

R.M.K GROUP OF ENGINEERING INSTITUTIONS

R.M.K GROUP OF INSTITUTIONS





Please read this disclaimer before proceeding:

This document is confidential and intended solely for the educational purpose of RMK Group of Educational Institutions. If you have received this document through email in error, please notify the system manager. This document contains proprietary information and is intended only to the respective group / learning community as intended. If you are not the addressee you should not disseminate, distribute or copy through e-mail. Please notify the sender immediately by e-mail if you have received this document by mistake and delete this document from your system. If you are not the intended recipient you are notified that disclosing, copying, distributing or taking any action in reliance on the contents of this information is strictly prohibited.

**22CS701- CRYPTOGRAPHY AND CYBER SECURITY
(Lab Integrated)
Unit-2**

Department: Computer Science and Engineering

Batch/Year : 2022-2026/IV



Created by:

Dr. A.Thilagavathy/ Associate Professor/CSE/RMKEC

Dr.D.Naveen Raju /Associate Professor/CSE/RMKEC

Ms.K.Padma Priya/ Associate Professor/CSE/RMDEC

Ms. J. Sherine Glory/Assistant Professor/CSE/RMDEC

Dr.Anish T P/Associate Professor/CSE/RMKCET

Mr.Vinoth Kumar V/ Assistant Professor/CSE(CS)/RMKCET

Date: 12.05.2025

Table of Contents

S NO	CONTENTS	PAGE NUMBER
1	Contents	1
2	Course objectives	6
3	Pre Requisites (Course Names with Code)	6
4	Syllabus (With Subject Code, Name, LTPC details)	7
5	Course outcomes	9
6	CO- PO/PSO Mapping	10
7	Lecture Plan	11
8	Activity based learning	14
9	Lecture Notes	16
10	Assignments	62
11	Part A Q & A	67
12	Part B Qs	75
13	GATE Questions	76
14	Supportive online Certification courses	85
15	Real time Applications in day to day life and to Industry	86
16	Mini Project Suggestions	87
17	Assessment Schedule	89

22CS701-CRYPTOGRAPHY AND CYBER SECURITY (Lab Integrated)

COURSE OBJECTIVES

The Course will enable learners to:

- Understand the fundamentals of network security and security architecture.
- Learn the different symmetric key cryptographic algorithms.
- Study the various asymmetric key cryptographic algorithms and techniques.
- Know the importance of message authentication and integrity.
- Learn the various cyber-crimes and cyber security

PREREQUISITE

- 22MA201 Transforms and Numerical Methods
- 2 2CS501 Computer Networks



22CS701 CRYPTOGRAPHY AND CYBER SECURITY (Lab Integrated)

SYLLABUS

22CS701 CRYPTOGRAPHY AND CYBER SECURITY

3 0 2 4

UNIT I INTRODUCTION TO SECURITY

9+6

Computer Security Concepts – The OSI Security Architecture – Security Attacks – Security Services and Mechanisms – A Model for Network Security – Classical encryption techniques: Substitution techniques, Transposition techniques, Steganography –Foundations of modern cryptography: Perfect security – Information Theory – Product Cryptosystem – Cryptanalysis.

List of Exercise/Experiments:

1. Perform encryption, decryption using the following substitution techniques
(i) Ceaser cipher, (ii) playfair cipher iii) Hill Cipher iv) Vigenere cipher
2. Perform encryption and decryption using following transposition techniques
i) Rail fence ii) row & Column Transformation

UNIT II SYMMETRIC CIPHERS

9+6

Number theory – Algebraic Structures – Modular Arithmetic - Euclid's algorithm – Congruence and matrices – Group, Rings, Fields, Finite Fields SYMMETRIC KEY CIPHERS: SDES – Block Ciphers – DES, Strength of DES – Differential and linear cryptanalysis – Block cipher design principles – Block cipher mode of operation – Evaluation criteria for AES – Pseudorandom Number Generators – RC4 – Key distribution.

List of Exercise/Experiments:

1. Apply DES algorithm for practical applications.
2. Apply AES algorithm for practical applications.

UNIT III ASYMMETRIC CRYPTOGRAPHY 9+6

MATHEMATICS OF ASYMMETRIC KEY CRYPTOGRAPHY: Primes – Primality Testing

– Factorization – Euler's totient function, Fermat's and Euler's Theorem – Chinese Remainder Theorem – Exponentiation and logarithm ASYMMETRIC KEY CIPHERS: RSA cryptosystem – Key distribution – Key management – Diffie Hellman key exchange -- Elliptic curve arithmetic – Elliptic curve cryptography.

List of Exercise/Experiments:

1. Implement RSA Algorithm using HTML and JavaScript.
2. Implement the Diffie-Hellman Key Exchange algorithm for a given problem.
3. Calculate the message digest of a text using the SHA-1 algorithm.

COURSE OUTCOMES

CO1	Understand cryptographical concepts.
CO2	Implement various cryptographic algorithms
CO3	Evaluate and apply network security protocols, to secure communications over networks
CO4	Identify common security threats and vulnerabilities and assess their impact on network security
CO5	Implement access control mechanisms and authentication techniques to protect information systems.
CO6	Develop and propose security policies and best practices for securing networks and information systems

✿ CO-PO MAPPING

COs	PO's/PSO's														
	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	PSO 1	PSO 2	PSO 3
CO1	3	3	3	2	2	-	-	2	2	1	-	2	2	2	1
CO2	3	3	3	2	2	-	-	2	2	1	-	2	2	2	1
CO3	3	3	3	2	2	-	-	2	2	1	-	2	2	2	1
CO4	3	3	3	2	2	-	-	2	2	1	-	2	2	2	1
CO5	3	3	3	2	2	-	-	2	2	1	-	2	2	2	1
CO6	3	3	3	2	2	-	-	2	2	1	-	2	2	2	1

1 – Low, 2 – Medium, 3 – Strong



R.M.K.
GROUP OF
INSTITUTIONS

LECTURE

S No	Topics	No of periods	Proposed date	Actual Lecture Date	Pertaining CO	Taxonomy level	Mode of delivery
1	Course objective, course outcome delivery & Course introduction	1	25.1.2024		CO2	K1	ICT Tools
2	Algebraic structures – Modular arithmetic-Euclid’s algorithm- Congruence and matrices	1	27.1.2024		CO2	K3	ICT Tools
3	Groups, Rings, Fields- Finite fields	1	29.1.2024		CO2	K3	ICT Tools
	SDES	1	30.1.2024				ICT Tools
4					CO2	K2	
5	-Block cipher Principles of DES Strength of DES	1	31.1.2024		CO2	K3	ICT Tools
6	Differential and linear cryptanalysis	1	01.2.2024		CO2	K2	ICT Tools
7	Block cipher design principles	1	03.2.2024		CO2	K2	ICT Tools
8	Block cipher mode of operation	1	06.2.2024		CO2	K2	ICT Tools
9	Evaluation criteria for AES – Advanced Encryption Standard	1	07.2.2024		CO2	K2	ICT Tools

LECTURE PLAN

S No	Topics	No of periods	Proposed date	Actual Lecture Date	Pertaining CO	Taxonomy level	Mode of delivery
10		1	03.01.2024		CO2	K2	ICT Tools
11		1	04.01.2024		CO2 CO2 CO2	K2	ICT Tools
12		1	06.01.2024		CO2	K2	ICT Tools
13		1	08.01.2024		CO2	K2	ICT Tools
14		1	09.01.2024		CO2	K3	ICT Tools
15		1	10.01.2024		CO2	K3	ICT Tools
16		1	11.01.2024		CO2	K2	ICT Tools

VIDEO LINKS

S.NO	TOPICS	Link
1	SDES	https://www.youtube.com/watch?v=QcKHfMgcnbw
2	Advanced Encryption Standard	https://www.youtube.com/watch?v=WLM5_cxeywY
3	Euclid Algorithm	https://www.youtube.com/watch?v=cOwyHTiW4KE
4	Block cipher modes of operation	https://youtu.be/VmNk-6GTqRE?si=c3UgnMN24pv-b5IA
5	Group and Abelian Group	https://youtu.be/8TjYHK804mU?si=ogSF_YYW44VKVZ7c



GROUP OF
INSTITUTIONS

ACTIVITY BASED LEARNING

1. Algebraic Structures (Groups, Rings, Fields, Finite Fields)

Activity: Math Sorting Game

- Provide a list of sets with operations.
- Students identify whether each is a group, ring, field, or none.
- Use a drag-and-drop or color-coded card system.

2. Euclid's Algorithm

Activity: GCD Race

- Students race to find GCD of given number pairs using Euclid's algorithm step-by-step.
- Bonus: Implement it in Python or C for a quick coding demo.

3. Block Cipher Modes of Operation

Activity: Mode Chain Puzzle

- Simulate ECB, CBC, CFB, OFB using a message broken into blocks.
- Give "plaintext cards," "key cards," and "IV cards" for students to perform the operations.
- Show weaknesses of ECB with repeated patterns.

4. Key Distribution Role-Play

Activity: Trusted Authority Scenario

- Assign students roles (Alice, Bob, Eve, KDC, attacker).
- Simulate a key distribution protocol (Needham-Schroeder, or basic symmetric key sharing).
- Include attacks like replay or man-in-the-middle to challenge students to think critically.

5. Cryptanalysis Simulation – Find the Pattern!

Activity: Find correlation

- Provide two ciphertexts and their related plaintexts.
- Ask students to find correlations (simulate linear or differential cryptanalysis).
- Use frequency tables, XOR differences, and provide guided hints.

ALGEBRAIC STRUCTURES

Abstract Algebra or Modern Algebra is a mathematical field which deals with study of algebraic structures. An algebraic structure consists of a nonempty set A, a collection of operations on A of finite arity (typically binary operations), and a finite set of identities, known as axioms, that these operations must satisfy.

❖ Set

Set is a well-defined collection of distinct objects. The individual objects that make up a set are called its elements.

Finite Set: It has finite number of elements.

Example: $\{1, 2, 3, 4, 5, 6\}$. It is finite and has cardinality of 6.

Infinite Set: A set that is not finite is called infinite.

Example: $Z = \{-3, -2, -1, 0, 1, 2, 3, \dots\}$

❖ Operations

Arity is the number of arguments or operands a function or operation takes. A binary operation is an operation of arity two. Examples of binary operations includes arithmetic operations like addition, subtraction, multiplication.

GROUPS

A group G, sometimes denoted by $\{G, \cdot\}$ is a set of elements with a binary operation, denoted by \cdot , that associates to each ordered pair (a, b) of elements in G an element $(a \cdot b)$ in G, such that the following axioms are obeyed:



(A1) Closure: If a and b belong to G, then $a \cdot b$ is also in G.

(A2) Associative: $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ for all a, b, c in G.

(A3) Identity element: There is an element e in G such that $a \cdot e = e \cdot a = a$ for all a in G.

(A4) Inverse element: For each a in G there is an element a' in G such that $a \cdot a' = a' \cdot a = e$.

If a group has a finite number of elements, it is referred to as a finite group, and the order of the group is equal to the number of elements in the group. Otherwise, the group is an infinite group.

ABELIAN GROUP

A group is said to be abelian if it satisfies the following additional condition:

❖ **(A5) Commutative:** $a \# b = b \# a$ for all a, b in G .

The set of integers (positive, negative, and 0) under addition is an abelian group.

The set of nonzero real numbers under multiplication is an abelian group.

CYCLIC GROUP

A group G is cyclic if every element of G is a power a^k (k is an integer) of a fixed element $a \in G$. The element a is said to generate the group or to be a generator of G . A cyclic group is always abelian and may be finite or infinite.

RINGS

A ring R , sometimes denoted by $\{R, +, \cdot\}$, is a set of elements with two binary operations, called addition and multiplication, such that for all a, b, c in R the following

axioms are obeyed:

❖ **(A1-A5)** R is an abelian group with respect to addition; that is, R satisfies axioms A1 through A5. For the case of an additive group, we denote the identity element as 0 and the inverse of a as $-a$.

❖ **(M1) Closure under multiplication:** If a and b belong to R , then ab is also in R .

❖ **(M2) Associativity of multiplication:** $a(bc) = (ab)c$ for all a, b, c in R .

(M3) Distributive laws:

$$a(b + c) = ab + ac \text{ for all } a, b, c \text{ in } R.$$

$$(a + b)c = ac + bc \text{ for all } a, b, c \text{ in } R.$$

In essence, a ring is a set in which we can do addition, subtraction [$a - b = a + (-b)$], and multiplication without leaving the set.

With respect to addition and multiplication, the set of all n -square matrices over the real numbers is a ring.

COMMUTATIVE RING

A ring is said to be commutative if it satisfies the following additional condition:

✿ **(M4) Commutativity of multiplication:** $ab = ba$ for all a, b in R .

Let S be the set of even integers (positive, negative, and 0) under the usual operations

of addition and multiplication. S is a commutative ring.

INTEGRAL DOMAIN

An integral domain is a commutative ring that obeys the following axioms.

✿ **(M5) Multiplicative identity:** There is an element 1 in R such that $a1 = 1a = a$ for all a in R .

✿ **(M6) No zero divisors:** If a, b in R and $ab = 0$, then either $a = 0$ or $b = 0$.

Let S be the set of integers, positive, negative, and 0, under the usual operations of addition and multiplication. S is an integral domain.

FIELDS

A field F , sometimes denoted by $\{F, +, \times\}$, is a set of elements with two binary operations, called addition and multiplication, such that for all a, b, c in F the following axioms are obeyed:

✿ **(A1-M6)** F is an integral domain; that is, F satisfies axioms A1 through A5 and M1 through M6.

✿ **(M7) Multiplicative inverse:** For each a in F , except 0, there is an element a^{-1} in F such that $aa^{-1} = (a^{-1})a = 1$.

In essence, a field is a set in which we can do addition, subtraction, multiplication, and division without leaving the set. Division is defined with the following rule: a/b

$$= a(b^{-1}).$$

Familiar examples of fields are the rational numbers, the real numbers, and the complex numbers. Note that the set of all integers is not a field, because not every element of the set has a multiplicative inverse; in fact, only the elements 1 and -1 have multiplicative inverses in the integers.

FINITE FIELDS (GALOIS FIELD)

Finite Field is a field with finite number of elements. Order of the field is equal to the number of elements in the field. The finite field of order p^n is generally written GF(p^n).

GF (p)

$Z_p = \{0, 1, \dots, (p - 1)\}$ where p is prime.

GF (p) is the set Z_p together with arithmetic operations modulo p .

The binary operations $+$ and \times are defined over the set. The operations of addition, subtraction, multiplication, and division can be performed without leaving the set.

Each element of the set other than 0 has a multiplicative inverse.

+ E	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	2	3	4	5	6	0
2	2	3	4	5	6	0	1
3	3	4	5	6	0	1	2
4	4	5	6	0	1	2	3
5	5	6	0	1	2	3	4
6	6	0	1	2	3	4	5

Table: Addition Modulo
7

\times	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	4	6	1	3	5
3	0	3	6	2	5	1	4
4	0	4	1	5	2	6	3
5	0	5	3	1	6	4	2
6	0	6	5	4	3	2	1

Table: Multiplication Modulo
7

w	0	1	2	3	4	5	6
-w	0	6	5	4	3	2	1
w ⁻¹	-	1	4	5	2	3	6

Table: Additive and Multiplication Inverse Modulo 7

✿ Polynomial Arithmetic

A polynomial of degree n (integer $n \geq 0$) is an expression of the form

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = \sum_{i=0}^n a_i x_i$$

$$f(x) = \sum_{i=0}^n a_i x_i$$

$$g(x) = \sum_{i=0}^m b_i x_i$$

$$\begin{aligned} f(x) + g(x) &= \sum_{i=0}^{\min(n, m)} (a_i + b_i) x_i \\ &\quad + \sum_{i=\max(n, m)+1}^{\max(n, m)} b_i x_i \end{aligned}$$

- Multiplication

$$f(x) \times g(x) = \sum_{i=0}^{n+m} c_i x^i$$

$$\text{where } c_k = a_0 b_k + a_1 b_{k-1} + \dots + a_{k-1} b_1 + a_k b_0$$

Examples:

$$f(x) = x^3 + x^2 + 2 \text{ and } g(x) = x^2 - x + 1$$

$$f(x) + g(x) = x^3 + 2x^2 - x + 3$$

$$f(x) - g(x) = x^3 + x + 1$$

$$f(x) \times g(x) = x^5 + 3x^2 - 2x + 2$$

❖ Irreducible Polynomial

A polynomial $f(x)$ over a field is called irreducible if and only if $f(x)$ cannot be expressed as a product of two polynomials, both over F , and both of degree lower than that of

$f(x)$. By analogy to integers, an irreducible polynomial is also called a prime polynomial. The primary use of irreducible polynomials is in the construction of larger finite fields.

Example: The polynomial $f(x) = x^4 + 1$ over GF (2) is reducible, because

$$x^4 + 1 = (x+1)(x^3 + x^2 + x + 1)$$



The polynomial $f(x) = x^4 + x + 1$ over GF (2) is irreducible.

GF (2^n)

Addition of two polynomials in GF (2^n) corresponds to a bitwise XOR operation.

Example:
 $f(x) = x^6 + x^4 + x^2 + x + 1$
 $g(x) = x^7 + x + 1$

$$\begin{aligned} \text{Polynomial Notation: } & (x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) \\ &= x^7 + x^6 + x^4 + x^2 \end{aligned}$$

$$\text{Binary Notation: } 01010111 \oplus 10000011 = 11010100$$

$$\text{Hexadecimal Notation: } \{57\} \oplus \{83\} = \{D3\}$$

- Multiplication

Finite Field used in AES is GF (2^8) and irreducible polynomial used is $m(x) = x^8 + x^4 + x^3 + x + 1$.

$$x^8 \bmod m(x) = [m(x) - x^8] = x^4 + x^3 + x + 1$$

In general, in GF (2^n) with an nth-degree polynomial $p(x)$, we have

$$x^n \bmod p(x) = [p(x) - x^n]$$

Polynomial in GF (2^8), has the form:

$$f(x) = b_7 x_7 + b_6 x_6 + b_5 x_5 + b_4 x_4 + b_3 x_3 + b_2 x_2 + b_1 x + b_0$$

$$x \times f(x) = (b_7 x_8 + b_6 x_7 + b_5 x_6 + b_4 x_5 + b_3 x_4 + b_2 x_3 + b_1 x_2 + b_0 x) \bmod m(x)$$

$$\boxed{x \times f(x) \quad \begin{cases} (b_6 b_5 b_4 b_3 b_2 b_1 b_0)^0 & \text{if } b_7 = 0 \\ (b_6 b_5 b_4 b_3 b_2 b_1 b_0 0) \oplus 00011011 & \text{if } b_7 = 1 \end{cases}}$$

$$\text{Example: } f(x) = x^6 + x^4 + x^2 + x + 1 \quad g(x) = x^7 + x + 1$$

$$f(x) \times g(x) \bmod m(x) \Rightarrow (01010111) \times (10000011)$$

$$\Rightarrow (01010111) \times [(00000001) \oplus (00000010) \oplus (10000000)]$$

$$\Rightarrow [(01010111) \times (00000001)] \oplus [(01010111) \times (00000010)] \oplus$$

$$[(01010111) \times (10000000)]$$

$$f(x) \times x^7 \Rightarrow (01010111) \times (00000010) = 01010111 \quad b_7 = 0$$

$$f(x) \times x^2 = (f(x) \times x) \times x \Rightarrow f(x) \times x^7 = (f(x) \times x^6) \times x$$

$$f(x) \times x^2 \Rightarrow (10101110) \times (00000010) = (01011100) \oplus (00011011) \quad b_7 = 1$$

$$f(x) \times x^3 \Rightarrow (01000111) \times (00000010) = 10001110$$

$$f(x) \times x^4 \Rightarrow (10001110) \times (00000010) = (00011100) \oplus (00011011) = 00000111$$

$$f(x) \times x^5 \Rightarrow (00000111) \times (00000010) = 00001110$$

$$f(x) \times x^6 \Rightarrow (00001110) \times (00000010) = 00011100$$

$$f(x) \times x^6 \Rightarrow (00011100) \times (00000010) = 00111000$$

$$\Rightarrow [(01010111) \times (00000001)] \oplus [(01010111) \times (00000010)] \oplus [(01010111) \times$$

$$(10000000)] = 01010111 \oplus 10101110 \oplus 00111000 = 11000001$$

$$f(x) \times g(x) \bmod m(x) = x^7 + x^6 + 1$$

Applications of Galois Field

- Advanced Encryption Standard (AES): GF (2⁸)
- Elliptic Curve Cryptography
- CMAC
- Galois/Counter Mode of operation (GCM): GF (2¹²⁸)

MODULAR ARITHMETIC

❖ Divisors

Given positive integers a and b , we use the notation $a|b$ to indicate that a divides b , that is, b is a multiple of a . If $a|b$, then we know that there is some integer k , such that $b=ak$. The following properties of divisibility follow immediately from this definition.

