Detailed Syllabus

Module No.	Unit No.	Detailed Content	Hrs.
1		**Introduction and Number Theory**	10
	1.1	Security Goals, Services, Mechanisms and attacks, The OSI security architecture, Network security model, Classical Encryption techniques, Symmetric cipher model, mono-alphabetic and poly - alphabetic substitution techniques : Vigenere cipher, playfair cipher, Hill cipher, transposition techniques : keyed and keyless transposition ciphers, steganography. **(Refer Chapter 1)**	
	1.2	Modular Arithmetic and Number Theory : Euclid's algorithm - Prime numbers – Fermat's and Euler's theorem - Testing for primality - The Chinese remainder theorem, Discrete logarithms. **(Refer Chapter 2)**	
2		**Symmetric and Asymmetric key Cryptography and key Management**	12
	2.1	Block cipher principles, block cipher modes of operation, DES, Double DES, Triple DES, Advanced Encryption Standard (AES), Stream Ciphers : RC5 algorithm. **(Refer Chapter 3)**	
	2.2	Public key cryptography : Principles of public key cryptosystems - The RSA algorithm, The knapsack algorithm, ElGamal Algorithm. **(Refer Chapter 4)**	
	2.3	Key management techniques : using symmetric and asymmetric algorithms and trusted third party. Diffie Hellman Key exchange algorithm. **(Refer Chapter 5)**	
3		**Hashes, Message Digests and Digital Certificates**	06
	3.1	Cryptographic hash functions, Properties of secure hash function, MD5, SHA-1, MAC, HMAC, CMAC. **(Refer Chapter 6)**	
	3.2	Digital Certificate : X.509, PKI. **(Refer Chapter 7)**	
4		**Authentication Protocols & Digital signature schemes**	08
	4.1	User Authentication and Entity Authentication, One - way and mutual authentication schemes, Needham Schroeder Authentication protocol, Kerberos Authentication protocol. **(Refer Chapter 8)**	

Scanned by CamScanner

Module No.	Unit No.	Detailed Content	Hrs.
	4.2	Digital Signature Schemes - RSA, ElGamal and Schnorr signature schemes. **(Refer Chapter 9)**	
5	**Network Security and Applications**		10
	5.1	Network security basics : TCP/IP vulnerabilities (Layer wise), Packet Sniffing, ARP spoofing, port scanning, IP spoofing, TCP syn flood, DNS Spoofing. **(Refer Chapter 10)**	
	5.2	Denial of Service : Classic DOS attacks, Source Address spoofing, ICMP flood, SYN flood, UDP flood, Distributed Denial of Service, Defenses against Denial of Service Attacks. **(Refer Chapter 11)**	
	5.3	Internet Security Protocols : SSL, IPSEC, Secure Email : PGP, Firewalls, IDS and types, Honey pots. **(Refer Chapter 12)**	
6	**System Security**		06
	6.1	Software Vulnerabilities : Buffer Overflow, Format string, cross - site scripting, SQL injection, Malware : Viruses, Worms, Trojans, Logic Bomb, Bots, Rootkits. **(Refer Chapter 13)**	

□□□

Scanned by CamScanner

Lab Syllabus

Lab Code	Lab Name	Credit
CSL604	System Security Lab	01

Lab Outcome :

Learner will able to

1. To be able to apply the knowledge of symmetric cryptography to implement simple ciphers.
2. To be able to analyze and implement public key algorithms like RSA and El Gamal.
3. To analyze and evaluate performance of hashing algorithms.
4. To explore the different network reconnaissance tools to gather information about networks.
5. To explore and use tools like sniffers, port scanners and other related tools for analysing packets in a network.
6. To be able to set up firewalls and intrusion detection systems using open source technologies and to explore email security.
7. To be able to explore various attacks like buffer - overflow, and web - application attacks.

Suggested Experiment List : (Any 10)

Sr. No.	Description
1	Design and Implementation of a product cipher using Substitution and Transposition ciphers.
2	Implementation and analysis of RSA cryptosystem and Digital signature scheme using RSA/El Gamal.
3	Implementation of Diffie Hellman Key exchange algorithm.
4	For varying message sizes, test integrity of message using MD - 5, SHA - 1, and analyse the performance of the two protocols. Use crypt APIs.

Scanned by CamScanner

Sr. No.	Description
5	Study the use of network reconnaissance tools like WHOIS, dig, traceroute, nslookup to gather information about networks and domain registrars.
6	Study of packet sniffer tools : wireshark, : 1. Download and install wireshark and capture icmp, tcp, and http packets in promiscuous mode. 2. Explore how the packets can be traced based on different filters.
7	Download and install nmap. Use it with different options to scan open ports, perform OS fingerprinting, do a ping scan, tcp port scan, udp port scan, xmas scan etc.
8	Detect ARP spoofing using nmap and/or open source tool ARPWATCH and wireshark. Use arping tool to generate gratuitous arps and monitor using wireshark.
9	Simulate DOS attack using Hping, hping3 and other tools.
10	Simulate buffer overflow attack using Ollydbg, Splint, Cppcheck etc.
11	a. Set up IPSEC under LINUX. b. Set up Snort and study the logs.
12	Setting up personal Firewall using iptables.
13	Explore the GPG tool of linux to implement email security.
14	SQL injection attack, Cross - Cite Scripting attack simulation.

Scanned by CamScanner

Module 1

Chapter 1 : Introduction to Cryptography 1-1 to 1-57

Syllabus : Security Goals, Services, Mechanisms and attacks, The OSI security architecture, Network security model, Classical Encryption techniques, Symmetric cipher model, mono-alphabetic and poly-alphabetic substitution techniques : Vigenere cipher, playfair cipher, Hill cipher, transposition techniques : keyed and keyless transposition ciphers, steganography.

1.1 Introduction
✓ **Syllabus Topic :** Security Goals 1-1
1.2 Security Goal **(Dec. 15)** 1-2
✓ **Syllabus Topic :** Services 1-2
1.3 Security Service **(Dec. 16)** 1-4
✓ **Syllabus Topic :** Mechanisms and Attacks 1-4
1.4 Security Mechanisms **(Dec. 15, May 16)** 1-8
1.4.1 Specific Security Mechanism / Attack Prevention 1-8
1.4.2 Pervasive Security Mechanisms / Attack Detection 1-8
1.4.3 Attack Avoidance 1-9
1.5 Security Attack **(May 18)** 1-9
1.5.1 Difference between Active Attack and Passive Attack 1-10
✓ **Syllabus Topic :** The OSI Security Architecture 1-16
1.6 The OSI Security Architecture 1-16
✓ **Syllabus Topic :** Network Security Model 1-16
1.7 Operational Model for Network Security 1-17
1.8 Basic Terminology in Network Security 1-17
1.8.1 Cryptanalysis/ Cryptographic Attacks 1-18
✓ **Syllabus Topic :** Classical Encryption Techniques 1-20
1.9 Encryption Methods 1-23
✓ **Syllabus Topic :** Symmetric Cipher Model 1-23
1.9.1 Symmetric Key Cryptography 1-23
1.9.2 Asymmetric Key Cryptography 1-25
1.9.3 Difference between Symmetric and Asymmetric Key Cryptography 1-28
1.10 Block Cipher Principles 1-29
1.10.1 Stream Cipher 1-29

Scanned by CamScanner

1.10.2 Block Cipher 1-31
1.10.3 Differentiate between Stream and Block Cipher (Dec. 16) 1-32
1.10.4 Confusion and Diffusion 1-32
1.10.4(A) Difference between Confusion and Diffusion 1-33
✓ **Syllabus Topic :** Mono-alphabetic and Poly-alphabetic Substitution Techniques 1-34
1.11 Substitution Cipher Techniques (Dec. 15) 1-34
1.11.1 Caesar Cipher 1-35
1.11.2 Monoalphabetic Cipher 1-36
1.11.3 Polyalphabetic Cipher (Dec. 15) 1-37
1.11.3(A) Procedure of Polyalphabetic Cipher 1-37
1.11.3(B) Difference between Polyalphabetic and Monoalphabetic (Dec. 17) 1-39
✓ **Syllabus Topic :** Playfair Cipher 1-39
1.11.4 Playfair Cipher 1-39
1.11.5 One Time Pad (Vernam Cipher) 1-46
✓ **Syllabus Topic :** Hill Cipher 1-48
1.11.6 Hill Cipher 1-48
✓ **Syllabus Topic :** Transposition Techniques 1-51
1.12 Transposition Cipher Techniques (May 16) 1-51
✓ **Syllabus Topic :** Keyed Transposition Cipher 1-52
1.12.1 Columnar Transposition Technique 1-52
1.12.2 Keyless Transposition Techniques 1-54
✓ **Syllabus Topic :** Steganography 1-56
1.13 Steganography Applications and Limitations 1-56
• Chapter Ends 1-57

Chapter 2 : Modular Arithmetic and Number Theory 2-1 to 2-21

Syllabus : Modular Arithmetic and Number Theory : Euclid's algorithm - Prime numbers-Fermat's and Euler's theorem - Testing for primality - The Chinese remainder theorem, Discrete logarithms.

2.1 Modular Arithmetic 2-1
2.1.1 Mathematical Background 2-1
✓ **Syllabus Topic :** Prime Numbers 2-2
2.1.1(A) Prime Numbers 2-2
2.1.1(B) What is GCD? 2-2

Scanned by CamScanner

✓ **Syllabus Topic :** Euclid's Theorem 2-4
2.2 Euclid's or Euclidean Algorithm 2-4
2.2.1 Extended Euclidean Algorithm 2-8
✓ **Syllabus Topic :** The Chinese Remainder Theorem 2-11
2.3 Chinese Remainder Theorem 2-11
✓ **Syllabus Topic :** Euler's Theorem 2-15
2.4 Euler Totient Function $\phi(n)$ 2-15
2.4.1 Euler's Theorem 2-16
✓ **Syllabus Topic :** Discrete Logarithm 2-17
2.5 Discrete Logarithm 2-17
2.6 Fermat Theorem 2-19
• Chapter Ends 2-21

Module 2

Chapter 3 : Symmetric Key Cryptography 3-1 to 3-28

Syllabus : Block cipher principles, block cipher modes of operation, DES, Double DES, Triple DES, Advanced Encryption Standard (AES), Stream Ciphers : RC5 algorithm.

✓ **Syllabus Topic :** Block Cipher Principles 3-1
3.1 Block Cipher Principles 3-1
✓ **Syllabus Topic :** Block Cipher Modes of Operation 3-1
3.2 Block Cipher Modes of Operation 3-1
3.2.1 Electronic Codebook (ECB) Mode **(Dec. 16)** 3-2
3.2.2 Cipher Block Chaining (CBC) Mode **(Dec. 16)** 3-3
3.2.3 Cipher Feedback (CFB) Mode 3-4
3.2.4 Output Feedback (OFB) Mode 3-6
3.2.5 Counter (CTR) Mode 3-7
3.2.6 Algorithm Mode Details and Usage 3-9
✓ **Syllabus Topic :** DES 3-9
3.3 Data Encryption Standard (DES) **(Dec. 15, May 16, May 17, May 18)** 3-9
3.3.1 Conceptual View of DES 3-10
3.3.2 Detail Steps of DES 3-11
3.3.3 Initial Permutation (IP) 3-12
3.3.4 Rounds 3-12

Scanned by CamScanner

3.3.5 Final Permutation 3-15
3.3.6 Strength of DES 3-15
3.3.7 Weakness in DES 3-16
✓ **Syllabus Topic :** Double DES 3-16
3.3.8 Double DES 3-16
✓ **Syllabus Topic :** Triple DES 3-17
3.3.9 Triple DES 3-17
✓ **Syllabus Topic :** Advance Encryption Standard (AES) 3-18
3.4 Advance Encryption Standard (AES) 3-18
3.4.1 Introduction to AES 3-18
3.4.2 Silent Features of AES 3-18
3.4.3 AES Encryption and Decryption Process 3-19
3.4.4 Detail Steps for AES Encryption 3-20
3.4.5 AES Decryption 3-24
3.4.5(A) Difference between Data Encryption Standard (DES) and Advance Data Encryption Standard (AES) 3-24
✓ **Syllabus Topic :** RC5 Algorithm 3-25
3.5 RC5 Algorithm 3-25
• Chapter Ends 3-28

Chapter 4 : Public Key Cryptography **4-1 to 4-30**

Syllabus : Public key cryptography : Principles of public key cryptosystems - The RSA algorithm, The knapsack algorithm, ElGamal Algorithm.

4.1 Public Key Cryptosystem with Applications 4-1
4.1.1 Applications for Public - Key Cryptosystem 4-2
✓ **Syllabus Topic :** Requirements and Cryptanalysis 4-3
4.2 Public Key Requirements and Cryptanalysis 4-3
4.2.1 Public Key Cryptanalysis 4-4
✓ **Syllabus Topic :** The RSA Algorithm 4-5
4.3 RSA Algorithm : Working, Key Length, Security **(May 17)** 4-5
4.3.1 Computational Aspects 4-7
4.3.1(A) Exponentiation in Modular Arithmetic 4-7
4.3.1(B) Efficient Operation using the Public Key 4-8

Scanned by CamScanner

4.3.1(C) Efficient Operation using the Private Key 4-9
4.3.1(D) Key Generation (May 16) 4-10
4.3.1(E) The Security of RSA 4-11
4.3.2 Solved Examples on RSA Algorithm 4-11
✓ **Syllabus Topic :** The Knapsack Algorithm 4-25
4.4 Knapsack Algorithm 4-25
4.4.1 Problem Statement 4-26
4.4.2 Dynamic-Programming Approach 4-26
4.4.3 Analysis 4-27
✓ **Syllabus Topic :** ElGamal Algorithm 4-27
4.5 ElGamal Algorithm 4-27
4.5.1 Generation of ElGamal Key Pair 4-28
4.5.2 Encryption and Decryption 4-29
4.5.3 ElGamal Encryption 4-29
4.5.4 ElGamal Decryption 4-29
4.5.5 ElGamal Analysis 4-30
- Chapter Ends 4-30

Chapter 5 : Key Management Techniques **5-1 to 5-24**

Syllabus : Key management techniques : using symmetric and asymmetric algorithms and trusted third party. Diffie Hellman Key exchange algorithm.

5.1 Key Distribution and Management 5-1
5.1.1 Management 5-1
✓ **Syllabus Topic :** Key Management Techniques - Using symmetric and Asymmetric Algorithms and Trusted Third Party 5-2
5.1.2 Symmetric Key Distribution using Symmetric and Asymmetric Encryptions 5-2
5.1.2(A) Symmetric Key Distribution using Symmetric Encryption 5-2
5.1.2(B) Distribution of Symmetric Key (Secret Key) using Asymmetric Encryption 5-8
5.1.3 Distribution of Public Keys 5-10
5.1.4 Key Generation, Distribution, Storage and Usage 5-13
5.1.4(A) Key Generation 5-13
5.1.4(B) Key Distribution 5-13

Scanned by CamScanner

5.1.4(C) Key Storage 5-13
5.1.4(D) Key Usage 5-14
5.1.4(E) Key Validation 5-15
5.1.4(F) Key Updation 5-15
5.1.5 Importance of Key Management 5-15
✓ **Syllabus Topic :** Diffie Hellman Key Exchange Algorithm 5-15
5.2 Diffie Hellman Key Exchange **(Dec. 15, May 17)** 5-15
• Chapter Ends 5-24

Module 3

Chapter 6 : Cryptographic Hash Functions 6-1 to 6-25

Syllabus : Cryptographic hash functions, Properties of secure hash function, MD5, SHA-1, MAC, HMAC, CMAC.

6.1 Hash Functions 6-1
✓ **Syllabus Topic :** Cryptographic Hash Functions 6-1
6.1.1 Cryptographic Hash Functions 6-1
6.1.2 Applications of Cryptographic Hash 6-2
6.2 Simple Hash Functions 6-6
✓ **Syllabus Topic :** Properties of Secure Hash Function 6-8
6.2.1 Properties of Hash Function **(Dec. 17)** 6-8
6.2.2 Characteristics of Simple Hash Function 6-8
6.2.3 Simple Hash Function Requirement and Security **(Dec. 17)** 6-8
6.2.4 Hash Functions Based on Cipher Block Chaining 6-9
✓ **Syllabus Topic :** MD5 6-10
6.3 MD5 Message Digest Algorithm 6-10
✓ **Syllabus Topic :** SHA-1 6-13
6.4 Secure Hash Algorithm (SHA) **(May 17, May 18)** 6-13
6.4.1 Difference between SHA-1 and MD5 **(Dec. 15, Dec. 16, May 18)** 6-16
6.4.2 Applications of Cryptographic Hash Functions 6-17

Scanned by CamScanner

✓ **Syllabus Topic : MAC** 6-18

6.5 Message Authentication Codes **(May 18)** 6-18

6.5.1 Significance of MAC 6-19

6.5.2 Message Authentication Code based on DES 6-19

6.5.3 Mathematical Equation 6-20

✓ **Syllabus Topic : HMAC** 6-20

6.5.4 HMAC (Hash based Message Authentication Code) 6-20

6.5.4(A) Complete HMAC Operation 6-21

6.5.5 Difference between Message Digest and Message Authentication Code 6-22

✓ **Syllabus Topic : CMAC** 6-23

6.6 CMAC (Cipher Based Message Authentication Code) 6-23

6.6.1 The CMAC Tag Generation Process 6-24

6.6.2 The Verification Process 6-24

- Chapter Ends 6-25

Chapter 7 : Digital Certificate **7-1 to 7-07**

Syllabus : Digital Certificate : X.509, PKI.

✓ **Syllabus Topic :** Digital Certificate X.509 7-1

7.1 X.509 Authentication Service/ Digital Certificate **(Dec. 15, Dec. 17)** 7-1

7.1.1 Importance of Digital Certificate 7-3

✓ **Syllabus Topic : PKI** 7-4

7.2 Public Key Infrastructure (PKI) 7-4

7.2.1 Components of PKI 7-5

7.2.1(A) Certification Authority (CA) 7-5

7.2.1(B) Registration Authority (RA) 7-5

7.2.1(C) PKI Clients 7-6

7.2.1(D) Public Key Certificate/ Digital Certificate 7-6

7.2.1(E) Certificate Distribution System (CDS) Repository 7-6

7.2.2 PKI Applications / Services 7-7

- Chapter Ends 7-7

Scanned by CamScanner

Module 4

Chapter 8 : Authentication Protocols 8-1 to 8-11

Syllabus : User Authentication and Entity Authentication, One-way and mutual authentication schemes, Needham Schroeder Authentication protocol, Kerberos Authentication protocol.

✓ **Syllabus Topic :** User Authentication 8-1
8.1 User Authentication 8-1
8.1.1 Means of User Authentication 8-2
✓ **Syllabus Topic :** Entity Authentication 8-2
8.2 Entity Authentication 8-2
8.2.1 Physical and Legal Identities 8-3
8.3 Authentication Protocol 8-3
8.3.1 Why there is a Need of Mutual Authentication Protocol ? 8-5
✓ **Syllabus Topic :** Needham Schroeder Authentication Protocol 8-5
8.3.2 Needham - Schroeder Protocol 8-5
✓ **Syllabus Topic :** Kerberos Authentication Protocol 8-7
8.4 Kerberos Authentication Protocol **(May 16, May 18)** 8-7
8.4.1 Difference between Kerberos Version 4 and Version 5 8-11
• Chapter Ends 8-11

Chapter 9 : Digital Signature Schemes 9-1 to 9-08

Syllabus : Digital Signature Schemes - RSA, ElGamal and Schnorr signature schemes.

9.1 Digital Signature **(May 16, Dec. 16)** 9-1
9.2 Digital Signature Goals 9-2
✓ **Syllabus Topic :** Digital Signature Schemes 9-4
9.3 Digital Signature Algorithms/ Schemes **(May 16, Dec. 16)** 9-4
✓ **Syllabus Topic :** RSA Signature Scheme 9-4
9.3.1 RSA Signature Scheme 9-4
✓ **Syllabus Topic :** ElGamal Signature Scheme 9-5
9.3.2 ElGamal Scheme 9-5
✓ **Syllabus Topic :** Schnorr Signature Schemes 9-7
9.3.3 Schnorr Digital Significance Scheme 9-7
• Chapter Ends 9-8

Scanned by CamScanner

Module 5

Chapter 10 : Network Security Basics 10-1 to 10-13

Syllabus : Network security basics : TCP/IP vulnerabilities (Layer wise), Packet Sniffing, ARP spoofing, port scanning, IP spoofing, TCP syn flood, DNS Spoofing.

✓ **Syllabus Topic :** TCP/IP Vulnerabilities (Layer wise) 10-1

10.1 TCP/IP Vulnerabilities (Layer Wise) 10-1

10.1.1 Application Layer 10-1

10.1.2 Transport Layer 10-3

10.1.3 Network Layer 10-4

10.1.4 Data Link Layer 10-7

10.1.5 Physical Layer 10-8

✓ **Syllabus Topic :** Packet Sniffing 10-9

10.2 Packet Sniffing 10-9

✓ **Syllabus Topic :** ARP Spoofing 10-9

10.3 ARP Spoofing 10-9

10.3.1 What Is ARP Spoofing? 10-9

10.3.2 ARP Spoofing Attacks 10-9

✓ **Syllabus Topic :** Port Scanning 10-10

10.4 Port Scanning 10-10

10.4.1 Types of Port Scans 10-10

✓ **Syllabus Topic :** IP Spoofing 10-11

10.5 IP Spoofing **(Dec. 15)** 10-11

✓ **Syllabus Topic :** TCP SYN Flood 10-11

10.6 TCP SYN Flood 10-11

10.6.1 Attack Description 10-11

✓ **Syllabus Topic :** DNS Spoofing 10-12

10.7 DNS Spoofing 10-12

10.7.1 Methods for Executing a DNS Spoofing Attack 10-12

• Chapter Ends 10-13

Scanned by CamScanner

Chapter 11 : Denial of Service 11-1 to 11-14

Syllabus : Denial of Service : Classic DOS attacks, Source Address spoofing, ICMP flood, SYN flood, UDP flood, Distributed Denial of Service, Defenses against Denial of Service Attacks.

✓ **Syllabus Topic :** Denial of service - Classic DOS Attacks 11-1
11.1 DOS and DDOS Attacks 11-1
11.1.1 DOS Attacks **(Dec. 15, May 16, Dec. 16, May 17, Dec. 17)** 11-1
11.1.1(A) Classification of Attacks 11-2
11.1.1(B) Types of DOS Attacks **(May 17)** 11-3
✓ **Syllabus Topic :** Source Address Spoofing 11-5
11.2 Source Address Spoofing 11-5
✓ **Syllabus Topic :** ICMP Flood 11-6
11.3 ICMP Flood 11-6
✓ **Syllabus Topic :** SYN flood 11-7
11.4 SYN Flood 11-7
✓ **Syllabus Topic :** UDP Flood 11-8
11.5 UDP Flood 11-8
✓ **Syllabus Topic :** Distributed Denial of Service 11-9
11.6 Distributed Denial of Service Attacks 11-9
11.6.1 Distributed Denial of Service Attacks 11-9
11.6.2 Characteristics of Distributed Denial of Service Attacks 11-10
11.6.3 Methods of Denial of Service Attacks 11-12
✓ **Syllabus Topic :** Defenses against Denial of Service Attacks 11-14
11.7 Defenses against Attacks 11-14
• Chapter Ends 11-14

Chapter 12 : Internet Security Protocols 12-1 to 12-48

Syllabus : Internet Security Protocols : SSL, IPSEC, Secure Email : PGP, Firewalls, IDS and types, Honey pots.

✓ **Syllabus Topic :** Internet Security Protocols - SSL, IPSEC 12-1
12.1 Secure Socket Layer (SSL) **(Dec. 15, Dec. 17)** 12-1
12.1.1 Working of SSL 12-3
12.1.1(A) Handshake Protocol **(May 16)** 12-4

Scanned by CamScanner

12.1.1(B) Alert Protocol 12-7
12.1.1(C) Record Protocol 12-8
✓ **Syllabus Topic :** IPSEC 12-11
12.2 IP Security Protocols **(May 16)** 12-11
12.2.1 Authentication Header 12-11
12.2.2 Encapsulating Security Payload 12-14
12.2.3 Security Association Database **(Dec. 17)** 12-18
✓ **Syllabus Topic :** Firewalls 12-19
12.3 Firewall Introduction **(May 16, Dec. 16)** 12-19
12.3.1 Firewall Characteristics 12-20
12.3.2 Limitations of Firewalls 12-20
12.3.3 Firewall Architecture and Types **(Dec. 16, May 17)** 12-22
12.3.4 Firewall Configurations 12-25
12.4 Introduction to Intrusion Detection 12-27
12.4.1 Intrusion Detection 12-28
✓ **Syllabus Topic :** IDS and Types 12-29
12.4.2 Intrusion Detection System : Need, Methods, Types of IDS **(May 16)** 12-29
12.4.3 Intrusion Detection Methods/ Techniques 12-33
12.4.3(A) Signature Based Detection 12-33
12.4.3(B) Anomaly Based Detection 12-34
12.4.3(C) Stateful Protocol Analysis 12-36
12.4.4 Types of IDS 12-36
12.4.4(A) Network based IDS (NIDS) 12-37
12.4.4(B) Host Based IDS (HIDS) 12-39
✓ **Syllabus Topic :** Secure Email - PGP 12-39
12.5 Electronic Mail Security : Pretty Good Privacy **(May 16)** 12-39
12.5.1 Working of Pretty Good Privacy **(Dec. 15)** 12-41
12.5.2 Backdoors and Key Escrow in PGP 12-45
✓ **Syllabus Topic :** Honey Pots 12-47
12.6 Honeypot 12-47
• Chapter Ends 12-48

Scanned by CamScanner

Module 6

13-1 to 13-21

Chapter 13 : Software Vulnerabilities

Syllabus : Software Vulnerabilities : Buffer Overflow, Format string, cross-site scripting, SQL injection, Malware : Viruses, Worms, Trojans, Logic Bomb, Bots, Rootkits. 13-1

13.1 Program Security 13-1

13.1.1 Secure Programs 13-2

13.1.2 Non-malicious Program Errors **(Dec. 15)** 13-4

✓ **Syllabus Topic :** Malware - Logic Bomb, Bots 13-4

13.1.3 Malicious Software 13-6

✓ **Syllabus Topic :** Viruses and Worms 13-6

13.1.4 Virus and Worms **(Dec. 15, Dec. 16)** 13-7

13.1.4(A) Types of Virus 13-8

13.1.4(B) Types of Computer Worms 13-9

13.1.4(C) Difference between Virus and Worm 13-10

✓ **Syllabus Topic :** Malware - Trojans, Rootkits 13-10

13.1.5 Targeted Malicious Code 13-12

13.1.6 Controls against Program Threats 13-13

✓ **Syllabus Topic :** Software Vulnerability - Buffer Overflow 13-13

13.2 Buffer Overflow **(Dec. 15)** 13-15

✓ **Syllabus Topic :** Software Vulnerability - Format String 13-15

13.3 Format String Attacks 13-18

✓ **Syllabus Topic :** Cross Site Scripting 13-18

13.4 Cross Site Scripting **(Dec. 16, Dec. 17)** 13-19

13.4.1 Stored and Reflected XSS Attacks 13-19

13.4.2 Stored XSS Attacks 13-19

13.4.3 Reflected XSS Attacks 13-19

13.4.4 Other Types of XSS Vulnerabilities 13-20

13.4.5 XSS Attack Consequences 13-20

✓ **Syllabus Topic :** SQL Injection 13-20

13.5 SQL Injection **(Dec. 17)** 13-21

- Chapter Ends 13-21

- **Lab Manual** ..... L-1 to L-66

□□□

Scanned by CamScanner

CHAPTER 1

Module 1

Introduction to Cryptography

Syllabus

Security Goals, Services, Mechanisms and attacks, The OSI security architecture, Network security model, Classical Encryption techniques, Symmetric cipher model, mono-alphabetic and poly-alphabetic substitution techniques : Vigenere cipher, playfair cipher, Hill cipher, transposition techniques : keyed and keyless transposition ciphers, steganography.

1.1 Introduction

- In today's high technology world, organizations are becoming more and more dependent on information systems. Computer data often travel from one computer to another. Sensitive and confidential information must be protected.
- The threat to information security from criminals and terrorist are increasing. Hence, there is a need to protect the information that is being exchanged between individuals through internet.
- The Internet is ever growing and we are truly pebbles in a vast ocean of information. When it comes to the Internet there are millions and millions of users logging on and off on a daily basis.
- Among those millions upon millions look at where we are. The fact is that about 30 - 40% of all users are aware of the things happening on their computers. The other simply either don't care or don't have the proper knowledge to recognize if their system is under attack and or being used by unauthorized users.
- On the Internet nothing is quite what it appears to be, because information is just transferred from one computer to another in a heartbeat. The uninformed will get hurt in many ways.

Scanned by CamScanner

Syllabus Topic : Security Goals

1.2 Security Goal

→ (MU - Dec. 15)

Q. 1.2.1 Define the goals of security. (Ref. sec. 1.2) **Dec. 15, 2 Marks**

- Main goal of information security is to protect data or information which is being transmitted and achieve the confidentiality, integrity and availability of the data.
- Following are the main goal of information security :

Fig. 1.2.1 : Goal of information security

→ 1. Confidentiality

Confidentially is most common aspect of information security. Confidentially is defined as the contents of a message are accessed only by intended person. Aim of confidentially is that only sender and his intended receiver should be able to access the contents of a message.

For example

- In military application information from one higher authority is sending to another higher authority. During this transmission process when third unknown person is trying to get this confidential information which is not desired. This type of information leakage caused because of interception of third person. Here sender and receiver are unable to access the contents of message which causes loss of message confidentially.
- The attack threatening the confidentiality is traffic analysis.
- Because of interception occurred between sender of receiver, sender is losing message confidentially.

Scanned by CamScanner

2. Integrity

- Principle of integrity states that contents of message should not be modified until it reaches to authorize person.
- In this case change in the information need to be done only by authorized person and through authorized mechanisms only. Integrity gives assurance that data received exactly as sent by an authorized entity. (No alteration, no modification, no deletion and no intersection etc.). The attack threatening integrity is modification and masquerade.
- The two important concepts :
  - **Data integrity :** Assures information is changed only in authorized manner.
  - **System integrity :** Assures that the system performs its intended function properly and free from unauthorized manipulation.

3. Availability

- Principle of availability states that resources must be available to authorized users at all the time as on when required. It assures that system works correctly and service is available to authorized users. The term resources may refer as confidential information, software and hardware components.

For example

- Information stored in bank, student's information stored in universities information stored on email accounts. All these information need to be available to all authorized users at any time as on when required. Imagine the situation if all above information is not available to authorized users.
- There is only one attack which threatening principle of availability called denial of service attack.
- These three concepts are termed as CIA triad and represent fundamental security objectives for data and information services as shown in Fig. 1.2.2.

Fig. 1.2.2 : CIA triad

Scanned by CamScanner

➜ **4. Data Authentication**

- Data authentication is important in many applications in networks.
- Data authentication allows the user or receiver to check whether that data really was sent by the actual sender or not.
- In the two - party communication this mechanism is achieved through symmetric cryptography.
- The sender and receiver share a secret key to calculate a Message Authentication Code (MAC) of all communication data.
- Receiver knows that the data is send by exact or actual sender, if and only if message arrives with a correct MAC.
- Data origin authentication is a property that a data has not been modified when it will transmit, this means data integrity.

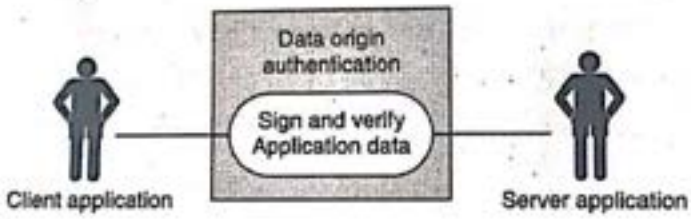


Fig. 1.2.3 : Data Authentication

Syllabus Topic : Services

1.3 Security Service

➔ **(MU - Dec. 16)**

Q. 1.3.1	Define authentication and non repudiation and show with examples how each one can be achieved. (Ref. sec. 1.3)	**Dec. 16, 5 Marks**
Q. 1.3.2	Write short note on security service. (Ref. sec. 1.3)	

- *X.800* is a service provided by a protocol layer of communicating open systems, to ensure the enough security of the system/ organization or of the data transfer.
- RFC 2828 defines security service as a communication service provided by a system to give protection to the system resources.

Scanned by CamScanner

- X.800 defines security services into following categories :

Fig. 1.3.1 : Categories of security services

1. Authentication

- It is assurance of parties that they are authentic user in the communication network.
- Authentication helps to identify the claimed identity of an entity, such as username password or any other important information such as encryption or decryption keys stolen during transmission between sender and receiver.
- It can be further classified as follows :

(i) **Peer entity authentication :** It checks that the entities connected in communication network are authentic and do not perform any attack like masquerade and replay of messages in network.

(ii) **Data origin authentication :** It checks that the data is authentic without any changes received by the receiver.

2. Authorization

Authorization service helps for checking whether the entity has the right to perform action requested. Authorization means providing authority or permission of accessing the system or privilege of accessing data, directories, files etc of the system.

Authorization is one of the most important security aspects. It provides identification of the user as authorized user. It is a kind of permission given by the network administrator for accessing the network.

Scanned by CamScanner

For example

Password used for sever login, ATM pin while withdrawing money and biometric machine used in an office for checking employee identification. An authorization principle helps whether server administrator, ATM card user and company employee are valid user or not.

→ 3. Access Control

Access Control is the ability to limit and control the access to the host systems. It prevents unauthorized use of a resource. The service used to prevent unauthorized use of a resources i.e. complete control over who can access to resources, under what conditions access can occur and what are different accessing methodology.

For example

It controls the access of resources which is to be made available only to legitimate user. Secondly it looks to the conditions of accessing the resource or network and what is allowed to be done to the resources.

→ 4. Non-Repudiation

Principle of non-repudiation states that if sender sends some information and later on he denied that he never sends that information called non-repudiation.

(i) **Non-repudiation, Origin :** Proof that the message was sent by the specified party.

(ii) **Non-repudiation, Destination :** Proof that the message was received by the specified party.

For example

Ramesh sends a request to bank about money transfer into Suresh account but later on Ramesh Denying the money transfer request to bank. Principle of non-repudiation does not allow such type of refuses of sender. Non-repudiation prevents either sender or receiver from denying a transmitted message.

→ 5. Auditing

- Auditing services helps to trace which user accessed what ? when ? and which way ?
- In general auditing does not provide protection but can be the tool for analysis of problems.

Scanned by CamScanner

- There are different security mechanism are used to provide security services and also helps to prevent all types of attacks.

➔ 6. Data Integrity

To assure that the message received is as sent with no duplications, insertions or modifications, delays or replays. The destructions of messages have also been recovered.

(i) **Connection integrity with recovery :** It provides integrity of the user data on a connection and detects modifications, insertions, deletions or replay if any with a recovery attempted.

(ii) **Checks Connection integrity without recovery :** It checks the integrity of the data in communication network and detects various attacks like modification, deletion replay of messages in network but without any recovery of same.

(iii) **Selective-Field Connection recovery :** It provides integrity of selected fields within a user data or a data block to determine whether any of the selected fields are modified, inserted, deleted or replayed.

(iv) **Connectionless integrity :** It determines and checks the modification of single data block and for preserving its integrity in connectionless network.

(v) **Selective-Field Connectionless recovery :** It determines and checks the modification of single data block of selected fields and for preserving its integrity in connectionless network.

➔ 7. Data Confidentiality

It is protection of data to be accessed by unauthorized user.

(i) **Connection confidentiality :** In case of a TCP connection set up between two systems, to protect user data that is transmitted over the connection.

(ii) **Connectionless confidentiality :** To protect data in a single data block.

(iii) **Selective-Field confidentiality :** To protect selective fields with a user data or a connection or a single data block.

(iv) **Traffic flow confidentiality :** To protect data that might be derived from observing the data flow.

Scanned by CamScanner

1.4 Security Mechanisms

→ (MU - Dec. 15, May 16)

Q. 1.4.1	Specify mechanisms to achieve each goal. (Ref. sec. 1.4)	Dec. 15, 3 Marks
Q. 1.4.2	List with examples the different mechanisms to achieve security. (Ref. sec. 1.4)	May 16, 5 Marks
Q. 1.4.3	What are eight security mechanisms to implement security ? (Ref. sec. 1.4)	

As discussed earlier ITU-T Recommendation X.800, Security Architecture for OSI defines systematic way to Defining the requirements for security. Characterizing the approaches to satisfying those requirements.

Following are the eight different security mechanisms.

1.4.1 Specific Security Mechanism / Attack Prevention

These mechanisms are incorporated into the appropriate protocol layer in order to provide some OSI security service. It is the security mechanism implemented to prevent against various types of attack before they can actually reach and affect the target systems.

1. **Encipherment**

   To use mathematical algorithms to transform data into a form that is not easily understandable. The transformation and subsequent recovery depends on the algorithm and the number of keys used.

2. **Digital Signature**

   The data is appended to, or a cryptographic transformation of, a data unit that allows the receiver of the message to prove the source and integrity of data unit against forgery.

3. **Access Control**

   Various mechanisms used to enforce access rights to the resources or it is the process of limiting the access to the resources of the Information System. Firewall is the best example of limiting the access control.

4. **Data Integrity**

   Various mechanisms used to assure the integrity of the data. Content should not modify before it reaches to intended person.

Scanned by CamScanner

5. Authentication Exchange

The mechanism used to ensure the identity of the entity by information exchange.

6. Traffic Padding

To insert bits into gaps in the data stream to frustrate traffic analysis attempt.

7. Routing Control

To allow some selected routes in network for routing or can change the route if any attack is detected in the network.

1.4.2 Pervasive Security Mechanisms / Attack Detection

These mechanisms are not specific to any of the OSI security service or protocol layer. This technique also called attack detection which is implemented to prevent, if attacker bypass the installed security measures to access the desired target/information. Attack detection technique notifies such incidents happens and takes the responsibility to report someone that something went wrong somewhere in the system. Such type of mechanisms used to inform the administrator or authorized user that something went wrong in the system now its job of administrator or authorized user to take action against detected attack.

1. Event detection

Detection of security related events. Intrusion Detection technique is the best example of event detection.

2. Security audit trail

Data collected and used to facilitate security audit.

3. Security recovery

It deals with the recovery action and management functions for data that is lost or disrupted in the network during communication.

1.4.3 Attack Avoidance

In these techniques data is sent over an insecure channel such as Internet in encrypted format and decrypted at receiver side using keys under assumptions that attacker may have access to the transmitted data.

Scanned by CamScanner

The encryption and decryption is performed on sending data by using well known cryptographic mechanisms such as :

- Private Key Cryptography.
- Public Key Cryptography.
- Hash Functions.

We will discuss all these cryptographic mechanisms in upcoming chapters.

1.5 Security Attack

→ (MU - May 18)

Q. 1.5.1 List and explain various types of attacks on encrypted message. (Ref. sec. 1.5)

May 18, 5 Marks

Q. 1.5.2 Categories the different attacks. Illustrate how passive attacks lead to business loss. (Ref. sec. 1.5)

- In computer and computer networks an attack is any attempt to alter, disable and destroy or gain unauthorized access of confidential information.
- The X.800 and RFC2828 (the communication service provide by a system to give specific kind of security to system resources) classify security attacks into two types namely :

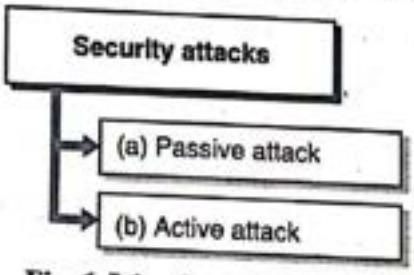


Fig. 1.5.1 : Security attacks

→ (a) Passive Attacks

- The information is only monitored during data transmission between two persons and doesn't involve any modification to the contents of original message is called passive attack. Passive attacks are furthered classified into two subcategories as shown in Fig. 1.5.2.
- A passive attack makes attempt to collect information from the system but does not modify or alter the system data or resources. Eavesdropping or monitoring of information is example of passive attacks.
- The goal of cryptanalyst is to gain the information that is being transmitted.

Scanned by CamScanner

- The two types of passive attacks are :

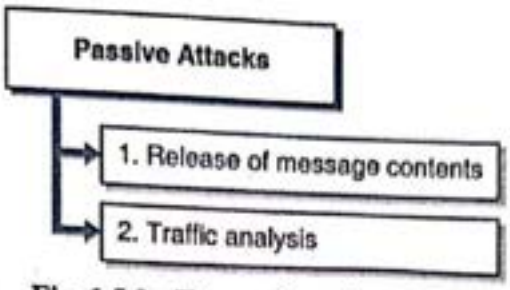


Fig. 1.5.2 : Types of passive attacks

1. Release of message contents

- Release of message contents attack is quite simple to understand. When we send a confidential email to our friend, our aim is that only intended person should access this mail. If this mail is accessed by unauthorized users then contents of message are released against somewhere else. Such type of attack is called release of message contents.
- There are different security mechanisms are available to prevent such type of attacks.

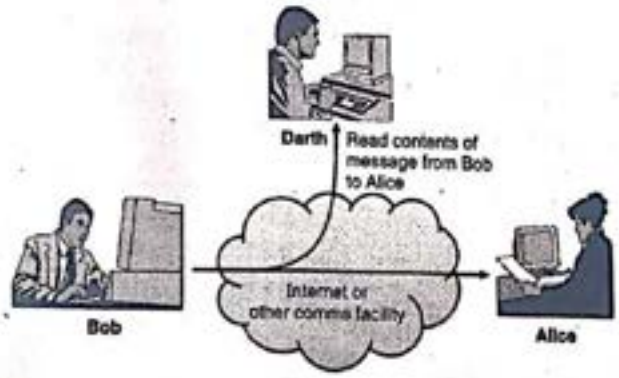


Fig. 1.5.3 : Release of message contents

For Example

- Telephonic conversation between two people, an electronic mail and a file may contents sensitive information sent/ transfer over insecure channel such as Internet. We would like to prevent third person from modification of these type of transmission as shown in Fig. 1.5.3.
- The main goal is to prevent the cryptanalyst from learning sensitive and confidential information through transmissions that take place through telephone calls or email messages or files transferred on network.

Scanned by CamScanner

→ **2. Traffic analysis**

- Suppose we mask the contents of the message using encryption.
- The opponent (here it is called third person) is able to capture the contents of the message but not extract the information from the message.
- The opponent might observe a pattern of messages to get the location, or any clue regarding the origin of message.
- Passive attacks are difficult to detect, because they do not involve modification of the information.
- The message sent and received is in normal fashion and neither sender nor receiver is aware that a third party has read message or observed pattern of messages as shown in Fig. 1.5.4. These attacks can be prevented by means of encryption.

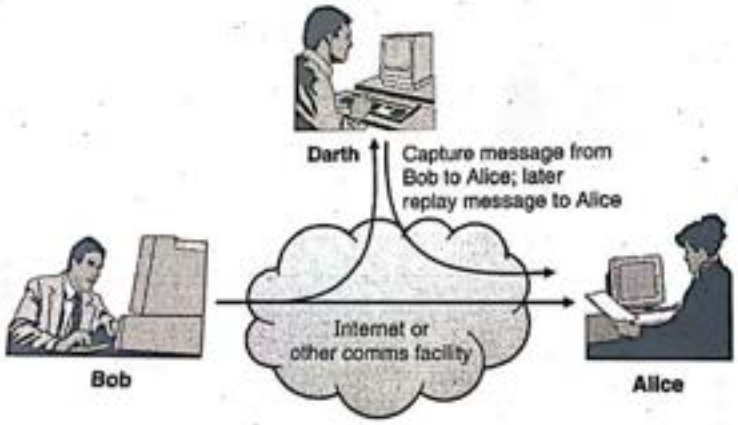


Fig. 1.5.4 : Traffic analysis

- The emphasis when dealing with passive attacks is on prevention rather than detection.

→ **(b) Active Attacks**

- Active attacks involve modification of a data stream or creation of a false stream of messages. Attacker aim in such type of attack is to corrupt or destroy the data as well as network itself. Active attacks means information is modified or gets altered during transmission between sender and receiver.
- Active attacks are divided into four categories as shown in Fig. 1.5.5.

Scanned by CamScanner

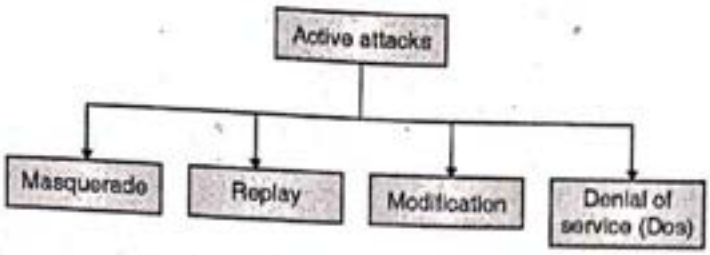


Fig. 1.5.5 : Types of Active attacks

1. Masquerade

- A masquerade takes place when an attacker pretends to be an authentic user. It is generally done to gain access to a system, or steal important data from system.
- It is generally done by stealing login id and password of authentic user to gain access to a secure network.
- Once attacker gain access, they get full access to the network for deletion or changing of data or network policies of organization as shown in Fig. 1.5.6.

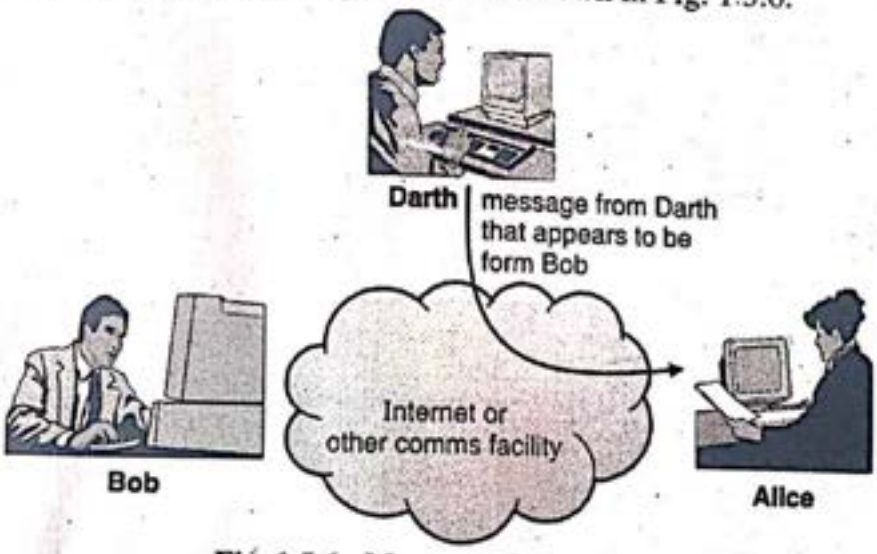


Fig. 1.5.6 : Masquerade attack

Such type of attack involves pretending the user from accessing authorized information.

2. Replay Attack (Rewrite)

- It is a network attack in which original data get modified and new malicious code added into valid data, during transmission.
- The newly generated malicious code retransmitted again and again to receiver called as replay attack (Reusing information).
- Replay attack involves passive capturing of data and retransmission of subsequent information in order to create unauthorized effect as shown in Fig. 1.5.7.

Scanned by CamScanner

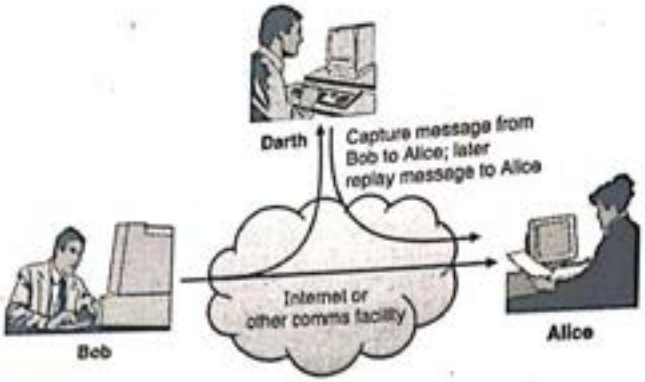


Fig. 1.5.7 : Replay attack

3. Modification of messages

- In modification, the original data that has been sent by the authentic user is been disrupt or modified by the attacker to make it non meaningful for the receiver. Usually the content sequence is been changed.
- Modification is also called replay attack. When contents of message modified after sender sends it but before it reaches to indented recipients, such type of attack is called modification of message as shown in Fig. 1.5.8.

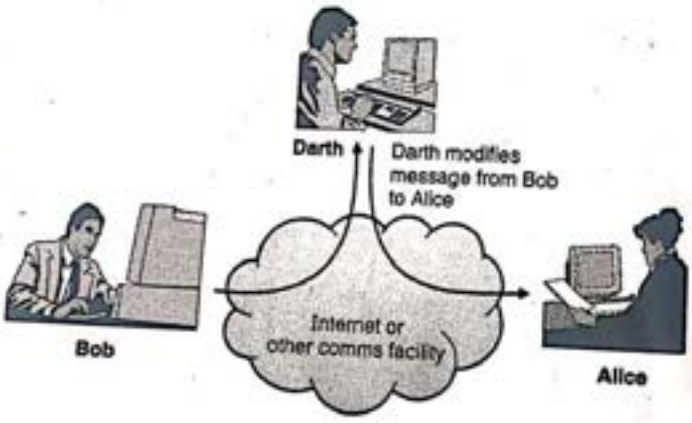


Fig. 1.5.8 : Modification of messages

For Example

- If Bob wants to transfer amount of Rs.1000/- into Alice account, during this transmission process an attacker called Darth capture the conversation and modified the actual amount of Rs.1000/- and sends just Rs. 100/- into Alice account.

Scanned by CamScanner

- The case happed here contents of message get altered during transmission process such type of attack called modification. In this case Integrity of original message is lost.

4. Denial of Service (DoS)

- Denial of service attack means making the network unavailable for the user to communicate securely.
- It is generally done by interrupting in the network connection between the users or making some services unavailable for user or disrupts the entire network by overloading with unwanted messages, so that network becomes slow and unavailable for users shown in Fig. 1.5.9.

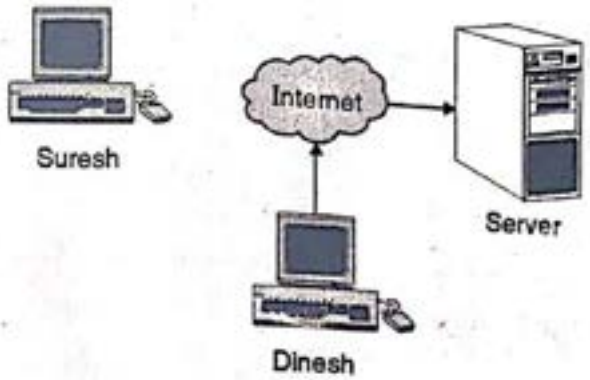


Fig. 1.5.9 : Denial of service

- Due to intentional action of Dinesh, Suresh is unable to access the data from server. Because of Dinesh intention he denied use or services of sever to Suresh; even if Suresh is authorized user. DoS attack attempt to shut down the network, computer services and deny the use of resources or services to authorized users.
- Once Dinesh got entire access of network or server he can do the following things :
  - Flood the entire network or server with traffic until shutdown occurs because of overload.
  - Block ongoing traffic which results in a loss of access to network resources to the authorized users. Different security policies like firewall, Intrusion detection system helps to protect such type of attacks.
  - Different security policies like firewall, Intrusion detection system helps to protect such type of attacks.

Scanned by CamScanner

1.5.1 Difference between Active Attack and Passive Attack

Sr. No.	Active Attack	Passive Attack
1.	Attacker needs to have physical control of the media or network.	Attacker merely needs to observe the communication in the media or network.
2.	It can be easily detected.	It cannot be easily detected.
3.	It affects the system.	It does not affect the system.
4.	It involves in modification of data.	It involves in monitoring of data.
5.	Types of active attack are : Masquerade, replay, denial of service, distributed denial of service.	Types of passive attack are : Release of message, traffic analysis.
6.	It does not check for loopholes or vulnerabilities.	It scans the ports and network in search for loopholes and vulnerabilities.
7.	It is difficult to prevent network from active attack.	Passive attack can be prevented.

Syllabus Topic : The OSI Security Architecture

1.6 The OSI Security Architecture

- ITU-T International Telecommunication Union Telecommunication Standardization Sector X.800 (It is a service provided by a protocol layer of communicating open systems, to ensure the enough security of the system/ organization or of the data transfer) security architecture for OSI (Open Systems Interconnections) defines a systematic approach for providing security at each layer or it is a standard which provide systematic approach of defining and providing security requirements.

Scanned by CamScanner

- The OSI security architecture focuses mainly on the following three concepts :
  - **Security Attack :** An action that may compromise the security of the information that is owned by an organization is called security attack.
  - **Security Techniques/ Mechanism :** Security mechanism is the process that designed how to detect, prevent or recover from a security attack.
  - **Categories of Security Service :** A processing or communicating service that enhances security of data processing and information transfers of an organization. The services are intended to counter security attacks. The security service can use of one or more security mechanisms to provide security. All these features of OSI security architecture has been discussed in detail in the following section.

Syllabus Topic : Network Security Model

1.7 Operational Model for Network Security

Consider a message/ data is to be transferred from sender to receiver or from one party to another across internet. During this data transmission process it is necessary to protect security aspects of this information from an opponent or attacker.

The technique used to provide security is as follows :

- The original message is encrypted with the help of a key, which scrambles the message so that it is not readable to any third party.
- An additional code can be attached to the encrypted data which is based on the contents of the message, which can be used to verify the identity of the sender.
- The message is now transmitting through an insecure channel such as Internet. The message when received at the receiver side is unscrambled either using same or different key to obtain the original message.
- A trusted third party (such as Virtual Private Network) is required for secure transmission.
- The trusted third party is responsible to distribute the private key and secret information to the sender and receiver while keeping it away from any opponent or attacker.
- The model for security is shown in Fig. 1.7.1.

Scanned by CamScanner

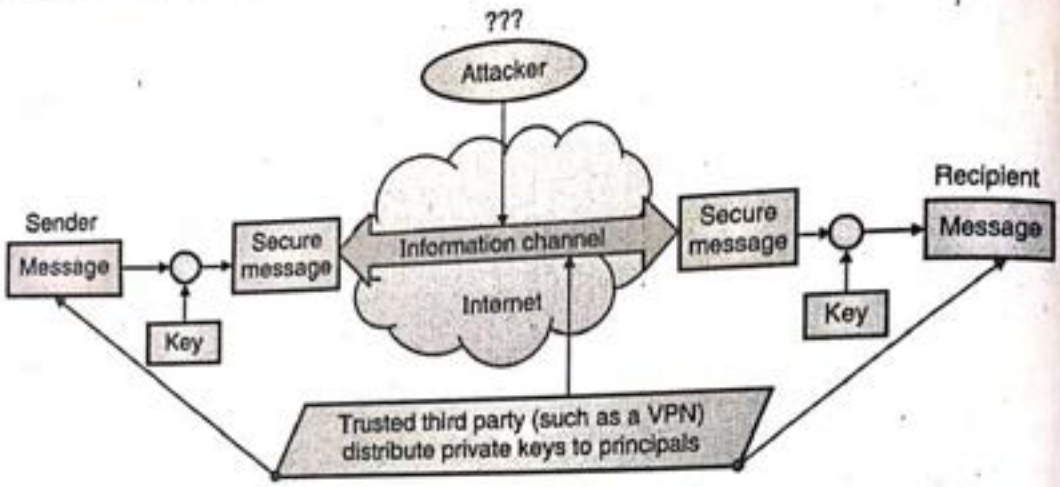


Fig. 1.7.1 : The model for network security

1.8 Basic Terminology in Network Security

As we are living in information age we need to keep watch on our data or we must aware about every aspect of our lives. Information/Data of every individuals having same value as like our personal assets. The main aim of encryption technique is to hide the information/data from unauthorized users, protection from unauthorized change and data should available to authorized users as an when required or needed. Before studying this whole subject one need to understand the following definitions called as basic terminology in network security.

1. **Plain text**

Plaintext is ordinary readable text before being encrypted into cipher text or after being decrypted OR the original message is known as plain text.

Example of plain text

The book we are writing uses normal English language so that everybody should understood the meaning and concept written in clear text form. Means the text written in this book is in readable form called as plaintext.

2. ***Ciphertext***

Cipher text is the output of encryption performs on plaintext using suitable techniques and algorithms. When plaintext get converted into non-readable form with the help of some modified scheme the resulting text is called as cipher text OR the coded or scrambled message is known as cipher text.

Scanned by CamScanner

Basically there are two process required to convert plaintext into cipher text and cipher text into plaintext, called encryption and decryption.

3. Encryption

The process of converting the plain text message to cipher text message is known as encryption.

Let, P be the plaintext, E is encryption and C is cipher text. It we perform encryption on P we get C as shown in Fig. 1.8.1.

Fig. 1.8.1 : Encryption

i.e. $C = E(P)$

4. Decryption

The process of restoring the plain text message from the cipher text message is known as decryption OR process of converting cipher text message into plaintext message is called as decryption.

Let, C be the cipher text, D is decryption, and P is plaintext. If we perform decryption on C it results original plain text P as shown in Fig. 1.8.2.

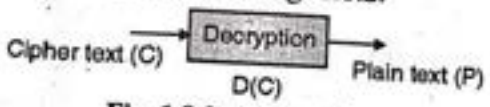


Fig. 1.8.2 : Decryption

i.e. $P = D(C)$

The process of encryption and decryption are controlled by cryptographic keys.

5. Cryptography

The word cryptography comes from the Greek words KPVIIT (hidden or secret) and $y_p \propto \phi_n$ (writing). **Cryptography** is the art as well as science of secret writing of information / message and makes them non-readable.

The many schemes used for encryption constitute the area of cryptography.

6. Cryptanalyst

The person who studies encryption and decryption methods and finds contents of hidden message called as cryptanalyst. The process of studying methods of breaking ciphertext message called as **cryptanalysis**.

Scanned by CamScanner

An attacker also called as cryptanalyst attempt to use all possible keys, algorithms and techniques to break ciphertext and obtain original plaintext message.

7. **Cryptology**

The area of cryptography and cryptanalysis together is known as cryptology.

8. **Key**

It is the secret information or number used in encryption and decryption algorithm which is known only to the sender and receiver.

1.8.1 Cryptanalysis/ Cryptographic Attacks

- Cryptographic attacks are designed to discover the loopholes in cryptographic algorithms; these attacks are designed to decrypt data without prior permission and without access to a key. This is job of Cryptanalysis to find the weakness into the algorithm used for encryption and decryption of data and then decipher the data. Before studying different attacks against Data Encryption Standard we must know different types of cryptographic attack methods.
- As mentioned above the process of trying to break any cipher text message to obtain the corresponding plain text message is called **cryptanalysis** and the person who is attempting cryptanalysis is called cryptanalyst.

Cryptographic Attack Methods

There are five cryptographic attack methods that include plaintext-based as well as cipher text based attacks.

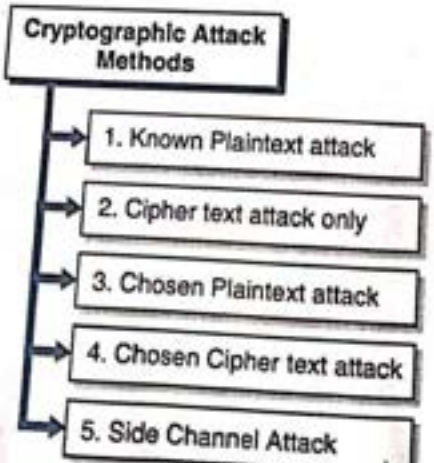


Fig. 1.8.3 : Cryptographic Attack Methods

Scanned by CamScanner

→ **1. Known Plaintext attack**

- In this type of attack cryptanalyst try to access plain text and its corresponding cipher text.
- He will find is there any correlation between plain text and cipher text produced; such type of attack is called known plain text attack. Example of such type of attack we will discuss in mono-alphabetic cipher technique.

→ **2. Ciphertext attack only**

In this type of attack cryptanalyst has only access to cipher text but doesn't have access to corresponding plain text such type of attack is called as Cipher text attack only. Such type of attack we will discuss in Caesar cipher technique.

→ **3. Chosen Plaintext attack**

- In this type of attack cryptanalyst can encrypt plain text of his own choice (guess) and later on find cipher text obtained from corresponding plain text such type of attack is called chosen plain text attack.
- This attack helps cryptanalyst to find the encryption key as well. This mapping helps attacker to find which plain text is encrypted. This is most common attack technique used against asymmetric key cryptography, where a cryptanalyst has access to a public key.

→ **4. Chosen cipher text attack**

- In this type of attack cryptanalyst chooses a cipher text and attempts to find a matching plaintext.
- Such type of attack generally associates with decryption process because cryptanalyst may get the temporary access to decryption process.

→ **5. Side channel attack**

- In this type of attack cryptanalyst always try to find out which technology used to designed cryptographic algorithms and which are the different software or hardware and keys used during encryption and decryption process.
- Cryptanalyst may find the additional information like CPU usage, time taken to perform particular task, voltage used and so on. Such type of attack is called side channel attack.

Scanned by CamScanner

Threats and Vulnerability

- Threat is a potential cause of an incident that may result in harm to a system or organization.
- Vulnerability is a weakness of an asset (resource) or a group of assets that can be exploited by one or more threats.
- Risk is potential for loss, damage, or destruction of an asset as a result of a threat exploiting a vulnerability.
- **Example :** In a system that allows weak passwords,
  - **Vulnerability :** Password is vulnerable for dictionary or exhaustive key attacks
  - **Threat :** An intruder can exploit the password weakness to break into the system
  - **Risk :** The resources within the system are prone for illegal access/ modify/ damage by the intruder.
- Threat agent-entities that would knowingly seek to manifest a threat.

Difference between Security and Privacy

(1) Definition of Security and Privacy

- While both are interlinked terms that are often used in conjunction with each other. While one cannot exist without the other, they are often misappropriated.
- Security is the state of personal freedom or being free from potential threats, whereas privacy refers to the state of being free from unwanted attention.

(2) Objectives of Security and Privacy

- The three main goals of security are confidentiality, integrity and availability.
- Security means safeguarding your information assets and confidential data from unauthorized access. It affects both information security and cyber security.
- All security protocols address at least one of the three goals. Privacy, on the other hand, refers to the rights of individual and organizations with respect to personal information.

(3) Programs for Security and Privacy

- A security program refers to a set of protocols and regulations set in place to protect all the confidential information assets and resources that an organization collects and owns.

- It focuses on
- Privacy prog
  such as log i

(4) Principles of Sec

- The three co
  integrity of i
- Privacy defi
  information.
  security depe

1.9 Encryption

Before discussing
types of cryptography
and asymmetric key c

1.9.1 Symmetric

- Symmetric key c
- In symmetric key
- As shown in Fig
  cipher text is tra
  side the cipher te
- Note that encryp
  called as Data E

Scanned by CamScanner

- It focuses on the data and information rather than personal information of individuals.
- Privacy program, on the other hand, focuses on protecting only personal information such as log in credentials, passwords, etc.

(4) Principles of Security and Privacy

- The three core principles of security include protecting confidentiality, preserving integrity of information assets, and promoting availability of data and information.
- Privacy defines the rights of individual and organizations with respect to personal information. To some extent, privacy can be achieved with security initiatives and security depends on privacy of credentials and access to data.

Syllabus Topic : Classical Encryption Techniques

1.9 Encryption Methods

Before discussing concept of encryption methods we must know which are the different types of cryptography ? There are two types of cryptography i.e. symmetric key cryptography and asymmetric key cryptography as shown in Fig. 1.9.1.

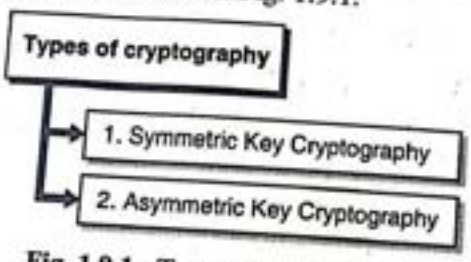


Fig. 1.9.1 : Types of cryptography

Syllabus Topic : Symmetric Cipher Model

1.9.1 Symmetric Key Cryptography

- Symmetric key cryptography also called as secret key cryptography.
- In symmetric key cryptography a single key is used for encryption as well as decryption.
- As shown in Fig. 1.9.2 sender encrypt plain text using shared secret key and the resultant cipher text is transmitted through communication medium such as Internet, at the receiver side the cipher text is decrypted using same decryption key to obtain original plain text.
- Note that encryption and decryption process uses well known symmetric key algorithm called as Data Encryption Standard (DES).

Scanned by CamScanner

Mathematically it is represented as P = D (K, E(P)).

Where P = Plain Text, E (P) = Encryption of plain text, D (K, E(P)) = Decryption of Pla text using shared key K.

- **For Example :** Stream and block cipher, Data Encryption Standard (DES), Advan Encryption Standard (AES) and BLOFISH.

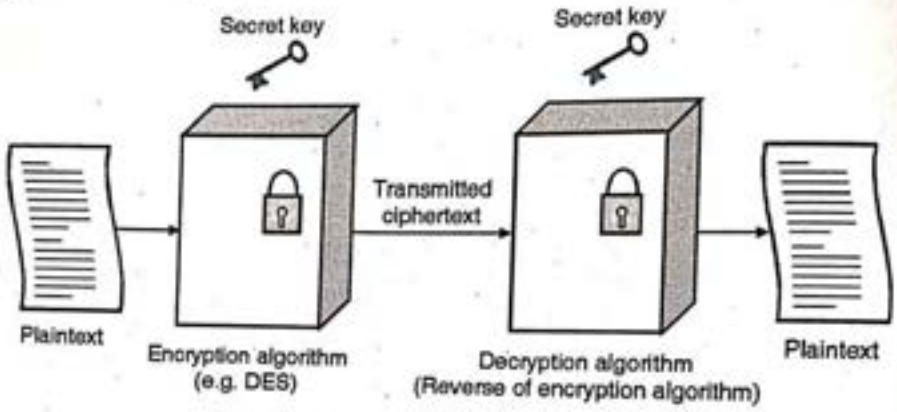


Fig. 1.9.2 : Symmetric Cipher Model

- Here the possibility is that if an attacker/opponent got cipher text ?? He/she may app different permutations and combinations to decrypt and obtain the original plain text. He the main aim of cryptography is came into picture. Always sender has to think on apply different encoding technique on plain text message and convert it into cipher text mess so that attacker cannot read the actual plain text easily.
- Symmetric cipher model convert the plain text message into cipher text by usi following techniques.

Advantages of Symmetric key cryptography

- Symmetric key is faster than asymmetric key cryptography.
- Because of single key data cannot decrypt easily at receiver side even if it is intercepte by attacker.
- As the same key is used for encryption and decryption receiver must have the senders ke he cannot decrypt it without sender permission.
- Symmetric key achieve the authentication principle because it checks receivers identity.
- DES and AES techniques are implemented using symmetric key cryptography.
- System resources are less utilized in symmetric key cryptography.

Scanned by CamScanner

Disadvantages of Symmetric key cryptography

- Once the key is stolen while transmitting data between sender and receiver it is very easy to decrypt the message as same key is used for encryption and decryption.
- In symmetric key cryptography, key is transmitted first and then message is transfer to the receiver. If attacker intercepts the communication between sender and receiver then he can decrypt the message before it reaches to intended recipients.

1.9.2 Asymmetric Key Cryptography

- Asymmetric key cryptography is also called as public key cryptography.
- In asymmetric key cryptography two keys are used, one for encryption and other for decryption.
- As mentioned asymmetric key cryptography involves use of two keys one is public key that may know to everyone and can be used to encrypt messages, and verify signatures. Other is private key known only to the receiver of the message or verifier, used to decrypt messages, and sign (create) signatures.
- It is also called as asymmetric key cryptography because one key is used for encryption only its corresponding key must be used for decryption. No other key can decrypt the message.
- The sender and receiver can encrypt messages using encryption key (public) or verify signatures, he cannot decrypt messages or create signatures because he required decryption key (private) which is known only to the receiver of the message. Public key cryptosystem /asymmetric key cryptography as shown in Fig. 1.9.3.

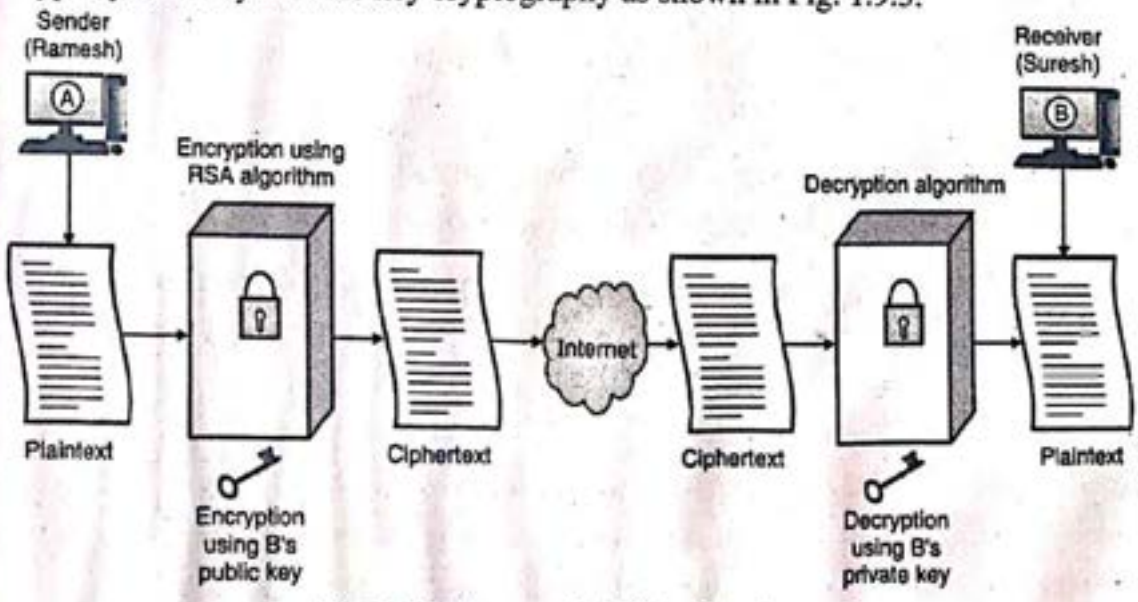


Fig. 1.9.3 : Asymmetric Key Cryptography

Scanned by CamScanner

Mathematically it is represented as P = D(Kd, E (Ke,P))

Where P = Plain Text, E (P) = Encryption of plain text, D = Decryption, Ke = Encryption key, Kd = Decryption Key.

- **For example**, sender Ramesh wants to communicate with the receiver Suresh then they must have each one of this i.e. private key and public key then and then communication will be successful.
- Table 1.9.1 shows the possible pair of keys that Ramesh and Suresh must know while communicating with each other.

