that the 1st j bits are known.
* Based on the i/p, time taken to encrypt/decrypt differs

Countermeasures:

* Constant exponentiation time
* Random Delay
* Blinding

DIFFIE-HELLMAN KEY EXCHANGE

* Enables 2 users to securely exchange a key that can be used for subsequent encry. of msgs.
* Its difficult to calc. discrete logarithms.
* If **a** is a primitive root of the prime no. p, then the no.'s

$a \bmod p, a^2 \bmod p, \ldots a^{p-1} \bmod p$

are distinct & consist of the integers from 1 to p-1 in some permutation.

* For any integer b & a primitive root a of prime no. p, we can find a unique exponent i such that,

$$b \equiv a^i \pmod p \qquad 0 \le i \le (p-1)$$

i is called discrete logarithm of b for the base a, mod p.

The Algorithm

* There are 2 publicly known no.'s → prime no. q & an integer α that is a primitive root of q.
* Suppose A & B wish to exchange a key.
  - A selects a random integer $X_A < q$ & computes $Y_A = \alpha^{X_A} \bmod q$
  - B " " $X_B < q$, " " $Y_B = \alpha^{X_B} \bmod q$.
  - Each side keeps the X value private & makes Y available publicly to the other side.
  - A computes the key → $K = (Y_B)^{X_A} \bmod q$
  - B " " → $K = (Y_A)^{X_B} \bmod q$

Both calculations produce identical results.