- If $a|b$ and $b|c$, then $a|c$.
- If $a|b$ and $a|c$, then $a|(ib+jc)$, for all integers i and j .
- If $a|b$ and $b|a$, then $a=b$ or $a=-b$.
- Any $a \neq 0$ divides 0.

Examples: The positive divisors of 24 are 1, 2, 3, 4, 6, 8, 12, and 24.

$13|182$; $-5|30$; $17|289$; $-3|33$; $17|0$

❖ Floor Function

The floor function $\lfloor x \rfloor$, also called the greatest integer function or integer value, gives the largest integer less than or equal to x .

Examples:

$$\lfloor 10/2 \rfloor = \lfloor 2.5 \rfloor = 2$$

$$\lfloor -10/2 \rfloor = \lfloor -2.5 \rfloor = -3$$

❖ Division algorithm

Given any positive integer n and any nonnegative integer a , if we divide a by n , we get an integer quotient q and an integer remainder r that obey the following relationship:

$$a = qn + r \quad 0 \leq r < n; q = \lfloor a/n \rfloor$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x . The above equation is referred to as the division algorithm.

MODULUS

If a is an integer and n is a positive integer, we define $a \bmod n$ to be the remainder when a is divided by n . The integer n is called the modulus.

$$a = qn + r \quad 0 \leq r < n; \quad q = \lfloor a/n \rfloor$$

$$a = \lfloor a/n \rfloor \times n + (a \bmod n)$$

Examples: $-13 \bmod 7 =$

$$\begin{aligned} -13 &\equiv \lfloor -13/7 \rfloor \times 7 + 6 = \lfloor -1.857 \rfloor \times 7 + 6 = 1 \times 7 + 6 \\ -13 &= \lfloor -13/7 \rfloor \times 7 + 1 = \lfloor -1.857 \rfloor \times 7 + 1 = -2 \times 7 + 1 \end{aligned}$$

✿ Congruence

Two integers a and b are said to be congruent modulo n , if $(a \bmod n) = (b \bmod n)$.

This is denoted as $a \equiv b \pmod{n}$.

✿ Properties of Congruences

1. $a \equiv b \pmod{n}$ if $n | (a-b)$.
2. $a \equiv b \pmod{n}$ implies $b \equiv a \pmod{n}$.
3. $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n}$ implies $a \equiv c \pmod{n}$

Examples:

$$29 \equiv 4 \pmod{5} \quad 5 | (29-4) \Rightarrow 5 |$$

$$29 \equiv 4 \pmod{5} \quad 25$$

$$29 \equiv 4 \pmod{5} \text{ and } 4 \equiv 24 \pmod{5} \Rightarrow 29 \equiv 24 \pmod{5}$$

✿ Modular Arithmetic Operations

Modular arithmetic exhibits the following properties:

1. $[(a \bmod n) + (b \bmod n)] \bmod n = (a + b) \bmod n$
2. $[(a \bmod n) - (b \bmod n)] \bmod n = (a - b) \bmod n$
3. $[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$

Examples:

Addition:

$$[(12 \text{ mod } 7) + (13 \text{ mod } 7)] \text{ mod } 7 = (12 + 13) \text{ mod } 7$$

$$(5+6) \text{ mod } 7 = 25 \text{ mod } 7$$

$$\Rightarrow 11 \text{ mod } 7 = 25 \text{ mod } 7$$

$$\Rightarrow 11 \text{ mod } 7 = 4 \Rightarrow 11 \equiv 4 \text{ mod } 7$$

Subtraction

$$[(12 \text{ mod } 7) - (13 \text{ mod } 7)] \text{ mod } 7 = (12 - 13) \text{ mod } 7$$

$$(5-6) \text{ mod } 7 = -1 \text{ mod } 7 \Rightarrow -1 \equiv 6 \text{ (mod } 7)$$

Multiplication

$$[(12 \text{ mod } 7) \times (13 \text{ mod } 7)] \text{ mod } 7 = (12 \times 13) \text{ mod } 7$$

$$(5 \times 6) \text{ mod } 7 = 156 \text{ mod } 7$$

$$30 \text{ mod } 7 = 156 \text{ mod } 7 \Rightarrow 2 \text{ mod } 7 = 2 \text{ mod } 7$$

Exponentiation

$$7^{19} \text{ mod } 11$$

$$(7^{16} \times 7^2 \times 7^1) \text{ mod } 11 = (7^{16} \text{ mod } 11 \times 7^2 \text{ mod } 11 \times 7^1 \text{ mod } 11) \text{ mod } 11$$

$$7^2 \text{ mod } 11 = 49 \text{ mod } 11 = 5 \Rightarrow 7^2 = 49 \equiv 5 \text{ mod } 11$$

$$7^4 \text{ mod } 11 = (7^2 \text{ mod } 11 \times 7^2 \text{ mod } 11) \text{ mod } 11$$

$$= 5 \times 5 \text{ (mod } 11) = 3 \text{ mod } 11 \Rightarrow 7^4 \equiv 3 \text{ mod } 11$$

$$7^{16} \text{ mod } 11 = (7^4 \text{ mod } 11 \times 7^4 \text{ mod } 11 \times 7^4 \text{ mod } 11 \times 7^4 \text{ mod } 11) \text{ mod } 11$$

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

$$= 81 \text{ (mod } 11) = 4 \Rightarrow 7^{16} \equiv 4 \text{ (mod } 11)$$

$$7^{19} \text{ mod } 11 = (7^{16} \times 7^2 \times 7^1) \text{ mod } 11$$

$$= (4 \times 5 \times 7) \text{ mod } 11 = 140 \text{ mod } 11 = 8$$

$$7^{19} \equiv 8 \text{ mod } 11 \text{ (or) } 7^{19} \text{ mod } 11 = 8$$

❖ Properties of Modular Arithmetic

- Residue Class (or) Congruence Class

The set (represented by $Z_n = \{0, 1, 2, \dots, (n-1)\}$) of nonnegative integers less than n

is referred to as the set of residues or residue classes modulo n or congruence classes

Example: The residue classes (mod 6)

are

$$[0] = \{\dots, -18, 12, -6, 0, 6, 12, 18, \dots\}$$

$$[1] = \{\dots, -17, -11, -5, 1, 7, 13, 19, \dots\}$$

$$[2] = \{\dots, -16, -10, -4, 2, 8, 14, 20, \dots\}$$

$$[3] = \{\dots, -15, -9, -3, 3, 9, 15, 21, \dots\}$$

Of all the integers in a residue class, the smallest nonnegative integer is the one

used to represent the residue class.

$$[4] = \{\dots, -14, -8, 2, 4, 10, 16, 22, \dots\} [5] = \{\dots, -13, -7, -1, 5, 11, 17, 23, \dots\}$$

- If $(a + b) \equiv (a + c) \pmod{n}$ then $b \equiv c \pmod{n}$

Example: $(7 + 23) \equiv (7 + 7) \pmod{8}$; $23 \equiv 7 \pmod{8}$

- If $(a \times b) \equiv (a \times c) \pmod{n}$ then $b \equiv c \pmod{n}$ if a is relatively prime to n
An integer has a multiplicative inverse in Z_n if that integer is relatively prime to n.

Property	Expression
Commutative Laws	$(w + x) \pmod{n} = (x + w) \pmod{n}$ $(w \times x) \pmod{n} = (x \times w) \pmod{n}$
Associative Laws	$[(w + x) + y] \pmod{n} = [w + (x + y)] \pmod{n}$ $[(w \times x) + y] \pmod{n} = [w \times (x \times y)] \pmod{n}$
Distributive Law	$[w \times (x + y)] \pmod{n} = [(w \times x) + (w \times y)] \pmod{n}$
Identities	$(0 + w) \pmod{n} = w \pmod{n}$ $(1 \times w) \pmod{n} = w \pmod{n}$
Additive Inverse ($-w$)	For each $w \in Z_n$, there exists a z such that $w + z = 0 \pmod{n}$

Table: Properties of Modular Arithmetic for Integers in

Z_n

EUCLID ALGORITHM

Euclidean algorithm is used for determining the greatest common divisor of two positive integers.

The notation $\text{gcd}(a, b)$ is used to mean the greatest common divisor of a and b . The greatest common divisor of a and b is the largest integer that divides both a and b . The $\text{gcd}(0, 0) = 0$.

- Since gcd is positive, $\text{gcd}(a, b) = \text{gcd}(|a|, |b|)$
- $\text{gcd}(a, 0) = |a|$ because all nonzero integers divide 0

❖ Relatively Prime or Co-Prime

Two integers a and b are relatively prime if their only common positive integer factor is 1. Equivalently we can say that if $\text{gcd}(a, b) = 1$, then a and b are relatively prime.

Example: $\text{gcd}(8, 9)$

Positive divisors of 8 are 1, 2, 4, and 8.

Positive divisors of 9 are 1, 3, and 9.

Only common positive divisor of 8 and 9 is 1. So, 8 and 9 are relatively prime.

❖ Euclid Algorithm Formula

Step 1: **$\text{gcd}(a, b) = \text{gcd}(b, a \bmod b)$**

Step 2: Step 1 is repeated till we get the form $\text{gcd}(a, 0)$

Step 3: **$\text{gcd}(a, 0) = a$ gives the $\text{gcd}(a, b)$**

Euclid Algorithm can be expressed as Recursive Function:

Euclid (a, b)

```
if (b=0) then return a;  
else return Euclid (b, a mod b);
```

#Problem 1: Determine $\text{gcd}(5264, 15472)$ using Euclid Algorithm.

Solution:

$$\text{gcd}(a, b) = \text{gcd}(b, a \bmod b) \quad \text{gcd}(a, 0) = a$$

$$\begin{aligned}
 \gcd(5264, 15472) &= \gcd(15472, 5264) \\
 \gcd(15472, 5264) &= \gcd(5264, 15472 \bmod 5264) \\
 &= \gcd(5264, 4944) \\
 &= \gcd(4944, 5264 \bmod 4944) \\
 &= \gcd(4944, 320) \\
 &= \gcd(320, 4944 \bmod 320) \\
 &= \gcd(320, 144) \\
 &= \gcd(144, 320 \bmod 144) \\
 &= \gcd(144, 32) \\
 &= \gcd(32, 144 \bmod 32) \\
 &= \gcd(32, 16) \\
 &= \gcd(16, 32 \bmod 16) \\
 &= \gcd(16, 0) \\
 &= 16 \\
 \gcd(5264, 15472) &= 16
 \end{aligned}$$

EXTENDED EUCLID ALGORITHM

It is used to find multiplicative inverse of an integer. If a and b are relatively prime, then b has a multiplicative inverse modulo a.

EXTENDED EUCLID (m, b)

1. $A_1, A_2, A_3 \square 1, 0, m$
2. $B_1, B_2, B_3 \square 0, 1, b$
If $B_3 = 0$ return no
3. If $B_3 = 1$ return $B_2 = b^{-1} \bmod m$
4. $Q = \lfloor A_3 / B_3 \rfloor$
5. $T_1, T_2, T_3 \square A_1 - QB_1, A_2 - QB_2, A_3 - QB_3$
6. $A_1, A_2, A_3 \square B_1, B_2, B_3$
7. $B_1, B_2, B_3 \square T_1, T_2, T_3$
8. Goto Step 2

#Problem 2: Find the multiplicative inverse of 826 mod

2789. Solution:

$m = 2789, b=826$

A_1	A_2	A_3	B_1	B_2	B_3	$Q = \lfloor A_3 / B_3 \rfloor$
1	0	m	0	1	b	
1	0	2789	0	1	826	3
0	1	826	1	-3	311	2
1	-3	311	-2	7	204	1
-2	7	204	3	-10	107	1
3	-10	107	-5	17	97	1
-5	17	97	8	-27	10	9
8	-27	10	-77	260	7	1
-77	260	7	85	-287	3	2
-85	87	3	-247	834	1	

MATRICES

Row vector X is represented as $(x_1 \ x_2 \ \dots \ x_m)$

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

Column vector is represented as

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

Matrix is represented as

In a matrix, the first subscript of an element refers to the row and the second subscript refers to the column.

❖ Operations

□ **Addition:** Two matrices of the same dimensions can be added or subtracted element by element. Thus, for $C = A + B$, the elements of C are $c_{ij} = a_{ij} + b_{ij}$.

Example:

$$\begin{pmatrix} 1 & -2 & 3 \\ -4 & 5 & -6 \\ 6 & 8 & 2 \end{pmatrix} + \begin{pmatrix} 3 & 5 \\ 8 & 9 \\ 2 \end{pmatrix} = \begin{pmatrix} 4 & 3 \\ 13 & 3 \\ 8 & 10 \end{pmatrix}$$

□ **Multiplication:** To multiply a matrix by a scalar, every element of the matrix is multiplied by the scalar. Thus, for $C = kA$, we have $c_{ij} = k \times a_{ij}$

$$\begin{pmatrix} 1 & -2 & 3 \end{pmatrix} \cdot 6 \begin{pmatrix} -12 & 18 \\ -36 & 54 \end{pmatrix} = \begin{pmatrix} -4 & 5 & -6 \\ 7 & -8 & 9 \\ 42 & -48 & 54 \end{pmatrix} = -2430$$

The product of a row vector of dimension m and a column vector of dimension m is a scalar:

$$(x_1 \ x_2 \ \dots \ x_m) \times \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} = x_1 y_1 + x_2 y_2 + \dots + x_m y_m$$

If A is an $m \times n$ matrix and B is an $n \times p$ matrix, the matrix product $C = AB$ is defined to be the $m \times p$ matrix. The product AB is defined if and only if the number of columns in A equals the number of rows in B , in this case n .

$$C_{ij} = \sum_{k=1}^n a_{ik} b_{kj}$$

Row Vector Matrix Multiplication

Example:

$$(6 \ 14 \ 3) \begin{pmatrix} 17 & 11 & 5 \\ 21 & 18 & 12 \\ 2 & 3 & 19 \end{pmatrix} = (6*17+14*21+3*2 \ 6*11+14*18+3*3 \\ 6*5+14*12+3*19) \\ = (402 \ 327 \\ 255)$$

Column Vector Matrix Multiplication Example:

$$\begin{pmatrix} 17 & 11 \\ 21 & 18 \\ 2 & 3 \end{pmatrix} \begin{pmatrix} 5 \\ 14 \\ 19 \end{pmatrix} = \begin{pmatrix} 17 * 6 + 11 * 14 + \\ 21 * 6 + 18 * 14 + 12 \\ 2 * 6 + 3 * 14 + 19 * 3 \end{pmatrix} * 3 = \begin{pmatrix} 271 \\ 111 \end{pmatrix}$$

Determinants

Determinant can be calculated only for a square matrix (matrix with equal number of rows and columns).

For a 2×2 Matrix A, $\det(A) = a_{11}a_{22} - a_{12}a_{21}$

For a 3×3 Matrix A, $\det(A) = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32}$

$- a_{31}a_{22}a_{13} - a_{32}a_{23}a_{11} - a_{33}a_{21}a_{12}$

Example: $A = \begin{pmatrix} 17 & 11 & 5 \\ 21 & 18 & 12 \\ 2 & 3 & 19 \end{pmatrix}$

$$\begin{aligned} \det(A) &= |A| = 17(18.19 - 12.3) - 11(21.19 - 12.2) + 5(21.3 - 18.2) \\ &= 17(306) - 11(375) + 5(27) = 5202 - 4125 + 135 = 1212 \end{aligned}$$

Inverse of a Matrix

If a matrix A has a nonzero determinant, then it has an inverse, denoted as A^{-1} .

$$A^{-1} = \frac{1}{\det(A)} \cdot \text{adj}(A)$$

Adjugate of A is the transpose of the cofactor matrix C of

$$A \quad \text{adj}(A) = C^T$$

For 2×2 matrix, Inverse is calculated

as $A^{-1} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$

Example: 2×2 matrix

$$\left(\begin{array}{cc} 9 & 4 \\ 5 & 7 \end{array} \right)^{-1} = \frac{1}{63 - 20} \begin{bmatrix} 7 & -4 \\ -5 & 9 \end{bmatrix} = \frac{1}{43} \begin{bmatrix} 7 & -4 \\ -5 & 9 \end{bmatrix}$$

Example: 3×3
matrix

$$\left(\begin{array}{ccc} 17 & 17 & 2 \\ 5 & 18 & 2 \\ 21 & 18 & 21 \end{array} \right)^{-1}$$

Let the matrix be A. Then,

$$\begin{aligned} \det(A) &= |A| = 17 [(18.19) - (21.2)] - 17 [(21.19) - (21.2)] + 5 [(21.2) - (18.2)] \\ &= 17 [342 - 42] - 17 [399 - 42] + 5 [42 - 36] = 5100 - 6069 + 30 = -939 \end{aligned}$$

$$|A|^{-1} = -939^{-1}$$

Transpose of A
 $A^T = \begin{pmatrix} 17 & 21 & 2 \\ 17 & 18 & 2 \\ 5 & 21 & 19 \end{pmatrix}$

$$+ \begin{vmatrix} 18 & 2 \\ 2 & 21 \\ 21 & 19 \end{vmatrix} = 300 - \begin{vmatrix} 17 & 2 \\ 2 & 5 \\ 5 & 19 \end{vmatrix} = -313 \quad + \begin{vmatrix} 17 & 18 \\ 5 & 21 \end{vmatrix} = 267 \quad - \begin{vmatrix} 21 & 2 \\ 21 & 19 \end{vmatrix} = -399$$

$$+ \begin{vmatrix} 17 & 2 \\ 5 & 19 \end{vmatrix} = 313 - \begin{vmatrix} 17 & 21 \\ 5 & 21 \end{vmatrix} = -252 \quad + \begin{vmatrix} 21 & 2 \\ 18 & 6 \end{vmatrix} = 6 \quad 2 \begin{vmatrix} 17 & 2 \\ 17 & 2 \end{vmatrix} = 0$$

$$+ \begin{vmatrix} 17 & 21 \\ 17 & 18 \end{vmatrix} = -51$$

$$A^{-1} = -939^{-1} \cdot \begin{pmatrix} 300 & -313 & 267 \\ -399 & 313 & -252 \\ 6 & 0 & -51 \end{pmatrix}$$

SDES

SDES is Simplified Data Encryption Standard developed by a professor at Santa Clara University to understand the concepts of original DES algorithm.

