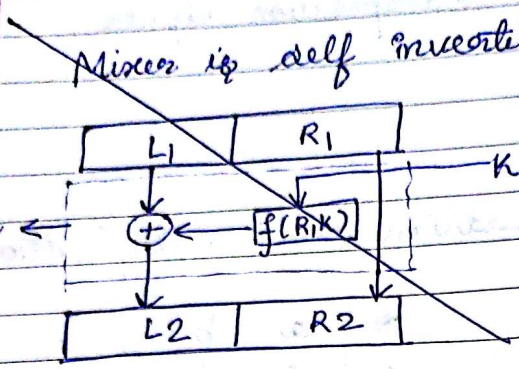
03/09/19
~~06/09/19~~ ①



Modern Block Ciphers

Substitution.

exhaustive search for key is difficult

difficult to implement

key space = 2^n!.

Transposition

$\frac{1}{2}$ no. of 1's and no. of 0's remain same., exhaustive search for key is comparitively easier.

key space = $n!$

easier to implement

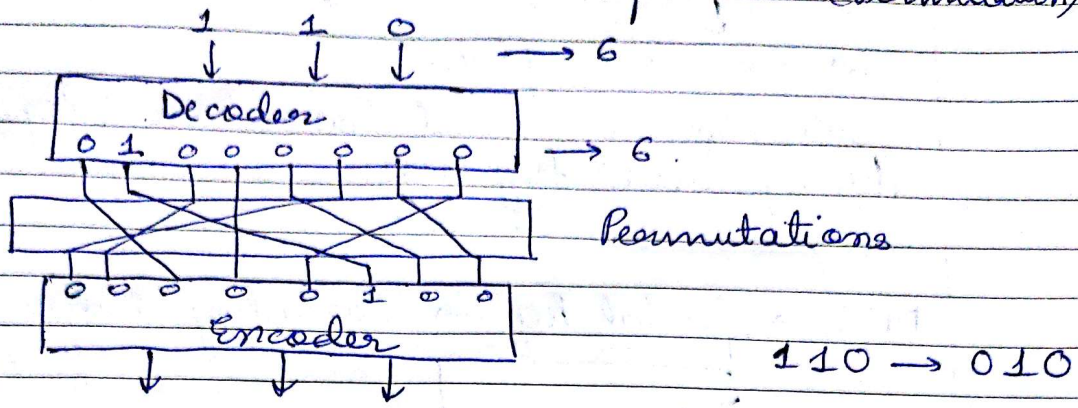
To be resistant to exhaustive search for key attack, a modern cipher needs to be designed as a substitution cipher.

Full key size ciphers : all keys in key space can be used in cipher.

Partial key size ciphers : some keys in key space not in cipher

Page No.:
Date:

Substitution modelled as Transposition (Permutation)



A full size key n-bit transposition / substitution block cipher can be modelled as a permutation but their key sizes are different.

(i) For a transposition cipher, key size is

$$\lceil \log_2(n!) \rceil \text{ bits}$$

(ii) For a substitution cipher, key size is

$$\lceil \log_2(2^n!) \rceil \text{ bits}.$$

Full key size is not practical, large number of bits required.

∴ Partial Key size : only some keys used and less bits required to represent them.

★ Partial key size cipher

Key becomes too large
∴ cannot use full size key

④

Eg: DES (partial key size)

↓

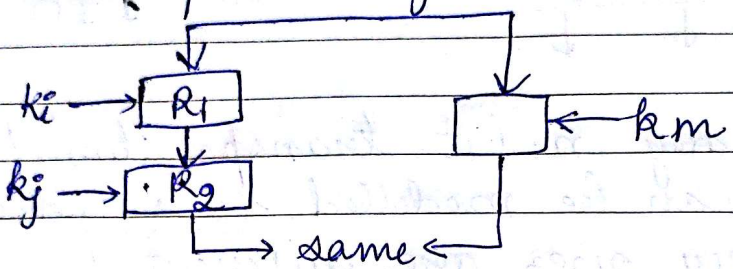
64 bits block cipher

Full size key would have been $\lceil \log_2(2^n) \rceil$

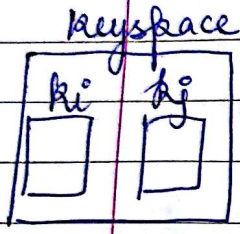
Actually used key size

= 56 bits $\approx 2^{70}$ bits

DES → partial key → cascading improves strength of cipher.



not possible to have key k_m that gives same O/P.

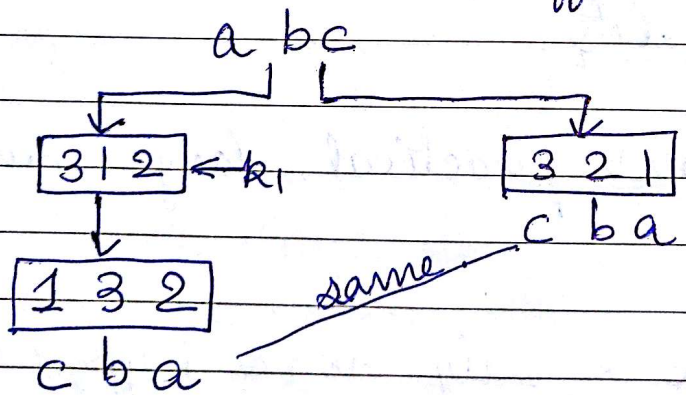


k_m : overlap of K_i & K_j

not possible

∴ strength increases.

① less key bits ② cascading same operation with different key improves strength



no use :

Page No.: Date: YOUVA

Full size key ciphers

Transposition

1) For n objects, possible permutations $= n!$

2) For $n!$ possible keys we need $\lceil \log_2 n! \rceil$ bits

Substitution

1) For n objects, possible substitutions $= 2^n!$

2) For $2^n!$ possible keys we need $\lceil \log_2 2^n! \rceil$ bits

DES : block size 64 bits

If block size = 32 bits

$= 32!$

$\log_2 (32!) \approx 118$

$\log_2 (2^{32}!) >>> 118$

∴ We implement substitution as transposition

04/09/19

Partial-key size cipher

2)

3)

Block.

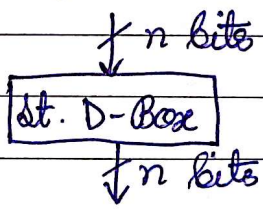
Components of Modern ~~bit~~ ciphers.

1) P-Boxes / D-Boxes
   - P-Boxes → permutation
   - D-Boxes → Diffusion

All components to convert plain text to cipher text should be invertible

Invertible a) Straight

size I/P = size O/P

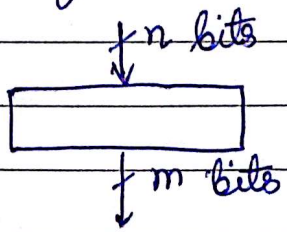


Invertible b) Expansion

size I/P < size O/P.

one to many mappings

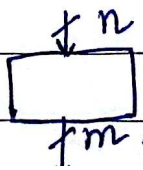
$m > n$.



c) Compression

size I/P > size O/P.

ignoring some inputs → not invertible



$m < n$.

2) S-Boxes.
↓
substitution — using mathematical equations for substitution
* May or may not be invertible ↓ based on how it is modelled.
equation – linear or non linear
Inverted : I/P size = O/P size.

3) Exclusive-OR (XOR).

Properties : In $GF(2^n)$ field $\Rightarrow$ $GF(2^1)$.
1) associative
2) closure
3) Inverse
4) Identity
5) commutativity — Feistal-cipher

$x \oplus \overline{x} = 111\ldots11$

$x \oplus 111\ldots11 = \overline{x}$

$of1 \oplus of2 = of3$

Known → then of2 can be found out

DES.
↓
this property is used in this

Transportation over insecure channel.

$CT = PT \oplus K$

→ Receiver has CT and key

$PT = CT \oplus K$.

The inverse of an exclusive OR operation can make sense only if one of the I/P is fixed.

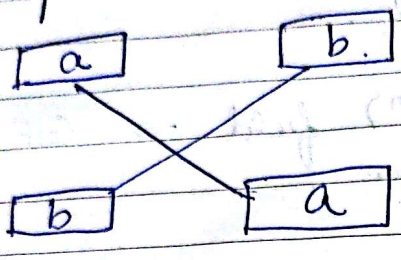
Here, key is fixed.

4) Circular shift

If encryption is shifting to left, inverse will be shifting to right.

Diffusion

5) Swap.



Inverse – swap operation performed again.

Confusion

6) split and combine

split into equal sized blocks

split and combine are inverses of each other.

Product Ciphers

⇒ introduced by Shannon
⇒ complex ciphers which combines substitution, permutation and many other components.
⇒ enables to have two important properties

- diffusion → hides the relation between cipher text and plain text.
- confusion → hides the relation between cipher text and key.

motive of attacker → deducing key (more than deducing plain text)

Page No.:
Date:

Diffusion PT_1, PT_2 have hamming distance of 1 bit

CT_1, CT_2 should have hamming distance of more than 50% of # of bits

Confusion K_1, K_2 used for encryption having Hamming distance of 1 bit

C_1, C_2 should have hamming distance more than 50% of # of bits

Cascading / multistaging of an operation — Rounds.

possible in DES.

Good cipher.

Rounds

Diffusion and confusion can be achieved using iterated product cipher.

Each iteration ⇒ S boxes, P boxes ....

Each iteration ⇒ a round.

each round has different key. — round key generator.

It uses another module i.e round key generator to create different keys for each round using the cipher key.

Receiver should also have key generation and cipher text.

2 classes of product ciphers

feistal

non-feistal.

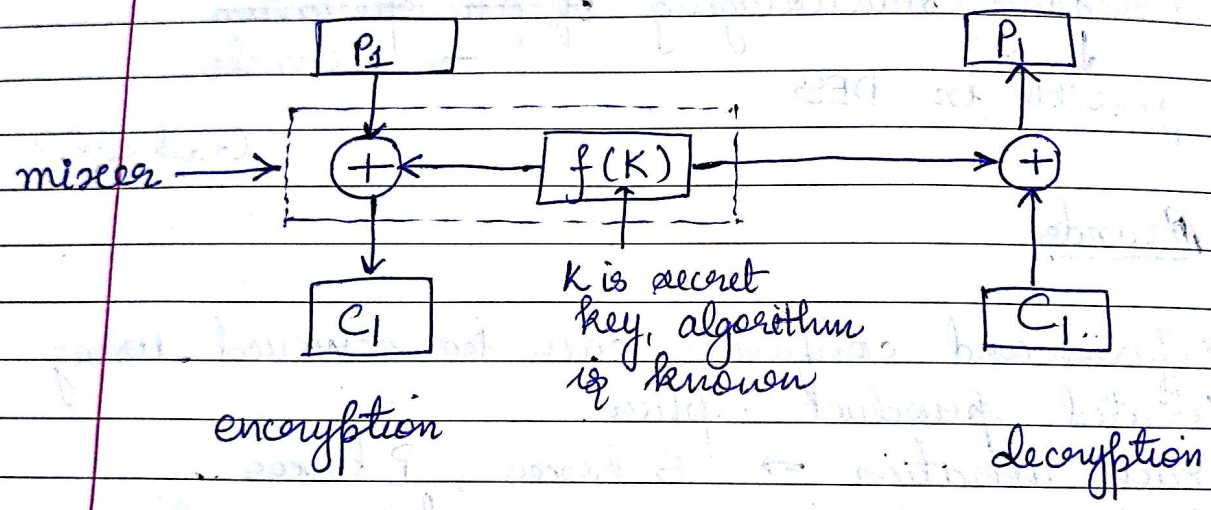
Page No.:
Date:

Feistal : makes use of both invertible and non-invertible components.

Non-feistal : makes use of only invertible components.

How to design feistal ciphers ?

They explore the inverse property of XOR operation (in condition that one of the I/Ps is constant).



$C_1 = P_1 \oplus f(K)$.