Table 1.9.1 : Pair of private and public keys

Key Details	Ramesh (A) should know	Suresh (B) should know
Ramesh Private Key (A)	Yes A must know	Not known to B
Ramesh Public Key (A)	Yes A must know	Yes it is known to Suresh also
Suresh private key (B)	Not known to Ramesh (A)	Yes Suresh (B) must know
Suresh public key (B)	Yes known to Ramesh (A)	Yes Suresh (B) also known it

Following are the possible cases of public key cryptography as per the table mentioned above.

Case 1

1. When Ramesh wants to send a message to Suresh. Ramesh can encrypt the message using Suresh public key. This is possible because Ramesh and Suresh knows the public key.
2. Ramesh can send this message to Suresh (Keep in mind this it is encrypted using Suresh public key).
3. Suresh can decrypt the Ramesh message by using Suresh own private key. Because only Suresh knows his private key Ramesh is not aware about Suresh private key.
4. It is important to note that the message only decrypted using Suresh private key and nothing else.

Case 2

- If Suresh wants to send the message to Ramesh, then reverse the above case 1. Suresh can encrypt the message only with Ramesh public key. The reason only Ramesh can decrypt the message to obtain its original plain text format using his private key.

Scanned by CamScanner

- Public key cryptography achieves authentication (authentication helps to identify the claimed identity of an entity, such as username password or any other important information such as encryption or decryption keys stolen during transmission between sender and receiver) and non-repudiation (it prevents either sender or receiver from denying a transmitted message).
- Principles of public key cryptography also include mathematical background to understand the use of key pairs in algorithms like Rivest Shamir Adlman (RSA) and Diffie Hellman Algorithm.
- **For Example :** Rivest Shamir Adlman (RSA) and Diffie Hellman key exchange algorithm.

Advantages of Asymmetric key cryptography

- In Asymmetric key cryptography, key cannot be distribute among sender and receiver as both have their own key, so there is no problem of key distribution while transmitting the data over insecure channel.
- The main advantage of asymmetric key cryptography is that two separate keys are used for used encryption and decryption; even if encryption key is stolen by attacker he/ she cannot decrypt the message as decryption key is only available with receiver only.
- RSA algorithm and Diffie Hellman key exchange are implemented using asymmetric key cryptography.
- Easy to use for user and scalable does not require much administrative work.

Disadvantages of Asymmetric key cryptography

- Because of different key used between sender and receiver require more time to get the transmission done as compare to symmetric key cryptography. (Slower that symmetric key cryptography very few asymmetric encryption methods achieve the fast transmission of data).
- Asymmetric key cryptography utilizes more resource as compare to symmetric key cryptography.

Scanned by CamScanner

1.9.3 Difference between Symmetric and Asymmetric Key Cryptography

Sr. No.	Symmetric Key Cryptography	Asymmetric Key Cryptography
1.	In Symmetric key cryptography single or same key is used for encryption and decryption.	In asymmetric key cryptography two keys are used, one is for encryption and other is for decryption.
2.	Symmetric key cryptography is also called as secret key cryptography or private key cryptography.	Asymmetric key cryptography is also called as public key cryptography or conventional cryptographic system.
3.	Mathematically it is represented as P = D (K, E(P)). Where K is encryption and decryption key. P = plain text, D = Decryption E(P) = Encryption of plain text	Mathematically it represented as P = D(Kd, E (Ke,P)), Where Ke and Kd are encryption and decryption key. D = Decryption E(Ke, P) = Encryption of plain text using private key Ke.
4.	Symmetric key is faster than asymmetric key cryptography.	Because of two different key used asymmetric key is slower than asymmetric key cryptography.
5.	For encryption of large message symmetric key cryptography still play an important role.	In asymmetric key cryptography plain text and cipher text treated as integer numbers.
6.	Symmetric key cryptography utilizes less resource as compare to asymmetric key cryptography.	Asymmetric key cryptography utilizes more resource as compare to symmetric key cryptography.
7.	For Example : AES, DES and BLOWFISH	For Example : RSA, Diffie Hellman Key exchange algorithm.

Scanned by CamScanner

1.10 Block Cipher Principles

- Basically cryptographic algorithm is used for transformation of plaintext into ciphertext. Stream cipher and Block cipher are main method of encrypting text using key and algorithm.
- Block cipher principles are explained on the basis of two different algorithm types as shown in Fig. 1.10.1.

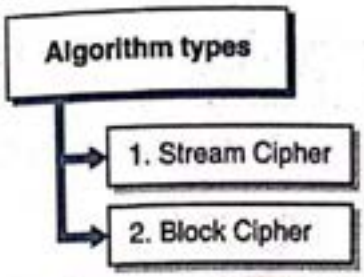


Fig. 1.10.1 : Algorithm types / Block Cipher Principles

1.10.1 Stream Cipher

- In stream cipher keys and algorithms are applied to each binary digit in a data stream, one bit at a time, rather than encrypting block of data (a stream cipher operates on plaintext accepting only one bit at a time).
- One time pad is the best example of stream cipher in which each bit of plaintext message XOR with each bit of key to obtain cipher text message. It is a symmetric cipher operates on varying time transformation individually on each bit. This is shown in Fig. 1.10.2.

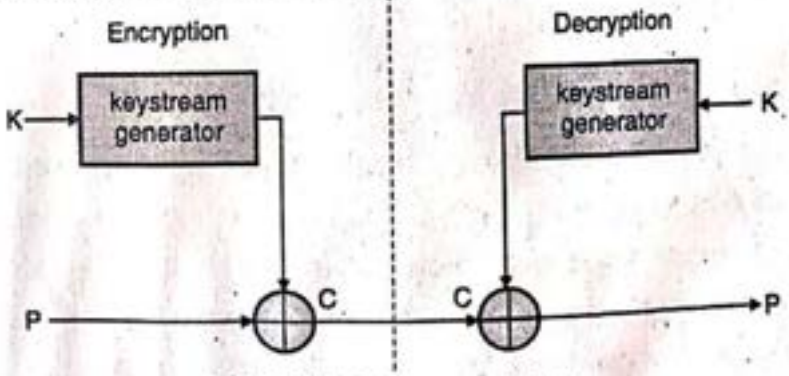


Fig. 1.10.2 : Stream Cipher

- There is a keystream generator which outputs a stream of bits : $k_1, k_2, k_3, \ldots, k_i$ XOR with a stream of plaintext bits, $p_1, p_2, p_3, \ldots, p_i$, to produce the stream of ciphertext bits $c_1, c_2, c_3, \ldots, c_i$. Broadly it can represented as shown below

$$c_i = p_i \oplus k_i \qquad \ldots(1.10.1)$$

Scanned by CamScanner

- During decryption, the cipher text bits are XOR with a same key stream to recover the plaintext bits.

$$p_i = c_i \oplus k_i \quad \text{...(1.10.2)}$$

- The stream cipher security depends on the simple XOR and one time pad. If the key stream output is random that, it will take harder time for a cryptanalyst to break it. However if it will keep on repeating same stream bits then it can result an attack on the cryptosystem.

- **For example :** Suppose we have plaintext as *pay 100* in ASCII (i.e. text format). When it is converted to binary values let us take that it is translating as 010111001 (hypothetically) also applying XOR logic in encryption algorithm. We can see effect as,

In text format	In binary format	
Pay 100	010111001	Plaintext
+	100101011	XOR operation with the key
ZTU91^%D	11010010	Cipher text

XOR Logic is shown in table below

Input 1	Input 2	Output
0	0	0
0	1	1
1	0	1
1	1	0

Rather than divide bit stream into discrete blocks stream cipher convert plaintext message into ciphertext message by encrypting one bit at a time. At receiver side use same key and XOR logic to extract plaintext message. A known plaintext attack can succeed against a stream cipher because if two messages are encrypted with the same keystream, XORing the two ciphertexts will remove the keys and result in the XOR of the plaintexts.

$$C1 \oplus C2 = (p1 \oplus k_i) \oplus (p2 \oplus k_i)$$

$$= (p1 \oplus p2) \oplus (k_i \oplus k_i)$$

$$= (p1 \oplus p2) \oplus 00\ldots0$$

$$C1 \oplus C2 = (p1 \oplus p2)$$

Scanned by CamScanner

1.10.2 Block Cipher

- Block Cipher break plain text message into fixed blocks and encrypt each block with some key size (fixed). Divide each plaintext message into block of 64, 128, 256 bits and apply common key size 40, 56, 64, 80, 128, 168, 192 and 256 bits which generate cipher text message same as size of plaintext block or as per size of plaintext message.
- Fig. 1.10.3 shows encryption of 64-bit plaintext using block cipher with 56-bit key.

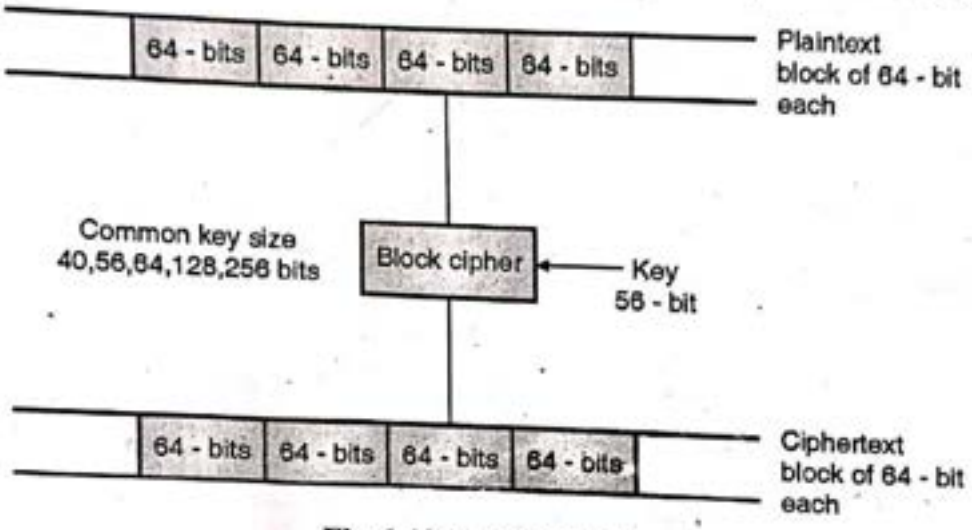


Fig. 1.10.3 : Block cipher

- Block cipher is main method of encrypting text in which keys and algorithm are applied to block of data rather that individual bits like stream cipher. Data Encryption Standard (DES) is the best example of block cipher in which each block of 64-bit get encrypted using 56-bit key and cipher text of 64-bit get generated.
- At receiver side decrypt message with same key to generate plaintext.
- Like in stream cipher, block cipher also uses the concept of key generator. Block cipher are used in **Chaining mode,** this is because for repeating text pattern, the same cipher block will be generated which can give clue to cryptanalyst regarding what is the original plaintext hence chaining mode is used for block ciphers. We will discuss concept of chaining mode in block cipher modes of operation.
- As in chaining method, previous block is mixed with current block to avoid repeats in patterns. Block cipher is little time consuming but secure than stream cipher so generally used in computer based cryptographic algorithms. Stream cipher is faster than block cipher.

Scanned by CamScanner

1.10.3 Differentiate between Stream and Block Cipher

→ (MU - Dec. 16)

Q. 1.10.1 Compare and contrast - Block and stream ciphers. (Ref. sec. 1.10.3) **Dec. 16. 5 Marks**

Sr. No.	Stream Cipher	Block Cipher
1.	In stream cipher keys and algorithms are applied to each binary digit in a data stream, one bit at a time, rather than encrypting block of data.	Block cipher is main method of encrypting text in which keys and algorithm are applied to block of data rather that individual bits like stream cipher.
2.	Stream cipher is less time consuming.	Block cipher is more time consuming.
3.	Because of one bit encrypting at a time, stream cipher is faster than block cipher.	As block of data is encrypting at a time block cipher is slower than stream cipher.
4.	Stream Cipher doesn't used in chaining modes of operation.	Block is used in chaining modes of operation.
5.	Hardware implementation is easy using stream cipher.	Software implementation is easy using block cipher.
6.	One Time Pad is the best example of stream cipher.	Data Encryption Standard (DES) is the best example of block cipher.

1.10.4 Confusion and Diffusion

Claude Shannon introduced two properties of operation of secure cipher.

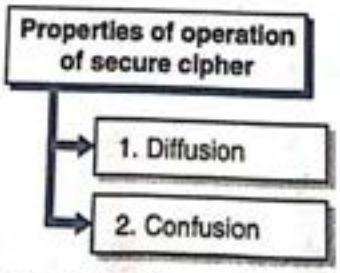


Fig. 1.10.4 : Properties of operation of secure cipher

Scanned by CamScanner

➔ 1. Diffusion

It means any of the character in plaintext is changed, then simultaneously several characters of the cipher text should also be changed. Similarly if the character of cipher text is changed then simultaneously several characters of plaintext should be changed. It is a classical transposition cipher.

"Diffusion" = Transposition or Permutation

abcd → dacb

Data Encryption Standard is the best example of diffusion.

➔ 2. Confusion

Each character of cipher text depends on different part of the key. In confusion the key does not directly related to cipher text. It is a classical substitution cipher.

"Confusion" = Substitution

a → b Caesar cipher

For Example

Suppose we have a Hill cipher with an matrix $n \times n$, and suppose we have a plaintext-ciphertext pair of length n^2 with which we are able to solve for the encryption matrix. If we change one character of the ciphertext, one column of the matrix can change completely.

Of course, it would be more desirable to have the entire key change. When a situation like that happens, the cryptanalyst would probably need to solve for the entire key simultaneously, rather than piece by piece.

1.10.4(A) Difference between Confusion and Diffusion

Sr. No.	Confusion	Diffusion
1.	Confusion obscures the relationship between the plaintext and ciphertext.	Diffusion spreads the plaintext statistics through the ciphertext.
2.	A one-time pad relies entirely on confusion while a simple substitution cipher is another (weak) example of a confusion-only cryptosystem.	A double transposition is the classic example of a diffusion-only cryptosystem.

Scanned by CamScanner

Sr. No.	Confusion	Diffusion
3.	Confusion alone is, apparently, "enough", since the one-time pad is provably secure.	Diffusion alone is, perhaps, not enough, at least using relatively small blocks. A stream cipher is simply a weaker version of a one-time pad.
4.	The codebook aspects of such systems provide confusion analogous to though on a much grander scale a simple substitution.	Well-designed block ciphers spread any local statistics throughout the block, thus employing the principle of diffusion.

Syllabus Topic : Mono-alphabetic and Poly-alphabetic Substitution Techniques

1.11 Substitution Cipher Techniques

→ **(MU - Dec. 15)**

Q. 1.11.1 Define the following with example - Substitution cipher. (Ref. sec. 1.11) **Dec. 15, 2 Marks**

A substitution is a technique in which each letter or bit of the plaintext is substituted or replaced by some other letter, number or symbol to produce ciphertext. Substitution means replacing an alphabet of plaintext with an alphabet of ciphertext. Substitution technique also called confusion. The best example of substitution cipher is Caesar cipher invented by Julius Caesar.

Substitution Cipher techniques are as follows :

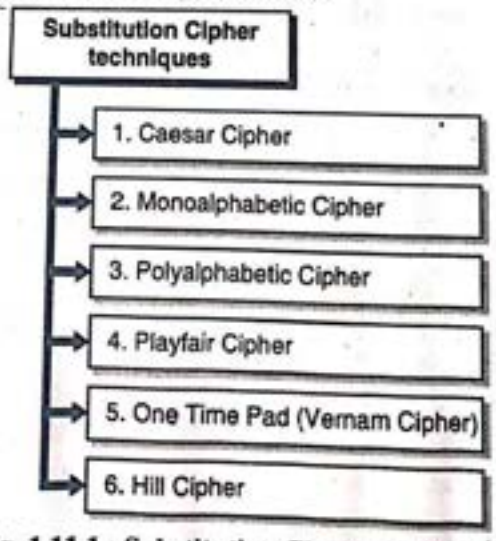


Fig. 1.11.1 : Substitution Cipher techniques

Scanned by CamScanner

1.11.1 Caesar Cipher

- Julius Caesar introduced the easiest and the simplest use of substitution cipher.
- In Caesar cipher technique each letter is replaced by the letter /alphabet which is three places next to that letter which is to be substituted. Or In Caesar cipher technique, each alphabet of a plaintext is replaced with another alphabet but three places down the line as mentioned in table below.

For example

- **Plaintext :** Sun rises in the East
- **Ciphertext :** VXQULVHVLQWKHHDVW

Following is the list of possible combination showing the letters 3 places down of each alphabet :

Plaintext	a	b	c	d	e	f	g	h	i	j	k	L	m	n	o	p	q	r	s	t	u	v	w	x	y	z		
Ciphertext	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C		

The corresponding number equivalent to each alphabet is given below :

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	X	y	z			
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25			

Mathematically the Caesar cipher algorithm can be expressed as

$$C = E(3,P) = (P + 3) \bmod 26$$

$$P = D(3,C) = (C - 3) \bmod 26$$

Where C = Ciphertext/ or alphabet

P = Plaintext/ alphabet

E = Encryption

D = Decryption

Mod 26 because in English there are total 26 alphabets.

Scanned by CamScanner

It is very easy to break ciphertext obtained from plaintext message with the help of Brute-Force attack because the attacker will be having only 25 possible keys to decrypt the ciphertext.

Further some more characteristics which lead to easy brute force attacks are :

1. The encryption and decryption algorithms are known.
2. Only 25 possible keys.
3. And plaintext language is easy to recognize with few repetition of alphabet having same ciphertext letter.

☞ **Brute-Force-attack**

A brute - force - attack means trying every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. We can create more complex cipher text from given plaintext with the help of another substitution technique called monoalphabetic cipher to prevent brute-force-attack.

1.11.2 Monoalphabetic Cipher

- In Caesar cipher the attacker can easily guess the plaintext as it is easily recognizable. In Monoalphabetic cipher substitutes one letter of the alphabet with any random letter from the alphabet.
- It is not necessary that if A is substituted with B then compulsorily B has to be substituted with C. It can be replaced with any other letter of the alphabet. The only weakness in this algorithm is that if more repetition occurs then attacker can easily guess the plaintext.
- This random substitution is just done to have uniqueness.
- In this the substitution of characters are random permutation of the 26 letters of the alphabet.

For example

- Following is the substitution that we are taking :

Plaintext	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z		
Cipher-text	e	o	u	n	a	f	p	v	b	m	w	l	c	q	x	d	g	k	s	y	z	i	t	r	j	h		

Scanned by CamScanner

Plaintext : East or West

Ciphertext : aesy xk taay

- Ciphertext obtain with this technique yields completely different text as compare to Caesar cipher. In this method, each letter provides multiple substitutes for a single letter.

For example,

- A can be replaced by : d, j, r, y
- B can be replaced by : h, u, m, p etc.

Such type of large key space makes this cipher technique extremely difficult to break by brute-force attack. But this can make the cryptanalysis attacker straight forward to guess the pattern.

1.11.3 Polyalphabetic Cipher

➔ **(MU - Dec. 15)**

Q. 1.11.2 Define the following with example - Poly-alphabetic cipher. (Ref. sec. 1.11.3) **Dec. 15, 2 Marks**

As in monoalphabetic cipher we use only one fixed alphabet, but draw back in monoalphabetic is these are fairly easy to break. So to make it harder to break, the concept of polyalphabetic cipher arises which uses **more than one** alphabet and switching between them systematically.

1.11.3(A) Procedure of Polyalphabetic Cipher

1. Pick a keyword (for our example, the keyword will be "MEC" as shown in Example 1).
2. Write your keyword across the top of the text you want to encipher, repeating it as many times as necessary.
3. For each plain text letter, look at the letter of the keyword above it (if it was 'M', then you would go to the row that starts with an 'M'), and find that row in the Vigenere table given below.

Scanned by CamScanner

ROW																										
0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
2	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
3	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
4	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
6	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
7	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
8	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
9	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
10	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
11	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
12	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
13	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
14	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
15	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
16	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
17	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
18	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
19	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
20	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
21	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
22	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
23	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
24	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
25	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

4. Then find the column of your plaintext letter (for example, 'w', so the twenty-th column).

5. Finally, trace down that column until you reach the row you found before and write do the letter in the cell where they intersect (in this case, you find an 'I' there). Repeat t same till you will convert all plain text messages into ciphertext.

Example 1

Keyword	MECMECMECMECMECMECMECM
Plaintext	weneedmoresuppliesfast
Ciphertext	IIPQIFYSTQWWBTNUIUREUF

Thus, the plain text message "We need more supplies fast!" comes out :

IIPQIFYSTQWWBTNUIUREUF

Scanned by CamScanner

1.11.3(B) Difference between Polyalphabetic and Monoalphabetic

→ (MU - Dec. 17)

Q. 1.11.3 With the help of examples compare and contrast mono-alphabetic ciphers and poly-alphabetic ciphers? (Ref. sec. 1.11.3(B)) **Dec. 17, 5 Marks**

Sr. No.	Polyalphabetic Cipher	Monoalphabetic Cipher
1.	Polyalphabetic cipher is more secure and hard to be broken.	Monoalphabetic cipher are not very secure and can be easily broken.
2.	More than one alphabet is used for substitution.	One fixed single alphabet is used for substitution.
3.	In a polyalphabetic cipher, the substitution rule changes continuously from letter to letter according to the elements of the encryption key.	In monoalphabetic, the same substitution rule is used for each substitution.
4.	In polyalphabetic for particular alphabet, different substitution can be done using Vignere table.	In monoalphabetic, for a particular alphabet, only one substitution can be used.
5.	Polyalphabetic cipher includes, Playfair cipher, Vigenere, Hill cipher, one-time pad etc.	Monoalphabetic cipher includes additive, multiplicative, and monoalphabetic substitution cipher.

Syllabus Topic : Playfair Cipher

1.11.4 Playfair Cipher

Q. 1.11.4 Write a short note on Playfair cipher. (Ref. sec. 1.11.4)

- It was invented by Charles Wheatstone in 1854 but known by name Playfair because Lord Playfair made this technique popular. It was used by Britisher in World War 1.
- It is multiple letter encryption technique, which uses 5 × 5 Matrix table to store the letters of the phrase given for encryption which latter on becomes key for encryption and decryption.

For example : Keyword is FAIR EXAMPLE.

Scanned by CamScanner

- In the first step all letters are to be filled in that matrix from left to right, the letters which are already been placed is not be placed again in that matrix.
- After filling up of the given letter, fill rest of the space in the matrix with the remaining letters alphabetically with no repetitions.
- The letters I and J will be considered as one letter. So If I is already placed then no need to place J in rest of the matrix.
- The letters which are already written/ mentioned need not to be placing that letter in given matrix.
- For in given example A and E are already mentioned in matrix so it is not mandatory to write that letter again in the given matrix.

Table 1.11.1

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

Encryption using Table 1.11.1 keyword matrix

1. The plaintext received is to be broken in pair of two letters.
2. For example CYBER can be broken into CY BE R(X).
3. If both letters are same or only one letter is left then put X with that alphabet.
4. If both pair alphabet appears in same row replace the letter with the immediate right alphabet (wrapping around to the left side of the row if a letter in the original pair was on the right side of the row).
5. If both letters appear in same column replace it with alphabet immediate below to that letter (wrapping around to the top side of the column if a letter in the original pair was on the bottom side of the column).
6. If none of the condition explained above meet, then replace them with the letters on the same row respectively but at the other pair of corners of the rectangle defined by the original pair.

Scanned by CamScanner

For Example : The plaintext CYBER can be encrypted as : CY BE R(X)

1. For pair CY we check that CY does not occur in same row or column so we see step 6.

   The pair CY forms a rectangle, replace it with HU. If pair forms a rectangle, pick same row letter but opposite corners.

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

   So CY will encrypt as H U.

2. Checking for BE both are in same column so replace it with immediate next in that column.

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

   B E will be encrypted as K B (Below to E is B).

   Checking for R(X)

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

   The pair RX forms a rectangle, replace it with FL. If pair forms a rectangle, pick same row letter but opposite corners.

   R(X) will be encrypted as FL.

Scanned by CamScanner

Solved Examples on Play fair cipher

Ex. 1.11.1 **MU - Dec. 15, 5 Marks**

Encrypt "The key is hidden under the door" using Playfair cipher with keyword "domestic".

Soln. :

Keyword - domestic

- Keyword is domestic.
- In the first step all letters are to be filled in 5×5 matrix from left to right, the letters which are already been placed is not be placed again in that matrix.
- After filling up of the given letter, fill rest of the space in the matrix with the remaining letters alphabetically with no repetitions.
- The letters I and J will be considered as one letter. So If I is already placed then no need to place J in rest of the matrix.

d	o	m	e	s
t	i	c	a	b
f	g	h	k	l
n	p	q	r	u
v	w	x	y	z

- **By using Playfair Cipher** (Use following steps to encrypt given word or message) we want to encrypt the plain text message "The key is hidden under the door" using keyword domestic.

1. The plaintext received is to be broken in pair of two letters, if duplicate letter put x
2. Th, ek, ey, is, hi, dx, de, nu, nd, er, th, ed, ox, or
3. If both letters are same or only one letter is left then put X with that alphabet.
4. If both pair alphabet appears in same row replace the letter with the immediate right alphabet (wrapping around to the left side of the row if a letter in the original pair was on the right side of the row).
5. If both letters appear in same column replace it with alphabet immediate below to that letter (wrapping around to the top side of the column if a letter in the original pair was on the bottom side of the column).

Scanned by CamScanner

6. If none of the condition explained above meet, then replace them with the letters on the same row respectively but at the other pair of corners of the rectangle defined by the original pair.

7. Refer above matrix for the same.

th → Step 6 → cf

ek → Step 5 → ar

ey → Step 5 → ae

is → Step 6→ bo

hi → Step 6 → gc

dx → Step 6 → mv

de → Step 4 → os

nu → Step 4 → pn

nd → Step 5 → vt

er → Step 5 → ay

th → Step 6 → cf

ed → Step 4 → so

ox → Step 6 → mw

or → Step 6 → ep

The plain text message **"The key is hidden under the door"** encrypted as :

cf, ar, ae, bo, gc, mv, os, pn, vt, ay, cf, so, mw, ep.

Ex. 1.11.2

Is playfair cipher monoalphabetic cipher ? Justify. Construct a playfair matrix with the key "moonmission" and encrypt the message "greet".

Soln. :

Is playfair cipher monoalphabetic cipher

- Playfair cipher is not technically mono-alphabetic. Monoalphabetic means that each plain text letter mapped with ciphertext letter and playfair is a digraph substitution - it maps two letter pairs to two letter pairs.

Scanned by CamScanner

- For playfair, the order is unchanged, we just substitute common diagraphs for randomize looking digraph. It is a substitution cipher.
- It uses pre-arranged key. So playfair is a private key cryptosystem.
- **Construct a playfair matrix with the key "moonmission" and encrypt the message "greet".**

  Use 5×5 matrix

m	o	n	i	s
a	b	c	d	e
f	g	h	k	l
p	q	r	t	u
v	w	x	y	z

- The message is "greet" divide the latter's into set of two characters.

  Message greet : gr ex et

  Ciphertext is : hq, cz, du

Ex. 1.11.3

Construct a playfair matrix with the key "occurrence". Generate the cipher text for plain text "tall tress".

Soln. :

Draw matrix 5×5

o	c	u	r	e
n	a	b	d	f
g	h	i/j	k	l
m	p	q	s	t
v	w	x	y	z

The message is "Tall trees" divide the latter's into the set of two character.

Massage Tall trees : Ta lx lt re es

Ciphertext is : pf, i/jz, tz, eo, rt

Scanned by CamScanner

Ex. 1.11.4

Use playfair algorithm with key "monarchy" and encrypt the text "jazz".

Soln. :

Draw matrix 5×5

m	o	n	a	r
c	h	y	b	d
e	f	g	i/j	k
u	p	q	s	t
u	v	w	x	z

The message is "Jazz" divide the latter's in the set of two characters.

Message : Jazz

ja zx, zx

Ciphertext : sb, uz, uz

Ex. 1.11.5

Using playfair cipher encrypt the plaintext "Why, don't you?". Use the key "keyword".

Soln. :

Draw matrix 5×5

k	e	y	w	o
r	d	a	b	c
f	g	h	i/j	l
m	n	p	q	s
t	u	v	x	z

The message is "why, don't, you ? " divide the latter's in to the set of two character.

Massage : Why, don't, you ?

Wh, yd, on, 'ty, ou

The ciphertext : yi/j, ea, es, vk, ez.

Scanned by CamScanner

Ex. 1.11.6

Use Play fair cipher to encrypt the following message "This is a columnar transposition." Use key APPLE.

Soln. :

The key used is APPLE so put it into 5 × 5 matrix.

Draw matrix 5 × 5

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

The plain text message is "This is a columnar transposition" divide the latter's in to the set of two character.

Message : "This is a columnar transposition"

- Now break the message "This is a columnar transposition" into pairs of two alphabets each. So message will look like as given

TH IS IS AC OL UM NA RT RA NS PO SI TI ON

- By using play fair cipher the cipher text obtained is given below:-

UG MQ MQ CI MB SO IE SU QP MT BK QM QN

1.11.5 One Time Pad (Vernam Cipher)

- One time pad invented by Vernam called as Vernam cipher that improves the security over substitution and transposition techniques.
- The one time pad technique uses a random key of the same length of the message (as long as the message), so that the key is not repeated. The case happens here is sender is generating new key for every new message while sending the message to the receiver called as one-time pad. The key is used to encrypt and decrypt a single message.
- Each new message requires a new key of the same length as the new message. This method is unbreakable. It produces random output with no relationship to the plaintext.

Scanned by CamScanner

- The algorithm used are as follows :
  1. Each alphabet will be treated as number following a = 0, b = 1.....and so on.
  2. Do the same for the key used for encrypting.
  3. Add both the key numbers and plaintext numbers.
  4. If the sum is greater than 26 (0 to 25), then subtract it from 26.
  5. Then the result number is to be translated into alphabets again.
- For Example plain text message is How Are You.

Plaintext	H	O	W	A	R	E	Y	O	U
	7	14	22	0	17	4	24	14	20
Key	n	c	b	t	z	q	a	r	x
	13	2	1	19	25	16	0	17	23
Total	20	16	23	19	42	20	24	31	43
Subtract 26 if >26 (0 to 25)	20	16	23	19	16	20	24	5	17
Ciphertext	U	Q	X	T	Q	U	Y	F	R

- So the ciphertext obtained for the plain text *how are you* is uqxtquyfr

For example

- The best example of one time pad is recharge voucher of any mobile company.
- All recharge voucher having different key or code imprinted on it. Once that code entered into mobile, customer will get talk time according to the voucher cost. If another customer trying to use same code of voucher he/she get recharge failure message. The company is regenerating all keys or code in such a way that every recharge voucher having new and unique code on it called one-time pad.
- Another example of one time pad is license software or license copy of operating system and antivirus having few keys available according to license. If license key is of 50 users ? Only 50 users can activate their software after 50 users the new user has to buy the new software along with new key. Once the key has been used nobody can use same key for activation.

Scanned by CamScanner

- Vernam Cipher has two **disadvantages :**
  1. Large random key cannot be created.
  2. Key distribution and generation of keys can be problematic.
  3. In terms of cryptography it is not possible to implement one time pad commercially because generating new key every time for sending a new message took more time to complete transmission process.

Syllabus Topic : Hill Cipher

1.11.6 Hill Cipher

Q. 1.11.5 Write short note on : Hill Cipher. (Ref. Sec.1.11.6)

- We can work on multiple letters at the same time using hill cipher.
- Hill cipher techniques were developed by the mathematician lester hill in 1929.
- When we use encryption algorithm we take the m successive plaintext letters and substitutes them in m cipher text latter.
- Hill cipher is a polygraphic substitution cipher based on linear algebra.
- Each letter is represented by a number modulo 26. Often the simple scheme (A = 0, B = 1, C = 2, ... Z = 25) is used, but this is not an essential feature of the cipher.
- To encrypt a plaintext message, each block of m letters (considered as an m-component vector) is multiplied by an invertible m × m matrix, against modulus 26. To decrypt the plaintext message, each block is multiplied by the inverse of the matrix used for encryption.
- The technique can be described as following way :

$$\boxed{\begin{aligned} C_{i1} &= (K_{11}, P_{i1} + K_{12} P_{i2} + K_{13} Pi_3) \bmod 26 \\ C_{i2} &= (K_{21}, P_{i1} + K_{22} P_{i2} + K_{23} P_{i3}) \bmod 26 \\ C_{i3} &= (K_{31}, P_{i1} + K_{32} P_{i2} + K_{33} P_{i3}) \bmod 26 \end{aligned}}$$

- This techniques uses column vectors and matrices :

$$\begin{bmatrix} C_{i1} \\ C_{i2} \\ C_{i3} \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} \begin{bmatrix} P_{i1} \\ P_{i2} \\ P_{i3} \end{bmatrix} \bmod 26$$

- This technique can be different way like,

$$C_i = KP_i \bmod 26$$

Scanned by CamScanner

Where,

C_i and P_i are the column vector which is hold length 3. K is used to representing the encryption key. Which is used 3×3 matrix for Example :

$$\text{Key (k)} = \begin{bmatrix} 2 & 5 & 4 \\ 9 & 1 & 2 \\ 3 & 7 & 17 \end{bmatrix}$$

- Using the vector we can represent the Key and first three letters of the plaintext HEL as follow.

$$C_i = KP_i \bmod 26$$

$$= \begin{bmatrix} 2 & 5 & 4 \\ 9 & 1 & 2 \\ 3 & 7 & 17 \end{bmatrix} \begin{bmatrix} 7 \\ 4 \\ 11 \end{bmatrix} \bmod 26$$

$$= \begin{bmatrix} 78 \\ 89 \\ 236 \end{bmatrix} \bmod 26$$

$$= \begin{bmatrix} 0 \\ 11 \\ 2 \end{bmatrix} = \text{ALC}$$

- For the given plaintext HEL, we get ciphertext is ALC.
- When we get the entire ciphertext it requires to do decryption using the inverse of the matrix K.
- Using the general terms in Hill cipher techniques is

Ciphertext $C_i = E_n(K, P_i) = KP_i \bmod 26$

Plaintext $P_i = D_n(K, P_i) = K^{-1}C_i \bmod 26 = K^{-1} KP_i = P_i$

Advantages of Hill Cipher Techniques

(1) In this techniques, when we used the large matrix, in hill cipher more frequency information hiding is possible when we uses a large matrix.

(2) Using the hill cipher techniques completely secrete single letter frequency.

Disadvantage of Hill Cipher Techniques

When we know the plaintext attack, it is easily broken.

Scanned by CamScanner

Ex. 1.11.7

Encrypt the message "Exam" using the Hill cipher with the key $\begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}$.

Soln. :

$$\text{key (k)} = \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix}$$

$$\text{Plaintext (Pi)} = \text{"Exam"}$$

$$\text{Ci} = \text{KPi mod 26} \begin{bmatrix} 9 & 4 \\ 5 & 7 \end{bmatrix} \times \begin{bmatrix} 4 & 23 \\ 0 & 12 \end{bmatrix} \text{mod 26}$$

$$= \begin{bmatrix} 151 & 167 \\ 60 & 84 \end{bmatrix} \text{mod 26}$$

$$= \begin{bmatrix} 21 & 11 \\ 8 & 6 \end{bmatrix}$$

$$= \text{VLIG}$$

For the given plaintext, we get ciphertext is **VLIG.**

Ex. 1.11.8

Use Hill cipher to encrypt the text DEF. The key to be used is $\begin{bmatrix} 2 & 4 & 5 \\ 9 & 2 & 1 \\ 3 & 8 & 7 \end{bmatrix}$

Soln. :

$$\text{Plain text (Pi)} = \text{DEF}$$

$$\text{Ci} = \text{KPi mod 26}$$

$$\text{Key (K)} = \begin{bmatrix} 2 & 4 & 5 \\ 9 & 2 & 1 \\ 3 & 8 & 7 \end{bmatrix} \times \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix} \text{mod 26}$$

$$= \begin{bmatrix} 47 \\ 40 \\ 76 \end{bmatrix} \text{mod 26} = \begin{bmatrix} 21 \\ 14 \\ 24 \end{bmatrix} = \text{voy}$$

For the given plaintext, we get ciphertext is **voy.**

Scanned by CamScanner

Ex. 1.11.9

Using hill cipher encrypt plain text "COE" use key "ANOTHERBZ".

Soln. :

Plain text (Pi) = COE

By using Hill Cipher

Key = ANOTHERBZ

It is represented as

$$Ci = KPi \bmod 26$$

$$= \begin{bmatrix} A & N & O \\ T & H & E \\ R & B & Z \end{bmatrix} \times \begin{bmatrix} C \\ O \\ E \end{bmatrix} \bmod 26$$

$$\text{Cipher Text (Ci)} = \begin{bmatrix} 0 & 13 & 14 \\ 19 & 7 & 4 \\ 17 & 1 & 25 \end{bmatrix} \times \begin{bmatrix} 2 \\ 14 \\ 4 \end{bmatrix} \bmod 26$$

$$= \begin{matrix} 238 \\ 152 \\ 148 \end{matrix} \bmod 26$$

$$= \begin{matrix} 4 \\ 22 \\ 18 \end{matrix} \text{ Mod } 26$$

$$= EWS$$

For the given plaintext, we get ciphertext is **EWS.**

Syllabus Topic : Transposition Techniques

1.12 Transposition Cipher Techniques

→ **(MU - May 16)**

Q. 1.12.1 Explain with examples, keyed and keyless transposition ciphers. (Ref. sec. 1.12) **May 16, 5 Marks**

Q. 1.12.2 Explain transposition cipher. (Ref. sec. 1.12)

Scanned by CamScanner

Q. 1.12.3 What is keyless Transposition Cipher? Give any example of rail fence cipher. (Ref. sec. 1.12)

Q. 1.12.4 What is transposition technique? (Ref. sec. 1.12)

- In transposition cipher technique plaintext message is hidden by rearranging the order of plain text letters without altering the original letter.
- In transposition cipher, the letters are written in a row under the key and then arrange the column as per alphabetical order.
- There are two types of transposition ciphers: single columnar and double columnar transposition ciphers.
- In transposition technique, there is no replacement of alphabets or numbers occurs instead their positions are changed or reordering of position of plaintext is done to produce ciphertext.
- Transposition cipher is a kind of mapping achieved by performing some sort of permutation on the plaintext message. Transposition cipher also called diffusion which performs permutations on plaintext.
- Diffusion means permutation of bit or byte positions.
- **There are two types of transposition techniques**

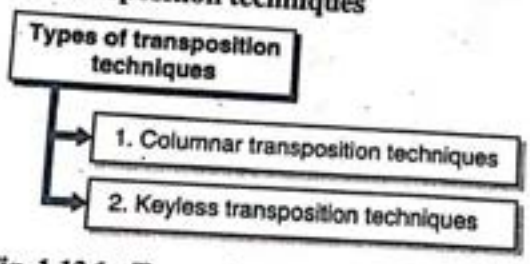


Fig. 1.12.1 : Types of transposition techniques

Syllabus Topic : Keyed Transposition Cipher

1.12.1 Columnar Transposition Technique

Q. 1.12.5 Describe the columnar transposition technique in detail. (Ref. sec. 1.12.1)

Columnar transposition technique is very simple to understand having following steps :

(1) Write plaintext message into a rectangle of some predefined size (rows and columns).

(2) Select the random key according to the size of rectangle also called columns. (In this technique order of the columns is the key).

Scanned by CamScanner

(3) Read the text present in each selected random key columns.

(4) Combine all text present in each column as per selected random key order.

(5) The resultant text called ciphertext shown in Fig. 1.12.2.

Example 1

Step 1 : Select any **Plaintext :** are you missing somebody.

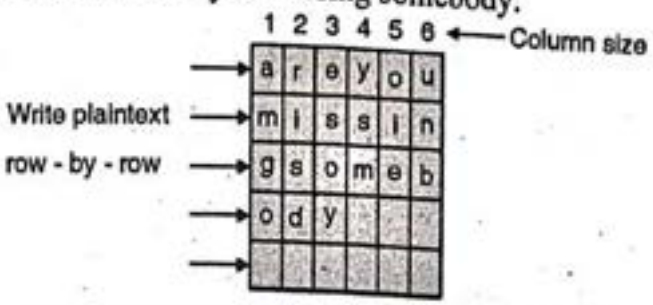


Fig. 1.12.2 : Columnar transposition technique

Step 2 : Select random key (according to column size) 5 4 2 3 1 6

Step 3 : Read text present in each column according to key.

Step 4 : oieysmrisdesoyamgounb

Step 5 : Final ciphertext is

Ciphertext : oieysmrisdesoyamgounb

The ciphertext obtained in step 5 can be made more complicated by performing multiple rounds of such permutations.

A more complex way to encrypt the message would be to write it in a rectangle, row by row, and then read off the message column by column, but to decide the order of the columns. The order of the column will be the key of the algorithm.

Example 2

Plaintext : the book is related to history.

1	2	3	4	5	6	7
t	h	e	b	o	o	K
i	s	r	e	l	a	T
e	d	t	o	h	i	S
t	o	r	y			

Scanned by CamScanner

Select the order of columns (Key) : 4351267

Ciphertext : beoyertryolhtiethsdooaikts

1.12.2 Keyless Transposition Techniques

- Keyless transposition technique also called **Rail fence technique.**
- Algorithm for keyless transposition technique is given below :
  (1) Write plaintext message into Zigzag order.
  (2) Read plaintext message of step 1 in order of row by row as shown in Fig. 1.12.3.

For example

- Plaintext message is : be care full while chatting.

Fig. 1.12.3 : Zigzag order of plaintext

- Write plaintext obtained in row 1 and row 2. The resultant ciphertext is

  Ciphertext : eaeulhlcatnbcrflwiehtig.
- This technique doesn't want any key. Rows are also fixed (2) so that attacker may get clu to break the ciphertext obtained using rail fence technique.

Ex. 1.12.1

Use Transposition Cipher to encrypt the plain text "WE ARE THE BEST" use key "HEAVEN".

Soln. :

Single Columnar Transposition

Single columnar transposition cipher is the simple cipher. Read the key, and numbered each letter of the key as per their appearance in the alphabet. The total encryption process is divided into three parts :

1. Preparing the Key
2. Preparing the Plaintext
3. Encryption

Scanned by CamScanner

1. **Preparing the Key :** Suppose the key is another. We can assign the number to each letter in this key as shown below :

   another

   1457326

   That is, the first letter a is numbered 1. There are no B's or C's, so the next letter to be numbered is the e. So e is numbered 2, followed by h, and so on. In the key. if the same letter has occurred more than one time, it should be numbered 1, 2, 3, etc. from left to write. For example, the key is heaven. Here e is occurred two times. So first 'e' from left hand side is numbered as 2, whereas second e is numbered as 3.

   heaven

   421635

2. **Preparing the Plaintext :** The letters from the message is written in rows under the numbered letters of the key. One letter from message is to be written under each letter of the key. Let us say that the message is - we are the best. We can write it as shown below :

   heaven

   421635

   WEARET

   HEBEST

3. **Encryption :** Now, arrange the above message written in rows under the numbered letters of the key as per ascending order of the numbers at the top of the plaintext letters.

   aeehnv

   123456

   AEEWTR

   BESHTE

   Then the letters are copied down column wise from top to bottom. The result is ciphertext, i.e.

   ABEEESWHTTRE

Scanned by CamScanner

Syllabus Topic : Steganography

1.13 Steganography Applications and Limitations

Q. 1.13.1 What is steganography ? Give its advantages and disadvantages. (Ref. sec. 1.13)

Q. 1.13.2 What is steganography? What are applications and limitations of steganography? (Ref. sec. 1.13)

- Steganography is a technique of hiding the message, file and image within other message, file or image. Steganography word is of Greek origin that means "covered writing" or "secret writing".
- In other words, it is the art as well as science of hiding the communication in such way that know body aware the existence of communication.
- The goal of hiding messages, files and images is to fool attacker and not even allow attacker to detect that there is another message hidden in original message.
- The main aim of Steganography is to achieve high security and encode the sensitive data in any cover media like images, audio, video and send it over insecure channel such as Internet. Even if there is small change in stenographic image or data will change complete meaning of the messages.
- In Steganography the term called as **Steganalysis** which is similar to cryptanalysis in cryptography. Like cryptanalysis, the goal of steganalysis is to identify suspected messages, files and images, to determine whether any hidden or encoded information is available into those messages, and also to try and gain access to that messages, files and images.
- Attacks on steganographic techniques are very similar to attacks on cryptographic techniques/ algorithms and similar techniques apply. The strength of a steganographic algorithm depends on its ability to successfully withstand attacks.

Following are the possible attacks on steganography.

1. **Stego-only attack :** In this type of attack, only the medium (files and images) containing hidden data is available for analysis. This attack also called as Visual attack.
2. **File only attack :** The attacker has access the file he must determine if there is a message hidden information inside that file.

Scanned by CamScanner

3. **Known carrier attack :** In this type of attack, the steganalyst has access to both the original image and the image containing the hidden information are available and compared to assume the message. The stego object (that contains the hidden information) is compared with the cover object and the differences are detected. This attack also called as file an original copy : It might happen that attacker have a copy of both the file i.e. the encoded message and a copy of the original file. If the two files are different, then attacked may guess that there must be some hidden information inside a file. Such type attacks helps attacker to destroy the hidden information by simply replacing it with original message.

4. **Reformat Attack :** Most popular attack on steganography is reformat attack, in this attack attacker change the format of the file (BMP, GIF, JPEG) because different file formats don't store data in exactly same way.

5. **Destroy Everything Attack :** An attacker could simply destroy the message and all related information. This can works correctly because there are different file formats are used to store data in different ways.

6. **Known message attack :** In this type of attack, the original message prior to embedding and when transmitting over Internet is known to sender. This type of attack analysis can help against attacks in the future.

7. **Multiple Encoding of a Files :** The attacker gets *n* different copies of the files with *n* different messages. It might happen if some companies are inserting different tracking information into each file. If the attacker tracks all the data during transmission then he may try to replace the tracking information with its own available information.

8. **Compression Attack :** One of the simplest attacks is to compress the file. This type of attack tries to remove the unrelated information from a file during compression then what is the use of hiding the data if extraneous information is removed.

Chapter Ends...

□□□

Scanned by CamScanner

CHAPTER 2

Module 1

Modular Arithmetic and Number Theory

Syllabus

Modular Arithmetic and Number Theory : Euclid's algorithm - Prime numbers-Fermat's and Euler's theorem - Testing for primality - The Chinese remainder theorem, Discrete logarithms.

2.1 Modular Arithmetic

2.1.1 Mathematical Background

- In cryptography, there are certain mathematical backgrounds for understanding modern data encryption methods and also to introduce certain application in cryptography area.
- Mostly certain areas of Number theory and algebra techniques are used for designing cryptosystems. One of the examples of cryptosystem in DES and AES algorithms.

Basics of Number Theory and Modular Arithmetic

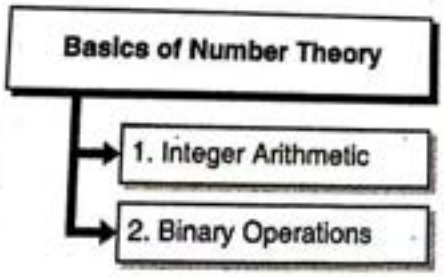


Fig. 2.1.1 : Basics of Number Theory

1. Integer Arithmetic

- Integer arithmetic is used to create background for modular arithmetic. The set of integer number represented by single alphabet i.e. Z.

Scanned by CamScanner

- For example :

  $Z = \{\ldots -3, -2, -1, 0, 1, 2, 3, \ldots\}$ set of integers.

➜ **2. Binary Operations**

- In cryptography different binary operations are applied to set of integers. Among those three basic operations are addition, subtraction and multiplication.
- The concept here is to provide two inputs and generate only one output. Set of integers is not closed under operation division, i.e. quotient of two integers may not be integer.

Syllabus Topic : Prime Numbers

2.1.1(A) Prime Numbers

- A prime number is a positive integer which is greater than one and which is divisible by 1 and itself i.e. the only factors of prime number are 1 and itself.

  Example : 2, 3, 5, 7, 11 etc.
- The prime number concept is use by cryptographic algorithms heavily.
- **Relatively prime number :** Two numbers are relatively prime to one another when they have no common factors except 1.
- **Example :** 21 and 44 are relatively prime numbers, since common factors between both the number is one i.e. Greatest Common Divisor GCD of a and n is 1 it is written as gcd (a, n) = 1 hence they both are relatively prime.

2.1.1(B) What Is GCD?

GCD stands for Great Common Divisor, also called as greatest common divisor of two positive integer a and b. The GCD of two integers is the largest integer that can divide both integers. Most needed concept of cryptography is GCD and prime number. Two positive integers may have common divisors but we are interested in largest divisor.

For example :

gcd(3,5) = 1 hence 3 and 5 are relatively prime numbers to each other.

gcd(12,60) = 12

Scanned by CamScanner

Explanation

- Factors of 12 = 1, 2, 3, 4, 6, 12
- Factors of 60 = 1, 2, 3, 5, 6, 12, 60
- So looking to above example it is observed that 12 is the greatest and also common factor between both the numbers so.

  gcd(12,60) = 12.
- **For example :** Find GCD (40, 20) :

Method 1

Find prime factors of given numbers,

$$40 = \underline{2}*\underline{2}*2*\underline{5}$$

$$20 = \underline{2}*\underline{2}*\underline{5}$$

Select common divisor from above

$\therefore$ gcd (40, 20) = $2 * 2 * 5 = 20$

$\therefore$ **gcd (40, 20) = 20**

Method 2

$$40 = 2 * 20$$

$$20 = 1 * 20$$

$\therefore$ gcd (40, 20) = 20

1. R = a mod b
2. If (R = 0) then go to step 4 else
3. a = b, b = R go to step 2
4. GCD = b

Example

$$GCD = (15, 12)$$

$$15 \% 12 = 3$$

$$12 \% 3 = 0$$

Hence

GCD (15, 12) = 3

Scanned by CamScanner

Modular Arithmetic

- Modular arithmetic is a simple concept of using Remainder which is left after an integer division since *MODULO* is the remainder in mathematical terminology.
- Let a, b, $\in$ Z and n $\in$ N then a $\equiv$ b (mod n) if (a-b/n) where Z is set of integers and N is set of natural numbers.
- **For example :** 23 $\equiv$ 1 (mod 11). Congruence calculus is often called as Modular arithmetic. Modular arithmetic considers that 23 and 11 mod (12) is equivalent. Because both leaves same remainder when divided by 12.
- If a, b, $\in$ Z be any integers, then $\exists$ q, r such that b = aq + r where $0 \leq r < a$ where q and r quotient and remainder respectively, when b is divided by a.
- Most of the cryptographic concepts and algorithms use the concept of GCD, modular arithmetic.
- Modular Arithmetic exhibits the following properties :
  1. [(a mod n) + (b mod n)] mod n = (a + b) mod n
  2. [(a mod n) – (b mod n)] mod n = (a – b) mod n
  3. [(a mod n) * (b mod n)] mod n = (a * b) mod n

Syllabus Topic : Euclid's Theorem

2.2 Euclid's or Euclidean Algorithm

Q. 2.2.1 State Euclid's Algorithm with example. (Ref. sec. 2.2)

It is a basic technique or a method for calculation of GCD of two positively integer. Suppose we have two integers a, b such that d = gcd (a, b) assuming a > b > 0.

Now dividing a by b we can state that

$$a = q_1 b + r_1 \qquad 0 \leq r_1 < b \qquad \text{...(2.2.1)}$$

Where q implies quotient and r implies remainder.

Suppose that $r_1 \neq 0$ because b > r1, we can divide b by r_1 and apply division algorithm to obtain :

$$b = q_2 r_1 + r_2 \qquad 0 \leq r_2 < r_1$$

if $r_2 = 0$ then $d = r_1$ and if $r_2 \neq 0$ then d = gcd (r_1, r_2). The division process continues till the remainder is 0. The following equation describes the process.

Scanned by CamScanner

$$
\begin{aligned}
a &= q_1 b + r_1 & 0 < r_1 < b \\
b &= q_2 r_1 + r_2 & 0 < r_2 < r_1 \\
r_1 &= q_3 r_2 + r_3 & 0 < r_3 < r_2 \\
&\vdots & \vdots \\
r_n - 2 &= q_n r_{n-1} + r_n & 0 < r_n < r_{n-1} \\
r_{n-1} &= q_{n+1} r_n + 0 \\
d &= \gcd(a, b) = r_n
\end{aligned}
$$

At each stage we have $d = \gcd(r_i, r_{i+1})$ finally $d = \gcd(r_n, O) = r_n$. Thus, the greatest common divisor of two integers by repetitive can be calculated by repetitive application of the division algorithm This is known as the Euclidean algorithm.

Euclid's algorithm : EUCLID (x, y)

1. $x \rightarrow x, y \rightarrow y$
2. If $y = 0$, return $x = \gcd(x, y)$
3. $R = x \bmod y$
4. $x \leftarrow y$
5. $y \leftarrow R$
6. Go to 2.

Ex. 2.2.1

Find GCD (40, 20) using Euclid's algorithm.

Soln. :

We know that,

$$\gcd(x, y) = \gcd(y, x, \bmod y)$$

$$\therefore \gcd(40, 20) = \gcd(20, 40 \bmod 20) = \gcd(20, 0)$$

From step 2 of algorithm,

If $y = 0$, return $x = \gcd(x, y)$

$$\therefore \quad \mathbf{gcd = 20}$$

Scanned by CamScanner

Ex. 2.2.2

Solve / find gcd (36, 10).

Soln. :

We know that,

$$\gcd(x, y) = \gcd(y, x \bmod y)$$

$$\gcd(36, 10) = \gcd(10, 36 \bmod 10)$$

$$= \gcd(10, 6)$$

$$\therefore \gcd(10, 6) = \gcd(6, 10 \bmod 6)$$

$$= \gcd(6, 4)$$

$$\gcd(6, 4) = \gcd(4, 2)$$

$$\gcd(4, 2) = \gcd(2, 0)$$

$$\therefore \gcd(36, 10) = 2$$

If gcd (x, y) = 1 then x and y are relatively prime to each other.

e.g. gcd (8, 15) = 1

∴ 8 and 15 are relatively prime.

Ex. 2.2.3

Using Euclidean algorithm calculate gcd(48,30) and gcd(105,80).

Soln. :

(I) gcd(48,30)

$$= \gcd(30, 48 \bmod 30)$$

$$= \gcd(30, 18)$$

$$= \gcd(18, 30 \bmod 18)$$

$$= \gcd(18, 12)$$

$$= \gcd(12, 18 \bmod 12)$$

$$= \gcd(12, 6)$$

$$= \gcd(6, 12 \bmod 6)$$

$$= \gcd(6, 0)$$

$$\therefore \gcd(48,30) = 6$$

Scanned by CamScanner

(ii) gcd(105,80)

$$= \gcd(105,80)$$
$$= \gcd(80,105 \bmod 80)$$
$$= \gcd(80,25)$$
$$= \gcd(25,80 \bmod 25)$$
$$= \gcd(25,5)$$
$$= \gcd(5,25 \bmod 5)$$
$$= \gcd(5,0)$$

$$\therefore \gcd(105,80) = 5$$

Ex. 2.2.4

Using Euclidean algorithm calculate gcd(20,16) and gcd(50,60).

Soln. :

(i) gcd(20,16)

$$= \gcd(16,20 \bmod 16)$$
$$= \gcd(16,4)$$
$$= \gcd(4,16 \bmod 4)$$
$$= \gcd(4,0)$$

$$\therefore \gcd(20,16) = 4$$

(ii) gcd(50,60)

$$= \gcd(50,60 \bmod 50)$$
$$= \gcd(50,10)$$
$$= \gcd(10,50 \bmod 10)$$
$$= \gcd(10,0)$$

$$\therefore \gcd(50,60) = 10$$

Scanned by CamScanner

2.2.1 Extended Euclidean Algorithm

Q. 2.2.2 Write Extended Euclidian algorithm. (Ref. sec. 2.2.1)

- As we learn in school days, when we divide integer values by other nonzero integer we get an integer *quotient* (the "answer") plus a *remainder* (generally a rational number).
- For instance,

$$13/5 = 2 \text{ ("the quotient")} + 3/5 \text{ ("the remainder")}.$$

- We can rephrase this division, totally in terms of integers, without reference to the division operation :

$$13 = 2(5) + 3$$

- Note that this expression is obtained from the one above it by multiplying through by the divisor 5.
- The *greatest common divisor* of integers a and b, denoted by *gcd(a, b)*, is the largest integer that divides (without remainder) both a and b. So, for example :

  gcd(15, 5) = 5, gcd(7, 9) = 1,

  gcd(12, 9) = 3, gcd(81, 57) = 3.
- The gcd of two integers can be found by repeated application of the division algorithm, this is known as the ***Euclidean Algorithm***. You repeatedly divide the divisor by the remainder until the remainder is 0. The gcd is the last non-zero remainder in this algorithm. The following example shows the algorithm.
- Finding the gcd of 81 and 57 by the Euclidean Algorithm :

$$81 = 1(57) + 24$$
$$57 = 2(24) + 9$$
$$24 = 2(9) + 6$$
$$9 = 1(6) + 3$$
$$6 = 2(3) + 0$$

- It is well known that if the gcd(a, b) = r then there exist integers p and s so that :

$$p(a) + s(b) = r$$

- By reversing the steps in the Euclidean Algorithm, it is possible to find these integers p and s. We shall do this with the above example :

  Starting with the next to last line, we have : $3 = 9 - 1(6)$

Scanned by CamScanner

- From the line before that, we see that $6 = 24 - 2(9)$, so :

$$3 = 9 - 1(24 - 2(9)) = 3(9) - 1(24)$$

- From the line before that, we have $9 = 57 - 2(24)$, so :

$$3 = 3(57 - 2(24)) - 1(24)$$
$$= 3(57) - 7(24)$$

- And, from the line before that $24 = 81 - 1(57)$, giving us :

$$3 = 3(57) - 7(81 - 1(57))$$
$$= 10(57) - 7(81)$$

- So we have found $p = -7$ and $s = 10$.
- The procedure we have followed above is a bit messy because of all the back substitutions we have to make. It is possible to reduce the amount of computation involved in finding p and s by doing some auxiliary computations as we go forward in the Euclidean algorithm (and no back substitutions will be necessary). This is known as the ***extended Euclidean Algorithm.***
- Given two integers a and b, we after need to find other two integers S and t such that

$$S \times a + t \times b = \gcd(a, b)$$

- The extended Euclidean algorithm can calculate the gcd (a, b) and at the same time calculate the value of S and t. As show the Fig. 2.2.1.

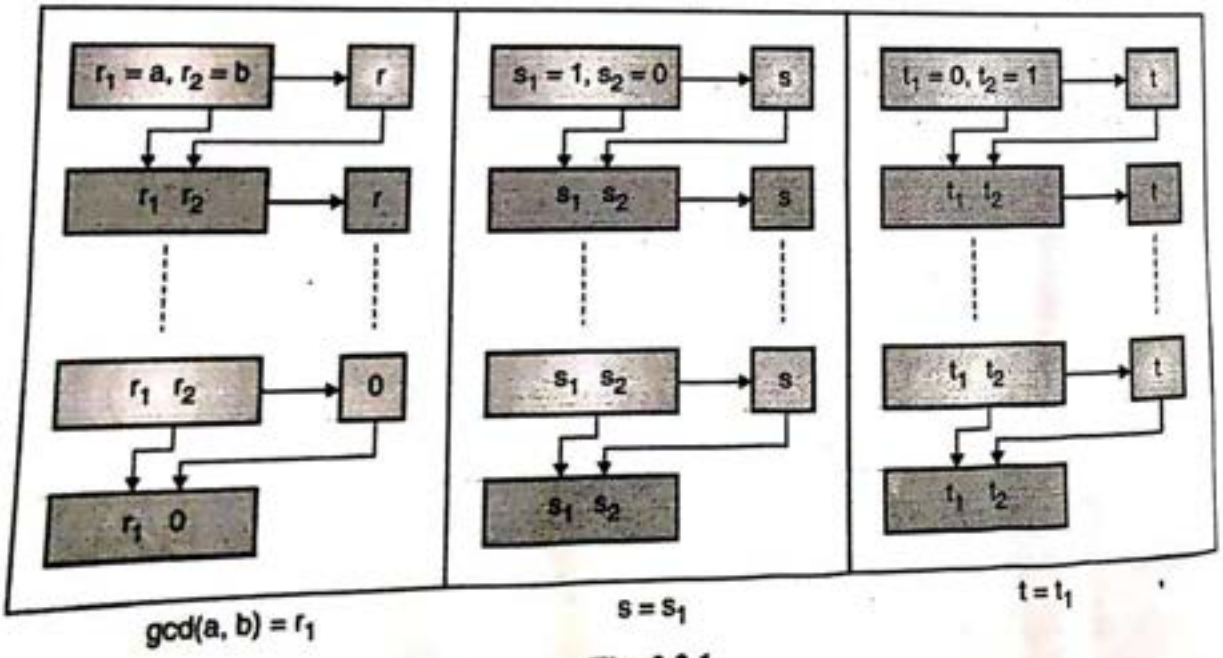


Fig. 2.2.1

Scanned by CamScanner

- The extended Euclidean algorithm uses the same number of steps as the Euclidian algorithms.
- The algorithm of extended Euclidean is as shown below :

$$\left.\begin{array}{l} r_1 \leftarrow a \;;\; r_2 \leftarrow b\,; \\ s_1 \leftarrow 1 \;;\; s_2 \leftarrow 0\,; \\ t_1 \leftarrow 0 \;;\; t_2 \leftarrow 1\,; \end{array}\right\} \text{Initialization}$$

while ($r_2 > 0$)

{

$q \leftarrow r_1/r_2\,;$

$$\left.\begin{array}{l} r \leftarrow r_1 - q \times r_2\,; \\ r_1 \leftarrow r_2\,;\, r_2 \leftarrow r\,; \end{array}\right\} \text{updating r's}$$

$$\left.\begin{array}{l} s \leftarrow s_1 - q \times s_2\,; \\ s_1 \leftarrow s_2\,;\, s_2 \leftarrow s; \end{array}\right\} \text{updating s's}$$

$$\left.\begin{array}{l} t \leftarrow t_1 - q \times t_2\,; \\ t \leftarrow t_2\,;\, t_2 \leftarrow t; \end{array}\right\} \text{updating t's}$$

}

$\gcd(a, b) \leftarrow r_1\,;\, s \leftarrow s_1\,;\, t \leftarrow t_1$

Ex. 2.2.5

Given a = 161 and b = 28 find gcd (a, b) and value of s and t.

Soln. :

$r = r_1 - q \times r_2$

$s = s_1 - q \times s_2$

$t = t_1 - q \times t_2$

Scanned by CamScanner

Q	r_1	r_2	r	s_1	s_2	S	t_1	t_2	t
5	161	28	21	1	0	1	0	1	−5
1	28	21	7	0	1	−1	1	−5	6
3	21	7	0	1	−1	4	−5	6	−23
	7	0		−1,	4		6	−23	

We get, gcd (161, 28) = 7 s = − 1 and t = 6. The answer can be tested ,

$$(-1) \times 161 + 6 \times 28 = 7$$

Syllabus Topic : The Chinese Remainder Theorem

2.3 Chinese Remainder Theorem

Q. 2.3.1 State the Chinese remainder theorem with example. (Ref. sec. 2.3)

- According to D. Wells, the following problem was posed by Sun TsuSuanChing. There are certain number repeatedly divided by 3 and remainder is 2, repeatedly divided by 5 and remainder is 3, repeatedly divided by 7 and remainder is 2.
- What will be that number? The problem can be solved by well known theorem called Chinese remainder theorem. If that it is possible to reconstruct integers in a certain range from their residue modulo a set of pair wise relatively prime modulo.

$$\begin{aligned} x &= a_1 \pmod{m_1} \\ x &= a_2 \pmod{m_2} \\ &\vdots \\ x_n &= a_n \pmod{m_n} \end{aligned}$$

- Consider n positive integers which are relatively prime in pairs i.e. $m_1, m_2 \ldots m_n$. Then the congruence of above equation have common solutions.
- Two integers a and b are said to be congruent modulo n, if (a mod n) = (b mod n)

i.e. $\boxed{a \equiv b \pmod{n}}$

where $X \equiv a_1 \pmod{m_1}$ $X \equiv a_2 \pmod{m_2}$ $X \equiv a_3 \pmod{m_3} \ldots\ldots X \equiv a_r \pmod{m_r}$

where $M = m_1 x m_2 x m_3$

which is given by $X \equiv a_1M_1y_1 + a_2M_2y_2 + a_3M_3y_3 + \ldots\ldots\ldots\ldots a_rM_ry_r \pmod{M}$,

where $M_i = M/m_i$ and $y_i = (M_i)^{-1} \bmod (m_i)$ for $1 \le i \le r$.

Scanned by CamScanner

Ex. 2.3.1

State CRT. Find x for the following equations,

$$x = 1 \bmod 2$$
$$x = 1 \bmod 3$$
$$x = 3 \bmod 5$$
$$x = 1 \bmod 7$$

Soln. :

The general equation of Chinese remainder theorem is,

$$x = a_1 \pmod{m_1}$$
$$x = a_2 \pmod{m_2}$$
$$\vdots$$
$$x_n = a_n \pmod{m_3}$$

Let, $a_1 = 1$, $a_2 = 1$, $a_3 = 3$, $a_4 = 1$ and $m_1 = 2$, $m_2 = 3$, $m3 = 5$ and $m_4 = 7$

$$M = m_1 * m_2 * m_3 * m_4$$
$$= 2 * 3 * 5 * 7$$
$$M = 210$$

For all set of elements

$$M1 = M/m1 = \frac{210}{2} = 105,$$

$$M2 = M/m2 = \frac{210}{3} = 70,$$

$$M3 = M/m3 = \frac{210}{5} = 42,$$

$$M4 = M/m4 = \frac{210}{7} = 30$$

Find out inverse modulo.

Let $y_1, y_2, y_3 \ldots y_n$ represented as inverse.

$$y_1 = \text{Inverse of } (M/m1) \bmod m_1 = \text{Inverse of } 105 \bmod 2$$
$$= 105^{2-2} \bmod 2 = 105 \bmod 2 = 1$$
$$y_2 = \text{Inverse of } (M/m2) \bmod m_2 = \text{Inverse of } 70 \bmod 3 = 70^{3-2} \bmod 3$$
$$= 70 \bmod 3 = 1$$

Scanned by CamScanner

y_3 = Inverse of (M/m3) mod m_3 = Inverse of (42) mod 5 = 42^{5-2} mod 5

= 42^3 mod 5 = 3

y_4 = Inverse of (M/m4) mode m_4 = Inverse of 30 mod 7 = 30^{7-2} mod 7

= 4

According CRT

$$X \equiv a_1M_1y_1 + a_2M_2y_2 + a_3M_3y_3 + \ldots\ldots\ldots\ldots a_rM_ry_r \pmod{M},$$

$$X = [(105) * 1 * 1 + (70) * 1 * 1 + (42) * 3 * 3 + (30) * 1 * 4] \bmod 210$$

$$= [105 + 70 + 378 + 120] \bmod 210 = 673 \bmod 210$$

$$\therefore X = 43$$

Ex. 2.3.2

A bag has certain number of pens. If you take out 3 pen at a time, 2 pens are left, If you take 4 pens at a time, 1 pen is left and if you take out 5 pen at a time, 3 pens are left in the bag. What is the smallest number of pens in the bag ?

Soln. :

$$x \equiv 2 \bmod 3, \quad x \equiv 1 \bmod 4, \quad x \equiv 3 \bmod 5 \qquad \ldots(1)$$

The general equation of Chinese remainder theorem is,

$$x = a_1 \pmod{m_1}$$

$$x = a_2 \pmod{m_2}$$

$$\vdots$$

$$x_n = a_n \pmod{m_3}$$

According to given equation a1= 2, a2 = 1 & a3 = 3 and m_1= 3, m_2 = 4 and m_3 = 5

$$M = m_1 * m_2 * m_3 * m_4$$

$$= 3 * 4 * 5$$

$$M = 60$$

For all set of elements

M1 = M/m1 = 60/ 3= 20,

M2 = M/m2 = 60/4= 15,

M3 = M/m3 = 60/5= 12,

Find out inverse modulo.

Let $y_1, y_2, y_3 \ldots y_n$ represented as inverse.

Scanned by CamScanner

y_1 = Inverse of (M/m1) mod m_1 = Inverse of 20 mod 3

= 20^{2-2} mod 3 = 20 mod 3 = 2

y_2 = Inverse of (M/m2) mod m_2 = Inverse of 15 mod 3 = 15^{4-2} mod 4

= 225 mod 4 = 1

y_3 = Inverse of (M/m3) mod m_3 = Inverse of 12 mod 5 = 12^{5-2} mod 5

= 12^3 mod 5 = 3

According CRT

$$X \equiv a_1M_1y_1 + a_2M_2y_2 + a_3M_3y_3 + \ldots\ldots\ldots\ldots a_rM_ry_r \pmod{M},$$

$$X = [2 * 20 * 2 + 1 * 15 * 1 + 3 * 12 * 3] \bmod 60$$

$$= [80 + 15 + 108] \bmod 60 = 203 \bmod 60$$

$$\therefore \mathbf{X = 23}$$

Ex. 2.3.3

Define the Chinese remainder theorem find the solution to the simultaneous equations. x = 2 mod 3, x = 3 mod 5, x = 2 mod 7.

Soln. :

The general equation of Chinese remainder theorem is,

$$x = a_1 \pmod{m_1}$$

$$x = a_2 \pmod{m_2}$$

$$\vdots$$

$$x_n = a_n \pmod{m_3}$$

According to given simultaneous equations

x = 2 mod 3, x = 3 mod 5, x = 2 mod 7

Let, $a_1 = 2$, $a_2 = 3$, $a_3 = 2$, and $m_1 = 3$, $m_2 = 5$, $m_3 = 7$

$$M = m_1 * m_2 * m_3 * m_4$$

$$= 3 * 5 * 7$$

$$M = 210$$

For all set of elements

$$M_1 = M/m_1 = 105/3 = 35,$$

$$M_2 = M/m_2 = 105/5 = 21,$$

$$M_3 = M/m_3 = 105/7 = 15,$$

Scanned by CamScanner

Find out inverse modulo.

Let $y_1, y_2, y_3 \ldots y_n$ represented as inverse.

$$y_1 = \text{Inverse of } (M/m_1) \bmod m_1 = \text{Inverse of } 35 \bmod 3$$
$$= 105^{3-2} \bmod 3 = 35 \bmod 3 = 2$$
$$y_2 = \text{Inverse of } (M/m_2) \bmod m_2 = \text{Inverse of } 21 \bmod 5 = 21^{5-2} \bmod 5$$
$$= 21^3 \bmod 5 = 1$$
$$y_3 = \text{Inverse of } (M/m_3) \bmod m_3 = \text{Inverse of } (15) \bmod 7 = 15^{7-2} \bmod 7$$
$$= 15^5 \bmod 7 = 1$$

According Chinese Remainder Theorem

$$X \equiv a_1M_1y_1 + a_2M_2y_2 + a_3M_3y_3 + \ldots\ldots\ldots\ldots a_rM_ry_r \pmod{M},$$
$$X = [2 * 35 * 2 + 3 * 21 * 1 + 2 * 15 * 1] \bmod 105$$
$$X = [140 + 63 + 30] \bmod 105 = 233 \bmod 105$$
$$\mathbf{X = 23}$$

Syllabus Topic : Euler's Theorem

2.4 Euler Totient Function $\phi(n)$

$\phi(n)$ is called Euler Totient function which states that how many numbers are between 1 and n – 1 that are relatively prime to n.