There are five ingredients in any conventional algorithm like

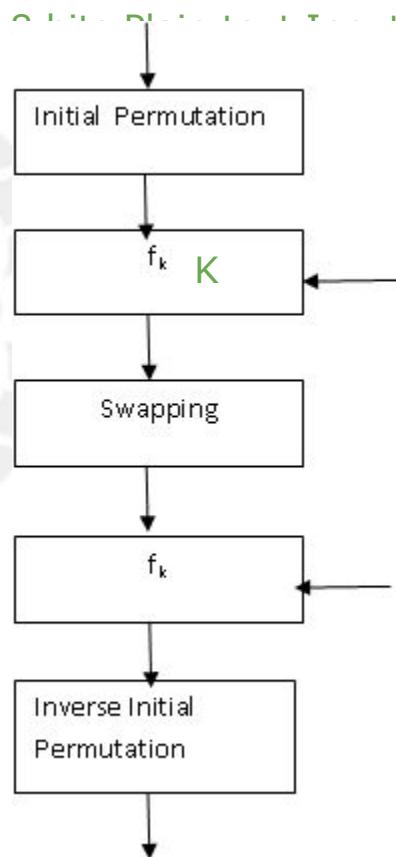
1. Plaintext
2. Secret Key
3. Encryption algorithm
4. Cipher text
5. Decryption algorithm

All operations of encryption and decryption are based on Permutation and Substitution. SDES Encryption algorithm takes 8 bit as an input and generates a 8 bit output. Message or Plain text is divided into two halves as L_0 and R_0 . Where L_0 consists of the first 4 bits and R_0 will have next 4 bits. Key K is of 10 bits length and two keys will be generated from that 10 bit key each of length 8 bits each for each round. Initially initial permutation will be carried out followed by function f_k . Then a swapping is done followed by performing function f_k . Finally inverse initial permutation will be performed to generate 8 bit cipher text.

Decryption is reverse of the encryption where 8 bit cipher text will be taken as input and will generate 8 bit plain text. The keys used in the decryption process will be in reverse order. That is k_2 for first round of decryption and k_1 for second round of decryption.

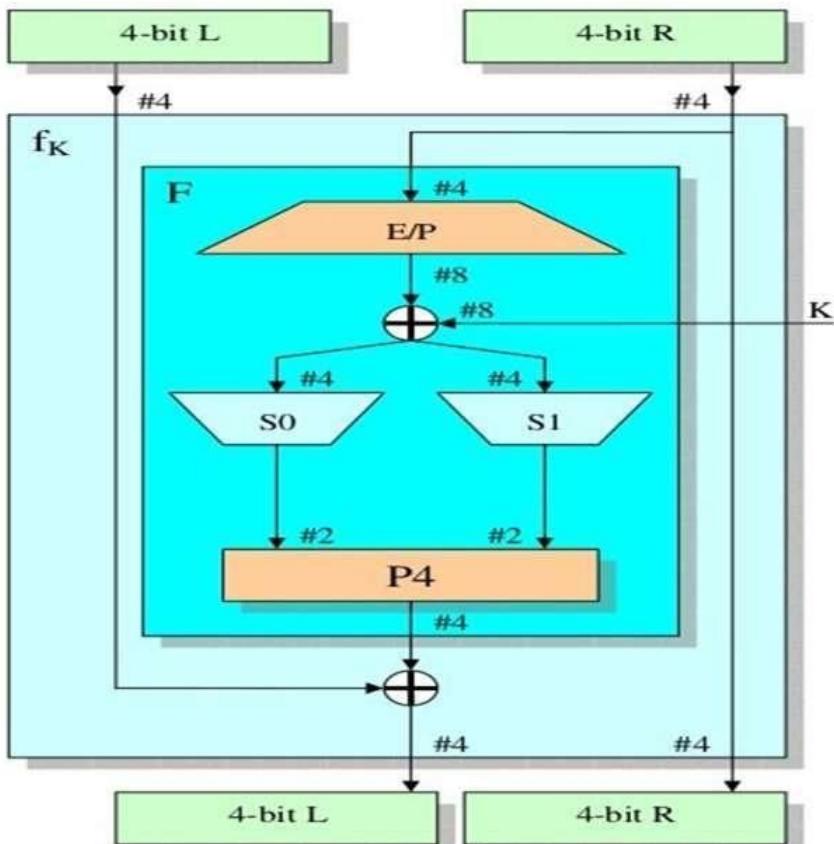
The main part of the encryption process is a function $f(R_{i-1}, K_i)$ that takes a 8-bit input

SDES Encryption



8 bits Cipher text Input

S DES f_k implementation



Courtesy: Internet source

As depicted in the above figure , the 8 bits input is divided into two halves as L and R . Function F is implemented for the right half of the data. Then left part of the data is XORed with the output of function F. It can be written as

$$L_i = R_{i-1} \text{ and } R_i = L_{i-1} \text{ XOR } F(R_{i-1}, K_i)$$

Function F:

Function F comprises of four functionalities like

1. Expansion / Permutation

2. XOR

3. S-Box Substitution

4. Permutation

Consider an example eight bits as 10100011 where right part four bits are 0011

1	2	3	4
0	0	1	1

Expansion / Permutation

In this the number of bits is expanded from 4 bits to 8 bits and the order of bits is rearranged.

E/P

4	1	2	3	2	3	4	1
1	0	0	1	0	1	1	0

Place the four bits in the specified permutation order.

After E/P , the output will be XORed with key which is of 8 bits.

Say the key k as 11010110

Output generated after XOR operation is 01000000

Next S-Box substitution is performed where four bits will be reduced to 2 bits. First two bits will determine the row value in S-Box and last two bits will determine the column value in S-Box. Consider the S-Box as

		0	1	2	3
S0	0	1	0	3	2
	1	3	2	1	0
	2	0	2	1	3
	3	3	1	3	2

		0	1	2	3
S1	0	0	1	2	3
	1	2	0	1	3
	2	3	0	1	0
	3	2	1	0	3

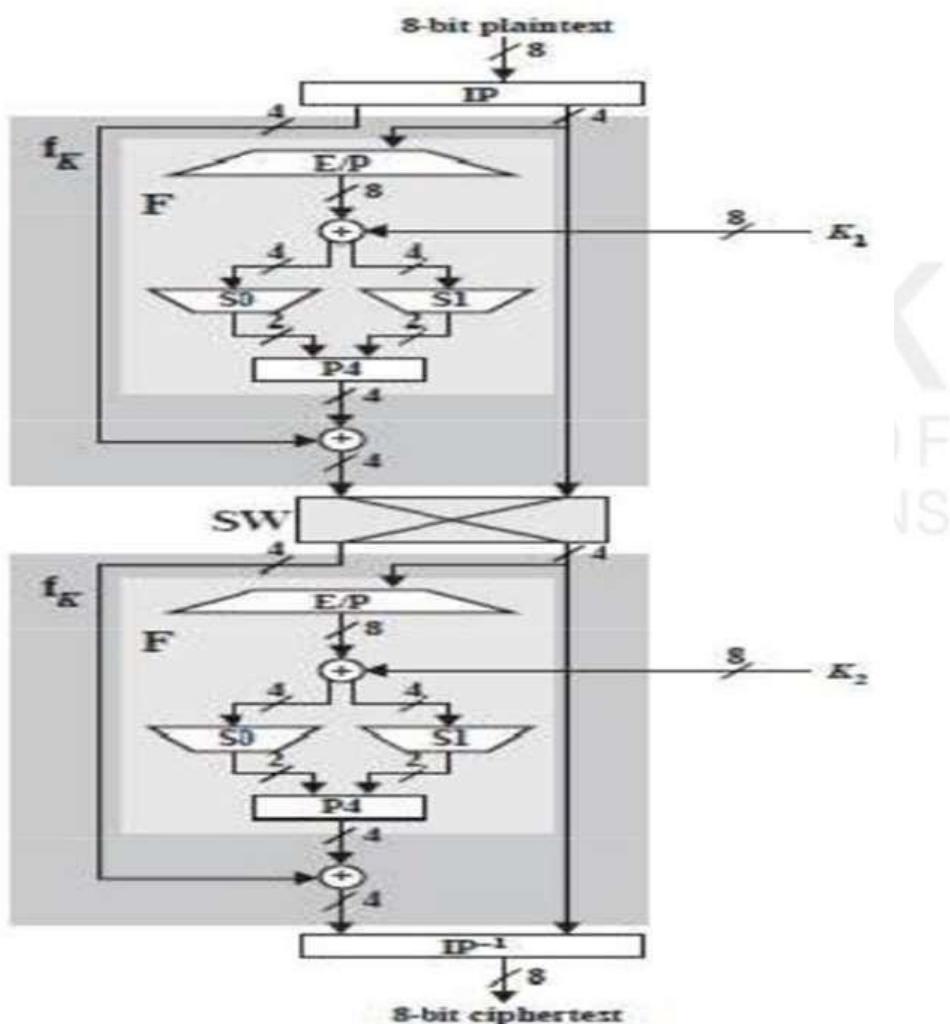
output generated is 01000000. For S0 bits considered are 0100 and for S1 it is 0000

When first two bit 01 is considered which results in row 1 and next two bit 00 is considered it is resulted in column 0. So the value in that row 1 and column 0 is 3 which will be written as 11.

Similarly, S1 value will be 00.

Final output after S-Box substitution is 1100. This will be rearranged using permutation function 2431.

After permutation the output of function F is 1001. In f_k next step is to perform XOR operation with left half of the input. From the input left half bits are 1010 which will be XORED with 1001(output of F) to generate 0011. These four bits will be the left half to the swapping function and right half bits are considered directly to the swapping function as depicted in the below figure. Same process will be repeated and finally inverse permutation will be performed to get the cipher text.



BLOCK CIPHER PRINCIPLES

Most of all symmetric block encryption algorithms are based on the Feistel structure.

Before examining the principles of Feistel structure, let us discuss about stream cipher

and block cipher.

Stream Cipher: Any cipher algorithm which encrypts a single bit or byte data at a time it is termed as stream cipher.

Examples of stream cipher : Auto keyed Vigenere cipher and vernam cipher

Block Cipher: Any algorithm which takes a block of inputs as input to generate cipher text block. A block size is of 64 bits or 128 bits.Examples of block cipher are

DES, AES.

Motivation for the Feistel Cipher Structure

Feistel structure is used in block ciphers. As said earlier a block cipher takes n bit block as input for encryption and generates n bit cipher text block. Each cipher text

block should be unique to make encryption process as reversible process. This transformation is reversible or nonsingular.

Plaintext Ciphertext

Consider an example transformation for 2 bit blocks.

Reversible Mapping

10	00
----	----

11	01
----	----

In the above plain text and cipher text there is an unique cipher text for each plain text blocks.

Irreversible Mapping

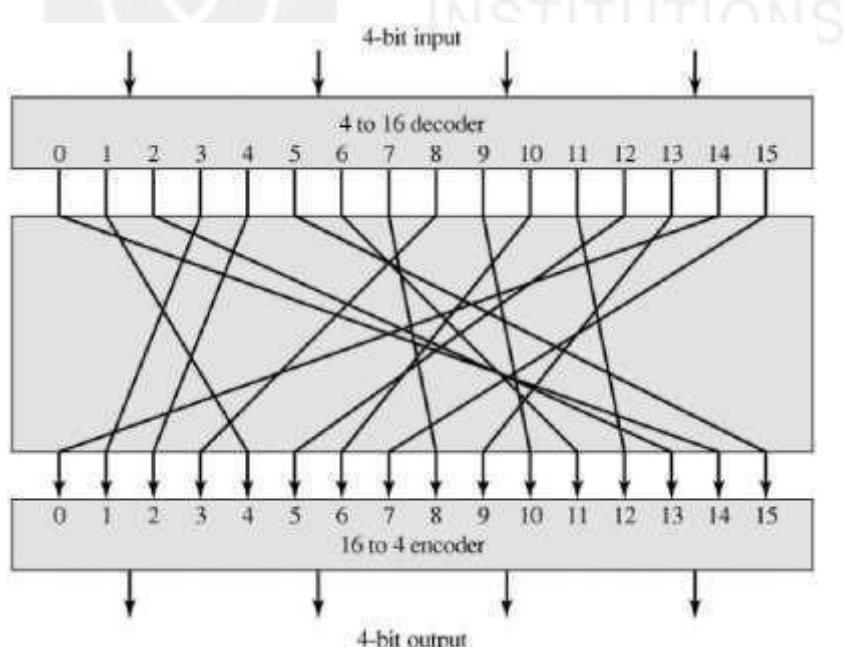
Plaintext Ciphertext

00	11
01	10
10	01
11	01

In this there are two plain text blocks 10 and 11 which are encrypted to the same cipher text block 01. So reverse mapping is alone considered where there are $2n!$ possible transformations.

Ideal Block cipher:

Maximum numbers of possible encryption mappings are possible in an ideal block cipher. If suppose the number of bits are less say 4 bits, then this block cipher is similar to substitution techniques. So it is vulnerable to statistical analysis of the plain text. When the number of bits is more in number and reversible substitution is performed between plain text and cipher text then it is not easy to perform statistical analysis.



General n -bit- n -bit Block Substitution (shown with $n = 4$)

Consider an example transformation for 4 bit blocks.

Plaintext	Ciphertext
0000	1110
0001	0100
0010	1101
etc.,	

For the above transformation , the key is the mapping itself.

The Feistel Cipher

Feistel proposed a concept based on product cipher to approximate the ideal block cipher. Product cipher is that sequentially two or more ciphers will be executed

and the output generated is cryptographically stronger. This approach is to develop a block cipher with a key length of k bits and a block cipher of n bits with the possible transformations as 2^k instead of $2^n!$ as in case of ideal block cipher. A product cipher proposed by Claude Shannon makes use of confusion and diffusion functions. Feistel proposed similar to that where substitution and permutation functions are used alternatively.

Diffusion and Confusion

These two terms defined by Claude Shannon in-order to thwart cryptanalysis based on statistical analysis. For example in a message of certain languages there are some terms which are frequently used. If these terms are reflected in the cipher text there is a possibility to get the whole key or a part of the key used for encryption process. To make a cipher as strong ideal ciphers, cipher text generated should be independent of the key used. For this to achieve Claude Shannon, suggested diffusion and confusion functions.

In diffusion each plain text digit affects many cipher text digits. It is also equivalent that each cipher text digit is affected by many plain text digits. In a block cipher this function can be achieved by performing repeated permutation followed by the implementation of any function on the permuted input. Hence, statistical relationship between plain text and cipher text becomes complex.

In confusion statistical relationship between cipher text and the key becomes more complex to thwart attempts to discover the key used for encryption process. If suppose an attacker tries to get the pattern used in the key based on the cipher text generated also not able to get the original key due to confusion involved in the substitution algorithms.

Feistel Cipher Structure

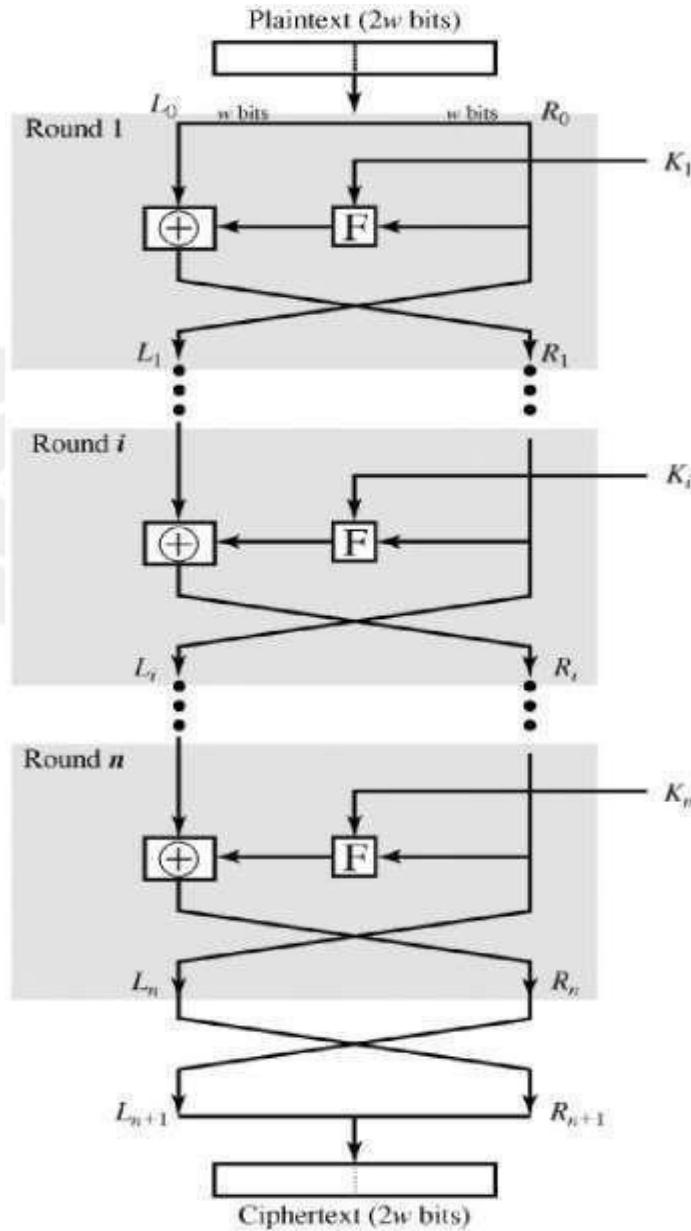
As discussed earlier in the block cipher there are block of inputs provided as an input along with the key. Consider the plain text block is of length $2w$ bits and a key

K. This $2w$ input block is divided into two halves as left part L_0 and right part R_0 . Based on the number of rounds these two halves will be passed for processing.

Finally these two halves will combine to generate the output block.

Each round i has as inputs L_{i-1} and R_{i-1} , extracted from the previous round, a subkey K_i , generated from the overall K . Sub keys generated are different from each other and it is also not same as original key K . This is depicted in the below diagram.

Figure Classical Feistel Network



A round function F is performed on the right half of the input and exclusive-OR operation is performed on the output of the function F and left half of the data. Round function performs the same functionality in each round but the key used in each round varies. This is termed as substitution followed by permutation where the two halves of the data are interchanged. This feistel structure is similar to the substitution-permutation network (SPN) proposed by Shannon.

Working of the feistel structure is based on the following parameters like

1. Block size

Larger block size makes the algorithm more secured. As length of the input to be processed is more the speed of the encryption / decryption process gets reduced.

2. Key size

When key size is more, it is tough for performing brute force attacks.

3. Number of rounds

More number of rounds increases the security of the algorithm. A typical size incorporated is 16 rounds.

4. Subkey generation algorithm

This algorithm should be so complex such that cryptanalysis is difficult to perform.