$P_1 \oplus f(K) \oplus f(K) \longrightarrow P_1$

C_1

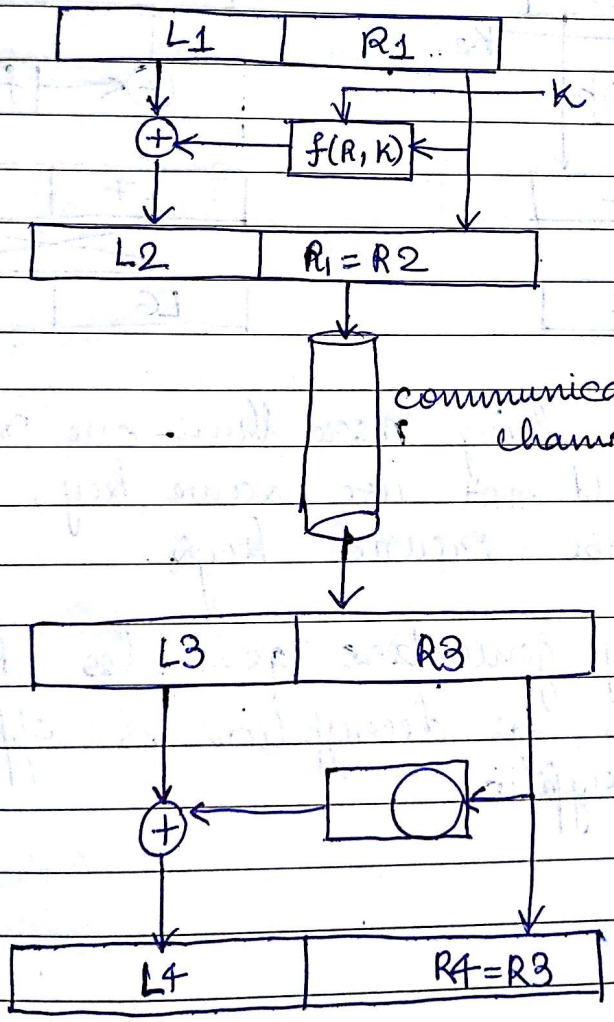
Even when component is invertible or not. ($f(K)$) → everything is put in the mixer. Just keep the component fixed → XOR inverse property will help in decryption.

If P_1 and P_2 differ in 1 bit, $f(K)$ is same, then C_1 and C_2 will also differ in 1 bit.

If any one plain text is compromised – it is helpful to get other plain text.

The function of mixing should be a function of both key as well as plain text.
(input to the mixer)

Split the I/P in two boxes / parts / blocks.



(Assuming no transmission error)

$R_4 = R_3 = R_2 = R_1$

$L_4 = L_3 \oplus f(K, R_1)$
$= L_2 \oplus f(K, R_1)$
$= L_1 \oplus f(K, R_1) \oplus f(K, R_1)$
$= L_1.$

$\therefore L_4 = L_1.$

(decryption)

$R4 = R1.$

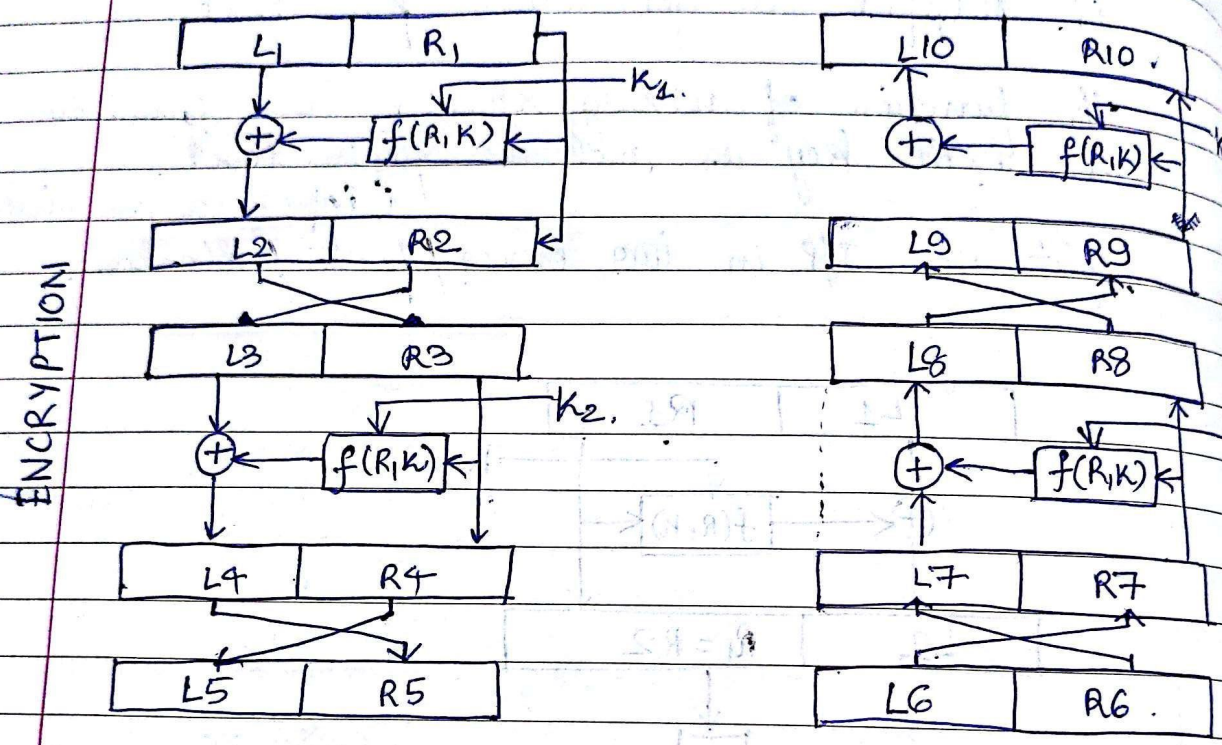
Page No.:
Date:

06/09/19

Right part remains same. — easier for decryption or attack.

↓

Disadvantage.



Here, we are doing more than one round.

∴ we should not use same key, we should use round keys.

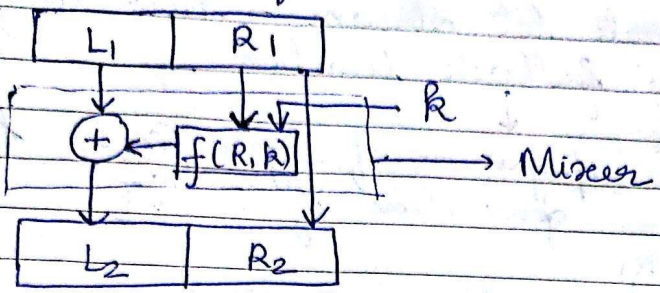
⇒ A round key generators generates K_1, K_2.

⇒ order of keys in decryption is opposite to that of encryption

06/09/19

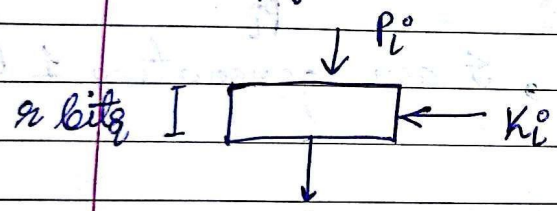
Page No.:
Date:

Mixer is self invertible



Modern Stream Ciphers

For incoming bits, if we are not encrypting on the fly, it becomes stream cipher.



$r << n$.

n bits ciphertext
r bits plaintext

Challange : generating the key stream – has to be random.

$C_i^o = P_i^o \oplus K_i^o$

Here, feedback shift register is used.

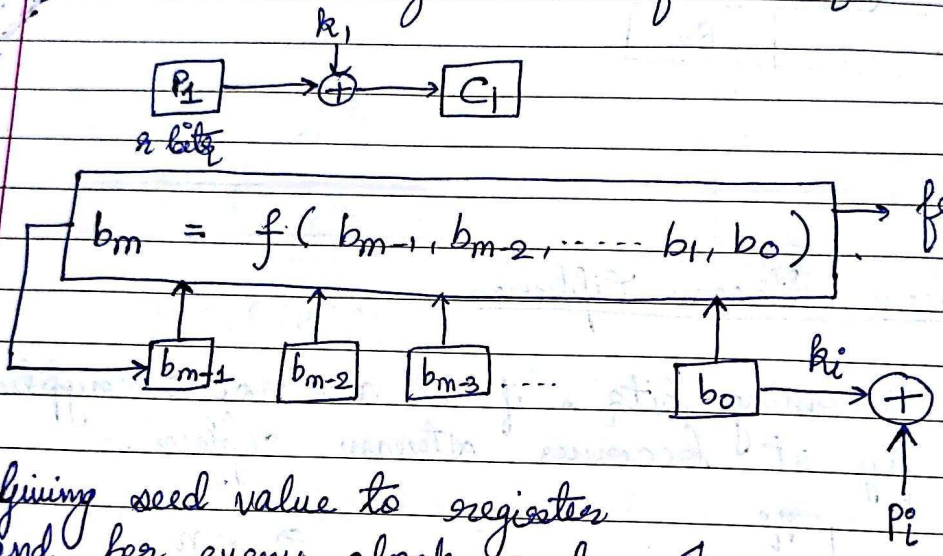
Synchronous
Non synchronous
] Types of modern stream ciphers.

Synchronous – Key stream is independent of plain text and cipher text.

Asynchronous – Key stream dependent on P and C.

Feedback function in the feedback shift register.

Register
→ It will generate bit stream – how to generate is given by feedback function
↓
polynomial of degree $(r-1)$ for generation of r bits.



giving seed value to register and for every clock cycle, I am generating 1 bit (after shifting).

∴ It is synchronious.

According to $f(\text{---})$ (→ feedback function), the feedback shift register can be linear or non linear.

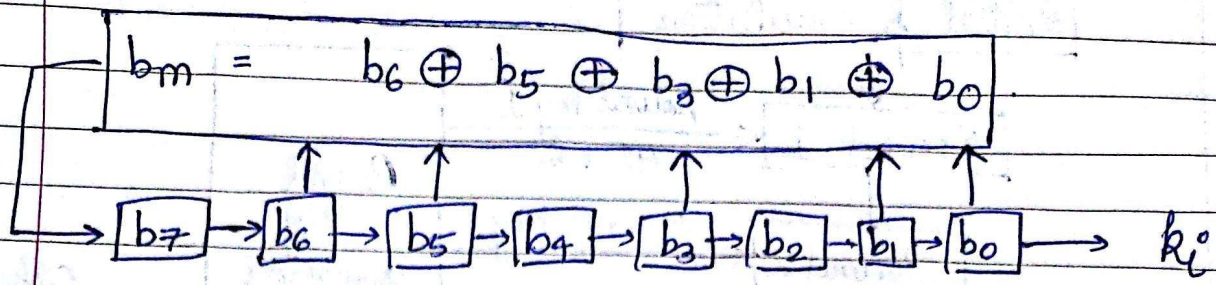
$$b_m = c_{m-1}b_{m-1} + c_{m-2}b_{m-2} + \dots c_0 b_0$$

↓ has to be 1.

Whatever characteristic polynomial we use for feedback function, its last bit should be 1 to maintain link.

Date:

$n = 8$



b_0 is the output.
Then shift to right.
We need b_7 – calculated from function

Linear feedback shift register → if feedback function is linear.

Non linear : operations on bit pairs or more # of bits rather than single bits like linear function.

Eg : $b_m = (b_6 \cdot b_4) \oplus (\overline{b_5} \cdot b_3) \oplus (b_5 \cdot b_2)$

How are standards standardized ?

International forum for all standards of connection networks, information security etc.

Data Encryption Scheme. (1975).