All positive integers less than n are relatively prime to it.

For example : if n = 4 find $\phi(n) = ?$

1. $\phi(n) = \phi(4) = 1$ and 3 are relatively prime to 4 because their gcd is 1. (The number is said to relatively prime if their gcd is always 1)

   In this case 4 and 2 are not relatively prime, because gcd (4, 2) = 2

   $$\therefore \phi(4) = 2 \text{ (i.e. 1 and 3)}$$

2. $\phi(5) = ?$

   $\phi(5) = 1, 2, 3, 4$ are relatively prime to 5

   $\therefore \phi(5) = 4$

3. $\phi(7) = ?$

   $\phi(7) = 1, 2, 3, 4, 5, 6$ are relatively prime 7

Scanned by CamScanner

$\therefore \phi(7) = 6$

This implies that $\phi(n)$ will be easy to calculate if n has exactly two different prime factors say p and q

$\therefore \phi(n) = p \times q$

i.e. $\phi(n) = \phi(p) * \phi(q)$

According to definition of Euler Totient function,

$\therefore \phi(p * q) = (p-1) * (q-1)$ if p and q are prime

Hence proved.

2.4.1 Euler's Theorem

It states that for every a and n that are relatively prime :

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

For example : Prove using Euler's theorem.

$a = 3; \quad n = 10; \quad \phi(n) = ?$

Let, $\phi(n) = \phi(10) = \{1, 3, 7, 9\} = 4$

According to Euler's theorem,

$\therefore \quad 3^4 = 1 \pmod{10}$

$3^4 = 1 \bmod 10$

$81 = 1 \bmod 10$ (81 mod 10 = 1 & 1 mod 10 = 1)

Hence proved.

Ex. 2.4.1

Solve using Euler's theorem : a = 2 and n = 11.

Soln. :

Calculate $\phi(n)$ = First

$\phi(n) = \phi(11) = \{1 \text{ to } 10\} = 10$

According Euler's theorem

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

$\therefore 2^{10} = 1 \pmod{n}$

$\therefore \quad 1024 = 1 \bmod 11$ (1024 mod 11 = 1 and 1024 mod 11 = 1)

Hence proved.

Scanned by CamScanner

Syllabus Topic : Discrete Logarithm

2.5 Discrete Logarithm

To understood discrete logarithm we must know what is primitive root ?

Primitive root

- A primitive root of a prime number p is one whose power of modulo p generates all integers from 1 to p – 1.
- If a is a primitive root of prime number p, the numbers a mod p, a^2 mod p, ... a^{p-1} mod p are distinct and consists of integers from 1 to p – 1.
- For any integer b of primitive root a of prime number p, we can find a unique exponent i such that,

$$b = a^i \bmod p$$

- The exponent i is referred as the discrete logarithm of b for base a mod p.
- This can be written as,

$$d \log_{a,} p(b)$$

- The equation $d \log_{a,} p(b)$ called as discrete logarithm which is used in different cryptographic algorithms like diffie Hellman algorithm and digital signature algorithms.

For a fix prime number P. Let a, b be non zero integers (mod p). The problem of finding X such that $a^X \equiv b(\text{mod } p)$ is called Discrete Logarithm Problem. Suppose that n is the smallest integer such that $a^n \equiv 1 \pmod p$, i.e., n = or $d_p(a)$. By assuming $0 \le x < n$, we denote $x = L_a(b)$, and call it the discrete log of b w.r.t. a (mod p).

Example : P = 11, a = 2, b = 9, then $x = L_a(b) = L_2(9) = 6$

To explain discrete Logarithm considers the following example.

$$7^1 \equiv 7 \pmod{19}$$

$$7^2 = 49 = 2 \times 19 + 11 \equiv 11 \pmod{19}$$

$$7^3 = 343 = 18 \times 19 + 1 \equiv 1 \pmod{19}$$

$$7^4 = 2401 = 126 \times 19 + 7 \equiv 7 \pmod{19}$$

$$7^5 = 16807 = 884 \times 19 + 11 \equiv 11 \pmod{19}$$

There is no poin
noting that $7^3 \equiv 1$
positive a < 19.

A	a^2	a^3	a^4
1	1	1	1
2	4	8	16
3	9	8	5
4	16	7	9
5	6	11	17
6	17	7	4
7	11	1	7
8	7	18	1
9	5	7	
10	5	12	
11	7	1	
12	11	18	
13	17	12	
14	6	8	
15	16	12	
16	9	11	
17	4	11	
18	1	18	

In general t
(mod n) is $\phi(n)$.

If a number i

Scanned by CamScanner

There is no point in continuing because the sequence is repeating. This can be proven by noting that $7^3 \equiv 1 \pmod{19}$. Table 2.5.1 shows all the powers of a, modulo 19 for all positive a < 19.

Table 2.5.1

A	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1
6	17	7	4	5	11	9	16	1	6	17	7	4	5	11	9	16	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
8	7	18	11	12	1	8	7	18	11	12	1	8	7	18	11	12	1
9	5	7	6	16	11	4	17	1	9	5	7	6	16	11	4	17	1
10	5	12	6	3	11	15	17	18	9	14	7	13	16	8	4	2	1
11	7	1	11	7	1	11	7	1	11	7	1	11	7	1	11	7	1
12	11	18	7	8	1	12	11	18	7	8	1	12	11	18	7	8	1
13	17	12	4	14	11	10	16	18	6	2	7	15	5	8	9	3	1
14	6	8	17	10	7	3	4	18	5	13	11	2	9	12	16	15	1
15	16	12	9	2	11	13	5	18	4	3	7	10	17	8	6	14	1
16	9	11	5	4	7	17	6	1	16	9	11	5	4	7	17	6	1
17	4	11	16	6	7	5	9	1	17	4	11	16	6	7	5	9	1
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

In general terms, that the highest possible exponent to whom a number can belong (mod n) is $\phi(n)$.

If a number is of this order, then it is called as **primitive root** of n.

Scanned by CamScanner

The importance of this notion is that if is a primitive root of n then its powers is,

a, a^2 .....a $^{\phi(n)}$ are distinct(mod n) and are all relatively prime to n. In particular, for a prime number p, if is a primitive root of p, then,

a, a^2 .....a $^{\phi(n)}$ are distinct (mod n). For the prime number 19, its primitive roots are 2, 3, 10, 13, 14, and 15.

2.6 Fermat Theorem

Q. 2.6.1 State and Prove Fermat's theorem. (Ref. sec. 2.6)

Fermat theorem plays an important role in public key cryptography. For this theorem to understand one has to have knowledge of Prime number, Co-prime number, prime factorization and GCD i.e. greatest common divisor that has already been explained in this chapter.

Theorem

- For any prime number p , a is the integer which is not divisible by p then

$$a^{p-1} \equiv 1(\text{mod } p) \quad ...(2.6.1)$$

A variant of this theorem is

- If p is a prime and a is a co prime to p (i.e gcd(a, p) = 1) then,

$$a^p \equiv a(\text{mod } p) \quad ...(2.6.2)$$

- Basically this theorem is useful in public key RSA and primarily testing.

Let us have a = 3 and p = 5 then as per the above theorem in Equation (2.6.1) we have

$3^{5-1} \equiv 3^4 = 81 \equiv 1(\text{mod } 5)$.

Since on dividing 81 with 5 will have remainder 1.

Hence proof above theorem.

- Considering another form of theorem in Equation (2.6.2).

Let us have a = 3 and p = 5 then we have

$$a^p = 3^5 = 243.$$

- Now we calculate 243 mod 5, we will have result 3.

a mod p = 3 mod 5 = 3

Hence, $3^5 = 3 \text{ mod } 5$.

Scanned by CamScanner

Fermat theorem used to find mod of large number for example :

Ex. 2.6.1

Solve 6^{10} mod 11.

Soln. :

We know that Fermat theorem.

$$a^{p-1} = 1 \bmod p$$

Hence $$6^{10} = 1 \bmod 11 \quad \text{...(1)}$$

$$a = 6, p = 11 \text{...put a and p in equation (1)}$$

i.e. $$6^{11-1} = 1 \bmod 11$$

$$6^{10} = [(6^8 \bmod 11) \times (6^2 \bmod 11)] \bmod 11$$

$$= [4 * 3] \bmod 11 = 12 \bmod 11 = 1$$

i.e. $$\mathbf{6^{10} \bmod 11 = 1}$$

Ex. 2.6.2

We can solve any large mod operation using this method. Solve 6^{12} mod 11.

Soln. :

According to Fermat's little theorem $a^{p-1} = 1 \bmod p$, 6^{12} mod 11

Hence a = 6, p = 11 if we put these values in above equation then will get,

$$6^{12-1} \bmod 11 = 6^{11} \bmod 11$$

$$= [(6^8 \bmod 11) \times (6^3 \bmod 11)] \bmod 11 = [4 \times 7] \bmod 11$$

$$= 28 \bmod 11 = 6$$

$$\therefore \mathbf{6^{11} \bmod 11 = 6}$$

Ex. 2.6.3

Define Fermat's little theorem find the result of :

(i) 3^{12} mod 11 (ii) 3^{10} mod 11

Soln.:

Refer section 2.6 for definition

(I) 3^{12} mod 11

According to Fermat's little theorem $a^{p-1} = 1 \bmod p$, 3^{12} mod 11

Hence a = 3 and p = 11, Put these values into above equation,

Scanned by CamScanner

$3^{12-1} \bmod 11 = 3^{11} \bmod 11$

$3^{11} \bmod 11 = [(3^8 \bmod 11) \times (3^3 \bmod 11)] \bmod 11 = [5 \times 5] \bmod 11$

$3^{11} \bmod 11 = (25 \bmod 11) = 3$

$\therefore$ **$3^{11} \bmod 11 = 3$**

(ii) $3^{10} \bmod 11$

According to Fermat's little theorem $a^{p-1} = 1 \bmod p$, $3^{10} \bmod 11$

Hence a = 3 and p = 11, Put these values into above equation,

$3^{10-1} \bmod 11 = 3^9 \bmod 11$

$3^9 \bmod 11 = [(3^5 \bmod 11) \times (3^4 \bmod 11)] \bmod 11 = [1 \times 4] \bmod 11$

$3^9 \bmod 11 = (4 \bmod 11) = 4$

$\therefore$ **$3^9 \bmod 11 = 4$**

Chapter Ends

□□□

Scanned by CamScanner

Module 2

CHAPTER 3

Symmetric Key Cryptography

Syllabus

Block cipher principles, block cipher modes of operation, DES, Double DES, Triple DES, Advanced Encryption Standard (AES), Stream Ciphers : RC5 algorithm.

Syllabus Topic : Block Cipher Principles

3.1 Block Cipher Principles

- Basically cryptographic algorithm is used for transformation of plaintext into cipher text.
- Stream cipher and Block cipher are main method of encrypting text using key and algorithm.
- For more details refer Section 1.10 of Chapter 1.

Syllabus Topic : Block Cipher Modes of Operation

3.2 Block Cipher Modes of Operation

Q. 3.2.1 Explain with examples the CBC and ECB modes of block ciphers. (Ref. secs. 3.2.2 and 3.2.1) **Dec. 16, 3 Marks**

Q. 3.2.2 Describe the block cipher modes of operation (ECB, CBC, CFB, OFB and CTR mode) with the help of block diagram.(Ref. sec. 3.2)

Q. 3.2.3 What are block cipher algorithmic modes ? Describe any two modes. (Ref. sec. 3.2)

Scanned by CamScanner

- The block cipher is basic building block for providing data security. In Block cipher rather that encrypting one bit at a time, block of bits is encrypted at one go.
- The Federal Information Processing Standard (FIPS) defines four modes of operation for block cipher that may be used in a wide variety of applications like symmetric key cryptographic algorithms (DES, AES etc). The modes specify how data will be encrypted and decrypted.
- The modes included in this standard are :

1. Electronic Codebook (ECB) mode
2. Cipher Block Chaining (CBC) mode
3. Cipher Feedback (CFB) mode and
4. Output Feedback (OFB) mode
5. Counter (CTR) Mode

3.2.1 Electronic Codebook (ECB) Mode

→ (MU - Dec. 16)

Q. 3.2.4 Describe ECB. (Ref. sec. 3.2.1)

- In Electronic Codebook (ECB) mode the given plaintext message is divided into blocks of 64 bits each and each 64-bits blocks get encrypted independently. The plaintext block produces ciphertext of same size (64-bits each).
- The given plaintext is encrypted using same key and transfers the encrypted data (ciphertext) to receiver.
- At the receiver end each block is decrypted independently using same key in order to produce original plaintext message of same size i.e. blocks of 64-bits each.
- The Electronic Codebook (ECB) mode encryption and decryption process is shown in Fig. 3.2.1 and Fig. 3.2.2.

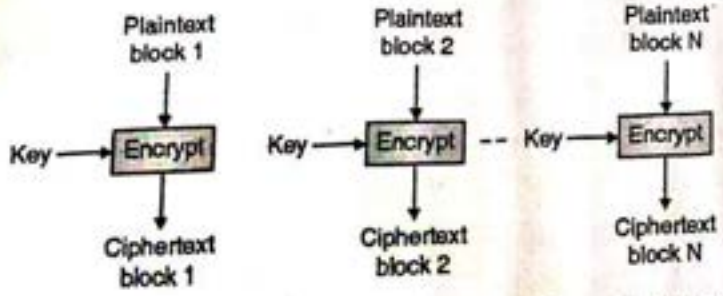


Fig. 3.2.1 : The Electronic Code Book (ECB) mode encryption process

Scanned by CamScanner

- The drawback of ECB mode is that for the occurrence of more than one plaintext block in the input generates the same ciphertext block in the output, which gives clue to the attacker or cryptanalyst.

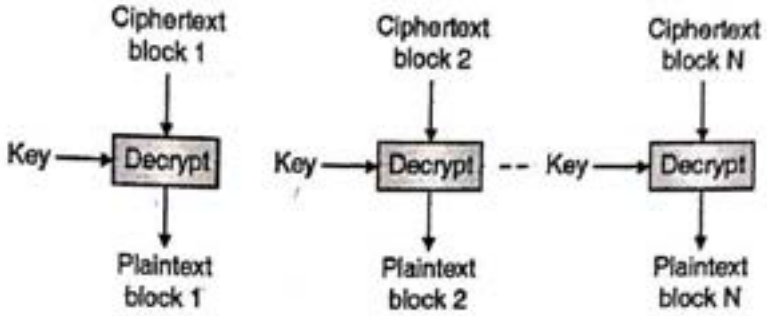


Fig. 3.2.2 : The Electronic Code Book (ECB) mode decryption process

- Only small messages can be encrypted using ECB mode of operation where the chances of repeating the same plaintext message are quite less.

3.2.2 Cipher Block Chaining (CBC) Mode

→ **(MU - Dec. 16)**

Q. 3.2.5 Describe CBC. (Ref. sec. 3.2.2)

- To overcome the problem of repetition and order independence in ECB even for repeated plaintext the Cipher Block Chaining (CBC) mode is used. In the cipher-block chaining mode, each block of plaintext is XORed with the previous ciphertext block before being encrypted.
- In CBC mode the first block of the message (plaintext block 1) is XORed with an Initialization Vector (IV) which is then encrypted using key k and produces ciphertext block 1. In next step each block of plaintext ((plaintext block 2) is XORed with the previous ciphertext block 1 before being encrypted and produces output i.e. ciphertext block 2. The procedure is continuing till all plain text block gets encrypted as shown in Fig. 3.2.3.
- Initialization Vector doesn't have special meaning it is simply used to make input message more complicated or unique.
- Different criteria for IV are fixed-size input value it should be random or pseudorandom. A good initialization vector should be unique and unpredictable. In all modes of operation plaintext blocks are represented by using P1, P2, P3,.....Pn and corresponding ciphertext blocks are represented by using C1, C2, C3,.....Cn.
- In case of cipher block chaining mode even if plaintext block repeats in the input, output of CBC mode yields totally different ciphertext blocks as shown in Fig. 3.2.3.

Scanned by CamScanner

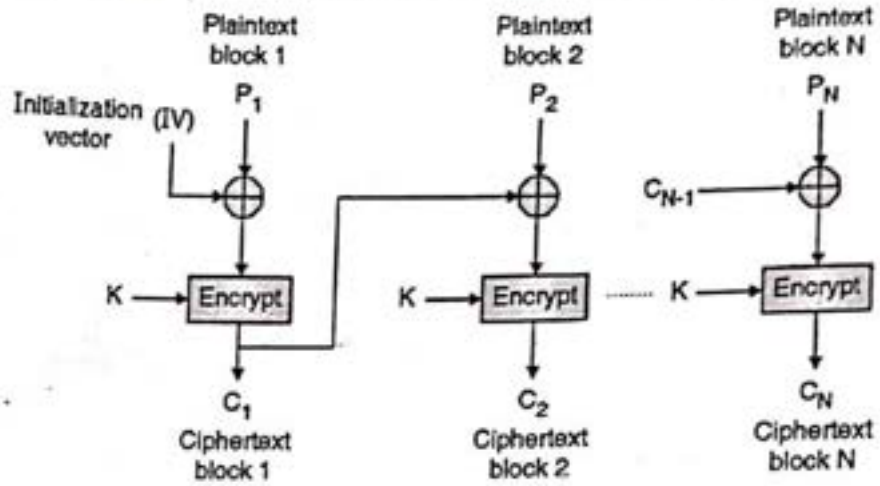


Fig. 3.2.3 : Cipher Block Chaining (CBC) mode encryption process

- CBC mode is applicable whenever large amounts of data need to be sent securely, provided that all data is available in advance (e.g. email, FTP, web etc.).
- In CBC decryption, cipher text block 1 get decrypted using same key used earlier during encryption process and the output of this step is then XOR with initialization vector (IV) and produces plaintext block 1.
- In next step the ciphertext block 2 is decrypted and its output is XOR with ciphertext block 1 which results plaintext block 2. Repeat the process for all ciphertext block in order to produce original plaintext blocks as shown in Fig. 3.2.4.

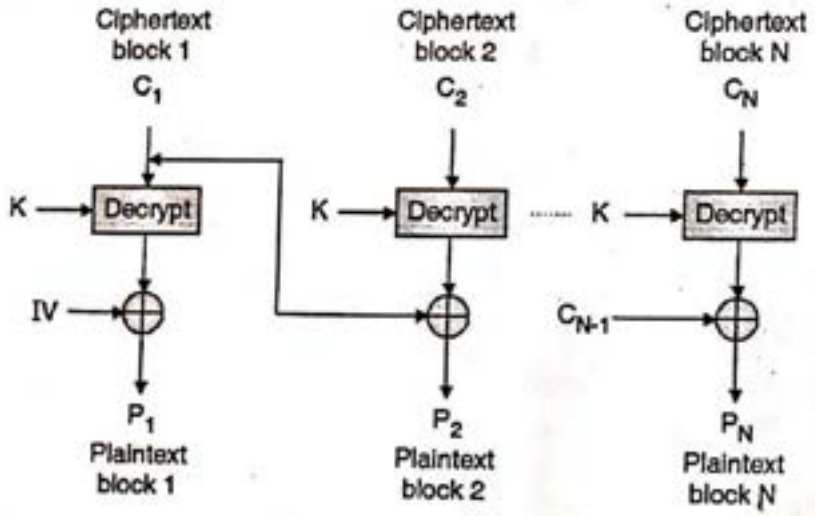


Fig. 3.2.4 : Cipher Block Chaining (CBC) mode decryption process

3.2.3 Cipher Feedback (CFB) Mode

Q. 3.2.6 Describe CFB. (Ref. sec. 3.2.3)

- CFB mode uses block cipher as stream cipher meaning is that data is encrypted in smaller units of block say 8-bits rather than predefined size of 64 bits.

Scanned by CamScanner

- CFB mode may be used as a stream cipher. In CFB encryption process 64 bits initialization vector is used which is kept in 64 bits of shift register.
- The IV (shift register) is then encrypted and produces 64-bits of ciphertext i.e. encrypted IV. Now the leftmost *S* bits (size of 8 bits) of the encrypted IV are XORed with the first S bits (size of 8 bits) of plaintext P_1 to produce the first S bits of ciphertext C_1, which is then transmitted to next step.
- In next step contents of the 64 bit shift register are shifted left by S bits and C_1 is placed in the rightmost S bits of the shift register and which again undergoes to encryption process as shown in Fig. 3.2.5.
- This process continues until all plaintext units have been encrypted. Here same key K is used during encryption and decryption process.

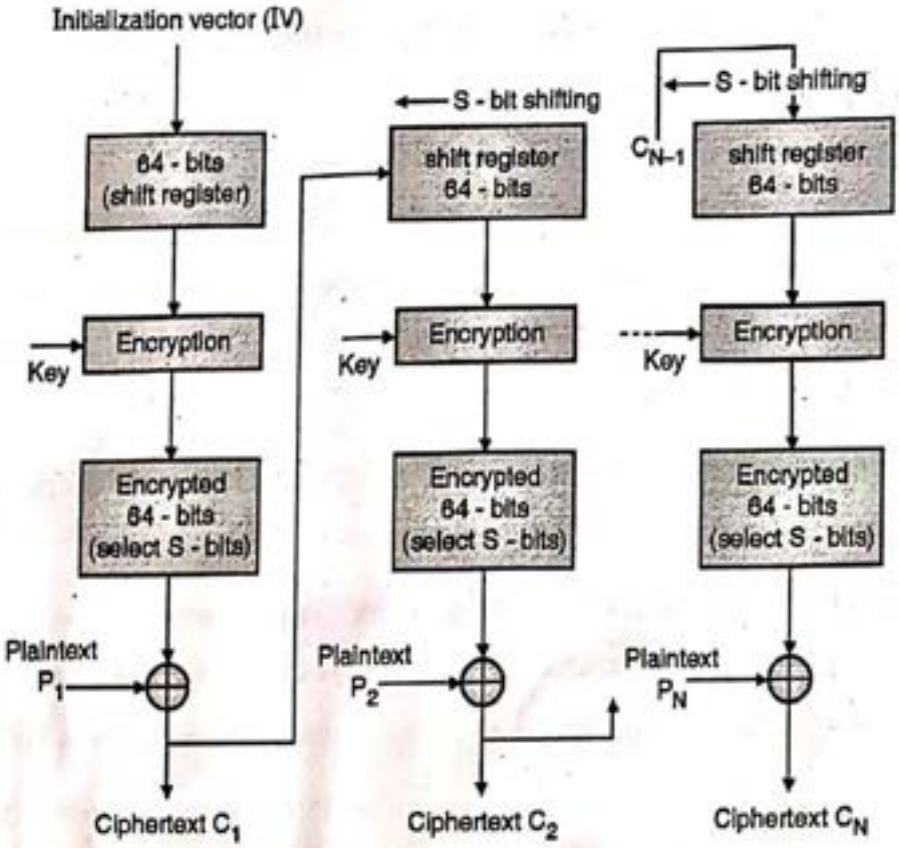


Fig. 3.2.5 : Cipher Feedback (CFB) mode encryption process

- Decryption of CBC mode reverse of CBC encryption, the same technique is used, except that the output of encryption process is XOR with the received ciphertext block to produce the original plaintext block as shown in Fig. 3.2.6.

Scanned by CamScanner

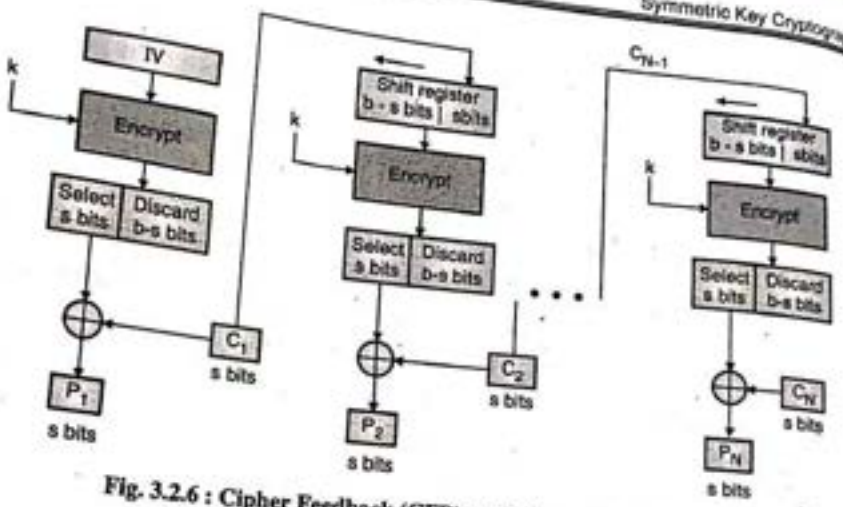


Fig. 3.2.6 : Cipher Feedback (CFB) mode Decryption Process

3.2.4 Output Feedback (OFB) Mode

Q. 3.2.7 Describe OFB. (Ref. sec. 3.2.4)

- In CFB encryption process 64 bits initialization vector is used which is kept in 64 bits of shift register then selected the most significant *s* bits from encrypted IV and XOR with plain text P_1 to produce ciphertext C_1.
- Output of encryption process is placed in least significant s bits of shift register (left shift) of next stage and repeat the process until all units encrypted.
- In case of output feedback mode (OFB), difference is that output of encryption process O_1 instead of generating cipher text C_1 is directly placed in next stage of shift register without XOR operation to avoid bit errors during transmission as shown in Fig. 3.2.7.
- In OFB mode, if there is a small error in individual bits, it remains an error in individual bits which do not corrupt the whole encrypted message (to avoid bit errors during transmission) which is the biggest advantages of OFB mode over all other mode.
- In this case Initialization Vector is extracted from a double length encryption key.

Scanned by CamScanner

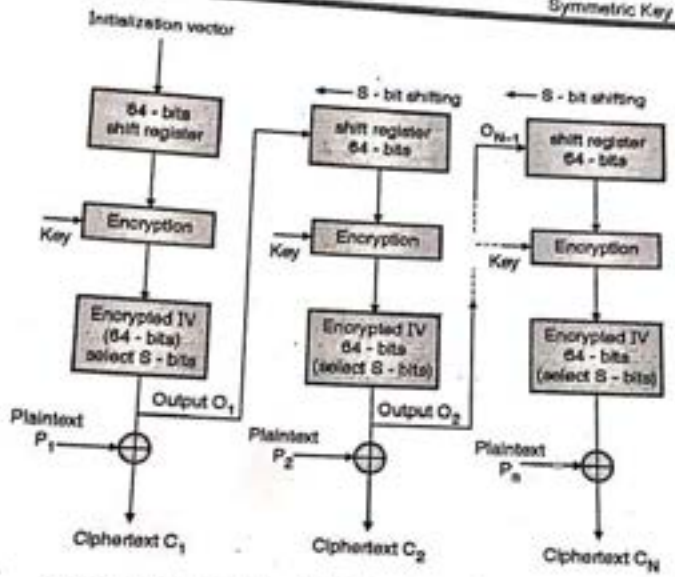


Fig. 3.2.7 : Output feedback (OFB) mode encryption process

- Decryption process of OFB is reverse of encryption process as shown in Fig. 3.2.8.

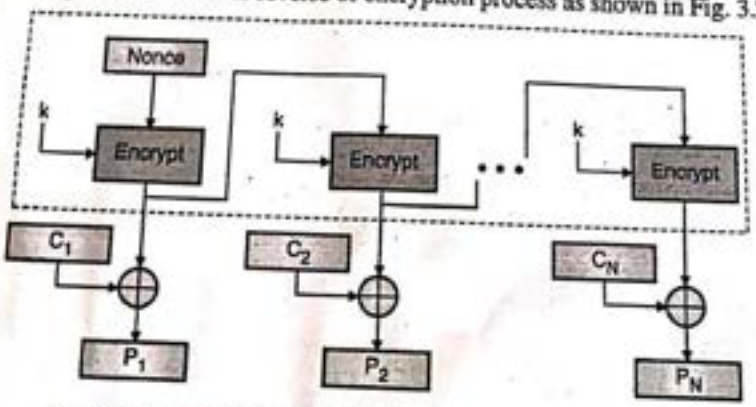


Fig. 3.2.8 : Output feedback (OFB) mode decryption process

3.2.5 Counter (CTR) Mode

Q. 3.2.8 Compare ECB and counter mode of operation. (Ref. sec. 3.2.5)

Q. 3.2.9 Explain Counter mode of operation. (Ref. sec. 3.2.5)

Scanned by CamScanner

- This mode is similar to OFB with the difference is that it uses counters or sequence numbers as input to the algorithm. We put a constant value as initial value of counter which will be of same size as that of a plaintext block.
- In counter mode, the counter is encrypted and then XOred with the plaintext to ciphertext .There is no chaining process is done as shown in Fig. 3.2.9.

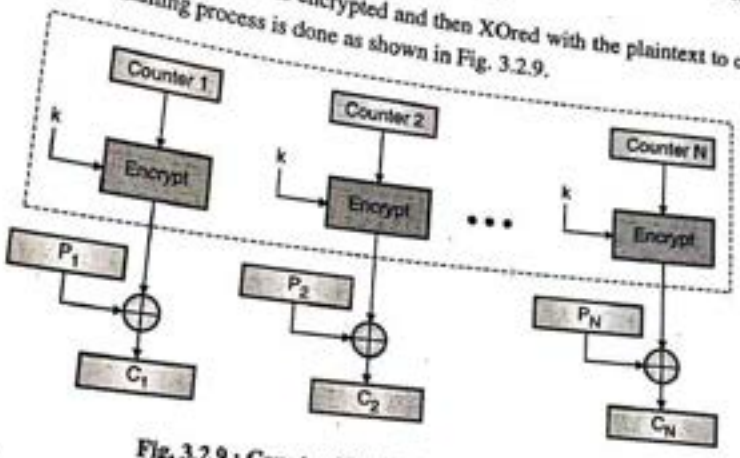


Fig. 3.2.9 : Counter (CTR) Encryption Process

- During decryption process same counter is used which was encrypted earlier and later XORed with ciphertext to get the original plain text plaintext as shown in Fig. 3.2.10.

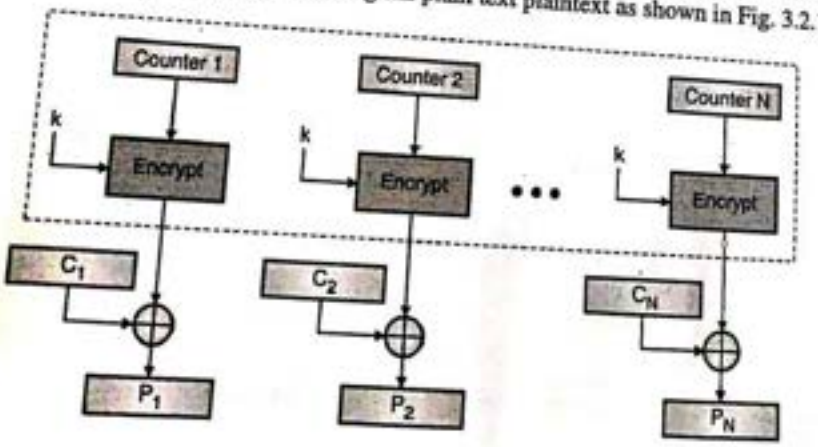


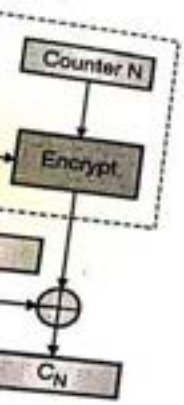
Fig. 3.2.10 : Counter (CTR) Decryption Process

- The advantage of counter mode is that it can make execution speed up which can help in multiprocessing system.

Scanned by CamScanner

uses counters or sequence as initial value of counter

the plaintext to ciphertext



ed earlier and later in Fig. 3.2.10.

can help in

3.2.6 Algorithm Mode Details and Usage

Table 3.2.1 : Summarization of algorithm modes

Mode	Details	Usage
Electronic code book (ECB)	Same key is use to encrypt and decrypt 64 bit at a time.	A single value is transmitted in secure fashion.
Cipher Block Chaining (CBC)	Ciphertext of previous step and plaintext in next step are XORed.	It is used for authentication purpose.
Cipher Feedback (CFB)	K number of random from ciphertext bits of previous step and k bit from plaintext of the next step are XOred.	Encrypted stream of data is transmitted for authentication
Output Feedback (OFB)	It is similar to CFB, only difference is that input to encryption step is preceding DES output.	For transmission of encrypted stream of data.
Counter (CTR)	Both counter and plaintext block are encrypted together.	It is used in the applications which need high speed of transmission.

Syllabus Topic : DES

3.3 Data Encryption Standard (DES)

→ **(MU - Dec. 15, May 16, May 17, May 18)**

Q. 3.3.1 Explain working of DES detailing the Fiestel structure. (Ref. sec. 3.3) **Dec. 15, 10 Marks**

Q. 3.3.2 Explain working of DES. (Ref. sec. 3.3) **May 16, 10 Marks**

Q. 3.3.3 Explain DES, detailing the Fiestel structure and S-block design. (Ref. sec. 3.3) **May 17, 10 Marks**

Q. 3.3.4 What is the purpose of S-boxes in DES? Explain the avalanche effect? (Ref. sec. 3.3) **May 18, 5 Marks**

Scanned by CamScanner

- DES is block cipher published by National Institute of Standards and Technology (NIST).
- DES was originally developed by an IBM team formed in early 1970 in response to customer request for a method to secure data.
- In 1973, NIST published a request for proposals for a symmetric key cryptography. A proposal from IBM, a change in the project called Lucifer, was accepted as DES.
- DES was published in federal register in 1975 as a draft of the Federal Information Processing Standard (FIPS).This draft was rejected because of small key length (56-bits) which could make ciphertext weaker to many attacks.
- Later on IBM developer mentioned that internal architecture of DES was designed to prevent cryptanalysis.
- Finally NIST published DES in January 1977 and defines DES as the standard for use in unclassified applications.
- The day onwards DES has been proven most widely used symmetric key cryptographic algorithm.

3.3.1 Conceptual View of DES

- Data encryption standard takes 64-bit plaintext as an input and creates 64-bit Ciphertext i.e. it encrypts data in block of size 64-bits per block.
- Divide plaintext message into 64-bit block each.

OR

- The given plaintext message is divided into size 64-bits block each and encrypted using 56-bit key at the initial level. Fig. 3.3.1 shows conceptual view of DES.

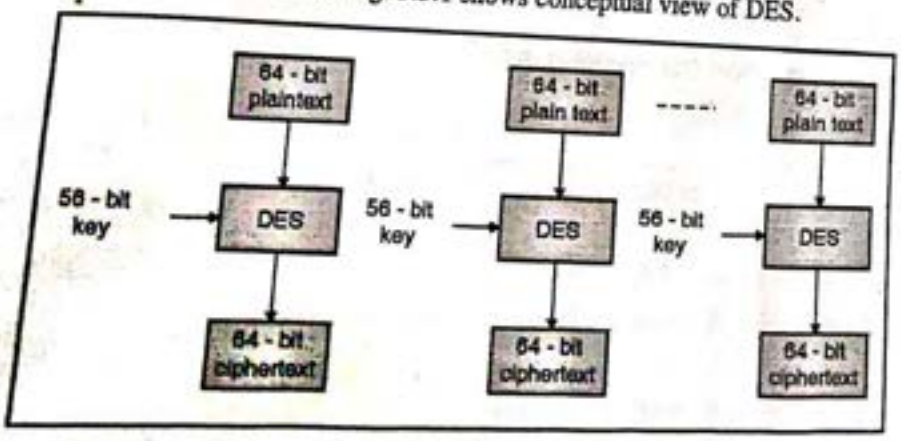


Fig. 3.3.1 : Conceptual view of DES

- At the decryption side, DES
56-bit key.

3.3.2 Detail Steps of DES

- The principle of DES is very s
each, which is initial permutati
- After initial permutation on 6
called left plaintext and right p
- The left plaintext and right pla
with 16 different keys for each
right plaintext gets combined
blocks.
- The result of final permutation

Scanned by CamScanner

- At the decryption side, DES takes 64-bit ciphertext and creates 64-bit plaintext and 56-bit key.

3.3.2 Detail Steps of DES

- The principle of DES is very simple. Divide plaintext message into block of size 64-bits each, which is initial permutation.
- After initial permutation on 64-bit block, the block is divided into two halves of 32-bit called left plaintext and right plaintext.
- The left plaintext and right plaintext goes through 16 rounds of encryption process along with 16 different keys for each rounds. 16 rounds of encryption process left plaintext and right plaintext gets combined and final permutation is performed on these combined blocks.
- The result of final permutation produces 64-bit of ciphertext as shown in Fig. 3.3.2.

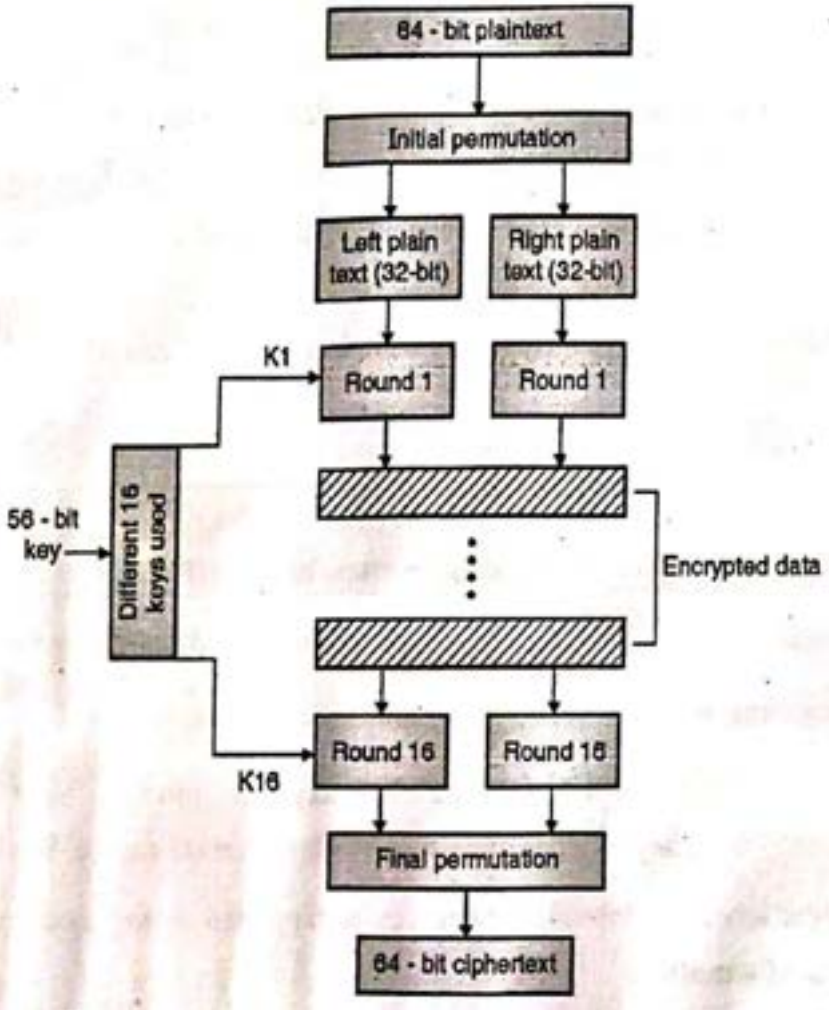


Fig. 3.3.2 : Detail Steps in DES

Scanned by CamScanner

3.3.3 Initial Permutation (IP)

Q. 3.3.5 Explain initial permutation steps in DES. (Ref. sec. 3.3.3)

- Initial permutation is the process of rearranging or shuffling each bit of original plaintext block with any other random bit of same plaintext message block.
- **For example :** First bit of original plaintext block replace with 48th bit of original plaintext block, the 2nd bit replaces with 57th bit of original plaintext message shown in Table 3.3.1.

Table 3.3.1 : Initial permutation

Plaintext block (64 bits)

48	57	59	63	8	9	43	64
47	42	10	6	4	11	13	18
20	5	16	19	32	1	25	27
2	3	23	29	30	31	34	7

- This process called jugglery of bit position of plaintext block which is applied to all original plaintext blocks in a sequence.
- After initial permutation the 64-bit plaintext block get divided into two halves LPT (32-bit) and RPT (32-bit).
- Now 16 rounds of encryption process were completed on LPT and RPT.

3.3.4 Rounds

Q. 3.3.6 Explain 16 Rounds of DES in details. (Ref. sec. 3.3.4)

Q. 3.3.7 Explain permutation and substitution steps in DES. (Ref. sec. 3.3.4)

Before discussing about DES rounds let us know about key discarding process.

Step 1 : Key discarding process

- We know that 56-bit key is used during encryption process. Here 56-bit key is transformed into 48-bit key by discarding every 8th bit of initial key i.e. 8th, 16th, 32nd ...
- From this 56-bit key a different 48-bit sub-key is generated during each round the process called **key transformation**.

Scanned by CamScanner

Step 2 : Expansion permutation process

- As shown in Fig. 3.3.3 general steps of DES, we had two 32-bit plaintext called as Left Plaintext (LPT) and 32-bit Right Plaintext (RPT). In step 2 Right Plaintext is expanded from 32-bits to 48-bits the process called as **expansion permutation.**

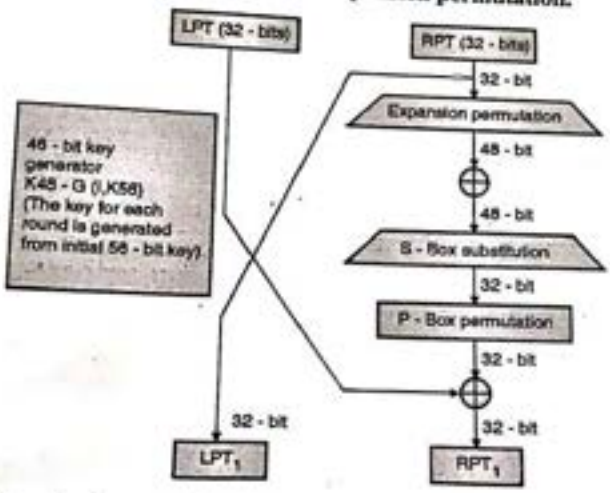


Fig. 3.3.3 : Expansion Permutation (Details steps of Single Round of DES Algorithm)

- During this process bits are permuted hence called as expansion permutation shown in Fig. 3.3.3.
- From Fig. 3.3.3 it is clear that 32-bits of RPT get expanded to 48-bit which is XORed with 48-bit key generated during key discarding process. The resulting output of XOR operation is given to next step called as S-box substitution.

Step 3 : S-box substitution

- S-box substitution is the process which accepts 48-bit key and expanded right plaintext of 48-bit which get XORed and produces 32-bit output as shown in Fig. 3.3.4.
- S-box performs just jugglery on each bit positions and produces compressed bits.
- The substitution performed using 8 substitution boxes the reason called as S-boxes. The input for each S-box is 6-bit which produces output of 4-bit each. The output of S-box is then combined to form 32-bit block.

Scanned by CamScanner

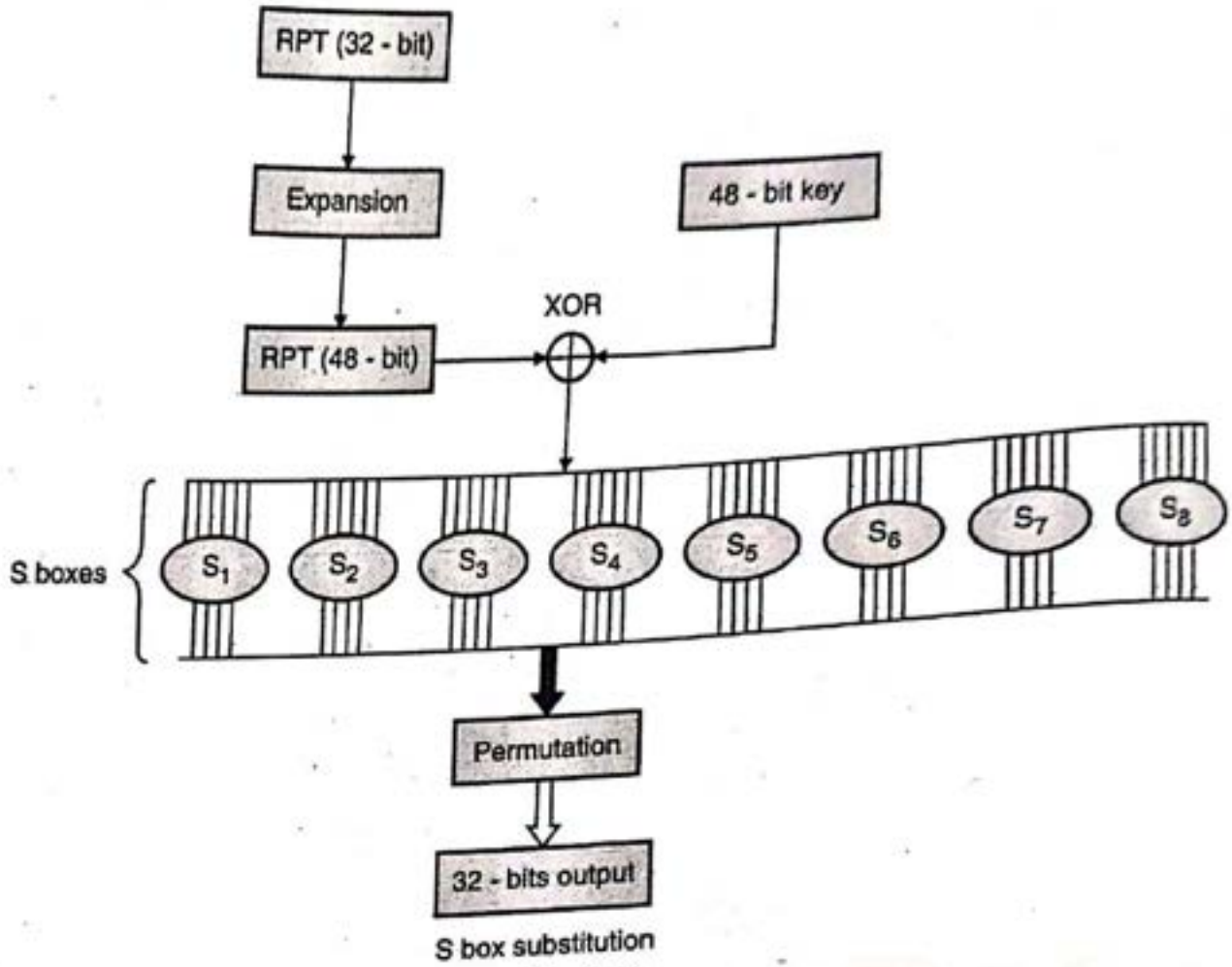


Fig. 3.3.4

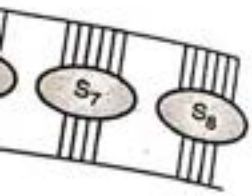
Step 4 : P-box permutation

- We know that the 32-bit output from 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.
- The output of S-box is input to P-box in which 32-bits are permuted using straight forward permutation mechanism i.e. replacement of each bit with another bit in the specified table called as P-box permutation.

Step 5 : XOR and swapping

- Till this step we have performed all operations on 32-bit right plaintext only left plaintext was not touched yet. Recall step 2 as given in Fig. 3.3.4 detail steps of each round, output produced by P-box permutation get XOR with the left plaintext (32-bit) to produce the new 32-bit right plaintext.
- It is quite obvious that output of XOR operation becomes new right plaintext and the old right plaintext becomes new left plaintext, the complete process is called as XOR and swapping operations.

Scanned by CamScanner



so that on the next
possible.

g straight forward
he specified table

y left plaintext
round, output
o produce the

t and the old
s XOR and

3.3.5 Final Permutation

- The last operation a data encryption standard is a permutation of 32-bit LPT and 32-bit RPT produced from XOR and swapping process. Final permutation was performed after successful completion of 16 rounds.
- **For example :** 7th bit of the input bit takes the position of second bit and so on. 40th input bit takes the position of 4th output bit and so on. Table 3.3.2 shows final permutation.

Table 3.3.2 : Final permutation

16	07	20	40	21	29	64	32	39	8	47	15	55	23	63	31
34	38	6	46	54	14	22	60	30	37	45	5	13	53	22	21
29	36	3	4	44	12	20	52	35	28	26	11	10	9	19	27
17	57	25	50	18	2	1	33	56	53	32	44	16	24	15	59

- Inverse of initial permutation is also called final permutation. This produces 64-bit encrypted block. In this way all the plaintext get encrypted by performing broad level steps on it in order to produced encrypted ciphertext.

3.3.6 Strength of DES

1. The DES encryption technique ideally suited for implementation of hardware (bit shifts, lookups etc.).
2. It makes Brute force attack impractical as DES uses 56-bit keys so that there are 2^{56} possible key combinations which make Brute Force attack impractical.
3. Dedicated hardware could run DES at 200 M byte/s.
4. Technique well suited for voice, video encryption.
5. DES uses 56-bit keys so that there are 2^{56} possible key combinations which is roughly equal to 7.2×10^{16} keys required to break DES cipher.
6. A machine performing one DES encryption per microsecond would take more than a thousand year to break the cipher.
7. If a small change in either plaintext or the key, the ciphertext should change markedly.

Scanned by CamScanner

3.3.7 Weakness In DES

1. Trying all possible combination of 2^{56} possible keys is not that much hard these days.
2. In an exhaustive search known plaintext attack, the cryptanalyst will obtain the solution after 2^{55} i.e. 3.6029×10^{16} trials on an average.
3. If you spend \$25 K you can build DES password crackers that will successes in few hours.
4. Two chosen input to an S-box can create the same output.
5. The purpose of initial and final permutation is not clear.

Syllabus Topic : Double DES

3.3.8 Double DES

Q. 3.3.8 Explain Double DES. (Ref. sec. 3.3.8)

Q. 3.3.9 Write short note on : Multiple DES. (Ref. sec. 3.3.8)

- Double performs the same operation as DES only difference is that double DES use two keys K1 and K2.
- First it perform encryption on plaintext which is encrypted using K1 obtains first ciphertext again this ciphertext is encrypted by using another key called K2 and converted into final ciphertext.
- Mathematically Double DES is represented as

$$Pt \Rightarrow EK1(Pt) \Rightarrow TEMP = EK1(Pt) \Rightarrow EK2(E(K1(P))) \Rightarrow Cp = EK2(E(K1(Pt)))$$

Where

Pt = Plaintext

$EK1(Pt)$ = Encrypted plaintext with Key K1

$TEMP$ = $EK1(Pt)$ = Temporary Variable to store results

$EK2(E(K1(P)))$ = Encrypted Results of first step using K2

Cp = Final Ciphertext.

- Decryption of Double DES is reverse of Encryption. Whatever the ciphertext obtained after double DES encryption process get decrypted using K2 and obtain the first ciphertext, the result of previous step (ciphertext) decrypted using K1 which yields the original plaintext. To decrypt the cipher text Cp and obtain the plain text Pt Double DES need to perform the following operation.

$$Pt = DK2\,(DK1(Cp))$$

Scanned by CamScanner

Syllabus Topic : Triple DES

3.3.9 Triple DES

Q. 3.3.10 Explain Triple DES.(Ref. sec. 3.3.9)
Q. 3.3.11 Write short note on : Multiple DES.(Ref. sec. 3.3.9)

- Triple DES performs the same operation as double DES only difference is that triple DES uses three keys K1, K2 and K3 while encrypting plaintext.
- First it perform encryption on plaintext which is encrypted using K1 obtains first ciphertext again this ciphertext is encrypted by using another key called K2 which obtains the second ciphertext which is again encrypted using K3 and converted into final ciphertext Cp.
- Mathematically, Double DES is represented as,

$$Pt \Rightarrow EK1(Pt) \Rightarrow TEMP = EK1(Pt) \Rightarrow EK2(E(K1(P))) \Rightarrow EK3\ (EK2(EK1(Pt)))$$

$$\Rightarrow Cp = EK3\ (EK2(EK1(Pt)))$$

Where
Pt = Plaintext
EK1(Pt) = Encrypted plaintext with Key K1
TEMP = EK1(Pt) = Temporary Variable to store results
EK2(E(K1(P))) = Encrypted Results of first ciphertext using K2
EK3 (EK2(EK1(Pt))) = Encrypted Results of second step using K2
Cp = EK3 (EK2(EK1(Pt))) Final ciphertext encrypted using K1, K2 and K3

- Decryption of Triple DES is reverse of Encryption.
- The final ciphertext obtained after Triple DES encryption process get decrypted using K3 which results second ciphertext, second ciphertext decrypted using K2 which results first ciphertext, first ciphertext again decrypted using K1 which generate the original plaintext Pt.
- To decrypt the cipher text Cp and obtain the plain text Pt, we need to perform following operation.

$$Pt = DK3(DK2(DK1(Cp)))$$

Scanned by CamScanner

Syllabus Topic : Advance Encryption Standard (AES)

3.4 Advance Encryption Standard (AES)

Q. 3.4.1 Write working of AES algorithm in detail. (Ref. sec. 3.4)

Q. 3.4.2 Draw AES block diagram and explain the steps in detail. (Ref. sec. 3.4)

Q. 3.4.3 How AES Encryption does not solve any integrity problem ? If not what will be the solution. (Ref. sec. 3.4)

3.4.1 Introduction to AES

- The Advanced Encryption Standard (AES Algorithm) is a Symmetric key cryptographic algorithm published by National Institute for Standards and Technology (NIST) in December 2001.
- The algorithm was proposed by Rijndael the reason also called Rijndael encryption algorithm. Advance encryption standard is a replacement of data encryption standard.

3.4.2 Silent Features of AES

- The AES algorithm is a subset of the Rijndael algorithm and is a block cipher, meaning that it operates on an *input block* of data of a known size and outputs a block of data which is the same size.
- An *input key* is also required as input to the AES algorithm. It allows the data length (plain text size) of 128, 192 and 256 bits, and supporting three different key lengths, 128, 192, and 256 bits.
- The AES algorithm is a symmetric key algorithm which means the same key is used to both encrypt and decrypt a message.
- Also, the cipher text produced by the AES algorithm is the same size as the plain text message.
- AES consists of multiple rounds for processing different key bits like 10 rounds for processing 128 - bit keys, 12 rounds for processing 192 - bit keys, and 14 rounds for processing 256 - bit keys.

Scanned by CamScanner

3.4.3 AES Encryption and Decryption Process

- The plaintext given is divided into 128-bit block as consisting of a 4 × 4 matrix of bytes.
- Therefore, the first four bytes of a 128-bit input block occupy the first column in the 4 × 4 matrix of bytes. The next four bytes occupy the second column, and so on.
- AES operates on a 4 × 4 column-major order matrix of bytes; called as ***state array*** shown in Fig. 3.4.3. AES also has the notion of a word. A word consists of four bytes that is 32 bits. The overall structure of AES encryption and decryption process is shown in Fig. 3.4.1.

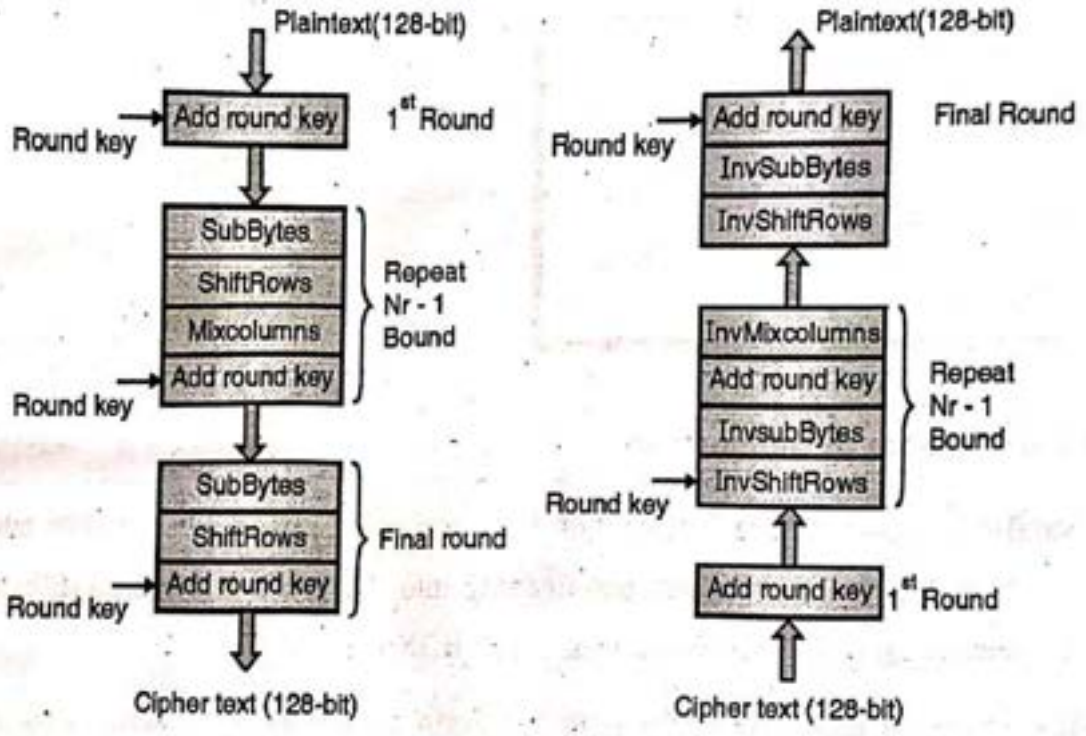


Fig. 3.4.1 : AES Encryption and Decryption process

- The numbers of rounds are 10, for the case when the encryption key is 128 bit long. (As mentioned earlier, the numbers of rounds are 12 when the key is 192 bits and 14 when the key is 256.) Before any round-based processing for encryption can begin each byte of the state (plaintext) is combined with the round key using bitwise XOR operation. Nr stands for number of rounds.
- AES divide plaintext into 16 byte (128-bit) blocks, and treats each block as a 4 × 4 State arrays as shown in Fig. 3.4.3. It then performs four operations in each round consists of several processing steps like substitution step, a row-wise permutation step, a column-wise mixing step, and the addition of the round key. Except for the last round in each case,

Scanned by CamScanner

all other rounds are identical. Final Round doesn't have (MixColumns) it includes only SubBytes, ShiftRows and AddRoundKey.

- The process of transforming the cipher text back into the original plaintext using same encryption key is called as decryption process of AES, during decryption process the set of rounds are reversed.

3.4.4 Detail Steps for AES Encryption

For encryption, each round consists of the following four steps :

(1) SubBytes

(2) ShiftRows

(3) MixColumns, and

(4) AddRoundKey

1. The SubByte step/Substitute byte

- SubBytes() consists of replacement of each byte using a fixed S-box lookup table as shown in Fig. 3.4.2 to achieve non-linearity into the 4×4 state array (16 byte). It performs roughly the same function as the S- BOX in DES.
- It operates on each byte in the state and performs a non-linear substitution in the Galios Filed GF (2^8) field, which is what makes AES a non-linear cryptographic system.
- Fig. 3.4.3 shows the state transformation using SubBytes techniques and if apply reverse called as InvSubBytes transformation which will create original values.
- For every same two byte value the resulting transformation is also same. It also shows that the InvSubBytes transformation creates the original one.
- Note that if the two bytes have the same values, their transformation is also the same.
- The corresponding substitution step used during decryption is called InvSubBytes.

Scanned by CamScanner

X \ Y	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	1	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	F1	71	D8	31	15
3	4	C7	23	C3	18	96	5	9a	7	12	80	E2	Eb	27	B2	75
4	9	83	2c	1a	1b	6e	5a	A0	52	3b	D6	B3	29	E3	2f	84
5	53	D1	0	Ed	20	Fc	B1	5b	6a	Cb	Be	39	4a	4c	58	Cf
6	D0	Ef	Aa	Fb	43	4d	33	85	45	F9	2	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	Ec	5f	97	44	17	C4	A7	7e	3d	64	5d	19	73
9	60	81	4f	Dc	22	2a	90	88	46	Ee	B8	14	Be	5e	0b	Db
a	E0	32	3a	0a	49	6	24	5c	C2	D3	Ac	62	91	95	E4	79
b	E7	C8	37	6d	8b	D5	4e	A9	6c	56	F4	Ea	65	7a	Ae	8
c	Ba	78	25	2e	1c	A6	B4	C6	E8	Dd	74	1f	4b	Bd	8b	8a
d	70	3e	B5	66	48	3	F6	0e	61	35	57	B9	86	C1	1d	9e
e	E1	F8	98	11	69	D9	8e	94	9b	1e	87	E9	Ce	55	28	Df
f	8c	A1	89	0d	bf	E6	42	68	41	99	2d	0f	B0	54	bb	16

Fig. 3.4.2 : S-Box Lookup table for SubBytes

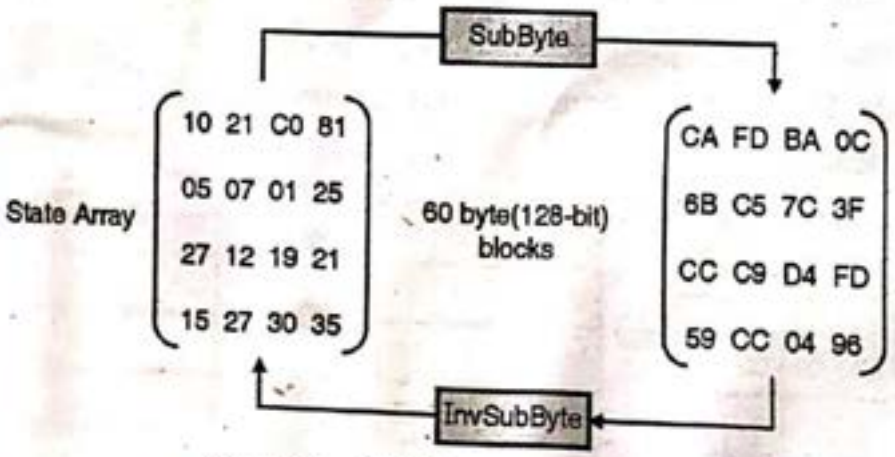


Fig. 3.4.3 : SubByte transformation

Scanned by CamScanner

2. ShiftRows

- The output of the SubByte transformation is input to the ShiftRows transformation which consists of rotation of each byte of the state array in the order of a row of data matrix (rotation of row byte positions are done in this step).
- Each byte of the first row remains unchanged. Each byte of the second row is rotate over one byte to the left position. Similarly the third and fourth rows are also rotatedleft by two and three position as shown in Fig. 3.4.4.
- The corresponding transformation during decryption process is called Inverse shift row transformation (InvShiftRows).

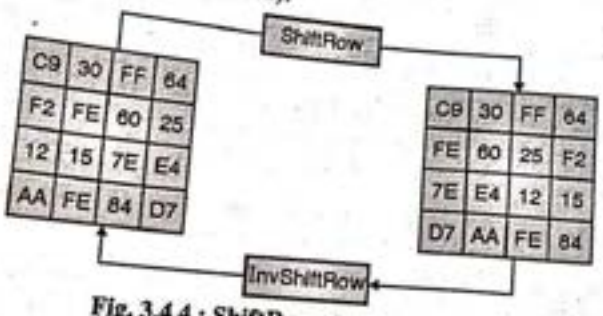


Fig. 3.4.4 : ShiftRows transformation

3. MixColumns

- MixColumns performs operation on the state array obtained from ShiftRows column-by-column and each column is multiplied with row of a fixed matrix. This step takes four bytes as an input and produces outputs of four bytes (each input byte affects the output bytes).
- The four numbers of state arrays of first column are modulo multiplied in Rijandeal's Galios Filed (GF) by a given matrix as shown in Fig. 3.4.5. In AES MixColumn step along with ShiftRows are primary source for providing complete diffusion to the cipher produced.

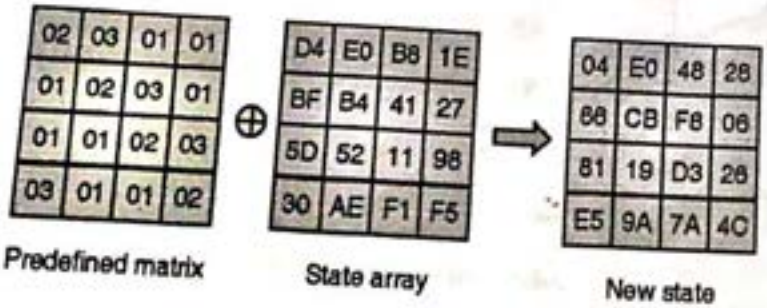


Fig. 3.4.5 : MixColumns transformation

Scanned by CamScanner

- From Fig. 3.4.5 on the left hand side, the row of the leftmost matrix is multiply with column of state array (XOR operations) which produces the new state. Perform the same operation on all columns which provides diffusion (mixing data within columns).
- The 4 bytes of each column in the State are treated as a 4-byte number and transformed to another 4-byte number via finite field mathematics (modulo multiplied in Rijndael's Galois Filed by a given matrix) as shown. MixColumns step is primary source of diffusion in AES.

4. AddRoundKey

- In the AddRoundKey step, the Round key one generated using Rijndael's key schedule is combined with the new state obtained from MixColumns transformation state.
- The round key is added by combining each byte of the state array using bitwise XOR operations. The actual 'encryption' is performed in the AddRoundKey() function, when each byte of state array is XORed with the round key to produce final cipher text as shown in Fig. 3.4.6.

04	E0	48	28
66	CB	F8	06
81	19	D3	26
E5	9A	7A	4C

XOR

A0	88	23	2A
FA	54	A3	6C
FE	2C	39	76
17	B1	39	05

=

A4	68	6B	02
9C	9F	5B	6A
7F	35	EA	50
F2	2B	43	49

Fig. 3.4.6 : AddRoundKey

- The same process of AddRoundKey is applied for nine rounds i.e Repeat SubByte, ShiftRows, MixColumns step and XOR with Round key 9 more times.
- Except for the last round in each case, all other rounds are identical. Final Round doesn't have MixColumns step it includes only SubBytes, ShiftRows and RoundKey.
- Finally an output cipher text will obtain after performing detail steps of AES. A set of reverse rounds are applied (i.e. InvShiftRows, InvSubBytes, AddRoundKey and InvMixcolumns) to transform cipher text back into the original plaintext using the same encryption key called Decryption process of AES as shown in Fig. 3.4.1.

Scanned by CamScanner

3.4.5 AES Decryption

Decryption occurs through the function AddRoundKey ( ), plus the inverse AES functions InvShiftRows (), InvSubBytes (), InvMixColumns () and AddRoundKey () does not require an inverse function, as it simply XORs the state with the subkey (XOR encrypts when applied once, and decrypts when applied again).

3.4.5(A) Difference between Data Encryption Standard (DES) and Advance Data Encryption Standard (AES)

Q. 3.4.4 Compare AES and DES. (Ref. sec. 3.4.5(A))

Sr. No.	DES	AES
1.	Data encryption standard takes 64-bit plaintext as a input and creates 64-bit Ciphertext i.e. it encrypts data in block of size 64-bits per block.	It allows the data length (plain text size) of 128, 192 and 256 bits.
2.	In DES plaintext message is divided into size 64-bits block each and encrypted using 56-bit key at the initial level.	AES divide plaintext into 16 byte (128-bit) blocks, and treats each block as a 4 ×4 State array and supporting three different key lengths, 128, 192, and 256 bits.
3.	The left plaintext and right plaintext goes through 16 rounds of encryption process along with 16 different keys for each rounds.	The number of rounds are 10, is for the case when the encryption key is 128 bit long. (As mentioned earlier, the number of rounds is 12 when the key is 192 bits and 14 when the key is 256.)
4.	DES uses 56-bit keys so that there are 2^{56} possible key combinations which is roughly equal to 7.2×10^{16} keys required to break DES cipher.	AES is stronger than DES because of key size vary from round to round.
5.	Different versions of DES are double DES and triple DES is added.	AES doesn't have any future version.
6.	DES doesn't use Mix Column, Shift Rows method during encryption and decryption process.	AES uses Mix Column, Shift Rows method during encryption and decryption process
7.	DES, double DES and Triple DES (168-bit key) are vulnerable to brute force attacks.	AES also are vulnerable to brute force attacks.

Scanned by CamScanner

Ex. 3.4.1

AES encryption does not solve an integrity problem? What is the solution ?

Soln. :

We aware that AES encrypt the plain text by applying different permutation and combinations, shift rows, mixing columns, it plays jugglery with the input bits and uses 128 bit key during encryption process. What if cryptanalyst stores the results of every previous steps into some temporary variable and if he apply the same technique used in AES i.e. all possible permutation and combinations, shifting rows and mixing columns cryptanalyst may clue that what will be the original message. So it violates the principle of integrity, meaning is that when sender sending plaintext messages to receiver and performing the AES steps still cryptanalyst able to modify the contents of original message, the get contents of original message get modified before it reaches to intended receipts then we say that AES encryption does not solve integrity problems. Only solution is that use AES tool in efficient & secure way.

Syllabus Topic : RC5 Algorithm

3.5 RC5 Algorithm

RC5 is a symmetric key block encryption algorithm designed by Ron Rivest in 1994. It is notable for being simple, fast (on account of using only primitive computer operations like XOR, shift, etc.) and consumes less memory.

Example

Key : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Plain Text : 00000000 00000000

Cipher Text : EEDBA521 6D8F4B15

RC5 is a block cipher and addresses two word blocks at a time. Depending on input plain text block size, number of rounds and key size, various instances of RC5 can be defined and each instance is denoted as RC5 - w/r/b where w = word size in bits, r = number of rounds and b = key size in bytes.

Scanned by CamScanner

Allowed values are :

Parameter	Possible Value
block/word size (bits)	16, 32, 64
Number of Rounds	0 – 255
Key Size (bytes)	0 – 255

Note : Since at a time, RC5 uses 2 word blocks, the plain text block size can be 32, 64 or 128 bits.

Notation used in the algorithm

Symbol	Operation
x <<< y	Cyclic left shift of x by y bits
+	Two's complement addition of words where addition is modulo
^	Bit wise Exclusive-OR

Step 1 : Initialization of constants P and Q.

– RC5 makes use of 2 magic constants P and Q whose value is defined by the word size w.

Word Size (Bits)	P (Hexadecimal)	Q (Hexadecimal)
16	b7e1	9e37
32	b7e15163	9e3779b9
64	b7e151628aed2a6b	9e3779b97f4a7c15

– For any other word size, P and Q can be determined as :

$P = Odd((e-2)^{2^w})$

$Q = Odd((\phi-2)^{2^w})$

– Here, Odd(x) is the odd integer nearest to x, e is the base of natural logarithms and ϕ is the golden ratio.