5. Round function

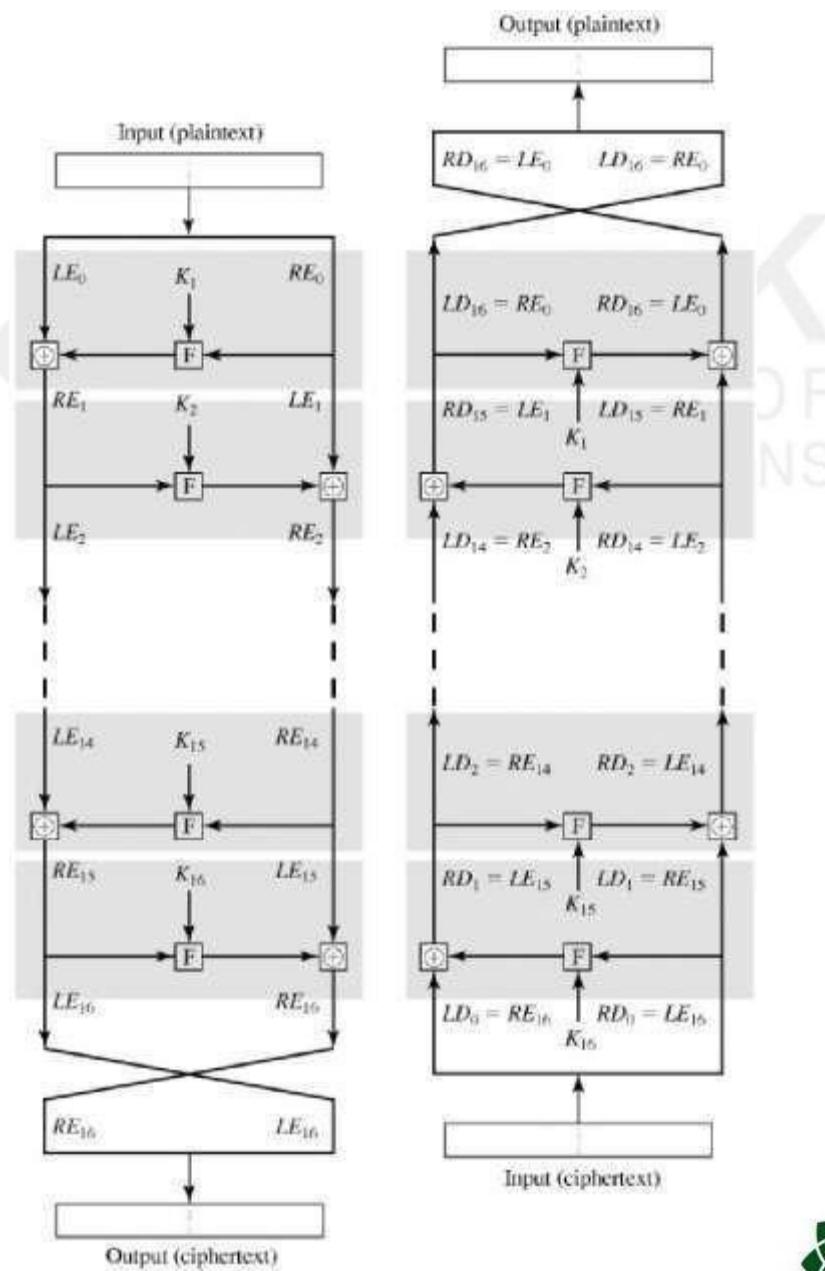
Same as sub key generation algorithm , this function should be too complex to make greater resistance to cryptanalysis.

There are two other considerations in the design of a Feistel cipher: Fast software encryption/decryption
Ease of analysis

Feistel Decryption Algorithm

Similar to the encryption process depicted in the above diagram, the decryption process accepts cipher text as input and the key is provided in reverse order. That is, use K_n in the first round, K_{n-1} in the second round, and so on until K_1

is used in the last round. Same algorithm is used for both encryption and decryption.



Above diagram shows that the encryption process is from top to bottom and the decryption process is shown in the right side of the diagram. To differentiate the data provided for the encryption and decryption it is specified as LE_i and RE_i for data in the encryption algorithm and LD_i and RD_i in the decryption algorithm. Finally after the last round the two halves output are swapped.

For the i th iteration of the encryption

$$\text{algorithm, } LE_i = RE_{i-1}$$

$$RE_i = LE_{i-1} \times F(RE_{i-1}, K_i)$$

Rearranging terms,

$$RE_{i-1} = LE_i$$

$$LE_{i-1} = RE_i \times F(RE_{i-1}, K_i) \quad 2 = RE_i \times F(LE_i, K_i)$$

STRENGTH OF DES

There are security concern about key size and the nature of the algorithm of DES.

The Use of 56-Bit Keys

In DES algorithm 56 bits are used as key size. So there are 2^{56} possible keys approximating to $7.2 * 10^{16}$. So to perform brute force attack it is infeasible.

Brute

force attack is trying all combinations and trying to find the match. In order to break the cipher, a machine capable of performing one encryption per microsecond will take thousand years. As parallel processing comes into existence when 1 million machines are performing one encryption per one micro second parallel then it will take 10 hours to crack the cipher. But the cost incurred is more. It is proven that DES is insecure in July 1998 when Electronic Frontier Foundation has identified the key used in three days using DES cracker. Suppose the text message is compressed before encryption

then performing brute force attack is difficult. To attack there is a necessity to know

The Nature of the DES Algorithm

Other than brute force attack there is also an option to perform attack by analyzing the characteristics of DES algorithm. It is termed as cryptanalysis. Main focus is on the eight substitution tables (S-box) used. These S-boxes are constructed in such a way that cryptanalysis is possible if weakness in s-box is known. No one has so far succeeded in identifying the weakness in the S-box.

Timing Attacks

This type of attack is more prominent for public key algorithms. But it is also relevant for symmetric ciphers. How long the algorithms take to perform an operation provide the relationship between the key and the plain text. This makes the attacker possible to crack the key.

Differential and Linear Cryptanalysis

Cryptanalysis is used to analyse and find the encrypted messages without knowing the key. There are two approaches in cryptanalysis as Differential and Linear Cryptanalysis

Differential Cryptanalysis

Trying to discover the non-random behavior of cipher text and using these hints trying to recover the secret key is termed as Differential Cryptanalysis. It uses collection of techniques to find the differences in the transformation performed. DES algorithm is cracked in less than 2^{55} complexity by using Differential Cryptanalysis. Cryptanalysis attack on Data Encryption Standard is performed by choosing 2^{47} plain texts. Behavior of pairs of plain text blocks along each round is observed to perform cryptanalysis.

Consider x as XOR difference between x and x' . Then

$$\begin{aligned}x_{i+1} &= x_{i+1} \text{ XOR } x'_{i+1} \\&= [x_{i-1} \text{ XOR } f(x_i, K_i)] \text{ XOR } [x'_{i-1} \text{ XOR } f(x'_i, K_i)] \\&= x_{i-1} \text{ XOR } [f(x_i, K_i) \text{ XOR } f(x'_i, K_i)]\end{aligned}$$

If number of such differences is determined then it is possible to identify the sub keys used by comparing the probable difference and the actual difference.

Linear Cryptanalysis

Based on affine or linear approximations in the transformations performed this attack is possible. In this attack, instead of 2^{47} chosen plain texts as in case of differential cryptanalysis only 2^{43} known plaintexts are used to find the key.

Consider n bits plain texts and cipher texts and m bit key which is labeled as $P[1], \dots, P[n]$, $C[1], \dots, C[n]$, and $K[1], \dots, K[m]$ respectively.

Then define

$$M[i, j, \dots, k] = M[i] \text{ XOR } M[j] \text{ XOR } \dots \text{ XOR } M[k]$$

Linear equation is formed as

$$P[1, 2, \dots, a] \text{ XOR } C[1, 2, \dots, b] = K[1, 2, \dots, c]$$

Left hand side result to be computed for a large number of plain text-ciphertext pairs. If the result is zero form more number of plain text and ciphertext pairs then the value of K is considered as zero otherwise it is considered as one. All round values can be combined to get the final key used.

Block Cipher Design Principles

DES Design Criteria:

When DES algorithm is considered there are criteria like design of the S-boxes and P- function.

When S-box is considered there are criteria like

1. S-box to be designed in such a way that output should not have any relevance with the inputs.
2. All 16 possible output combinations are included in each row of S-box.
3. Is there is a single bit variant in the two inputs of S-box, output must differ in at least two bits.
4. If suppose there is a change in first two bits and it is identical in last two bits in the input to S-box, output generated should not be the same.

The criteria for the permutation P are as follows:

Permutation to be done in such a way that four output bits from each S-box affect six different S-boxes on the next round.

The criteria for the permutation P are as follows:

Permutation to be done in such a way that four output bits from each S-box affect six different S-boxes on the next round.

There are three critical aspects of block cipher design like

- a. Number of rounds
- b. Design of the function F
- c. Key scheduling

Number of Rounds

If the numbers of rounds are more then there is a difficulty in the cryptanalysis even if the function F implemented in each round is weak. So, number of rounds should be chosen such that more cryptanalytic efforts are required to perform attack.

Design of Function F

Main functionality implemented in Feistel block cipher is the function F. Main criteria for designing the function F is that it should be nonlinear. If there is any change in a single bit in the input, there are more number of bits change in the output. It is termed as an avalanche effect.

There are more criteria available as

- 1. strict avalanche criterion (SAC)**
- 2. Bit independence criterion (BIC)**

Data Encryption Standard (DES)

It is a symmetric block cipher based on Feistel structure. Plaintext is processed in 64-bit blocks using a 56-bit key (remaining 8 bits are used as parity bits or neglected). The algorithm transforms 64-bit input in a series of steps into a 64-bit output.

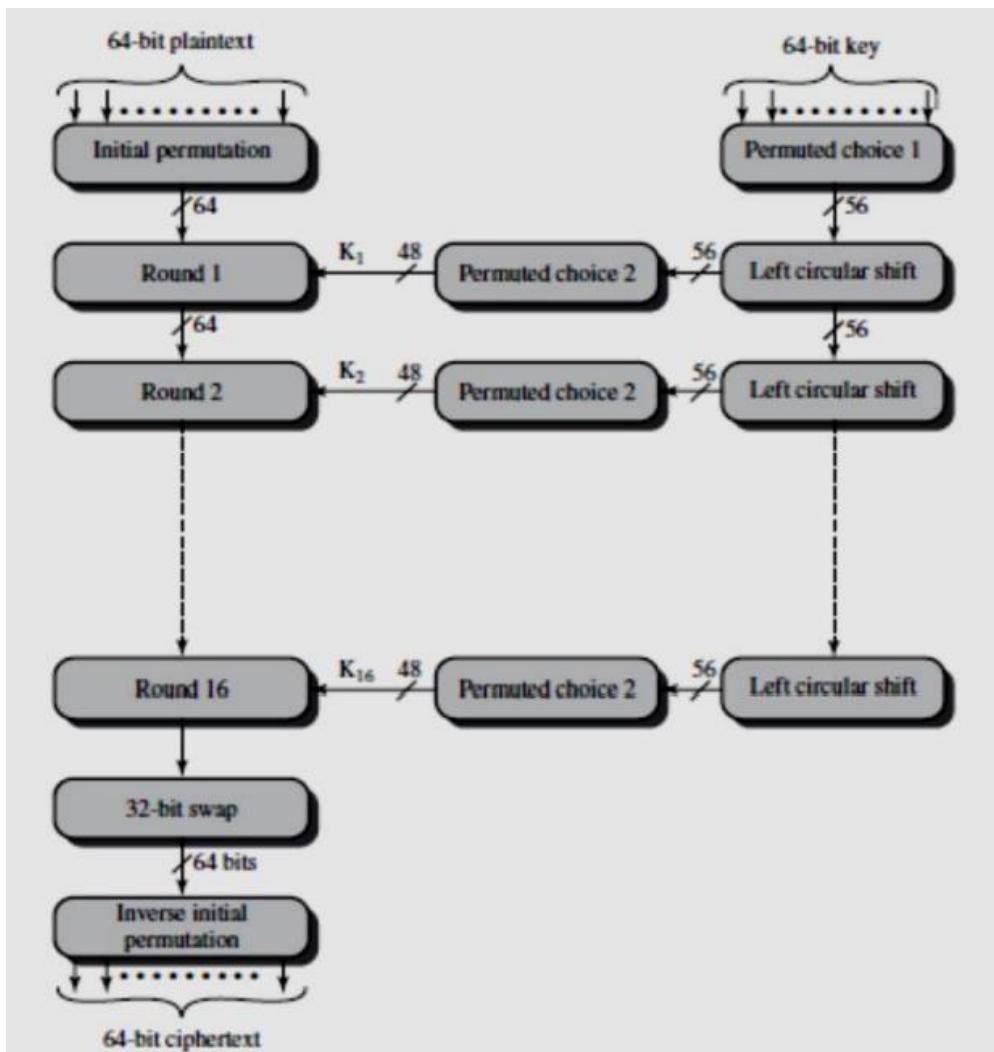


Fig: General Depiction of DES Encryption Algorithm

Looking at the left-hand side of the figure, we can see that the processing of the plaintext proceeds in three phases. First, the 64-bit plaintext passes through an initial permutation (IP) that rearranges the bits to produce the permuted input. This is followed by a phase consisting of sixteen rounds of the same function, which involves both permutation and substitution functions.

The output of the last (sixteenth) round consists of 64 bits that are a function of the input plaintext and the key. The left and right halves of the output are swapped to produce the preoutput. Finally, the preoutput is passed through a permutation [IP-1] that is the inverse of the initial permutation function, to produce the 64-bit ciphertext.

The right-hand portion of the above figure shows the way in which the 56-bit key is used. Initially, the key is passed through a permutation function. Then, for each of the sixteen rounds, a subkey (K_i) is produced by the combination of a left circular shift and a permutation. The permutation function is the same for each round, but a different subkey is produced because of the repeated shifts of the key bits.

Details of Single Round

The below figure shows the internal structure of a single round. The left and right halves of each 64-bit intermediate value are treated as separate 32-bit quantities, labeled L (left) and R (right).

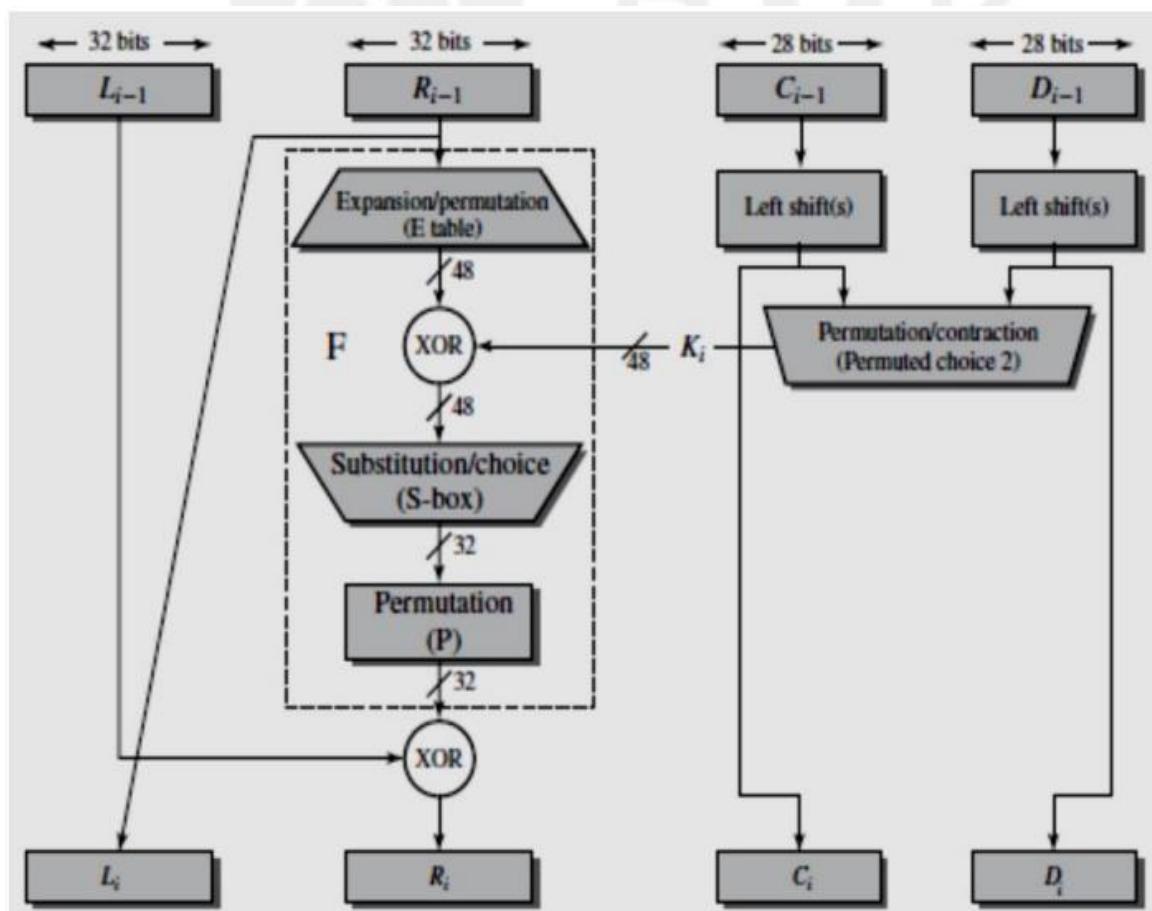


Fig: Single Round of DES Algorithm

As in any classic Feistel cipher, the overall processing at each round can be summarized in the following formulas:

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

The round key K_i is 48 bits. The R input is 32 bits. This R input is first expanded to 48 bits by using a table that defines a permutation plus an expansion that involves duplication of 16 of the R bits. The resulting 48 bits are XORed with K_i . This 48-bit result passes through a substitution function that produces a 32-bit output.

S-Box

The substitution consists of a set of eight S-boxes, each of which accepts 6 bits as input and produces 4 bits as output. The first and last bits of the input to box S_i form a 2-bit binary number to select one of four substitutions defined by the four rows in the table for S_i . The middle four bits select one of the sixteen columns. The decimal value in the cell selected by the row and column is then converted to its 4-bit representation to produce the output. For example, in S_1 , for input 011001, the row is 01 (row 1) and the column is 1100 (column 12). The value in row 1, column 12 is 9, so the output is 1001.

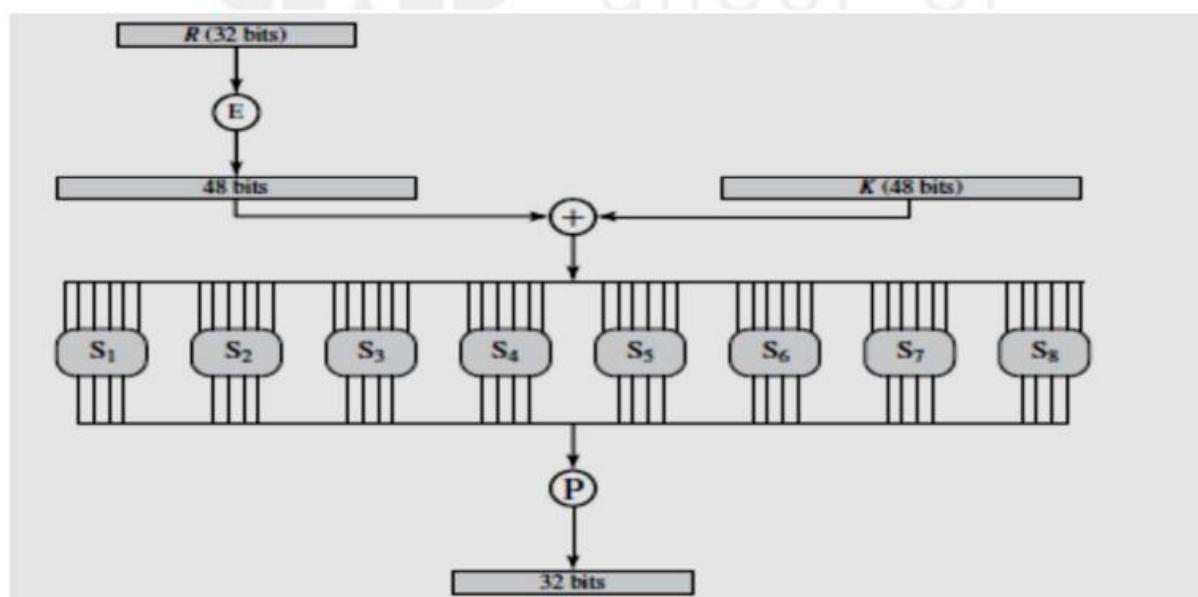


Fig: Calculation of $F(R, K)$

The 32-bit output from the eight S-boxes is then permuted, so that on the next round, the output from each S-box immediately affects as many others as possible.