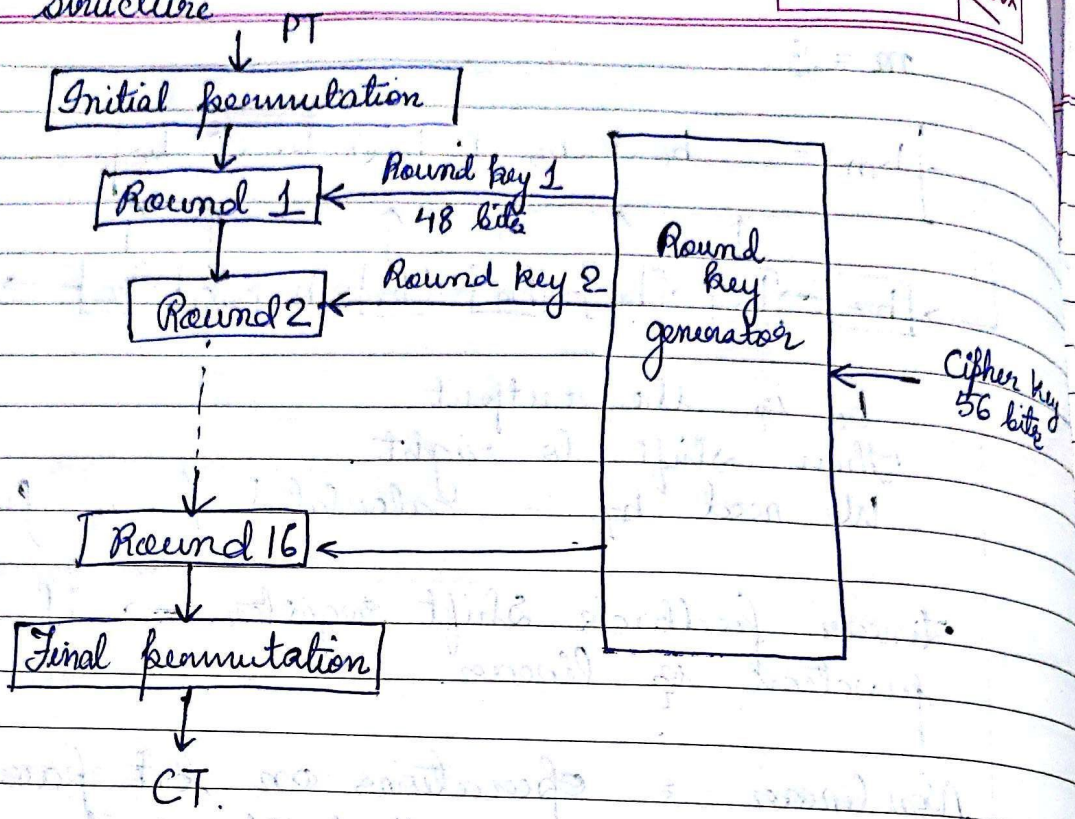
Feistal Ciphers

⇒ 64 bits block
⇒ 56 bits cipher key
⇒ 16 rounds
⇒ 48 bits round key

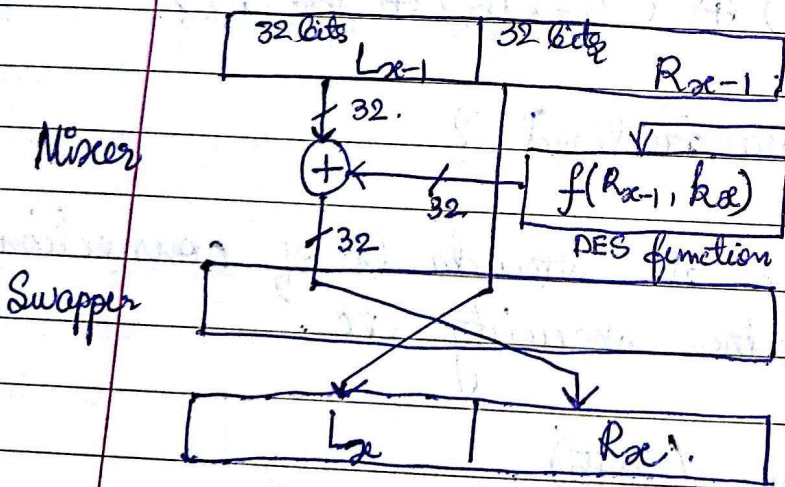
→ each round is simp
heavily depends on S-boxes

Page No.:
Date:

Structure

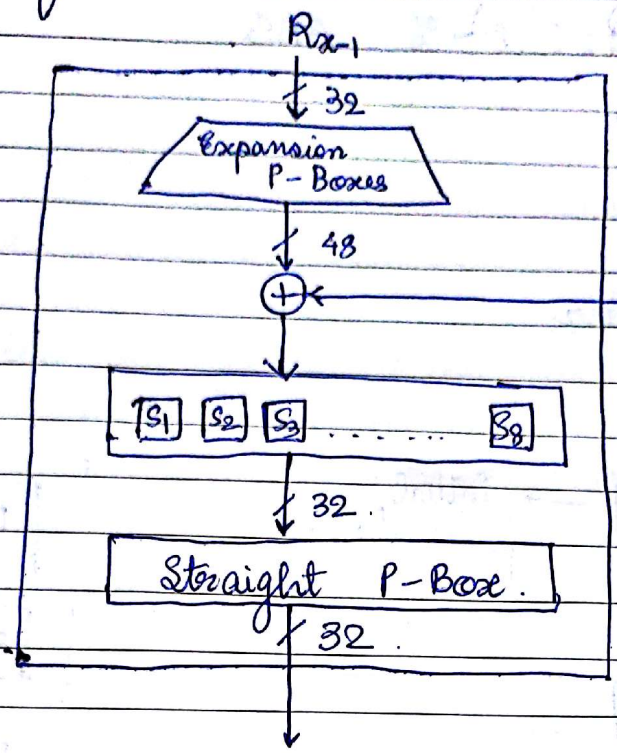


Initial and final are inverses of each other.

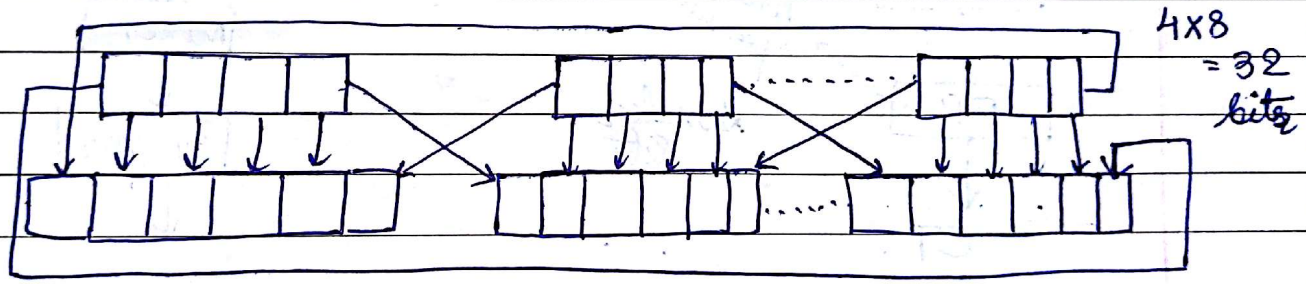


Page No.:
Date:
YOUVA

DES function

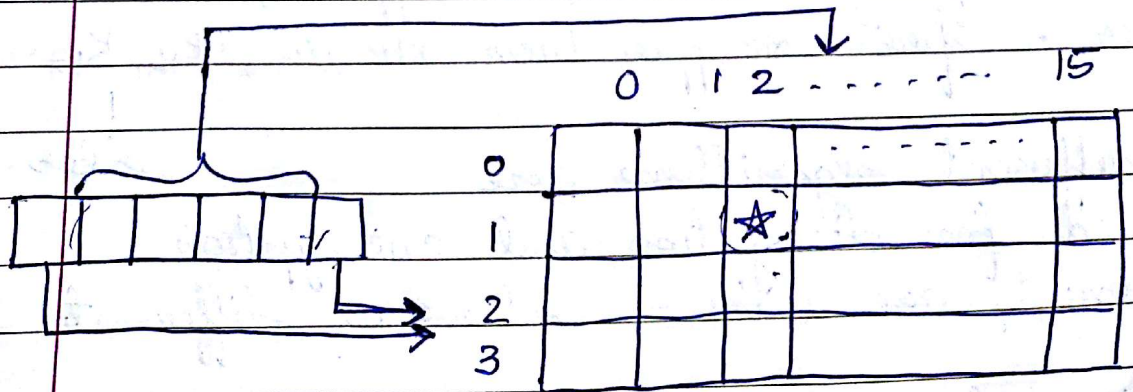


Expansion P-boxes.



6x8 = 48 bits

Going from 48 bits to 32 bits using 8 S-Boxes



These values are pre-filled.

→ 2 bits to access row

→ 4 bits to access columns

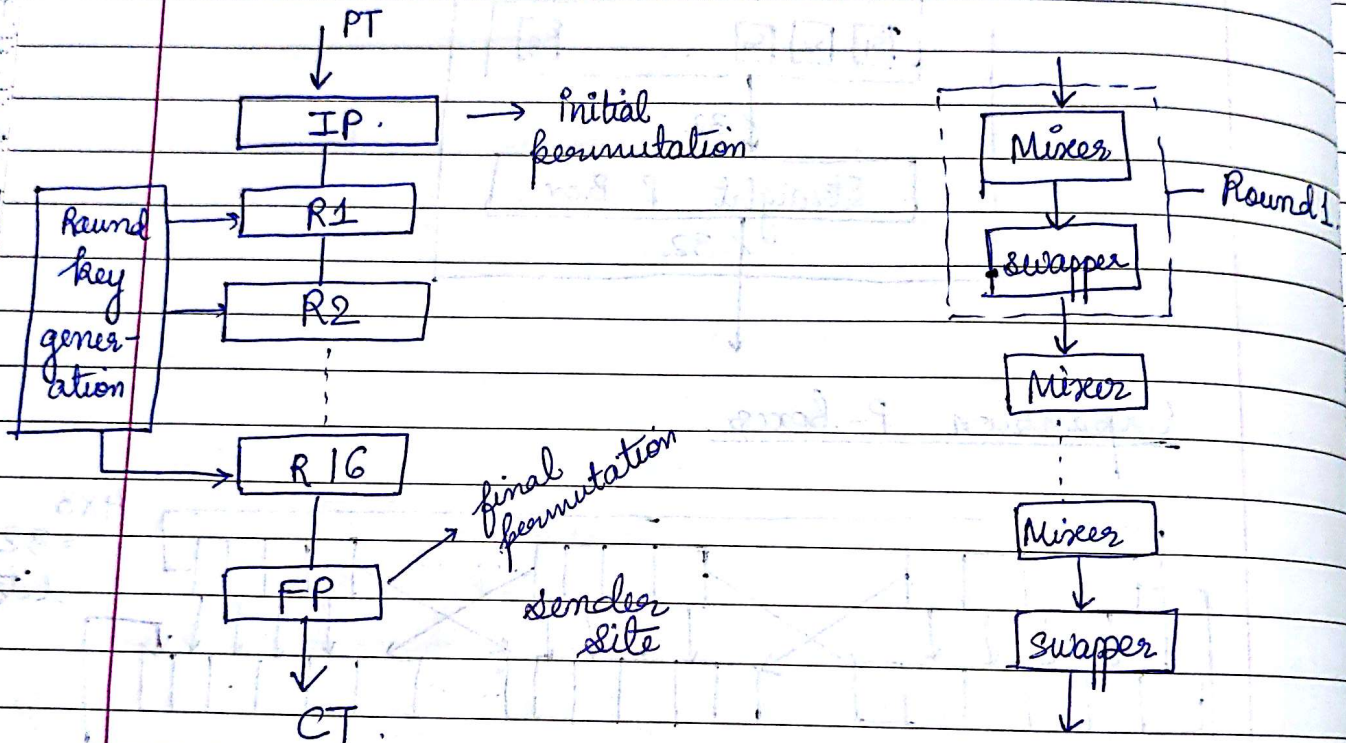
→ The bits we get is 4 bits value needed

pre-filled values → 4 bits

Straight boxes – public ∴ criticised.
⇒ Switched to AES.

11/09/19 DES

Reverse Cipher



Sender side : I/P : plain text, round keys.
Receiver side : I/P : cipher text, round keys.

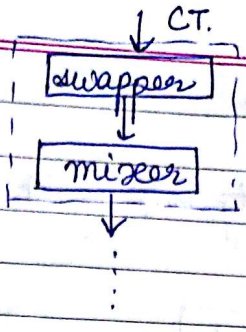
sender : first mixer then swapper, $K_1, K_2 \dots K_{16}$
receiver : first swapper then mixer, $K_{16}, K_{15} \dots K_1$ ↓ order.

Two different algorithms are required for decryption and encryption
Components are same - order is different.

Order of keys also changes.

Page No.:
Date:

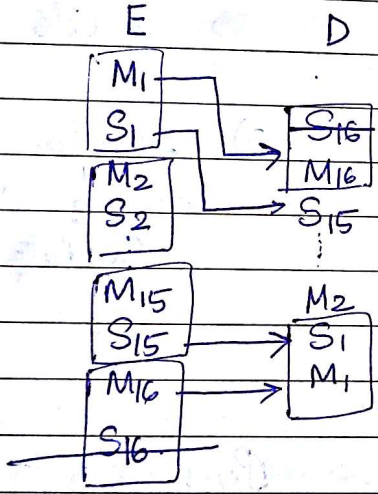
Receiver side :



Round 1 for decryption ← K_{16}.