Step 2 : Converting secret key K from bytes to words.

– Secret key K of size b bytes is used to initialize array L consisting of c words where $c = b/u$, $u = w/8$ and w = word size used for that particular instance of RC5.

Scanned by CamScanner

- For example, if we choose w = 32 bits and Key k is of size 96 bytes then, u = 32/8 = 4, c = b/u = 96/4 = 24.
- L is pre initialized to 0 value before adding secret key K to it.

```
for i=b-1 to 0
L[i/u] = (L[u/i] <<< 8) + K[i]
```

Step - 3 : Initializing sub-key S.

- Sub-key S of size t = 2(r+1) is initialized using magic constants P and Q.

```
S[0] = P
for i = 1 to 2(r+1)-1
  S[i] = S[i-1] + Q)
```

Step 4 : Sub-key mixing.

- The RC5 encryption algorithm uses Sub key S. L is merely, a temporary array formed on the basis of user entered secret key. Mix in user's secret key with S and L.

```
i = j = 0
A = B = 0
do 3 * max(t, c) times:
  A = S[i] = (S[i] + A + B) <<< 3
  B = L[j] = (L[j] + A + B) <<< (A + B)
  i = (i + 1) % t
  j = (j + 1) % c
```

Step 5 : Encryption.

- We divide the input plain text block into two registers A and B each of size w bits. After undergoing the encryption processes the result of A and B together forms the cipher text block.

RC5 Encryption Algorithm

1. One time initialization of plain text blocks A and B by adding S[0] and S[1] to A and B respectively. These operations are mod .
2. XOR A and B. A = A ^ B
3. Cyclic left shift new value of A by B bits.
4. Add S[2*i] to the output of previous step. This is the new value of A.
5. XOR B with new value of A and store in B.
6. Cyclic left shift new value of B by A bits.

Scanned by CamScanner

7. Add S[2*i + 1] to the output of previous step. This is the new value of B.

8. Repeat entire procedure (except one time initialization) r times.

```
A = A + S[0]
B = B + S[1]
for i = 1 to r do:
   A = ((A ^ B) <<< B) + S[2 * i]
   B = ((B ^ A) <<< A) + S[2 * i + 1]
return A, B
```

9. Alternatively, RC5 Decryption can be defined as :

```
for i = r down to 1 do:
   B = ((B - S[2 * i + 1]) >>> A) ^ A
   A = ((A - S[2 * i]) >>> B) ^ B
B = B - S[1]
A = A - S[0]
return A, B
```

Chapter End.

□□□

Scanned by CamScanner

CHAPTER 4

Module 2

Public Key Cryptography

Syllabus

Public key cryptography : Principles of public key cryptosystems - The RSA algorithm, The knapsack algorithm, ElGamal Algorithm.

4.1 Public Key Cryptosystem with Applications

- Public key Cryptosystem also called as asymmetric key Cryptography or Public Key Cryptography already discussed in section 1.9.2. (Chapter 1)
- Two different keys are used during encryption and decryption process (one key for encryption and second key used at the time of decryption). RSA algorithm is the best example of asymmetric key cryptography as shown in Fig. 4.1.1.
- Private Key (only known to owner).
- Public Key(possibly known to everyone).

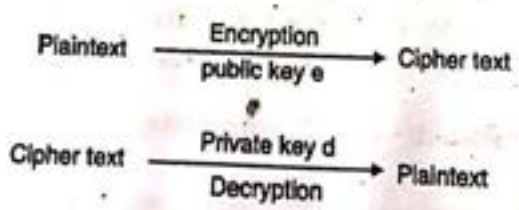


Fig. 4.1.1 : Asymmetric Key Cryptography

Scanned by CamScanner

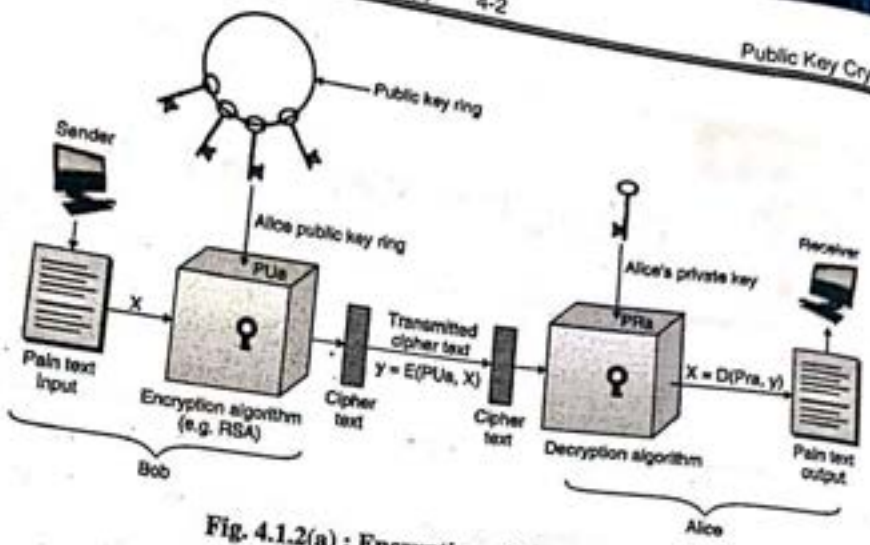
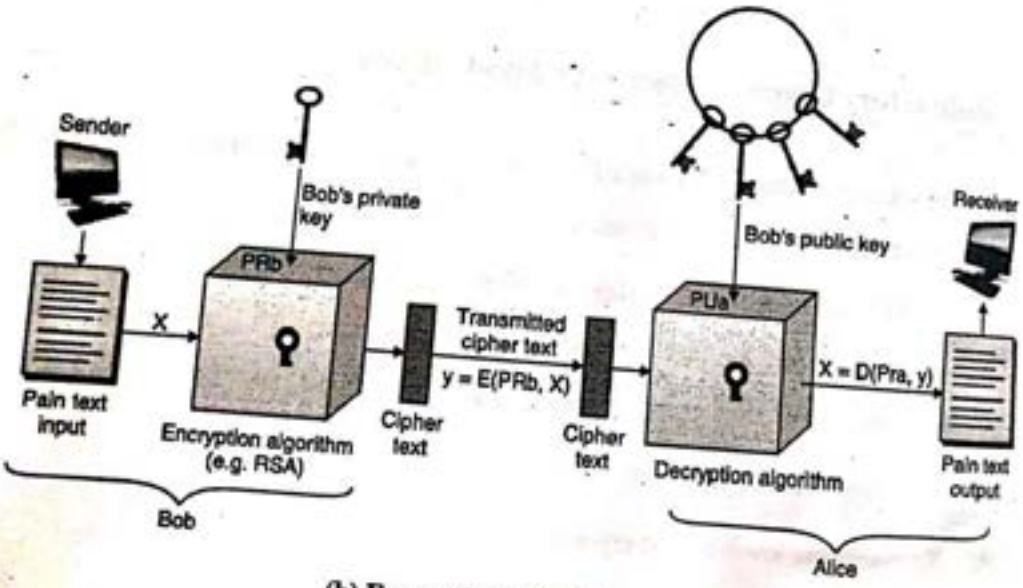


Fig. 4.1.2(a) : Encryption with Public Key

- It is easily configurable than secret key.



(b) Encryption with Private Key

Fig. 4.1.2 : Public Key Cryptography

4.1.1 Applications for Public - Key Cryptosystem

Applications of public key cryptosystem are classified into three categories :

1. **Encryption/decryption :** During this process the sender encrypts the message with the receiver's public key.

Scanned by CamScanner

2. **Digital signature :** During this process the sender "signs" a message with his private key.
3. **Key exchange :** Both sender and receiver cooperate to exchange a session key, typically for conventional encryption.

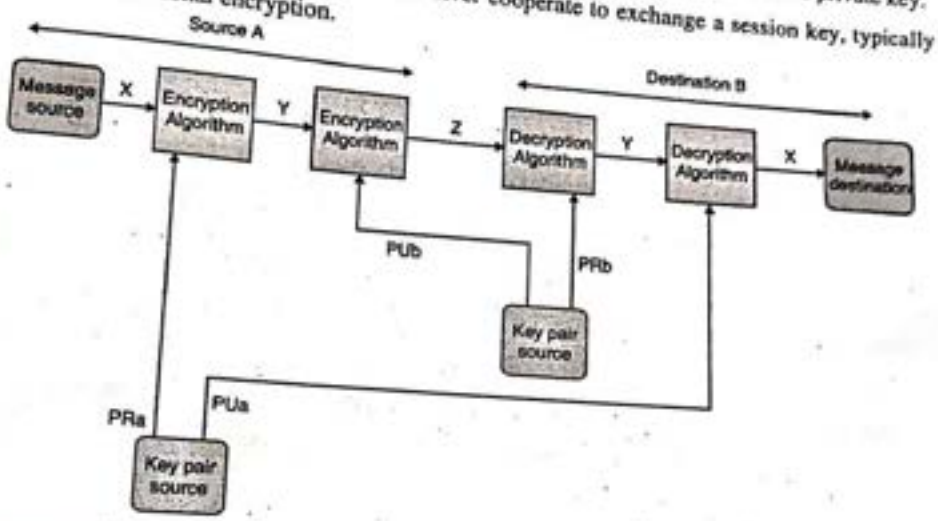


Fig. 4.1.3

Syllabus Topic : Requirements and Cryptanalysis

4.2 Public Key Requirements and Cryptanalysis

Cryptographic algorithm is based on two related key one is diffie and second is Hell man.

1. It is computationally easy for receiver B to generate a key pair (public (PU) and private (PR)).
2. It is computationally easy for sender A knowing PUb and the message to be encrypted to generate the corresponding ciphertext Cp = EnPUb (Pt).
3. It is computationally easy for the receiver B to decrypt the resulting ciphertext using his private key (PRb) to recover the original message.

$$Pt = DrPRb\ (Cp) = DrPRb\ [EnPUb\ (Pt)].$$

4. It is computationally infeasible for a cryptanalyst, knowing the public key PUb, to determine the private key PRb.
5. It is computationally infeasible for a cryptanalyst, knowing PUb and Cp to recover the plaintext message Pt.

Scanned by CamScanner

6. A sixth requirement that, although useful, is not necessary for all public-key applications the encryption and decryption can be applied in either order:

$$Pt = EnPUb\ [DrPRb\ (Pt)] = DrPUb\ [EnPRb\ (Pt)].$$

4.2.1 Public Key Cryptanalysis

- Public key encryption method is accessible to the brute force attack with symmetric encryption.
- Public key systems depend on the use of mathematic equation or some function.
- Because of that the key size is larger enough to avoid brute force attack i.e. brute force attack cannot work on the large size key. Whenever key size is small that time enough for practical encryption and decryption.
- When key size is large that time speed of encryption/decryption are too slow. This key is generating for to make brute force attack impractical.
- Public key encryption is bound to signature application and management.
- Some other form of attack is that if hacker has a public key using that key hacker find some way to calculate the private key, but it is not conform that this type of attack is feasible for a particular public key algorithm.
- Because it is not mathematically proven. So any given algorithm as like RSA is work like suspect.
- If problem is difficult from one angle can be found to have a answer or solution if looked at in an entirely multiple ways. This is the history of cryptanalysis.
- There is some style of attack that is distinct to public key system.
- Suppose take one example, a message has to be send that lie totally of a 56 bit DES key and attacker could encrypted possible 56 bit DES key using the public key and find the encrypted key by matching the transmitted ciphertext.
- In the public key scheme there is no matter that the key length is how large, the attack is minimized to brute force attack on a 56 bit key.
- This attack can be countered by fixing some random bits to such simple message.

Scanned by CamScanner

Syllabus Topic : The RSA Algorithm

4.3 RSA Algorithm : Working, Key Length, Security

→ (MU - May 17)

Q. 4.3.1	Briefly define idea behind RSA and also explain (1) Give public key and private key, (2) Describe security in this system. (Ref. sec. 4.3) **May 17, 5 Marks**
Q. 4.3.2	Explain RSA Algorithm in Details along with suitable example. (Ref. sec. 4.3)
Q. 4.3.3	Explain RSA algorithm used for public key cryptography. (Ref. sec. 4.3)

Ron Rivest, Adi Shamir and Len Aldeman have developed this algorithm (Rivest-Shamir-Aldeman) in 1978. It is a public-key encryption algorithm. It is a block-cipher which converts plain text into cipher text at sender side and vice versa at receiver side.

The algorithm works as follows

1. Select two prime numbers a and b where $a \neq b$.
2. Compute $n = a * b$ (n is used as the modulus for both the public and private keys)
3. Compute$\phi(n) = (a - 1) * (b - 1)$.
4. Select e (public key) such that, e is relatively prime to $\phi(n)$ i.e. gcd $(e, \phi(n)) = 1$ and $1 < e < \phi(n)$.
5. Calculate d (private key) such that,$d = e^{-1} \bmod \phi(n)$ or $\bmod \phi(n) = 1$.
6. Public key = {e,n}, private key = {d,n}.
7. Computecipher textusing,

   $C = P^e \bmod n$ where, $P < n$ where C = Ciphertext,

   P = Plaintext, e = Encryption key
8. Compute Plaintext P using the given formula.

   $P = C^d \bmod n$.

   Where, d = decryption key.

Both sender and receiver know the value of n. In addition, the sender must know encryption key 'e' and receiver must know decryption key'd'.

Scanned by CamScanner

For example

1. Select two prime numbers a = 13, b = 11.
2. n = a*b = 13 * 11 = 143.
3. $\phi(n) = (13 - 1) * (11 - 1) = 12 * 10 = 120$.
4. Select e such that, e is relatively prime to $\phi(n)$ i.e. gcd (e, $\phi(n)$) = 1 and $1 < e < \phi(n)$ e is selected as 13, gcd (13, 120) = 1.
5. **Finding d**

   $e*d \bmod \phi(n) = 1$; $13 * d \bmod 120 = 1$

   d is calculated using following method. Do the following procedure till you are not getting a integer numbers

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

$$d = \frac{(120 + 1)}{13} = \frac{121}{13} = 9.30 \ (i = 1)$$

where, $i = 1$ to 100

$$= \frac{240 + 1}{13} = \frac{241}{13} = 18.53 \ (i = 2)$$

$$= \frac{360 + 1}{13} = \frac{361}{13} = 27.76 \ (i = 3)$$

$$d = \frac{480 + 1}{13} = \frac{481}{13} = 37$$

Hence $\mathbf{d = 37}$

6. Hence public key = {13, 143} and
   Private key = {37, 143}
7. **Encryption**

   Plain text message (P) which is in binary format converted into integer.

   Here P is selected as 13 such that P < n

   $\because (13 < 143)$

   Now, $C = P^e \bmod n = 13^{13} \bmod 143$

   Here to find out $13^{13} \bmod 143$, use the following procedure

   $13 \bmod 143 = 13$

Scanned by CamScanner

$13^2 \bmod 143 = 169 \bmod 143 = 26$

$13^4 \bmod 143 = 26^2 \bmod 143 = 104$

$13^8 \bmod 143 = 104^2 \bmod 143 = 91$

$\therefore\ C = [(13^8 \bmod 143) * (13^4 \bmod 143) * (13 \bmod 143)] \bmod 143$

$= [91 * 104 * 13] \bmod 143 = 52$

8. Decryption

To decrypt given plain text message we must know the C and d.

$$P = C^d \bmod n = 52^{37} \bmod 143$$

Again use above mentioned procedure to find out $52^{37} \bmod 143$. As

$52 \bmod 143 = 52$

$52^2 \bmod 143 = 130$

$52^4 \bmod 143 = (130)^2 \bmod 143 = 26$

$52^8 \bmod 143 = (26)^2 \bmod 143 = 104$

$52^{16} \bmod 143 = (104)^2 \bmod 143 = 91$

$52^{32} \bmod 143 = (91)^2 \bmod 143 = 130$

Hence, $P = 52^{37} \bmod 143$

$= [(52^{32} \bmod 143) * (52^4 \bmod 143) * (52 \bmod 143)] \bmod 143$

$= [130 * 26 * 52] \bmod 143 = 13$

4.3.1 Computational Aspects

Computational Aspects

There are two main issues arises in complexity of the computation we can see that one by one. At the first we see the process of encryption and decryption.

4.3.1(A) Exponentiation in Modular Arithmetic

In RSA, both encryption and decryption method, integer can increased as power and mod n. i.e. $[(p \bmod n) \times (q \bmod n)] \bmod n = (p \times q) \bmod n$ so the given equation say's that, we can reduced the result using modulo n. This calculation we can do practically.

Scanned by CamScanner

Main consideration is the efficiency of that exponentiation. Because when we use the RSA algorithm at the time of large exponents that time efficiency might be increases.

We can take one example of exponentiation. Let's compute x15.

As per regular or straight forward approach we required 14 multiplications as :

$$x = x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x \times x$$

We can find same result using only four multiplication i.e. $(x)\ (x^2)\ (x^4)\ (x^8)$

- In this method we can take square of each partial result.
- Another example is suppose we want to calculate x7 mod n for some integers x and n.
- We can compute that value is as follow :

$$x^7 = x^{1+2+4} = (x^1)\ (x^2)\ (x^4)$$

- In the above case we calculate that (x) mod n, (x^2) mod n, (x^4) mod n and then calculate $[(x \bmod n) \times (x^2 \bmod n) \times (x^4 \bmod n)] \bmod n$
- So we can define that as a mathematical way as : if their is value 'a' we want to find the a power x i.e. a^x.

$$x = \sum_{x_i \neq 0} 2^i = a^x \bmod n$$

4.3.1(B) Efficient Operation using the Public Key

- In RSA algorithm we can increases the speed of the operations using public key, because of that it create a particular choice called as 'e'.
- Almost all common choices are used for the value of 'e' that is the choice is 65537 (216 + 1) and the choice 3 and 17 are the popular common choices.
- These choices has only two 1 bits because of that it requires minimum multiplications to find the exponentiations
- When we use very small public key like e = 3. So RSA algorithm becomes accessible to a simple attack.
- Let's take one example having 3 different RSA users. That 3 users are uses the value e = 3 in this RSA algorithm. But they have different value for 'n' each have unique value called as (N1, N2, N3). If user x sends the encrypted message M to all three users, then it creates 3 cipher texts that are as follows :

$$Ct1 = M3 \bmod N1$$

Scanned by CamScanner

$$Ct2 = M3 \bmod n2$$

$$Ct3 = M3 \bmod N3$$

Where, N1, N2, N3 are pair wise relatively prime.

- Chinese Reminder Theorem (CRT) says that, for computing M3 mod (N1, N2, N3) we use CRT.
- Using the Rules of RSA algorithm : Message M is less that each of the Ni; that is represented as M3< N1, N2, N3.
- Attacker can easily find or decrypt the message like in above example attacker only need to find the cube root of M3. So he can directly find or decrypt that message easily.
- In this attack, padding is done that means it adds a unique pseudo-random bit string. For every instance of message M. for the encryption.
- In RSA algorithm, when key generation process is done that time user need to selects the value of e that becomes relatively prime to ϕ (n)
- If value of e is selected and then prime p and q are generated.
- We can find the gcd (ϕ(n),e) is $\neq$ 1 then user can discard the p, q values and they can generate a new p, q pair.

4.3.1(C) Efficient Operation using the Private Key

- At the time of operation we choose a constant value called as 'd'. When we use that constant value d is small value, that does not give the efficient result.
- When we choose a small value of d that is accessible to a brute-force attack, and also other forms of cryptanalysis.
- So there is one way to speed up the computation that is use of CRT. CRT is Chinese remainder theorem, in that if we want to calculate the value M = cd mod n it gives intermediate result is as follows :

$$X_p = C^d \bmod p$$

$$X_q = C^d \bmod q$$

Where, p and q are the relatively prime numbers.

$$X_p = C^d \bmod p = C^{d \bmod (p-1)} \bmod p$$

$$X_q = C^d \bmod q = C^{d \bmod (q-1)} \bmod q$$

Scanned by CamScanner

- So quantities of d mod (p – 1) and also d mod (q – 1) can be calculated first when we check the end result that result is four time faster evaluates than $M = C^d$ mod n directly.

4.3.1(D) Key Generation

→ (MU - May 16)

Q. 4.3.4 Elaborate the steps of key generation using RSA algorithm. (Ref. sec. 4.3.1(D)) **May 16, 5 Marks**

- In the public key cryptosystem each user need to create a pair of keys. This procedure involves the following tasks :
  (1) Finding two prime numbers called p and q.
  (2) Select the e or d for calculating the other.
- First we want to select p and q where p and q are the relatively prime numbers.
- When we choose the prime numbers that are might be sufficient large set means we choose p and q as large numbers.
- At the present there is no any useful technique that finds the large prime number.
- So general procedure which we used for finding large prime number is to pick the random an odd number and test that number whether that number is prime or not if that number is not prime then choose the next random number and check that number until one is found that tests are prime.
- **For example :** One efficient and popular algorithm is used for finding the prime number that is "Miller-Robin Algorithm"
- In this algorithm, the procedure for testing the given number is prime or not, if integer n is prime to perform the calculation we want to n and randomly chosen integer a.
- If n "Fails" the test that means n is not prime.
- If n "passes" the test, then n may be prime or not prime.
- But if n passes many tests with different randomly chosen values for a
- Then we can say's that the value of n is exactly prime number.
- In short,
  (1) Choose an odd value or n at randomly.
  (2) Choose an integer a < n at randomly.

Scanned by CamScanner

(3) Perform the primary test using Miller-Rabin Algorithm. If n fails the test, then discard the value of n and go to step 1

(4) If n has passed the number of tests which are sufficient to decide the prime number then accept n; otherwise go to the step 2.

- When we find the prime numbers i.e. p and q that time process of key generation is completed by selecting a value of e and computing d or vice versa.

4.3.1(E) The Security of RSA

There are four possible attacks on RSA as follows :

1. **Brute force attack :** Hacker tries all possible private keys.
2. **Mathematical attacks :** Hackers attacks on n i.e. tries to factorize the product of two prime numbers.
3. **Timing attacks :** It totally depends on running time of decryption algorithm.
4. **Chosen Ciphertext attack :** Hacker tries to attack on the properties of RSA algorithm.

4.3.2 Solved Examples on RSA Algorithm

Ex. 4.3.1

Prime number p = 3, q = 11, e = 3, m = 00111011 (m-message) then calculate private key d and cipher text C.

Soln. :

Use RSA Algorithm [Refer Section 4.3]

Step 1 : Prime numbers $p = 3, \quad q = 11$

Step 2 : $n = p * q = 33$

Step 3 : $\phi(n) = (p-1) * (q-1) = (3-1) * (11-1)$

$= 2 * 10 = 20$

Step 4 : Select e such that it is relatively prime to ϕ (n) i.e. gcd(e, ϕ (n)) = 1

$\gcd(e, 20) = 1$

$\gcd(3, 20) = 1$

$e = 3$ is given.

Step 5 : Calculate d such that

$$d = e^{-1} \bmod \phi(n)$$

Scanned by CamScanner

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

$$3 * d \bmod 20 = 1$$

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

Find d such that it is divisible by e.

Where $i = 1$ to 9

$$d = \frac{(20 \times i) + 1}{3} = \frac{(20 \times i) + 1}{3} = \frac{21}{3} = 7$$

$$d = 7$$

Step 6 : Public key = $\{e, n\} = \{3, 33\}$

Private key = $\{d, n\} = \{7, 33\}$

Step 7 : Calculate cipher text message for given plain text message.

Plain text message given in binary 00111011 which can be written as 59

Binary to decimal conversion

$00111011 \Rightarrow 59$ (P = 59)

$$c = P^e \bmod n \text{ where } P < n = 59^3 \bmod 33$$

$$= [59^2 \bmod 33] * [59 \bmod 33] \bmod 33$$

$$= 3481 \bmod 33 * [59 \bmod 33] \bmod 33$$

$$c = [16 * 26] \bmod 33$$

$$c = 20$$

Step 8 : Calculate plain text message.

$$P = c^d \bmod n = 20^7 \bmod 33$$

$$P = [20^4 \bmod 33] * [20^3 \bmod 33] \bmod 33$$

$$= [20^2 \bmod 33] * [20^2 \bmod 33] * [20^2 \bmod 33] * [20^1 \bmod 33] \bmod 33$$

$$= [400 \bmod 33] * [400 \bmod 33] * [400 \bmod 33] * [20 \bmod 33] \bmod 33$$

$$= [4] * [4] * [4] * [20] \bmod 33 = 1280 \bmod 33$$

$$P = 26$$

Scanned by CamScanner

Ex. 4.3.2

Calculate cipher text using RSA algorithm given data as follows : Prime numbers p, q as 7, 17 respectively and plain text message is to be send is 10.

Soln. : By using RSA Algorithm : [Refer Section 4.3]

Step 1 : Prime numbers are 7 and 17 $a = 7, \ b = 17$

Step 2 : $n = a * b = 7 * 17 = 119.$

Step 3 : $\phi(n) = (a-1) * (b-1) = (7-1) * (17-1)$

$= 6 * 16 = 96$

Step 4 : Select e such that it is relatively prime to ϕ (n) i.e. $\gcd(e, \phi(n)) = 1$

If we select s then it is not relatively prime p 96 because

$$3 = 1 * 3$$

$$96 = 2 * 2 * 2 * 2 * 2 * 3$$

gcd must be 1.

We select e as 5 (gcd must be 1)

$$5 = 1 * 5$$

$$\gcd(5, 96) = 1$$

Step 5 : Calculate d such that

$$d = e^{-1} \bmod \phi(n)$$

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

$$5 * d \bmod 96 = 1$$

Using RSA algorithm

$$d = \left(\frac{(\phi(n) * i) + 1}{5}\right)$$

where $i = 1$ to $9 = \frac{(96 * 1) + 1}{5} = 19.4$

d must be completely divisible by 'e'.

$$= \frac{(96 * 2) + 1}{5} = 38.6 = \frac{(96 * 3) + 1}{5} = 57.8$$

$$= \frac{(96 * 4) + 1}{5} = 77$$

$$\mathbf{d = 77}$$

Scanned by CamScanner

Step 6 : Public key = {e, n} = {5, 119}

Private key = {d, n} = {77, 119}

Step 7 : Calculate cipher text message for given plain text message m = 10.

Plain text denoted as p = 10 (m denoted as p)

$$c = p^e \bmod n = 10^5 \bmod 119$$

It can be represented as

$$10^5 \bmod 119 = [10^3 \bmod 119] * [10^2 \bmod 119] \bmod 119$$

$$= [1000 \bmod 119] * [100 \bmod 119] \bmod 119$$

$$= 100000 \bmod 119$$

$$c = 40$$

Step 8: Now calculate plain text P required at the time of decryption. Once sender sends 40 to the receiver then receiver can calculate plain text p.

$$P = c^d \bmod n = 40^{77} \bmod 119$$

Now represent 40^{77} mod 119 as mention above it will results p as 10.

Because decryption process always yields original message / plain text

$$\therefore \; P = 40^{77} \bmod 119 = 10$$

$$P = 10$$

Ex. 4.3.3

Calculate cipher text using RSA algorithm given data is as follows :

Prime numbers P, Q as 13, 17 and the plain text to be send is 12. Assume public key e as 19

Soln. :

Using RSA Algorithm [Refer Section 4.3]

Step 1 : P and Q denoted as a and b in our Algorithm

$$a = 13, \quad b = 17$$

Step 2 : $n = a * b = 13 * 17 = 221.$

Step 3 : $\phi(n) = (a - 1) * (b - 1) = (13 - 1) * (17 - 1)$

$$= 12 * 16 = 192$$

Step 4 : Select e such that it is relatively prime to ϕ (n) e is given as 19.

Scanned by CamScanner

Step 5 : Calculate d such that,

$$d = e^{-1} \bmod \phi(n)\ e^{d} \bmod \phi(n) = 1$$

$$d = \frac{(\phi(n) * i) + 1}{e} \quad \text{where } i = 1 \text{ to } 9$$

$$= \frac{(192 * 1) + 1}{19} = 10.1 = \frac{(192 * 2) + 1}{19} = 20.2$$

$$= \frac{(192 * 3) + 1}{19} = 30.3 = \frac{(192 * 4) + 1}{19} = 40.4$$

$$= \frac{(192 * 5) + 1}{19} = 50.5 = \frac{(192 * 6) + 1}{19} = 60.6$$

$$= \frac{(192 * 7) + 1}{19} = 70.7 = \frac{(192 * 8) + 1}{19} = 80.8$$

$$= \frac{(192 * 9) + 1}{19} = 91$$

$$\therefore \quad \mathbf{d = 91}$$

Step 6 : Public key = {e, n} = {19, 221}

Private key = {d, n} = {91, 221}

Step 7: Calculate cipher text c for given plain text message 12.

$$\therefore \ c = p^{e} \bmod n$$

$$= 12^{19} \bmod 221$$

$$= [12^{10} \bmod 221] * [12^{5} \bmod 221] * [12^{4} \bmod 221]$$

$$= [12^{5} \bmod 221] * [12^{5} \bmod 221] * [12^{5} \bmod 221] * [12^{4} \bmod 221] \bmod 221$$

$$= [207] * [207] * [207] * [183] \bmod 221$$

$$\mathbf{c = 181}$$

Step 8 : Send c = 181 to receiver as if required for decryption to obtain original plain text p.

$$P = c^{d} \bmod n = 181^{91} \bmod 221$$

This yields value of original plain text message i.e. 12

$$\mathbf{P = 12}$$

Scanned by CamScanner

Ex. 4.3.4

In public key cryptosystem given N = 187 and encryption key (E) as = 17. Find o corresponding private key (D).

Soln. :

RSA Algorithm [Refer Section 4.3]

Step 1 : Select two large random prime numbers a and b. if we select a = 17 and b = 1 which results n = 187.

$$n = a * b = 17 * 11 = 187.$$

Step 2 : Calculate $\phi(n) = (a-1)\ (b-1)$

$$= (17-1)(11-1)$$

$$= 16 * 10 = 160$$

Step 3 : Select e such that it is relatively prime to ϕ (n) and less than ϕ (n). But it is give in problem statement that e = 17.

Step 4 : Calculate d such that,

$$d = e^{-1} \bmod \phi(n)$$

$$e^d \bmod \phi(n) = 1$$

$$d = \frac{(\phi(n) * i) + 1}{e} \quad \text{where } i = 1 \text{ to } 20$$

$$= \frac{(160 * 1) + 1}{17} = 9.4 = \frac{(160 * 2) + 1}{17} = 18.8$$

$$= \frac{(160 * 3) + 1}{17} = 28.2 = \frac{(160 * 4) + 1}{17} = 37.70$$

$$= \frac{(160 * 5) + 1}{17} = 47.11 = \frac{(160 * 6) + 1}{17} = 56.52$$

$$= \frac{(160 * 7) + 1}{17} = 65.94 = \frac{(160 * 8) + 1}{17} = 75.35$$

$$= \frac{(160 * 9) + 1}{17} = 84.76$$

$$= \frac{(160 * 12) + 1}{17} = 113$$

$$\therefore \quad \mathbf{d = 113}$$

Scanned by CamScanner

Ex. 4.3.5

Using the RSA algorithm encrypt the following :

(i) $p = 3$, $q = 11$, $e = 7$, $M = 12$

(ii) $p = 7$, $q = 11$, $e = 17$, $M = 25$

(iii) Find the corresponding ds for (i) and (ii) and decrypt the cipher texts.

Soln. :

Use RSA Algorithm

(i) Consider p as *a* and q as *b* as per our notations for prime numbers.

Step 1 : Prime numbers $a = 3$, $b = 11$

Step 2 : $n = a * b = 33$

Step 3 : $\phi(n) = (a - 1) * (b - 1)$

$= (3 - 1) * (11 - 1)$

$= 2 * 10 = 20$

Step 4 : Select e such that it is relatively prime to ϕ (n) i.e. $\gcd(e, \phi(n)) = 1$

$\gcd(e, 20) = 1$

$\gcd(7, 20) = 1$

$e = 7$ is given.

Step 5 : Calculate d such that

$$d = e^{-1} \bmod \phi(n)$$

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

$7 * d \bmod 20 = 1$

$$d = \frac{(\phi(n) * i) + 1}{e} \quad \text{where } i = 1 \text{ to } 100$$

Find d such that it is divisible by e.

Consider i = 1 you can continue till d will get integer value, Q(n) = 20 and e = 7

$d = ((20 * 1) + 1)/7 = 21/7 = 3$

$\mathbf{d = 3}$

Step 6 : Public key $= \{e, n\} = \{7, 33\}$

Private key $= \{d, n\} = \{3, 33\}$

Scanned by CamScanner

Step 7 : Calculate cipher text message for given plain text message.

Plain text message given is M = 12 we consider M as i.e. P = 12

$$C = p^e \bmod n \text{ where } p < n$$

$$= 12^7 \bmod 33$$

$$C = 12$$

Step 8 : Calculate plain text message.

$$P = c^d \bmod n = 12^3 \bmod 33$$

$$\mathbf{P = 12}$$

When we convert plain text message into cipher text the corresponding cipher text yields the same plain text.

(ii) p = 7 , q = 11, e = 17, M = 25

By using RSA Algorithm : [Refer Section 4.3]

Step 1 : Prime numbers are 7 and 11 as per our notations a = 7, b = 11

Step 2 : $n = a * b = 7 * 11 = 77.$

Step 3 : $\phi(n) = (a - 1) * (b - 1)$

$$= (7 - 1) * (11 - 1) = 6 * 10 = 60$$

Step 4 : Select e such that it is relatively prime to ϕ (n) i.e. $\gcd(e, \phi(n)) = 1$

e is given as 17

gcd (17,60) = 1 (gcd must be 1)

Step 5 : Calculate d such that

$$d = e^{-1} \bmod \phi(n)$$

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

$$17 * d \bmod = 1$$

Using RSA algorithm

$$d = \frac{(\phi(n) * i) + 1}{e} \quad \text{where } i = 1 \text{ to } 100$$

$$= ((60*1) + 1)/17)$$

$$= \mathbf{3.58}$$

d must be completely divisible by 'e'.

Scanned by CamScanner

= After putting value of i = 15 into above formula we got value of d

= ((60*15) +1)/ 17) = 53

d = 53

Step 6 : Public key = {e, n} = {17, 77}

Private key = {d, n} = {53, 77}

Step 7 : Calculate cipher text message for given plain text message M = 25.

Plain text denoted as P = 25 (m denoted as p)

$$C = P^e \bmod n$$

$$= 25^{17} \bmod 77$$

It can be represented as

$$C = 9$$

Step 8 : Now calculate plain text P required at the time of decryption. Once sender sends 9 to the receiver then receiver can calculate plain text p.

$$P = C^d \bmod n$$

$$= 9^{53} \bmod 77$$

$$P = 25$$

Decryption process always yields original plain text message

$$\therefore \mathbf{P = 25}$$

(iii) Find the corresponding ds for (i) and (ii) and decrypt the cipher texts

Decryption key for question (i) is d =3 and for question (ii) is d= 53 which will decrypt the message successfully.

Ex. 4.3.6 **MU - Dec. 15, 10 Marks**

In an RSA system the public key (e, n) of user A is defined as (7,119). Calculate $\phi(n)$ and private key d. What is the cipher text when you encrypt message m =10, using the public key?

Soln. :

By using RSA Algorithm : [Refer Section 4.3]

In the problem statement Public key (e, n) = (7,119) is given, means we don't need to select e & n. if we select following prime numbers which results n = 119 as shown below.

Step 1 : Prime numbers are 7 and 17 a = 7, b = 17

Scanned by CamScanner

Step 2 : $n = a * b = 7 * 17 = 119.$

Step 3 : $\phi(n) = (a - 1).(b - 1)$

$= (7 - 1) * (17 - 1)$

$= 6 * 16 = 96$

Step 4 : Select e such that it is relatively prime to ϕ (n) i.e. $\gcd(e, \phi(n)) = 1$

E = 7 as per problem statement.

Step 5 : Calculate d such that

$$d = e^{-1} \bmod \phi(n)$$

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

$$7 * d \bmod 96 = 1$$

Using RSA algorithm

$$d = ((\phi(n) * i) + 1)/7 \quad \text{where } i = 1 \text{ to } 100$$

$$= (96*1 + 1)/7 = 13.85$$

d must be completely divisible by 'e'.

$$= ((96*2) + 1)/7 = 21.57$$

$$= ((96*3) + 1)/7 = 48.28$$

$$= ((96*4) + 1)/7 = 55$$

$$d = 55$$

Step 6 : Public key = {e, n} = {7, 119}

Private key = {d, n} = {55, 119}

Step 7 : Calculate cipher text message for given plain text message m = 10.

Plain text denoted as p = 10 (m denoted as p)

$$C = P^e \bmod n$$

$$= 10^7 \bmod 119$$

$$= 10000000 \bmod 119$$

$$C = 73$$

Step 8 : Now calculate plain text P required at the time of decryption. Once sender sends 40 to the receiver then receiver can calculate plain text p.

$$P = C^d \bmod n = 73^{55} \bmod 119$$

Now represent 40^{55} mod 119 as mention above it will results p as 10.

Scanned by CamScanner

Because decryption process always yields original message / plain text

$$\therefore \quad P = 73^{35} \bmod 119 = 10$$

$$P = 10$$

Ex. 4.3.7

Perform encryption using the RSA algorithm p = 3, q = 11(two random numbers), e(encryption key) = 7, M(plaintext message) = 5

Soln. :

Using RSA algorithm

$p = 3, \quad q = 11, \quad e = 7 \quad$ and $M = 5$

Step 1 : Prime number $p = 3, q = 11$

Step 2 : $n = p * q = 3 \times 11 = 33$

Step 3 : $\phi(n) = (p-1)*(q-1) = (3-1)*(11-1)$

$$\phi(n) = 2*10 = 20$$

Step 4 : Select e such that it is relatively prime to $\phi(n)$ i.e. $\gcd(e, \phi(n)) = 1$

$\gcd(e, 20) = 1;$ $\qquad \gcd(7, 20) = 1$

$$e = 7 \text{ is given.}$$

Step 5 : Calculate d such that

$$d = e^{-1} \bmod \phi(n)$$

$$e\,d \bmod \phi(n) = 1$$

$$7*d \bmod 20 = 1$$

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

Find d such that is divisible by e.

where, $\quad i = 1$ to 100

$$d = \frac{(20 \times i)+1}{7} \text{ where } i = 1;$$

$$= \frac{20+1}{7} = \frac{21}{7} = 3$$

$$d = 3$$

Step 6 : Public key = {e, n} = {7, 33}

Private key = {d, n} = {3, 33}

Scanned by CamScanner

Step 7 : Calculate cipher text message for given plain text message plain text M = 5.

$$C = M^e \bmod n \quad \text{where } M < n$$

$$= 5^7 \bmod 33$$

$$= (5^5 \bmod 33) * (5^2 * \bmod 33) * \bmod 33$$

$$= (5^3 \bmod 33) * (5^2 * \bmod 33) * (5^2 * \bmod 33) * \bmod 33$$

$$= (125 \bmod 33) * (25 \bmod 33) * (25 \bmod 33) * \bmod 33$$

$$= (26 * 25 * 25) * \bmod 33$$

$$= 16250 * \bmod 33$$

$$c = 14$$

Ex. 4.3.8

The encryption algorithm to be used is RSA. Given two prime numbers 11 and 3 and public key (e) is 3. Calculate the decryption key and Calculate the ciphertext if the given plaintext is 7.

Soln. :

Using RSA algorithm

Given : a = 11, b = 3, e = 3 and plain text P = 7

Step 1 : Prime number a = 11, b = 3

Step 2 : $n = a * b = 11 * 3 = 33$

Step 3 : $\phi(n) = (a-1) * (b-1)$

$$= (11-1) * (3-1) = 10 * 2$$

$$= 20$$

Step 4 : Select e such that it is relatively prime to $\phi(n)$ i.e. $\gcd(e, \phi(n)) = 1$

$$\gcd(e, 20) = 1$$

$$\gcd(3, 20) = 1$$

$$e = 3 \text{ is given.}$$

Step 5 : Calculate d such that

$$d = e^{-1} \bmod \phi(n)$$

$$e\,d \bmod \phi(n) = 1$$

Scanned by CamScanner

$$3 * d \bmod 20 = 1$$

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

Find d such that it is divisible by e

Where $i = 1$ to 100

$$d = \frac{(20 * i) + 1}{3} \text{ where } i = 1;$$

$$d = \frac{(20 \times i) + 1}{3} = \frac{21}{3} = 7$$

$$\mathbf{d = 7}$$

Step 6 : Public key = {e, n} = {3, 33}

Private key = {d, n} = {7, 33}

Step 7 : Calculate cipher text message for given plain text message.

Plain text message = 7

$$c = P^e \bmod n \text{ where } P < n$$

$$= 7^3 \bmod 33$$

$$= [7^3 \bmod 33] * [7 \bmod 33] * \bmod 33$$

$$= [49 \bmod 33] * [7 \bmod 33] * \bmod 33$$

$$= [16 * 7] \bmod 33$$

$$\mathbf{c = 13}$$

so, **cipher text = 3**

Ex. 4.3.9

For the given parameters 'P' = 3 and 'Q' = 19 find the value of 'e' and 'd' using RSA algorithm and encrypt message 'M' = 6.

Soln. :

Step 1 : Prime numbers are P = 3 and Q= 19 we are denoting P & Q as a = 3, b = 19

Step 2 : $n = a * b = 3 * 19 = 57$

Step 3 : $\phi(n) = (a - 1).(b - 1)$

$$= (3 - 1) * (19 - 1)$$

Scanned by CamScanner

$\phi(n) = 2 * 18 = 36$

Step 4 : Select e such that it is relatively prime to ϕ (n) i.e. gcd(e, ϕ(n)) = 1 $1 < e < \phi(n)$ e is selected as 5, gcd (5, 36) = 1.

Here gcd(5,36) = 1 One can select e as 7 because 7 is also relatively prime to ϕ (n)

It better if you select large private key therefore we will select e as 7

gcd(7,36) = 1

Step 5 : Calculate d such that

$$d = e^{-1} \bmod \phi(n)$$

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

$$7 * d \bmod 36 = 1$$

Using RSA algorithm

$$d = ((\phi(n) * i) + 1)/7 \text{ where } i = 1 \text{ to } 100$$

$$= ((36*1) + 1)/7 = 5.28$$

d must be completely divisible by 'e'.

$$= ((36*2) + 1)/7 = 10.42$$

$$= ((36*3) + 1)/7 = 15.57$$

$$= ((36*4) + 1)/7 = 20.71$$

$$= ((36*5) + 1)/7 = 25.85$$

$$= ((36*6) + 1)/7 = 31$$

$$d = 31$$

Step 6 : Public key = {e, n} = {7, 57}

Private key = {d, n} = {31, 57}

Step 7 : Calculate cipher text message for given plain text message m = 6.

Plain text denoted as p = 6 (m denoted as p)

$$C = P^e \bmod n$$

$$= 6^7 \bmod 57$$

$$= 279936 \bmod 57$$

$$C = 9$$

Scanned by CamScanner

Step 8 : Now calculate plain text P. Once sender sends 9 to the receiver then receiver can calculate plain text p.

$$P = C^d \bmod n$$

$$= 9^{31} \bmod 57 = 6$$

Decryption process always yields original message / plain text i.e. = 6

$$\therefore P = 9^{31} \bmod 57 = 6$$

$$P = 6$$

Syllabus Topic : The Knapsack Algorithm

4.4 Knapsack Algorithm

- In 0-1 Knapsack, items cannot be broken which means the thief should take the item as a whole or should leave it. This is reason behind calling it as 0-1 Knapsack.
- Hence, in case of 0-1 Knapsack, the value of x_i can be either *0* or *1*, where other constraints remain the same.
- 0-1 Knapsack cannot be solved by Greedy approach. Greedy approach does not ensure an optimal solution. In many instances, Greedy approach may give an optimal solution.
- The following examples will establish our statement.

Example 1

- Let us consider that the capacity of the knapsack is W = 25 and the items are as shown in the following table.

Item	A	B	C	D
Profit	24	18	18	10
Weight	24	10	10	7

- Without considering the profit per unit weight (p_i/w_i), if we apply Greedy approach to solve this problem, first item *A* will be selected as it will contribute maximum profit among all the elements.

Scanned by CamScanner

- After selecting item A, no more item will be selected. Hence, for this given set of items total profit is *24*. Whereas, the optimal solution can be achieved by selecting items, B and C, where the total profit is 18 + 18 = 36.

Example 2

- Instead of selecting the items based on the overall benefit, in this example the items are selected based on ratio p_i/w_i. Let us consider that the capacity of the knapsack is $W = 40$ and the items are as shown in the following table.

Item	A	B	C
Price	100	280	120
Weight	10	40	20
Ratio	10	7	6

- Using the Greedy approach, first item A is selected. Then, the next item B is chosen. Hence, the total profit is **100 + 280 = 380**. However, the optimal solution of this instance can be achieved by selecting items, B and C, where the total profit is **280 + 120 = 400**.
- Hence, it can be concluded that Greedy approach may not give an optimal solution.
- To solve 0-1 Knapsack, Dynamic Programming approach is required.

4.4.1 Problem Statement

A thief is robbing a store and can carry a maximal weight of W into his knapsack. There are n items and weight of ith item is w_i and the profit of selecting this item is p_i. What items should the thief take?

4.4.2 Dynamic-Programming Approach

Let i be the highest-numbered item in an optimal solution **S** for **W** dollars. Then $S' = S - \{i\}$ is an optimal solution for $W - w_i$ dollars and the value to the solution S is V_i plus the value of the sub-problem.

We can express this fact in the following formula: define **c[i, w]** to be the solution for items **1,2, … , i** and the maximum weight **w**.

Scanned by CamScanner

The algorithm takes the following inputs :

- The max,mum weight **W**
- The number of items **n**
- The two sequences $v = \langle v_1, v_2, \ldots, v_n \rangle$ and $w = \langle w_1, w_2, \ldots, w_n \rangle$

Dynamic-0-1-knapsack (v, w, n, W)

```
for w = 0 to W do
    c[0, w] = 0
for i = 1 to n do
    c[i, 0] = 0
for w = 1 to W do
    ifw_i ≤ w then
    if v_i + c[i-1, w-w_i] then
        c[i, w] = v_i + c[i-1, w-w_i]
    else c[i, w] = c[i-1, w]
else
    c[i, w] = c[i-1, w]
```

- The set of items to take can be deduced from the table, starting at **c[n, w]**and tracing backwards where the optimal values came from.
- If *c[i, w] = c[i-1, w]*, then item *i* is not part of the solution, and we continue tracing with **c[i-1, w]**. Otherwise, item *i* is part of the solution, and we continue tracing with **c[i-1, w-W]**.

4.4.3 Analysis

This algorithm takes $\theta(n, w)$ times as table *c* has $(n + 1).(w + 1)$ entries, where each entry requires $\theta(1)$ time to compute.

Syllabus Topic : ElGamal Algorithm

4.5 ElGamal Algorithm

- Along with RSA, there are other public-key cryptosystems proposed. Many of them are based on different versions of the Discrete Logarithm Problem.
- ElGamal cryptosystem, called Elliptic Curve Variant, is based on the Discrete Logarithm Problem. It derives the strength from the assumption that the discrete logarithms cannot be

Scanned by CamScanner

found in practical time frame for a given number, while the inverse operation of the prod can be computed efficiently.

- Let us go through a simple version of ElGamal that works with numbers modulo p. In the case of elliptic curve variants, it is based on quite different number systems.

4.5.1 Generation of ElGamal Key Pair

Each user of ElGamal cryptosystem generates the key pair through as follows :

- **Choosing a large prime p.** Generally a prime number of 1024 to 2048 bits length is chosen.
- **Choosing a generator element g.**
  - This number must be between 1 and p – 1, but cannot be any number.
  - It is a generator of the multiplicative group of integers modulo p. This means for every integer m co-prime to p, there is an integer k such that $g^k = a \bmod n$.
- For example, 3 is generator of group 5 ($Z_5 = \{1, 2, 3, 4\}$).

N	3^n	3^n mod 5
1	3	3
2	9	4
3	27	2
4	81	1

- **Choosing the private key.** The private key x is any number bigger than 1 and smaller than p–1.
- **Computing part of the public key.** The value y is computed from the parameters p, g and the private key x as follows :

$$y = g^x \bmod p$$

- **Obtaining Public key.** The ElGamal public key consists of the three parameters (p, g, y)
- For example, suppose that p = 17 and that g = 6 (It can be confirmed that 6 is a generator of group Z_{17}). The private key x can be any number bigger than 1 and smaller than 71, so we choose x = 5. The value y is then computed as follows :

$$y = 6^5 \bmod 17 = 7$$

Scanned by CamScanner

- Thus the private key is 62 and the public key is (17, 6, 7).

4.5.2 Encryption and Decryption

The generation of an ElGamal key pair is comparatively simpler than the equivalent process for RSA. But the encryption and decryption are slightly more complex than RSA.

4.5.3 ElGamal Encryption

Suppose sender wishes to send a plaintext to someone whose ElGamal public key is (p, g, y), then :

- Sender represents the plaintext as a series of numbers modulo p.
- To encrypt the first plaintext P, which is represented as a number modulo p.
- The encryption process to obtain the ciphertext C is as follows :
  - Randomly generate a number k;
  - Compute two values C1 and C2, where –

$$C1 = g^k \bmod p$$

$$C2 = (P^* y^k) \bmod p$$

- Send the ciphertext C, consisting of the two separate values (C1, C2), sent together.
- Referring to our ElGamal key generation example given above, the plaintext P = 13 is encrypted as follows :
  - Randomly generate a number, say k = 10
  - Compute the two values C1 and C2, where –

$$C1 = 6^{10} \bmod 17$$

$$C2 = (13^* 7^{10}) \bmod 17 = 9$$

- Send the ciphertext C = (C1, C2) = (15, 9).

4.5.4 ElGamal Decryption

- To decrypt the ciphertext (C1, C2) using private key x, the following two steps are taken :
  - Compute the modular inverse of $(C1)^x$ modulo p, which is $(C1)^{-x}$, generally referred to as decryption factor.
  - Obtain the plaintext by using the following formula –

$$C2 \times (C1)^{-x} \bmod p = \text{Plaintext}$$

Scanned by CamScanner

- Thus the private key is 62 and the public key is (17, 6, 7).

4.5.2 Encryption and Decryption

The generation of an ElGamal key pair is comparatively simpler than the equivalent process for RSA. But the encryption and decryption are slightly more complex than RSA.

4.5.3 ElGamal Encryption

Suppose sender wishes to send a plaintext to someone whose ElGamal public key is (p, g, y), then :

- Sender represents the plaintext as a series of numbers modulo p.
- To encrypt the first plaintext P, which is represented as a number modulo p.
- The encryption process to obtain the ciphertext C is as follows :
  - Randomly generate a number k;
  - Compute two values C1 and C2, where –

$$C1 = g^k \bmod p$$

$$C2 = (P^* y^k) \bmod p$$

- Send the ciphertext C, consisting of the two separate values (C1, C2), sent together.
- Referring to our ElGamal key generation example given above, the plaintext P = 13 is encrypted as follows :
  - Randomly generate a number, say k = 10
  - Compute the two values C1 and C2, where –

$$C1 = 6^{10} \bmod 17$$

$$C2 = (13^* 7^{10}) \bmod 17 = 9$$

- Send the ciphertext C = (C1, C2) = (15, 9).

4.5.4 ElGamal Decryption

- To decrypt the ciphertext (C1, C2) using private key x, the following two steps are taken :
  - Compute the modular inverse of $(C1)^x$ modulo p, which is $(C1)^{-x}$, generally referred to as decryption factor.
  - Obtain the plaintext by using the following formula –

$$C2 \times (C1)^{-x} \bmod p = \text{Plaintext}$$

Scanned by CamScanner

- In our example, to decrypt the ciphertext C = (C1, C2) = (15, 9) using private key x = 5, the decryption factor is

$$15^{-5} \bmod 17 \quad = 9$$

- Extract plaintext P = (9 × 9) mod 17 = 13.

4.5.5 ElGamal Analysis

- In ElGamal system, each user has a private key x. and has **three components** of public key – **prime modulus p, generator g, and public Y = g^x mod p**. The strength of the ElGamal is based on the difficulty of discrete logarithm problem.
- The secure key size is generally > 1024 bits. Today even 2048 bits long key are used. On the processing speed front, Elgamal is quite slow, it is used mainly for key authentication protocols. Due to higher processing efficiency, Elliptic Curve variants of ElGamal are becoming increasingly popular.

Chapter Ends...

□□□

Scanned by CamScanner

CHAPTER 5

Module 2

Key Management Techniques

Syllabus

Key management techniques : using symmetric and asymmetric algorithms and trusted third party. Diffie Hellman Key exchange algorithm.

5.1 Key Distribution and Management

Before discussing the key generation and usage let us first discuss what is Key.

5.1.1 Management

- The main aim of Key management is to generate a secret key between two parties and store it to prove the authenticity between communicating users.
- Key management is the techniques which support key generation, storage and maintenance of the key relationship between authorized users.
- The purpose of this unit is to give idea about the issues involved and a broad survey of the various aspects of key management and distribution of keys.
- Key management plays an important role in cryptography as the basis for securing cryptographic goals like confidentiality, authentication, data integrity, and digital signatures.
- It is not the case where communicating parties are using same key for encryption and decryption or whether two different keys are used for encryption and decryption the basic purpose of key management is key generation, key distribution, controlling the use of keys, updating, destruction of keys and storage, backup/recovery.

Scanned by CamScanner

- Key can be generated by using well known symmetric as well as asymmetric cryptographic algorithms like Rivest Shamir Adleman (RSA), Diffie Hellman Key exchange algorithms and can be used later for encryption and decryption of data.

Syllabus Topic : Key Management Techniques - Using symmetric and Asymmetric Algorithms and Trusted Third Party

5.1.2 Symmetric Key Distribution using Symmetric and Asymmetric Encryptions

5.1.2(A) Symmetric Key Distribution using Symmetric Encryption

1. Key distribution Scenario
2. Hierarchical Key control
3. Session key lifetime
4. A transparent key control scheme
5. Decentralized key control

- When two parties share the same key that protected from access by others, that key is nothing but symmetric key and the process between two parties that exchanges the key called as symmetric encryption.
- If two person wants to communicates with each other via messages or exchange data without interference of other.
- Two parties/persons X and Y achieved the key distribution in various ways :
  1. X can select a key and physically handover to the Y.
  2. A third person can select key and handover to X and Y.
  3. If X and Y uses a key previously then any one person i.e. X or Y can send the old key with encryption to other person.
  4. X and Y has its own encrypted connection to third person and that person can deliver a key on the encrypted links to X and Y.
- Point (1) and (2) are the manual delivery of the key. In link encryption device data can be transferred between only two partners. This is end to end encryption.
- But in distributed system one host can communicated with many others hosts or devices so because of that each device needs many keys that supplied dynamically.
- In wide-area distributed system it is difficult to manage number of keys.

Scanned by CamScanner

- If there are N hosts communicate each other than [N(N – 1)]/2 keys are required.
- When attacker succeeds to gaining access to any one key then other related keys will be opened.
- Key distribution centre is responsible for distributing keys in the form of pairs of users as per need. Every user shares a unique key with key distribution center.
- Using temporary key communication is encrypted between end users. This temporary key called as session key.
- Session key is used for some logical duration in between connection like, in frame relay connection, transport connection etc. after that session key will be discarded.
- End user uses same networking facilities which have session key provided by key distribution center. Session key is in the form of encryption. Master key is also provided by Key Distribution Center (KDC).
- So KDC shares that Master key to end user or system.
- Each user has one master key shares with key distribution center that can be distribution in some fashion.

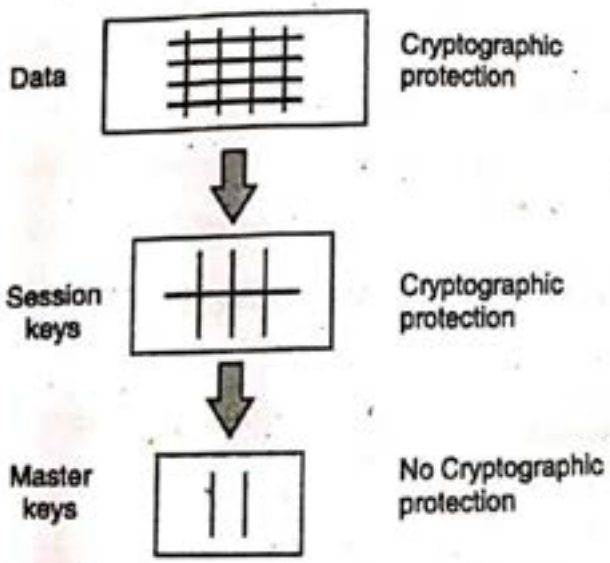


Fig. 5.1.1 : The use of key Hierarchy

1. **Key distribution Scenario**

- The scenario refers each user have its own unique key called master key with key distribution center.
- Let us assume that user X wishes to create a connection with user Y, So he requires one time session key for protecting data.

Scanned by CamScanner

- So X has mater key K_x that is known only itself and the key distribution center, and Y has also its own mater key K_y shares with key distribution center.

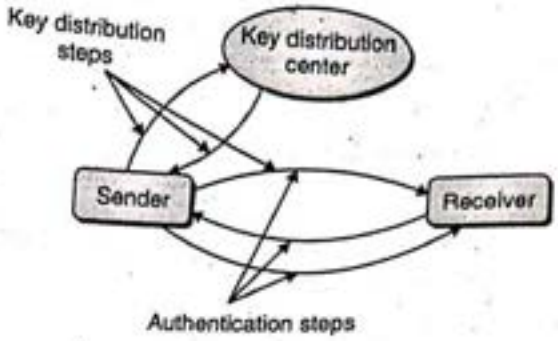


Fig. 5.1.2 : Key Distribution Scenario

- So following steps are required :
  1. X sends request for session key to the KDC which protect the connection between X and Y. The message includes the identity of X and Y and a unique identifier, which is used for the transaction.
  2. KDC sends back the message to sender X is in the form of encryption using K_x.
- So only X can read the original message
- The message includes 2 items for X :
  1. One-time session key
  2. Original request message called as nonce.
- In side Y it also includes 2 items.
  1. One-time session key
  2. Identifier of X
- X stores the session key for next session and forward that key to Y. that is encrypted with K_y.

2. Hierarchical key control

- In very large network area key distribution center does not work properly. It is difficult to handle in large area. So it is not done in practically.
- Because of that option of key distribution center replace with Hierarchical key control or hierarchy of Key Distribution Center (KDC).

Scanned by CamScanner

- For example : Each local KDC is responsible for small area like single building or single LAN.
- If two different domain wants to communicate or share some key that time local KDC connect or contact with global KDC.
- The hierarchical approach having three layer or more layer.
- It depends on the geographic scope number of the users.
- This scheme minimises the effort used in master key distribution

3. Session key lifetime

- Session key is used for exchanging data over the network from sender to receiver.
- It provides security because each cipher text with one session key is required. So if delay of session key affects on message and network.
- Because of that security manager must try to balance these competing connection oriented protocol, it uses same session key, so each have same life time or life period.
- Session key for the length of time or some amount of time connection is open.
- If connection less protocol each new session key have its own new life time so each session key has different life time or life period. For each exchange.
- So better way is that given session key for a fixed amount of time or period only and only fixed number of transaction.

4. A transparent key control scheme

- This scheme is used in network or transport level. It is useful because it provides end to end encryption.
- The scheme mostly used in connection oriented end to end protocol example is TCP.

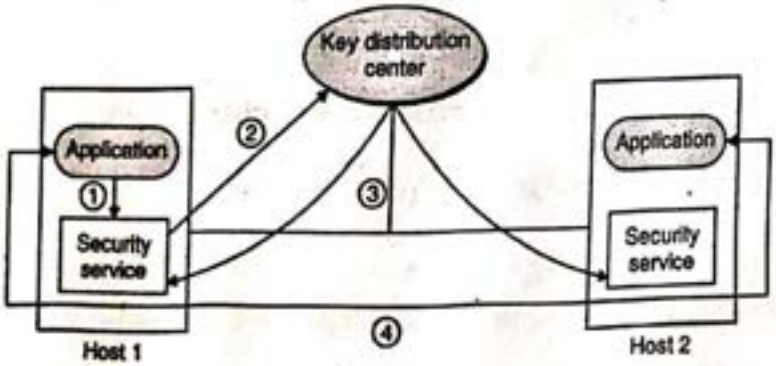


Fig. 5.1.3 : Automatic key Distribution for Connection Oriented Protocol

1. Host send the packet request connection

Scanned by CamScanner

2. Ask for KDC for session
3. KDC distributes session key to both hosts
4. Buffered packet transmitted.

- In Fig. 5.1.3, Host wants to established connection with another host then he sends the request packet the KDC.
- Key distribution center provides the session key to the host using the encryption that key also provide to the another host.
- KDC is encrypted with master key only with the host.

5. Decentralized key control

- This approach requires that each system can communicate in secure manner so there is need to use multiple master's keys for configuration.
- Full decentralization is not possible in practical for large area network.
- So session key will be created as follows :
  1. X sends the request to Y for session key including with original message M_1.
  2. Y replay to the X with original encrypted message using shared mater key. The replay attach with session key selected of $f(M_1)$, another M_2.
  3. Using the new session key x returns $f(M_2)$ to Y.

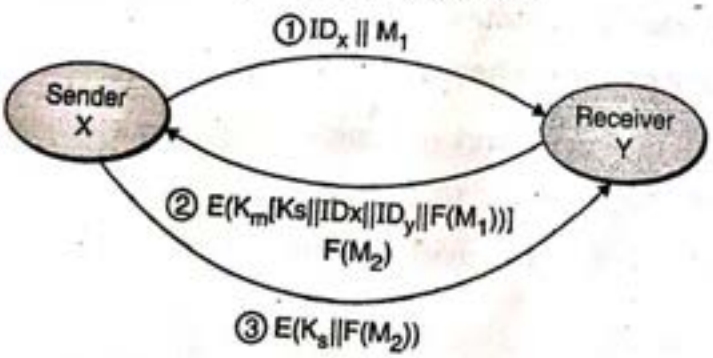


Fig. 5.1.4 : Decentralized Key distribution

- So each terminal contains at most (n – 1) master key as multiple session keys as required.
- Because message transferred using master key are difficult.

Controlling Key Usage

- Using key hierarchy and automated key distribution technique. We can reduce the number of keys.

Scanned by CamScanner

- We use different types of session key like :
  1. **Data Encryption :** Use for communication in the network.
  2. **File Encryption key :** Used for encrypting file which stored for publically available or accessible on the locations.
  3. **PIN Encryption key :** PIN is personal Identification number. It is mostly used for electronic transaction like banking or e-transactions.
- Basically master key is physically secured using cryptographic hardware of key distribution centre.
- Application program uses the session key which is encrypted with the master key.
- There are some limitation to use any key for that purpose of uses tag with session key.
- This proposed technique is used with DES in that it used 8 extra bits in each 64 bit DES key.
- That eight bits are reserved for parity checking from the key tag. That bits are used for following purposes :

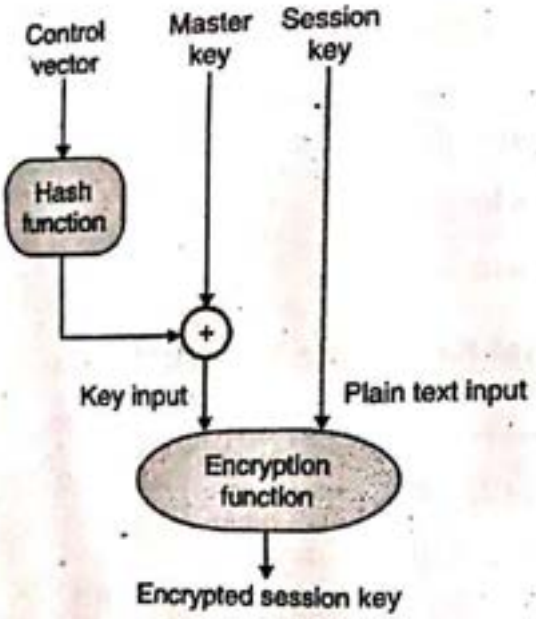


Fig. 5.1.5 : Control Vector Encryption

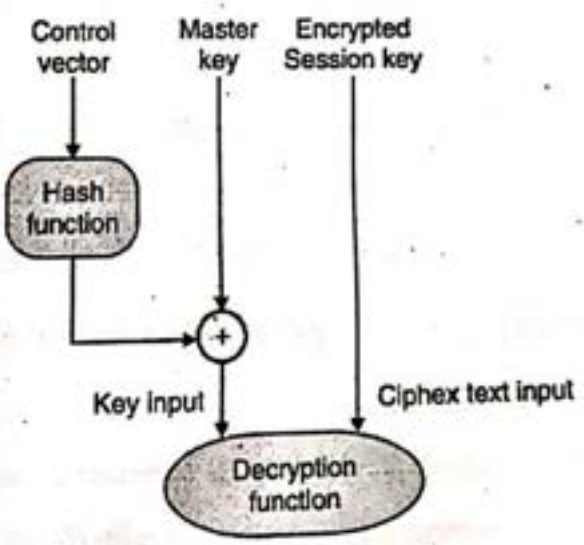


Fig. 5.1.6 : Control Vector Decryption

All the time of key generation control vector is cryptographically coupled with the key. Coupling and decoupling process can be given to Fig. 5.1.5.

- **Step 1 :** Control vector is going through the hash function that produces some value length of that value is equal to the encrypted key.

Scanned by CamScanner

- **Step 2 :** Hash function reduces or maps the value from large input range to the small input range.
- E.g. the number of range is 1 to 100 that can be reduces by 10% so range is now 1 to 10 approximately.
- **Step 3 :** After that Hash value is then XOR with the master key.to produce some output. This output is used as for key input.
- This key input is used for encrypting the session key thus,

$$\text{Hash value} = H = h(cv)$$

$$\text{Key input} = M_k \oplus H$$

$$\text{Cipher text} = E([M_k \oplus H], S_k)$$

Where, M_k is the master key and S_k is session key. For plain text we can recover the session key using following operation:

$D([M_k \oplus H], E([M_k \oplus H], S_k))$

- Recovering of session key we use both master key that user must share with KDC and the control vector. Because of that link between session key and its control vector is maintained.
- There are two main advantages of control vector over the use of an 8 bit tag.
  1. There is no restriction for control vector to its length.
  2. It is available in clear form at all stages of operation.

5.1.2(B) Distribution of Symmetric Key (Secret Key) using Asymmetric Encryption

(1) Simple Secret Key Distribution

(2) Secrete Key Distribution with Confidential and Authentication

(3) Hybrid Schemes

1. Simple Secret Key Distribution

If P wants to communicate with Q, following procedure is used shown in Fig. 5.1.7.

1. P generates a public key/private key pair and transmits a message to Q consists of public key (PU_P) and identifier P (ID_P).

Scanned by CamScanner

2. Q generates a session key K_s and encrypted using P's public key (PU_P) and transmits to P.
3. P decrypts the session key K_s by using its own private key. Now both P and Q know K_s.
4. P discards public/private key and Q discards P's public key.

At the end of communication both P and Q discards K_s. The communication is secure from eavesdropping. The communication becomes unsafe from man-in-middle attack.

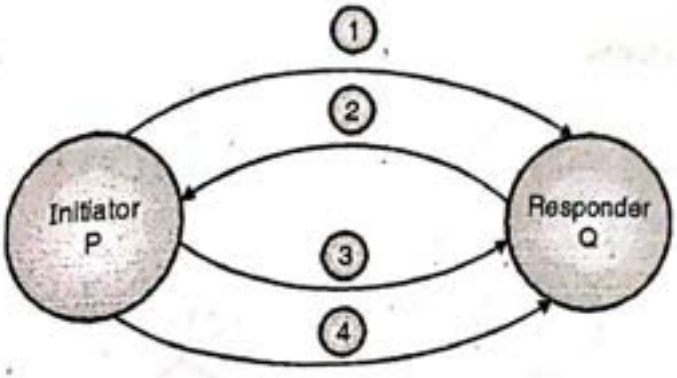


Fig. 5.1.7 : Distribution of Secret Keys using Public-key Cryptography

2. Secret Key Distribution with Confidentiality and Authentication

1. P sends encrypted message using Q's public key consists of identifier of P (ID_P) and none N_1.
2. Q sends an encrypted message using P's public key consists of P's nonce N_1 and Q's nonce N_2.
3. P returns N_2, encrypted using Q's public key to Q.
4. P selects a secret key K_S and send a message $M = E(PU_Q, E(PR_P, K_S))$ to Q.
5. Q decrypts it to recover the secret key.

3. Hybrid Schemes

This technique retains the use of KDC (Key Distribution Center) key Distribution Center that shares a secret master key with each user and distributes secret session keys encrypted with the master key.

Public key is used to distribute the master key.

Scanned by CamScanner

5.1.3 Distribution of Public Keys

Following techniques are used for the distribution of public keys :

1. Public Announcement
2. Publicly Available Directory
3. Public Key Authority
4. Public - key Certificates

1. Public Announcement

- In a public key cryptography, such as RSA, any user can send his or her key to any other user or broadcast it to the group as shown in Fig. 5.1.8.

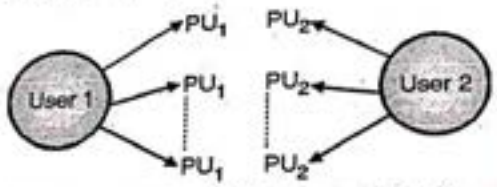


Fig. 5.1.8 : Public Announcement

- This type of approach is having a biggest drawback. Any user can pretend to be a user P and send a public to another user or broadcast it.
- Until user A has got this thing and alerts to other user, a pretender is able to read all encrypted messages for user P.

2. Publicly Available Directory

- A dynamic publically available directory is used to achieve the security. Maintenance and distribution of public directory is controlled by a trusted entity.
- This technique is explained as follows and shown in Fig. 5.1.9.
  - (a) A trusted entity maintains a directory for each user as < name, public key >.
  - (b) Each user has to register a public key with the directory.
  - (c) A user can replace the existing key with a new one at any time for any particular reason.
- It is more secure than public announcement but still having some weaknesses. A hacker can obtain the private key of directory or tamper with the information kept by directory.