Key Generation

A 64-bit key is used as input to the algorithm. The bits of the key are numbered from 1 through 64; every eighth bit is ignored. The key is first subjected to a permutation. The resulting 56-bit key is then treated as two 28-bit quantities, labeled C0 and D0. At each round, Ci and Di are separately subjected to a circular left shift or (rotation) of 1 or 2 bits. These shifted values serve as input to the next round. They also serve as input to the contraction permutation, which produces a 48-bit output that serves as input to the function F(Ri-1, Ki).

DES Decryption

As with any Feistel cipher, decryption uses the same algorithm as encryption, except that the application of the subkeys is reversed. Additionally, the initial and final permutations are reversed.

The Avalanche Effect

A desirable property of any encryption algorithm is that a small change in either the plaintext or the key should produce a significant change in the ciphertext. In particular, a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext. This is referred to as the avalanche effect. If the change were small, this might provide a way to reduce the size of the plaintext or key space to be searched.

BLOCK CIPHER MODES OF OPERATION

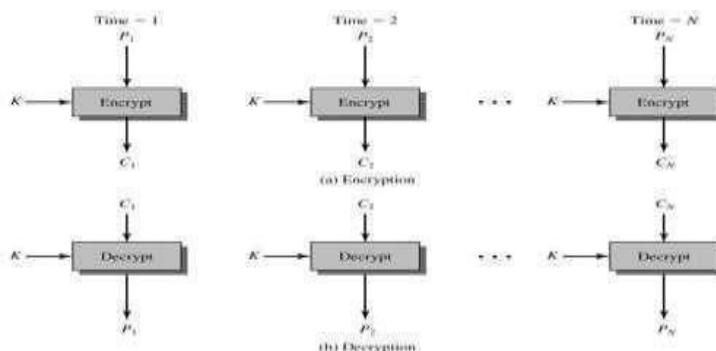
A block cipher considers fixed-length of text block say n bits and key as input and will generate n bit block of cipher text as output. If suppose the length of the input been considered is greater than n bits then the block cipher has to perform the encryption process or decryption process by breaking the input text block into number of blocks with n bits each. Similarly, there is an issue that if the same key been used for multiple blocks there are chances that the key can be compromised. To solve this problem NIST has proposed five modes of operation to apply block cipher in a variety of applications. These five modes can be used in symmetric

ciphers like DES and AES. Five modes of Operation:

1. Electronic codebook (ECB) mode
2. Cipher block chaining (CBC) mode
3. Cipher feedback (CFB) mode
4. Output feedback (OFB) mode
5. Counter (CTR) mode

Electronic Codebook (ECB) Mode

This is the simplest mode of operation. Here in this approach plain text is considered one block at a time and each block of plain text is been converted into cipher text block with a key k as shown in the figure below. As there is unique cipher text been generated for every plain text block it is termed as codebook.



Courtesy : Cryptography and Network Security by William Stallings

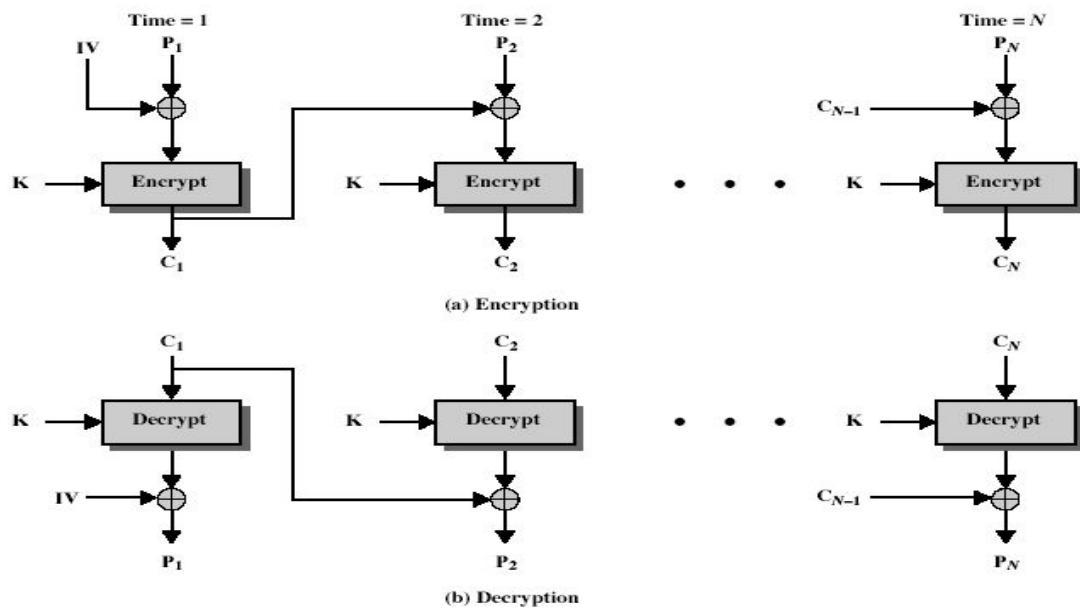
As specified if the plain text block has a greater number of bits then it will be divided into number of blocks each having the length of n bits. Last block may have bits lesser than n bits where padding of bits will be done. Above diagram depicts about encryption and decryption. In encryption plain text P_i of n bits considered and key K of n bits considered and it is converted to cipher text C_i of n bits based on the algorithm used either DES or AES.

Limitations:

If the plain text blocks are repeated in the input the cipher text blocks generated will also be same. For lengthy messages this mode of operation is not appropriate due to its weakness. So mostly this mode can be used if the data is less, for example encryption key.

Cipher block chaining (CBC) mode

To overcome the limitations specified earlier in ECB mode, if the same plain text block appears more than once in the input also it should generate different cipher text blocks.



This is achieved by using Cipher block chaining mode of operation. Here previous block cipher text output is XORed with the current plain text blocks. So, if the same plain text blocks are repeated also will generate new cipher text blocks. The figure depicts that it is like chaining process, repeating pattern bits are not exposed for the cryptanalyst to crack the cipher text. Now initially for the first block processing there is an Initialization vector (IV) been added and to the last block if the length is lesser, as in case of ECB padding bits will be added to match the length of plain text to the encryption algorithm. Decryption is just reverse of the encryption where cipher text blocks will be considered by the decryption algorithm and the output generated will be XORed with the preceding cipher text blocks to produce the plain text blocks as output.

It can be represented as

Encryption:

$$C_i = E(k [C_{i-1} \text{ XOR } P_i]) \quad \dots \quad (1)$$

Decryption:

$$P_i = C_{i-1} \text{ XOR } D(k, C_i)$$

From Encryption equation decryption equation can be derived as follows

That is D function applied to cipher text along with key K as $D(K, C_i)$

$$D(K, C_i) = D(K, E(k [C_{i-1} \text{ XOR } P_i])) \text{ -- substituting equation 1}$$

$$D(K, C_i) = C_{i-1} \text{ XOR } P_i \text{ (Encryption and Decryption gets cancelled)}$$

$$C_{i-1} \text{ XOR } D(K, C_i) = C_{i-1} \text{ XOR } C_{i-1} \text{ XOR } P_i \text{ (Applying } C_{i-1} \text{ XOR on both sides)} C_{i-1}$$

$$_1 \text{ XOR } D(K, C_i) = P_i \text{ (XOR operation it gets cancelled)}$$

Initial block operation can be specified as

$C_i = E(k [IV \text{ XOR } P_i])$ as depicted in the diagram for encryption and

$P_i = IV \text{ XOR } D(k, C_i)$ as depicted in the diagram for decryption

Thus, Decryption equation is derived from the encryption formula by applying decryption function.

This initialization vector value has to be shared between the sender and the receiver but it is assumed it is not predicted by an adversary or an attacker. So IV can also be

shared by encrypting using ECB mode of operation.

Limitation:

If there is any change in the first block output it will be propagated to all the remaining blocks. Consider a scenario that $n+1$ bits of plain text are converted to cipher text using

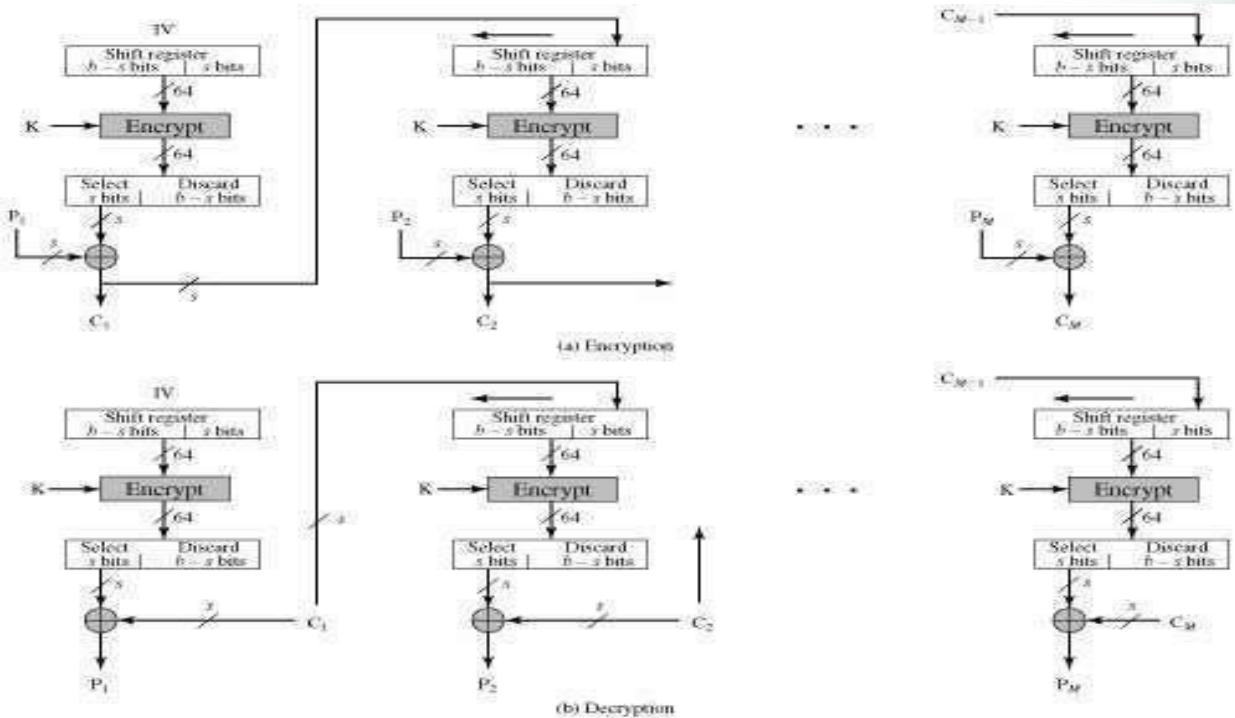
n bit encryption algorithm. As the number of bits are greater than n bits it will be divided into two blocks as n -bit block and one-bit block. As per the concepts of the two modes of operation if the number of bits is lesser than n then padding will be done.

Here

$n-1$ bits will be padded to make the number of bits as n -bits in the second block. So, for one bit to be converted to cipher text, $n-1$ values are added which is not necessary. This problem arises due to the concept of blockcipher. There are options

~~Output feedback (OFB) mode~~ converted into stream cipher so for processing stream ciphers there is a necessity of different modes of operation. Here comes these

modes of operation. In cipher feedback (CFB) mode, output feedback (OFB) mode and Counter (CTR) mode



The above diagram depicts the CFB mode of operation. Here the unit of transmission is in bits. Most commonly the bits considered is of length 8. Similar to CBC mode, chaining process is considered. In an encryption process, previous cipher text is provided to n bits shift register. Based on the encryption algorithm number of bits from MSB bits are considered. In the diagram above, 64-bit encryption algorithm is been used. So, 64 MSB bits are taken from the shift register and been provided as input to the algorithm along with key. After encryption process output generated is also represented as 64 bits shift register. Based on the plain text input size, the MSB bits from the shift register considered, which is XORed with plain text to generate the cipher text output. When the next plain text unit to be encrypted the shift register will be shifted left side by s bits, if the number of plain text bits considered is s bits. After shifted the previous cipher text units generated is fed to the LSB of shift register.

This process continues until there are no plain text units to be encrypted. When it is depicted in the diagram for the encryption algorithm input is previous cipher text units along with key K. So in decryption process also the same encryption algorithm been used. In the final stage instead of plain text units to be XORed with the output of encryption algorithm, cipher text units will be XORed to generate plain text units.

It can be represented as

Encryption:

$$C_i = P_i \text{ XOR MSB}(O_i) \quad \dots \quad (1)$$

Decryption:

$$P_i = C_i \text{ XOR MSB}(O_i) \quad \dots \quad (2)$$

Here O_i represents the output generated from encryption algorithm.

$O_i = E(k, I_i)$ where I_i is the input considered by the encryption algorithm.

$$I_i = \text{LSB}(I_{i-1} || C_{i-1})$$

Limitation:

Here the number of bits XORed with plain text also have dependency with the plain text bits. As in case of CFB if there is small change (bit error) in the input, it will be propagated until the last blocks.

Output feedback (OFB) mode

To avoid error propagation instead of considering cipher text units as feed back units, output generated in the encryption algorithm O_i is fed back to the shift register. Similarly, IV is nonce in OFB and it is unique to each execution of the encryption process.

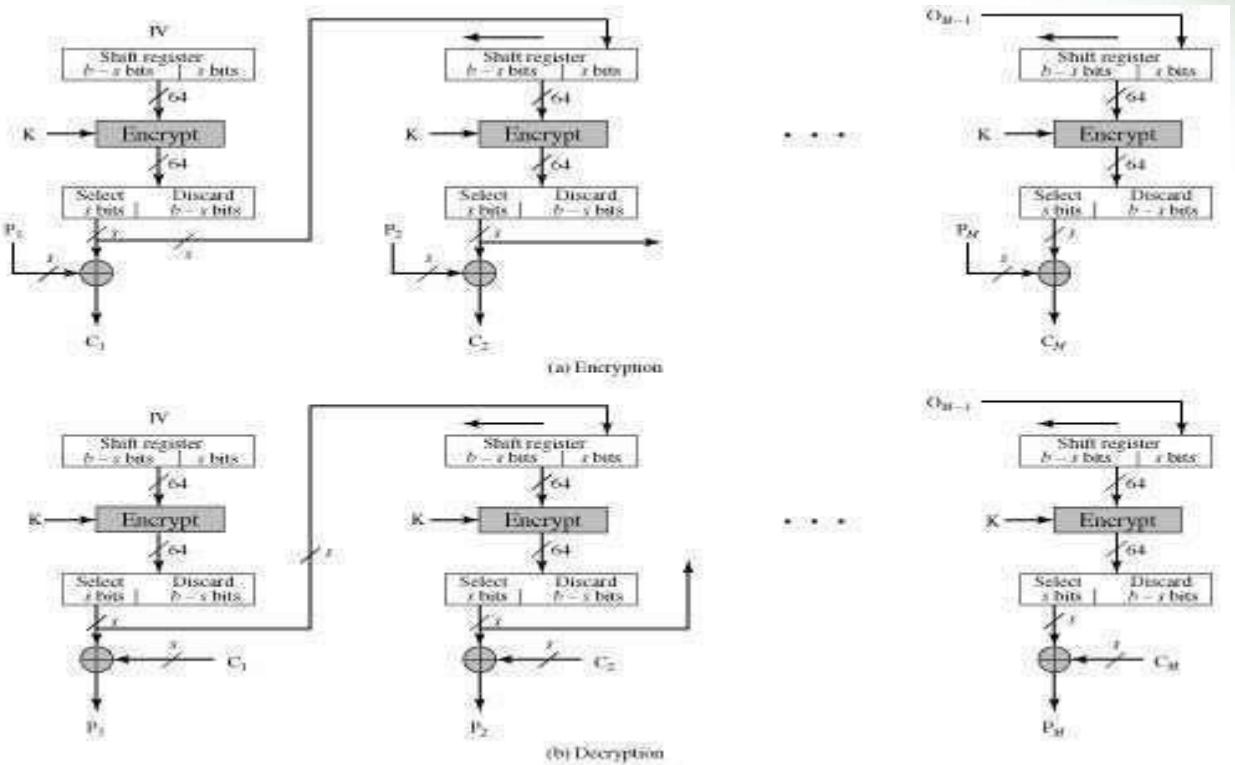
It can be represented

as Encryption:

$$C_i = P_i \text{ XOR } O_i \quad \dots \quad (1)$$

Decryption:

$$P_i = C_i \text{ XOR } O_i \quad \dots \quad (2)$$



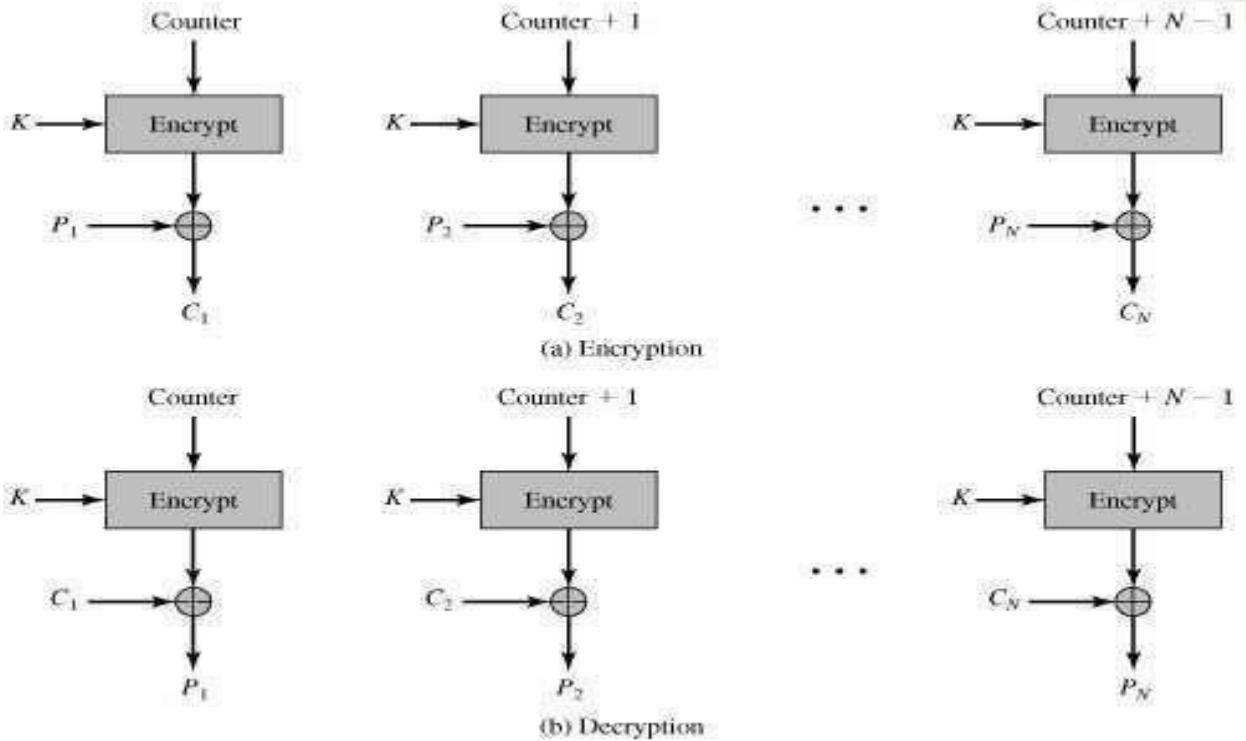
One advantage of the OFB method is that bit errors in transmission do not propagate.