We need to align the rounds → modify the last round a bit.
Remove the swapper in round 16 in encryption now my rounds are aligned.

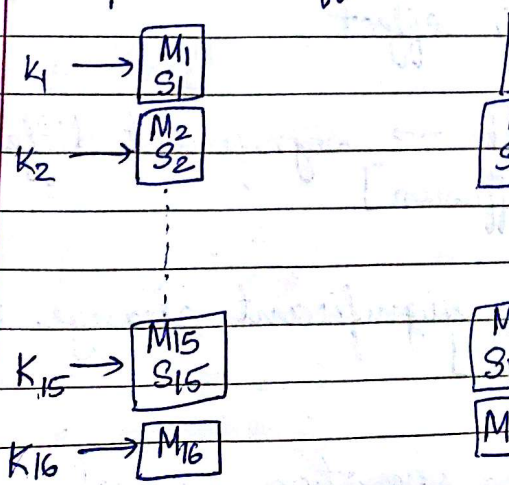
To have same algorithm for encryption and decryption — use same machine.



not required

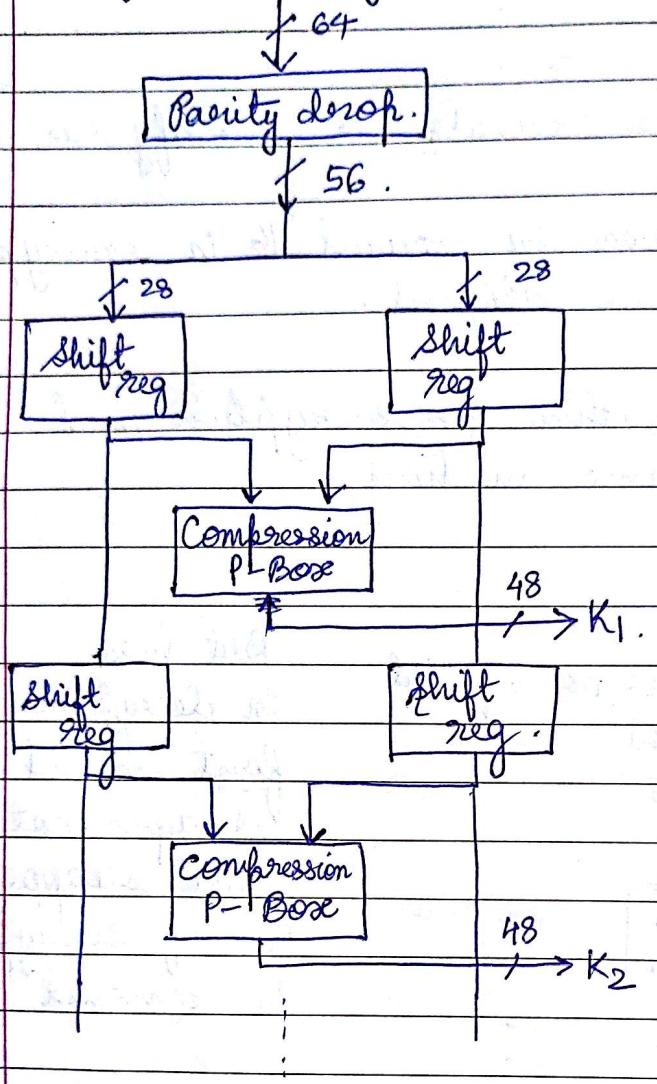
But here in decryption, first round swapper not needed since swapper of R_{16} of encryption is removed.

All mixers and swappers are same — Intermediate outputs are different.



Key generation

Key with parity bits – 64 bits



1) Parity drop
2) Shift left (Circular shift).

Round 1, 2, 9, 16 ⇒ 1 bit left shift.

Round 3, 4, 5, 6, 7, 8, 10-15 ⇒ 2 bits left shift

3) compression P-boxes ⇒ 56 bits to 48 bits

Properties

A single bit difference affects more than 50% of output – avalanching effect.

small change in plain text → significant difference in cipher text. [Diffusion]

small change in Key → significant change in cipher text [Confusion]

Each bit undergoes the operation – completeness

Page No.:
Date:

Criticism (DES - Weakness)

1) Hiding the structure of S-boxes / P-boxes.
2) Keys → semi weak, weak, possible weak keys.

For complete encryption of text - atleast 2 rounds (even number needed). because 1 round modifies only half of plain text.

Why 16 rounds only ? - according to DES standards :

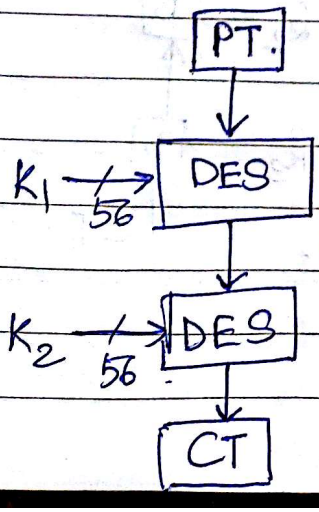
1) more than 16 rounds - does not increase the strength of cipher.
2) less than 16 rounds - strength of cipher reduces.

Security

1) "Brute force" attack is possible (2^{56}) with today's technology.
2) Single DES cannot withstand Brute force attack

∴ Multiple DES used.

Multiple DES



∴ Key space : 2^{112}.

$2^{56} \cdot 2^{56}$

$K_1 \cdot K_2$

Ideally security should be 2^{112}.

But there is a meet in the middle attack.

Meet in the middle attack

Start with PT, reach intermediate cipher text.
Start with CT, reach intermediate cipher text

PT to

K_i	ICT_i

CT to

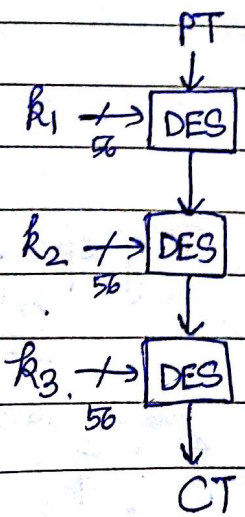
K_j	ICT_j

Sort tables based on ICT_i and ICT_j → look for matching ICT.

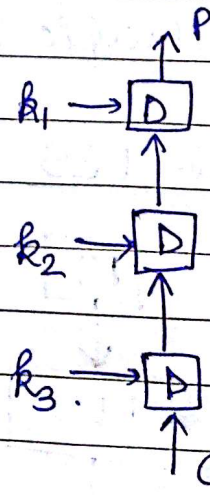
$2^{56} + 2^{56} \rightarrow 2^{57}$. (Hardly more than single DES)

Here, both plain text and cipher text needed for this attack.

Solution



3TDES



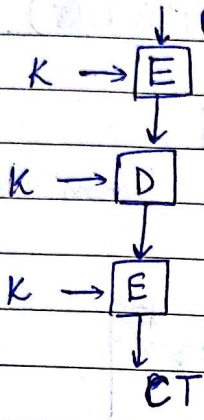
13/09/11.

Page No.:
Date:

Triple DES (TDES)

Key space : $2^{56} \times 3$. key : 56 bits

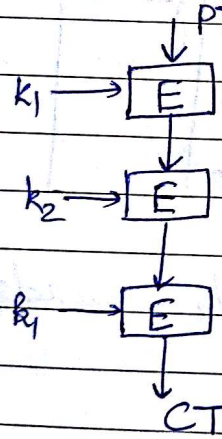
To make it backward compatible to single DES, it can also be designed as



key space : 2^{56}.

Now, this is compatible to single DES.

2TDES. $k_1 = k_2$, k_3.



Master Visa] TDES.

CAST, IDEA, Blowfish] self study algo.

Pretty good Privacy — TDES

DES : feistal cipher.

AES (Advanced Encryption Scheme)

With advancements in computing system — brute force attacks became a possibility

∴ AES was introduced.

some conventions/constraints to follow.

were standardized.

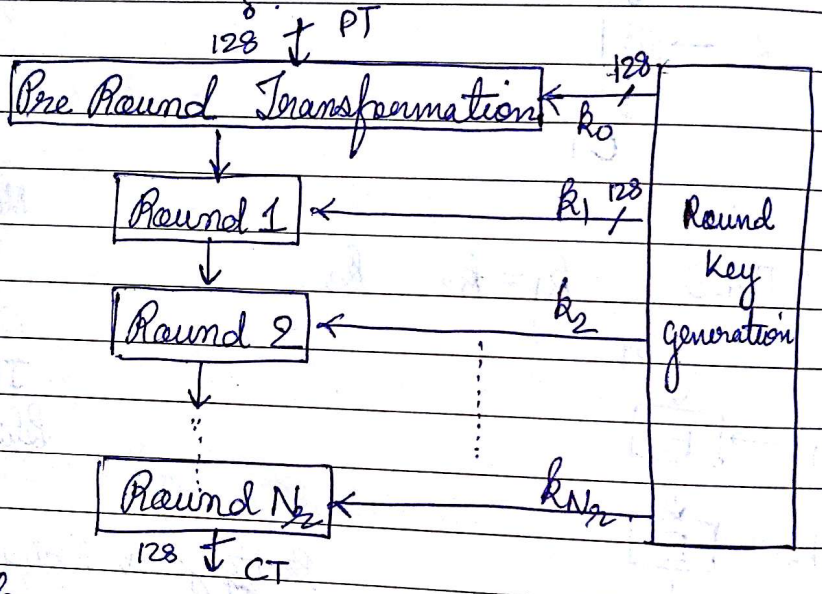
Eg : key size, ease of encryption etc.

Rijndael.

- ★ 128 bits block
- ★ Versions key size #rounds.
  - 128 10
  - 192 12
  - 256 14

★ bit (1), byte (8), word (32), block (128 bits).

★ state $S = [W_0, W_1, W_2, W_3]$.
preprocess blocks/ to represent them as states words ?



(N_r+1) keys are generated for N_r rounds.
↓
pre round.

Last round is different than others.

Structure of Each Round.

★ SubByte – Intrabyte substitution
→ using lookup table.
A lookup table is given → using $GF(2^8)$ with $x^8+x^4+x^3+x+1$

sub byte – substitute byte

Page No.: Date: YOUVA

			Byte in each cell.
			$S_{r,c}$.

Sub byte $d = X(S_{r,c})^{-1} \oplus Y$.

X, Y – constant polynomials

GF(2^8) → use this and not just 8 bit binary number because inverse is available for all elements of GF(2^8).

Plain text is arranged columnwise in state matrix to find $S_{r,c}$.

$128 = 16 \times 8$.

$X = \begin{bmatrix} 10001111 \\ 00011111 \\ 00111110 \\ \vdots \end{bmatrix}$ (rotations) $Y = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$.

Inverse : $[X^{-1}(d \oplus Y)]^{-1} = S_{r,c}$.

Put value as d and verify.

$$[X^{-1}(X(S_{r,c})^{-1} \oplus Y \oplus Y)]^{-1}$$

$$= [X^{-1}X(S_{r,c})^{-1}]^{-1}$$

$$= [(S_{r,c})^{-1}]^{-1} = S_{r,c}.$$

$(S_{r,c})^{-1}$ → calculating this inverse is non-linear transformation

∴ $d = X(S_{r,c})^{-1} \oplus Y$ transformation becomes non-linear

↓

required to get diffusion and confusion

Page No.:
Date:

☆ Shift row

Row 0 → No shift
1 → 1 byte
2 → 2 bytes
3 → 3 bytes

Polynomial Irreducible
= 100011011

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

shift left
keep shifting

X Y

$$d = \underset{8\times8}{X} \cdot \underset{8\times1}{S_{r,c}^{-1}} \oplus Y$$

$$S_{r,c} = [\bar{X}(d \oplus Y)]^{-1}$$

17/09/19.

Page No.:
Date:

☆ Mixing (Reversible transformation).

$S = [W_0, W_1, W_2, W_3]$

y', z', t', x' = individual bytes

$$\begin{bmatrix} x' \\ y' \\ z' \\ t' \end{bmatrix}_{W_i'} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix}_{W_i}$$

constant matrix AES standard.