Scanned by CamScanner

Fig. 5.1.9 : Publicly Available Directory

3. Public Key Authority

- It gives stronger security. As shown in the Fig. 5.1.10, a central authority keeps a dynamic directory of public keys of all users. Additional, each user knows the public key of authority.
- Working of this techniques explained by following steps :

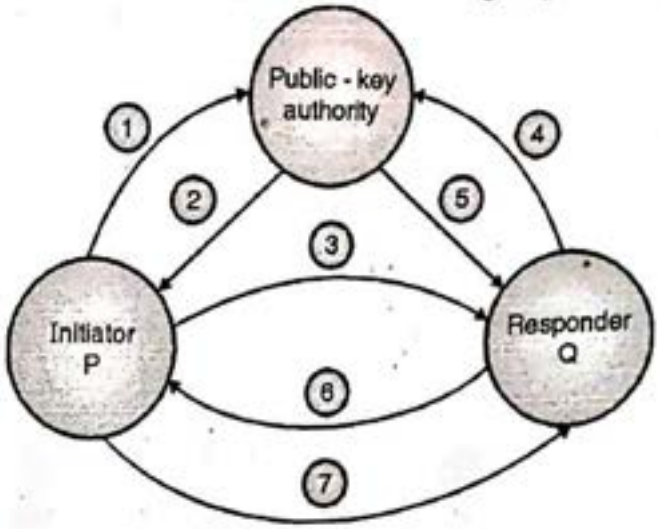


Fig. 5.1.10 : Public Key Authority

Steps

1. P sends a time stamped request to authority for public key of Q.
2. A authority sends an encrypted message. A message is encrypted using authority's private key, so that P can be decrypt it by using authority's public key. A message includes Q's public key which can be use by P for the encryption, the original request sends by P with the time stamp.
3. P uses Q's public key for an encryption and send an encrypted message to Q. An encrypted message contains the Identity of A (ID_P) and nonce (N_1), which is used to identify the transmission uniquely.

4& 5 Q retrieves P's public key from the authority similarly like 1 and 2.Now P and Q start the communication. Two additional steps may be requiring.

Scanned by CamScanner

6. Q sends an encrypted message to P. A message encrypted using P's public key which contains P's nonce N_1 and Q's nonce N_2.
7. P in return sent an encrypted message using Q's public key which contains Q's nonce N_2 to ensure Q that its correspondent is P.

4. Public - Key Certificates

- A public key authority has some drawbacks in the system.
- If a user has to communicate with authority for a public key for every other user that he wishes to contact. This approach provides certificates to users for exchanging the keys among them without contacting to the authority.
- The data and public key maintained by authority itself may be vulnerable to the tampering.
- The certificate authority shown in Fig. 5.1.11 is a government agency or trusted agency.

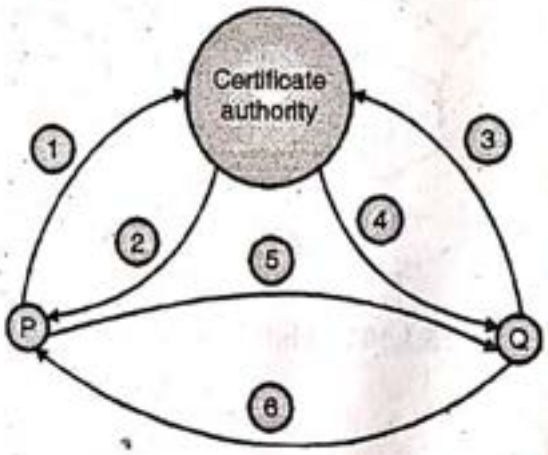


Fig. 5.1.11 : Public - Key Certificates

Steps

1. User 'P' sends its public key to the certificate authority to get the certificate.
2. A certificate authority sends a certificate in a encrypted form using its private key. A certificate consists of timestamp, identifier of P (ID_P) and public key of P.

3 &4. Q retrieves its certificate from the authority similarly explained in 1 and 2.

5. P publishes its certificate Cp to anyone (Q) needed P's public key.
6. Similarly Q also publishes its certificate C_Q to anyone (P).

Then the communication between P and Q is started.

Scanned by CamScanner

5.1.4 Key Generation, Distribution, Storage and Usage

5.1.4(A) Key Generation

- Key generation is the process generating keys using symmetric or asymmetric key cryptography.
- The key can be generated using random or pseudo-random key bit generator which uses the functions of passwords and PINs.
- It is standard (ANSI X9.17) way to generate pseudorandom DES keys.

5.1.4(B) Key Distribution

- Key Distribution also called as Key Transportation is the process by which keys are securely distributed from the location where they are generated or stored to where there be use of it then transmitted using manual transmission methods (e.g., file transfer, key loaders), automated methods (e.g., key transport and/or key agreement protocols), or a combination of automated and manual methods.
- Secure key distribution is the basic aim to achieve the integrity and trust in any cryptographic system. The goal of any key distribution process is :
  - To ensure that the keys are distributed to intended recipients only.
  - To ensure that the keys sent are the correct keys and sent to correct recipient.
  - To ensure that the keys achieving authentication and protection of recipient.
  - To ensure that it is not modified during transmission.
  - To ensure that transmission mechanism is secure and intact.

5.1.4(C) Key Storage

- After successful generation of key, it should not store on multi-user machine unless they are in encrypted or secured form (e.g., temper-resistant security module) (name of the physical storage device).
- It is important to define the key storage for data management of cryptographic key; the proper use of the function depends on the key type, protection requirements and lifecycle stage. There are different function provide by key storage :

1. Operational Storage
2. Backup Storage
3. Archive Storage

Scanned by CamScanner

1. Operational Storage

- If the key is required for operational purpose that keys can be taken from operational storage when not present in active memory (the memory which is presently using). If the key stored in operational storage is lost or corrupted, then that key must be recovered from backup storage.
- Certain key types such as PIN, master keys are always stored in physical hardware and never on software system. The operational storage device uses a key, such as encryption key store on local hard disk of a server which is directly connected to the network.
- Once the key is stored in a database, the database administrator should not access to the keys in the clear text form. If database administrator stolen the key, he/she should not use it to read the encrypted data. Apart from this, storage media must be protected by strong physical and logical security such as dual control and rigorous access logs.

2. Backup Storage

- In case of hardware or software failure if keys are lost then there should be strong back up mechanism required to recover the lost data or corruption of the operational storage. The backup of only the important keys are stored.
- Finally, the backup determination depends on key usage whether the application is important to use the key.

3. Archive Storage

- A Key Archive is the term used to recover the large historical encrypted data.
- The data backup storage keeps the data of today whereas archive storage addresses the data management of tomorrows or overcome the future challenges of data storage management.

5.1.4(D) Key Usage

- The important aspect of key management is key usage. Whether authorized user using key securely or whether they are properly maintaining the privacy and integrity of the key.
- Separate key should be used for encryption or digital signature to prevent the attacker who gain the access of keys and compromise the security.
- Means whatever the key used for encryption it should not use for any other purpose otherwise it usage is not per the rules and regulations.

Scanned by CamScanner

5.1.4(E) Key Validation

- Once key get generated, stored and properly used, then it time to validate. The application which is performing the encryption on data should validate the key, whether the encryption process is able to recover the data prior to placing particular application into production.
- The key validation detects the malfunction of the encrypted data and assures recovery of the encrypted data. The key validation is used to encrypt the data encrypt the data and compare the results with originally encrypted data.
- In-spite of this if attacker modifies or underlying structure then it should retest the encryption process.

5.1.4(F) Key Updation

A function performed on a cryptographic key in order to compute a new, but related key for the same purpose.

5.1.5 Importance of Key Management

- Because of strong key management privacy of customers' personal and corporate information is maintain (confidentiality);
- Preventing data modification (data integrity);
- Authentication of users was done prior to transaction over insecure channel (authentication);
- Replacing handwritten signatures with the electronic equivalent signature means every transaction gets signed.

Syllabus Topic : Diffie Hellman Key Exchange Algorithm

5.2 Diffie Hellman Key Exchange

→ (MU - Dec. 15, May 17)

Q. 5.2.1	Explain how a key is shared between two parties using playfair Diffie Hellman key exchange algorithm. What is the drawback of this algorithm? (Ref. sec. 5.2)	Dec. 15, 10 Marks
Q. 5.2.2	Explain "Diffie-Hellmen" key exchange algorithm with suitable example. Also explain the problem of MIM attack in it. (Ref. sec. 5.2)	May 17, 10 Marks

Scanned by CamScanner

Q. 5.2.3 Illustrate Diffie Hellman key exchange algorithm with suitable example. (Ref. sec. 5.2)

Q. 5.2.4 In what way, the Diffie Hellman key exchange algorithm, prone to man in the middle attack. (Ref. sec. 5.2)

Q. 5.2.5 Briefly explain Diffie Hellman key exchange. (Ref. sec. 5.2)

- The Diffie Hellman algorithm was widely known as Key exchange algorithm or key agreement algorithm developed by Whitfield Diffie and Martin Hellman in 1976. Diffie Hellman algorithm is used to generate same (symmetric) private cryptographic key at sender as well as receiver end so that there is no need to transfer this key from sender to receiver.
- Remember that Diffie Hellman algorithm is used only for key agreement not for encryption or decryption of message. If sender and receiver want to communicate with each other they first agree on the same key generated by Diffie Hellman Algorithm later on they can use this key for encryption or decryption. Let us start with the algorithm.

Steps of Diffie Hellman Algorithm

1. The first step is that if Ramesh wants to communicate with Suresh they must agree on two large prime numbers p and q.
2. Ramesh selects another secret large random integer number a, and calculate R such that

$$R = q^a \bmod p$$

3. Ramesh sends this R to Suresh.
4. Suresh independently selects another secret large random integer number b, and calculate S such that.

$$S = q^b \bmod p$$

5. Suresh sends the number S to Ramesh.
6. Now Ramesh is calculating his secret key by using $R_K = S^q \bmod p$
7. Suresh is calculating his secret key S_K by using

$$S_K = R^b \bmod p$$

8. If $R_K = S_K$ then Ramesh and Suresh can agree for future communication called as key agreement algorithm.
9. We have $R_K = S_K = K$ hence proved. (K is called symmetric key).

Scanned by CamScanner

For example

1. Ramesh and Suresh are agree on two large prime numbers say $p = 17$ and $q = 7$.
2. Ramesh selects another secret large random number 5 i.e. $a = 5$ and calculate R such that

$$R = q^a \bmod p = 7^5 \bmod 17 = 11$$
$$= (7 \times 7 \times 7 \times 7 \times 7) \bmod 17 = 11$$

3. Ramesh sends number R to Suresh.
4. Suresh selects another secret large random number 3. i.e. $b = 3$ and calculate s such that

$$S = q^b \bmod p = 7^3 \bmod 17 = 3$$
$$= (7 \times 7 \times 7) \bmod 17 = 3$$

5. Suresh sends number S to Ramesh.
6. Ramesh now calculates its secret key R_K as follows :

$$R_K = S^a \bmod p = S^5 \bmod 17$$
$$\therefore \quad R_K = 3^5 \bmod 17 = 5$$
$$= (3 \times 3 \times 3 \times 3 \times 3) \bmod 17 = 5.$$

7. Suresh is calculating his secret key S_K as follows :

$$S_K = R^b \bmod p = R^3 \bmod 17$$
$$= 11^3 \bmod 17 = 5$$

8. If $R_K = S_K$ then Ramesh and Suresh can agree for future communication.
9. We know that if $R_K = S_K = K = 5$. Hence proved.

Ex. 5.2.1

Solve if $p = 7$ and $q = 17$ using Diffie Hellman algorithm. Select $a = 6$, $b = 4$.

Soln. :

By using Diffie Hellman algorithm

1. Ramesh and Suresh are agree on two large prime numbers say $p = 7$ and $q = 17$.
2. Ramesh selects another secret large random number 6 i.e. $a = 6$ and calculate R such that

$$R = q^a \bmod p = 17^6 \bmod 7$$
$$= (17 \times 17 \times 17 \times 17 \times 17 \times 17) \bmod 7$$
$$R = 1$$

3. Ramesh sends R to Suresh.

Scanned by CamScanner

4. Suresh selects another secret large number 4 i.e. b = 4 and calculate S such that

$$S = q^b \bmod p = 17^4 \bmod p$$
$$= (17 \times 17 \times 17 \times 17) \bmod 7$$
$$S = 4$$

5. Suresh sends number S to Ramesh
6. Ramesh now calculates it's secret key R_K as follows :

$$R_K = S^a \bmod p = S^6 \bmod p = 4^6 \bmod 7$$
$$= (4 \times 4 \times 4 \times 4 \times 4 \times 4) \bmod 7$$
$$R_K = 1$$

7. Suresh is calculating his secret S_K as follows :

$$S_K = R^b \bmod p = 1^4 \bmod 7$$
$$S_K = 1$$

8. If $R_K = S_K$ then Ramesh and Suresh can agree for future communication.

Ex. 5.2.2

Solve p = 353 and q = 3, a = 97 and b = 233.

Soln. :

By using Diffie Hellman algorithm

1. Ramesh and Suresh are agree on two large prime numbers p = 353 and q = 3.
2. Ramesh selects another secret large random number a = 97 and calculate R such that

$$R = q^a \bmod p = 3^{97} \bmod 353$$
$$R = 40$$

3. Ramesh sends R to Suresh (value of R = 40)
4. Suresh selects another secret large random number b = 233 and calculate S such that

$$S = q^b \bmod p = 3^{233} \bmod 353$$
$$S = 248$$

5. Suresh sends value of S to Ramesh
6. Ramesh now calculates it's secret key R_K as follows :

$$R_K = S^q \bmod p = 248^{97} \bmod 353$$
$$R_K = 160$$

Scanned by CamScanner

7. Suresh is calculating his secret key S_K as follows :

$$S_K = R^b \bmod p = 40^{233} \bmod 353$$

$$S_K = 160$$

8. If $R_K = S_K$ then Ramesh and Suresh can agree on same key and can communicate in future.

Ex. 5.2.3

If generator g = 2 and n or p = 11 using Diffie Hellman algorithm solve the following :

(i) Show that 2 is primitive roof of 11

(ii) If A has public key 9 what is A's private key ?

(iii) If B has public key 3 what is B's private key ?

(iv) Calculate shared secret key.

Soln. :

(i) To show that 2 is primitive root of 11

In general terms, the highest possible exponent to which a number can belong (mod n) is Q(n).

Where Q(n) is called as Euler totient function which states that how many numbers are between 1 and n – 1 that are relatively prime to n.

According Euler's theorem

It states that for every a and n that are relatively prime.

If $a^{\phi(n)} = 1 \bmod n$

For example

As mention in (i)

a = 2 and n = 11

Calculate $\phi(n)$ i.e. $\phi(11) = \{ 1 \text{ to } 10 \} = 10$

According to Euler's theorem

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

$$2^{10} = 1 \bmod 11$$

$$1024 = 1 \bmod 11$$

i.e. $1024 \bmod 11 = 1$ and

Scanned by CamScanner

$1 \bmod 11 = 1$

Hence 2 is primitive root of 11.

(ii) If 'A' has public key 9 then private key is

1. Say A as Ramesh and B as Suresh.

   Representing g as a q (i.e. g = 2 = q)

   Using Diffie Hellman algorithm

2. Suresh now calculates R such that

$$R = q^a \bmod p \quad [\text{Here } q = 2 \text{ and } p = 11]$$

$$= 2^9 \bmod 11 \quad [\text{a is 9 public key}]$$

$$R = 6$$

3. Ramesh now sends R to Suresh.

(iii) Suresh has public key 3 (it's random number) Suresh calculates S such that

$$S = q^b \bmod p = [q = 2, p = 11, b = 3]$$

$$S = q^b \bmod p = 2^3 \bmod 11$$

$$S = 8$$

Suresh now sending S to Ramesh.

(iv) Now Ramesh and Suresh calculating their secret keys individually.

1. Ramesh calculates it's secret key R_K as follows :

$$R_K = S^a \bmod p \quad [S = 8, p = 11, a = 9]$$

$$= 8^9 \bmod 11 = 7$$

2. Suresh calculates it's secrete key S_K as follows :

$$S_K = R^b \bmod p \quad [R = 6, b = 3, p = 11]$$

$$= 6^3 \bmod 11 = 7$$

Shared secret keys of Ramesh and Suresh are 7.

Hence, $\mathbf{R_K = S_K = 7}$

Note : In above example we have solved by using our notations mentioned in Diffie Hellman algorithm. So here A denotes Ramesh, B denotes Suresh, g denotes q and p is same as p. Students can use any notations.

Scanned by CamScanner

Ex. 5.2.4

A and B decide to use Diffie Hellman key exchange where p = 13, g = 2, each choose his own secret no. and exchange numbers 6 and 11.

(i) What is common secret key ?

(ii) What are their secret numbers ?

(iii) Can intruder m gain any knowledge from protocol run if he sees p, 9 of and two keys 6 and 11. If yes, show how ?

Soln. :

According to Diffie Hellman algorithm,

Let us say A as Ramesh and B as Suresh

Also p = 13 and g = 2

Here in our example we are denoting g as q,

$$\therefore \; p = 13, \qquad q = 2$$

Secret numbers denoted as, a = 6 and b = 11 by using Diffie Hellman algorithm.

1. Ramesh and Suresh agree on two large prime numbers p = 13 and q = 2.
2. Ramesh selects another secret no. a = 6 and calculate R such that

$$R = q^a \bmod p \; [q = 2, a = 6, p = 13]$$
$$= 2^6 \bmod 13$$
$$R = 12$$

3. Ramesh sends R to Suresh
4. Suresh selects another large random number b = 11 and calculate S such that

$$S = q^b \bmod p \quad [q = 2, b = 11, p = 13]$$
$$= 2^{11} \bmod 13$$
$$S = 7$$

5. Suresh sends S to Ramesh
6. Ramesh now calculates it's secret key R_K as follows :

$$R_K = s^a \bmod p \; [S = 7, a = 6, p = 13]$$
$$= 7^6 \bmod 13$$
$$R_K = 12$$

Scanned by CamScanner

7. Suresh is calculating his secret key S_K as follows :

$$S_K = R^b \bmod p \; [R = 12, b = 11, p = 13]$$

$$= 12^{11} \bmod 13$$

$$S_K = 12$$

(i) Shared secret key of Ramesh and Suresh is

$$R_K = S_K = 12 \; [A \text{ and } B = 12]$$

(ii) Secret numbers of Ramesh and Suresh are

$$R = 12 \text{ and } S = 7$$

(iii) If intruder m knows p, g and a, b then what will happen. [Here g = q]

Case I : Value of p, q, a, b are known to m represented as,

Ramesh	m	Suresh
p = 13, q = 2	p = 13, q = 2	p = 13, q = 2

Use Diffie Hellman algorithm,

After selecting large prime numbers, it's time to select random numbers a and b.

The secret random number selected by Ramesh and Suresh are,

Ramesh	m	Suresh
a = 6	a = 8, b = 6	b = 11

Case 2 :

Consider m as intruder selected two random numbers say a = 8 and b = 6 as his own secret key, because he wants to calculate value as R and S, as he intercepted conversion between Ramesh and Suresh

Ramesh	Intruder m	Suresh
$R = q^a \bmod p$	$R = q^a \bmod p$	$S = q^{13} \bmod p$
$= 2^6 \bmod 13$	$= 2^8 \bmod 13$	$= 2^{11} \bmod 13$
$= 12$	$R = 9$	$S = 7$
	$S = q^b \bmod 13$	
	$= 2^6 \bmod 13 = 12$	

Scanned by CamScanner

Case 3 : Following are the values available with Ramesh, Suresh and intruder m

Ramesh	intruder m	Suresh S
R = 12	R = 9, S = 12	S = 7

Case 4 :

Ramesh sending his R = 12 to Suresh but intruder m sending his own R = 9 to Suresh instead of R = 12. Suresh sending his S = 7 to Ramesh, here again intruder m sending his own value of S = 12 to Ramesh. In this case Ramesh and Suresh doesn't aware that which values they are sending and receiving [Intruder m sending his own value Because of his interception]. Following are the new values with Ramesh, Suresh and intruder m.

Ramesh	Intruder m	Suresh
R = 12, S = 12	R = 12, S = 7	S = 7
		R = 9

Case 5 : Based on above values Ramesh, Suresh and Intruder m calculating secret keys.

Ramesh	Intruder m	Suresh
S = 12, a = 6,	S = 7, R = 12	R = 9, b = 11
p = 13	a = 8, b = 6, p = 13	p = 13
$R_K = S^a \bmod p$		
$= 12^6 \bmod 13$	$R_K = S^a \bmod p$	$S_K = R^b \bmod 11$
$R_K = 1$	$= 7^8 \bmod 13$	$= 9^{11} \bmod 11$
	$= 3$	$S_K = 3$
	$S_K = R^b \bmod p$	
	$= 12^6 \bmod 13$	
	$= 1$	

- Think what is happening ? Ramesh is thinking that value of his secret key is 1 and Suresh also thinking that value of his secret key is 3.
- But actual communication is intercepted by intrude m.

Scanned by CamScanner

- During real communication between Ramesh and Suresh intruder m sending his own secret keys to Ramesh and Suresh. If Ramesh sending his secret key RK = 1 to Suresh because of man-in-the-middle attack. Intruder m sending his secret key RK = 3 to Suresh. In return Suresh is sending his secret key SK = 3 to Ramesh, intruder m sending his secret key SK = 1 to Ramesh.
- Both Ramesh and Suresh not aware that communication intercepted by intruder m such type of attack is called as man-in-the-middle attack.

Chapter Ends...

❑❑❑

Scanned by CamScanner

CHAPTER 6

Module 3

Cryptographic Hash Functions

Syllabus

Cryptographic hash functions, Properties of secure hash function, MD5, SHA-1, MAC, HMAC, CMAC.

6.1 Hash Functions

Syllabus Topic : Cryptographic Hash Functions

6.1.1 Cryptographic Hash Functions

- In **hash function H** accepts a variable length block of input data called as 'M' and produces the fixed size hash value can be represented as

$$h = H(M)$$

- A good has function provides a property that hash function is applied on large amount of data input (M) and then it produces the fixed amount of output data.
- If any bit or bits changes in the data, then whole hash function output data will also change.
- When hash function provides security application this is called cryptographic hash functions.
- Cryptographic has function is one of the algorithm which is computationally infeasible.
- Input is padded with some fixed out of length (For e.g. 1024 bits). It includes original message in bits and padding bits.

Scanned by CamScanner

- Because of these securities will provided, the length field produces difficulty for the attacker to create alternative message with the same hash value.

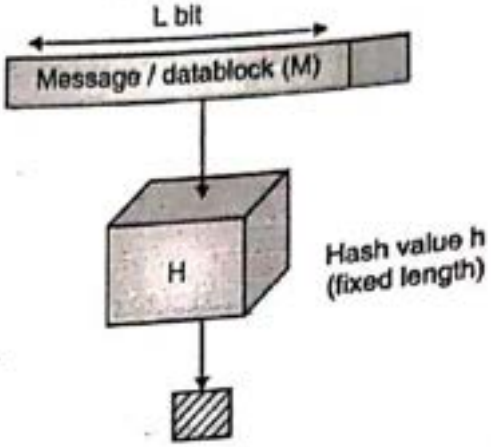


Fig. 6.1.1 : Block Diagram of Cryptographic Hash Function

- Cipher Block chaining is mostly used for to implementing cryptographic Hash function, whirlpool is another popular cryptographic hash function.
- Cryptographic hash algorithm is applied on original message when padding is done.
- In padding fix variable length input is added like 1024 bits.
- After this hash function algorithm is applied so it creates a Hash value h.

6.1.2 Applications of Cryptographic Hash

Most adaptable cryptographic algorithm is the cryptographic hash function. It is mostly used in Internet protocol and security application.

Following are some application which are used in Cryptographic Hash Function :

1. Message Authentication

- Message Authentication is a nothing but service that verify or checks the integrity of that message.
- Message Authentication checks that the data received from sender which is exactly same as sent to receiver. (i.e. content of the data will be same no modification will have done or insertion deletion or updation).
- Some authentication provides a mechanism that checks identity of sender is valid or not.
- When Hash Function provides a value, which is used for message authentication purpose then that value is invoked as Message Digest (MD).

Scanned by CamScanner

The Fig. 6.1.2(a), describes how many approaches in which a hash code can be helps to provide the message authentication.

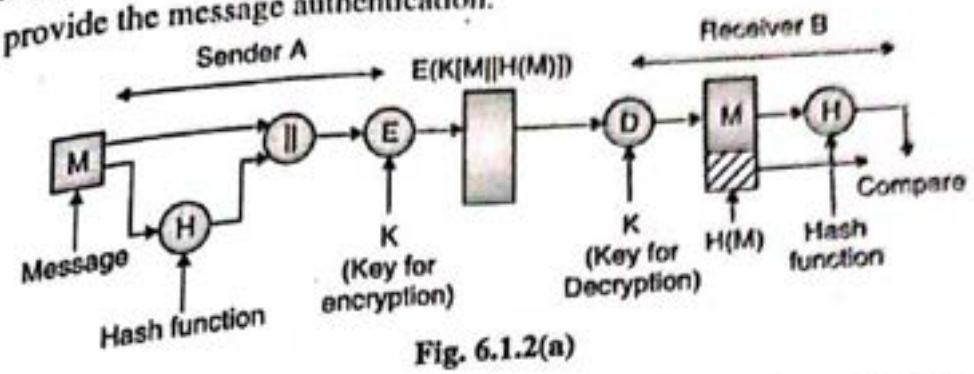


Fig. 6.1.2(a)

(a) Message added with hash code that encrypted using symmetric encryption. Sender A and receiver B shares the same secret key, A message is come from sender A that must not be changed. After that hash code provides a security or structure to active authentication.

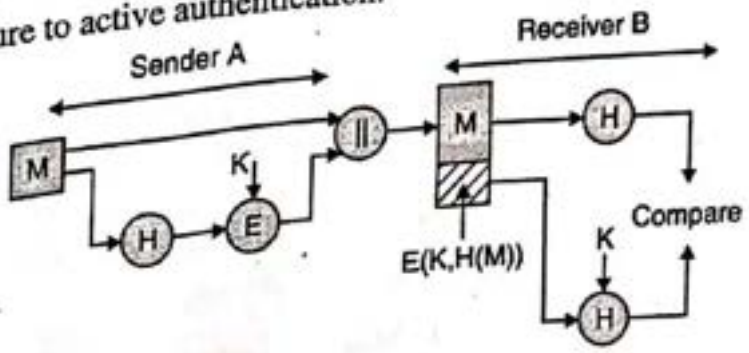


Fig. 6.1.2(b)

(b) Hash code is encrypted using symmetric key encryption. This minimizes the load for those applications that don't required any confidentiality.

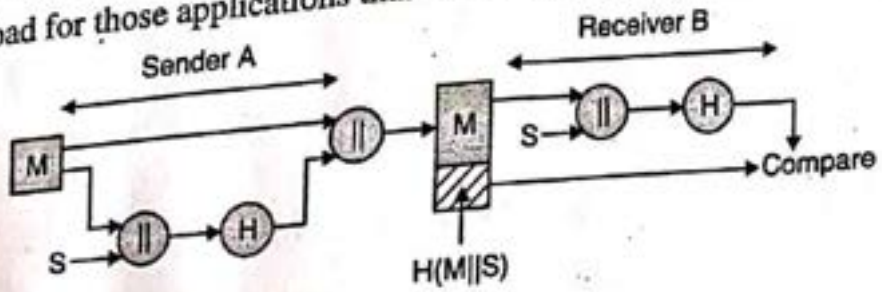


Fig. 6.1.2(c)

(c) No encryption required for message authentication only use hash function. In this process two person shares the same secret key when they communicate with each other.

Sender A calculates the hash value over adding the original message M and S and attach the result of hash value to message M.

Scanned by CamScanner

Because receiver B processes S and it can recalculate the hash value to check it.

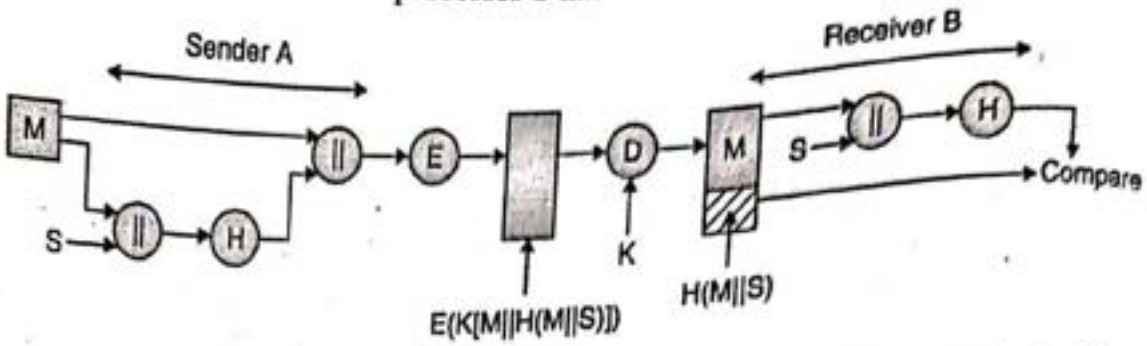


Fig. 6.1.2(d) : Simplified Examples of the use of a Hash Function for Message Authentication

(d) Confidentiality is added to this process and process (c) is encrypted with entire message concatenate with hash code.

- Process (b) has more advantage over process (a) and process (d) which encrypts the whole data or message.
- Minimum calculation is required in process (b).
- Using Message Authentication Code (MAC) message authentication is achieved. This is also called as a keyed hash function.
- MAC are used between two persons for communication that shares the same secret key to authenticate information transfer between two users.
- A MAC function takes input as data block and secret key and produce output as hash value.
- MAC function is applied to the original message and the produced result will be check or compare with stored MAC value.

2. Digital Signature

- It is another important application this digital signature is similar to the message authentication application.
- Operation of the digital signature is same as that of the MAC.
- In this approach message with hash value is encrypted with the user's private key. If anyone wants to check the integrity of the message with its digital signature when he knows the users public key only.
- For creating a one-way password file we use the hash functions.
- Hash Function password is stored in the operating system so hacker cannot access that actual hash value.

Scanned by CamScanner

- When user enters a password the hash value of the password is compared with the stored hash value for verification purpose.
- Hash function is also used for intrusion detection as well as virus detection.
- One system stores the H (F) for each file.

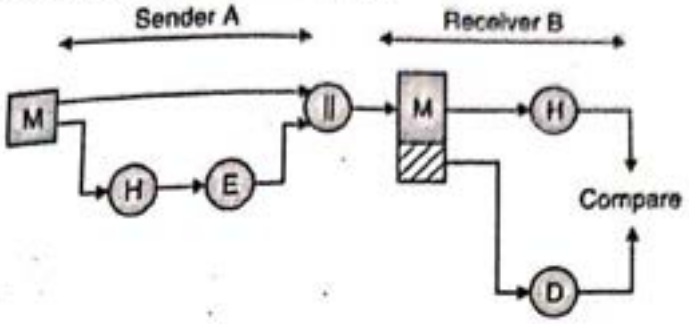


Fig. 6.1.3(a) : Simplified Example of Digital Signatures

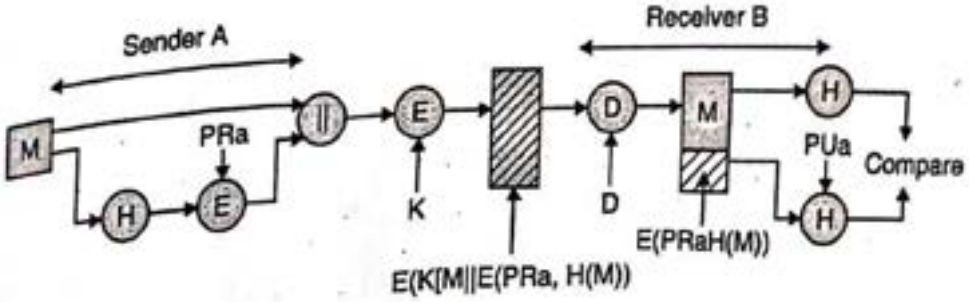


Fig. 6.1.3(b) : Simplified Example of Digital Signatures

- If attacker want to change or modify the message it is possible when he knows the users private key.
- Hash code is used to provide a digital signature is as follows :
- Hash code is encrypted with public key encryption the sender's private key. This is used to provide the digital signature as well as authentication. Because only sender can create a hash value that was encrypted.

3. Other Application

- To creating one way password file we commonly used Hash function. The password will stored on operating system because of that hacker cannot access the data or that password file.
- To construct the pseudo random function cryptographic hash function is used. Pseudo Random Number (PRNG) is another example. A common application is the PRF for creating symmetric key.

Scanned by CamScanner

6.2 Simple Hash Functions

- The process of transforming input message *m* into a fixed size string (called as hash value *h*) is called as hash function and it is denoted by *H*. Here *h* is the output of hash function applied on input message *m*.

$$h = H(m)$$

- Hash function protects the integrity of the message. If attacker tries to modify the original message, then the contents of original message may changed it can be identifies by applying Hashing algorithm. The most popular hashing algorithms are MD5 and SHA.
- Here, there are two simple hash function, all hash functions are operate using same principle.
  (1) The message file is like a simple input it open a sequence of n-bit blocks.
  (2) When input is processed only one block at the given time in iterative fashion to generate an n-bit hash function.
- The simple hash function is the bit-by-bit XORing done of every block.
- This can be shows the following ways :

$$CH_i = B_{i1} \oplus B_{i2} \oplus \ldots\ldots \oplus B_{im}$$

Where,

CH_i = is i^{th} bit of hash code, $1 \le i \le n$

m = m is the number of n-bit block in the input

B_{ij} = i^{th} bit in j^{th} block

$\oplus$ = XORing operation

- When this operation is perform it produces a simple parity for each bit location this process is known as a longitudinal redundancy check.
- In simple hash function each n-bit hash function value is equally possible. Thus, the probability of that data error will result should be in unchanged hash value 2 – n including the more predictably formatted data, and this function are less effective.
- We give the input as a simple or normal text files or message, the higher order bit of each octet is constantly zero.

Scanned by CamScanner

- When 128-bit hash value is used, rather effectiveness of 2 – 128 the hash function on this types of files or message has an effectiveness of 2 – 112.
- To improve the matter performance, use simple ways i.e. one-bit circular shift, and also the rotation on hash value after the every block is processed.
- The steps are summarized by as follows :

1. Set the n-bit hash value initially zero.
2. Process the each and every successive n-bit block of data is as follows :
   (a) The current hash value is rotate to the left by one bit.
   (b) Perform the XOR operation in between block and hash value.

- The effect of "randomizing" the more completely input and any regularities are overcome that appear in the input.
- If the hash code is encrypted, it is insufficient for simple XOR or rotated XOR.
- These techniques are proposed by the National Bureau of standards. They used the simple XOR function and applied to 64 -bit block of the message and then encrypted with entire message used the cipher block chaining i.e. (CBC) mode.
- We can define the design are as follows :

(1) The message M are consist a sequence of 64-bit blocks $A_1, A_2, A_3 \ldots A_N$.

(2) Define hash code h = H(M) as block-by-block XOR of every block and attach the hash code as the final blocks.

$$h = A_{N+1}$$
$$= A_1 \oplus A_2 \oplus \ldots\ldots \oplus A_N$$

Next,

Using CBC mode encrypt the whole message plus hash code to produce the encrypted message $B_1, B_2, B_3 \ldots\ldots B_{N+1}$

For example, the definition of cipher block chaining, we get

$$A_i = B_{i-1} \oplus D(K, B_i)$$

$$A_{N+1} = B_N \oplus D(K, B_{N+1})$$

But, Here A_{N+1} is the hash code :

$$A_{N+1} = A_1 \oplus A_2 \oplus \ldots\ldots \oplus A_N$$
$$= D(K, B_1) \oplus B_1 \oplus D(K, B_2) + .. \oplus B_{N-1} \oplus D(K, B_N)$$

Scanned by CamScanner

Syllabus Topic : Properties of Secure Hash Function

6.2.1 Properties of Hash Function

→ (MU - Dec. 17)

Q. 6.2.1 What are the properties of hash functions? (Ref. sec. 6.2.1) **Dec. 17, 3 Marks**

There are four main properties of has function :

1. It is quick to compute the hash value for any given message.
2. It is infeasible to generate a message from its has value except by trying all possible messages.
3. A small change to a message should change the hash value so extensively that the new hash value appears uncorrelated with the old hash value.
4. It is infeasible to find two different message with the same has value.

6.2.2 Characteristics of Simple Hash Function

Characteristics are needed in a secure has function are as follows :

1. H can be applied to a block of data of any size.
2. H produces a fixed length output.
3. H (x) is relatively easy to compute for any given x, making both hardware and software implementations practical.
4. For any given value h, it is computationally infeasible to find x such that H (x) = h. This is sometime referred to in the literature as the one-way property.
5. For any given block x, it is computationally infeasible to find $y \neq x$ with H (y) = H (x).
6. It is computationally infeasible to find any pair (x, y), such that H (x) = H (y).

6.2.3 Simple Hash Function Requirement and Security

→ (MU - Dec. 17)

Q. 6.2.2 What is the role of a hash function in security ? (Ref. sec. 6.2.3) **Dec. 17, 3 Marks**

Requirement and Security

- Before taking any action, we need to define the two points.
- For a hash value h = H (m), we can say that m is the pre-image of the h. That is m data block whose hash function, using function H, is h.

Scanned by CamScanner

- Here, the collision occurs if $A \neq y$ and $H(A) = H(B)$.
- For region, we use here hash function the data, integrity and also the collisions are clearly undesirable.
- Assume, suppose the length of hash code is n bit, and function H use as input messages or the block of length of bits with $b > n$.
- Here, the total number of possible messages is 2b.
- Total number of hash values is 2^b.

6.2.4 Hash Functions Based on Cipher Block Chaining

- Without using the secret key, the number of proposal have been made for the hash function based on a cipher block chaining.
- The Rabin is proposed first techniques that techniques are working the following ways.
- The message M are divide into the fixed size blocks i.e. $M_1, M_2, \ldots M_N$ use encryption method like DES to calculate hash code following manner ;

$$HF_0 = \text{Initial value}$$

$$HF_1 = En(M_i, HF_{i-1})$$

$$C = HF_N$$

- It is same like a CBC techniques but in this particular condition secret key is not used.
- Using the any hash code, that design is likely to the birthday attack, and if used the encryption algorithm is DES. It produced only 64-bit hash code then the design is the accessible.
- As well, the birthday attack used the another version and if the candidate access only one message and also used it's correct signature and not attain a multiple signings.
- Here used the outline :
- Individually, assume that the candidate prevents a message with a signature in the form of the encrypted hash code and then an encrypted message is m bit long.

1. To calculate the unencrypted hash code C used algorithm which is defined at beginning of the subsection.
2. To design any desired messages in the form of $P_1, P_2, P_3, \ldots P_{N-2}$
3. Count

$$HF_i = E_n(P_i, HF_{i-1}) \text{ for } i \leq (N-2)$$

Scanned by CamScanner

4. Accomplish $2^{m/2}$ any random blocks ; for any block A, compute E_n (A, HF_{N-2}) to accomplish the additional $2^{m/2}$ any random blocks; for any block B, compute D_n (B, C) where the D_n is the decryption function like to E_n.
5. Place on the birthday paradigm, along high probability there will be A and B as $E_n(A, HF_{N-2}) = D_n(B, C)$.
6. From the message $M_1, M_2, M_3, \ldots M_{N-2}$, A, B, this message has the hash code is C and used the prevent encrypted signature.

- This mode of attack is called as meet-in-the- middle-attack.
- The number of researcher are proposed insight intended to strength then the basic block chaining approach.
- Using example of Davies and price are describe change.

$$HF_i = E_n(M_i, HF_{i-1}) \oplus HF_{i-1}$$

another change proposed the

$$HF_i' = E_n(HF_{i-1}, M_i) \oplus M ;$$

- These systems shown to be unsafe variety of attack.
- Many of these attacks have also been shown to have weaknesses.

Syllabus Topic : MD5

6.3 MD5 Message Digest Algorithm

- It was developed by Ron Rivest. This algorithm takes an input of arbitrary length and 128 - bit message digest is produced. The input message is produced in 512 - bit blocks.

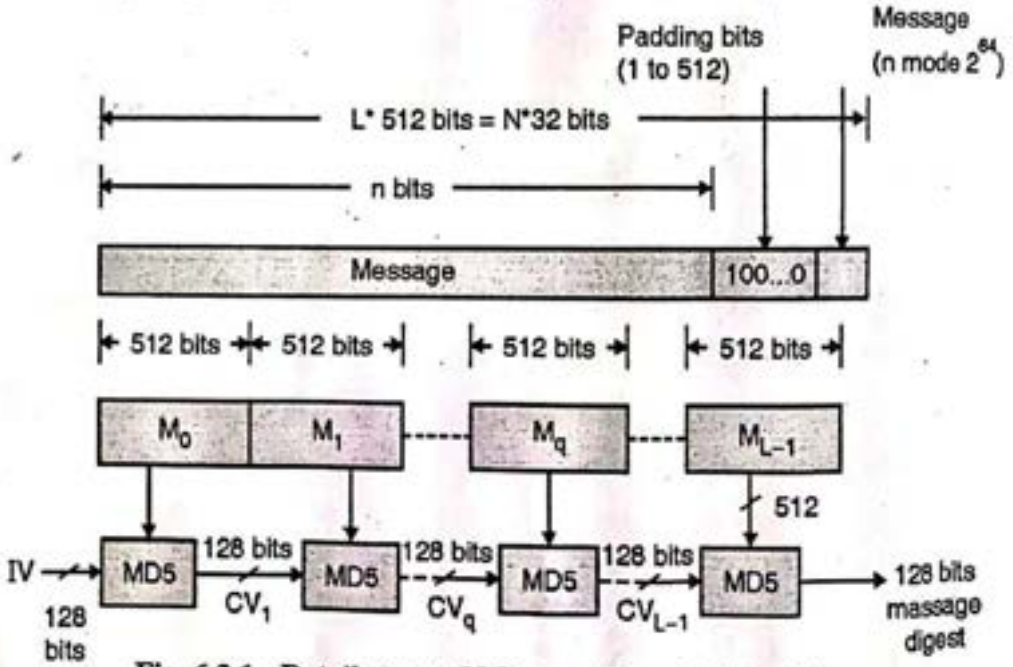


Fig. 6.3.1 : Detail steps of Message Digest 5 Algorithm

Scanned by CamScanner

- Fig. 6.3.1 shows processing of a message to produce message digest. Following steps explains the procedure of MD5.

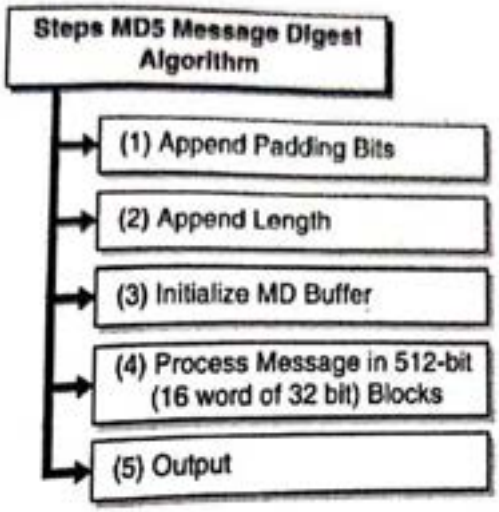


Fig. 6.3.2 : Steps MD5 Message Digest Algorithm

→ **(1) Append Padding Bits**

- The message is padded to make the length of message is 448 mod 512. The length of the padded message is 64 bits less than an integer multiple of 512.
- The padding message consists a single 1-bit followed by 0 bits. The length of padding bits is in between 1 to 512.

→ **(2) Append Length**

- 64 bits of original message is appended to the result of above step 1. It is appended such that least significant bytes to most significant byte.
- The output of step 2 yields a message of integer multiple of 512 bits.
- As $M_0, M_1, \ldots M_q \ldots M_{L-1}$. The total length of expanded message is L * 512 bits.

→ **(3) Initialize MD Buffer**

- A 128 - bit buffer is used to store the intermediate as well as final result. A buffer is represented as four 32-bit registers as P, Q, R, S.

P = 67452301

Q = EFCDA1389

R = 98BADCFE

S = 10325476.

Scanned by CamScanner

- It used a little-endian methods. Hence initial values (IV) are represented as,

$$P = 01\ 23\ 45\ 67$$

$$Q = 89\ AB\ CD\ EF$$

$$R = FE\ DC\ BA\ 98$$

$$S = 76\ 54\ 32\ 10$$

(4) Process Message in 512-bit (16 word of 32 bit) Blocks

- It consists of four rounds of processing as shown in Fig. 6.3.3. These four rounds have similar structure but differ in primitive logical function referred as A, B, C, D.
- Each round takes input 512-bit block, processed it and produces 128 bit output. The output of fourth round is added to the first round CV_q to produce CV_{q+1} using addition modulo 2^{32}.

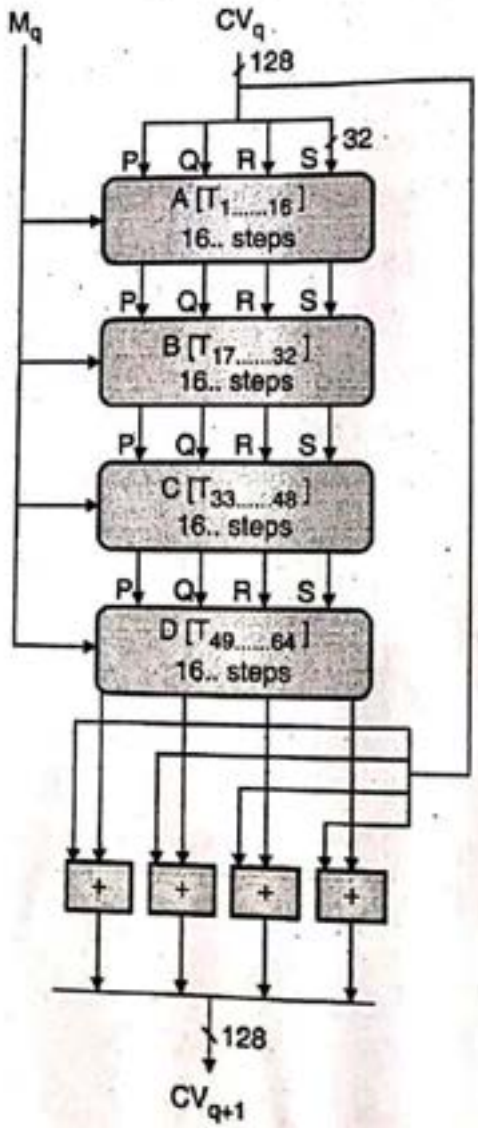


Fig. 6.3.3 : Four rounds of MD5 algorithm

Scanned by CamScanner

→ **(5) Output**

After processing all L 512-bit blocks, the 128 bit message digest is produced as a output.

The entire MD5 process can be summarized as follows :

CV0 = IV

CVq+1 = Sum32(CVq, RFd [Mq,RFc [Mq.,RFb [Mq, RFa [Mq, CVq]]]])

MD5Sum = CVL

Where,

IV = the initial value of the PQRS buffer, mentioned in step 3

Mq = the qth 512-bit block of the message

CVq = the chaining variable processed with the q-th block of message

RF = the round function using primitive logical function a, b, c, d.

MD5Sum = the final hash result or message digest

Sum32 = addition modulo 232

Syllabus Topic : SHA-1

6.4 Secure Hash Algorithm (SHA)

→ **(MU - May 17, May 18)**

Q. 6.4.1	Write note on SHA-1. (Ref. sec. 6.4)	May 17, 5 Marks
Q. 6.4.2	What characteristics are needed in secure hash function? Explain the operation of secure hash algorithm on 512 bit block. (Ref. sec. 6.4)	May 18, 10 Marks

The SHA was developed by NIST in 1993. It is referred as Secure Hash Algorithm-1.

SHA - 1 takes an input message of a maximum length less than 2^{64} bits and produced an output of 160 bit message digest. The overall processing of SHA-1 is much similar to MD5. The processing is explained as follows.

(1) Append Padding Bits

- Padding means addition of bits to the original message. To make length of original massage to a value 64 bits less than multiple of 512. The message is padded to make the length of message 448 mod 512.

Scanned by CamScanner

- The length of the padded message is 64 bits less than an integer multiple of 512. The padding message consists of a single 1-bit, followed by many 0 bits as required. The length of padding bits is in between 1 to 512.

(2) Append Length

- A block of 64-bit is appended to a message. 64 bits of original message is appended to the result of above step 1 (Original message + Padding).
- It is appended such that least significant bytes to most significant byte.

(3) Initialize MD5 Buffer

- A 160-bit buffer is used to store the intermediate as well as final result. The buffer is represented as five 32-bit registers as P, Q, R, S, T, as.

$$P = 67452301$$
$$Q = EFCDAB89$$
$$R = 98BADCFE$$
$$S = 10325476$$
$$T = C3D2E1FO$$

- It uses a big-endian method. First four registers are same as MD5. These five registers P, Q, R, S, T are represented as,

$$P = 67\ 45\ 23\ 01$$
$$Q = EF\ CD\ AB\ 89$$
$$R = 98\ BA\ DC\ FE$$
$$S = 10\ 32\ 54\ 76$$
$$T = C3\ D2\ E1\ FO$$

(4) Process Message in 512-bits (32 bit 16 word) Block

- It consists of four rounds of 20-step each as shown in Fig. 6.4.1. These rounds referred as F1, F2, F3, F4 have similar structure. These rounds used different primitive logical function.
- Each round takes input 512-bit block processed it and produced 160 bit output. The output of fourth round is added to the first round CV_q to produce CV_{q+1}.
- Each round also uses an additive constant k_t, where $0 \le + \le 79$.

$$K_1 = 5A\ 827999$$

Scanned by CamScanner

K_2 = 6E D9 EB A1

K_3 = 8F 1B BC DC

K_4 = CA 62 C1 D6

(5) Output

- After processing all L 512 bit blocks, the 160 bit message digest is produced as output.
- The SHA compression function uses a feed forward operation where the chaining variable CVq input of the first round is added to the output obtained (last step) after execution of 80 steps to produce the next chaining variable CVq+1 as shown in Fig. 6.4.1.

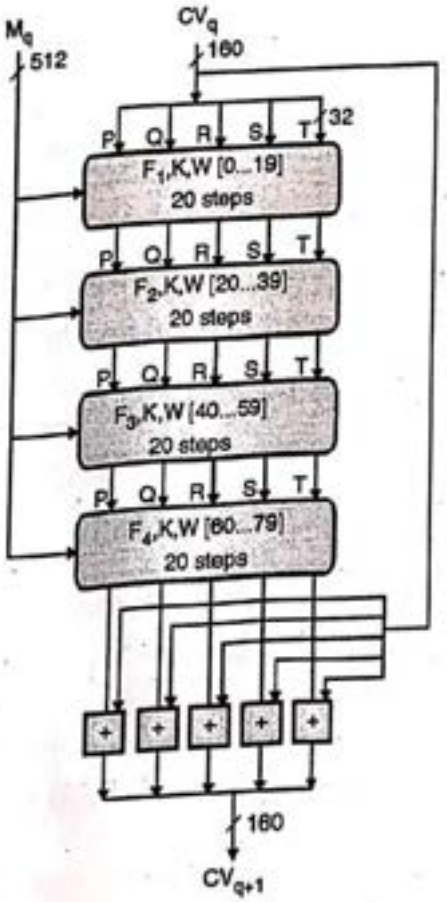


Fig. 6.4.1 : Four rounds of Secure Hash Algorithm

- The entire SHA1 process can be summarized as follows :

CV0 = IV

CVq+1 = Sum32 (CVq, F4 [M60...79, F3 [M40...59, F2 [M20...39, F1 [M0...19,CVq, K0...19], K20...39] ,K40...59] , K60...79])

SHAr = CVL

Scanned by CamScanner

where

IV = initial value of the PQRST buffer, used to deal with the first block in a chaining mode

Mq = the qth 512-bit block of the message

CVq = the chaining variable processed with the q-th block of message

F1[_ _ _] = output of the first round consisting of 20 steps

F2[_ _ _] = output of the second round

F3[_ _ _] = output of the third round

F4[_ _ _] = output of the fourth round

Sum32 = addition modulo 232

SHAr = the final hash result or message digest

6.4.1 Difference between SHA-1 and MD5

→ (MU - Dec. 15, Dec. 16, May 18)

Q. 6.4.2 Differentiate between MD-5 and SHA. (Ref. sec. 6.4.1) **Dec. 15, Dec. 16, May 18, 5 Marks**

Both are derived from MD4. Both are quite similar. They differs from each other in design goals.

Sr. No.	SHA-1	MD5
1.	It uses a 160-bit message digest. Hence it is stronger against Brute - force attacks than MD5.	It uses a 128 bit message digest. Hence it is weaker than SHA1 against Brute - force attacks.
2.	SHA-1 is not vulnerable against cryptanalysis.	MD5 is vulnerable against cryptanalysis
3.	SHA-1 is slower than MD5.	MD5 is faster than SHA-1.
4.	It uses big - endian method to represent the message.	It uses a little endian method to represent the message.
5.	SHA has 20 rounds.	MD5 has 64 rounds.
6.	Bit rotation counts for SHA - 1 are the same for all rounds.	In MD5 each round has its own bit rotation counts.

Scanned by CamScanner

6.4.2 Applications of Cryptographic Hash Functions

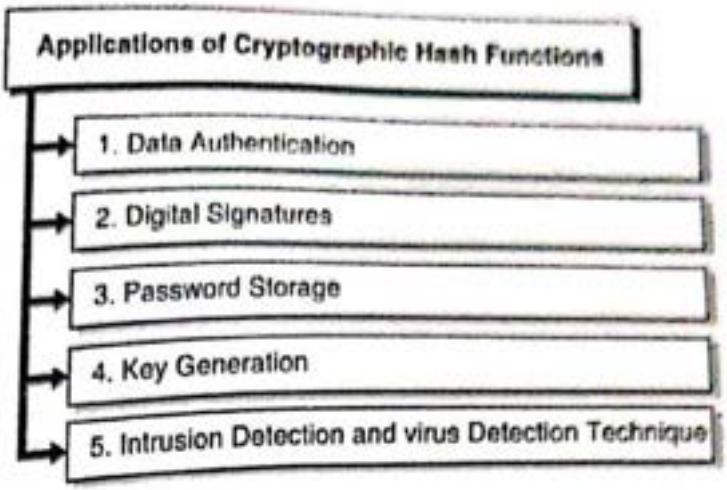


Fig. 6.4.2 : Applications of Cryptographic Hash Functions

→ **1. Data Authentication**

Used to establish proof of identities and ensure that the origin of an electronic message is correctly identified and to check if a message has been modified or not.

→ **2. Digital Signatures**

Encrypt message digest using private key and identify the proof of message.

→ **3. Password Storage**

Message digest of password is compared with that in the storage; hackers cannot get password from storage.

→ **4. Key Generation**

Key can be generated from digest of pass-phrase; can be made computationally expensive to prevent brute-force attacks.

→ **5. Intrusion Detection and virus Detection Technique**

Keep and check hash of files on system.

Scanned by CamScanner

Syllabus Topic : MAC

6.5 Message Authentication Codes

→ (MU - May 18)

Q. 6.5.1 What is the need for message authentication ? List various techniques used for message authentication. Explain any one. (Ref. sec. 6.5) **May 18, 10 Marks**

- As studied earlier encryption mechanism does not provide a good solution for message authentication because it is difficult for the receiver to identify the legitimate plain text. To tackle such type of problem, we can apply the method of error detection code into the message so that only legitimate plaintext can pass the error detection.
- Such type of error detection codes are used in the network communication for data integrity verification against the bit errors introduced during communication channel. The Message Authentication Code (MAC) is a small fixed-size block of data that is generated based on a message M of variable length using secret key K as follows. MAC is also called cryptographic checksum.

$$MAC = C(K,M)$$

- If sender A wants to send the secret message to receiver B via MAC, then the first condition is to share a secret key K.
  1. Sender A calculates the MAC by applying K to the message M.
  2. A sends the original message M and the MAC Hash1 to B
  3. When B receives the message, B also uses K to calculate its own MAC Hash2 over message M.
  4. B now compares Hash1 with Hash2. If matching is correct then B assures that Message M has not been changed or alters during transmission. In case if Hash1 and Hash2 doesn't match then B can reject the message, realizing that message was changed during transmit as shown in Fig. 6.5.1.

$$C_1 = E(k, P_1)$$

$$C_2 = E(k, [P_2 \oplus C_1]\}$$

$$C_3 = E(k, [P_3 \oplus C_2]\}$$

$$\vdots$$

$$C_n = E(k_1 [P_n \oplus C_{n-1}]\}$$

Scanned by CamScanner

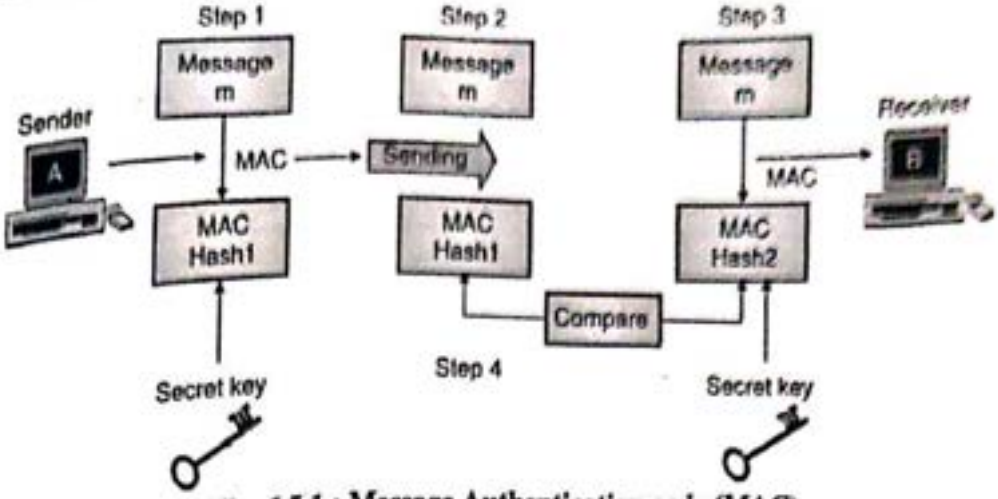


Fig. 6.5.1 : Message Authentication code (MAC)

6.5.1 Significance of MAC

1. As mentioned earlier MAC ensures that only receiver can identify the original message. Even if attacker modifies the original message M he cannot modify the MAC Hash1, in case of Hash 1 modification, receiver's calculations of MAC Hash2 will differ from it. Why attacker is not modifying the MAC? Because key used to calculate the MAC is known only to sender and receiver of the message. The attacker don't know the key used during transmission, the reason he attacker cannot modify the MAC.
2. In this case receiver B assures that message is coming from A not from any third person because the secret key K used at the time of encryption and decryption is known only to sender and receiver of the message.
3. MAC provides data confidentiality and authentication.

6.5.2 Message Authentication Code based on DES

- One of the most widely used MACs is referred to as the Data Authentication Algorithm. The algorithm is designed using the Cipher Block Chaining (CBC) mode of operation of DES, as shown in Fig. 6.5.2.
- In Cipher Block Chaining (CBC) mode the given plaintext message is divided into blocks of 64 bits each and each 64-bits blocks get encrypted independently. The plaintext block produces ciphertext of same size (64-bits each).
- The given plaintext is encrypted (DES encryption) using same key and transfers the encrypted data (ciphertext) to receiver.
- The algorithm can be defined as using the Cipher Block Chaining (CBC) mode of operation of DES with initialization vector (IV).

Scanned by CamScanner

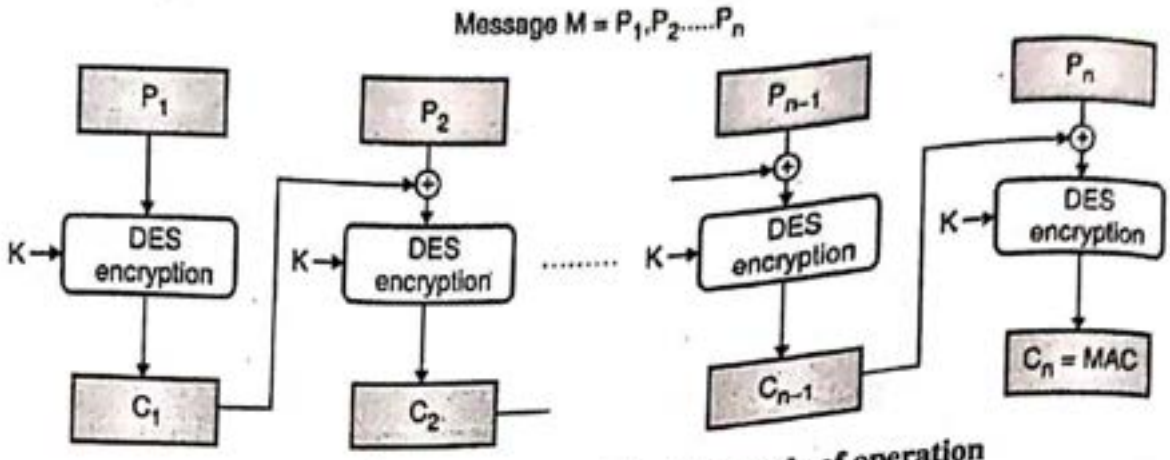


Fig. 6.5.2 : MAC design using DES in CBC mode of operation

- Using DES encryption algorithm, E and secret key K a Data Authentication Code (DAC) also called MAC is calculated as follows.

6.5.3 Mathematical Equation

- Message Authentication code (Data Authentication code generation)
- The MAC code consists of either the entire final block C_n or the leftmost m bits of the block with 16 <= m <= 64.

Syllabus Topic : HMAC

6.5.4 HMAC (Hash based Message Authentication Code)

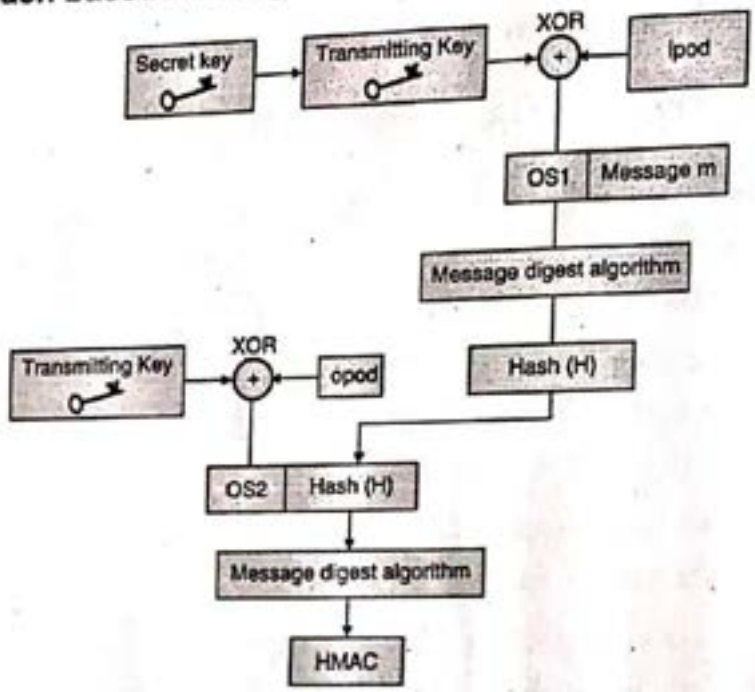


Fig. 6.5.3 : HMAC Operation

Scanned by CamScanner

6.5.4(A) Complete HMAC Operation

(i) Message Digest Algorithm : The message digest algorithm used (MDS, SHA-1 etc.)

(ii) M – The input message m whose MAC is to be calculated.

(iii) k – The shared secret key used in HMAC

ipad – The string OX36 byte repeated 64 times

opad – The string OX5C byte repeated 64 times

(Two fixed and different 64 byte strings ipad and opad)

OS1 = Output of step 1

OS2 = Output of step 2

Step 1 : The length of message m must be equal to length of key, i.e. length of key (K) must be equal to number of bits in the original message block. For example in original / initial length of key is 170 bits and message length (i) 512 bits then odd 342 bits into key length and make it equal size

Step 2 : Secret key now transmitting and XOR with ipad to produce OS1 (output of stage 2 (OS1 is a variable).

Step 3 : Append message m to output of step 2 i.e. original message m i) added with output of step2 which will produce message digest (OS1 + m)

Step 4 : The MDS and SHA -1 is applied on the output of step 3. This will produce output Hash (H).

Step 5 : XOR the secret key K with opad to produce output variable called OS2(output of step 1)

Step 6 : Add Hash H with OS2(Here the message digest calculated in step 4 is taken into consideration (Hash H) and appended with output of step 5.

Step 7 : Message digest algorithm is applied on output of step 6 (OS_2 + Hash H) to generate final output called as MAC.

We can also write above steps in mathematical form as,

$$HMAC = H(K \oplus opad, H(k \oplus ipad, message\ m))$$

Where,

HMAC = Final output

k = secret Kay

Scanned by CamScanner

⊕ = X OR operation

M = input message m

opad and ipad = fixed and different 64 byte strings repeated 64 times.

Step 1 : Append zeros to the end of key to make length 64 byte (i.e. key k and message must be equal)

Step 2 : XOR 64 – byte string computed in step 1 with ipad.

Step 3 : Append the message m with 64 byte string resulting from step 2.

Step 4 : Apply message digest algorithm on output of step 3.

Step 5 : XOR the 64 byte string computed in step with opad.

Step 6 : Append hash H output of step 4 with 64 byte string generated from step 5.

Step 7 : Apply message digest algorithm (MD5,SHA-I) on output of step (6) to generate final output i.e. HMAC.

6.5.5 Difference between Message Digest and Message Authentication Code

Sr. No.	Message Digest Algorithm	Message Authentication Code
1.	A message digest algorithm takes a single input like message and produces a "message digest" (called as hash) which helps us to verify and check the integrity of the message.	A Message Authentication Code algorithm takes two inputs one is a message and another is secret key which will produces a MAC which allow us to verify and check the integrity and the authentication of the message.
2.	Any change to input message produces different result i.e. different hash being generated.	Any change to in message or the secret key will result in a different MAC being generated.
3.	Once the hash is generated which will not give any clue to the attacker about original content of the message.	Without secret key it is not possible for attacker to identifies and validate the correct MAC.
4.	Most popular message digest algorithms are MD5 and SHA-I	Most popular MAC are MAC using DES in CBC mode and HMAC

Scanned by CamScanner

Syllabus Topic : CMAC

6.6 CMAC (Cipher Based Message Authentication Code)

- Cipher-based message authentication codes (or CMACs) are a tool for calculating message authentication codes with a block cipher coupled with a secret key. You can use a CMAC to prove both the integrity and authenticity of a message.
- CMAC (Cipher-based Message Authentication Code) is a block cipher-based message authentication code algorithm. It may be used to provide guarantee of the authenticity and, hence, the integrity of binary data. This mode of operation fixes security deficiencies of CBC-MAC (CBC-MAC is secure only for fixed-length messages).
- The core of the CMAC algorithm is a deviation of CBC-MAC that Black and Rogaway proposed and analyzed under the name XCBC and submitted to NIST.
- The XCBC algorithm efficiently addresses the safety deficiencies of CBC-MAC, but requires three keys.
- Iwata and Kurosawa planned an improvement of XCBC and named the resulting algorithm One-Key CBC-MAC (OMAC) in their papers. They later submitted OMAC1, a refinement of OMAC, and added security analysis.
- The OMAC algorithm reduces the amount of key material necessary for XCBC. CMAC is equivalent to OMAC1.

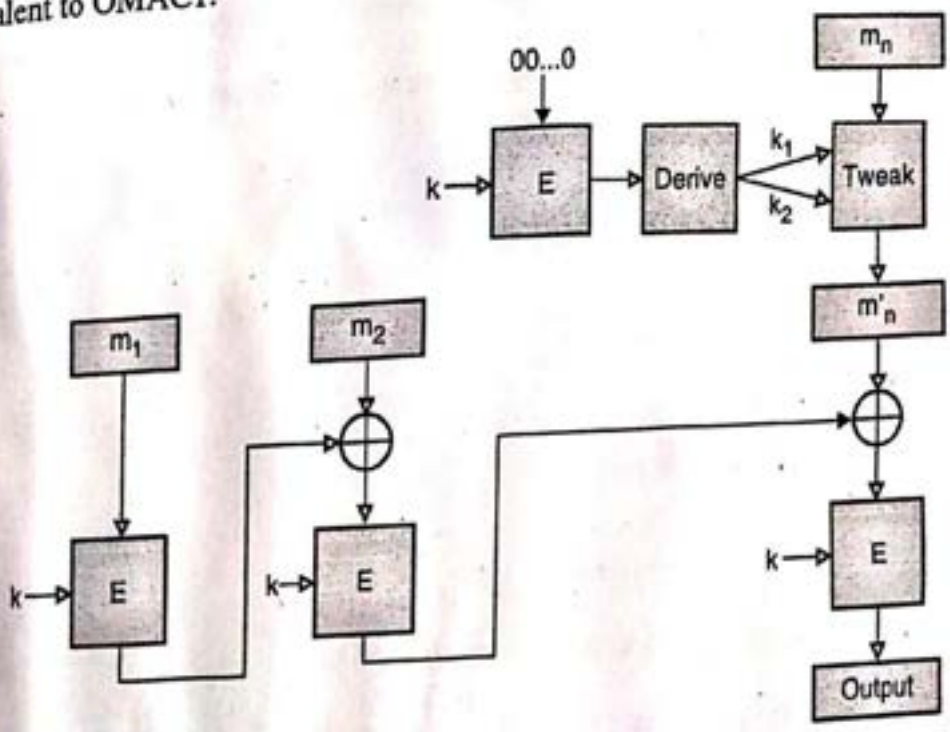


Fig. 6.6.1 : CMAC

Scanned by CamScanner

- To generate an t-bit CMAC tag (t) of a message (m) using a b-bit block cipher (E) and secret key (k), one first generates two b-bit sub-keys (k_1 and k_2) using the subkey algorithm (this is equivalent to multiplication by x and x^2 in a finite field $GF(2^b)$).
- Let $\ll$ denote the standard left-shift operator and $\oplus$ indicate bit-wise exclusive or:

1. Calculate a temporary value $k_0 = E_k(0)$.
2. If $\text{msb}(k_0) = 0$, then $k_1 = k_0 \ll 1$, else $k_1 = (k_0 \ll 1) \oplus C$; where C is a definite constant that depends only on b. (Specifically, C is the non-leading coefficients of the lexicographically first irreducible degree-b binary polynomial with the minimal number of ones: 0x1B for 64-bit, 0x87 for 128-bit, and 0x425 for 256-bit blocks.)
3. If $\text{msb}(k_1) = 0$, then $k_2 = k_1 \ll 1$, else $k_2 = (k_1 \ll 1) \oplus C$.
4. Return keys (k_1, k_2) for the MAC generation process.

As a small example, suppose $b = 4$, $C = 0011_2$, and $k_0 = E_k(0) = 0101_2$. Then $k_1 = 1010_2$ and $k_2 = 0100 \oplus 0011 = 0111_2$.

6.6.1 The CMAC Tag Generation Process

1. Divide message into b-bit blocks $m = m_1 \| \ldots \| m_{n-1} \| m_n$, where $m_1, \ldots, m_{n-1}$ are complete blocks. (The empty message is treated as one incomplete block.)
2. If m_n is a complete block then $m_n' = k_1 \oplus m_n$ else $m_n' = k_2 \oplus (m_n \| 10\ldots0_2)$.
3. Let $c_0 = 00\ldots0_2$.
4. For $i = 1, \ldots, n-1$, calculate $c_i = E_k(c_{i-1} \oplus m_i)$.
5. $c_n = E_k(c_{n-1} \oplus m_n')$
6. Output $t = \text{msb}_t(c_n)$.