Limitation:

It is vulnerable to message modification attack.

Counter (CTR) mode

In this mode of operation, counter value is used equivalent to the plain text block size. Main requirement is that the counter value should be unique for each block. To make this uniqueness the counter value will be incremented for subsequent blocks. Here there is no chaining process as in case of OFB or CFB. Encryption is done by performing XOR operation with the encrypted counter value and the plain text. In decryption, same order of counter value is used. Encrypted counter value is XORed with cipher text to generate the plain text. Here is the number of bits in the last block is lesser, only those number of bits are considered, remaining bits are discarded. As in case of OFB initial counter value is nonce value considered.



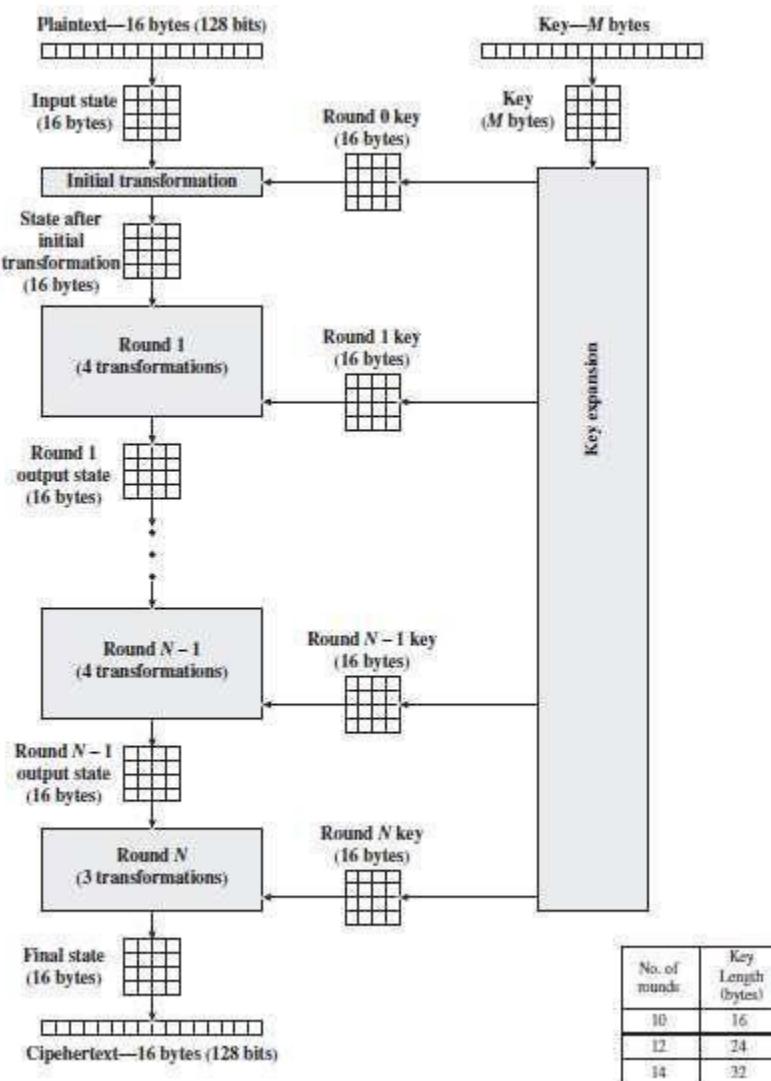
Advantages:

1. Encryption can be done parallel for multiple blocks of plaintext or ciphertext. As there is no chaining process involved.
2. Software Efficiency is achieved due to parallelism in process like aggressive pipelining, multiple instruction dispatch per clock cycle.
3. Preprocessing is achieved because the encryption algorithm does not depend on the plain text.
4. Random access is possible as any block of inputs can be processed. But in previous modes of operation as previous block output to be provided as input this random access is not possible.
5. Security is achieved
6. Simple to use

Advanced Encryption Standard

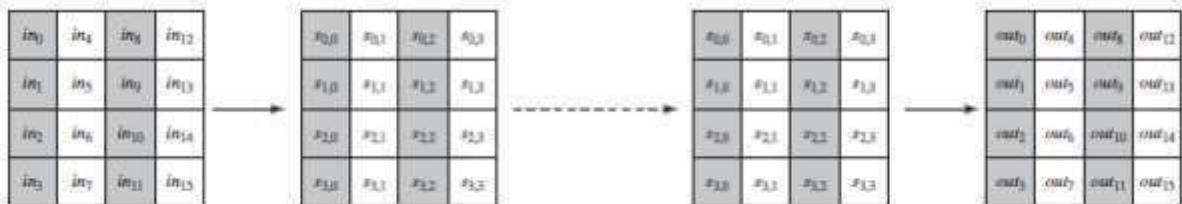
The Advanced Encryption Standard (AES) was published by the National Institute of Standards and Technology (NIST) in 2001. AES is a symmetric block cipher that is intended to replace DES as the approved standard for a wide range of applications.

The cipher takes a plaintext block size of 128 bits, or 16 bytes. The key length can be 16, 24, or 32 bytes (128, 192, or 256 bits). The algorithm is referred to as AES-128, AES-192, or AES-256, depending on the key length.



AES Encryption Process

The input to the encryption and decryption algorithms is a single 128-bit block. This block is depicted as a 4×4 square matrix of bytes. This block is copied into the State array, which is modified at each stage of encryption or decryption. After the final stage, State is copied to an output matrix. These operations are depicted in the below Figure.



Input, State Array and Output of AES

Similarly, the key is depicted as a square matrix of bytes. This key is then expanded into an array of key schedule words. Below shows the expansion for the 128-bit key. Each word is four bytes, and the total key schedule is 44 words for the 128-bit key.

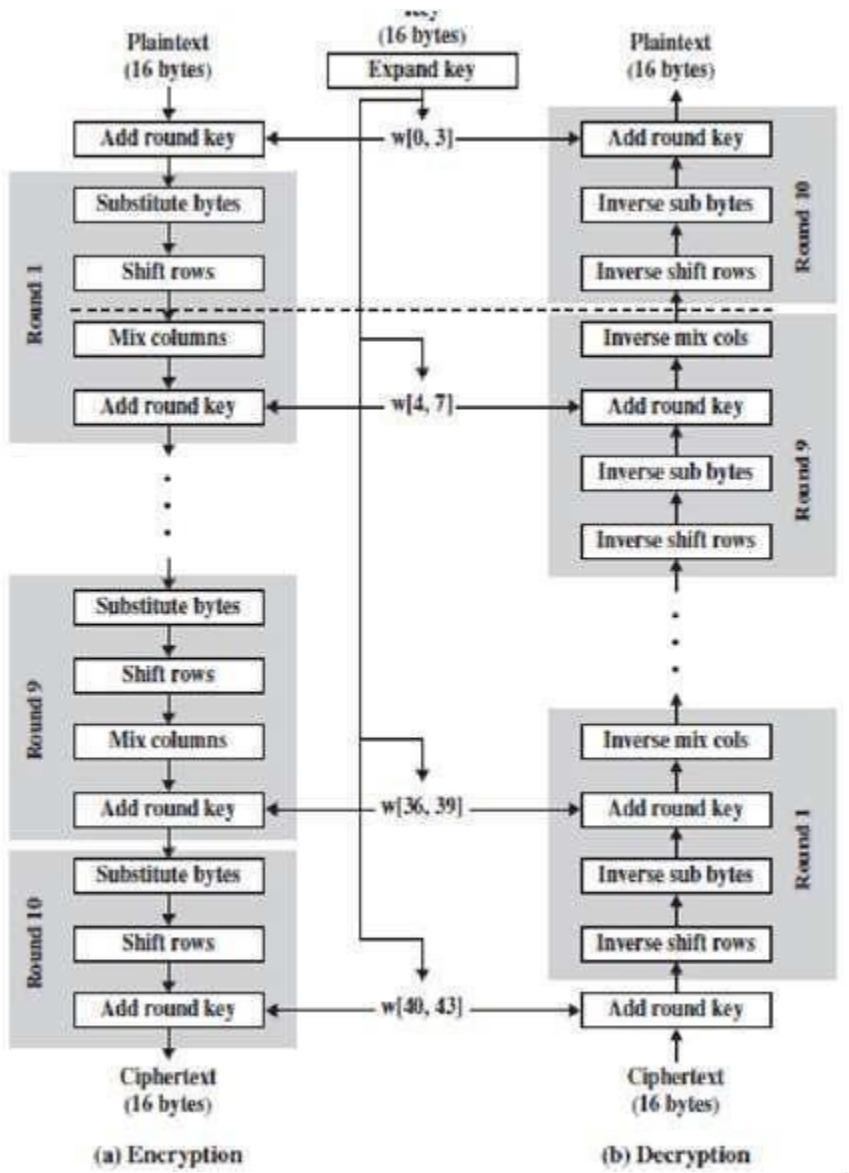


The AES cipher in more detail, indicating the sequence of transformations in each round and showing the corresponding decryption function. Some comments about the overall AES structure.

- AES processes the entire data block as a single matrix during each round using substitutions and permutation.
- The key that is provided as input is expanded into an array of forty-four 32-bit words, $w[i]$. Four distinct words (128 bits) serve as a round key for each round.

- Four different stages are used, one of permutation and three of substitution:
 - Substitute bytes:** Uses an S-box to perform a byte-by-byte substitution of the block
 - ShiftRows:** A simple permutation
 - MixColumns:** A substitution that makes use of arithmetic over
 - AddRoundKey:** A simple bitwise XOR of the current block with a portion of the expanded key

AES ENCRYPTION AND DECRYPTION



- The structure is quite simple. For both encryption and decryption, the cipher begins with an Add Round Key stage, followed by nine rounds that each includes all four stages, followed by a tenth round of three stages.
- AES processes the entire data block as a single matrix during each round using substitutions and permutation.

Substitute bytes

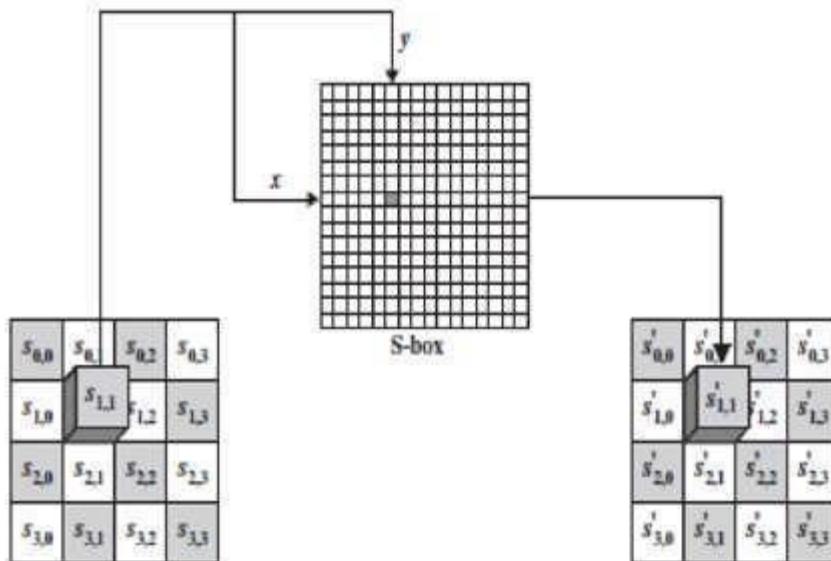
The forward substitute byte transformation, called SubBytes, is a simple table lookup. AES defines a 16×16 matrix of byte values, called an S-box, that contains a permutation of all possible 256 8-bit values.

Each individual byte of State is mapped into a new byte in the following way:

The leftmost 4 bits of the byte are used as a row value and the rightmost 4 bits are used as a column value. These row and column values serve as indexes into the S-box to select a unique 8-bit output value.

For example, the hexadecimal value {95} references row 9, column 5 of the S-box, which contains the value {2A}. Accordingly, the value {95} is mapped into the value

{2A}.



The S-box is constructed in the following fashion:

1. Initialize the S-box with the byte values in ascending sequence row by row. The first row contains {00}, {01}, {02}, ..., {0F}; the second row contains {10}, {11}, {12}, ..., {1F}, etc.; and so on. Thus, the value of the byte at row x, column y is {xy}.
2. Map each byte in the S-box to its multiplicative inverse in the finite field $GF(2^8)$; the value {00} is mapped to itself.
3. Consider that each byte in the S-box consists of 8 bits labeled(b7, b6, b5, b4, b3, b2, b1, b0). Apply the following transformation to each bit of each byte in the S-box:

, where c_i is the ith bit of byte c with the value {63}; that is, $(c_7' c_6' c_5' c_4' c_3' c_2' c_1' c_0') = (01100011)$. The prime (') indicates that the variable is to be updated by the value on the right.

The AES standard depicts this transformation in matrix form as follows:

In the above equation, each element in the product matrix is the bitwise XOR of products of elements of one row and one column. Furthermore, the final addition shown in equation is a bitwise XOR. As an example, consider the input value {95}. The multiplicative inverse in $GF(2^8)$ is ${95}' = {8A}$, which is 10001010 in binary. Using Equation above the result is {2A}, which should appear in row {09} column {05} of the S-box.

$$\begin{bmatrix} p_1^1 \\ p_1^2 \\ p_1^3 \\ p_1^4 \\ p_1^5 \\ p_1^6 \\ p_1^7 \\ p_1^0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} p_1^1 \\ p_1^2 \\ p_1^3 \\ p_1^4 \\ p_1^5 \\ p_1^6 \\ p_1^7 \\ p_1^0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

The inverse substitute byte transformation, called InvSubBytes, makes use of the inverse S-box. It could be noted that the input {2A} produces the output {95}, and the input {95} to the S-box produces {2A}. The inverse S-box is constructed by applying the inverse of the transformation in equation stated above, followed by taking the multiplicative inverse in GF(28). The inverse transformation is: where byted={05}, or 00000101. This transformation could be depicted as follows

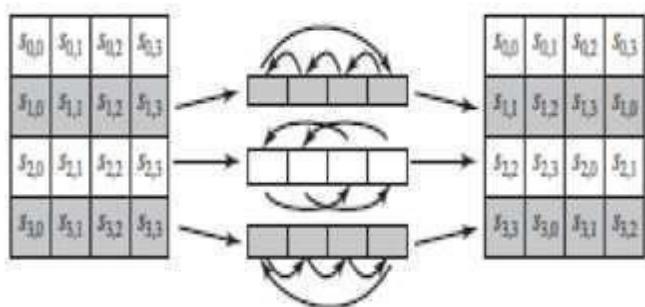
$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

	S															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F3	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	52	D4	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	1D	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	E6	5F	97	44	17	C4	A7	7E	3D	64	SD	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	ED	32	3A	0A	49	06	24	5C	C2	D5	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	8D	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	08	11	69	D9	3E	94	9B	11	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

	S															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	09	6A	D5	30	36	A5	38	BF	40	A2	9E	81	F3	D7	FB
1	7C	E3	79	82	9H	2F	19	87	34	87	43	41	C4	DF	E9	C9
2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	6B	42	FA	C3	4E
3	08	2B	A1	66	28	D9	24	H2	76	58	A2	49	6D	8B	D1	25
4	72	F8	F9	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
5	6C	70	4B	S0	FD	1D	BB	1A	5E	15	46	37	A7	8D	9D	84
6	90	D8	A8	00	8C	HC	D3	0A	F7	14	58	05	HB	43	45	06
7	D0	2C	1E	8F	CA	3F	08	02	C1	AF	BD	03	01	13	8A	60
8	3A	91	13	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	16	73
9	96	AC	74	22	E7	AD	35	85	E2	F9	37	18	1C	75	DF	6E
A	47	F1	1A	71	1D	29	C5	89	6F	17	62	01	AA	18	BB	1B
B	FC	56	3E	4B	C6	D2	79	20	9A	1B	C0	FE	76	C3	5A	F4
C	1F	D0	A8	33	88	07	C7	31	B1	12	16	59	27	80	EC	50
D	60	31	7F	A9	19	H5	4A	03	2D	E5	7A	9F	03	C9	9C	EF
E	A0	E0	3B	4D	A6	2A	P5	10	CB	EB	HB	3C	83	53	99	61
F	17	2B	04	70	BA	77	D6	26	E1	60	14	63	55	21	9C	7D

ShiftRows

The forward shift row transformation, called ShiftRows. The first row of State is not altered. For the second row, a 1-byte circular left shift is performed. For the third row, a 2-byte circular left shift is performed. For the fourth row, a 3-byte circular leftshift is performed. The inverse shift row transformation, called InvShiftRows, performs the circular shifts in the opposite direction for each of the last three rows, with a 1-byte circular right shift for the second row, and so on.



Mix column

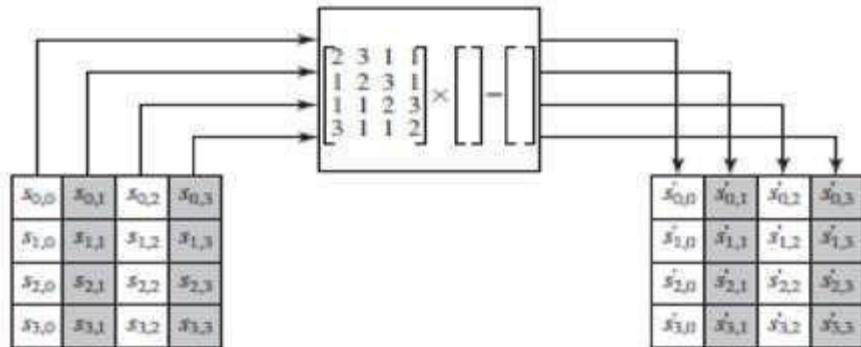
The forward mix column transformation, called Mix Columns, operates on each column individually. Each byte of a column is mapped into a new value that is a

function of all four bytes in that column.

$$\text{The transformation on State} \quad \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

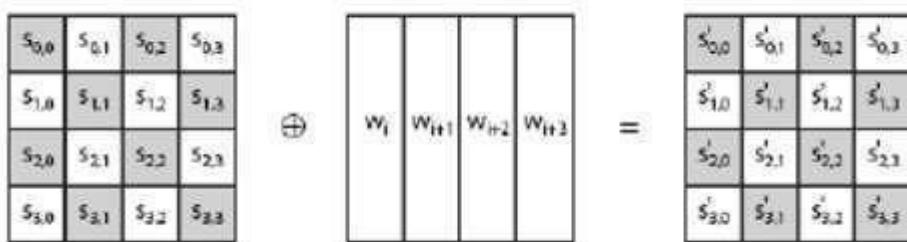
Each element in the product matrix is the sum of products of elements of one row and one column. In this case, the individual additions and multiplications are performed in GF(28). The Mix Columns transformation on a single column $j (0 \leq j \leq 3)$ of State can be expressed as:

$$\begin{aligned}
 s'_{0,j} &= (2 \bullet s_{0,j}) \oplus (3 \bullet s_{1,j}) \oplus s_{2,j} \oplus s_{3,j} \\
 s'_{1,j} &= s_{0,j} \oplus (2 \bullet s_{1,j}) \oplus (3 \bullet s_{2,j}) \oplus s_{3,j} \\
 s'_{2,j} &= s_{0,j} \oplus s_{1,j} \oplus (2 \bullet s_{2,j}) \oplus (3 \bullet s_{3,j}) \\
 s'_{3,j} &= (3 \bullet s_{0,j}) \oplus s_{1,j} \oplus s_{2,j} \oplus (2 \bullet s_{3,j})
 \end{aligned}$$



Add round key

In the forward add round key transformation, called Add Round Key, the 128 bits of State are bitwise XORed with the 128 bits of the round key. The operation is viewed as a column wise operation between the 4 bytes of a State column and one word of the round key; it can also be viewed as a byte-level operation. The inverse add round key transformation is identical to the forward add round key transformation, because the XOR operation is its own inverse.