$x' = ax + by + cz + dt$

$y' =$

$z' =$

$t' =$

inverse exists

$W_1 = [b_1\ b_2\ b_3\ b_4]$

$W_2 = [b_5\ b_6\ b_7\ b_8]$

$b_1 = b_5$

$b_2 = b_6$

$b_3 = b_7$

$b_4 \neq b_8$

$b_1' = b_5'$?

↓

NO

↓

context is different.

context dependent.

Even if one byte only is different and rest is same, still its transformation bytes will all be different.

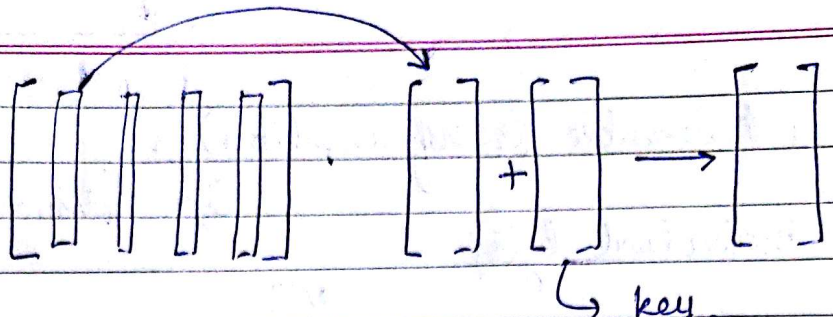
Interbyte transformation — substitution depends on contents of neighbouring bytes as well.

☆ Add Round Key.

It will work word by word.

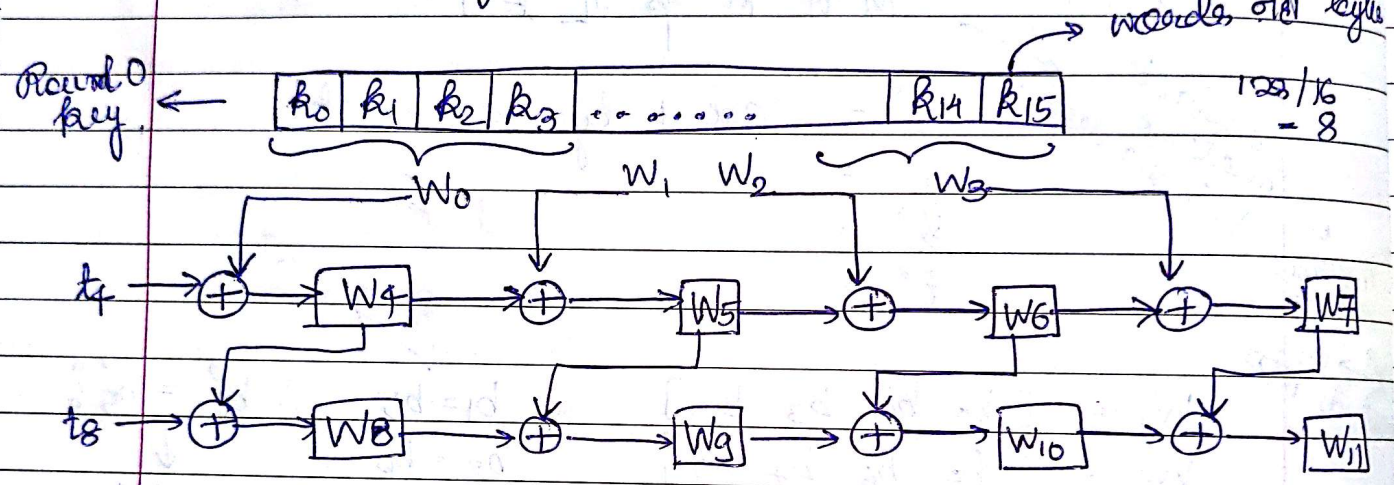
$S = [W_0\ W_1\ W_2\ W_3]$

$$= \begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{bmatrix}$$



word by word adding

☆ Round Key Generation



$W_4 W_5 W_6 W_7$ → round 1 key
$W_8 W_9 W_{10} W_{11}$ → round 2 key.

where t_i = subword (rotatedWord (W_{i-1})) ⊕ RotatedConstt $i/4$
t_4 =

1) if ($i \% 4 \neq 0$)

$W_i = W_{i-1} \oplus W_{i-4}$

else

$W_i = t_i \oplus W_{i-4}$.

After rotating Word ↓
$k_{13} k_{14} k_{15} k_{12}$.

Page No.:
Date:

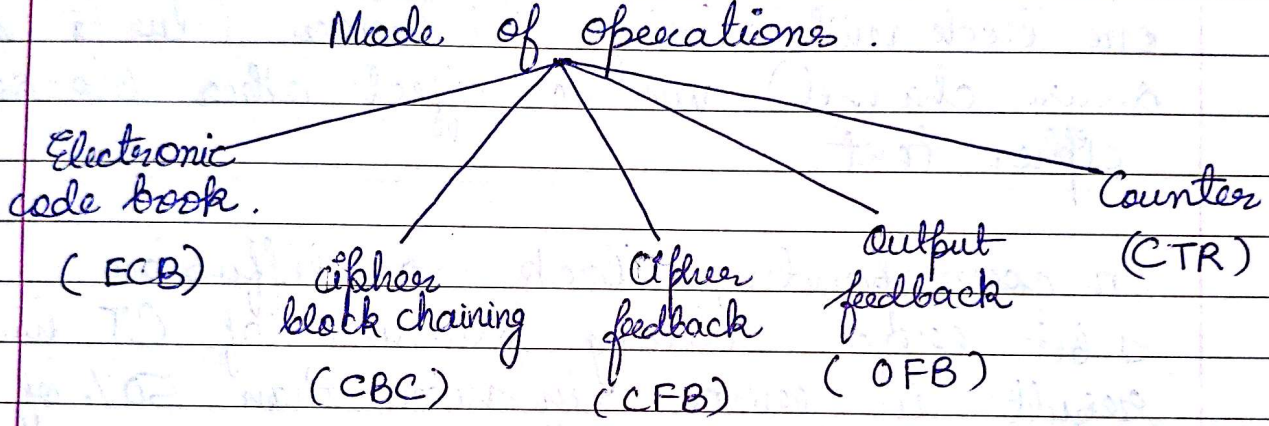
Make encryption and decryption compatible → last round # no mixing function

All operations reversible

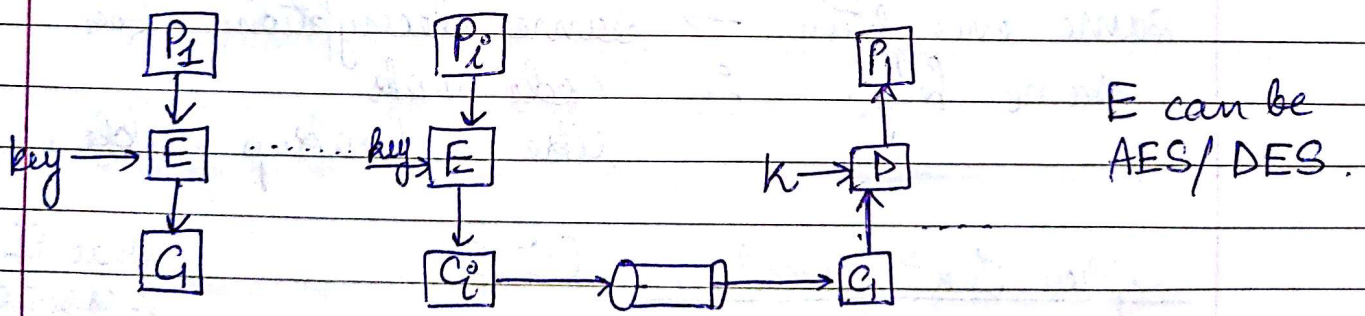
key space = 2^{128}

brute force attack not possible.

Mode of Operations (Use of modern ciphers)



1) ECB



Problem

$P_3' \neq P_3^2$

$C_1^1 = C_1^2$

$M_1 - P_1' \; P_2' \; P_3' \; P_4'$

$C_1' \; C_2' \; C_3' \; C_4'$

$M_2 - P_1^2 \; P_2^2 \; P_3^2 \; P_4^2$

$C_1^2 \; C_2^2 \; C_3^2 \; C_4^2$

If $(P_{11} = P_{21}) \Rightarrow C_{11} = C_{21}$ (for same key)
If $(P_{11} \neq P_{21}) \Rightarrow C_{11} \neq C_{21}$ (——— " ———)

$P_{11} = P_{31} \Rightarrow C_{11} = C_{31}$.
$P_{11} \neq P_{31} \Rightarrow C_{11} \neq C_{31}$.
} does not depend on continuous blocks.

Parallel computing possible here.
Each and every block is independent for encryption and decryption

one block with a single bit error (due to non-secure channel) will not affect other blocks of cipher text.

In corresponding block → diffusion
1 bit error during Xmission of CT will result in error in more than 50% of that CT block bits

Same encryption → same decryption for same key. ∴ code book.
like a lookup table.

Cipher Text Stealing (CTS).

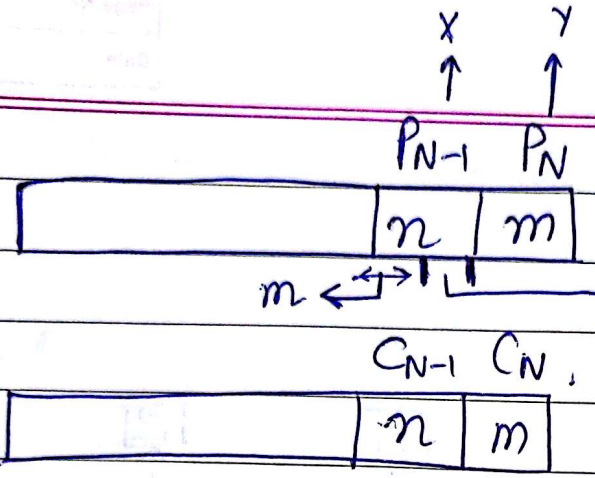
what if message is less than 128 bits.

P_{N-1} & P_N get encrypted differently and out of order.

$P_{N-1} \Rightarrow n$ bits & $P_N \Rightarrow m$ bits, $m \leq n$.

Encryption

$X = E_k(P_{N-1}) \Rightarrow C_N = \text{head}_m(X)$
$Y = P_N \,|\, \text{tail}_{(n-m)}(X) \Rightarrow C_{N-1} = E_k(Y)$.
$m + (n-m)$.



(n-m) bits of P_{N-1} are getting encrypted twice

Decryption

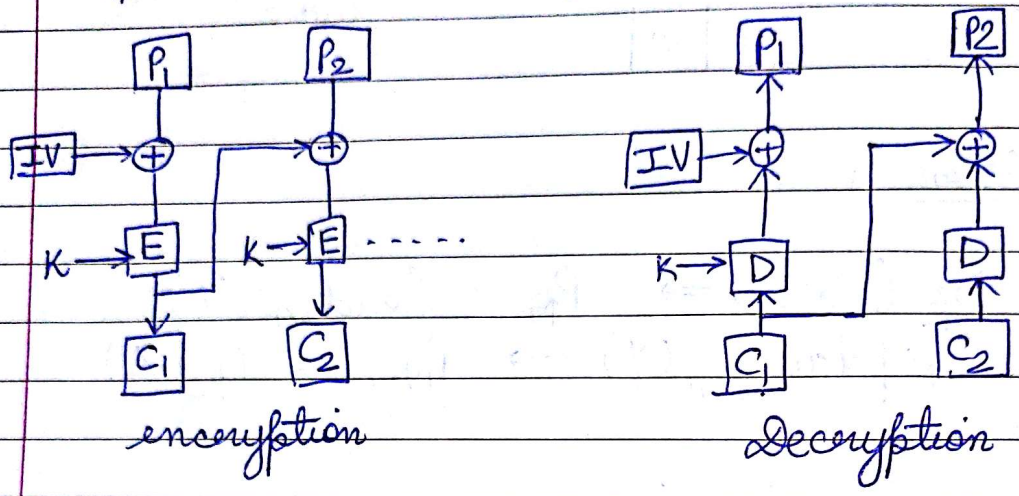
. $Y = D_k(C_{N-1}) \Rightarrow P_N = \text{head}_m(Y)$

$X = C_N \,|\, \text{tail}_{n-m}(Y) \Rightarrow P_{N-1} = D_k(X)$.