6.6.2 The Verification Process

1. Use the above algorithm to generate the tag.
2. Check that the generated tag is equal to the received tag.

   2.1. Apply the MAC generation process to M to produce T.

Scanned by CamScanner

2.2. If $T = T'$, return VALID; else, return INVALID.

- CMAC is widely used in Govt and industry.
- But it has message size limitation.
- It can be overcome using 2 keys and padding.
- Thus forming the Cipher-based Message Authentication Code (CMAC).

Chapter Ends...

Scanned by CamScanner

Module 3

CHAPTER 7

Digital Certificate

Syllabus

Digital Certificate : X.509, PKI.

Syllabus Topic : Digital Certificate X.509

7.1 X.509 Authentication Service/ Digital Certificate

→ (MU - Dec. 15, Dec. 17)

Q. 7.1.1 Give the format of X.509 digital certificate and explain the use of a digital signature in it. (Ref. sec. 7.1) **Dec. 15, 5 Marks**

Q. 7.1.2 What is a digital certificate? How does it help to validate the authenticity of a user? Explain the X.509 certificate format. (Ref. sec. 7.1) **Dec. 17, 10 Marks**

- Digital certificate is an electronic file that is used to identify people and resources over a insecure channel or a networks called Internet. Digital certificate also enable secure, confidential communication between sender and receiver using encryption.
- For example when we travel to another country, our passport provides a way to establish our identity and gain entry. Digital certificate provide similar identification in the electronic world.
- The role of Certification Authority (CA) is to issue certificates with authorized digital signature. Much like the role of the passport office, the role of the CA is to validate the certificate owner's identity and to "sign" the certificate so that it cannot be tampered by unauthorized user.

Scanned by CamScanner

Once a CA has signed a certificate, the owner can present their certificate to people, web sites and network resources to prove their identity for confidential communications over insecure channel.

A standard called as **X.509** defines structure of digital certificate. The International Telecommunication Union (ITU) permitted this standard in 1998.

Fig. 7.1.1 shows structure of X.509 digital certificate.

Digital Certificate contents
Certificate version number
Certificate serial number
Algorithm for signature identifier
Certificate Issuer name
Validity Details
Name of the certificate owner Public key of certificate owner
Issuer unique identifier
Owner unique identifier
Extensions to certificate
Certification Authority (CA) Digital Signature

Fig. 7.1.1 : Structure of X.509 Digital certificate

A standard digital certificate typically includes a variety of information pertaining to its owner and to the Certification Authority (a trusted agency that can issue digital certificate) such as :

1. **Certificate version number**

   Identifies a particular version of the X.509. Current version is X.509 v3.

2. **Certificate serial number**

   Unique integer number generated by certification authority.

3. **Algorithm for signature identifier**

   Identifies algorithm used by the certification authority to sign the certificate.

Scanned by CamScanner

4. **Certificate Issuer name**

   The name of the Certification Authority that issued the certificate.

5. **Validity Details**

   The validity period (or lifetime) of the certificate (a start and an end date).

6. **Name of the certificate owner**

   The name of the owner and other identification information required for identifying the owner such as email id and contact details.

7. **Public key of certificate owner**

   Certificate owner's public key, which is used to encrypt confidential information of the certificate owner.

8. **Issuer unique identifier**

   Indentify the CA uniquely i.e. whether single CA signed it or is any CA using same details.

9. **Owner unique identifier**

   Indentify the owner uniquely if two or more owner has used the same name over a time.

10. **Extensions to certificate**

    This is an optional field which allows a CA to add additional private information to certificate. These additional fields are called as extensions of version 2 or 3, respectively.

11. **Certification Authority (CA) Digital Signature**

    In creating the certificate, this information is digitally signed by the issuing CA. The CA signature on the certificate is like a tamper-detection seal on packaging any tampering with the contents is easily detected.

7.1.1 Importance of Digital Certificate

- Digital certificates are based on public-key cryptography, which uses a pair of keys for encryption and decryption. A digital certificate can securely attach your identity, verified by a trusted third party, with your public key.
- As defined earlier digital certificate is a mechanism for users to obtain assurance about the identity and authenticity of a web site.

Scanned by CamScanner

- By inspecting the digital certificate on a web site, users can help prevent identity theft and fraud.
- For example; a phishing site set up by attacker which shows itself as a legitimate web site (such as an online banking web site) can be identified by using invalid or absent of digital certificate.
- Digital certificates are implemented as part of a set of security mechanisms provided by Secure Socket Layer (SSL).
- SSL encrypts all data sent between a sender computer and a remote computer (server) to prevent data from reading during transmission process from a sender computer to the server computer. We will study SSL in unit IV.
- A general user can create their online digital certificate; lot of third party agencies companies providing facility of creating users own digital certificates.

Syllabus Topic : PKI

7.2 Public Key Infrastructure (PKI)

Q.7.2.1	What is PKI ? Explain the different PKI Architectures. (Ref. sec. 7.2)
Q.7.2.2	Explain Public key infrastructure X.509 with the help of architectural block diagram. (Ref. sec. 7.2)
Q.7.2.3	Does a public key infrastructure use symmetric or asymmetric encryption? Explain your answer. (Ref. sec. 7.2)

- Public Key Infrastructure (PKI) is cryptographic technique used to secure electronic information with the help of certain techniques such as digital certificates and digital signature and transmission of this information securely over internet.
- PKI consists of certain security policies, software's and techniques that are required for key generation, key management, secure storage of generated keys and distribution generated keys.
- A public key infrastructure is created by combining a number of services and technologies. To complete this technology, there are various components of PKI are required.

Scanned by CamScanner

7.2.1 Components of PKI

For this framework to be functional, we need various components of PKI,

Following are the different components of PKI :

1. Certification Authority (CA)
2. Registration Authority (RA)
3. PKI clients
4. Public key certificate/ digital certificate
5. Certificate Distribution System (CDS) Repository

7.2.1(A) Certification Authority (CA)

Q. 7.2.1 List the certifying authorities in India and worldwide. Also list the steps to acquire the digital certificate. (Ref. sec. 7.2.1(A))

- As mentioned in previous section Certification Authority (CA) is a trusted unit that helps to issue certificates.
- A CA takes the certificate request from owner, verifies the requested information according to the terms and conditions of the CA, and uses its private key to apply digital signature to the certificate.
- Responsibility of the CA is to identify the correct identity of the person who asks for a certificate to be issued, and make sure that the information contained within the certificate is legal and later digitally sign on certificate.
- The CA may generate a public key and a private key (a key pair) or the person applying for a certificate may have to generate their own key pair and send a signed request containing their public key to the CA for validation.
- After the verification from CA it sends certificate for final verification to registration authority (RA).

7.2.1(B) Registration Authority (RA)

- Registration Authority (RA) acts as an intermediate entity between the Certificate Authority (CA) and the certificate owner responsible for to user registration and accepting requests for certificates.

Scanned by CamScanner

- User registration is the process of collecting user information and verifies user identity, which is then used to register a user according to policies of CA and RA.
- Both CA and RA mutually authenticate user's identity and then issue certificates upon completion of registration process.

7.2.1(C) PKI Clients

- The users which request CAs or RAs to issue certificates are called to as PKI Clients. To obtain a digital certificate from a CA, a PKI client needs to perform the below three steps :

1. Send a request to generate a public and private key pair. The key pair contains the details of the client.
2. Then sends request to the CA for the CA certificate through RA.
3. After verification from CA and RA client can use certificate to prove itself authorized user and certificate owner.

- Every communication between a client and the CA is secure because client is responsible for ensuring the security of its private key. If the private key is lost or stolen, then the encrypted message cannot be decrypted or any unauthorized person can uses this private key to decrypt the messages.

7.2.1(D) Public Key Certificate/ Digital Certificate

Refer section 7.1 for digital certificate.

7.2.1(E) Certificate Distribution System (CDS) Repository

- The Certificate Distribution System (CDS) distributes certificates to users and organizations. Certificates have a specified life time, but CA scan reduce this life time by the process known as certificate revocation.
- The CA publishes Certificate Revocation List (CRL) which mentions serial number so certificates which are no longer usable, reasons for certificate cancelation, and date when to apply for new certificate.
- Certificate can distributed to the users directly or distributed with the help of a directory service server. CDS distributes certificates in support with the directory service server.

Scanned by CamScanner

- The role certificate distribution system is to perform following tasks :
  - Generate public and private key pairs. Certify the validity by signing with public key.
  - Revocation of expired or lost keys.
  - Distribute or publish the certificate along with the public keys in the directory service server.

7.2.2 PKI Applications / Services

Different services increase the importance of PKI like : e-mail, secure file transfer, document management services, remote access, e-commerce and Web-based transaction services etc.

E-Mail and Messaging

Secure e-mail and messaging use key pairs for encryption of messages and files, and for digital signatures. The most common secure e-mail / messaging protocol is Secure Multipurpose Internet Mail Extensions (S/MIME), which extends the Multipurpose Internet Mail Extensions (MIME) standard.

Web Access

Browsers and We observers use encryption for authentication and confidentiality and for applications like online bank in and online shopping. Typically, using Secure Sockets Layer (SSL), servers authenticate themselves to clients. SSL also encrypts traffic.

Chapter Ends...

□□□

Scanned by CamScanner

CHAPTER 8

Module 3

Authentication Protocols

Syllabus

User Authentication and Entity Authentication, One-way and mutual authentication schemes, Needham Schroeder Authentication protocol, Kerberos Authentication protocol.

Syllabus Topic : User Authentication

8.1 User Authentication

- User authentication is a process that allows a device to verify the identity of someone who connects to a network resource.
- There are many technologies currently available to a network administrator to authenticate users.

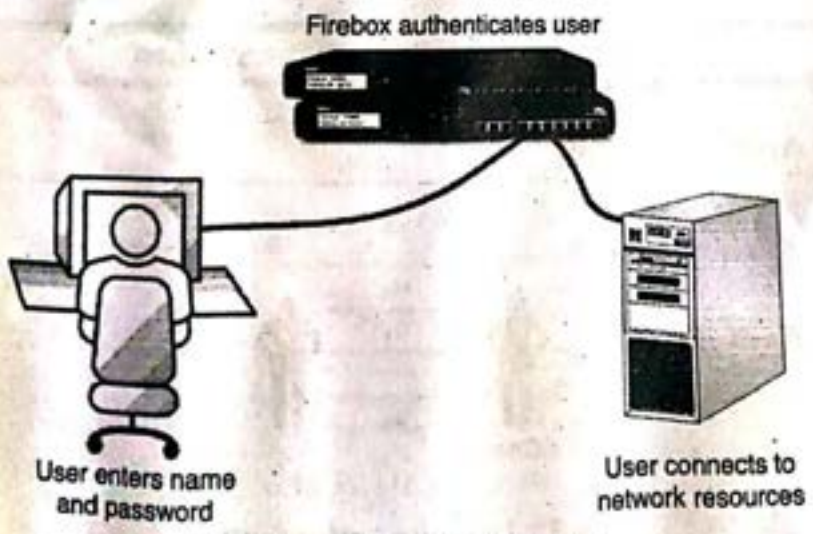


Fig. 8.1.1 : User Authentication

Scanned by CamScanner

- Authentication is very important when you use dynamic IP addressing (DHCP) for computers on the trusted or optional network.
- It is also important if you must identify your users before you let them connect to resources on the external network.
- To get access to services such as HTTP or FTP the user types a domain along with their login name and password.
- For the duration of authentication, the user name is associated with connections coming from the IP address from which the user authenticated.
- This makes it possible to monitor not only the computers from which connections originate, but also the users who start the connection.
- While the user is authenticated, all the connections that the user starts from the IP address include the session name.

8.1.1 Means of User Authentication

- Four means of authenticating user's identity based on something of the individual
  1. knows - e.g. password, PIN
  2. possesses - e.g. key, token, smartcard
  3. is (static biometrics) - e.g. fingerprint, retina
  4. does (dynamic biometrics) - e.g. voice, sign
- Can use alone or combined. All can provide user authentication and all have issues.

Syllabus Topic : Entity Authentication

8.2 Entity Authentication

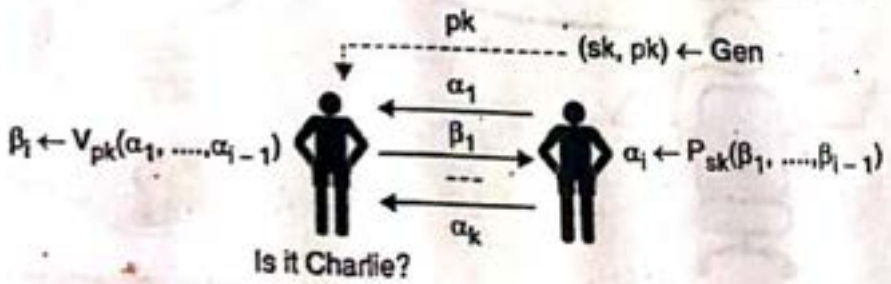


Fig. 8.2.1 : Entity Authentication

Scanned by CamScanner

- The communication between the prover and verifier must be authentic.
- To establish electronic identity, Charlie must generate (p_k, s_k) Gen and convinces others that the public information p_k represents him as show in Fig. 8.2.1.
- The entity authentication protocol must convince the verifier that his or her opponent possesses the secret s_k.
- An entity authentication protocol is functional if an honest verifier V_{pk}
- Always accepts an honest provider P_{sk}.

8.2.1 Physical and Legal Identities

- Entity authentication is possible only if all participants have set up a network with authenticated communication links.

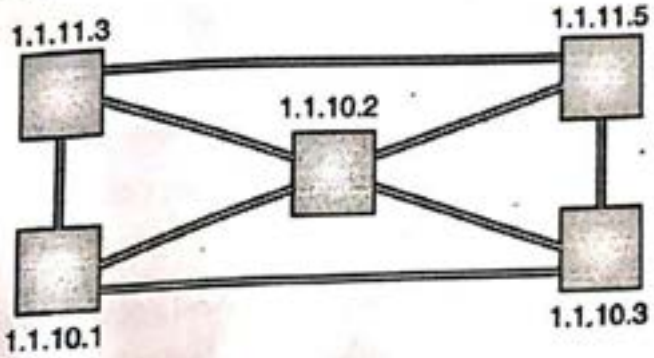


Fig. 8.2.2 : Physical and legal Identities

- A role of a entity authentication protocol is to establish a convincing bound between physical network address and legal identities.
- A same legal identity can be in many physical locations and move from one physical node to another node.

8.3 Authentication Protocol

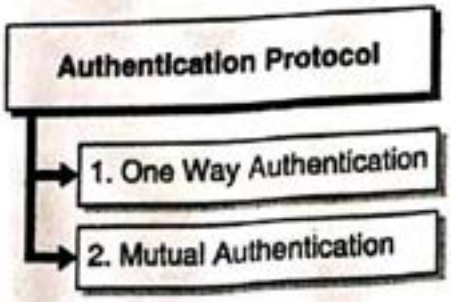


Fig. 8.3.1 : Authentication Protocol

Scanned by CamScanner

➜ 1. One Way Authentication

- As defined earlier authentication mechanisms help to prove the identity of the sender of the message.
- Authentication mechanisms ensure that who sends the message i.e. origin of an electronic message is correctly identified.
- One-way authentication refers to the authentication of only one end of communication users. For example, One-way authentication follows the flow : If there are two users, user A and B wants to communicate with each other user B authenticates user A, but user A cannot authenticate user B. This process called one-way authentication. Finally the integrity and originality of message is confirmed.
- There are different factors of authentication mechanisms used to give strength for authentication. First method is known as **one-factor authentication**. Password is the example of one-factor authentication because it is something that we know.
- Any operating system first ask for a user name and then for a password. It then looks up the name in a password table and sees if the passwords match. This is known as a **reusable password** since the same password is used for each login.
- Second method of authentication is known as **two-factor authentication.** Withdrawing cash from an ATM machine is an example of two-factor authentication. For authentication present the ATM card (something we have) and enter PIN (something we know) or use of **one-time passwords** - a new password must be used for each login.

➜ 2. Mutual Authentication

- Mutual authentication also called as two-factor authentication. Mutual Authentication is a security mechanism used to authenticate sender with the receiver. Sender must prove its identity to a receiver, and the receiver must prove its identity to the sender, before any unwanted threat sent between the sender and receiver.
- For example : If sender wants to communicate with the receiver over networks they must first mutually authenticate each other.
- Meaning is that when sender A sends confidential message which is intended to receiver B. If B can decrypt the message using A's public key, then B has verified that the message originated from A.
- Both communicating users (sender and receiver) are verifying each other i.e. mutual authentication mechanisms helps to verify identity of the sender.

Scanned by CamScanner

- The most important application of mutual authentication is that communication between client machine and server machine over a network must be secure before performing any data sending and receiving process.

8.3.1 Why there Is a Need of Mutual Authentication Protocol ?

- In order to achieve mutual authentication there must be certain provision of some protocols which suppose to verify identity of the sender over an insecure communication channel.
- To achieve this goal most of the protocols depends on an authentication server also called Key Distribution Center (KDC).
- If sender A wants to establish a secure communication with receiver B, then A can request for session key from Key Distribution Center for communicating with B. If group of people wants to securely communicate with Key Distribution Center then providing every group member a single key called a master key or secret key. Authentication servers are capable to delivers good quality session keys and distribute securely to client who requested it.
- Authentication server also maintains a table containing a name and a master key or secret key of each client.
- The secret key is used to authenticate client to the authentication server and then for securely transmission of data between client and the authentication server.
- There are different protocols are used to perform this task but among this the well known protocol called as Needham-Schroeder Protocol.

Syllabus Topic : Needham Schroeder Authentication Protocol

8.3.2 Needham - Schroeder Protocol

- The first mutual protocol was published in 1978 by Needham and Schroeder. This approach was proposed for various purposes that includes secret-key and public key generation and distribution of those keys between sender and receiver.
- Needham and Schroeder protocol uses a secret key known to the sender and also to an authentication server. Sender and receiver share a secret key and use it for secure communication with authentication server.

Scanned by CamScanner

Detail steps of Needham-Schroeder Secret-key Protocol

Step 1 : Sender A requests for a session key to authentication server for communication with receiver B as shown in Fig. 8.3.2. The message sent by A to authentication server includes A's secret key Ka, A's network address Na, B's network address Nb and a nonce. A nonce is basically a random number used to demonstrate the freshness of a request denoted by N. The request sent by A to authentication server which is in encrypted format E denoted by,

$$E (Ka, [Na, Nb, N])$$

Step 2 : Authentication server returns a message, containing a newly generated key Kab (used to encrypt communication between sender and receiver), nonce N (to match the response received from authentication server with the request sent), ticket (contains the same shared secret key Kab, as well as the name of the sender A) encrypted with B's secret key Kb and whole these message encrypted with sender private key or secret key Ka to ensure that no one else can read it. The message that authentication server sends back to A can be expressed as :

$$E (Kab, N, \{A, Kab\} Kb, B_{|}Ka$$

Step 3 : After receiving replay from authentication server, sender decrypt the ticket and sends the ticket {A, Kab} to the receiver B. A sends the ticket to B which is not in encrypted format because it was previously encrypted by authentication server using B's secret key Kb.

$$(A, Kab) Kb$$

Step 4 : B decrypts the ticket received from A using the secret key Kb and compares sender identity. B is again encrypting the ticket using shared secret key Kab and generates nonce N1 and sends it back to receiver. This can be represented as

$$E (N1) Kab$$

In this step B got the **session key** (Kab) for communicating securely with A.

Step 5 : Sender is decrypting the nonce N1; using the shared secret key Kab this proved the senders identity. The sender sends response N1+1 encrypted using the shared secret key Kab.

$$E(N1 + 1) Kab.$$

Scanned by CamScanner

Step 6 : Now sender A and receiver B can securely communicate with each other using session key generated.

- The main weakness of this protocol is that for large networks it is not possible for single authentication server to generate and distribute number of session key which is practically not possible.
- Another weakness is that if session key between sender A and receiver B is stolen, and the ticket to B is recorded, attacker can easily copy the contents of a sender A by performing last 3 steps.

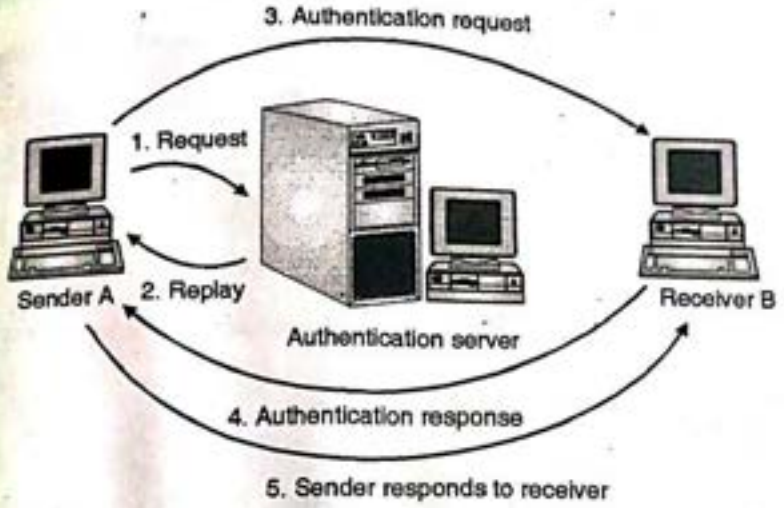


Fig. 8.3.2 : Needham - Schroeder Secret-key Protocol

Syllabus Topic : Kerberos Authentication Protocol

8.4 Kerberos Authentication Protocol

➔ **(MU - May 16, May 18)**

Q. 8.4.1	Explain working of Kerberos.(Ref. sec. 8.4)	**May 16, 10 Marks**
Q. 8.4.2	Explain Kerberos protocol that supports authentication in distributed system. (Ref. sec. 8.4)	**May 18, 10 Marks**

- Kerberos is also called as authentication protocol. Like when to start in journey we need a confirm ticket then only we can do our journey safely.

Scanned by CamScanner

- Kerberos uses the concept of the ticket as a token to prove the identity of the user. Microsoft introduced Kerberos in Windows 2000 server as a default authentication protocol.
- Kerberos uses the concept of a ticket as a token that proves the identity of a user.
- Tickets are digital documents that store session keys. Instead of password, tickets are issued during login session and then can be used in any Kerberos services.
- For client authentication phase requires two tickets :
  - Ticket Granting Ticket (TGT), which act a identifier for user and session key
  - A service ticket to authenticate user to gain access to user for particular service.
- The same concept of ticket is used likewise we use railway tickets it has time duration, expiration dates after that ticket become invalid.
- In Kerberos these ticket includes different contents like time stamps to indicate an, start and expiration time, after time expiration the ticket become invalid.
- The timestamp is the time set by Kerberos administrator depending upon how much time service is required to the client.

1. Kerberos Servers

- To accomplish the task of secure authentication, Kerberos uses a trusted third party is called a Key Distribution Center (KDC).
- The Key Distribution Center uses two techniques for authentication :
  - Authentication Server (AS), which performs user authentication.
  - Ticket- Granting Server (TGS), which permits/ grants tickets to users.
- The role of an Authentication server is to store a database like secret key of the users and its services.
- The secret key of a user is generated using one-way hash of user provide password.
- The main aim of the Kerberos is provide centralize authentication of entire network rather than storing the sensitive information at each user machine, the sensitive information will be maintained at particular secure location only.

Scanned by CamScanner

2. Kerberos Authentication

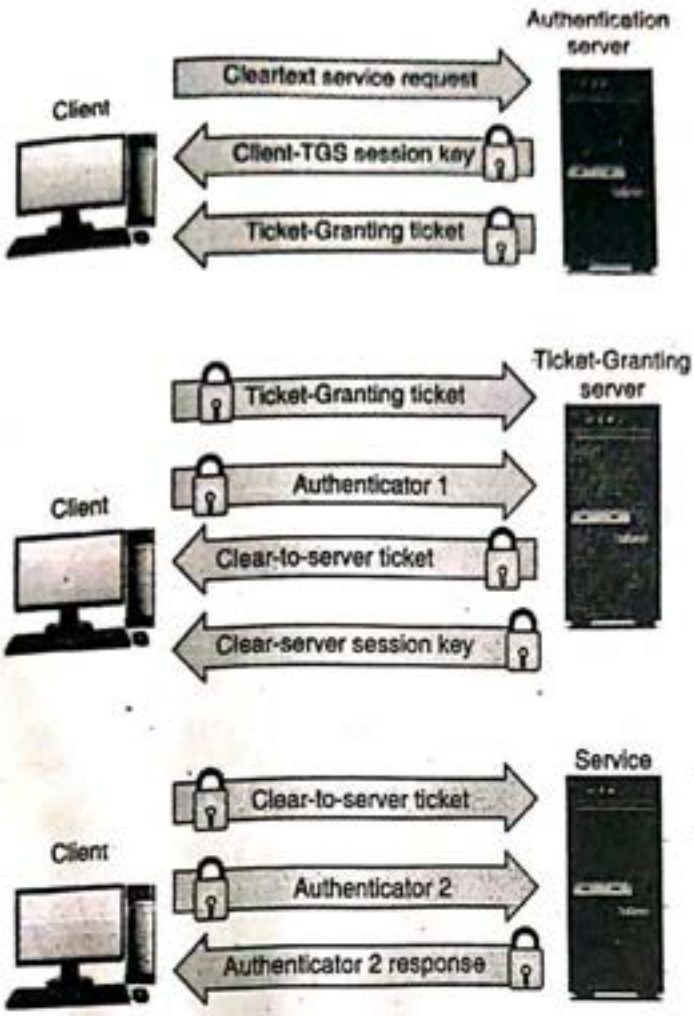


Fig. 8.4.1 : Kerberos athentication process

- This phase is called as Authentication phase because during this phase only authentication can be done between authentication server, ticket-granting server and service provider.
- As shown in Fig. 8.4.1, first client and authentication server authenticate themselves to each other.
- Client and Ticket granting server authenticate themselves. Finally client and requested service provider authenticate themselves to each other regarding which information/ service client wants.

3. Authentication Details

- During authentication phase user has to provide username and password on the client machine which cryptographically hashed to create a secret key for the client.

Scanned by CamScanner

- After client verification done with authentication server, AS will replies the following details to client as shown in 8.4.1.
  - The client Ticket Granting Sever (TGS) session key Kt, encrypted using clients secrets key Kc (which now stored in authentication server).
  - The Ticket Granting Ticket (TGT) encrypted using the secret key of the Ticket granting server. The ticket granting ticket includes the client ticket granting sever session key Kt and its validity period.
- The client now decrypt the Ticket Granting Server session key Kt using his secret key Kc. To request as service client sends following two message to Ticket Granting Server (TGS).
  - The Ticket Granting Ticket and the name of the service Sr that client wants to request.
  - Authentication token which includes client ID and time stamp, encrypted using client ticket granting server session key Kt.
- Upon receiving all the details from client Ticket Granting Server decrypts the Ticket Granting Ticket using Kt, thus retrieving the client Ticket granting server session key stand the validity of the ticket granting ticket. If it is valid then Ticket granting Server sends following messages to the client.
- New client server session key Ksc, encrypted using TGS session key Kt.
  - Client to server ticket, encrypted using specific services key Ks, known to Ticket
  - Granting Server only. (Client to server ticket contains the client ID, network address, validity period and the client server session key Ksc).
- Upon receiving all the details from Ticket Granting server client decrypt the client server session key Ksc, and authenticate him to service Sr by sending following messages.
  - The client server ticket sent by the ticket granting server in previous step.
  - The client ID and the time stamp encrypted suing client server session key Ksc.
- The service provider decrypts the client to server ticket using secret key Ks and obtains the client server session key Ksc. With the help of client server session key Ksc, service provider decrypt the client ID and time stamp information. To prove the final identity service providers increment the time stamp by 1 and send it back to the client.

Scanned by CamScanner

- The client decrypts and verifies this response using client to server session key Ksc. Once this verification get succeed, now client - server can start.
- Kerberos protocol was specially design to check the authentication of the client over insecure network.
- Two types Kerberos versions are exits i.e. Kerberos 4 and 5.

8.4.1 Difference between Kerberos Version 4 and Version 5

Sr. No.	Kerberos version 4	Kerberos version 5
1.	Kerberos v4 was released prior to the version 5 in the late 1980's.	The version 5 was published in 1993, years after the appearance of version 5.
2.	Ticket support is Satisfactory in this version	Ticket support is well extended. Facilitates forwarding, renewing and postdating tickets.
3.	It uses the "receiver-makes-right" encoding system.	It uses the ASN.1 coding system.
4.	Since the same key is used repeatedly to gain a service from particular server, there is a risk that an attacker can replay messages from an old session to the client or server.	In V5 this is avoided by requiring a sub session key which is used only for one connection.
5.	Kerberos V4 uses DES encryption techniques	In Kerberos V5 the cipher text is tagged with an encryption type identifier hence any type of encryption can be used.
6.	Kerberos uses IP addressing	Kerberos V5 can use any address since the address is now tagged with type and length.
7.	In V4 the ticket lifetime has to be specified in units of 5 minutes.	In V5 ticket lifetime one can specify an explicit start and finish times allowing arbitrary lifetimes.
8.	It contains only a few IP addresses and other addresses for types of network protocols.	Kerberos V5 tickets can now contain multiple IP addresses and addresses for different types of networking protocols.

Chapter Ends...

□□□

Scanned by CamScanner

Module 4

CHAPTER 9

Digital Signature Schemes

Syllabus

Digital Signature Schemes - RSA, ElGamal and Schnorr signature schemes.

9.1 Digital Signature

→ (MU - May 16, Dec. 16)

Q. 9.1.1 What is a digital signature? (Ref. sec. 9.1) **May 16, Dec. 16, 4/2 Marks**

- Digital signatures are essential in today's modern world to verify the sender of a document's and his identity.
- A digital signature is represented in a computer as a string of binary digits and computer is using a set of rules and regulations (algorithm) to identify the person signing the document as well as the originality of the data can be verified.
- A digital signature is defined the signature generated electronically from the digital computer to ensure the identity of the sender and contents of the message cannot be modified during transmission process.
- Digital signature techniques achieve the authenticity, integrity and non-repudiation of the data over Internet.
- Concept of digital signature is that sender of a message uses a signing key (Private Key) to sign the message and send that message and its digital signature to a receiver over insecure communication channel.
- The receiver uses a verification key (Public Key) of the sender only to verify the origin of the message and make sure that it has not been tampered with while in transit as shown in Fig. 9.1.1.

Scanned by CamScanner

- Hash value of a message when encrypted with the private key of a person is, his digital signature on that e-Document.
- Digital signature is an example of asymmetric key cryptography which uses three different algorithms to complete the process.

1. First step is key generation algorithm which generates private key and a corresponding public key.
2. Next step signing algorithm which selects sending message and a private key generated in step 1, to produce a signature.
3. Third step is signature verifying algorithm which verifies the authenticity of sending message and public key.

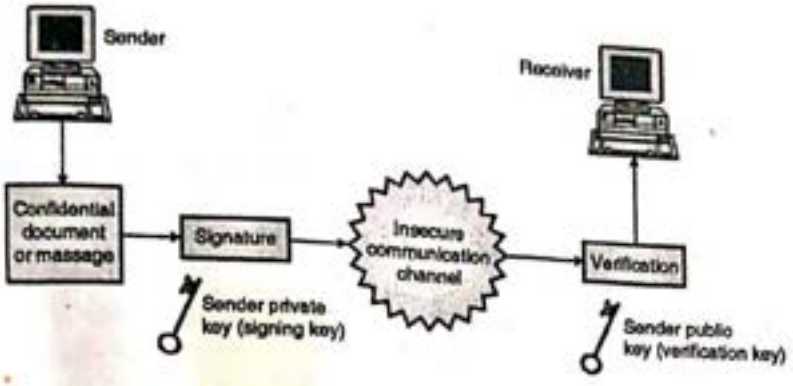


Fig. 9.1.1 : Digital Signature block diagram

- As mentioned above the signature is generated with the help of private key. The private key, which is never shared, is used in signature generation, known to sender only.
- Public keys, which are known by everyone, can be used to verify the signature of a sender. Every sender and receiver having a private and public key pair, the reason digital signature called public-key cryptography.

9.2 Digital Signature Goals

Fig. 9.2.1 : Digital Signature Goals

Scanned by CamScanner

→ 1. Message authentication

A digital signature technique can provide message authentication. Digital signature is used to establish proof of identities and ensure that the origin of an electronic message is correctly identified.

→ 2. Message integrity

Digital signatures are used to detect unauthorized modifications to data which assures that the contents of massage are not changed after sender sends but before it reaches to intended receiver.

→ 3. Non-repudiation

There are situation where a user sends a message and later on refuses that he had sent that message. This is known as non-repudiation because the person who signed the document cannot repudiate the signature at a later time as shown in Table 9.2.1.

Table 9.2.1 : Differences between Paper Signatures and Digital Signatures

Sr. No.	Security goals	Paper signature	Digital Signature
1.	Message Authentication	Paper signature may be forged.	Digital signature cannot be copied.
2.	Message Integrity	Independent to the contents of the document.	Depends on the contents of the document.
3.	Non-repudiation	Paper signature required handwriting expert to achieve non-repudiation.	Any computer user can achieve non-repudiation in Digital Signature.

Now-a-day's digital signature techniques is used in many application areas like sending confidential e-mails, during secure payment transfer and possibly all software companies, universities, educational institutions those want to achieve authentication and integrity of their confidential information.

Scanned by CamScanner

Syllabus Topic : Digital Signature Schemes

9.3 Digital Signature Algorithms/ Schemes

→ **(MU - May 16, Dec. 16)**

Q. 9.3.1 Explain any digital signature algorithm in detail. (Ref. sec. 9.3)

May 16, Dec. 16, 6/3 Marks

Following are the widely used digital signature schemes to generate the digital signatures.

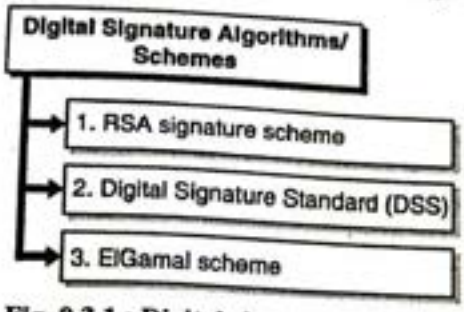


Fig. 9.3.1 : Digital signature schemes

Syllabus Topic : RSA Signature Scheme

9.3.1 RSA Signature Scheme

Ron Rivest, Adi Shamir and Len Aldeman have developed this algorithm (Rivest-Shamir-Aldeman) in 1978. It is a public-key encryption algorithm. It is a block-cipher which converts plain text into cipher text at sender side and vice versa at receiver side.

The algorithm works as follows :

1. Select two prime numbers a and b where $a \neq b$.
2. Calculate $n = a * b$
3. Calculate $\phi(n) = (a - 1) * (b - 1)$.
4. Select e such that, e is relatively prime to $\phi(n)$ i.e. $\gcd(e, \phi(n)) = 1$ and $1 < e < \phi(n)$.
5. Calculate of such that $d = e^{-1} \bmod \phi(n)$
   or $ed \bmod \phi(n) = 1$.
6. Public key = {e,n}, private key = {d,n}.
7. **Signature Generation**

   Use private key = {d,n}

Scanned by CamScanner

Compute signature $S = P^d \bmod n$ where, $P < n$ and

Where S = Generated Signature, P = Plaintext, e = Encryption key and n = Block size.

8. **Signature Verification**

Use signer's Public Key = {e,n} and Compute V_m

$$V_m = S^e \bmod n = m^{ed} \bmod n$$

Verify if V_m (Verified Message) = m (Original Message)

Both sender and receiver know the value of n. In addition, the sender must know encryption key 'e' and receiver must know decryption key 'd'.

Syllabus Topic : ElGamal Signature Scheme

9.3.2 ElGamal Scheme

The word cryptography is incomplete without mathematics. Cryptography is based on specific areas of mathematics like finding greatest common divisor using Euclidean algorithms. Finally most important concept called prime numbers used in different cryptography algorithms.

1. This scheme is variant of digital signature algorithm.
2. This scheme is based on computing assumption of discrete logarithms over finite field with a large prime factor.
3. It is computationally infeasible to compute q_r and s.

ElGamal scheme is based on difficulty of computing discrete logarithms. This scheme assures that the authenticity of message m sent by sender/ signer to verifier.

System parameters

- Let H be the hash function.
- Let a be the prime number.
- Let g < a be a randomly chosen generator.

Key generation

- Chose randomly a secret key x with $1 < x < p - 1$.
- Computey = $g^x \bmod a$.
- The public key (a, g, y)

Scanned by CamScanner

- These steps are performed by signer of documents.

- **Signature generation**

To sign a message m, signer needs to perform following steps.

- Chose random integer i such that

$(i : a-1) = 1 \; ; \; 1 < i < a-1$

$$gr = g^{i} \bmod (a)$$

Compute

$$s - (H(m) - xgr)\, k^{-1} \bmod (a-1)$$

If $s = 0$ then repeat above step.

- The pair (qr, s) is called as Digital signature of message m. The signer has to repeat all these steps during every signature generation.

- **Verification**

- Digital signature (gr, s) of message m was verified as follows,

$0 < q_r < a$ and $0 < s < a-1$

$$g^{H(m)} = y^{qr} q_r^{s} \bmod a.$$

- If $g^{H(m)} = y^{qr} q_r^{s} \bmod a$ is equal then verifier accept the signature otherwise he reject it.

Where,

$$H(m) = x\, q_r + Si \bmod a-1$$

- According to format s little theorem

$$g^{H(m)} = g^{xqr}\, g^{is}$$
$$= (g^{x})^{qr} \times (g^{i})^{S}$$

- We know that $y = q^{x} \bmod a$

- Above equation can be rewrite as follows,

$$g^{H(m)} = (y)^{qr} \times (gr)^{S} \bmod a$$

Hence proved.

- Here, H(m) = SHA – 1 (m) is 160- bit string output produced by secure Hash Algorithm.

Scanned by CamScanner

Syllabus Topic : Schnorr Signature Schemes

9.3.3 Schnorr Digital Significance Scheme

- Schnorr digital signature scheme is mainly based on discrete logarithm. Using this scheme we generate a digital signature.
- It minimizes the message-dependent amount of calculation required for generating the digital signature.
- The main aim of signature generation is it does not depend on the actual message.
- This is done, when the processor is in the idle mode.
- At the time of signature generation message dependent part requires multiplying a 2n-bit integer with n-bit integer.
- This method is mainly based on prime modulus m with m – 1 having a prime factor n of appropriate size so $m - 1 \equiv \pmod{n}$
- We use $m \approx 2^{1024}$ and $n = 2^{160}$
- P is a 1024 bit number and n is 160 bit number. In the first part this approach is the generation of public/private key.
- For this purpose we use following steps :
  1. Choose two prime m and n such a way that n is a prime factor of m – 1.
  2. choose one integer called as a such that $\propto^n = 1 \bmod m$. The value a, m and n comprise a global public key. That key is common to group of users.
  3. After the 2nd step choose any random integer called as 'P' that may consist of $0 < P < n$. this is we called users private key.
  4. After all of above we generate public key by calculating $U = a^{-p} \bmod m$ this value is nothing but user's public key.
- To generate a digital signature we use a users private key 'p' and the public key 'U'.
- For generating digital signature following steps are uses that are as following :
  1. Select any random integer A with $0 < A < n$ and calculate $x = a^A \bmod m$. This calculation is a preprocessing step independent of the message M to be signed.
  2. The Add/concatenated that message M with 'x' and hash value then that result to calculate the value e :

$$e = H(M \| x)$$

Scanned by CamScanner

3. Calculate $y = (A + Pe) \bmod n$. The signature contains the pair (e, y)

Any other user can find the signature using following way :

1. Calculate $x' = a^y u^e \bmod m$
2. Check that $e = H(M \| x')$

Then see that the verification work is same as like :

$$x' \equiv a^y u^e \equiv a^y a^{-pe} \equiv a^{y-pe} \equiv a^{\wedge} \equiv x \pmod m$$

hence,

$$H(M \| x') = H(M \| x)$$

Chapter Ends...

□□□

Scanned by CamScanner

CHAPTER 10

Network Security Basics

Syllabus

Network security basics : TCP/IP vulnerabilities (Layer wise), Packet Sniffing, ARP spoofing, port scanning, IP spoofing, TCP syn flood, DNS Spoofing.

Syllabus Topic : TCP/IP Vulnerabilities (Layer wise)

10.1 TCP/IP Vulnerabilities (Layer Wise)

10.1.1 Application Layer

- **Description** : The application layer is used largely by programs for network communication. Data is passed from the program in an application-specific format, and later encapsulated into a transport layer protocol.
- Since the IP stack has no layers among the application and transport layers, the application layer must contain any protocols that act like the OSI's presentation and session layer protocols. This is typically done through libraries.
- Data sent over the network is send into the application layer where it is encapsulated into the application layer protocol. From there, the data is send down into the lower layer protocol of the transport layer.
- The two most common end-to-end protocols are TCP and UDP. Common servers have particular ports assigned to them (HTTP has port 80; Telnet has port 23; etc.) while clients use ephemeral ports. Some protocols, such as File Transfer Protocol and Telnet may set up a session via a well-known port, but then forward the actual user session to ephemeral ports.

Scanned by CamScanner

- Routers and switches do not make use of this layer but bandwidth throttling applications do, as with the Resource Reservation Protocol (RSVP).

An example of an attack

SQL injection

- SQL injection is a technique that exploits a security weakness happening in the database layer of an application. The weakness is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed.
- It is in fact an occurrence of a more general class of vulnerabilities that can occur at any time one programming or scripting language is embedded inside another.
- Take a simple login page where a legitimate user would enter his username and password blend to enter a secure area to view his personal particulars or upload his comments in a forum.
- When the legitimate user submits his particulars, an SQL query is generated from these details and submitted to the database for verification.
- If valid, the user is given access. In other words, the web application that controls the login page will communicate with the database through a series of planned commands so as to verify the username and password combination. On verification, the legitimate user is granted suitable access.
- Through SQL Injection, the hacker may input exclusively crafted SQL commands with the target of bypassing the login form barrier and seeing what lies behind it.
- This is only doable if the inputs are not properly sanitised (i.e., made invulnerable) and sent directly with the SQL query to the database. SQL Injection vulnerabilities provide the means for a hacker to communicate directly to the database.
- The technologies vulnerable to this attack are dynamic script languages including ASP, ASP.NET, PHP, JSP, and CGI. All an attacker needs to perform an SQL Injection hacking attack is a web browser, knowledge of SQL queries and creative guess work to important table and field names. The utter simplicity of SQL Injection has fuelled its popularity.

A way to defense

- A network-based intrusion detection (IDS) tool such as Snort can be set up to detect certain types of SQL injection and XSS attacks as they take place. Snort actually has a default rule set that contains signatures for detecting these intrusions.

Scanned by CamScanner

- However, they can be easily bypassed by an attacker, mainly by converting the malicious input string into its hex-encoded value.

10.1.2 Transport Layer

- **Description :** The transport layer's responsibilities contain end-to-end message transfer capabilities independent of the core network, along with error control, fragmentation and flow control. End to end message transmission or connecting applications at the transport layer can be categorized as either :
  1. connection-oriented e.g. TCP
  2. connectionless e.g. UDP
- The transport layer can be thought of accurately as a transport mechanism e.g. a vehicle whose responsibility is to make sure that its contents (passengers/goods) reach its destination safely and completely, unless a higher or lower layer is responsible for safe delivery.
- The transport layer provides this service of connecting applications collectively through the use of ports. Since IP provides only a top effort delivery, the transport layer is the first layer of the TCP/IP stack to offer dependability. Note that IP can run over a dependable data link protocol such as the High-Level Data Link Control (HDLC). Protocols above transport, such as RPC, also can present reliability.

An example of an attack

Port Scan Attack

- A Port Scan is one of the most popular reconnaissance techniques attackers use to determine services they can crack into. All machines associated to a network run lots of services that use TCP or UDP ports. A port scan helps the attacker locate which ports are available.
- Essentially, a port scan consists of sending a message to each port, one at a time. The kind of response received indicates whether the port is used and can consequently be probed further for flaws.

A way to defense

- Placing a NIDS on the outside of the external firewall will give an early warning advantage, as it should allow the administrator to detect the port scans that typically indicate the start of hacker activity.

Scanned by CamScanner

- However, not all scans will be followed by an actual attack, as the hacker may determine that the network currently has no weaknesses that they can take advantage of. This could lead to large number of alerts that do not require attention.
- One common yet hazardous effect of this is that the staff may lose faith in the IDS and start ignoring alerts. External firewall can be used to provide alerts for the traffic that it has denied.
- By placing NIDS inside the DMZ (De-Militarized Zone, a part of the network that is neither "inside" nor "outside" the corporate entity) the advantage that could be taken is that the tailoring of NIDS attack signature database can be done to consider only those attacks that are appropriate to the systems in the DMZ; at the same time the firewall will have blocked all other traffic.

10.1.3 Network Layer

- **Description :** Network layer solves the problem of receiving packets across a single network. Examples of such protocols are X.25, and the ARPANET's Host/IMP Protocol.
- With the advent of the concept of internetworking, additional functionality was added to this layer, namely receiving data from the source network to the destination network. This usually involves routing the packet across a network of networks, known as an internetwork or (lower-case) internet.
- In the Internet protocol suite, IP performs the basic task of receiving packets of data from source to destination. IP can carry data for a number of different upper layer protocols; these protocols are each recognized by a unique protocol number: ICMP and IGMP are protocols 1 and 2, respectively.
- Some of the protocols carried by IP, such as ICMP (used to send out diagnostic information about IP transmission) and IGMP (used to handle IP Multicast data) are layered on top of IP but perform internetwork layer functions, illustrating an incompatibility between the Internet and the IP stack and OSI model.
- All routing protocols, such as OSPF, and RIP are also part of the network layer. What makes them part of the network layer is that their payload is totally alarmed with management of the network layer. The particular encapsulation of that payload is irrelevant for layering purposes.

Scanned by CamScanner

An example of an attack

Denial of Service attack - SYN Flooding

- The basis of the SYN flooding attack is in the design of the 3-way handshake that begins a TCP connection. In this handshake, the third packet verifies the initiator's ability to receive packets at the IP address it used as the source in its first request, or its return reachability.
- Fig. 10.1.1 shows the sequence of packets exchanged at the beginning of a normal TCP connection.

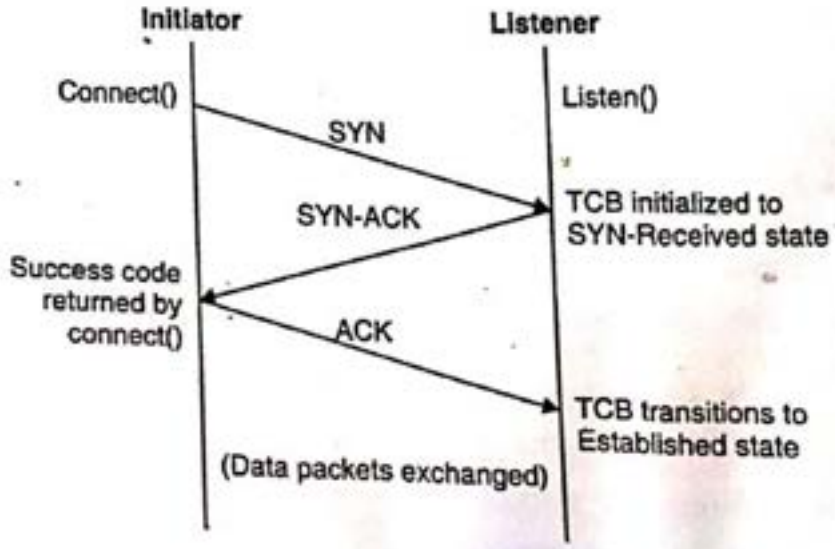


Fig. 10.1.1 : DOS Attack

- The Transmission Control Block (TCB) is a transport protocol data structure (in fact a s of structures in many operations systems) that holds all the information about connection.
- The memory footprint of a single TCB depends on what TCP options and other featu an accomplishment provides and has enabled for a connection.
- Usually, each TCB exceeds at least 280 bytes, and in some operating systems at pres takes more than 1300 bytes.
- The TCP SYN-RECEIVED state is used to point out that the connection is only half o and that the legitimacy of the request is still in question.
- The important aspect to note is that the TCB is allocated based on reception of the S packet before the connection is fully recognized or the initiator's return reachability been verified.

Scanned by CamScanner

- This situation leads to a clear potential DoS attack where incoming SYNs basis the allocation of so many TCBs that a host's kernel memory is exhausted.
- In order to avoid this memory exhaustion, operating systems generally associate a "backlog" parameter with a listening socket that sets a cap on the number of TCBs concurrently in the SYN-RECEIVED state.
- Although this action protects a host's available memory resource from attack, the backlog itself represents another (minor) resource vulnerable to attack.
- With no room left in the backlog, it is not possible to service new connection requests until some TCBs can be reaped or otherwise removed from the SYN-RECEIVED state.
- Depleting the backlog is the goal of the TCP SYN flooding attack, which attempts to send adequate SYN segments to fill the entire backlog.
- The attacker uses source IP addresses in the SYNs that are not likely to start any response that would free the TCBs from the SYN-RECEIVED state.
- Because TCP attempts to be dependable, the target host keeps its TCBs stuck in SYN-RECEIVED for a relatively long time before giving up on the half connection and reaping them.
- In the meantime, service is denied to the application process on the listener for rightful new TCP connection initiation requests. Fig. 10.1.1 shows a simplification of the sequence of events involved in a TCP SYN flooding attack.

A way to defense

- Both end-host and network-based solutions to the SYN flooding attack have merits. Both types of defense are frequently employed, and they usually do not interfere when used in combination.
- Because SYN flooding targets end hosts rather than attempting to tire out the network capacity, it seems logical that all end hosts should implement defenses, and that network-based techniques are an elective second line of defense that a site can employ.
- End-host mechanisms are there in current versions of most common operating systems. Some implement SYN caches, others use SYN cookies after a threshold of backlog usage is crossed, and still others acclimatize the SYN-RECEIVED timer and number of retransmission attempts for SYN-ACKs.
- Because some techniques are known to be futile (increasing backlogs and reducing the SYN-RECEIVED timer), these techniques should definitely not be relied upon. Based on

Scanned by CamScanner

experimentation and analysis, SYN caches seem like the best end-host mechanism on hand.

- This choice is motivated by the facts that they are capable of withstanding profound attacks, they are free from the negative effects of SYN cookies, and they do not need any heuristics for threshold setting as in a lot of hybrid approaches.
- Among network-based solutions, there does not seem to be any tough argument for SYN-ACK spoofing firewall/proxies.
- Because these spoofing proxies rip the TCP connection, they may disable some high-performance or other TCP options, and there seems to be little advantage to this approach over ACK-spoofing firewall/proxies.
- Active monitors should be used when a firewall/proxy solution is administratively impossible or too costly to deploy.
- Ingress and egress filtering is frequently done today (but not ubiquitous), and is a commonly acknowledged practice as part of being a good neighbor on the Internet.
- Because filtering does not cope with distributed networks of drones that use straight attacks, it needs to be supplemented with other mechanisms, and must not be relied upon by an end host.

10.1.4 Data Link Layer

- **Description :** The link layer, which is the method used to move packets from the network layer on two different hosts, is not actually part of the Internet protocol suite, because IP can run over a range of different link layers.
- The processes of transmitting packets on a certain link layer and receiving packets from a given link layer can be controlled both in the software device driver for the network card as well as on firmware or expert chipsets. These will perform data link functions such as adding a packet header to prepare it for transmission, and then in fact transmit the frame over a physical medium.
- For Internet access over a dial-up modem, IP packets are typically transmitted using PPP. For broadband Internet access such as ADSL or cable modems, PPPoE is often used.
- On a local wired network, Ethernet is usually used, and on local wireless networks, IEEE 802.11 is usually used. For wide-area networks, either PPP over T-carrier or E-carrier lines, Frame relay, ATM, or packet over SONET/SDH (POS) are frequently used.

Scanned by CamScanner

An example of an attack

Media Access Control (MAC) Address spoofing :

- MAC spoofing attacks involve the use of a known MAC address of another host to attempt to make the target switch forward frames intended for the remote host to the network attacker.
- By sending a single frame with the other host's source Ethernet address, the network attacker overwrites the CAM table entry so that the switch forwards packets intended for the host to the network attacker.
- Until the host sends traffic it will not get any traffic. When the host sends out traffic, the CAM table entry is rewritten once again so that it moves back to the original port.

A way to defense

- The best way to protect against MAC spoofing is for an intelligent WLAN system to automatically detect MAC spoofing attacks and prohibit offending machines from attaching to the WLAN. This is done in several ways:
- Detection and Containment - One way to avoid MAC spoofing attacks is to flag any occurrence in which the manufacturer name of a detected WLAN adapter differs from the known OUI (Organizationally Unique Identifier) for that equipment.
- Once detected, an intelligent WLAN system can avoid the known attacker from connecting to any nearby APs or any APs located throughout the entire WLAN.

10.1.5 Physical Layer

- **Description :** The Physical layer is responsible for encoding and transmission of data on network communications media.
- It operates with data in the structure of bits that are sent from the Physical layer of the sending (source) device and received at the Physical layer of the destination device.
- Ethernet, Token Ring, SCSI, hubs, repeaters, cables and connectors are standard network devices that function at the Physical layer.
- The Physical layer is also considered the domain of numerous hardware-related network design issues, such as LAN and WAN topology and wireless technology.

Scanned by CamScanner

An example of an attack

- There is not a lot to be said about the attack on this layer.
- Someone can physically carry away your network card or unplug your internet cable.

Syllabus Topic : Packet Sniffing

10.2 Packet Sniffing

- When any data has to be transmitted over the computer network, it is broken down into smaller units at the sender's node called data packets and reassembled at receiver's node in original format. It is the *smallest unit* of communication over a computer network. It is also called a block, a segment, a datagram or a cell.
- The act of capturing data packet across the computer network is called packet sniffing. It is similar to as wire tapping to a telephone network. It is mostly used by *crackers* and *hackers* to collect information illegally about network.

Syllabus Topic : ARP Spoofing

10.3 ARP Spoofing

10.3.1 What Is ARP Spoofing?

- ARP spoofing is a type of attack in which a malicious actor sends falsified ARP (Address Resolution Protocol) messages over a local area network. This results in the linking of an attacker's MAC address with the IP address of a legitimate computer or server on the network.
- Once the attacker's MAC address is connected to an authentic IP address, the attacker will begin receiving any data that is intended for that IP address. ARP spoofing can enable malicious parties to intercept, modify or even stop data in-transit. ARP spoofing attacks can only occur on local area networks that utilize the Address Resolution Protocol.

10.3.2 ARP Spoofing Attacks

The effects of ARP spoofing attacks can have serious implications for enterprises. In the most basic application, ARP spoofing attacks are used to steal sensitive information.

Scanned by CamScanner

Beyond this, ARP spoofing attacks are often used to facilitate other attacks such as :

Denial-of-service attacks

- DoS attacks often leverage ARP spoofing to link multiple IP addresses with a single target's MAC address.
- As a result, traffic that is intended for many different IP addresses will be redirected to the target's MAC address, overloading the target with traffic.

Syllabus Topic : Port Scanning

10.4 Port Scanning

- The act of systematically scanning a computer's ports. Since a port is a place where information goes into and out of a computer, port scanning identifies open doors to a computer.
- Port scanning has legitimate uses in managing networks, but port scanning also can be malicious in nature if someone is looking for a weakened access point to break into your computer.

10.4.1 Types of Port Scans

1. **Vanilla :** The scanner attempts to connect to all 65,535 ports.
2. **Strobe :** A more focused scan looking only for known services to exploit fragmented packets : The scanner sends packet fragments that get through simple packet filters in a firewall.
3. **UDP :** The scanner looks for open UDP ports.
4. **Sweep :** The scanner connects to the same port on more than one machine FTP bounce : the scanner goes through an FTP server in order to disguise the source of the scan.
5. **Stealth scan :** The scanner blocks the scanned computer from recording the port scan activities.

Port scanning in and of itself is not a crime. There is no way to stop someone from port scanning your computer while you are on the Internet because accessing an Internet server opens a port, which opens a door to your computer. There are, however, software products that can stop a port scanner from doing any damage to your system.

Scanned by CamScanner

Syllabus Topic : IP Spoofing

10.5 IP Spoofing

→ (MU - Dec. 15)

Q. 10.5.1 Write in brief about : IP spoofing. (Ref. sec. 10.5) **Dec. 15, 5 Marks**

- In this attack, attacker establishes a large number of "half-open" connections using IP spoofing.
- The attacker first sends SYN packets with the spoofed (faked) IP address to the victim in order to establish a connection.
- The victim creates a record in a data structure and responds with SYN/ACK message to the spoofed IP address, but it never receives the final acknowledgment message ACK for establishing the connection, since the spoofed IP addresses are unreachable or unable to respond to the SYN/ACK messages.
- Although the record from the data structure is freed after a time out period, the attacker attempts to generate sufficiently large number of "half-open" connections to overflow the data structure that may lead to a segmentation fault or locking up the computer.
- In session hijacking, the hacker takes over the control over the TCP session between two machines whereas in spoofing the attacker pretends to be the authenticate user and gain access to other machine.

Syllabus Topic : TCP SYN Flood

10.6 TCP SYN Flood

- TCP SYN flood (a.k.a. SYN flood) is a type of Distributed Denial of Service (DDoS) attack that exploits part of the normal TCP three-way handshake to consume resources on the targeted server and render it unresponsive.
- Essentially, with SYN flood DDoS, the offender sends TCP connection requests faster than the targeted machine can process them, causing network saturation.

Scanned by CamScanner

10.6.1 Attack Description

- When a client and server establish a normal TCP "three-way handshake", the exchange looks like this :

1. Client requests connection by sending SYN (synchronize) message to the server.
2. Server acknowledges by sending SYN-ACK (synchronize-acknowledge) message back to the client.
3. Client responds with an ACK (acknowledge) message, and the connection is established.

- In a SYN flood attack, the attacker sends repeated SYN packets to every port on the targeted server, often using a fake IP address.
- The server, unaware of the attack, receives multiple, apparently legitimate requests to establish communication. It responds to each attempt with a SYN-ACK packet from each open port.
- The malicious client either does not send the expected ACK, or if the IP address is spoofed never receives the SYN-ACK in the first place.
- Either way, the server under attack will wait for acknowledgement of its SYN-ACK packet for some time.

Syllabus Topic : DNS Spoofing

10.7 DNS Spoofing

- Domain Name Server (DNS) spoofing (a.k.a. DNS cache poisoning) is an attack in which altered DNS records are used to redirect online traffic to a fraudulent website that resembles its intended destination.
- Once there, users are prompted to login into (what they believe to be) their account, giving the perpetrator the opportunity to steal their access credentials and other types of sensitive information.
- Furthermore, the malicious website is often used to install worms or viruses on a user's computer, giving the perpetrator long-term access to it and the data it stores.

Scanned by CamScanner

10.7.1 Methods for Executing a DNS Spoofing Attack

- **Man in the middle (MITM)**

  The interception of communications between users and a DNS server in order to route users to a different/malicious IP address.

- **DNS server compromise**

  The direct hijacking of a DNS server, which is configured to return a malicious IP address.

Chapter Ends...

Scanned by CamScanner

CHAPTER 11

Module 5

Denial of Service

Syllabus

Denial of Service: Classic DOS attacks, Source Address spoofing, ICMP flood, SYN flood, UDP flood, Distributed Denial of Service, Defenses against Denial of Service Attacks.

Syllabus Topic : Denial of service - Classic DOS Attacks

11.1 DOS and DDOS Attacks

Denial of service and distributed denial of services is a type of attack that causes legitimate users unable to use services or the resource, or services become unavailable to the legitimate users.

11.1.1 DOS Attacks

→ **(MU - Dec. 15, May 16, Dec. 16, May 17, Dec. 17)**

Q.11.1.1 What is a Denial of service attack? What are the different ways in which an attacker can mount a DOS attack on a system? (Ref. sec. 11.1.1) **Dec. 15, May 17, 10/3 Marks**

Q.11.1.2 Write in brief about Denial of service attacks. (Ref. sec. 11.1.1) **May 16, 5 Marks**

Q.11.1.3 Explain briefly with example, How the following Denial of service attacks occurs. (Ref. sec. 11.1.1) **Dec. 16, Dec. 17, 5 Marks**

- In this attack, the attacker keeps on sending or makes the network or bandwidth overflow by e-mails or spam mail by depriving the victim to access services.
- It is a continuous effort of attackers to make victim unable to use any internet service or resources.

Scanned by CamScanner

- The attacker's main target for websites or services which include financial site bank site or credit card gateway systems.
- The targeted network which are root for DOS are mobile phone network or credit card gateway network.
- Buffer overflow technique is used to make denial service attack. What an attackers does is it takes packet (where is a unit of data) divide into small chunks, the attacker checks for the IP address of the particular network in that packet and floods the network of victim with repeated request. As IP is a fake, from attacker's machine.
- This acts consumes bandwidth which let other service to fail or unavailable for other user.

A DOS attack does following actions

1. Flood whole network with unnecessary traffic.
2. Damage connection between two systems so that communication cannot occur.
3. Disrupt services to legitimate users.
4. Prevents individuals to access network services.

11.1.1(A) Classification of Attacks

(1) Bandwidth attack

- Every website is given particular amount of bandwidth to host (e.g. 50 GB) loading of any websites takes certain amount of time to display whole webpage.
- If more visitors load particular websites page or consumes whole 50 GB bandwidth than particular websites can be ban.
- The attacker does the same by opening 100 pages of site and keeps on loading and refreshing, consuming all bandwidths to make the site out of services.

(2) Logic attack

Attack on the network software to make it vulnerable.

For example : in TCP/IP stack.

(3) Protocol attacks

This attack, consumes more amount of resources in victims system. It is an attack on the particular features of some protocol that are been installed in the victims systems.

Scanned by CamScanner

(4) Unintentional Dos attack

Sometimes because of huge popularity among users the particular wets suddenly end up.

11.1.1(B) Types of DOS Attacks

→ (MU - May 17)

Q. 11.1.4 Explain any three types of DOS attacks in detail. (Ref. sec. 11.1.1(B)) **May 17, 7 Marks**

(1) Flood attack

Attacker keeps on flooding or overloading victim's system with 'n' numbers of ping packets which result into huge traffic which the victim itself cannot handle.

It is very simple to launch but difficult handle.

(2) Ping of Death attack

Sending huge ICMP packet (These packets are used in IP layer or network layer for indicating error message). The attacker sends this huge oversize packet to the victim's system which causes victim's system to crash or freeze resulting in DOS.

(3) SYN attack

- It is a TCP SYN flooding attack, a denial of service attack. In TCP handshaking of network connection is done between sender and receiver through synchronous (SYN) and acknowledgement (ACK) messages.
- An attacker initiates a TCP connection with server with a SYN message. The server in reply sends an acknowledgement message. (SYN – ACK) message.
- The client (attacker) does not respond back with acknowledgement which causes server to wait.
- Due to which it is unable to connect with other client. This fills up the buffer space for SYN message preventing other for communicate.

1. Clients sends synchronize (SYN) packet to server.
2. Servers send syn-ack (SYN – ACK) to client.
3. Clients responds back with ACK packet and connect is established client as shown in Fig. 11.1.1.

Scanned by CamScanner

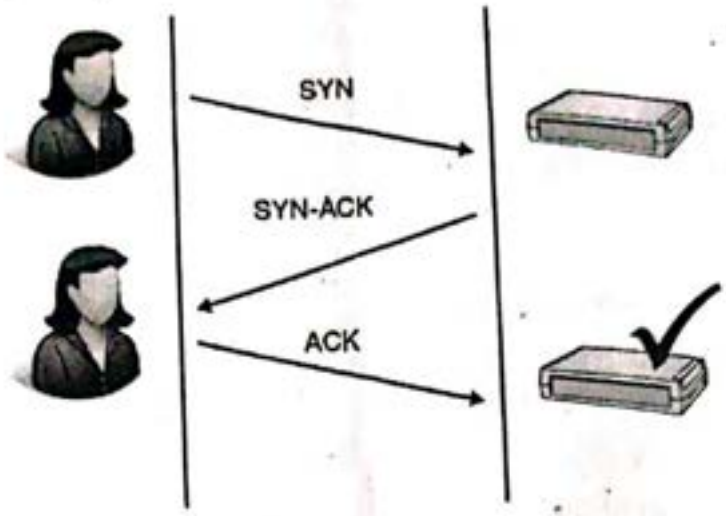


Fig. 11.1.1 : 3 way handshake

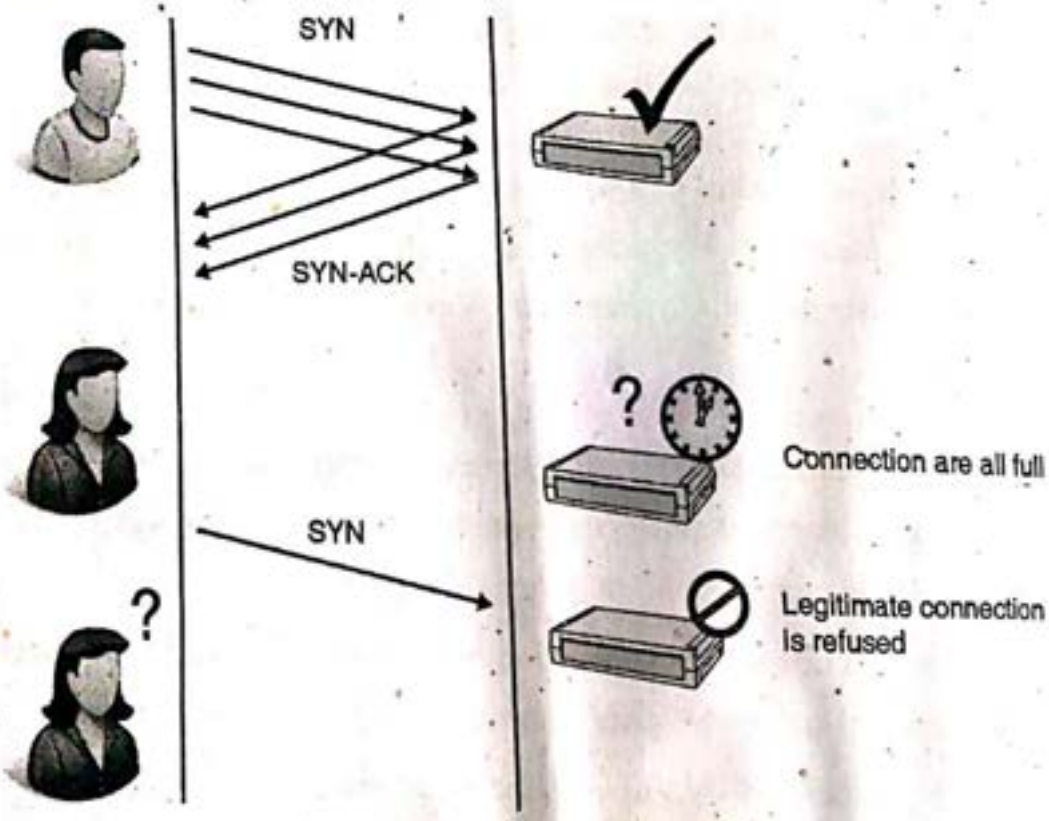


Fig. 11.1.2 : Chaotic handshake

1. Client sends multiple SYN packets to all with bad address.
2. **Server send SYN :** ACK packets to in correct address.
3. Legitimate user is denied because server cannot accept additional connection as shown in Fig. 11.1.2.

Scanned by CamScanner

4. **Teardrop attack :** It is an attack when packets are overlapped with each other and the receiver is not able to reassemble them, usually corrupted packets are send by attacker to hang or freeze the system.

5. **Nuke :** It is an attack of sending invalid ICMP packet to the target which slow down the affected computer till it is completely stop.

6. **Smurf attack :** It is an attack in which IP address broad casting is done. A Smurf program is used to make network inoperable. It builds a packet which seems to originate from another address. This packet contains ICMP ping. The echo responses are sent back to victim. Maximum ping and echo make network unusable.

The various tools used for DOS attack or Jolt2, Nemesy, Targa etc.

Syllabus Topic : Source Address Spoofing

11.2 Source Address Spoofing

- In this attack, attacker establishes a large number of "half-open" connections using IP spoofing.
- The attacker first sends SYN packets with the spoofed (faked) IP address to the victim in order to establish a connection.
- The victim creates a record in a data structure and responds with SYN/ACK message to the spoofed IP address, but it never receives the final acknowledgment message ACK for establishing the connection, since the spoofed IP addresses are unreachable or unable to respond to the SYN/ACK messages.
- Although the record from the data structure is freed after a time out period, the attacker attempts to generate sufficiently large number of "half-open" connections to overflow the data structure that may lead to a segmentation fault or locking up the computer.
- In session hijacking, the hacker takes over the control over the TCP session between two machines whereas in spoofing the attacker pretends to be the authenticate user and gain access to other machine.

Scanned by CamScanner

Syllabus Topic : ICMP Flood

11.3 ICMP Flood

Q. 11.3.1 Explain ICMP flood attack. (Ref. sec.11.3)

- Ping flood, also known as ICMP flood, is a common Denial of Service (DoS) attack in which an attacker takes down a victim's computer by overpowering it with ICMP echo requests, also known as pings.
- The attack involves flooding the victim's network with request packets, knowing that the network will react with an equal number of reply packets.
- Additional methods for bringing down a target with ICMP requests include the use of convention tools or code, such as hping and scapy.
- This strains both the incoming and outgoing channels of the network, consuming considerable bandwidth and resulting in a denial of service.

Attack Description

- Normally, ping requests are used to test the connectivity of two computers by measuring the round-trip time from when an ICMP echo request is sent to when an ICMP echo reply is established. During an attack, however, they are used to overload a target network with data packets.
- Executing a ping flood is reliant on attackers knowing the IP address of their target. Attacks can therefore be divided into three categories, based on the target and how its IP address is resolved.
- A targeted local disclosed ping flood targets a sole computer on a local network. The attacker needs to have physical access to the computer in order to discover out its IP address. A successful attack would result in the target computer being taken down.
- A router disclosed ping flood targets routers in order to interrupt communications between computers on a network. It is dependent on the attacker knowing the internal IP address of a local router. A successful attack would result in all computers connected to the router being taken down.
- A blind ping flood involves using an external program to disclose the IP address of the target computer or router before executing an attack.

Scanned by CamScanner

There are a number of ping commands that can be used to aid an attack, including :

- The –n command, which is used to specify the number of times a request is sent.
- The –l command, which is used to specify the amount of data sent with each packet.
- The –t command, which is used to continue pinging until the host times out.
- Note that in order for a ping flood to be sustained, the attacking computer must have access to extra bandwidth than the victim. This limits the ability to carry out a DoS attack, particularly against a large network.