AES Key Expansion

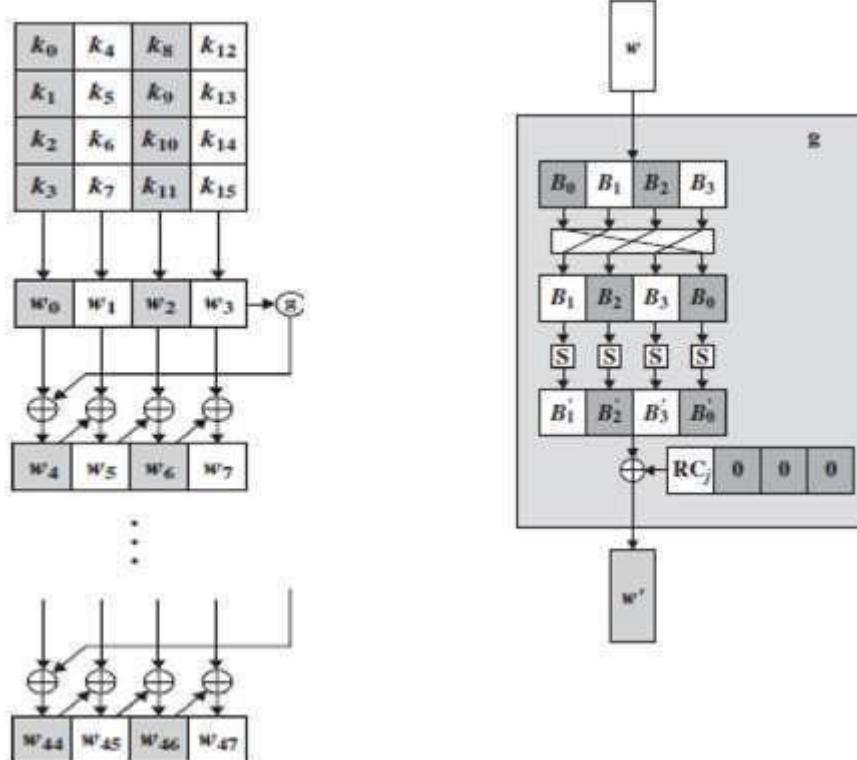
The AES key expansion algorithm takes as input a four-word (16-byte) key and produces a linear array of 44 words (176 bytes).

This is sufficient to provide a four-word round key for the initial Add Round Key stage and each of the 10 rounds of the cipher. The key is copied into the first four words of the expanded key. The remainder of the expanded key is filled in four words at a time. Each added word $w[i]$ depends on the immediately preceding word, $w[i-1]$, and the word four positions back, $w[i-4]$. In three out of four cases, a simple XOR is used. For a word whose position in the w array is a multiple of 4, a more complex function is used.

The function g consists of the following subfunctions:

- RotWord performs a one-byte circular left shift on a word. This means that an input word $[B_0, B_1, B_2, B_3]$ is transformed into $[B_1, B_2, B_3, B_0]$.
- SubWord performs a byte substitution on each byte of its input word, using the S-box.

- The result of steps 1 and 2 is XORed with a round constant RC_j



The round constant is a word in which the three rightmost bytes are always 0. Thus, the effect of an XOR of a word with Rcon is to only perform an XOR on the leftmost byte of the word. The round constant is different for each round and is defined as $Rcon[j] = (RC[j], 0, 0, 0)$, with $RC[1] = 1$, $RC[j] = 2 * RC[j-1]$ and with multiplication defined over the field $GF(2^8)$. The values of $RC[j]$ in hexadecimal are

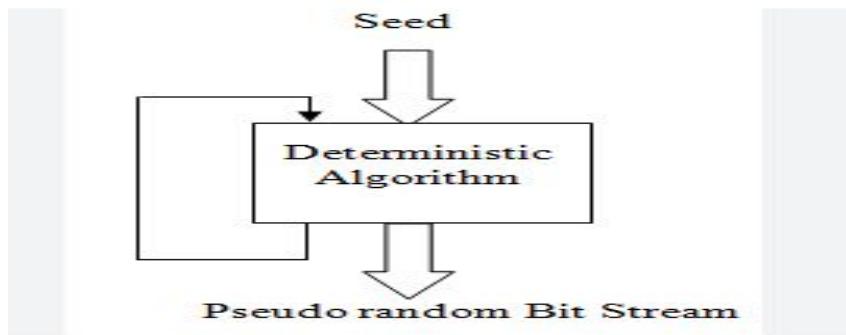
j	1	2	3	4	5	6	7	8	9	10
RC [j]	01	02	04	08	10	20	40	80	18	36



Pseudorandom Number Generators

1. Introduction

A Pseudorandom Number Generator (PRNG) is an algorithm that uses mathematical formulas or precomputed tables to produce sequences of numbers that appear random. Unlike true random number generators (TRNGs), which rely on physical processes, PRNGs are deterministic and generate numbers based on an initial value called a seed.



2. Characteristics of PRNGs

- Deterministic: Given the same seed, a PRNG will always produce the same output sequence.
- Efficient: Capable of generating numbers quickly.
- Periodic: Eventually, the sequence repeats after a fixed number of outputs.
- Predictable: If the seed and algorithm are known, the sequence can be reconstructed.

3. Basic Structure

- A PRNG typically includes:
- A seed value (initial input)
- A recurrence relation or transformation function
- An output function to produce numbers in a specific range

Mathematically:

$$X_0 = \text{seed}$$

$$X_1 = f(X_0), X_2 = f(X_1), \dots$$

Output: X_1, X_2, X_3, \dots

4. Common PRNG Algorithms

4.1 Linear Congruential Generator (LCG)

One of the simplest and oldest PRNGs.

Formula:

$$X_{i+1} = (aX_i + c) \bmod m$$

Where:

a = multiplier

c = increment

m = modulus

X_0 = seed

Example:

$$X_{i+1} = (5X_i + 3) \bmod 16$$

4.2 Mersenne Twister

- Developed by Matsumoto and Nishimura (1997)
- Period: $2^{19937} - 1$
- Highly popular in scientific computing
- Provides high-quality pseudorandom numbers

4.3 XORShift Generator

- Uses bitwise XOR and shift operations
- Very fast and suitable for applications needing lightweight randomness

4.4 Cryptographically Secure PRNG (CSPRNG)

Designed to be secure against reverse engineering and prediction.

Examples:

- Fortuna
- Yarrow
- AES-CTR-based PRNG
- /dev/urandom in Unix-like systems

5. Applications of PRNGs

- Simulations (e.g., Monte Carlo methods)
- Games (e.g., random enemy spawns)
- Cryptography (e.g., key generation, nonce generation)
- Statistical sampling
- Procedural content generation

6. PRNG vs. TRNG

Feature	PRNG	TRNG
Nature	Algorithm-based (deterministic)	Hardware-based (non-deterministic)
Speed	Fast	Slower
Repeatability	Yes, with same seed	No
Applications	Simulations, games, crypto	Secure key generation, lotteries
Example	Mersenne Twister, LCG	Radioactive decay, thermal noise

7. Cryptographic Requirements for PRNGs (CSPRNGs)

A secure PRNG used in cryptography must satisfy:

- Forward security: Past outputs should not be guessed even if current state is known.
- Backward security: Future outputs should not be predictable even if past outputs are known.
- Unpredictability: Without knowledge of the seed, outputs must appear random.

8. Seeding

- Seeding is the process of initializing the PRNG.
- The quality of randomness depends heavily on the seed.
- In secure systems, seeds should come from TRNGs or high-entropy sources.

9. Limitations

- Predictable if the seed is known
- Limited period (eventually repeats)
- Not suitable for cryptographic purposes unless designed as CSPRNG

RC4

In cryptography, **RC4** (Rivest Cipher 4, also known as **ARC4** or **ARCFOUR**, meaning Alleged RC4, see below) is a stream cipher. While it is remarkable for its simplicity and speed in software, multiple vulnerabilities have been discovered in RC4, rendering it insecure. It is especially vulnerable when the beginning of the output keystream is not discarded, or when nonrandom or related keys are used. Particularly problematic uses of RC4 have led to very insecure protocols such as WEP.

RC4 generates a pseudorandom stream of bits (a keystream). As with any stream cipher, these can be used for encryption by combining it with the plaintext using bitwise exclusive or; decryption is performed the same way (since exclusive or with given data is an involution). This is similar to the one-time pad, except that generated *pseudorandom bits*, rather than a prepared stream, are used.

To generate the keystream, the cipher makes use of a secret internal state which consists of two parts:

1. A permutation of all 256 possible bytes (denoted "S" below).
2. Two 8-bit index-pointers (denoted "i" and "j").

The permutation is initialized with a variable-length key, typically between 40 and 2048 bits, using the *key-scheduling algorithm* (KSA). Once this has been completed, the stream of bits is generated using the *pseudo-random generation algorithm* (PRGA).

Key-scheduling algorithm (KSA)

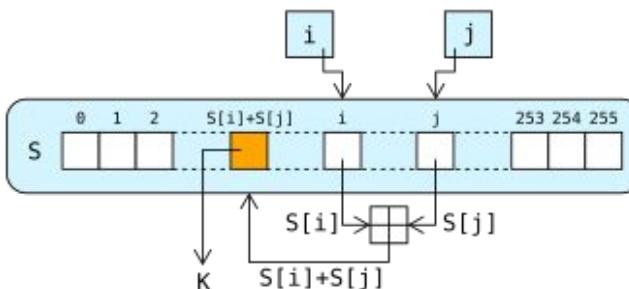
The key-scheduling algorithm is used to initialize the permutation in the array "S". "keylength" is defined as the number of bytes in the key and can be in the range $1 \leq \text{keylength} \leq 256$, typically between 5 and 16, corresponding to a key length of 40–128 bits. First, the array "S" is initialized to the identity permutation. S is then processed for 256 iterations in a similar way to the main PRGA, but also mixes in bytes of the key at the same time.

```

for i from 0 to 255
    S[i] := i
endfor
j := 0
for i from 0 to 255
    j := (j + S[i] + key[i mod keylength]) mod 256
    swap values of S[i] and S[j]
endfor

```

Pseudo-random generation algorithm (PRGA)



The lookup stage of RC4. The output byte is selected by looking up the values of $S[i]$ and $S[j]$, adding them together modulo 256, and then using the sum as an index into S ; $S(S[i] + S[j])$ is used as a byte of the key stream K .

For as many iterations as are needed, the PRGA modifies the state and outputs a byte of the keystream. In each iteration, the PRGA:

- increments i ;
- looks up the i th element of S , $S[i]$, and adds that to j ;
- exchanges the values of $S[i]$ and $S[j]$, then uses the sum $S[i] + S[j]$ (modulo 256) as an index to fetch a third element of S (the keystream value K below);
- then bitwise exclusive ORed (XORed) with the next byte of the message to produce the next byte of either ciphertext or plaintext.

```

i := 0
j := 0
while GeneratingOutput:
    i := (i + 1) mod 256
    j := (j + S[i]) mod 256
    swap values of S[i] and S[j]
    t := (S[i] + S[j]) mod 256
    K := S[t]
    output K
endwhile

```

Thus, this produces a stream of $K[0]$, $K[1]$, ... which are XORed with the *plaintext* to obtain the *ciphertext*. So $\text{ciphertext}[I] = \text{plaintext}[I] \oplus K[I]$.

RC4-based random number generators

Several operating systems include `arc4random`, an API originating in OpenBSD providing access to a random number generator originally based on RC4. The API allows no seeding, as the function initializes itself using `/dev/random`. The use of RC4 has been phased out in most systems implementing this API. Man pages for the new `arc4random` include the backronym "A Replacement Call for Random" for ARC4 as a mnemonic, as it provides better random data than `rand()` does.

- In OpenBSD 5.5, released in May 2014, `arc4random` was modified to use ChaCha20. The implementations of `arc4random` in FreeBSD, NetBSD also use ChaCha20.
 - ❖ Linux typically uses glibc, which did not offer `arc4random` until 2022. Instead, a separate library, `libbsd`, offers the function; it was updated to use ChaCha20 in 2016. In 2022, glibc added its own version of `arc4random`, also based on ChaCha20.
- According to manual pages shipped with the operating system, in the 2017 release of macOS and iOS operating systems, Apple replaced RC4 with AES in its implementation of `arc4random`.

Advantages of RC4

- Simple to implement in software
- Very fast encryption and decryption
- Works on variable key lengths
- Requires minimal system resources

Disadvantages & Weaknesses

- Initial bytes of keystream are biased (not truly random)
- Susceptible to key recovery attacks if key reuse occurs
- WEP (Wired Equivalent Privacy) using RC4 was broken due to improper key management
- Not recommended for secure communications today

Applications (Historical)

- SSL/TLS (older versions)
- WEP (Wireless LAN security)
- Microsoft Office file encryption
- Secure Shell (SSH)

Key distribution

Key distribution refers to the process of securely sharing cryptographic keys between parties so they can communicate securely using encryption. Without secure key distribution, even strong encryption algorithms cannot provide confidentiality or integrity.

In cryptography, two main types of keys need to be distributed:

- Symmetric key (same key for encryption and decryption)
- Public/private key pair (in asymmetric cryptography)

2. Objectives of Key Distribution

- Ensure secure delivery of keys over insecure channels
- Prevent interception or tampering of keys during transmission
- Ensure authenticity: the key comes from a trusted source
- Enable scalability and efficiency, especially in large networks

3. Key Distribution in Symmetric Cryptography

In symmetric systems, both parties must use the same secret key. Hence, the challenge lies in securely sharing that key.

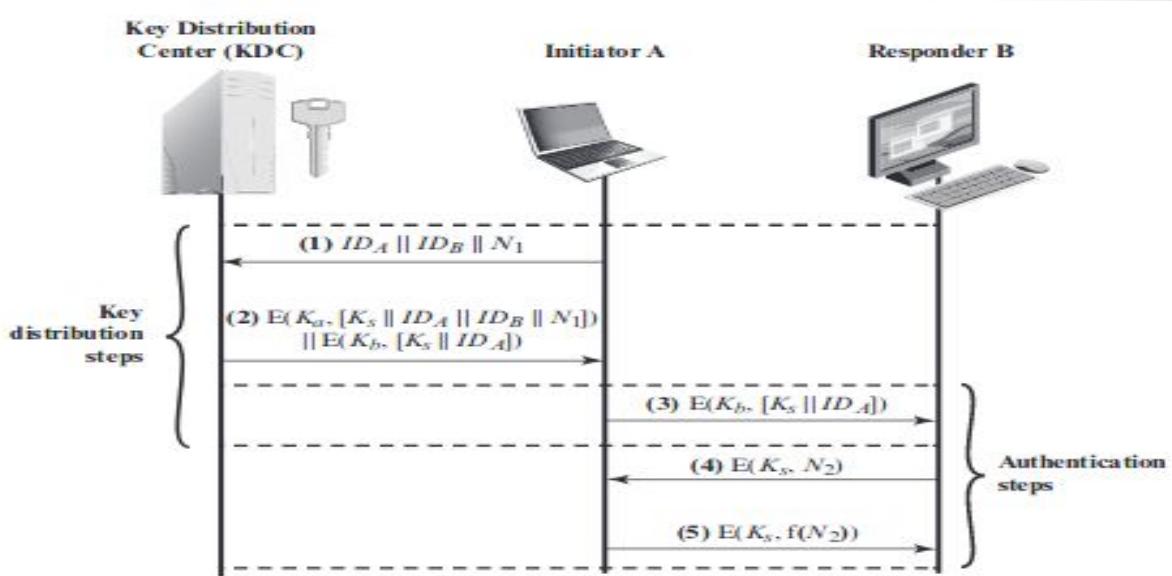
Methods of Key Distribution (Symmetric):

3.1 Manual Key Exchange

- Keys are physically handed over or transmitted by a secure channel (e.g., USB drive, paper). Secure but not scalable.

3.2 Key Distribution Center (KDC)

- A trusted third-party system that helps distribute symmetric keys to parties.
- Used in Kerberos protocol.
- Every user shares a unique secret key with the KDC.



3.3 Using Asymmetric Encryption for Key Exchange

- A temporary symmetric session key is encrypted using the receiver's public key.
- Example: Hybrid cryptosystems in SSL/TLS.

3.4 Diffie–Hellman Key Exchange (explained further below)

4. Key Distribution in Asymmetric Cryptography

In asymmetric cryptography, each user has a public key and a private key.

Methods of Key Distribution (Asymmetric):

4.1 Public Key Infrastructure (PKI)

- Uses a hierarchy of trusted Certificate Authorities (CAs) to bind public keys to identities.
- CAs issue digital certificates, ensuring that a public key truly belongs to a specific person or entity.

4.2 Web of Trust

- A decentralized model (used in PGP).
- Users validate and sign each other's public keys.

4.3 Direct Public Key Exchange

- The public key is sent directly, but must be authenticated to prevent man-in-the-middle (MitM) attacks.

5. Diffie–Hellman Key Exchange (D-H)

- First practical method for secure key exchange over an insecure channel.
- Based on the difficulty of computing discrete logarithms.

How it works:

- Two parties agree on public values p (a large prime) and g (a primitive root modulo p).
- Each selects a private key (a and b), and computes public values $A = g^a \text{ mod } p$, $B = g^b \text{ mod } p$.
- They exchange A and B .
- Each computes shared key:
Alice: $K = B^a \text{ mod } p$
Bob: $K = A^b \text{ mod } p$
 K is the same for both.