18/9/19

Mode of Operations

Cipher Block Chaining.



$C_i = E_k(P_i \oplus C_{i-1})$ for $i > 1$

$C_i = E_k(P_i \oplus IV)$ for $i = 1$

} Encryption

Where IV is initialisation vector
predefined agreed upon vector

$P_i = D_K(C_i \oplus C_{i-1})$

Preprocessing – XOR before encryption

IV can be kept secret in addition to key.

M_i ..... P_i

M_{i+1} ..... P_j

$P_i = P_j$.

$C_i \neq C_j$.

But if their position is same in both messages, say i and j to P_i are same, then cipher text will be same.

C_1 has 1 bit error
Corresponding P_1 will show Avalanching effect (diffusion)
But $C_2, C_3 \ldots$ will not be as affected.
C_i – 1 bit error, P_i – # bits – more than 50% error
C_{i+1} – 0 bit error, P_{i+1} – 1 bit error
Rest – no errors.

C_i – 1 bit error, P_i – lot of bits error
C_{i+1} – no error, P_{i+1} – 1 bit error as it has C_i as I/P.

Cipher Text Stealing

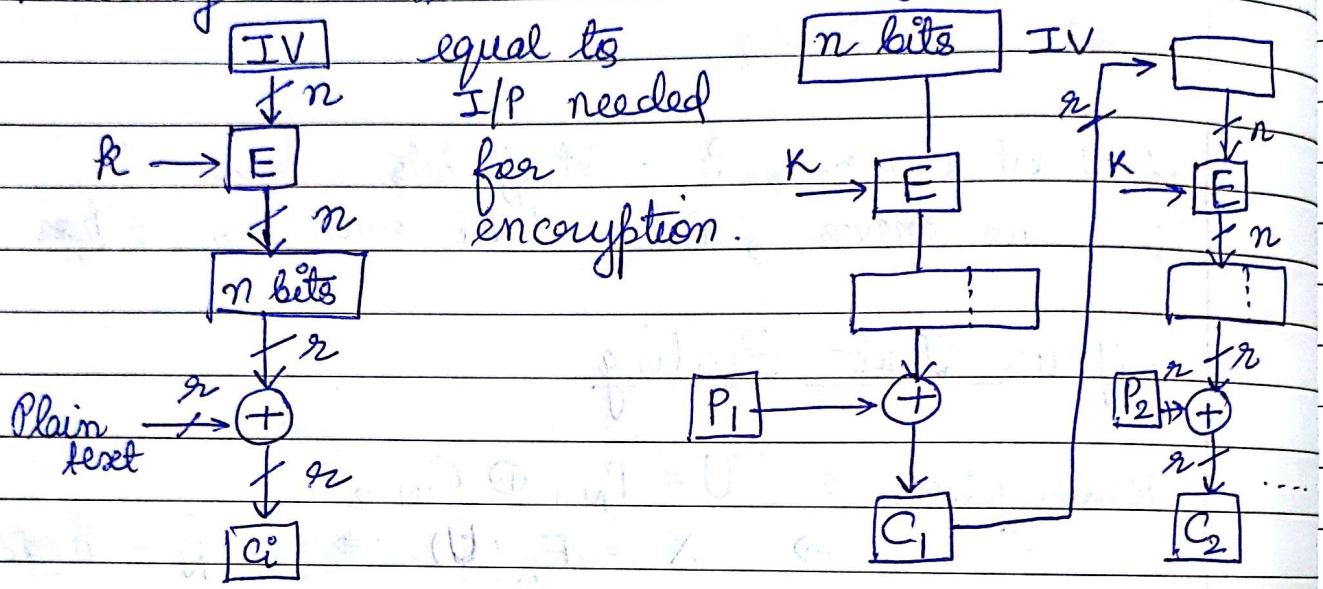
Encryption : $U = P_{N-1} \oplus C_{N-2}$
$\Rightarrow X = E_K(U) \Rightarrow C_N = head_m(X)$.

Decryption : $V = P_N \mid Pad_{n-m}$
$\Rightarrow Y = D_k(V) = X \oplus V$
$\Rightarrow C_{N-1} = E_k(Y)$

Cipher Text Feedback

IV encrypted using key → whatever is generated is XORed with plain text to get cipher text.

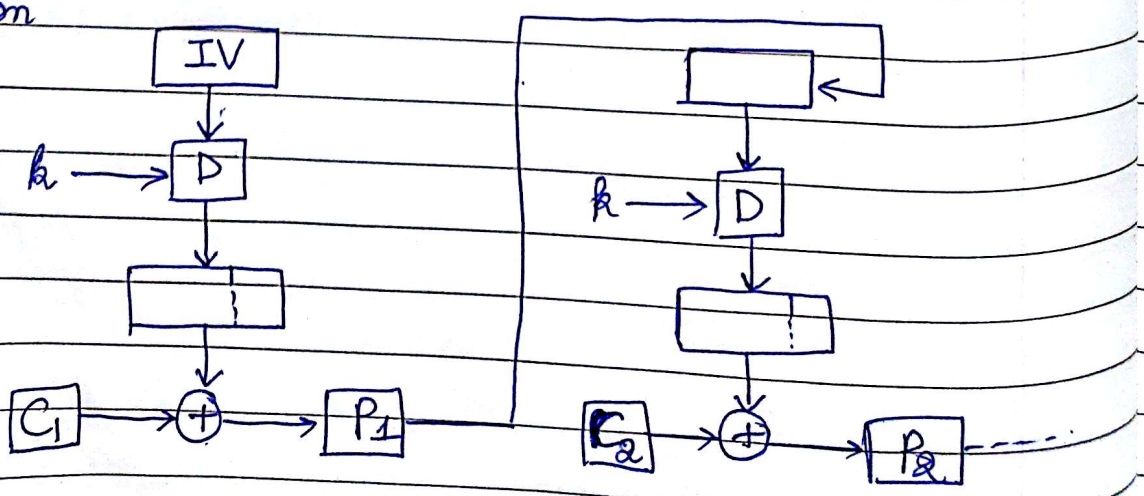
Advantage : Plain text block size need not be equal to I/P needed for encryption.



IV → shift registers
→ shifted left r bits for next round and C_{i-1} is appended.

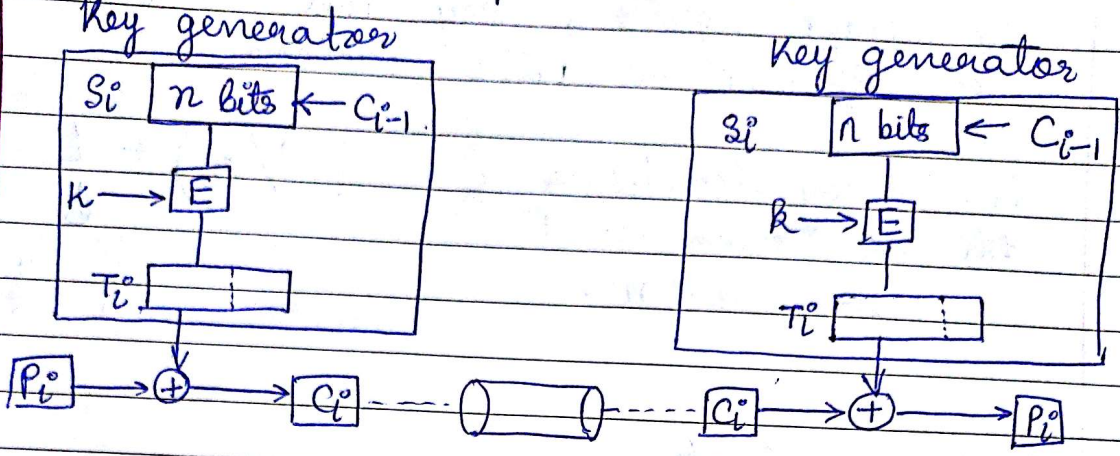
Operation is ⊕. — self invertible if one of the I/Ps is same.

Decryption



Page No.:
Date:

CFB as stream cipher



Are block patterns preserved?
No, heavily dependent on encryption of previous block cipher text.

If $P_k = P_j$ $\quad M_1 = P_1 \ldots P_k \ldots P_m$.
$C_k \neq C_j$ $\quad M_2 = P_1 \ldots P_j \ldots P_l$.

If first 'k' blocks of P are same for M_1 and M_2, $C_{k+1}^{M1} = C_{k+1}^{M2}$

If C_1 has 1 bit error, P_1 has 1 bit error.
Same C_1 is used to generate key for P_2.
no error in C_2 but key is corrupted.
∴ P_2 will have most bits in error.

C_3 - no error
C_2 - no error

if $r = n$, then no error in P_3 and ahead.

if $r < n$ → the corrupted bit of C_1 might still persist after shifting the shift register
∴ Key is corrupted ⇒ P_3 will be corrupted

more than 50% bits will be corrupted.

Page No.:
Date:

20/9

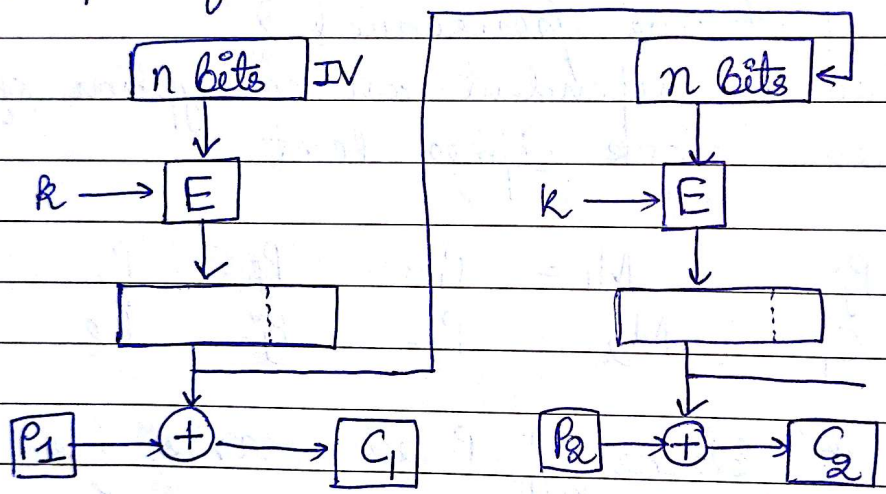
Until that error bit of C_1 does not leave the shift register, key will be corrupted and P_i will be with errors.

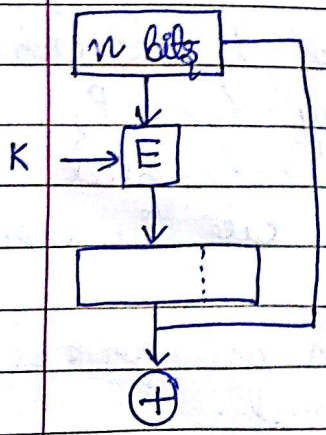
how many iterations ? $(n/r)+1$.

↓

if C_1 has 1 bit error

Output feedback

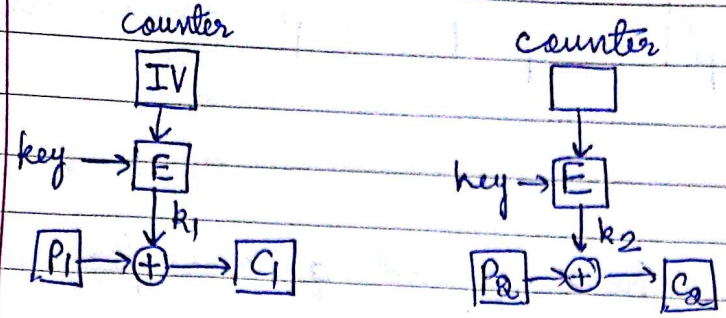




20/9/19.

Page No.:
Date:

Counter mode



content of counter is encrypted to generate random key.

Stream Ciphers

RC4 / A5/1. (self study)

RC4 → byte oriented, used in data communication and N/W protocols
Eg: SSL/TLS, IEEE802.11

A5/1 → used in GSM technologies

Asymmetric Key Cryptography

symmetric key cryptography : sharing secrecy, symbols are permuted / substituted

asymmetric key cryptography : personal secrecy, numbers are manipulated.