Syllabus Topic : SYN flood

11.4 SYN Flood

Q. 11.4.1 Write brief note on Syn flood. (Ref sec. 11.4)

- It is a TCP SYN flooding attack, a denial of service attack. In TCP handshaking of network connection is done between sender and receiver through synchronous (SYN) and acknowledgement (ACK) messages.
- An attacker initiates a TCP connection with server with a SYN message. The server in reply sends an acknowledgement message. (SYN – ACK) message.
- The client (attacker) does not respond back with acknowledgement which causes server to wait.
- Due to which it is unable to connect with other client. This fills up the buffer space for SYN message preventing other for communicate.

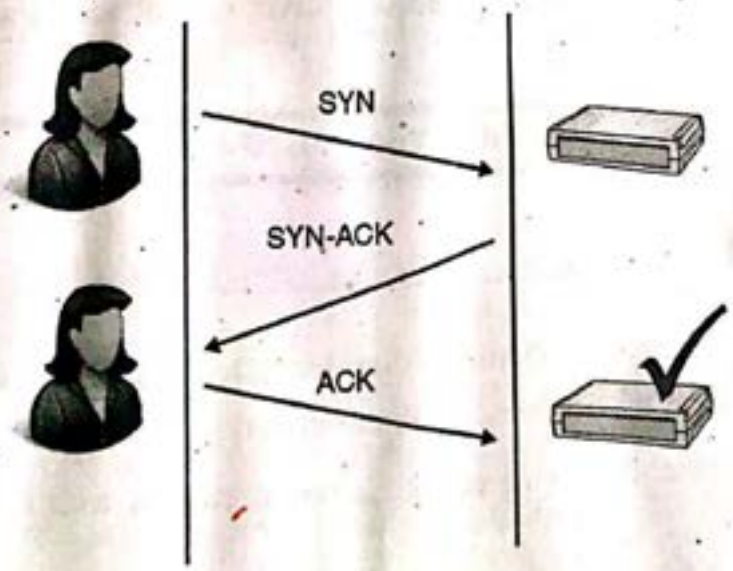


Fig. 11.4.1 : 3 way handshake

Scanned by CamScanner

1. Clients sends synchronize (SYN) packet to server.
2. Servers send syn-ack (SYN – ACK) to client.
3. Clients responds back with ACK packet and connect is established client as shown in Fig. 11.4.1.

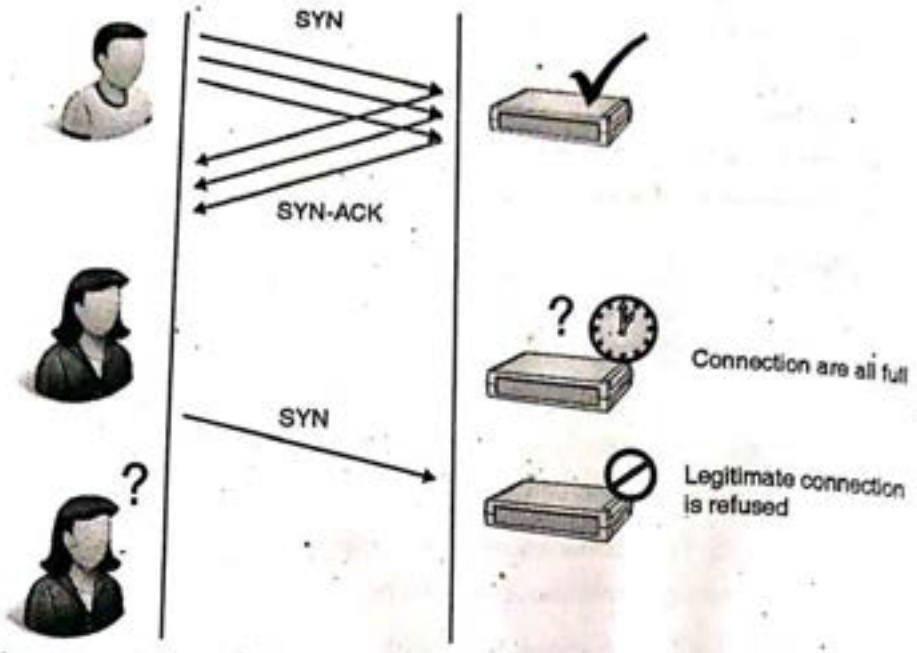


Fig. 11.4.2 : Chaotic handshake

1. Client sends multiple SYN packets to all with bad address.
2. **Server send SYN :** ACK packets to in correct address.
3. Legitimate user is denied because server cannot accept additional connection as shown in Fig. 11.4.2.

Syllabus Topic : UDP Flood

11.5 UDP Flood

- A **UDP flood attack** is a Denial-of-Service (DoS) attack using the User Datagram Protocol (UDP), a connectionless computer networking protocol.
- Using UDP for denial-of-service attacks is not as easy as with the Transmission Control Protocol (TCP).

Scanned by CamScanner

- However, a UDP flood attack can be initiated by sending a huge number of UDP packets to random ports on a remote host.
- As a result, the distant host will :
  1. Verify for the application listening at that port;
  2. See that no application is listening at that port;
  3. Reply with an ICMP Destination Unreachable packet.
- Thus, for a large number of UDP packets, the ill-treated system will be forced into sending many ICMP packets, eventually leading it to be unreachable by other clients.
- The attacker(s) may also spoof the IP address of the UDP packets, ensuring that the unnecessary ICMP return packets do not reach them, and anonymizing their network location(s).
- Most operating systems ease this part of the attack by limiting the rate at which ICMP responses are sent.

Syllabus Topic : Distributed Denial of Service

11.6 Distributed Denial of Service Attacks

Q.11.6.1 Explain Distributed Denial of Service Attack in detail. (Ref. sec. 11.6)

A Distributed Denial-of-Service (DDoS) attack is an attack in which multiple compromised computer systems attack a target, such as a server, website or other network resource, and cause a denial of service for users of the targeted resource. The flood of incoming messages, connection requests or malformed packets to the target system forces it to slow down or even crash and shut down, thereby denying service to legitimate users or systems.

11.6.1 Distributed Denial of Service Attacks

- Distributed denial of service, it is where an attacker uses your own computer to attack on another computer.
- It takes advantage of loopholes and security vulnerability to take control on for computer to send vulnerability spam or send huge data to other computers.
- The systems which are used for attacking victim computer are called as Zombie systems.
- Various tools to launch DDOS attack are Trinoo, Tribe flood, shaft etc.

Scanned by CamScanner

Measures to protect from DOS/DDOS attack are :

- Implementing filters on routers.
- Disable unused network services.
- Examine the physical security routinely.
- Maintain regular backup schedules and policies.
- Maintain password policies.
- Using fault tolerant network configuration.
- Tools for detecting DOS/DDOS attacks Zombie Zapper, find – DDOS, remote intrusion detector (RID).

11.6.2 Characteristics of Distributed Denial of Service Attacks

- A denial of service attack is characterized by an explicit attempt by an attacker to prevent legitimate users of a service from using the desired resources. Examples of denial of service attacks include :
  - Attempts to "flood" a network, thereby preventing legitimate network traffic.
  - Attempts to disrupt connections between two machines, thereby preventing access to a service.
  - Attempts to prevent a particular individual from accessing a service.
  - Attempts to disrupt service to a specific system or person.
- The distributed format adds the "many to one" dimension that makes these attacks more difficult to prevent. A distributed denial of service attack is composed of four elements, as shown in Fig. 11.6.1. First, it involves a victim, i.e., the target host that has been chosen to receive the brunt of the attack. Second, it involves the presence of the attack daemon agents.
- These are agent programs that actually conduct the attack on the target victim. Attack daemons are usually deployed in host computers. These daemons affect both the target and the host computers.
- The task of deploying these attack daemons requires the attacker to gain access and infiltrate the host computers. The third component of a distributed denial of service attack is the control master program. Its task is to coordinate the attack.

Scanned by CamScanner

Finally, there is the real attacker, the mastermind behind the attack. By using a control master program, the real attacker can stay behind the scenes of the attack. The following steps take place during a distributed attack :

1. The real attacker sends an "execute" message to the control master program.
2. The control master program receives the "execute" message and propagates the command to the attack daemons under its control.
3. Upon receiving the attack command, the attack daemons begin the attack on the victim.

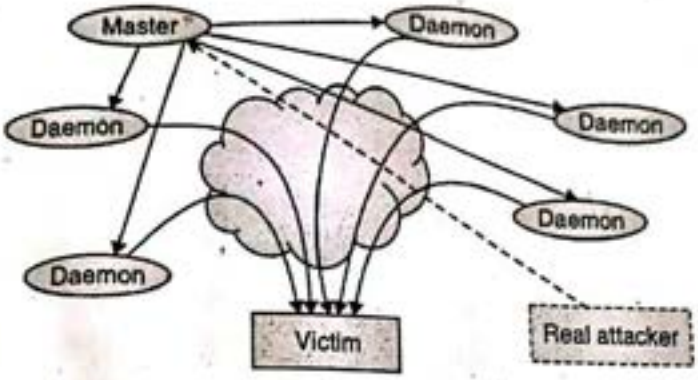


Fig. 11.6.1 : Chaotic handshake

- Although it seems that the real attacker has little to do but sends out the "execute" command, he/she actually has to plan the execution of a successful distributed denial of service attack.
- The attacker must infiltrate all the host computers and networks where the daemon attackers are to be deployed.
- The attacker must study the target's network topology and search for bottlenecks and vulnerabilities that can be exploited during the attack.
- Because of the use of attack daemons and control master programs, the real attacker is not directly involved during the attack, which makes it difficult to trace who spawned the attack.
- In the following subsections, we review some well-known attack methods (Smurf, SYN Flood, and User Datagram Protocol (UDP) Flood) and the current distributed denial of service methods (Trinoo, Tribe Flood Network, Stacheldraht, Shaft, and TFN2K).

We describe defense mechanisms that can be employed by networks.

Scanned by CamScanner

11.6.3 Methods of Denial of Service Attacks

- We described below some widely known basic denial of service attack methods that are employed by the attack daemons.
- Smurf attack involves an attacker sending a large amount of Internet Control Message Protocol (ICMP) echo traffic to a set of Internet Protocol (IP) broadcast addresses. The ICMP echo packets are specified with a source address of the target victim (spoofed address).
- Most hosts on an IP network will accept ICMP echo requests and reply to them with an echo reply to the source address, in this case, the target victim. This multiplies the traffic by the number of responding hosts. On a broadcast network, there could potentially be hundreds of machines to reply to each ICMP packet.
- The process of using a network to elicit many responses to a single packet has been labeled as an "amplifier". There are two parties who are hurt by this type of attack: the intermediate broadcast devices (amplifiers) and the spoofed source address target (the victim). The victim is the target of a large amount of traffic that the amplifiers generate. This attack has the potential to overload an entire network.
- SYN Flood attack is also known as the Transmission Control Protocol (TCP) SYN attack, and is based on exploiting the standard TCP three-way handshake.
- The TCP three-way handshake requires a three-packet exchange to be performed before a client can officially use the service. A server, upon receiving an initial SYN (synchronize/start) request from a client, sends back a SYN/ACK (synchronize/acknowledge) packet and waits for the client to send the final ACK (acknowledge).
- However, it is possible to send a barrage of initial SYN's without sending the corresponding ACK's, essentially leaving the server waiting for the non-existent ACK's. Considering that the server only has a limited buffer queue for new connections, SYN Flood results in the server being unable to process other incoming connections as the queue gets overloaded.
- *UDP Flood* attack is based on UDP echo and character generator services provided by most computers on a network. The attacker uses forged UDP packets to connect the echo service on one machine to the character generator (chargen) service on another machine.

Scanned by CamScanner

- The result is that the two services consume all available network bandwidth between the machines as they exchange characters between themselves. A variation of this attack called ICMP Flood, floods a machine with ICMP packets instead of UDP packets.
- The techniques are listed in chronological order. It can be observed that as time has passed, the distributed techniques (Trinoo, TFN, Stacheldraht, Shaft, and TFN2K) have become technically more advanced and, hence, more difficult to detect.
- *Trinoo* uses TCP to communicate between the attacker and the control master program. The master program communicates with the attack daemons using UDP packets
- Trinoo's attack daemons implement UDP Flood attacks against the target victim.
- *Tribe Flood Network* (TFN) uses a command line interface to communicate between the attacker and the control master program. Communication between the control master and attack daemons is done via ICMP echo reply packets. TFN's attack daemons implement Smurf, SYN Flood, UDP Flood, and ICMP Flood attacks.
- *Stacheldraht* (German term for "barbed wire") is based on the TFN attack. However, unlike TFN, Stacheldraht uses an encrypted TCP connection for communication between the attacker and master control program. Communication between the master control program and attack daemons is conducted using TCP and ICMP, and involves an automatic update technique for the attack daemons. The attack daemons for Stacheldraht implement Smurf, SYN Flood, UDP Flood, and ICMP Flood attacks.
- *Shaft* is modeled after Trinoo. Communication between the control master program and attack daemons is achieved using UDP packets. The control master program and the attacker communicate via a simple TCP telnet connection. A distinctive feature of Shaft is the ability to switch control master servers and ports in real time, hence making detection by intrusion detection tools difficult.
- *TFN2K* uses TCP, UDP, ICMP, or all three to communicate between the control master program and the attack daemons. Communication between the real attacker and control master is encrypted using a key-based CAST-256 algorithm. In addition, TFN2K conducts covert exercises to hide itself from intrusion detection systems. TFN2K attack daemons implement Smurf, SYN, UDP, and ICMP Flood attacks.

Scanned by CamScanner

Syllabus Topic : Defenses against Denial of Service Attacks

11.7 Defenses against Attacks

Many observers have stated that there are currently no successful defenses against a fully distributed denial of service attack. This may be true. Nevertheless, there are numerous safety measures that a host or network can perform to make the network and neighboring networks more secure. These measures include :

- **Filtering Routers :** Filtering all packets entering and leaving the network protects the network from attacks conducted from neighboring networks, and prevents the network itself from being an unaware attacker. This measure requires installing ingress and egress packet filters on all routers.
- **Disabling IP Broadcasts :** By disabling IP broadcasts, host computers can no longer be used as amplifiers in ICMP Flood and Smurf attacks. However, to defend against this attack, all neighbouring networks need to disable IP broadcasts.
- **Applying Security Patches :** To guard against denial of service attacks, host computers must be updated with the latest security patches and techniques. For example, in the case of the SYN Flood attack, there are three steps that the host computers can take to guard themselves from attacks : increase the size of the connection queue, decrease the time-out waiting for the three-way handshake, and employ vendor software patches to detect and circumvent the problem.
- **Disabling Unused Services :** If UDP echo or charge-n-services are not required, disabling them will help to defend against the attack. In general, if network services are unneeded or unused, the services should be disabled to prevent tampering and attacks.
- **Performing Intrusion Detection :** By performing intrusion detection, a host computer and network are guarded against being a source for an attack, as while as being a victim of an attack. Network monitoring is a very good pre-emptive way of guarding against denial of service attacks.

By monitoring traffic patterns, a network can determine when it is under attack, and can take the required steps to defend itself. By inspecting host systems, a host can also prevent it from hosting an attack on another network.

Chapter End

Scanned by CamScanner

Denial of Service
ks

nses against a
ess, there are
e network and

k protects the
prevents the
res installing

can no longer
efend against

attacks, host
niques. For
hat the host
size of the
dshake, and

ot required,
ork services
npering and

on, a host
as while as
ive way of

attack, and
t can also

apter Ends...

CHAPTER 12

Module 5

Internet Security Protocols

Syllabus

Internet Security Protocols : SSL, IPSEC, Secure Email : PGP, Firewalls, IDS and types, Honey pots.

Syllabus Topic : Internet Security Protocols - SSL, IPSEC

12.1 Secure Socket Layer (SSL)

→ **(MU - Dec. 15, Dec. 17)**

Q. 12.1.1 List the functions of the different protocols of SSL. Explain the handshake protocol. (Ref. secs. 12.1 and 12.1.1(A)) **Dec. 15, 5 Marks**

Q. 12.1.2 What are the different protocols in SSL? How do the client and server establish an SSL connection ? (Ref. sec. 12.1) **Dec. 17, 5 Marks**

- Secure Socket layer invented by Netscape communications in 1994. Secure Socket layer is an internet protocol used for securely exchanging the information between client's web browser and the web server.
- Secure socket layer ensure the authentication, data integrity and data confidentiality between web browser and web server i.e. it creates a secure tunnel between client and server. The main role of SSL is to provide the security to web traffic in all the way.
- The current version of SSL is 3.0. The position of SSL in TCP/ IP protocol suite is shown in Fig. 12.1.1.
- SSL is works in between application layer and transport layer the reason SSL is also called as **Transport Layer Security (TLS).**

Scanned by CamScanner

- **Transport Layer Security (TLS)** protocol is used to ensure security between communicating applications and their users on the Internet.
- Main function of transport layer protocol is to protect attacker when a server and client communicate, it ensures that attacker or third party should not modify or tamper with any message.
- TLS is the successor to the Secure Sockets Layer (SSL).

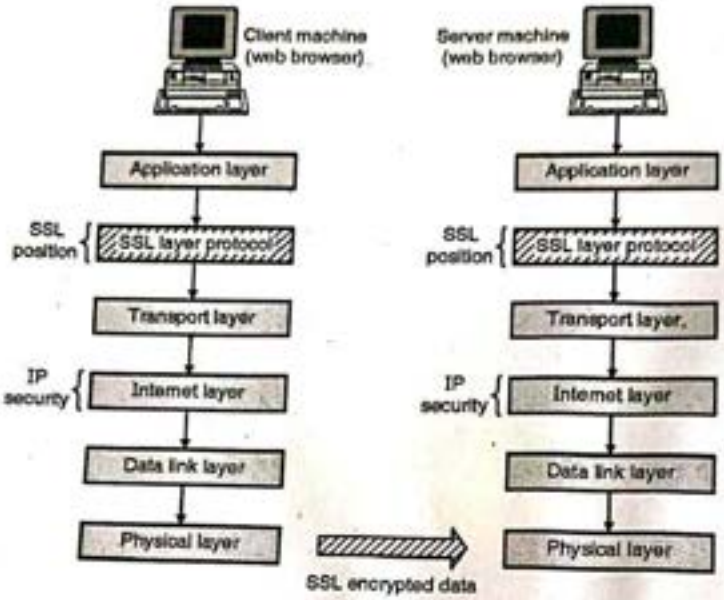


Fig. 12.1.1 : Position of SSL in TCP/ IP protocol suite

- The data will not be passed directly to transport layer instead it will passed to secure socket layer.
- Secure Socket Layer will perform encryption to the data received by application layer and add its own encryption information header called SSH i.e. Secure Socket Layer Header. In the receiver's end SSL will remove the SSH header and then pass data to application layer.
- The Fig. 12.1.1 shows position of SSL protocol in TCP/ IP protocol suite. SSL protocol uses digital certificate and digital signature for securely communication between client machine and server machine.
- SSL encrypt the data received from application layer of client machine and add its own header (SSL Header) into the encrypted data and send encrypted data to the server side.

Scanned by CamScanner

- Upon receiving encrypted data, server removes the SSL header and decrypts the data and sends the decrypted data to application layer.
- SSL is composed of four protocols in two layers, which support SSL as shown in Fig. 12.1.2.
- Out of the four, the two most important protocols that are at the heart of SSL are the SSL Handshake Protocol and the SSL Record Protocol, the other two protocols such as SSL Change Cipher Specification and the SSL Alert Protocol play a minor role relatively to previous two protocols.
- The role of these higher-level protocols is the connection establishment, use of required cipher techniques for data encryption and alert (warning, error if any) generation before starting actual data transmission process between client and server.

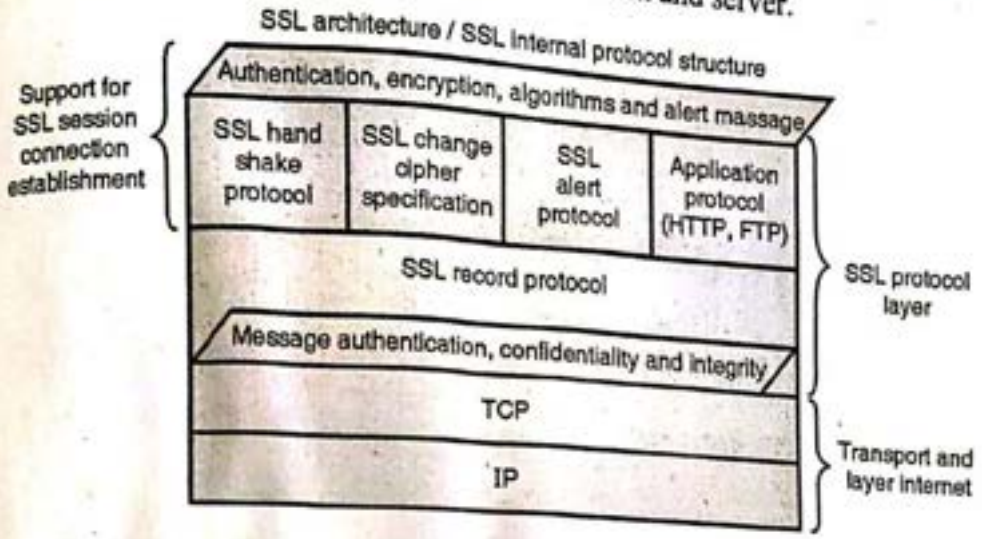


Fig. 12.1.2 : SSL protocols internal architecture

- The SSL Record Protocol is responsible for encrypted data transmission and encapsulation of the data sent by the higher layer protocols (handshake, alert, HTTP) also to provide basic security services to higher layer protocols.
- SSL was designed to make use of TCP protocol to provide a reliable secure process-to-process delivery of entire message/packets. We will discuss how client machine securely communicate with the server machine by using underlying network architecture.

12.1.1 Working of SSL

We will discuss SSL Handshake Protocol and the SSL Record Protocol in details.

Scanned by CamScanner

12.1.1(A) Handshake Protocol

→ (MU - May 16)

Q. 12.1.3 Write in brief about SSL handshake protocol. (Ref. sec. 12.1.1(A)) **May 16, 5 Marks**

- As the name suggests when we meet to our friend/colleagues, we have habit to say hi/hello and do the *shake-hands* with each other before starting our actual communication. SSL handshake protocol uses somewhat same ideology but in terms of client and server.
- The first sub-protocol of SSL called *handshake protocol* used for secure communication between client and the server using an SSL enabled connections.
- In this protocol client authentication to the server is more important that server authentication because server has different options available for client authentication. The details **steps of SSL handshake protocol** are shown in Fig. 12.1.3.

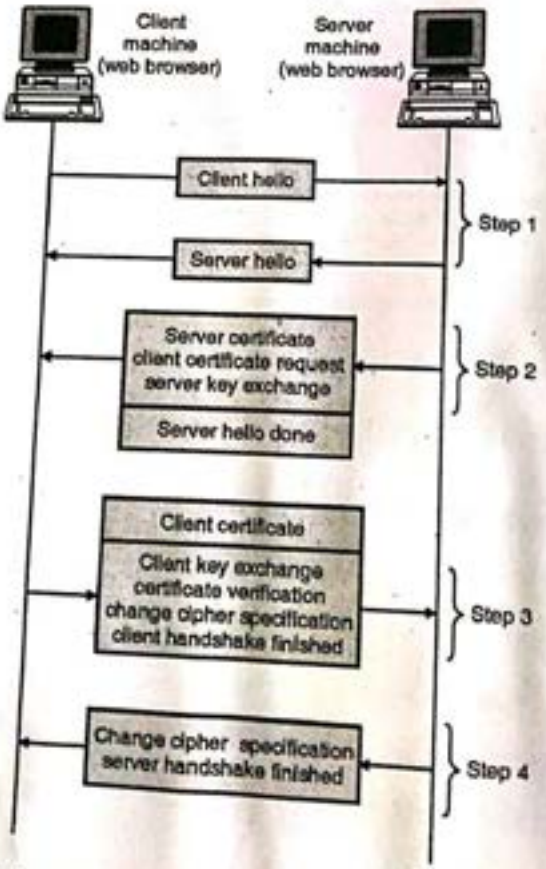


Fig. 12.1.3 : SSL handshake protocol

Scanned by CamScanner

1. It is used by client and server to start communication using SSL enabled connection.
2. The handshaking is done 4 phases :

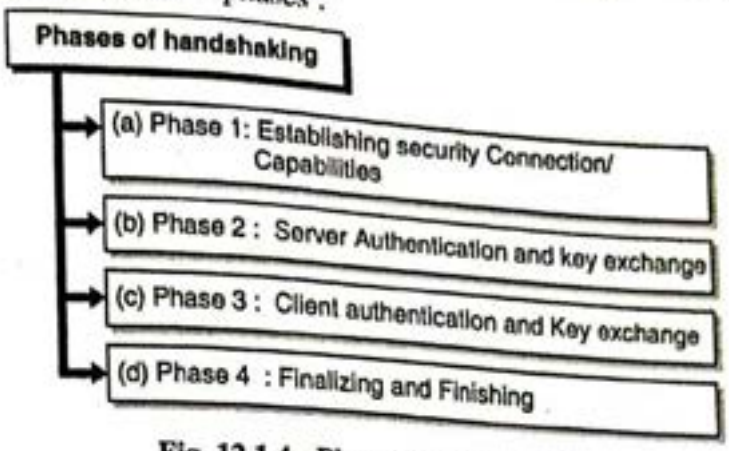


Fig. 12.1.4 : Phases of handshaking

(a) Phase 1 : Establishing Security Connection/Capabilities

In this phase logical connections is established between client and server and establish security capabilities associated with that connections. It consists of two messages, the client hello and the server hello.

Client hello

The client hello message contains the following parameters.

(i) The highest SSL version number which the client can support.

(ii) A 32-bit timestamp and a 28-byte random field that together serve as nonce during key exchange to prevent re- play attacks.

(iii) A session id that defines the session (a variable length session identifier).

(iv) There is a cipher suite parameter that contains the entire list of cryptographic algorithms which supports client's system.

(v) A list of compression methods that can be supported by client.

Server

(i) The SSL version number, the highest among both SSL number of client and server, will be supported by client and other will be supported by server.

(ii) A 32 byte random number that will be used for master secret generation, however this random number is totally independent from the random number of client.

(iii) A session id that defines the session.

Scanned by CamScanner

(iv) A cipher suite contains the list of all cryptographic algorithms that is sent by the client from which the server will select the algorithm.

(v) A list of compression methods sent by the client from which the server will select the method.

(b) Phase 2 : Server Authentication and Key Exchange

In this phase, the server authenticates itself if it is needed. The server sends its certificate, its public key, and also request certificate (digital certificate) from the client.

- **Certificate :** The server sends a certificate message to authenticate itself to the client. If the key exchange algorithm is Diffie Hellman than no need of authentication.
- **Server key Exchange :** This is optional. It is used only if the server doesn't sends its digital certificate to client.
- **Certificate Request :** The server can request for the digital certificate of client. The client's authentication is optional.
- **Server Hello done :** The server message hello done is the last message in phase 2. This indicates to the client that the client can now verify all the certificates received by the server. After this hello message done, the server waits for the client's side response in phase 3.

(c) Phase 3 : Client Authentication and Key Exchange

In this phase, the client authenticates itself if it is needed. The client sends its certificate, client key exchange and certificate verify to the server.

- **Certificate :** Client certificate is optional, it is only required if the server had requested for the client's digital certificate. If client doesn't have client's digital certificate it can send no certificate message or an Alert message to the server. Then it is upto server's decision whether to continue with the session or to abort the session.
- **Client key Exchange :** The client sends a client key exchange, the contents in this message are based on key exchange algorithms between both the parties.
- **Certificate verify :** It is necessary only if the server had asked for client authentication. The client has already sent its certificate to the server. But additionally if server wants then the client has to prove that it is authorized holder of the private key .The server can verify the message with its public key which was already sent to ensure that the certificate belongs to client.

Scanned by CamScanner

(d) Phase 4 : Finish

The client and server send messages to finish the handshaking protocol. It contains 4 steps. The first two messages are from the client i.e. change cipher specs, finished. The server responds back with change cipher spec and finished.

- **Change cipher spec :** It is a client side message telling about the current status of cipher protocols and parameters which has been made active from pending state.
- **Finished :** This message announces the finish of the handshaking protocol from client side.
- **Change Cipher spec :** This message is sent by server to show that it has made all the pending state of cipher protocols and parameters to active state.
- **Finished :** This message announces the finish of the handshaking protocol from server and finally handshaking is totally completed.

12.1.1(B) Alert Protocol

- SSL uses the Alert protocol for reporting error that is detected by client or server, the party which detects error sends an alert message to other party. If error is serious then both parties terminate the session.
- Table 12.1.1 shows the types of alert messages. SSL alert protocol is the last protocol of SSL used transmit alerts (warnings, errors, fatal etc.) if any via SSL record protocol to the client or server.
- The SSL alert protocol format is shown in Fig. 12.1.5. Alert protocol uses two bytes to generate alert. First 1 byte indicates two values either 1 or 2. "1" value indicate warning and "2" value indicate a fatal error (if fatal error terminate the session/ connection).
- Whereas second 1 byte indicates predefined error code either the server or client detects any error it sends an *alert* containing the error (error occurred during handshaking, error occurred during data processing at server or client side, certificate defeats, etc.)

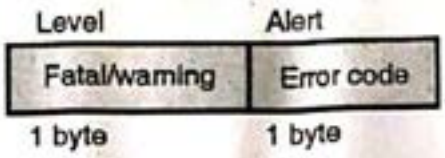


Fig. 12.1.5 : SSL Alert protocol

Scanned by CamScanner

Table 12.1.1 : Types of alert messages

Alert Code	Alert Message	Description
0	close_notify	No more message from sender
10	unexpected_message	An incorrect message received
20	bad_record_mac	A wrong MAC received
30	decompression_failure	Unable to decompress.
40	handshake_failure	Unable to finalize handshake by the sender.
42	bad_certificate	Received a corrupted certificate.
42	Nocertificate	Client has no certificate to send to server.
42	Certificate expired	Certificate has expired.

12.1.1(C) Record Protocol

- After completion of successful SSL handshaking the keen role of SSL record protocol starts now.
- SSL record protocol is second sub-protocol of SSL also called lower level protocol.
- As defined earlier the SSL Record Protocol is responsible for encrypted data transmission and encapsulation of the data sent by the higher layer protocols (handshake, alert, HTTP) also to provide basic security services to higher layer protocols.
- SSL record protocol is basics for data transfer and specially used to build a data path between client and server and encrypt the data path before communication.
- SSL record protocol provides different service like data authentication; data confidentiality though encryption algorithms and data integrity through message authentication (MAC) to SSL enabled connections.
- The details steps involved in SSL record protocol and SSL record header format as shown in Fig. 12.1.6.

Scanned by CamScanner

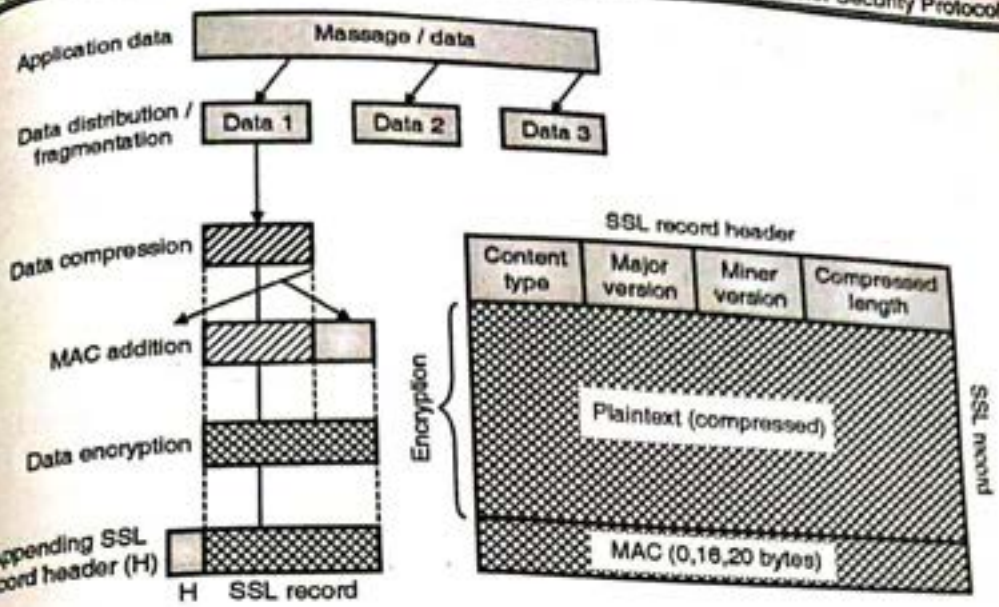


Fig. 12.1.6 : Record protocol and record header

- At this stage all necessary authentication and cryptographic parameters are exchanged between client and server now it's time of secure SSL data transmission through SSL record protocol.
- SSL record protocol takes application data i.e. actual data that client wants to send over server. Divide this data into the different blocks for each length should not exceed 16384 bytes this process is called as *data distribution* or *data fragmentation*.
- Next step is Data compression using lossless compression techniques; compression size of data should not exceed 1024 bytes.
- After the data fragmentation and compression step the MAC ((Message Authentication Code) is computed over the data and MAC is then appended to the compressed data (the data is now encapsulated) to form a new encrypted data / payload.
- The compressed data and MAC again goes through data encryption process. SSL record protocol uses symmetric key cryptographic techniques like DES, triple DES, AES, and IDEA because these techniques are specially designed to operate on block cipher.
- Finally SSL record header is pretended onto each encrypted blocks obtained from encryption process.
- Each output block produced by the SSL Record Protocol is referred to as an SSL record. The length of a record is not to exceed 32, 767 bytes.

Scanned by CamScanner

- **SSL record header** refer Fig 12.1.6 consist of 8-bit content type to which identify nature of the message whether any application data or connection termination or any error message.
- Next field is Major Version which is 8-bit field used to indicate latest version of SSL is in use (e.g., 3). Minor Version which is 8-bit field indicates the lowest version of SSL is in use (e.g., 0).
- Plaintext (compressed) / compressed length which is 16-bit field indicates the length of the plaintext being compressed.
- Finally sends SSL layer encrypted data to TCP and IP (Transport and Internet layer) for necessary transmission over network
- At the receiver end, the encrypted blocks are decrypted and then checked for data authentication, data confidentiality and data integrity, reassemble these data into single unit, and delivered to the application-layer protocol.
- The Record Protocol provides **two services** in SSL connection :

  a) **Confidentiality :** This can be achieved by using secret key, which is already defined by handshake protocol.

  b) **Integrity :** The handshake protocol defines a shared secret key that is used to assure the message integrity.
- Following are the **operations** performed in Record protocol after connection is established and authentication is done of both client and server.

  1. **Fragmentation :** The original message that is to be sent is broken into blocks. The size of each block is less than or equal to 2^{14} (16384)bytes.
  2. **Compression :** The fragmented blocks are compressed which is optional. It should be noted that the compression process must not result into loss of original data.
  3. **Addition of MAC :** The Message authentication code (a short piece of information used to authenticate a message for integrity and assurance of message) for each block is to be calculated using shared secret key.
  4. **Encryption :**The overall steps including message is encrypted using symmetric key but the encryption should not increase the overall block size.

Scanned by CamScanner

5. **Prepend Header :** After all the above operations, header is prepended in the encrypted block which contains following fields :
   - Content type (8 bits) specifies which protocol is used for processing.
   - Major Version (8 bits) specifies the major version of SSL used, for example if SSL version 3.1 is in use than this field contains 3.
   - Minor Version (8 bits) specifies the minor version of SSL used, for example if SSL version 3.0 is in use than this field contains 0.
   - Compressed length (16 bit) specifies the length in bytes of the original plain text block.

Syllabus Topic : IPSEC

12.2 IP Security Protocols

→ (MU - May 16)

Q. 12.2.1 Write in brief about IPSec protocols for security. (Ref. sec. 12.2) **May 16, 5 Marks**

Encryption of data and its authenticity is prime concern for secure communication, to avail this two features, IPSec provides two protocols at network layer :

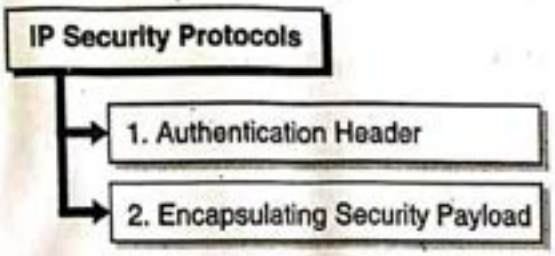


Fig. 12.2.1 : IP Security Protocols

12.2.1 Authentication Header

- It is designed for authentication, integrity of payload which is carried in IP Packet. It is first protocol of IPSec called Authentication Header (AH) protocol designed to provide data authentication (to identify source host), data integrity (if data get modified while in transit) and non-repudiation but doesn't provide data confidentiality (if attacker able to access the contents of a message) because Authentication Header does not encrypt the data/ IP packet.
- The main functionality of this protocol is protection against replay attacks (sending same data to receiver again and again) and protection against tampering of data over a network.

Scanned by CamScanner

- Authentication Header is also used to protect the upper-layer or the entire IP packet, with the help of message authentication code (MAC - used to generate fixed length value from message and secret key to provide authentication) using well known hashing algorithms like MD5 or SHA1.
- By using Hash function and symmetric key algorithm, message digest is calculated and inserted in authentication data as shown in Fig.12.2.2 because of this AH protocol provides data authentication and data integrity, but not confidentiality or privacy.
- The internal fields of authentication header format are shown in Fig. 12.2.2.
- This protocol uses cryptographic checksum which is similar to hash function or message digest, the checksum is inserted in authentication header and placed in location depends on which mode it is using (tunnel mode or transport mode).

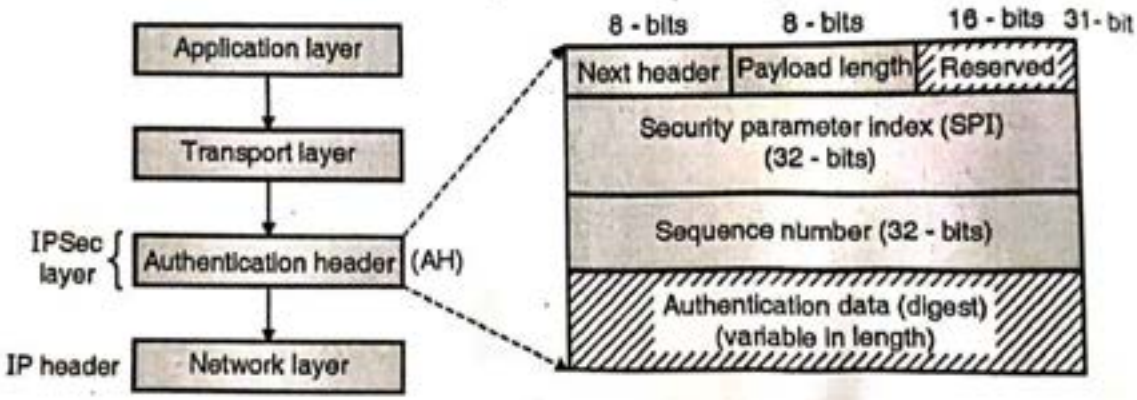


Fig. 12.2.2 : Authentication Header

☞ **A brief description of each field**

Next header (8 bits)

- The next header is an 8 - bit filed which is used to identify the type of payload/ data carried by IP packet.
- Identifies the type of header immediately following this header.

Payload length (8 bits)

- The payload header is also an 8 - bit filed which defines length of the authentication header.
- Length of the AH in 32-bit words minus 2.

Reserved (16 bits)

AH contains 16 - bit field which is reserved for future use and always set to zero.

Scanned by CamScanner

Security Parameter Index (SPI) (32 bits)

- SPI is a 32-bit field used in combination with source IP address, destination IP address and AH security protocol to uniquely identify a security association (SA) for the traffic to which IP packet belongs, we will discuss SA it next bit.
- This field also defining which different security algorithms and keys were used to calculate the Message Authentication Code (MAC).

Sequence number (32 bits)

- It is also a 32 bit field. It prevents the retransmission of datagram which is also known as **Replay attack.**
- A monotonically increasing counter value.

Authentication Data

- This is variable length field whose length depends upon encryption algorithm used. Authentication data field of AH protocol is the output of hashing algorithm or message digest algorithm.
- AH protocol performs the integrity check value (ICV) on packet header or MAC is computed over the complete IP packet including the outer IP header to ensure that the data has not been changed during transmission process.
- As mentioned earlier AH doesn't encrypt the data the reason it doesn't provide confidentiality during transmission.

Modes of Operation

AH can work in two modes :

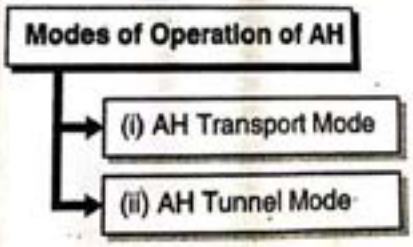


Fig. 12.2.3 : Modes of Operation of AH

(i) AH Transport Mode

In Transport mode the authentication header is placed between original IP Header and original TCP header as shown in Fig. 12.2.4.

Scanned by CamScanner

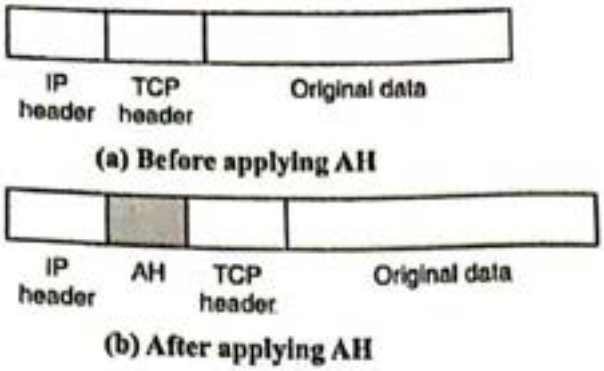


Fig. 12.2.4 : AH transport mode

➜ **(ii) AH Tunnel Mode**

- In Tunnel Mode the AH is inserted between the new IP header and original IP header.
- The inner IP address contain source and destination address of sender and receiver and the out IP address contain the address of security gateway or firewall as shown in Fig. 12.2.5.

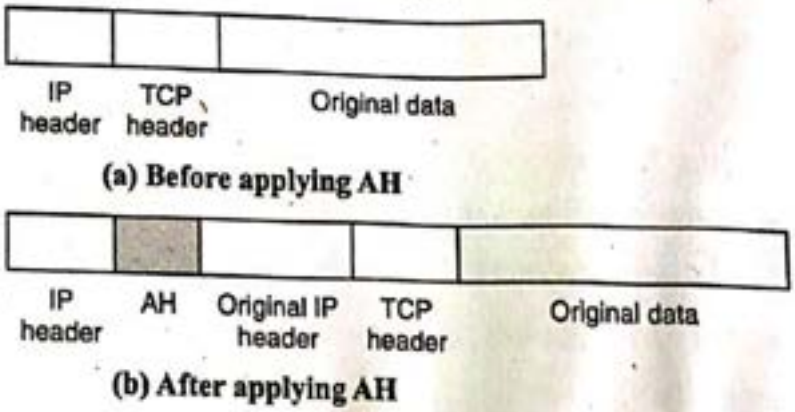


Fig. 12.2.5 : AH tunnel mode

12.2.2 Encapsulating Security Payload

- One of the most important feature that Authentication Header was unable to provide called data confidentiality (if attacker able to access the contents of a message).
- An Encapsulating Security Payload is primarily designed to provide encryption, authentication and confidentiality for the data or payload that is being transferred in an IP network
- As defined earlier ESP is used to encrypt the entire payload of an IPSec packet the reason ESP alone can provide data authentication, protection against replay attacks and data integrity by adding ESP header, ESP trailer and MAC to the packet.

Scanned by CamScanner

- ESP has the same fields as defined in AH, but it integrates these fields in a different way instead of having just a header, it divides these fields into three components: An ESP header, ESP trailer and ESP authentication block as shown in Fig. 12.2.6.
- It is designed for confidentiality and integrity of messages. ESP can be used alone or with combination of AH.ESP adds a header and a trailer to the payload. Following are the steps for adding ESP header and trailer.

Step 1 : In the initial step, ESP trailer is added to IP payload.

Step 2 : Payload and trailer or encrypted

Step 3 : After the encryption ESP header is added to the encrypted packet.

Step 4 : ESP header, payload and ESP trailer are used to create authenticated data.

Step 5 : This authentication data is added at the End of Trailer.

Step 6 : Lastly the IP header is added.

- The main functionality of ESP is to provide the confidentiality to IP packet by encrypting them. Encryption algorithms (Triple DES, Blowfish, and IDEA etc.) used to combines the data in the packet with a key and transform it into an encrypted form. The encrypted packet now then transmitted to the destination, and decrypts it using the same algorithm.
- The detail description of Encapsulating Security Payload (ESP) fields is given below :

ESP Header

This contains two fields, Security Parameter Index (SPI) of 32 bits and Sequence Number of 32 bits, as defined in AH protocol SPI is a 32-bit field used in combination with source IP address, destination IP address and ESP security protocol to identify a security association (SA) for the traffic to which IP packet belongs.

Sequence number

It is also a 32 bit field. It prevents the retransmission of data gram which is also known as **Replay attack as defined earlier.** This field is not encrypted but it's authenticated to perform anti-replay checking before decryption.

Encrypted data

This is variable length filed contains transport layer segment or IP packet which is protected by performing ESP encryption.

Scanned by CamScanner

ESP Trailer

ESP trailer field contains padding (0-255 bytes), pad length 8-bits and next header 8 - bits.

Padding (0-255 bytes)

- Padding filed used to expand plain text message to required size or to align the encrypted data by adding padding bits to the actual data which provides confidentiality to traffic flow.
- If an encryption algorithm requires the plaintext to be a multiple of some number of bytes (e.g., the multiple of a single block for a block cipher), the Padding field is used to expand the plaintext (consisting of the Payload Data, Padding, Pad Length, and Next Header fields) to the required length.

Pad Length (8 bits)

- This is mandatory field in ESP protocol which used to indicate the number of pad (protection) bytes added into the packet.
- Indicates the number of pad bytes immediately preceding this field.

Next Header (8 bits)

- The same bit length as of pad length used to identifies the type of encrypted data in the Payload Data field.
- Identifies the type of data contained in the Payload Data field (an upper-layer protocol - TCP, UDP, or an IPv6 extension header).

ESP Authentication Data

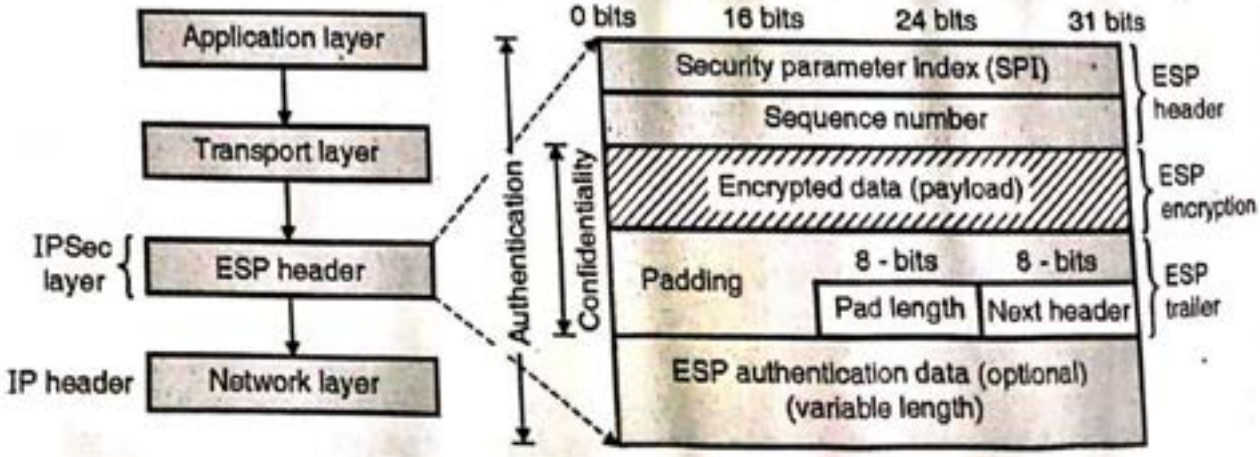


Fig. 12.2.6 : ESP header, trailer and encryption

Scanned by CamScanner

- This is variable length field whose length depends upon encryption algorithm used.
- A variable-length field (must be an integral number of 32-bit words) that contains the Integrity Check Value computed over the ESP packet minus the Authentication Data field.
- As mentioned earlier ESP encrypts the data the reason it provide data confidentiality during transmission.

Modes of Operation

ESP can work in both modes namely :

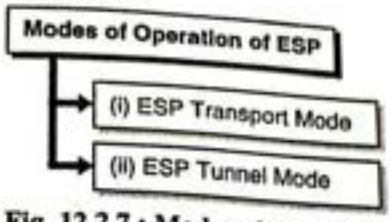
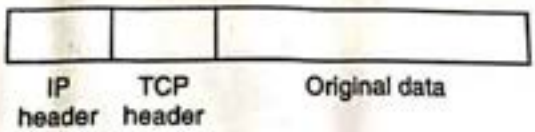


Fig. 12.2.7 : Modes of Operation

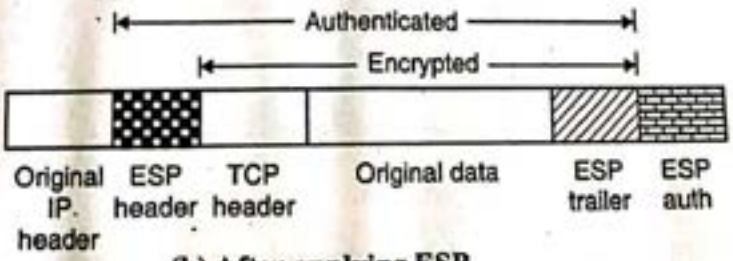
(i) ESP Transport mode

In this case ESP header is added before the transport layer header (like TCP,UDP) and trailer is added after the IP Packet whereas if authentication is required then authentication data is added after the ESP trailer.

Fig. 12.2.8 shows transport mode in ESP.



(a) Before applying ESP



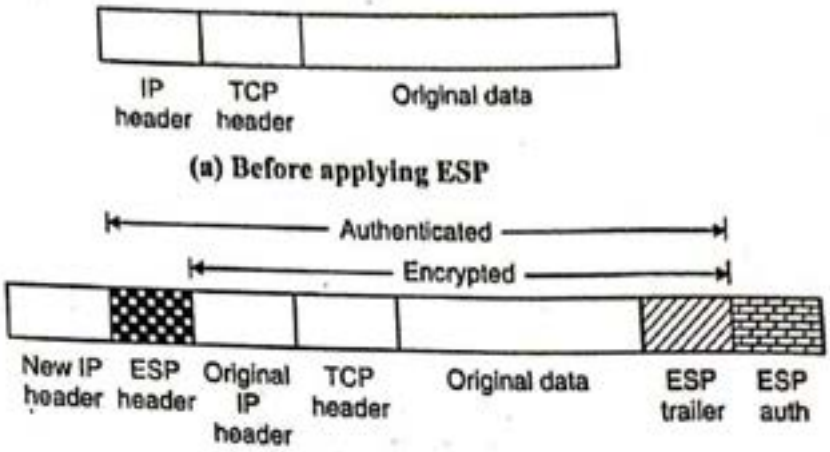
(b) After applying ESP

Fig. 12.2.8 : Transport mode in ESP

(ii) ESP tunnel mode

- In this case ESP header is added before original IP header and ESP trailer after the original data.

Scanned by CamScanner



(a) Before applying ESP

(b) After applying ESP

Fig. 12.2.9 : Tunnel mode in ESP

- It is an important aspect of IPSec, Security Association (SA) is a contract between the communication parties about factors like IPSec protocol version, mode of operation (tunnel or transport), cryptographic algorithm, key etc. Security Association creates a secure channel between two communicating parties.
- If both AH and ESP are used SA for actual operation then they will need two sets of SA one for AH and one for ESP.
- For communication each party needs two set of SA one for incoming transmission and one for outgoing transmission because SA is simplex unidirectional.
- The whole packet is encrypted. As whole packet is encrypted so New IP header will be added, which will contain information for routing from on network to another, so that Security Association

12.2.3 Security Association Database

→ (MU - Dec. 17)

Q. 12.2.2 What are security associations ? (Ref. sec.12.2.3) **Dec. 17, 4 Marks**

- Security Association can be very complex.
- Each participating parties need to have inbound and outbound SAs to allow bidirectional communication. It is a two directional table.
- Each row in table defines Security Association which is collectively called as Security Association Database. Each requires party requires maintaining its own database.
- For one way communication (called unidirectional) single SA is required whereas for two way communication (bidirectional) two security association are required.

Scanned by CamScanner

- SA uses different parameters to perform data handling between sender and receiver like security parameter index (SPI).
- IP address of the host (usually destination IP address of end user); encryption algorithms; protocol format (AH or ESP); and security protocol identifier (SPI). Almost all these parameters we have discussed in pervious bit still let us have small look out on these parameters.
  - **Security Parameter Index (SPI) :** A 32-bit number used to uniquely identify a particular security association between any connected devices. The SPI is placed in AH or ESP packet for linking the each secure packet to the security association.
  - **Destination IP Address : Destination** IP address of a host, router or firewall who involved in communication or the address of devices for which security associations are established.
  - **Security Protocol Identifier (SPI) :** To identify which protocol (AH or ESP) is used for security associations. If both are used then they have separate security associations

Syllabus Topic : Firewalls

12.3 Firewall Introduction

→ **(MU - May 16, Dec. 16)**

Q. 12.3.1	What is a firewall ? What are the firewall design principles ? (Ref sec. 12.3)	May 16, 5 Marks
Q. 12.3.2	What are firewals ? (Ref sec. 12.3)	Dec. 16, 3 Marks

- Firewall is called as barrier place between inside and outside network to protect organization from inside and outside hackers. It also filters all traffic between intranet and extranet which runs through it.
- The main purpose of the firewall is to keep attackers outside the protected environment. For that policies are set in the firewall to decide what is allowed and what is not allowed.
- Moreover we can decide the allowed places, allowed users, allowed sites, can provide different access rights to different category of the users.
- **Example :** Cyber am through which only educational sites are allowed through college internet and non-educational sites like facebook, twitter can be blocked using firewall.

Scanned by CamScanner

12.3.1 Firewall Characteristics

Q. 12.3.3 What are the various characteristics of firewall ? (Ref sec. 12.3.1)

- Following lists the characteristics as well as design goals for a firewall :
  1. All inside and outside traffic must pass through the firewall. This is possible only because of physically blocking of all access to the local network except via the firewall.
  2. The traffic defined by the local security policy will only allowed to pass through the network. Different types of firewall are used to define the policies as per the norms decided.
  3. The firewall itself is immune to penetration. Different techniques are used to control access and enforce the site's security policy.
- **Service control :** This policy helps to determine which type of internet services that can be accessed inbound and outbound. Firewall can filter traffic on the basis of IP address and TCP port number. It also act as proxy server that receives and interprets each service request before passing it on.
- **Direction control :** Direction control determines the direction in which particular service requests may be initiated and allowed to flow through the firewall.
- **User control :** This technique is used to controls access to a service according to which user is attempting to access it.
- **Behaviour control :** Controls how particular services are used. For example, the firewall may filter e-mail to eliminate spam.

12.3.2 Limitations of Firewalls

Q. 12.3.4 What are the disadvantages of firewalls ? (Ref sec. 12.3.2)

A firewall may be a pivotal component of securing your organize and is planned to address the issues of information integrity or activity verification (through stateful packet inspection) and secrecy of your inner network (through NAT). Your network picks up these benefits from a firewall by accepting all transmitted activity through the firewall. Your network picks up these benefits from a firewall by receiving all transmitted activity through the firewall.

Scanned by CamScanner

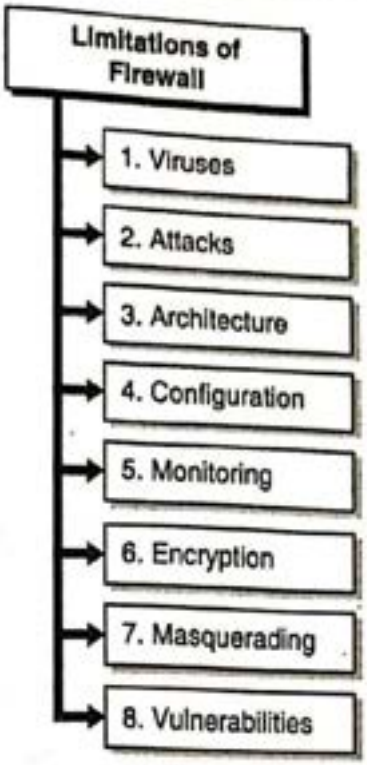


Fig. 12.3.1 : Limitations of Firewall

Following are the limitations of firewall :

1. Viruses

Not all firewalls have full protection against computer viruses because virus uses different encoding techniques to encode files and transfer them over Internet.

2. Attacks

A firewall cannot prevent users or attackers with modems from entering in to or out of the internal network, thus bypassing the firewall and its protection completely.

3. Architecture

Firewall architecture depends upon single security mechanism failure. If that security mechanism has a single point of failure, affects on entire firewall programs which opens the loop falls for intruders.

4. Configuration

Firewall doesn't have mechanism to tell administrator about incorrect configuration. Only trained professionals in the field of network security can configure firewall properly.

Scanned by CamScanner

→ **5. Monitoring**

Firewall doesn't give notification about hacking. It will notify only about threat occurrences. The reason is, organization demands additional hardware, software and different networking tools as per there requirement hence there is no control on it.

→ **6. Encryption**

Firewall and Virtual Private Networks (VPNs) don't encrypt confidential documents and E-mail messages sent within the organization or to outsiders. Dignified procedures and tools are needed to provide protection against confidential documents.

→ **7. Masquerading**

Firewalls can't stop hacker those who steal login id and password of authentic user to gain access to a secure network. Once attacker gains full access of the entire network, attacker can delete or change the network policies of organization.

→ **8. Vulnerabilities**

Firewall can't tell other venerability that might allow a hacker access to your internal network.

12.3.3 Firewall Architecture and Types

→ **(MU - Dec. 16, May 17)**

Q. 12.3.5 Explain the different types of firewalls and mention the layer in which they operate. (Ref sec. 12.3.3) **Dec. 16, 7 Marks**

Q. 12.3.6 What are the types of firewalls ? (Ref sec. 12.3.3) **May 17, 5 Marks**

- A firewall is a kind of reference monitor. All network traffic passes through firewall. That's why it is always in invoked condition.
- A firewall is kept isolated and cannot be modified by anybody other than administrator.
- Generally it is implemented on a separate computer through which intranet and extranets are connected.

Scanned by CamScanner

Following are the common architectural implementations of firewalls :

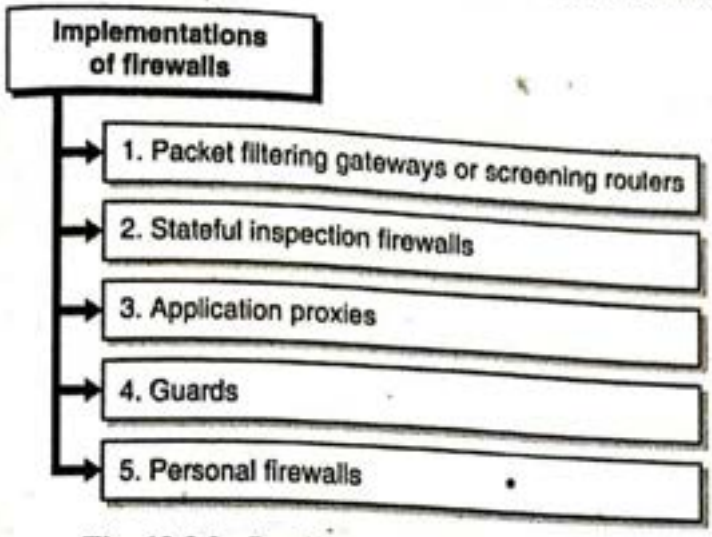


Fig. 12.3.2 : Implementations of firewalls

1. Packet Filtering Gateway

- It is the most simple and easy to implement firewall.
- Packet filtering is done on the basis of packets source or destination address or based on some protocol type like HTTP or HTTPs.
- If the firewall is placed just behind the router then the traffic can be analyzed easily.
- In the Fig. 12.3.3 it is shown that how packet filtering gateway can block traffic from network 1 and allow traffic from network 2.
- Also the traffic using telnet protocol is blocked. Packet filters do not analyze the contents of the packet rather they just check IP address of the packets as shown in Fig. 12.3.3.
- The biggest disadvantage of the packet filtering gateway is that it requires lot of detailing to set policies.

Example

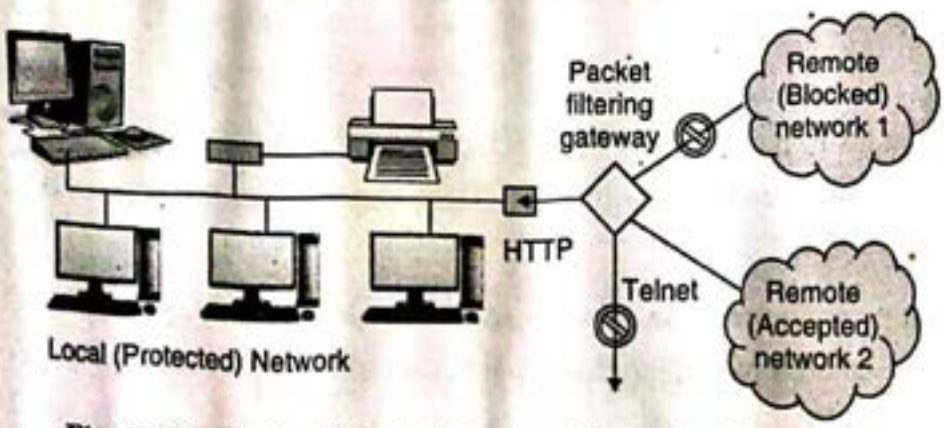


Fig. 12.3.3 : Packet Filter Blocking Addresses and Protocols

Scanned by CamScanner

If port 80 is blocked. If some applications essentially need use of port 80 then in this case we have to provide all the details of those applications for which port 80 is needed.

2. Stateful Inspection Firewall

- Packet filtering is done one packet at time. Sometimes attacker may use this technique for their attack. Attacker can split the script of attack into different packets so that the complete script of attack cannot be identified by packet filtering firewall.
- To avoid this stateful inspection firewall keeps record of states of the packets from one packet to another. Thus sequence of packets and conditions within the packets can be identified easily.

3. Application Proxy

- Packet filters cannot see inside the packets. From the packet headers they just get IP addresses for filtering.
- Application proxy is also known as a bastion host. Fig. 12.3.4 shows firewall proxies.

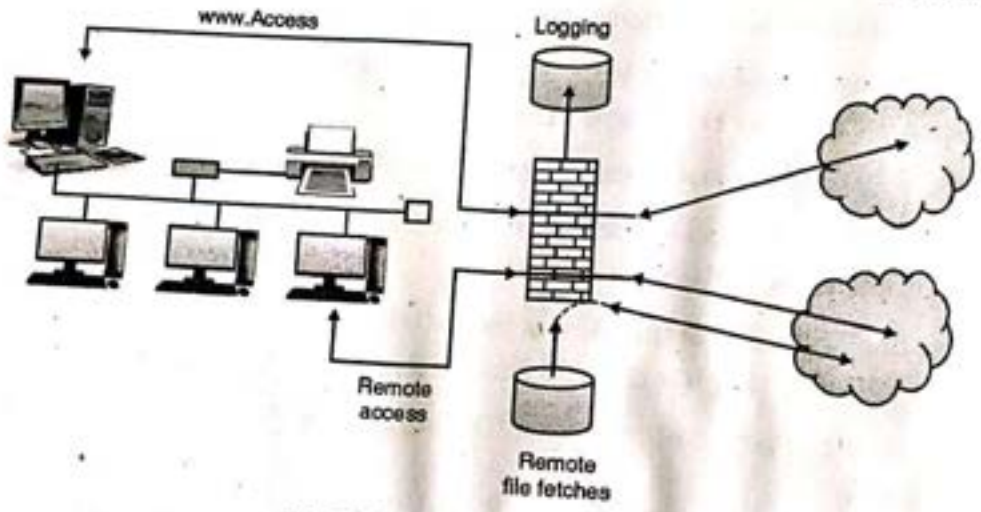


Fig. 12.3.4 : Firewall Proxies

Example

- A college wants to publish a list of selected students. Then they just want students to read that list. No student can change that list. Moreover students cannot access more data than the list.
- Application proxy helps us in this regard. Here it helps us to check only list is displayed on the screen and not more than that. That list should not have any modified contents.
- Proxies on the firewall can be customized as per the requirements.

Scanned by CamScanner

4. Guard

- A guard is kind of complex firewall. It works similar to proxy firewall. Only difference is that guard can decide what to do on behalf of the user by using available knowledge.
- It can use knowledge of outside users identity, can refer previous interactions, blocked list etc.

Example

- In order to increase the speed of the internet a school can set download limit for the students.
- A student can download only 20mb data per day etc.

5. Personal Firewalls

Q. 12.3.7 What is personal firewalls? (Ref sec. 12.3.3(5))

- For a personal use to keep separate firewall on a separate machine is quite difficult and costly. So personal users need a firewall capability at lower cost.
- An application program which can have capabilities of a firewall can solve this problem.
- It can screen incoming and outgoing traffic on a single host.
- Symantec, McAfee, Zone alarm are the examples of personal firewalls. Personal firewalls can be combined with antivirus systems.

12.3.4 Firewall Configurations

Q. 12.3.8 How firewalls are configured and managed? (Ref sec. 12.3.4)

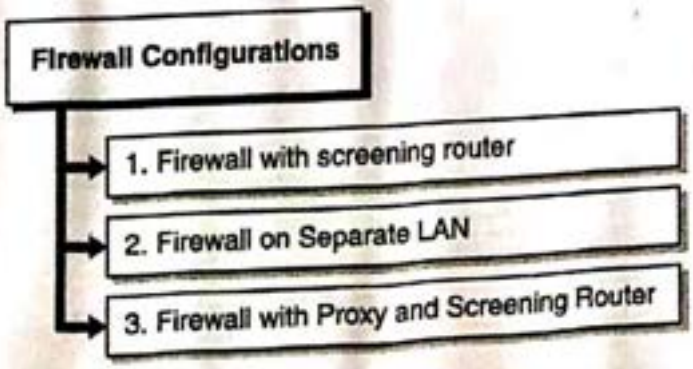


Fig. 12.3.5 : Firewall Configurations

Scanned by CamScanner

1. Firewall with screening router

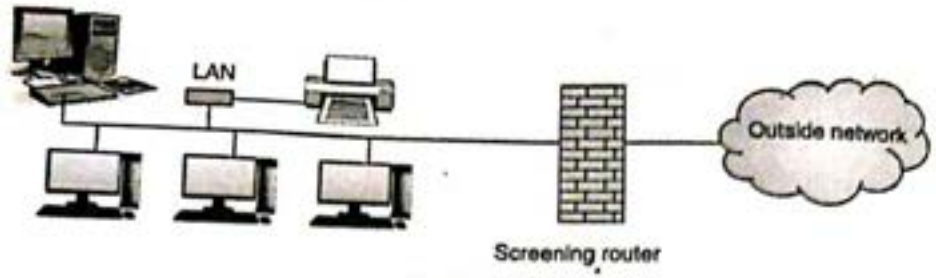


Fig. 12.3.6

The screening router is placed in between intranet and extranet. Another name for screening router firewall is network level or packet-filter firewall. Protocol attributes are used for performing the screening of incoming packets.

The attributes like source or destination address, type of protocol, source or destination port, or some other protocol-specific attributes plays a vital role. A screening router performs packet-filtering and is utilized as a firewall. In a few cases a screening router may be utilized as perimeter assurance for the internal network or as the whole firewall solution.

2. Firewall on Separate LAN

Unauthorized internet users from accessing private networks connected to the internet are prevented by firewall, especially intranets. All messages entering or leaving the intranet (i.e., the local network to which you are connected) must pass through the firewall, which examines each message and blocks those that do not meet the specified security constraint.

To overcome the problem of the exposure of LAN, a proxy firewall can be installed on its own LAN.

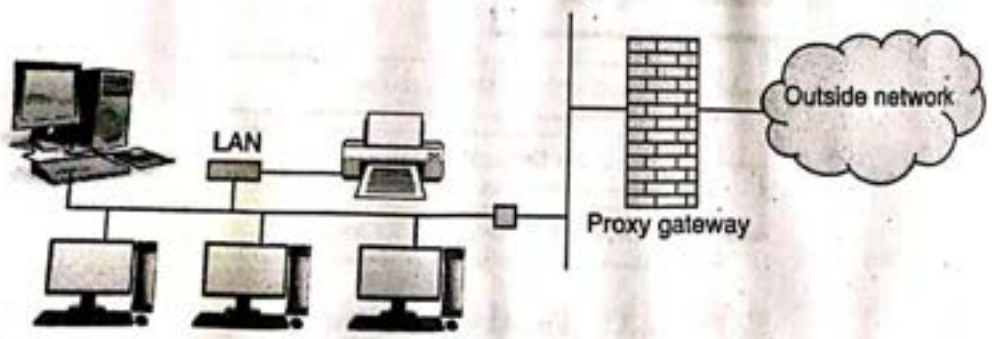


Fig. 12.3.7

Scanned by CamScanner

3. Firewall with Proxy and Screening Router

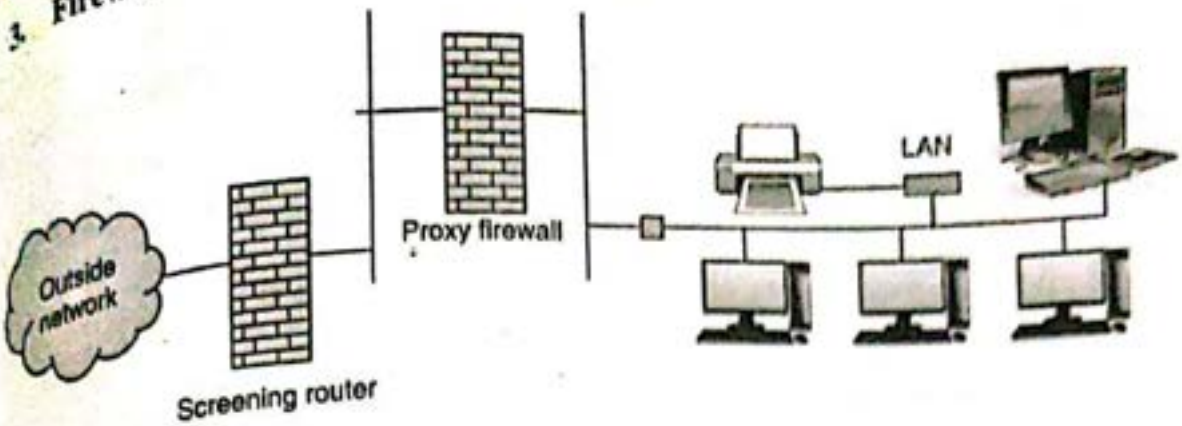


Fig. 12.3.8

If screening router is installed behind the proxy firewall, then it ensures the correct address to proxy firewall. In other words it is a double guard protection. If anyone fails LAN is not exposed.