Advantages:

- No prior shared secret needed
- Key never transmitted directly

Disadvantages:

- Vulnerable to MitM attacks without authentication

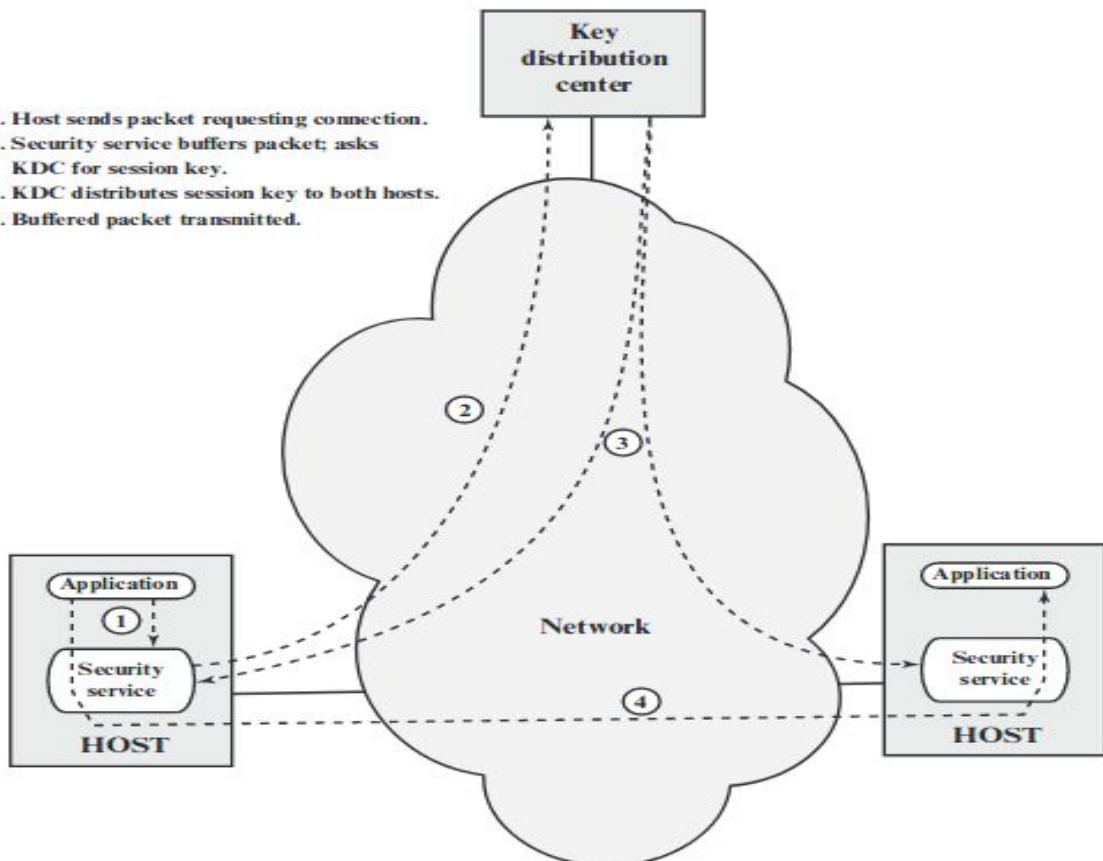
6. Challenges in Key Distribution

- Scalability: In a network of n users, symmetric key exchange requires $O(n^2)$ keys
- Trust: Users must trust the authenticity of distributed keys
- Security: Interception, spoofing, replay, or modification attacks can compromise keys
- Key freshness: Ensuring that old compromised keys cannot be reused

7. Solutions to Improve Key Distribution

- Use of session keys and ephemeral keys
- Certificate-based authentication (X.509)
- Secure key agreement protocols: TLS, IKE (used in IPsec), etc.
- Blockchain-based key management (emerging technology)

Automatic Key Distribution for Connection-Oriented Protocol



ASSIGNMENTS

Set 1

- 1) Explain in detail about Simplified Data Encryption Standard with given
Key:1010000010 PT:10010111
- 2) Perform SubByte Transformation, Shift Rows, Mix Columns and Add Round Key
for the following input matrix

Input Matrix				Round Key			
12	C0	F3	57	21	0C	3F	75
23	A9	B2	85	36	9A	B2	58
98	8A	F1	89	88	AF	11	99
39	69	47	07	96	19	74	70

- 3) Explain Euclid Algorithm find gcd of 1970 and 1066 by using Euclid Algorithm. Is the two number are relatively prime or not?
- 4) In AES, how the encryption key is expanded to produce keys for the 10 rounds.
- 5) Briefly define a Group, Ring and field. If every element of group G is its own inverse then group G is commutative .Prove.

Set 2

- 1) Using the extended Euclidean algorithm, find the greatest common divisor of the following pairs.
 - .a) 4 and 7
 - b) 291 and 42
 - c) 84 and 320
 - d) 400 and 60
- 2) What is the purpose of diffusion in the design of block ciphers?
- 3) What are the two best known general attacks against block ciphers?
- 4) Write a program to implement Simplified DES algorithm.
- 5) What is the purpose of confusion in the design of block ciphers?

ASSIGNMENTS

Set 3

- 1) Find modulo exponentiation for $5117 \text{ mod } 19$.
- 2) Find $1234^{-1} \pmod{4321}$.
- 3) Find GCD (12345,2424) using Euclid algorithm
- 4) Using Extended Euclidean Algorithm, find the multiplicative inverse of 826 mod 2789.
- 5) Find GCD(1424, 3084)

Set 4

- 1) Find GCD(2740, 1760) using Euclidean Algorithm
- 2) Find the modulo exponentiation of $5^{117} \text{ mod } 19$
- 3) Find $11^7 \text{ mod } 13$
- 4) Find the GCD of 270 and 192
- 5) Calculate the DES S-box output for the given data $B=101111$

Set 5

- 1) Distinguish between stream ciphers and block cipher.
- 2) Find GCD (17,22) using Euclidean Algorithm
- 3) Find GCD (20,35) using Euclidean Algorithm
- 4) Find GCD (30,75) using Euclidean Algorithm
- 5) Find GCD (27,45) using Euclidean Algorithm

TWO MARKS Q & A

Define Group. (CO2, K1)

A group is a set of elements with a single binary operation that satisfies additive closure, additive associativity, additive identity additive inverse axioms.

Example: The set of nonzero real numbers under multiplication is a group.

Define Ring. (CO2, K1)

A ring is a set of elements with two binary operation addition and multiplication that satisfies all the additive axioms like closure, associativity, identity, inverse, commutativity and multiplicative axioms like closure, associativity and distributive laws. Ring is a superset of group and abelian group.

Example: With respect to addition and multiplication, the set of all -square matrices over the real numbers is a ring.

Define Field. (CO2, K1)

A field is a set of elements on which two arithmetic operations (addition and multiplication) have been defined and which has the properties of ordinary arithmetic,

such as closure, associativity, commutativity, distributivity, and having both additive and multiplicative inverses.

Define Finite Field. (CO2, K1)

A finite field or Galois Field is simply a field with a finite number of elements. The order or number of elements in finite field is p^n . Finite Filed is represented as $GF(p^n)$

Mention the use of Galois Field is used in cryptography.(CO2, K2)

Galois field is useful for cryptography because its arithmetic properties allows it to be used for scrambling and descrambling of data.

Mention the role of polynomials in finite fields. (CO2, K2)

The elements of $GF(p)$ are integers, while elements of $GF(p^n)$ are polynomials, because integers modulo p^n do not form a field. So, to produce finite field of non-prime order (i.e. p^n . Only p is prime and p^n is non-prime), irreducible polynomials are required.

Differentiate block and stream cipher. (CO2, K2)

A stream cipher is one that encrypts a digital data stream one bit or one byte at a time. Example: RC4

A block cipher is one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length. Typically, a block size of 64 or 128 bits is used. Example: DES, AES.

List out the requirements for block cipher. (CO2, K2)

- The plaintext block of n bits produce a ciphertext block of n bits. There are 2^n possible different plaintext blocks.
- It should be reversible. $D(K, E(K, M)) = M$ for all M (Plaintext block)

Limitations of ideal block cipher. (CO2, K2)

- If the block size is small, for example $n=4$, then it is equivalent to classical substitution cipher and vulnerable to statistical analysis.
- If n is large, for example $n=64$, then for reversible mapping key of length $n \times 2^n$ bits are required (for $n=64$, it is $64 \times 2^{64} \approx 10^{21}$ bits). It is impractical.

Define Feistel Cipher. (CO2, K1)

It is a design or a structure used to construct block cipher. It consists of a number of identical rounds of processing. In each round, a substitution is performed on one half of the data being processed, followed by a permutation that swaps the two halves. The original key is expanded so that a different key is used for each round.

Why larger key size means greater security? (CO2, K2)

Larger key size ensures greater resistance against brute-force attacks and also ensures greater confusion.

How diffusion is achieved in a block cipher? (CO2, K2)

Diffusion can be achieved by repeatedly performing some permutation on the data followed by applying a function to that permutation.

How confusion is achieved in a block cipher? (CO2, K2)

Confusion is achieved by the use of a complex substitution algorithm.

Why larger block size means greater security? (CO2, K2)

Larger block size ensures greater diffusion. So that plain-text patterns are dissipated into cipher-text. Greater diffusion corresponds to greater security.

What is avalanche effect? (CO2, K2)

It is a desirable property of an encryption algorithm in which a small change in either the plaintext or the key should produce a significant change in the ciphertext. In particular, a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext.

What is the purpose of the S-boxes in DES? (CO2, K2)

It does substitution function to achieve confusion in cipher. The substitution function is non-linear and hence cryptanalysis will be difficult.

What is differential cryptanalysis? (CO2, K2)

It is a form of chosen plaintext attack. The chosen plaintexts with particular XOR difference patterns are encrypted. The difference patterns of the resulting ciphertext provide information that can be used to determine the encryption key.

Compare Feistel structure and Substitution-Permutation Network (SPN). (CO2, K2)

Feistel Structure	SPN
In a round, only one half of the data block is involved in transformation.	In a round, the full data block is subjected to transformation and hence desired effect is achieved in fewer rounds.
DES is an example of Feistel structure	AES is an example of SPN
Both use Substitution and Permutation to achieve Confusion and Diffusion respectively.	

Compare DES and AES. (CO2, K2)

DES	AES
DES uses Feistel structure.	AES uses SPN network.
DES uses 64 bit plaintext block and 56 bit key. It uses 16 rounds.	AES uses 128 bit plaintext block. Key length can be 128 or 192 or 256 bit. Number of rounds can be 10 or 12 or 14 corresponding to key length.
DES is slower and less secure. It is no longer used for commercial applications.	AES is faster and more secure.

PART B

1. Find $17^{-1} \bmod 192$ using Extended Euclid Algorithm. (CO2 , K3)
2. Find gcd (12345,2424) using Euclid algorithm. (CO2,K3)
3. Explain groups, abelian group, rings. (CO2,K2)
4. Explain in detail about finite fields (CO2,K2)
5. Discuss in detail about SDES algorithm. (CO2,k1)
6. Illustrate in detail about Block Cipher principles.(CO2,K3)
7. Discuss in detail about Block cipher modes of operation.(CO2,K2)
8. What do you mean by AES? Diagrammatically illustrate the structure of AES and describe steps in AES encryption process with example. (CO2,K3)
9. Find the multiplicative inverse of 13 modulo 50. Find the multiplicative inverse of 50 modulo 13. Can you use previous work to find the multiplicative inverse of 7 modulo 27? (CO2,K3)
10. With respect to the modes of operations of modern block ciphers, answer the following questions: (CO2,K4)
 - (i). In OFB mode, the entire cipher text block 11 is corrupted ($r=8$), find the possible corrupted bits in the plaintext.
 - (ii) In CTR mode, blocks 3 and 4 are entirely corrupted. Find the possible corrupted bits in the plain text.
 - (iii) Show the encryption and decryption diagram for ECB mode (only the last two blocks) when cipher text stealing is used.

GATE QUESTIONS

1. The value of the expression $13^{99} \text{ (MOD } 17)$ in the range of 0 to 16 is----- (GATE 2016 set 2)

Answer:

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

So, $13^{16} \equiv 1 \pmod{17}$.

And, $13^{96} = 13^{16 \times 6} \equiv 1 \pmod{17}$.

We are left with $13^{99} = 13^{96} \times 13^3 \equiv 13^3 \pmod{17} \equiv 2197 \pmod{17}$ which is 4.

2. Which of the following are symmetric key cryptographic algorithms? (GATE 2013)

- A) Feistel Structure
- B) DES
- C) RC4
- D) RSA

Answer: B) DES and C) RC4

3. In the AES algorithm, the key size can be: (GATE 2015)

- A) 128, 192, or 256 bits
- B) 64, 128, or 256 bits
- C) 128, 160, or 192 bits
- D) 256, 512, or 1024 bits

Answer: A) 128, 192, or 256 bits

4. Which of the following statements is true for pseudorandom number generators? (GATE 2014)

- A) They produce truly random numbers
- B) Their output is predictable if the seed is known
- C) They do not require a seed
- D) They are slower than hardware random generators

Answer: B) Their output is predictable if the seed is known

GATE QUESTIONS

5. What is the value of the expression $1399 \bmod 17$, in the range 0 to 16?(GATE 2016 (Set 2))

Options:

- A) 3
- B) 5
- C) 7
- D) 9

Answer: B) 5

Explanation:

To compute $1399 \bmod 17$, divide 1399 by 17 and find the remainder.

$1399 \div 17 = 82$ with a remainder of 5.

6. Evaluate the expression: $(7^{100}) \bmod 13$. (GATE 2019)

Options:

- A) 1
- B) 3
- C) 9
- D) 11

Answer: C) 9

Explanation:

Using Fermat's Little Theorem, since 13 is prime and 7 is not divisible by 13:

$$7^{12} \equiv 1 \pmod{13}$$

$$\text{So, } 7^{100} = 7^{(12*8 + 4)} = (7^{12})^8 * 7^4 \equiv 1^8 * 7^4 \pmod{13}$$

$$\text{Compute } 7^4 = 2401$$

$$2401 \bmod 13 = 9$$

GATE QUESTIONS

7.Which of the following are symmetric key cryptographic algorithms? (GATE 2013)

Options:

- A) Feistel Structure
- B) DES
- C) RC4
- D) RSA

Answer: B) DES and C) RC4

Explanation:

DES and RC4 are symmetric key algorithms. RSA is an asymmetric key algorithm. Feistel Structure is a design model used in symmetric ciphers but is not an algorithm itself.

8.In the AES algorithm, the key size can be: (GATE 2015)

Options:

- A) 128, 192, or 256 bits
- B) 64, 128, or 256 bits
- C) 128, 160, or 192 bits
- D) 256, 512, or 1024 bits

Answer: A) 128, 192, or 256 bits

Explanation:

AES supports key sizes of 128, 192, and 256 bits.

GATE QUESTIONS

9.Which of the following statements is true for pseudorandom number generators? (GATE 2014)

Options:

- A) They produce truly random numbers
- B) Their output is predictable if the seed is known
- C) They do not require a seed
- D) They are slower than hardware random generators

Answer: B) Their output is predictable if the seed is known

Explanation:

Pseudorandom number generators produce sequences that are deterministic and can be reproduced if the initial seed is known.

10.Suppose that everyone in a group of NNN people wants to communicate secretly with the $N-1N - 1N-1$ others using a symmetric key cryptographic system. The communication between any two persons should not be decodable by the others in the group. The number of keys required in the system as a whole to satisfy the confidentiality requirement is:

Options:

- A) $2N$
- B) $N(N - 1)$
- C) $\frac{N(N-1)}{2}$
- D) $(N - 1)^2$

Answer: C) $\frac{N(N-1)}{2}$

Explanation:

In a symmetric key cryptographic system, each pair of users requires a unique shared key to ensure secure communication. For a group of NNN people, the total number of unique pairs is given by the combination formula

$$\binom{N}{2} = \frac{N(N-1)}{2}$$

Therefore, $\frac{N(N-1)}{2}$ keys are needed to ensure that every pair can communicate securely without others being able to decode their messages.

Supportive online Certification courses

NPTEL

- ❖ Cryptography and Network Security
- ❖ Introduction to Cryptology
- ❖ Foundations of Cryptography
- ❖ Computational number theory and cryptography

COURSERA

- ❖ Cryptography
- ❖ Applied Cryptography
Number theory and cryptography
- ❖ Cryptography and Information theory
- ❖ Asymmetric cryptography and key management Symmetric Cryptography

UDEMY

- ❖ Introduction to Cryptography
- ❖ Cryptography with python
- ❖ Complete Cryptography master class

Real time Applications

- ❖ Using OpenSSL to encrypt/decrypt
- ❖ data Image encryption
- ❖ AES algorithm in Transactions



Prescribed Text Books & Reference Books

TEXT BOOK:

- ❖ William Stallings, Cryptography and Network Security: Principles and Practice, PHI
8th Edition, 2020.

REFERENCES:

- ❖ 1. C K Shyamala, N Harini and Dr. T R Padmanabhan: Cryptography and Network Security, Wiley India Pvt.Ltd
- ❖ 2.Behrouz A.Foruzan, Cryptography and Network Security, Tata McGraw Hill 2007.
- ❖ 3.Charlie Kaufman, Radia Perlman, and Mike Speciner, Network Security:
PRIVATE Communication in a PUBLIC World, Prentice Hall, ISBN
0-13-046019-2



Mini Projects Suggestions

Set 1

1. Create a software for secret video chat in terminal using strong AES encryptions.
2. Implement and visualize the Rabin public key cipher .
3. Develop a Real-Time Chat software to:
 - i. Implement a real-time chat interface where users can send and receive messages securely.
 - ii. Encrypt messages using the symmetric key before transmission.

Set 2

1. Develop an application for encryption & decryption of Image through Triple DES.
2. Implement a general dialog to filter and transform a text document (e.g. before encrypting it)
3. Implement a Key Exchange software including the following:
 1. Implement a secure key exchange mechanism for symmetric encryption.
 2. Use a key derivation function to derive encryption and decryption keys from user passwords.

Set 3

1. Develop an Encrypted wallet system, for transfer and receipt.
2. Develop a software for Authentication and Authorization:
 1. Ensure that users are authenticated before allowing access to the chat.
 2. Implement basic authorization mechanisms to control user access.

Mini Projects Suggestions

Set 4

1. Develop an Encryptor for Visual Cryptography.
2. Develop a software to model Symmetric Encryption to include the following features:
 - i. Choose a symmetric encryption algorithm (e.g., AES) and implement it for message encryption and decryption.
 - ii. Allow users to select different encryption modes (e.g., CBC, CTR).

Set 5

1. Develop a GUI software that allow you to encrypt messages with secure symmetric and asymmetric ciphers.
2. Develop an User Registration software that includes the following:
 - ii. Allow users to register with a username and password.
 - iii. Store user credentials securely using a hash function.



Mini Projects

Suggestions

- 1. Visual DES Cryptosystem**
 - Create a visual simulation of DES, breaking down each step (initial permutation, Feistel rounds, etc.) for educational purposes.
 - Include an analysis section to demonstrate differential and linear cryptanalysis.
- 2. SDES Cryptosystem with Attack Simulation**
 - Implement SDES (Simplified DES) and simulate known plaintext and brute force attacks to measure its strength.
 - Compare this with actual DES to highlight the differences.
- 3. Block Cipher Playground**
 - Develop an interactive tool for experimenting with different block cipher modes (ECB, CBC, CFB, OFB, CTR).
 - Include visualizations to show how plaintext changes affect ciphertext under each mode.
- 4. RC4 Implementation and Analysis**
 - Implement RC4 and analyze its weaknesses.
 - Demonstrate how key biases can lead to security vulnerabilities.
- 5. PRNG Security Testing**
 - Implement a few common PRNGs (e.g., Linear Congruential Generator, Mersenne Twister) and evaluate their security.
 - Test them for randomness using statistical tests like the Diehard or NIST test suite.
- 6. Hybrid Key Distribution System**
 - Create a hybrid key distribution system that uses symmetric ciphers for fast data encryption and public-key cryptography for secure key exchange.
- 7. Custom Cryptographic Protocol Design**
 - Design and implement a secure messaging protocol using symmetric ciphers, key distribution, and finite field arithmetic.
 - Include features like message integrity, replay protection, and confidentiality.
- 8. Lightweight Cryptography for IoT**
 - Implement lightweight block ciphers like PRESENT or SPECK, and evaluate their performance for low-power devices.

ASSESSMENT SCHEDULE

S.NO	Name of the Assessment	Portion	Proposed Date
1	First Internal Assessment	Unit-1 &Unit 2	
2	Second Internal Assessment	Unit-3 &Unit 4	
3	Model Examination	Unit 1-Unit 5	



R.M.K.
GROUP OF
INSTITUTIONS

Thank you

Disclaimer:

This document is confidential and intended solely for the educational purpose of RMK Group of Educational Institutions. If you have received this document through email in error, please notify the system manager. This document contains proprietary information and is intended only to the respective group / learning community as intended. If you are not the addressee you should not disseminate, distribute or copy through e-mail. Please notify the sender immediately by e-mail if you have received this document by mistake and delete this document from your system. If you are not the intended recipient you are notified that disclosing, copying, distributing or taking any action in reliance on the contents of this information is strictly prohibited.