Keys : private and public

locking done using public key
unlocking done using private key
] for confidentiality

sender $\xrightarrow{f}$ receiver

$x \xrightarrow{f} y \Rightarrow g(t, y) = x$ (t → trapdoor)

only receiver has t, sender won't be able to decrypt y, even when it knows x.

$\phi(n) = |Z_n^*|$

$\phi(p) = |Z_p^*| = p-1$ p: prime number

RSA

Let $n = p * q$, where p and q are primes

Let $P = C = Z_n$ and define

P: plaintext
C: ciphertext

$K = \{(n, p, q, a, b) : ab \equiv 1 \pmod{\phi(n)}\}$

a and b : multiplicative inverses in $\phi(n)$

Encryption : $PT = x$, $CT = y$.

$y = E_k(x) = x^b \pmod n$

b: public key
Sender key.

Decryption : $x = D_k(y) = y^a \pmod n$

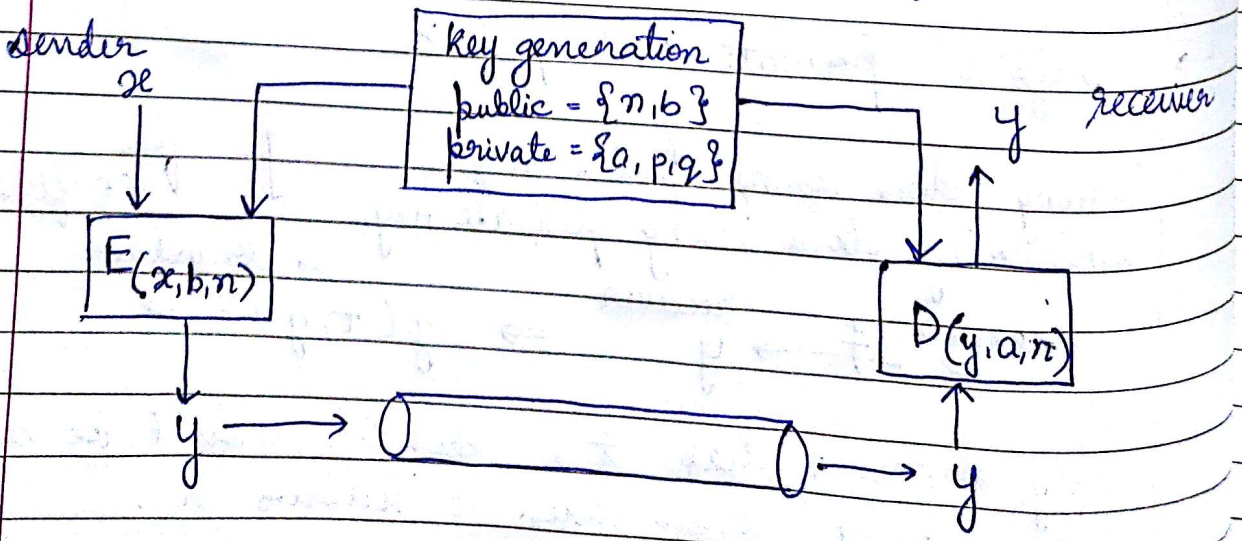
$= x^{ab} \pmod n$

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

$= x \pmod n$

a: private key
Receiver key

public key = $\{n, b\}$, private key = $\{p, q, a\}$



$\phi(n) = |Z_n^*|$ — Difficult to compute
no polynomial time algo, base of asymm key cipher.

$n = p * q$ (receiver does key generation)

$\phi(n) = \phi(p) \cdot \phi(q)$
$= (p-1)(q-1)$
↙ receiver knows

For attacker, if she doesn't know private key 'a', then she must find inverse of public key 'b' in $Z\phi(n)$ and $\phi(n) = \phi(p-1) \cdot \phi(q-1)$.

For computing $\phi(n)$, attacker requires values p and q. If factors of n are unknown and very large then factoring n is not feasible
— problem of factorisation

However, if factors of p and q are known, then there is efficient algo for computing modulo.

+ve integers → primes and composites

Availability of prime nos should be infinite since asymmetric key ciphers heavily use primes.

Euler's Phi Function

elements relatively prime to n.
elements in Z_n^*

(i) $\phi(1) = 0$
(ii) $\phi(p) = (p-1)$ if p is prime
(iii) $\phi(m*n) = \phi(m) * \phi(n)$ if m, n are coprimes

(iv) $\phi(p^e) = p^e - p^{e-1}$, if p is prime.

$n = p_1^{e_1} \cdot p_2^{e_2} \cdot p_3^{e_3} \cdots p_k^{e_k}$

$\phi(n) = (p_1^{e_1} - p_1^{e_1-1})(p_2^{e_2} - p_2^{e_2-1}) \cdots$

$n = 15$, $p = 3$, $p' = 5$, $e_1 = e_2 = 1$.

$\phi(n) = (3^1 - 3^0)(5^1 - 5^0) = 2 \times 4 = 8$.

Difficulty of finding $\phi(n)$ depends on that of factorising n.

For $n > 2$, $\phi(n)$ is even.

24/9/19.

Page No.:
Date:
YOUVA

Fermat Little Theorem

* 1st version

$a^{p-1} \equiv 1 \bmod p$, where p is prime, a and p are relatively prime

* 2nd version

$a^p \equiv a \bmod p$, where a & p are relatively prime.

Application

1) Exponential

Ex) $n = 17,\ a = 9,\ e = 50$

$9^{16} \equiv 1 \bmod 17$

$9^{50} \equiv (9^{16})^3 \times 9^2 = 1^3 \times 9^2$

$\equiv 81 \bmod 17.$

2) Inverse.

$a \cdot a^{-1} \equiv 1 \bmod p \equiv a^{p-1}.$

$a^{-1} \equiv a^{p-2} \bmod p.$

$a = 6$
$p = 17$

$a^{-1} = 6^{15} \bmod 17.$

$=$

Euler's Theorem

★ Generalisation of Fermat Little Theorem.

Modulus in Fermat Little Theorem is prime whereas that in Euler's Theorem is an integer

1) $a^{\phi(n)} \equiv 1 \bmod n$.

2) If $n = p \times q$, $a < n$ and k is an integer

$a^{k*\phi(n)+1} \equiv a \pmod n$ [Used in RSA]

Three cases are possible

a) a is neither a multiple of p nor q, then a and n are coprime $(n = p*q)$.

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

$\equiv 1^k \times a^1 \pmod n$.

$\equiv a \pmod n$.

b) a is a multiple of p but not of q. $(a = i*p)$

$a^{\phi(n)} \bmod q \equiv (a^{\phi(q)} \bmod q)^{\phi(p)} \bmod q = 1$.

↓ not mod n. because $n = pq$ and $a = ip$.

$\Rightarrow a^{\phi(n)} \bmod q \equiv 1 \bmod n$.

$a^{k*\phi(n)} \bmod q \equiv (a^{\phi(n)})^k \bmod q$.

$\equiv 1 \bmod q$.

$\equiv 1$.

$a^{k*\phi(n)} \bmod q = 1 + j*q$.

$$a^{k*\phi(n)+1} \bmod q = a^{1} * (1 + j*q).$$

$$= a + ajq$$

$$= a(1 + jq)$$

$$= ip(1 + qj)$$

$$= a + ipjq$$

$$= a + ijn.$$

$$a^{k*\phi(n)+1} \equiv a + (ij)n.$$

$$\Downarrow$$

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

$$\downarrow$$

congruence relation we will get.

c) a is a multiple of q but not of p. ($a = j*q$)

proof of correctness for RSA algorithm.

RSA CryptoSystem

Rewrite 4 lines of RSA.

$y = E_k(x) = x^b \bmod n.$

$x = D_k(y) = y^a \bmod n.$

$k_{public} = \{n, b\}.$

$k_{pvt} = \{p, q, a\}$; $x, y \in Z_n.$

Proof of correctness

$a * b \equiv 1 \pmod{\phi(n)}$

$a * b = 1 + t * \phi(n)$, $t \geq 1$

Suppose

$x \in Z_n^*$ $\Rightarrow$ n and x are coprime

$x^{ab} \equiv x^{1+t*\phi(n)} \pmod n.$

$\equiv x \cdot x^{\phi(n)*t} \pmod n.$

$\equiv x \cdot (x^{\phi(n)})^t \pmod n$

$\equiv x \cdot 1^t \pmod n$ (Fermat Little Theorem) → (Euler's Th^m)

$\star\star \cdot x^{ab} \equiv x. \pmod n.$

Suppose

$x \in Z_n - Z_n^*$ $\Rightarrow$ x and n are not coprime. $\star\star$

$p * q = n.$

$x \in Z_n = \{0, 1, \ldots n-1\}.$

Since x and n are not coprime,

1) x is a multiple of p.
2) x is a multiple of q.

If x is a multiple of both p and q, $x \notin Z_n$ because $x \geq n$.

Page No.:
Date:
YOUVA

1) x is a multiple of p.

$gcd(x, pq) = p$.

$gcd(x, q) = 1$.

Apply fermat little theorem

$$x^{\phi(n)} \equiv 1 \pmod{q}.$$

$$x^{t \cdot \phi(p) \cdot \phi(q)} \equiv 1 \pmod{q}$$

$$x^{t\phi(n)} \equiv 1 \pmod{q}$$

$$\Rightarrow \quad x^{t\phi(n)} = 1 + kq, \quad k \text{ is an integer.}$$

Multiply both sides by x.

$$x^{t\phi(n)+1} = x + xkq.$$

Since $gcd(x, p) = p \Rightarrow x = cp, \quad c \geq 1.$

$$\therefore \quad x^{\phi(n)*t+1} = x + xckpq.$$

$$\Rightarrow \quad x^{t\phi(n)+1} = x^{ab} \equiv x \pmod{n}.$$

25/9/19

Page No.:
Date:

Primality Testing

Deterministic Algorithm

- ★ Divisibility Algo $O(2^{n_b})$. if (prime) no. to test is n_b bits long.
- ★ AKS.
  $(x-a)^n \equiv (x^n - a) \bmod n$.
- ★★ $O[(\log n_b)^{12}]$

Probabilistic Algorithm

- ★ Fermat Test
  If n is prime, $a^{n-1} \equiv 1 \pmod n$

  But $2^{561-1} \equiv 1 \bmod 561$ passes Fermat test but 561 is not a prime
  $33 * 17$

- ★ Square-root test

  If n is prime, $\sqrt{1} \bmod n = \pm 1$.
  If n is composite, $\sqrt{1} \bmod n = \pm 1$ and possibly more values.

$x \in Z_6$

x	x^2	$\sqrt{x^2}$
1	1	$\sqrt{1} = 1$
2	4	$\sqrt{4} = 2$
3	3	$\sqrt{3} = 3$
4	4	$\sqrt{4} = 4$
5	1	~~$\sqrt{5} \neq 5$~~ → $\sqrt{1} = 5 = -1$.

$x \in Z_7$

x	x^2	$\sqrt{x^2}$
1	1	$\sqrt{1} = 1$
2	4	$\sqrt{4} = 2$
3	2	$\sqrt{2} = 3$
4	2	$\sqrt{2} = 4 = -3$
5	4	$\sqrt{4} = 5 = -2$
6	1	$\sqrt{1} = 6 = -1$

$\therefore \sqrt{1} \bmod 7 = \pm 1$.

$x \in Z_8$

x	$x^2 = y$	$\sqrt{y}$
1	1	$\sqrt{1} = \boxed{1}$
2	4	$\sqrt{4} = 2$
3	1	$\sqrt{1} = \boxed{3}$
4	0	$\sqrt{0} = 4$
5	1	$\sqrt{1} = \boxed{5}$ $\boxed{-3}$
6	4	$\sqrt{4} = 6$
7	1	$\sqrt{1} = \boxed{7}$ $\boxed{-1}$.

$\sqrt{1} \bmod 8 = \pm 1, \pm 3$.

For $x \in Z_7$, 7 is prime

$\sqrt{1} = \pm 1$.

For $x \in Z_6$,

$\sqrt{1} = \pm 1$. but 6 is not prime.

$\therefore$ Sometimes square root test fails.

★ Miller - Robin Test
(Fermat test + square root test)

$O(\log n_b)^3$.

Represent $n-1 = m \cdot 2^k$. where m is odd.

Fermat test $a^{n-1} \equiv 1 \pmod n$, n is prime

$n-1 = m \cdot 2^k$

$\therefore a^{m \cdot 2^k} \equiv 1 \pmod n$ if n is prime

Let $a^m = t$ (k times)

$((t^2)^2)^{2 \cdots}$

			if ↓	
1st iteration	t^2	t_1		$\Rightarrow \sqrt{t_3} = t_2$
2nd	t_1^2	t_2	$\neq \pm 1$	$\sqrt{1} \neq \pm 1$
3rd	t_2^2	$t_3 = 1$		some other value.

∴ n is composite.

⇒ We can terminate the loop

Choose random no. 'a', $1 \leq a \leq n-1$

$a^{n-1} = a^{m \cdot 2^k}$

$= ((a^m)^2)^{2 \cdots}$ k times

```
Algo_MRT (n, a)
{
    Find m, k s.t. a^(n-1) = m.2^k.
    T ≡ a^m mod n.
    if (T == ±1)
        then return (prime)
```

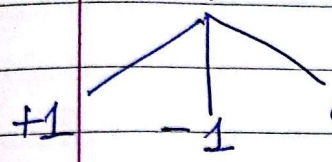
] other value ⇒ continue; +1, −1 ⇒ break

```
    for (i = 1 to k-1)
    {
        T = T^2 mod n.
        if (T == +1)
            then return (composite)
        if (T == (-1))
            then return (prime).
    }
    return (composite)
}
```

(k−1) iterations
(−1) : prime in next iteration it will become 1, remain 1.
(+1) : composite if it was other value in prev — fail sq. root test.
other : continue

Page No.:
Date:

$T = a^m \bmod n$



Iteration 1.

Iteration 2

k^{th} iteration $\neq (-1) \rightarrow$ fails Fermat
Passes square root

01/10/19.

Page No.:
Date:

Select good number of biases (different values of a for primality test) and perform primality test for integer 'n' using all those biases.

Solving set of equations with one variable with different modulus.

$x \equiv a_1 \bmod n_1$
$x \equiv a_2 \bmod n_2$
$\vdots$
$x \equiv a_k \bmod n_k$

If n is prime

If n is composite

n : prime — apply Chinese Remainder Theorem (CRT)

CRT.

1) Find $N = n_1 \cdot n_2 \cdot n_3 \cdots n_k$
2) Find $N_1 = N/n_1$, $N_2 = N/n_2$ ....
3) Find N_1^{-1}, N_2^{-1}, ...
4) $x = (a_1 N_1 N_1^{-1} + a_2 N_2 N_2^{-1} + \ldots) \bmod N$.

$x \equiv 2 \pmod 3$
$x \equiv 3 \pmod 5$
$x \equiv 2 \pmod 7$

$N = 3 * 5 * 7$
$= 105$
$N_1 = 35$ $N_1^{-1} = 2$
$N_2 = 21$ $N_2^{-1} = 1$
$N_3 = 15$ $N_3^{-1} = 1$.

x =

=

$N_1^{-1} \bmod 3 = 2$

$N_2^{-1} \bmod 5 = 1$

$N_3^{-1} \bmod 7 = 1$

$z = x + y$

$x = 120$

$y = 133$

$x \equiv 120 \bmod 98$

$x \equiv 22 \bmod 98$

$= \underline{22}$

$y \equiv 133 \bmod 98$

$= \underline{35}$

$z = (22 + 35) \bmod 98$

$x \equiv 120 \pmod{97}$

$x \equiv 120 \pmod{77}$

$y \equiv 133 \pmod{97}$

$y \equiv 133 \pmod{77}$

** ~~When~~ Factorise modulo into prime factors and apply CRT. — When n is composite.

Applications : Rabit Cryptosystem

Page No.:
Date:

Quadratic Congruence $x^2 \equiv a \pmod n$.

1) QC Modulo a prime

Z_6	x	x^2
	1	1
	2	4
	3	3
	4	4
	5	1.

Quadratic residue $\{1, 3, 4\}$ Quadratic non-residue $\{2, 5\}$

$x^2 \equiv 3 \pmod 6$ ✓

$x^2 \equiv 4 \pmod 6$ ✓

$x^2 \equiv 2 \pmod 6$ X.

$x^2 \equiv a \bmod n$.

$a \in QR$ (solution exists)

$a \in QNR$ (no solution exists)

Euler's Criteria

a) If $a^{(p-1)/2} \equiv 1 \pmod p$, then a is QR mod p.

b) If $a^{(p-1)/2} \equiv -1 \pmod p$, then a is QNR mod p.

Solving Quadratic Equation modulo prime

$p = 4k+1$ or $p = 4k+3$.

Special case : $p = 4k+3$ & $a \in QR$.

$x \equiv a^{(p+1)/4} \pmod p$

$x \equiv -a^{(p+1)/4} \pmod p$.

Z_{11}

x	x^2
1	1
2	4
3	9
4	5
5	3
6	3
7	5
8	9
9	4
10	1

$QR = \{1, 3, 4, 5, 9\}$

$QNR = \{2, 6, 7, 8, 10\}$.

For Z_p^*, $|QR| = (p-1)/2$.
$|QNR| = (p-1)/2$.

$x^2 \equiv 3 \pmod{23}$.

1) check if 23 is prime - yes.
2) check if 3 is QR.

$23 = 4k + 3$.

$3^{(22-1)/2} \equiv 1 \pmod{23}$.

$3^{11} \bmod 23 = 1 \bmod 23$ - yes.

$\therefore a \in QR$

$\Rightarrow x \equiv 3^{(24/4)} \pmod{23}$

$\Rightarrow x \equiv (3^6) \pmod{23}$

$\therefore x =$

Page No.:
Date:

$n \longrightarrow$ factorisation if it is composite

$x^2 \equiv a_1 \pmod{p_1}$ $\quad x \equiv \pm b_1 \pmod{p_1}$
$x^2 \equiv a_2 \pmod{p_2} \Rightarrow x \equiv \pm b_2 \pmod{p_2}$
$x^2 \equiv a_3 \pmod{p_3}$ $\quad x \equiv \pm b_3 \pmod{p_3}$

solve using CRT.

$x^2 \equiv 36 \pmod{77}$
$x^2 = 36 \pmod 7 \equiv 1 \pmod 7$
$x^2 \equiv 36 \pmod{11} \equiv 3 \pmod{11}$

Exponentiation and Logarithm

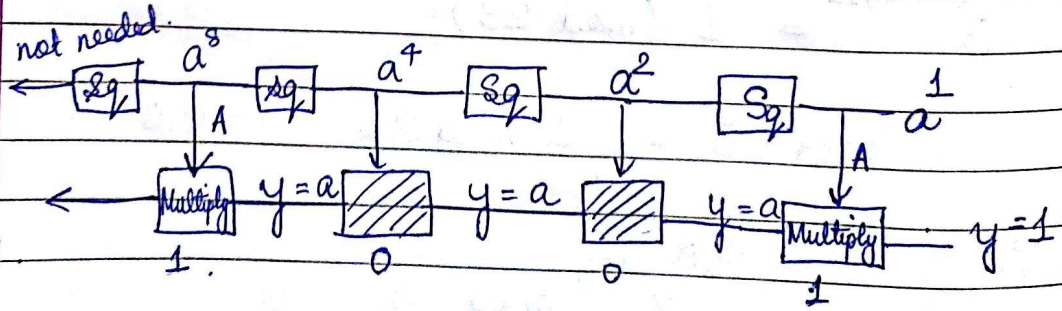
$z = a^x \Rightarrow$ logarithm $x = \log_a z$.

1) exponentiation : $z = a^x \pmod n$.

04/10/19 **Fast Exp Algorithm**

$z = a^x \Rightarrow$ Represent x as binary string (4b)

Eg: $y = a^b$.
$y = a^9$ $\qquad b = 9 = (1001)_b$
$= a^8 \times 1 \times 1 \times a^1$

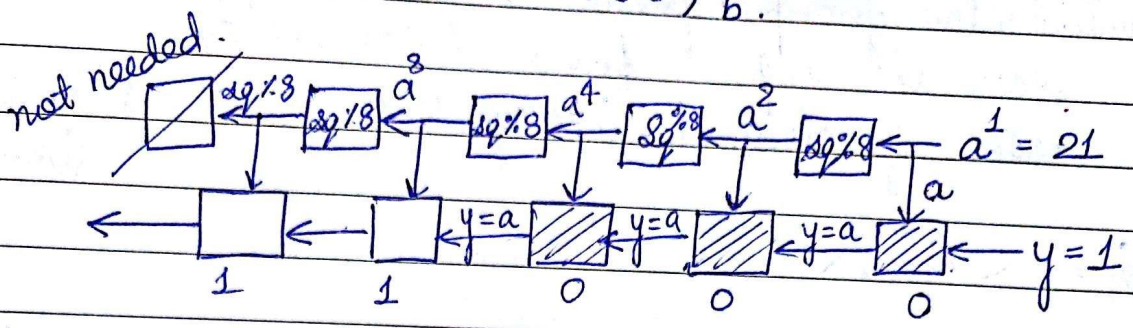


$y = a \cdot a^8$
$= a^9$

Every arithmetic operation will be % & n if exponentiation is % n.

$a^x \bmod n = 21^{24} \bmod 8$

$x = 24 = (11000)_b$.



Modular Logarithm

$y = a^x \bmod n$

$x = \log_a y$

If n is large, brute force in finding log will not be feasible

If encryption algo is exponential, decryption is logarithmic
- Create a table and do that for every possible bias (a).

Exhaustive search.

$O(2^{n_b})$ n_b bits to represent n.

$n = 6$ vary value of x from 1 to 5.

$a = 3$.

$y = 3^1 \bmod 6 = 3$ not one to one

$y = 3^2 \bmod 6 = 3$

$y = 3^3 \bmod 6 = 3$ mapping

⋮

Take $n = 7$.

x	$a^x \bmod n$.	
1	3	
2	2	one to one
3	6	mapping
4	4	
5	5	
6	1.	

Discrete Logarithm.

Multiplicative group. finite
order of group : finite (cardinality).
order of element (every bias) : $a^i = e$
i – least.

$G = \langle Z_{10}^*, \times \rangle$

$Z_{10}^* = \{1, 3, 7, 9\}$. Order of group = 4.

$1^1 = 1 \bmod 10 \Rightarrow$ Order (1) = 1.
$3^4 = 1 \bmod 10 \Rightarrow$ Order (3) = 4
$7^4 = 1 \bmod 10 \Rightarrow$ Order (7) = 4
$9^2 = 1 \bmod 10 \Rightarrow$ Order (9) = 2.

Euler's Theorem.

$a^{\phi(n)} \equiv 1 \bmod n$. if n is prime number.

$i < \phi(n)$, $a^i \equiv 1 \bmod n$. (atleast once)

For one a, different x^is, all values of Z_{10}^* are generated
$\Rightarrow$ a is a primitive root.

If order (an element) = $\phi(n)$, then that element is a root
$\phi(n) = 4$ $(|Z_{10}^*|)$.

element 1 will generate only {1} o/p space
element 9 will generate only {1, 9} o/p space

But we want I/p space = O/p space.

How many primitive roots present in the group of elements — because only these can be used as bias 'a' in an encryption algorithm.

Primitive root (used in ElGamal Cryptology)

In group $G_1 < Z_n^*, * >$ when the order of an element is the same as $\phi(n)$.

The group $G_1 = < Z_n^*, * >$ has primitive only if n is of the form $2, 4, p^t, 2p^t, \ldots$ $10 = 2 * 5^1$.

primitive roots $= \phi(\phi(n))$

cyclic group : $Z_n^* = \{ g^1, g^2 \ldots \}$.

Discrete Logarithm

$G_1 = < Z_p^*, \times >$

Its elements include all int from 1 to $p-1$

It has primitive root.

It is cyclic.

Eg: $p = 10$:

I/p	O/p
1	1
3	3
7	7
9	9

many to one. decryption/encryption - problematic

primitive root can be seen as $\log_a y$; base of log.

Traditional	Dis Log.
$\log_a 1 = 0$	$\log_a 1 \equiv 0 \pmod{\phi(n)}$.
$\log_a (x * y) = \log_a x + \log_a y$	$\log_a (x * y) \equiv \log_a x + \log_a y \pmod{\phi(n)}$
$\log_a x^k \equiv k \log_a x$	$\log_a (x^k) \equiv k \log_a x \pmod{\phi(n)}$