12.4 Introduction to Intrusion Detection

- With the rapid expansion of Internet during recent years, security has becomes an essential issue for computer networks and computer systems.
- As defined earlier the main aim of a security system is to protect the most valuable assets (data/secret information) of an organizations like banks, companies, universities and many others, because these organizations have data or secret information in some form, and their security policies are keen for protecting the privacy, integrity, and availability of these valuable information or data.
- As these organizations will have different security policies and requirements depending on their vision and missions.
- Many efforts have been carried out to accomplish this task are security policies, firewalls, anti-virus software even *Intrusion Detection Systems* (IDSs) to configure different services in operating systems and computer networks.
- But still detecting different attacks (like denial service attacks, IP spoofing, ping of death, network scanning etc.) against computer networks is becoming a crucial problem to solve in the field of cryptography and network security.

Scanned by CamScanner

- To overcome all above problems researcher in the field of computer security came with existing but different solution called Intrusion Detection System (IDS). Before discussing on IDS let us understand some key points like what is intrusion? What is intrusion detection and then what is intrusion detection system?

12.4.1 Intrusion Detection

Q. 12.4.1 What are the strengths and limitations of Intrusion Detection System? (Ref. sec. 12.4.1)

Q. 12.4.2 What is intruder and intrusion detection system? (Ref. sec. 12.4.1)

Before defining Intrusion Detection first understand what is an Intruder?

- An *Intruder* is a person who intercepts system availability, confidentiality and data integrity. Intruder's gains unauthorized access to a system with criminal intensions. Intruder may damage that system or disturbs data.
- When an attacker or intruder attempts to break into an information system or performs an illegal action such as denial of service attacks, scanning a networks, ping scan, sending many request for connection setup using fake IP address, etc. which is legally not allowed, that is called as an *intrusion*.
- Intrusion detection is an important technology that monitors network traffic, events and identifies network intrusions such as abnormal network behaviours, unauthorized network access and malicious attacks to computer systems.
- The general example of intrusion detection is when we suffer from some disease and asking doctor what happen to me. Doctor suggests for blood checking and sends blood sample to laboratory for detection.
- The blood report given by pathologies is just detection of disease (number of platelets count, WBC, RBC, hemoglobin, etc.) then after checking the entire history of blood report doctor suggests medicine to cure the disease.
- Here blood report is intrusion detection where as medicine given by the doctor after checking blood report is called intrusion detection system. Finally how fast patient get relief depends upon the doctor's education, experience and knowledge, joke apart let us move towards technical definition of IDS.

Scanned by CamScanner

Syllabus Topic : IDS and Types

12.4.2 Intrusion Detection System : Need, Methods, Types of IDS

→ (MU - May 16)

Q. 12.4.3 Explain the significance of an Intrusion Detection System for securing a network. (Ref. sec. 12.4.2) **May 16, 6 Marks**

Q. 12.4.4 What is IDS? Differentiate statistical Anomaly detection and rule base intrusion detection. (Ref. sec. 12.4.2)

Q. 12.4.5 What is intrusion detection system? Enlist and explain different types of IDS. (Ref. sec. 12.4.2)

Q. 12.4.6 What are the challenges of intrusion detection? (Ref. sec. 12.4.2)

- Intrusion Detection system has some policies or mechanisms to protect computer systems from many attacks. As the use of data transmission and receiving over the internet increases the need to protect the data of these connected systems also increases. Many scientists have different definition of IDS but as per our point of view IDS can be defined as below point.
- "An Intrusion Detection System is software that monitors the events occur in a computer systems or networks, analyzing what happens during an execution and tries to find out indications that the computer has been misused in order to achieve confidentiality, integrity and availability of a resource or data".
- The IDS will continuously run on our system in the background, and only generate the alert when it detects something suspicious as per its own rules and regulation or attack signature present into it and taking some immediate action to prevent damage.

Intrusion detection

System examines or monitors system or network activity to find possible attacks on the system or network. Signs of violation of system security policies, standard security practices are analyzed.

Intrusion Prevention is the process of detecting intruders and preventing them from intrusive effort to system.

Challenges of Intrusion Detection

In order to better understand intrusion detection systems, it is important to realize that threats to networked computer systems come in a number of forms. According to the source of

Scanned by CamScanner

threats, potential intruders can be roughly classified into two categories :

1. **Outside Intruders :** The attack is launched by an unauthorized computer user. The attacker will stole or broken passwords, using system vulnerabilities or improper configurations, human engineering techniques, to gain access to computers.

2. **Inside Intruders :** Internal intruders, who have permission to access the system with some restrictions. In this case, the intruder already has legitimate access to a computer system, but utilizes any of the previously mentioned techniques to gain additional privileges and misuse the computer system. Sometimes inside intruders are more harmful than outside intruders. It is observed that 80% of intrusions and attacks come from within organizations.

Following are the possible type of attacks that intrusion detection needs to face :

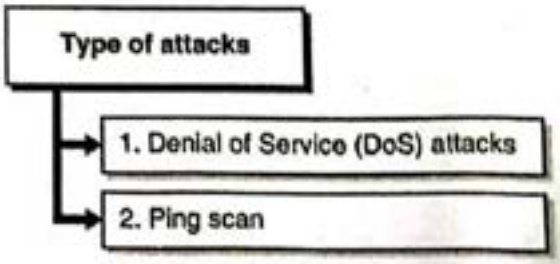


Fig. 12.4.1 : Type of attacks

➜ 1. Denial of Service (DoS) attacks

- These attacks attempt to *"shut down a network, computer, or process; or otherwise deny the use of resources or services to authorized users"*.
- There are two types of DoS attacks:
  - (i) Operating system attacks, which target bugs in specific operating systems and can be fixed with patches;
  - (ii) Networking attacks, which exploit inherent limitations of networking protocols and infrastructures.
- An example of operating system attack is teardrop, in which an attacker exploits a vulnerability of the TCP/IP fragmentation re-assembly code that do not properly handle overlapping IP fragments by sending a series of overlapping packets that are fragmented.
- Typical example of networking DoS attack is a "SYN flood" attack, which takes advantage of three-way handshake for establishing a connection. In this attack, attacker establishes a large number of "half-open" connections using IP spoofing. The attacker first

Scanned by CamScanner

sends SYN packets with the spoofed (faked) IP address to the victim in order to establish a connection.

- The victim creates a record in a data structure and responds with SYN/ACK message to the spoofed IP address, but it never receives the final acknowledgment message ACK for establishing the connection, since the spoofed IP addresses are unreachable or unable to respond to the SYN/ACK messages.
- Although the record from the data structure is freed after a time out period, the attacker attempts to generate sufficiently large number of "half-open" connections to overflow the data structure that may lead to a segmentation fault or locking up the computer.

2. Ping scan

The simplest form of scan, an attacker sends an ICMP echo request packet to every candidate machine (which is the same way the *ping* tool works).

Any addresses that respond are noted as active.

(1) **TCP Connect ( ) scan :** Another simple scan, an attacker attempts to open a standard TCP connection to a typical port on the candidate machine (such as the HTTP port 80). Any machine where such a connection succeeds is noted as active. Since many systems log any connection attempts, this type of scan is relatively easy to recognize from standard audit data.

(2) **UDP scans :** This scan consists of sending UDP packets to likely ports on candidate machines at worst, scanning for any open UDP ports. Since UDP is connectionless, such attempts are harder to control using filtering firewalls, and may be capable of finding unprotected services and hosts. Many variations on these scanning techniques exists - including scans using fragmented packets, and scans spread across a long period or a number of source machines. In practice, completely blocking scans is probably infeasible - but may give an administrator early warning of an impending attack.

(3) **Rlogin :** The RLOGIN attack is characterized by a high rate of connections from one node to another, often within a small period of time. In this attack, the intruder is attempting to gain access to the system.

Need of IDS

- Intrusion Detection has its primary goal the detection of abuses of computer systems also it performs a variety of functions like :

1. Monitoring and analyzing user and system activity.

Scanned by CamScanner

2. Auditing system configurations and vulnerabilities.
3. Assessing the integrity of critical system and data files.
4. Recognition of activity patterns reflecting known attacks.
5. Statistical analysis for abnormal activity patterns.
6. Operating-system audit-trail management, with recognition of user activity reflecting policy violations.

- IDS should offer reports of attacks in real time, ideally as the intrusion is in progress allowing security personnel to take corrective action.
- IDS should cooperate with other security mechanisms, increasing the overall security of systems. Ideally, IDS should be capable of detecting failures or attacks on other security mechanisms, forming a second level of defence.
- IDS should be capable of responding to intrusive behaviour: by increasing its monitoring in the relevant sections, or by excluding or restricting intrusive behaviour.
- IDS should protect itself against attacks, ensuring that the integrity of the greater system, and audit information up to the point of compromise remains intact, and ensuring that a compromised or hostile component cannot adversely affect the functioning of the system as a whole.
- Other than monitoring network intruder and policy violations, the IDS can be useful in many other ways :
  - To identify problem based on security policies.
  - To maintain the logs of all the threat those are detected by IDS.
- As users are monitored continuously in network, making them analyze so that less violations cannot be committed.
- Using some preventive measures so that violation cannot be occur like terminating the network connections, user session or block access to the targets or the accounts that are likely to be violated.
- The IDPS (Intrusion Detection and Prevention System) can acts like proxy, which helps in un-packaging the payload of the request and remove header. This helps to invalidate the intruder attacks.
- The IDPS can sometimes change the security environment to prevent it from attacks.

Scanned by CamScanner

12.4.3 Intrusion Detection Methods/ Techniques

Q. 12.4.7 Explain methods for intrusion detection system (IDS). (Ref. sec. 12.4.3)

The categorization of Detection methodologies are : Signature Based, anomaly based, stateful protocol analysis. Most of the IDPS uses these techniques to reduce or make network error free.

12.4.3(A) Signature Based Detection

- It is a process of comparing the signatures of known threat with the events that are been observed. Here the current packet is been matched with log entry of the signatures in the network.
- Signature is defined as the pattern (structure) that we search inside a data packet. The data packet may contain source address, destination address, protocol, port number etc.
- If an attacker adds any malicious code into these data packet he is generating attack pattern or signature.
- Signature based IDS create databases of such attack pattern for detecting the known or documented attacks. Single signature is used to detect one or more types of attacks which are present in different parts of a data packet.
- Signature based IDS used to monitor the events occurred in the network and match those events against a database of attack signatures to detect intrusions.
- It also uses a rule set to identify intrusions by watching for patterns of events specific to known and documented attacks.
- For example, we may get signatures in the IP header, transport layer header (TCP or UDP header) and application layer header or payload.
- Signature based intrusion detection system sometimes also called misuse detection techniques. It checks for the attack pattern with the existing stored database pattern and if match is found then generates the alert.
- Signature based IDSs are unable to detect unknown and newly generated attacks because it requires manual updating of each new type of attacks into to the existing database.
- The most well known example of signature - based IDS is SNORT IDS freely available for attack detection and study purpose.

Scanned by CamScanner

Advantages

- An advantage of misuse-detection IDS is that it is not only useful to detect intrusions, but it will also detect intrusion attempts.
- Effective at detecting known attack without too many false alerts as compare to anomaly detection technique.
- Most of the current network intrusion detection system uses misuse detection technique for finding the attack pattern and detect them according to the rules and regulation used.
- Furthermore, the misuse detection IDS could detect port - scans and other events that possibly precede an intrusion.

Disadvantages

- Detecting only known attacks therefore it cannot identify new attacks efficiently.
- If there is single variation into attack signature it invalidates the attack signature or unable to detect it.
- Constant updating of attack pattern is required.

12.4.3(B) Anomaly Based Detection

Q. 12.4.8 Explain Anomaly-based Instruction Detection System. (Ref sec. 12.4.3(B))

- It is the process of comparing activities which are supposed to be normal against observed events to identify deviation.
- An IDPS uses Anomaly based detection techniques, which has profiles that represent normal activities of user, host, connections or applications.

For example

- Web activities are a normal activity done in a network. Anomaly based IDS works on the notation that "attack behavior" enough differ from "normal behavior" (IDS developer may define normal behavior).
- Normal or acceptable behaviors of the system (e.g. CPU usage, job execution time etc.) if the system behavior looks abnormal i.e. increasing CPU speed, too many job execution at a time then it is assumed that the systems is out of normal activity. Anomaly based detection is based on the abnormal behavior of a host or network.
- Database for such type of IDS is the events generated by user, host and network, and the "normal" behavior of the systems. These events (historical data) are collected from the

Scanned by CamScanner

research laboratories which continuously work on normal and abnormal behavior systems over a period of time.

- Anomaly based IDS checks ongoing traffic, host activities, transactions and behavior in order to identify intrusions by detecting anomalies. Host - based IDS generally uses anomaly based techniques.
- This can be done in two ways :

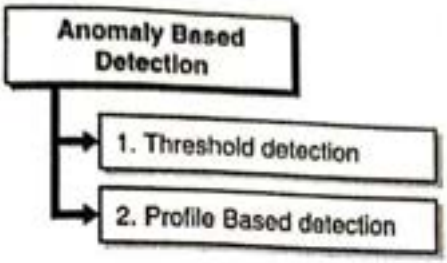


Fig. 12.4.2 : Anomaly based detection

1. Threshold detection

Threshold is defined for all users for all groups and frequency of all events is measured comparing with threshold.

2. Profile Based detection

Profiles of individuals are created and they are matched against the collected statistics for checking the irregular patterns.

Advantages

An anomaly detection system observes and checks the deviation of normal network. If it observes any changes or suspicious in the network from normal deviations it will immediately inform and alert about the unknown attack.

Disadvantages

- Anomaly detection techniques generate large number of false alarms due to the unpredictable behaviors of users and networks.
- It also requires extensive "training data set" of system events, records in order to characterize normal behavior patterns.
- In addition, because a user's normal behavior usually changes over time (for example, a user's behavior may change when he moves from one host to another host), it is very difficult to collect the historical data of normal and abnormal behavior.

Scanned by CamScanner

12.4.3(C) Stateful Protocol Analysis

Unlike anomaly based detection which uses host and network specific profiles., the stateful protocol analysis relies on Vendor developed universal profiles. The stateful protocol analysis means the IDPS is able of checking the network, applications, and protocols that are pre defined in them. It can identify unexpected sequence of threats in form of commands.

Disadvantage of stateful protocol analysis

- Stateful protocol analysis are extensively resource demanding.
- These methods don't capture threats or attacks that don't hamper the general accepted protocol in network.

12.4.4 Types of IDS

Q. 12.4.9 Explain types of Intrusion detection systems (IDS). (Ref. sec. 12.4.4)
Q. 12.4.10 Describe the different types of IDS and their limitations. (Ref. sec. 12.4.4)

The types of IDS are differentiated mainly by the types of event they monitor or scrutinize.

There are four types of IDS.

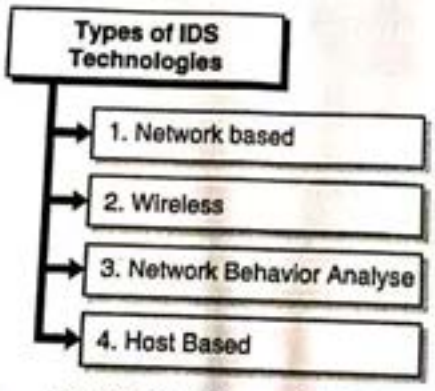


Fig. 12.4.3 : Types of IDS

1. Network based

The IDS monitors network traffic. It analyzes the network activities and protocol activities to identify suspicious activity of the network.

2. Wireless

The IDS monitors the wireless network traffic. It analyzes the network activities and protocol activities of wireless network.

Scanned by CamScanner

3. Network Behaviour Analyse

These network behavior analyze identify the treats that create unusual traffic overflow, DDOS(Distributed Denial of Service) attacks, malwares, and policy violations.

4. Host Based

- These IDS monitors the host and the event occurs within that host.
- Among above four types of IDS two are important and most commonly used to monitor the networks and hosts.

12.4.4(A) Network based IDS (NIDS)

- As the usage and popularity of Internet is increasing tremendously, the attacks to the network are increasing for example TCP hijacking, DOS, IP Spoofing etc.

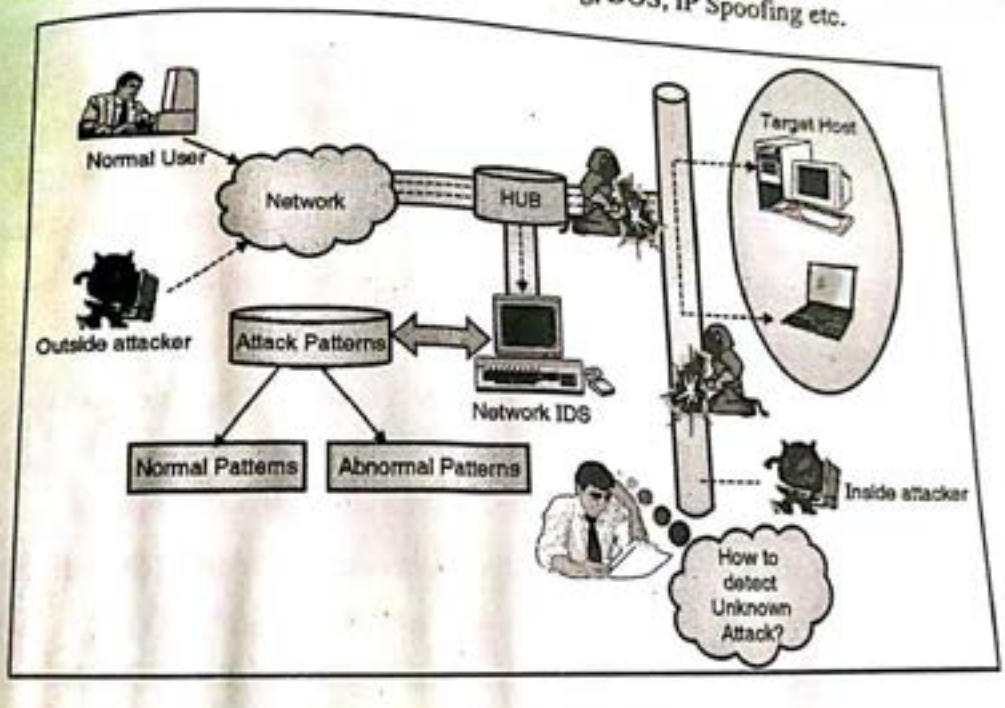


Fig. 12.4.4 : NIDS architecture

- These network attacks cannot be detected by host based IDS It need Network based IDS to detect the attack and resolve it. General architecture of NIDS is shown in Fig. 12.4.4.
- NIDS detects attacks by monitoring, capturing and analyzing packets or network traffic and tries to give indication that computer has been misused. It detects malicious data present into packets by monitoring network traffic.

Scanned by CamScanner

- NIDS continually monitors network traffic and discovers that if hacker/ intruder are attempting to break into a system.
- When NIDS installed on main server which consist of multiple hosts in a single network, it detects attacks present in the multiple hosts by checking incoming packets that looks unordinary.
- NIDS uses raw network packets as the training dataset for offline detection collected from well known research laboratory such as Defence Advance Research Project Agency (DARPA).
- As defined earlier it can be installed on servers, workstations, personal computers or machines dedicated to monitor incoming network packets from switches, routers and probes for intrusions.

Advantages of NIDS

- A well placed network - Based IDS can monitor a large network.
- NIDS just listen to the network; it does not interfere in the network.
- NIDS can be made very secure against attack and made invisible to many attackers.
- Network-based IDS use live network traffic for real time attack detection and also operating system independent.

Disadvantages of NIDS

- It becomes difficult for NIDS to recognize the attack in large or busy network due to high traffic is there in network. It will be difficult for NIDS to analyze.
- NIDS cannot analyze the network if communication is in encrypted format.
- Difficult to detect the whole process of attack, usually detect only the initial level of attack.
- We have seen a different type of IDS but we must know how these IDS detect whether given packet is malicious and the system behaviour is abnormal. There are two main types of detection techniques for analyzing events generation, system logs, audit trails, and malicious packet activities namely : anomaly detection and misuse detection also called signature based IDS.
- (NIDS) usually consists of a network sensor with a Network Interface Card (NIC) or LAN card operating in casual mode. The IDS is placed along a network segment or boundary and it monitors all traffic on that network segment.

Scanned by CamScanner

12.4.4(B) Host Based IDS (HIDS)

HIDS usually collects information from the operating system audit trails, and system logs. (An audit trail is a series of records of computer events, about an operating system, an application, or user activities generated by an auditing system that monitors system activity). HIDS generally installed on individual host which is connected to the internet.

Features of HIDS

- HIDS focus monitoring and analyzing the computer system they are installed on.
- It continuously monitors the state of system. It check content of RAM and the file system to check that their content do not look suspicious.
- It generally looks for the real time malicious, suspicious activity of system log.

Advantages of HIDS

- As defined earlier Host-based IDS operate on OS audit trails; they can help detect Trojan horse or other attacks that creates the software integrity violation.
- HIDS analyze most of the encrypted network traffic, which usually encrypted or decrypted by the sender and/or receiver.
- It is able to monitor and detect attack, which is sometimes not possible for Network IDS.

Disadvantages of HIDS

- Host-based IDS are difficult to manage, because they generally installed on individual host. Monitoring to individual host is difficult because of different system configuration and log generation.
- When host-based IDS use operating system logs as an information source the amount of data can be increase, requiring additional local storage on the system.
- Host - based IDS are not suitable for detecting network denial of service and network scan attacks because it only checks only those packets received by individual host.

Syllabus Topic : Secure Email - PGP

12.5 Electronic Mail Security : Pretty Good Privacy

→ (MU - May 16)

Q.12.5.1 Write in brief about Email security. (Ref. sec. 12.5) May 16, 5 Marks

Scanned by CamScanner

- We all are aware that most popular use of Internet is to send the email and chatting with the friend's, partner etc. But have you ever think that if we are sending mail to intended person is going in his inbox only?
- Security concerns have estimated that only about one in every 100 messages is secured against interception and modification attacks. Are we aware that sending an email to business partner or friends in clear text message is going through thousands of machines (between sender and receiver before it reaches to intended recipients?) these machines might read and saved the contents of email for future use?
- Many people think that name given in sender of the mail identifies who actually sends it.
- When you send a message through email, we cannot guarantee that it will be deliver to correct destination or received exactly what you sent. And even there is a no way of knowing that the message is received read and forwarded by attacker.
- Because of wide spared problem of email modifications, sending it to wrong destination by intermediate parties, email spoofing, we need a competing solution to overcome and address the issues of authentication, integrity and reliability of the messages between sender and receiver.
- The public key cryptography play an important role because of two keys used, only intended sender can decrypt the message using his public key as message encrypted using private key of the sender.
- The solution is called as Pretty Good Privacy (PGP) program/ software which provide the secrecy and non-repudiation of data sent over Internet especially by email.
- Pretty Good Privacy (PGP) is a popular open-source freely available software package/ techniques used to encrypt and decrypt email messages over the Internet.
- PGP is an e-mail security program written by Phil Zimmermann in 1991, PGP program become a de facto standard for e-mail security used to store the encrypted files so that it can benon-readable by other users or intruders.
- This program also be used to send an encrypted digital signature, let the receiver verify the sender's identity and know that the message was not changed or modified while transmission.
- Once the file is encrypted using PGP program only the intended recipient can decrypt it. Once message content digitally singed by sender, the sender guarantee to the recipients that message or file comes from valid sender and not by attacker.

Scanned by CamScanner

- Digital signature functionality of PGP guarantees that the message or file come from the sender and not from an intruder.

12.5.1 Working of Pretty Good Privacy

Q. 12.5.2 How does PGP achieve confidentiality and authentication in emails? (Ref. sec. 12.5) → **(MU - Dec. 15)**

Dec. 15, 5 Marks

- As mentioned earlier PGP uses the concept of public key cryptography, if user encrypts the plain text message using PGP, it first compress the plain text message.
- BGP uses data compression technique to encrypt the plain text message which saves the transmission time and disk space and more important it strengthen the security.
- After data compression PGP generate the session key. Table 12.5.1 shows how PGP encrypt the message in order to achieve the confidentiality, integrity and non-repudiation.

Table 12.5.1 : Encryption and Decryption of Pretty Good Privacy

Sr. No.	Parameter	Algorithm	Description
1.	Digital signature	SHA or RSA	A hash code of a message is created using SHA-1. This message digest is encrypted using RSA with the sender's private key and added with the message.
2.	Message encryption	IDEA or Triple DES with Diffie-Hellman or RSA	A message is encrypted using IDEA or 3DES with a one-time session key generated by the sender. The session key is encrypted using Diffie-Hellman or RSA with the recipient's public key and Rdded with the message.
3.	Compression	ZIP	A message is compressed, which saves the transmission time and disk space.
4.	Email compatibility	Radix 64 conversion	For email applications transparency, an encrypted message converted to an ASCII string using Radix 64 conversion.
5.	Segmentation		To resemble the segments before decryption process.

Scanned by CamScanner

Following are the detail encryption and decryption steps of PGP :

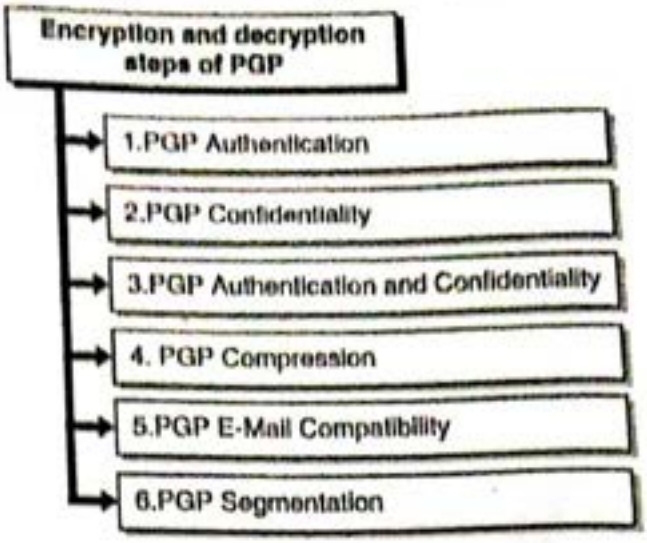


Fig. 12.5.1 : Encryption and decryption steps of PGP

➜ 1. PGP Authentication

1. Ramesh has (private/public) key pair *(Rd/Re)* and he wants to send a digitally signed message *m* to Suresh.
2. Ramesh hashes the message using SHA-1 to obtain *SHA(m)*.
3. Ramesh encrypts the hash using his private key *Rd* to obtain ciphertext *c* given by

$$c = encrypt_{Rd}(SHA(m))$$

4. Ramesh sends the pair *(m,c)to Suresh*
5. Suresh receives *(m,c)* and decrypts *c* using Ramesh's public key *Re* to obtain signature S

$$S = decrypt_{Re}(c)$$

6. He computes the hash of *m* using SHA-1 and if this hash value is equal to S then the message is authenticated.

   Suresh is sure that the message is correct and that came from Ramesh. Furthermore Ramesh cannot later deny sending the message since only Ramesh has access to his private key *Rd* which works with respective public key *Rd*.

➜ 2. PGP Confidentiality

1. Ramesh wishes to send Suresh a confidential message *m*.
2. Ramesh generates a random session key *k* for a symmetric cryptosystem.
3. Ramesh encrypts *k* using Suresh's public key *Be* to get.

Scanned by CamScanner

$$k' = encrypt_{Be}(k)$$

4. Ramesh encrypts the message m with the session key k to get ciphertext c

$$c = encrypt_k(m)$$

5. Ramesh sends Suresh the values (k',c)
6. Suresh receives the values (k',c) and decrypts k' using his private key B_d to obtain k.

$$k = decrypt_{Bd}(k')$$

7. Suresh uses the session key k to decrypt the ciphertext c and recover the message m

$$m = decrypt_k(c)$$

Public and symmetric key cryptosystems are combined in this way to provide security for key exchange and then efficiency for encryption. The session key k is used only to encrypt message m and is not stored for any length of time.

3. PGP Authentication and Confidentiality

The schemes for authentication and confidentiality can be combined so that Ramesh can sign a confidential message which is encrypted before transmission. The steps required are as follows :

1. Ramesh generates a signature c for his message m as in the Authentication scheme

$$c = encrypt_{Rd}(SHA(m))$$

2. Ramesh generates a random session key k and encrypts the message m and the signature c using a symmetric cryptosystem to obtain ciphertext C

$$C = encrypt_k(m,c)$$

3. He encrypts the session key k using Suresh public key

$$k' = encrypt_{Be}(k)$$

4. Ramesh sends Suresh the values (k',C)
5. Suresh receives k' and C and decrypts k' using his private key Bd to obtain the session key k

$$k = decrypt_{Bd}(k')$$

6. Suresh decrypts the ciphertext C using the session key k to obtain m and c

$$(m, c) = decrypt_k(C)$$

7. Suresh now has the message m. In order to authenticate it he uses Ramesh public key Re to decrypt the signature c and hashes the message m using SHA-1.

Scanned by CamScanner

If $SHA(m) = decrypt_{R_u}(c)$

Then the message is authenticated.

4. PGP Compression

PGP can also compress the message if desired. The compression algorithm is ZIP and the decompression algorithm is UNZIP.

1. The original message m is signed as before to obtain

$$c = encrypt_{R_d}(SHA(m))$$

2. Now the original message m is compressed to obtain

$$M = ZIP(m)$$

3. Ramesh generates a session key k and encrypts the compressed message and the signature using the session key

$$C = encrypt_k(M,c)$$

4. The session key is encrypted using Suresh's public key as before.
5. Ramesh sends Suresh the encrypted session key and ciphertext C.
6. Suresh decrypts the session key using his private key and then uses the session key to decrypt the ciphertext C to obtain M and c

$$(M,c) = decrypt_k(C)$$

7. Suresh decompresses the message M to obtain the original message m

$$m = UNZIP(M)$$

8. Now Suresh has the original message m and signature c. He verifies the signature using SHA-1 and Ramesh's public key as before.

5. PGP E-Mail Compatibility

- Many electronic mail systems can only transmit blocks of ASCII text. This creates a problem when sending encrypted data which is in cipher text form might not correspond to ASCII characters that can be transmitted.
- PGP overcomes this problem by using **Radix-64 conversion.**
- Suppose the text to be encrypted has been converted into binary using ASCII coding and encrypted to give a ciphertext stream of binary. Radix-64 conversion maps arbitrary binary into printable characters.

1. The binary input is split into blocks of 24 bits (3 bytes).

Scanned by CamScanner

2. Each 24 block is then split into four sets each of 6-bits.
3. Each 6-bit set will then have a value between 0 and $2^6 - 1$ (= 63).
4. This value is encoded into a printable character.

6. PGP Segmentation

- Another constraint of e-mail is that there is usually a maximum message length.
- PGP automatically blocks an encrypted message into segments of an appropriate length.
- On receipt, the segments must be re-assembled before the decryption process.
- Following are the service offered by the PGP :

1. Authentication
2. Confidentiality
3. Non-repudiation
4. Integrity
5. Compression
6. E-mail compatibility
7. Segmentation

12.5.2 Backdoors and Key Escrow in PGP

- Suppose, we have saved your password in laptop. So, anyone who has access the laptop, can get unauthorized access to your account. And that is a simple way of saying what a **Backdoor** is.
- A **Backdoor** is a method for bypassing normal authentication in a system and thus, provide unauthorized remote access to the system to malicious users.
- A backdoor is a "feature" in the software of PGP like an utility functions but not in the encryption algorithm that allows an outside party to decrypt which is encrypted by PGP.
- A Backdoor may be implemented as a hidden part of a program or a separate program or even be implemented by hardware.
- Just to give an example, in 2003 a Backdoor was planted in Linux Kernel. In a conditional statement for checking root access permission, '= =' was replaced with '='. As a result, it gave unauthorizd access to malicious callers. Even very recently, in 2015, **Juniper Networks** have warned about a malicious Backdoor in their firewalls that automatically decrypts VPN traffic.
- There are two types of Backdoors - **Object Code Backdoors** and **Asymmetric Backdoors.**

Scanned by CamScanner

- In **Object Code Backdoors**, software source code remains unchanged, but the object code gets modified maliciously. As the object code is designed to be machine readable, it becomes much more difficult to detect. These types of Backdoors are inserted in the on-disk object code or inserted at some point during compilation, linking or loading.
- Recompiling the software source code may get rid of the Backdoors. So, malicious users sometimes change the compiler source code in such a way that, whenever it compiles, links and loads the source code, the Backdoor is inserted. These Backdoors can be fixed by recompiling the compiler and removing the Backdoor inserting codes.
- Normally, Backdoors are symmetric. Anyone who finds the Backdoor, can in turn use it. But, **Asymmetric Backdoors** can be exploited only by the attacker who plants it, even if the Backdoor implementation becomes public. This type of attacks are termed as **Kleptography** and they can be carried out in software, hardware or in combination of both. The theory of Asymmetric Backdoors is a part of a larger field named **Cryptovirology**.

Counter measures

- Once Backdoors are detected, rebuild a clean system and transfer data.
- Another method is to use Diverse Double Compiling or DDC. It requires a different compiler and the source code of the compiler to be tested. That source code, while compiled with two different compilers, would result in two different stage-1 compilers showing same behaviour.
- Thus, the same source code compiled in two different stage-1 compilers, must result in two identical stage-2 compilers. This method was applied to verify that C compiler of GCC Suite contained no Trojan, using the icc as the other compiler. Normally, Operating Systems vendors implement these type of methods to make sure they are not distributing a compromised system.

Key Escrow

- Key escrow is a cryptographic key exchange process in which a key is held in escrow, or stored, by a third party. A key that is lost or compromised by its original user(s) may be used to decrypt encrypted material, allowing restoration of the original material to its unencrypted state.
- Key escrow systems provide a backup source for cryptographic keys. Escrow systems are somewhat risky because a third party is involved.

Scanned by CamScanner

- The Clipper Chip was a U.S. government encryption chipset introduced in 1993. The chipset was promoted as an encryption device with a government-held (escrow) master key to facilitate encryption in the face of security threats. The controversial Clipper Chip was defunct by 1996, but the concept evolved into the Pretty Good Privacy (PGP) encryption tool, which is used worldwide.
- **Key escrow** (also known as a "fair" cryptosystem) is an arrangement in which the keys needed to decrypt encrypted data are held in **escrow** so that, under certain circumstances, an authorized third party may gain access to those keys.

Syllabus Topic : Honey Pots

12.6 Honeypot

- A honeypot is a decoy computer system for trapping hackers or tracking unconventional or new hacking methods. Honeypots are designed to purposely engage and deceive hackers and identify malicious activities performed over the Internet.
- Multiple honeypots can be set on a network to form a honeynet.
- There are many advantages to honeypots. The main one is the ease with which they are employed. Another advantage is that although honeypots seek small amounts of hacker information, the information is considered highly valuable for studying and uncovering hackers' motivations.
- Honeypot systems are not perfect, however. They contain the usual technology risks such as firewall penetration, broken encryption methods and failure to detect attacks. In addition, honeypots are unable to detect attacks against systems that are not honeypot systems.
- There are two different kinds of honeypots. They are classified based on their deployment method :
  - **Production Honeypot :** Used by companies and corporations for the purpose of researching the motives of hackers as well as diverting and mitigating the risk of attacks on the overall network.
  - **Research Honeypot :** Used by nonprofit organizations and educational institutions for the sole purpose of researching the motives and tactics of the hacker community for targeting different networks.

Scanned by CamScanner

- Honeypots are not always designed to identify hackers.
- Honeypot developers are often more interested in getting into the minds of hackers, which then permits them to design more secure systems, as well as to educate other professionals about the lessons learned through their efforts.
- Overall, honeypots are considered an effective method to track hacker behavior and heighten the effectiveness of computer security tools.

Chapter Ends...

□□□

Scanned by CamScanner

Module 6

CHAPTER 13

Software Vulnerabilities

Syllabus

Software Vulnerabilities : Buffer Overflow, Format string, cross-site scripting, SQL injection, Malware : Viruses, Worms, Trojans, Logic Bomb, Bots, Rootkits.

13.1 Program Security

13.1.1 Secure Programs

Q. 13.1.1 What is secure program ? (Ref. sec. 13.1.1)

- A secure program is which provides or enforces availability, confidentiality and integrity.
- But different people may have different security requirements and needs.

Examples

1. **Naïve user :** Fit for his or her work is good enough.
2. **Programmer :** Passes all his / her tests while programming.
3. **Manager :** All requirements and specifications are covered.
4. **Developer :** Correct implementation, functioning and testing.

- This security can be judged by following ways,

1. **By fixing software faults**

   While fixing software faults especially by quick fix method new faults can be introduced. Mainly these faults are caused by work pressure, deadlines, side effects of the fault fixing, system performance requirements etc.

2. **By testing program behavior**

   While coding typing errors may cause faults / failures in the program. In requirement

Scanned by CamScanner

analysis phase if requirements are not clearly understood then again it may cause wrong coding and implementation. Again different components in the program have to communicate with each other which may produce intentional or accidental flaws in the program. Accidental faults are used by attackers to exploit vulnerability. Therefore testing must be done thoroughly to avoid such faults.

But there are some limitations for this testing also. Testing can check that what program should do. It cannot check what program should not do. Again program complexity i.e. complex coding, complex data structures as well as changing technologies are the challenges for testing.

3. **By program analysis**

It is one of the best approaches for judging software security. Care should be taken right from requirement analysis to deployment stage. While developing program, testing and debugging should be performed carefully by considering all the scenarios. Based on the analysis specialized security methods and techniques can be implemented.

13.1.2 Non-malicious Program Errors

→ (MU - Dec. 15)

Q. 13.1.2 With the help of examples explain non malicious programming errors. (Ref. sec. 13.1.2) **Dec. 15, 5 Marks**

Q. 13.1.3 What is non malicious program errors ? (Ref. sec. 13.1.2)

While programming, a programmer can make mistakes/errors. Most of these errors are not intentionally done. Many such kind of errors do not have huge impact on security. Program may produce wrong or incorrect results but it is non-malicious.

Following are the three types of non-malicious program errors,

1. **Buffer overflows**

Q. 13.1.4 What is buffer overflow in software security ? (Ref. sec. 13.1.2(1))

- Attacker can insert malicious data values / instruction codes into overflow space.
- Array bound checking is not performed by C compiler, pointer limits cannot be defined as well.
- **Example :** int B[15];
- Here the array bound is (0 to 14). i.e. B[0].....B[14].

Scanned by CamScanner

- If anything inserted after that bound then the adjacent data is overwritten.
- Attacker can overwrite users data, changes users instruction, overwrite OS data, changes OS instructions. Thus can get complete control of a program or OS. This is also known as aliasing.

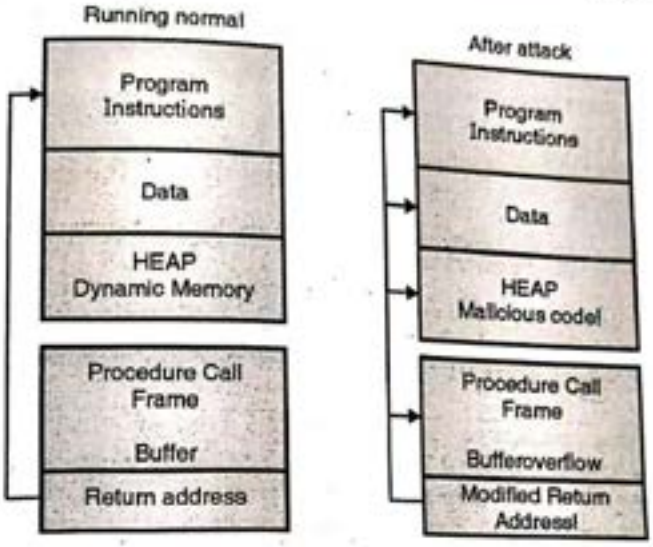


Fig. 13.1.1 : Buffer overflow attack

- As shown in Fig. 13.1.1 attacker changes the return address and thus can transfer the control of the program.

2. Incomplete mediation

Q.13.1.5 What is incomplete mediation in software security ? (Ref. sec. 13.1.2(2))

- Due to incomplete mediation serious security threats can be introduced as sensitive data may get exposed and can result in uncontrolled condition.
- **Example :** URL : http://www.onlinestore.com/purchase/total = 935.
- User can edit the total cost and resubmit the request to the server, URL : http://www.onlinestore.com/purchase/total = 035.
- Such kinds of vulnerabilities are very dangerous.
- Proper care should be taken to avoid such vulnerabilities. Such editing permissions should not available to the user.

Scanned by CamScanner

3. Time-of-check to time-of-use errors (TOCTOU)

- This is one of the best examples of RACE condition.
- RACE condition is very vulnerable to attack.
- **Example :** If two threads are sharing their root and current directories then, Let Thread X's current working directory is /college.

```
Thread X calls open("shadow");
Y calls chdir("/department")
system monitor permits both the calls
open("shadow") executes with /department as working directory
X's call now opens "/department/shadow"
```

- Proper locking mechanism can prevent this kind of attack. Time lags should be considered. After checking values it must be locked using digital signatures and certificates. Thus after check data cannot be modified.

Syllabus Topic : Malware - Logic Bomb, Bots

13.1.3 Malicious Software

Q. 13.1.6 What are different types of malicious code ? (Ref. sec. 13.1.3)

Q. 13.1.7 What is malicious code ? (Ref. sec. 13.1.3)

Q. 13.1.8 What are the different types of malicious software's ? (Ref. sec. 13.1.3)

Malicious software is software where an attacker can get partial or full control of the program. Thus attacker is free to do anything that he / she want to do.

Fig. 13.1.2 shows different types of malicious software's.

Malware is currently the major source of attacks and fraudulent activities on the Internet. Malware is used to infect computers. Malware, short form is malicious software or also called as malicious software.

Scanned by CamScanner

Types of Malicious Software

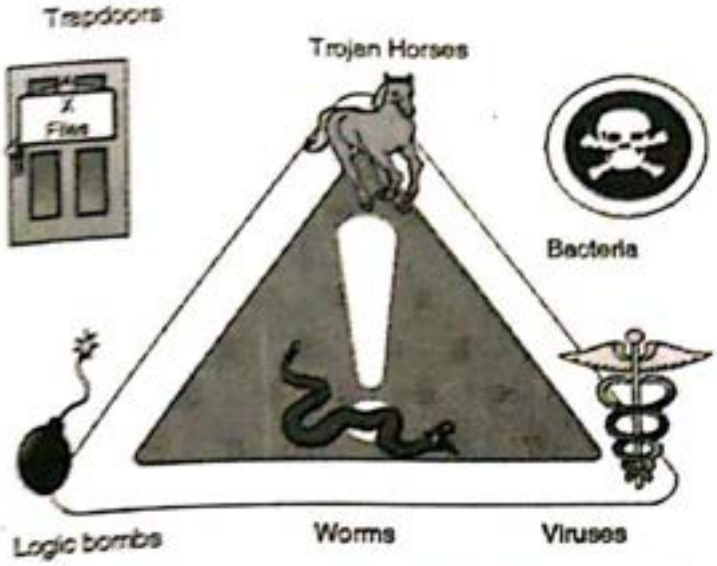


Fig. 13.1.2 : Different types of malicious software's

(1) Botnet

- **Botnet** is a network of zombies, i.e. compromised computers under control of an attacker. Bot is a program loaded on zombie computer (a **zombie** is a **computer** connected to the Internet that has been compromised by a hacker) that provides remote control mechanisms to an attacker. Bot - a small program to remotely control a computer.
- Bot is characterized by Remote control and communication (C&C) channels to command a victim (Means of receiving and sending commands and information between the bot master and the zombies) as shown in Fig 13.1.3. For example, perform denial-of service attack, send spam.

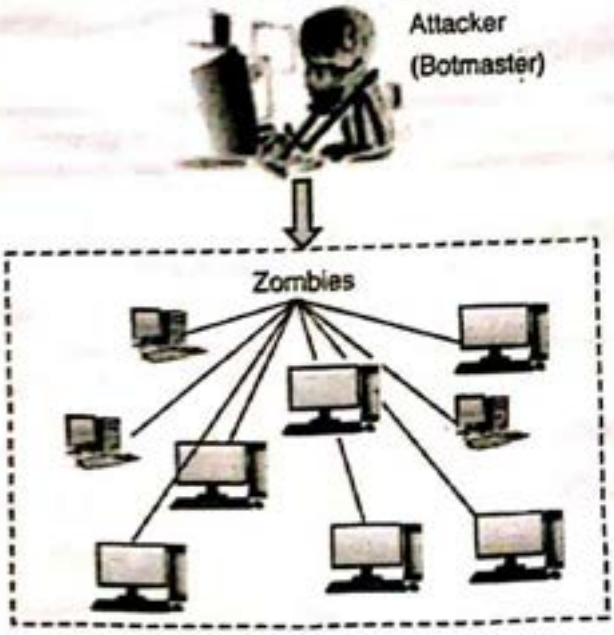


Fig 13.1.3 : Botmet

Scanned by CamScanner

(2) Trojan horse

It is a computer program. Along with some useful code or function, some hidden malicious code or function is there which may hamper performance of security mechanisms. Useful information can be stolen by attackers.

(3) Bacterium

Bacterium is a special kind of virus. Virus is getting attached with different files but bacterium does not get attached to a specific file.

(4) Logic bomb

Logic bomb is generally used in DoS (Denial of Service) attacks. When specified conditions are met it activates malicious program logic. It may damage system resources greatly.

(5) Time bomb

This gets activated when specified time occurs.

(6) Rabbit

It is a kind of virus / worms that replicates itself without any limits. The intension is to exhaust resources.

(7) Trapdoor / backdoor

An intruder can enter into the system by bypassing all security services or mechanisms. Thus intruder knows the flaws or loopholes in the system and can get these loopholes to gain access to the computer.

Syllabus Topic : Viruses and Worms

13.1.4 Virus and Worms

→ **(MU - Dec. 15, Dec. 16)**

Q. 13.1.9 Write a brief on - Viruses and their types. (Ref. sec. 13.1.4) **Dec. 15, 5 Marks**

Q. 13.1.10 What are the different types of viruses and worms? How do they propagate? (Ref. sec. 13.1.4) **Dec. 16, 10 Marks**

- Virus and worms are the classes of malicious software which are capable of replicate itself or copy the contents many times or even can modifies the system settings or data.

Scanned by CamScanner

- The basic differences between worm and virus are, virus needs a host programme to propagate or spread itself whereas worm does not need host it propagates independently but slowly.
- Virus spreads or infects system without priory informing the user the activities like deletion of file, halting of system etc. virus can affect system mildly, effecting the system's data or can cause severe like denial of service.
- Almost all viruses come with some of the executable files. Whereas worm are standalone software they enter system by finding loop hole in the system and take advantage of file transport features of system.

13.1.4(A) Types of Virus

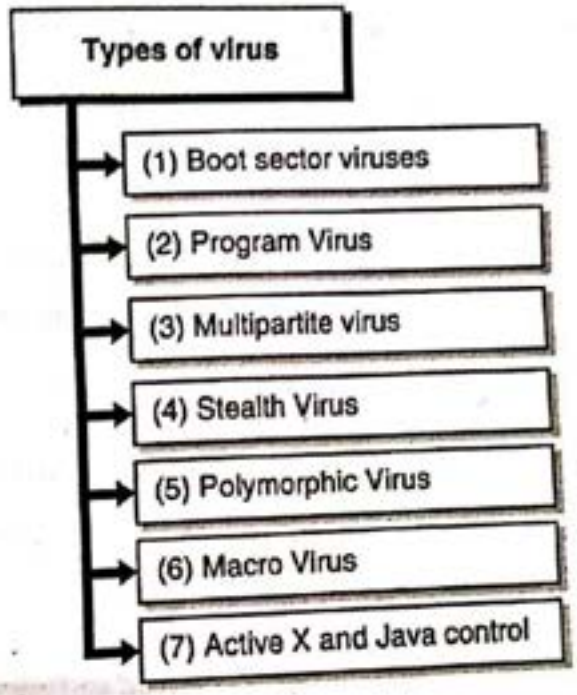


Fig. 13.1.4 : Types of virus

- **(1) Boot sector viruses**
  - It infects the storage media like dislects and hard drives. All disks or hard serves contain sector and the first sector is called as **Boot Sector**.
  - This boot carries **Master Boot Record** which is used to read and load operating system.
  - The virus infect itself sector while rebooting system Boot sector also spreads other computers if same disk is shared to other system.

Scanned by CamScanner

➜ **(2) Program Virus**

A program virus gets active when program containing these virus gets opened (.bin, .exe, .ove), once if gets open it starts copying itself and infect other program.

➜ **(3) Multipartite virus**

- It is combination/hybrid of boot sector and program virus.
- It infects the program files. When this virus is active it will affect boot sector also after booting or starting up it will affect other computer also.

➜ **(4) Stealth Virus**

- "Dubbed Brain" the first computer virus was a stealth virus it tries to disguise itself, so that antivirus software may not able to recognize it.
- It alters the file size, concealing file's memory and so on.

➜ **(5) Polymorphic Virus**

It keeps on changing it patterns or signature to get undetected. Usually it acts like a 'chamleon'. These are not actual virus, it is a virus which hides actual virus of the system.

➜ **(6) Macro Virus**

Applications such as MS word, excel sheets has macro supportive language. These virus infects victim every documents once it get into victims systems.

➜ **(7) Active X and Java control**

- All web browser need java control active X enable to function properly.
- Awareness is needed about managing and controlling settings of browser to check for enabling or disabling popups, downloading files and sounds, since these can invite virus which can affect computer by downloading unwanted software.

13.1.4(B) Types of Computer Worms

1. E-mail worms

It spreads through infected email message of any infected websites.

2. Instant messaging worms

It spreads by sending link to contact list of instant messaging application.

Scanned by CamScanner

3. **Internet worm**

It scans all network resources which are available and system. If it found vulnerable, it will take advantage and gain access.

4. **IRC (Internet Relay Chat) worms**

It places a copy of itself through link in infected websites.

5. **File sharing Network worms**

It places a copy of itself in a folder which is sharable and spread via P2P network.

13.1.4(C) Difference between Virus and Worm

Q. 13.1.11 What is the difference between Virus and Worm ? (Ref. sec. 13.1.4(C))

Sr. No.	Feature	Virus	Worms
1.	Types	Stealth virus polymorphic, metamorphic etc.	Email worm, IRC worm, file sharing worm etc.
2.	Mode of spreading	Need host program to spread.	It does not need host it spreads by itself.
3.	What it is ?	It is a software program that can copy itself and infect the data or information without the knowledge.	It is self-replicating spreads through a network.
4.	Inception	'Creeper' virus first known virus spread through ARPANET in 1970.	The name originated by the shock wave rider a novel of science fiction in 1975 from where name is adopted.
5.	Prevalence	More than 100, 000 known computer virus have been there through only few have attacked system.	Worm existence is moderate as compare to virus.

Famous virus and worm are I LOVE YOU virus Morris worm, melissa, conficker etc.

Scanned by CamScanner

Syllabus Topic : Malware - Trojans, Rootkits

13.1.5 Targeted Malicious Code

Q. 13.1.12 Explain different targeted malicious code. (Ref. sec. 13.1.5)

Q. 13.1.13 Write a short note on targeted malicious code. (Ref. sec. 13.1.5)

This is a computer code which is written to attack a particular system, a particular application and for a particular purpose.

Example

(1) Trapdoor / backdoor

- An intruder can enter into the system by bypassing all security services or mechanisms. Thus intruder knows the flaws or loopholes in the system and can get these loopholes to gain access to the computer.
- Trapdoors are the entry points which are notdocumented but still inserted during code development for testing purpose, for future extensions or for an emergency access if software fails. These loopsholes are purposely kept in the system with good intension.
- **Major sources of Trapdoors / Backdoors**
  - During testing of the system stubs, drivers are created. These are temporary functions which then further replaced by actual functions. Sometimes some malicious code is intentionally injected into the system for testing purpose.
  - Poor error checking conditions.
  - Undefined opcodes in hardware processors.

(2) Salami Attack

→ **(MU - Dec. 15, Dec. 16)**

Q. 13.1.14 Define the following with example - Salami Attack. (Ref. sec. 13.1.5(2)) **Dec. 15, 2 Marks**

Q. 13.1.15 Explain briefly with example, how the following attacks occur - Salami Attack. (Ref. sec. 13.1.5(2)) **Dec. 16, 2 Marks**

It is series of small attacks which results in large attack. It works on "collect and roundoff" trick. It is a fraudulent practice of stealing money repeatedly. It takes advantage of rounding operation in financial transactions. It always rounds down and thus the fractions of amount remained will be transfered into some another account. Thus the transaction will go undetected. Such type of attacks can be easily automated.

Scanned by CamScanner

(3) Covert channels

Q. 13.1.16 Write a short note on covert channel. (Ref. sec. 13.1.5(3))

In covert channel the processes which are not allowed to communicate and transfer the information by security policy can communicate and transfer data using current system objects. Such types of attacks are virtually no detectable by system or administrators. Fig. 13.1.5 shows channel creation.

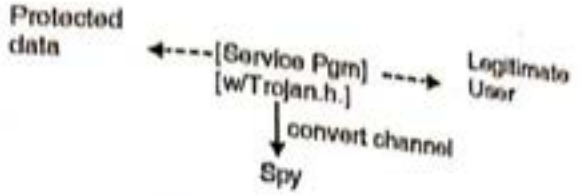


Fig. 13.1.5 : Covert channel creation

(4) Trojan

Q. 13.1.17 Write a short note on trojan. (Ref. sec. 13.1.5(4))

It is a computer program. Along with some useful code or function some hidden malicious code or function is there which may hamper performance of security mechanisms. Useful information can be stolen by attackers.

(5) Rootkits

A rootkit is a computer malware program installed by an intruder. Intruder installs it by avoiding detection. The purpose is to gain control of the computer system. Basically it is a kind of Trojan horse malwares. System scan cannot detect it. If system is infected with a rootkit, then it becomes very hard to trust infected operating system. The best solution is to shut down the infected system and boot that system by some CD-ROM, Pendrive and clean it.

Following are the types of rootkits :

1. **User Mode**

   User mode rootkits can get administrator privileges. They are automatically activated at system startup. These are detectable rootkits and can be easily removed.

Scanned by CamScanner

2. **Kernel Mode**

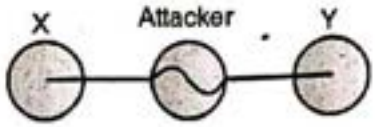
   Kernel mode rootkits are installed like an OS hence can corrupt the functionality of complete OS. These Rootkits are very hard to detect. It can be detected only after some event or crash.

3. **Firmware**

   Firmware's are dangerous amongst all. Malcode is created inside a firware. At system startup this malware will be reinstalled. It is very hard to remove.

(6) Man in the middle attack (MITM/MIMA)

Attacker relays and sometimes alters the communication between two parties without knowing to communicating parties.



– It is explained as follows :

1. X sends a message to Y, which is intercepted by Attacker :

   X "I want to deposit money in your account. Please send account number".

2. Attacker relays this message to Y; Y cannot tell it is not really from X.

3. Y receives a message from X and responds it with account number.

   Y "My account number is 012345".

4. Attacker again intercepts a message from Y replaces Y's account number with his own account number and relays this to X, claiming that it is Y's message.

– Attacker "My account number is 067891" :

1. X receives message from Y and gets the account number of Y. Thus X believes that it is Y's account number and deposits money in that account.

2. X and Y both think that it is a secure communication.

13.1.6 Controls against Program Threats

Q. 13.1.18 Write different control measures against program threats. (Ref. sec. 13.1.6)

Program can fail by many different ways which may give rise to a security threat. Care must be taken during the development of the program itself.

Scanned by CamScanner

lowing are the controlling techniques :

Use of fundamental principles of programming like encapsulation, modularity and information hiding.

Peer reviews are most effective. Reviews, Walkthrough and inspection techniques can be used to control program threats.

Hazard analysis gives systematic approach to identify potential threats.

Testing can be performed to minimize the flaws in the system. It ensures correct implementation and working of the program.

Good design makes it easy for development and testing.

Risk prediction and mangement ensures easy risk management.

Static analysis examines design and code before release to identify flaws.

Configuration management ensures system controling, modifications during development and maintenance.

Syllabus Topic : Software Vulnerability - Buffer Overflow

13.2 Buffer Overflow

→ **(MU - Dec. 15)**

Q. 13.2.1 Write in brief about : Buffer overflow attack. (Ref. sec. 13.2) **Dec. 15, 5 Marks**

It is also known as buffer overrun. It deviates from a standard, where the process stores data in buffer overruns the buffer's boundary and overwrites adjacent memory locations. Buffer overflow can be triggered by inputs that are designed to execute code or alter the way the program operates. Bound check can prevent buffer overflow.

The languages which are commonly associated with buffer overflow are : C and C++, as it provides no built in protection against accessing or overwriting data in any part of memory. Buffer overflow occur when a process tries to store data in buffer then it was intended to hold.

Scanned by CamScanner

Types of buffer overflow

1. **Stack based buffer overflow :** When program writes in memory address, on program's call stack outside the intended data structure, then stack overflow occurs. The condition where Buffer being overwritten is allocated on the stack (i.e., is a local variable or a parameter to a function).

2. **NOP (No Operation) :** It is an assembly language instruction command that effectively does nothing at all. NOP enables developer to force memory alignment to act as a place holder to be replaced by active instruction later on in program development. NOP opcode can be used to form an NOP slide, which allows code to execute when exact value of instruction pointer is indeterminate. Fig. 13.2.1 shows NOP operation.

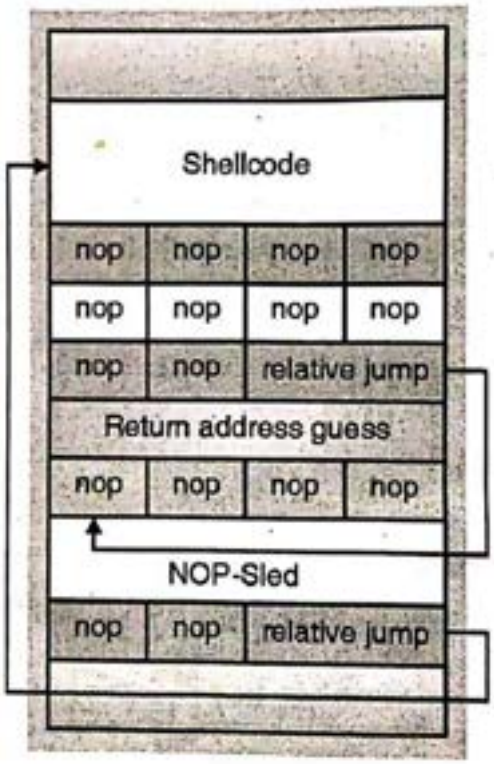


Fig. 13.2.1 : NOP operation

3. **Heap buffer overflow :** In Buffer overflow, the overflow occurs when an application copies more data into buffer then the buffer was designed to contain. The heap space is dynamically allocated by new(), malloc(), calloc() dynamically allocated in runtime.

Scanned by CamScanner

Syllabus Topic : Software Vulnerability - Format String

13.3 Format String Attacks

Introduction

- The format string attacks occur when the submitted data of an input string is evaluated as a command by particular application.
- In format string attacks attacker execute the code, read the stack, or cause a segmentation fault in the running application, which compromise the security breaches of the system or programs.
- To understand this attack it very important to understand the following definitions.
- The **format function** is an ANSI C conversion function, like **printf, fprintf, sprint, snprintf** etc which converts a primitive variable of the programming language into a human-readable string representation.
- Table 13.3.1 shows the different format functions.

Table 13.3.1 : Format functions

Format function	Description
fprint	Writes the printf to a file
printf	Displays output of a formatted string
sprintf	Prints data into a string
snprintf	Prints data into a string and checking its the length

- The **format string** is the argument of the Format Function and is an ASCII Z string which contains text and format parameters, for example **printf ("This is my roll number : %d\n", 0007).**
- The example display output on screen the text **This is my roll number** and content of its format parameters i.e. 0007.
- Table 13.3.2 shows list of common parameters used in a format string attack.

Scanned by CamScanner

Table 13.3.2 : Common parameters used in a format string attack

Parameters	Output	Passed as
%d	Decimal number	Value
%c	Character	----
%u	Unsigned decimal number	Value
%x	Hexadecimal number	Value
%s	String or words	Reference
%n	Writes the number of characters into a pointer	Reference
%%	% character (literal)	Reference
%p	External representation of a pointer to void	Reference

- The **format string parameter**, like **%d %x %s %c %p** defines the type of conversion of the format function.
- We have demonstrated the following examples on Linux environment by using ***Linux cross compiler*** (gcc) which results how the application can behave when the format function does not receive the necessary values for validation in the input of format string.
- First example shows the application operating with normal behaviour and normal inputs, then, we will discuss the application operating when the attacker inputs the format string and the resulting behaviour.

Example 1

```
#include <stdio.h>

void main()
{
 int i = 77;
 char a = 'a';
 printf("The value of int i & char a is : %d %d\n", i, a);
 printf("The value of int i & char a is : %c %c\n", i, a);
}
```

Scanned by CamScanner

Save above program as example.c and compile and run it using

```
[root@localhost santosh]# gcc example1.c
[root@localhost santosh]#./a.out
int : 77 97
char : @ a
```

The first `printf()` writes the value of the integer variable `i` and of the character variable `a` as `int` (using `%d`), which display its ASCII value. On the other hand, the second `printf()` converts the integer variable `i` to the corresponding ASCII character code.

Example 2

If the programmer passes an attacker-controlled buffer as the argument to a `printf()` the attacker can perform and writes to arbitrary memory addresses. The following program contains such an error :

```
#include<stdio.h>
void main(int argc, char** argv)
{
char buf[100];
strncpy(buf, argv[1], 100);
printf(buf);
}
```

As `printf` has a variable number of arguments, it must use the format string to determine the number of arguments. In the case above, the attacker can pass the string `"%p %p %p %p %p %p %p %p %p %p %p %p"` and try to fool the `printf` statement to think about it has 12 arguments ?

It will print the next 12 addresses on the stack, thinking they are its arguments as shown in the output statement.

```
[root@localhost santosh]# gcc example2.c
[root@localhost santosh]#./a.out "%p %p %p %p %p %p %p %p %p %p %p %p"
0xffffeeee 0x64 0xf7ec145 0xffffdbdf 0xffffdbde (nil) 0xffffdcc4
```

The attacker can pass a sequence of format strings, making the program show the memory address where a lot of other data are stored, then, the attacker increases the possibility that the program will read an illegal address, crashing the program and causing its non-availability.

```
printf ("%s%s%s%s%s%s%s%s%s%s%s%s");
```

Scanned by CamScanner

If attacker pass %s into the printf() function which will fetch a number from the stack, treat this number as an address, and prints the memory contents pointed by this address as a string, until a NULL character (i.e., number 0) is encountered.

In this case whatever number fetched by the printf function might not be the address, the memory displayed by this number might not exist because of such illegal fetching of memory address the program will crash such type of attack is called as **format string attacks.**

Syllabus Topic : Cross Site Scripting

13.4 Cross Site Scripting

→ (MU - Dec. 16, Dec. 17)

Q. 13.4.1 Explain briefly with example, how the Cross-Site scripting attack. (Ref. sec. 13.4) **Dec. 16, Dec. 17, 3 Marks**

- Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites.
- XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user.
- Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.
- An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script.
- Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.
- Cross-Site Scripting (XSS) attacks occur when :
  1. Data enters a Web application through an untrusted source, most frequently a web request.
  2. The data is included in dynamic content that is sent to a web user without being validated for malicious content.
- The malicious content sent to the web browser often takes the form of a segment of JavaScript, but may also include HTML, Flash, or any other type of code that the browser may execute.

Scanned by CamScanner

- The variety of attacks based on XSS is almost limitless, but they commonly include transmitting private data, like cookies or other session information, to the attacker, redirecting the victim to web content controlled by the attacker, or performing other malicious operations on the user's machine under the guise of the vulnerable site.

13.4.1 Stored and Reflected XSS Attacks

- XSS attacks can generally be categorized into two categories : stored and reflected. There is a third, much less well-known type of XSS attack called DOM Based XSS.

13.4.2 Stored XSS Attacks

- Stored attacks are those where the injected script is permanently stored on the target servers, such as in a database, in a message forum, visitor log, comment field, etc.
- The victim then retrieves the malicious script from the server when it requests the stored information. Stored XSS is also sometimes referred to as Persistent or Type-I XSS.

13.4.3 Reflected XSS Attacks

- Reflected attacks are those where the injected script is reflected off the web server, such as in an error message, search result, or any other response that includes some or all of the input sent to the server as part of the request.
- Reflected attacks are delivered to victims via another route, such as in an e-mail message, or on some other website.
- When a user is tricked into clicking on a malicious link, submitting a specially crafted form, or even just browsing to a malicious site, the injected code travels to the vulnerable web site, which reflects the attack back to the user's browser.
- The browser then executes the code because it came from a "trusted" server. Reflected XSS is also sometimes referred to as Non-Persistent or Type-II XSS.

13.4.4 Other Types of XSS Vulnerabilities

- In addition to Stored and Reflected XSS, another type of XSS, DOM Based XSS was identified by Amit Klein in 2005.
- OWASP recommends the XSS categorization as described in the OWASP Article : Types of Cross-Site Scripting, which covers all these XSS terms, organizing them into a matrix of Stored vs. Reflected XSS and Server vs. Client XSS, where DOM Based XSS is a subset of Client XSS.

Scanned by CamScanner

13.4.5 XSS Attack Consequences

- The consequence of an XSS attack is the same regardless of whether it is stored or reflected (or DOM Based). The difference is in how the payload arrives at the server.
- Do not be fooled into thinking that a "read-only" or "brochure ware" site is not vulnerable to serious reflected XSS attacks. XSS can cause a variety of problems for the end user that range in severity from an annoyance to complete account compromise.
- The most severe XSS attacks involve disclosure of the user's session cookie, allowing an attacker to hijack the user's session and take over the account.
- Other damaging attacks include the disclosure of end user files, installation of Trojan horse programs, redirect the user to some other page or site, or modify presentation of content.
- An XSS vulnerability allowing an attacker to modify a press release or news item could affect a company's stock price or lessen consumer confidence.
- An XSS vulnerability on a pharmaceutical site could allow an attacker to modify dosage information resulting in an overdose.

Syllabus Topic : SQL Injection

13.5 SQL Injection

→ (MU - Dec. 17)

Q. 13.5.1 Explain briefly with example, how the SQL injection attack. (Ref. sec. 13.5) **Dec. 17, 3 Marks**

- It is a source code injection technique in which malicious SQL statements are inserted into entry field of database to dump data base content.
- Attacker targets the database organization where confidential data is stored.
- Its main focus is to get information from the database server stored in database table by sending malicious query since database can be accessible by query.
- When legitimate user enters an additional database via web form, the attacker sends its own command through same web form field. The attackers before proceeding always checks whether organization's database has any loop is it vulnerable or not.

Scanned by CamScanner

Steps for SQL Injection

(1) The attacker looks for login pages search pages or feedback pages or pages that display HTML commands like POST or GET.

(2) Attacker checks the source code of the web page by right click on web page and view source.

(3) It checks term <form> tag everything insides <form> tag </form >have potential of getting vulnerabilities.

(4) The attacker puts single quote under the text which accepts username and password. If response is an error message such as "a" = 'a' (something like) then website is vulnerable.

(5) Attacker than uses SQL command such as SELECT to retrieve data or INSERT command to add information to database.

Benefits for attacker using SQL Injection

(1) Obtain basic information about website or organization.

(2) May gain access to database by obtaining username, password from SELECT command FROM command where command.

(3) Can add new data to the database by executing INSERT command.

(4) Can modify data in the database by UPDATE command.

Prevention from SQL Injection

SQL injection attacks happen because of poor website coding and poor administration of website

Step which can prevent SQL injection

(1) Replace all single quotes to two single quotes.

(2) Check the user input of any character and string that should not be malicious.

(3) Numeric value should also be checked.

(4) If there is SQL error it should be modified immediately but not be displayed to outsiders.

(5) SQL server 2000 which is a default server should never be used.

(6) Both database server and web server be reside in different machine.

Chapter Ends...

Scanned by CamScanner

Lab Manual

Scanned by CamScanner

List of Experiments

- **Experiment 1 :** Implementation and analysis of RSA cryptosystem and Digital signature scheme using RSA/ElGamal L-1
- **Experiment 2 :** Implementation of Diffie Hellman key exchange algorithm. L-6
- **Experiment 3 :** For varying message sizes, test integrity of message using MD-5, SHA-1, and analyse the performance of the two protocols. Use crypt APIs. L-9
- **Experiment 3(a) :** Write program in Java to implement MD5 algorithm for key generation and cipher verification. L-9
- **Experiment 3(b) :** Write a program in Java to implement SHA-1 algorithm using Libraries (API). L-22
- **Experiment 4 :** Study of packet sniffer tools - wireshark L-25
- **Experiment 4(a) :** Download and install wireshark and capture icmp, tcp, and http packets in promiscuous mode. L-25
- **Experiment 4(b) :** Explore how the packets can be traced based on different filters. L-33
- **Experiment 5 :** Download and install nmap. Use it with different options to scan open ports, perform OS fingerprinting, do a ping scan, tcp port scan, udp port scan, xmas scan etc. L-40
- **Experiment 6 :** Simulate DOS attack using Hping, hping3 and other tools. L-47
- **Experiment 6(a) :** DoS using hping3 ..... L-47
- **Experiment 6(b) :** DoS attack detection using Intrusion Detection System. L-50
- **Experiment 7 :** Setting up personal Firewall using iptables. L-53
- **Experiment 8 :** Set up Snort and study the logs. L-60

□□□

Scanned by CamScanner

'stem
mal L-1
ge algorithm..... L-6
sage using
the two
..... L-9
thm
..... L-9
orithm
..... L-22
..... L-25
tcp,
..... L-25
..... L-33
ions
ing
..... L-40
ols. L-47
..... L-47
..... L-50
..... L-53
..... L-60

□□□

List of Experiments

Experiment 1

Aim : Implementation and analysis of RSA cryptosystem and Digital signature scheme using RSA/El Gamal.

Objective

The objective of this assignment is implement RSA algorithm in C++ or in Java programming language. Using RSA algorithm we can generate public and private keys along encryption and decryption key.

Solution :

The algorithm works as follows

1. Select two prime numbers a and b where $a \neq b$.
2. Calculate $n = a * b$
3. Calculate $\phi(n) = (a - 1) * (b - 1)$.
4. Select e such that, e is relatively prime to $\phi(n)$ i.e. gcd $(e, \phi(n)) = 1$ and $1 < e < \phi(n)$.
5. Calculate of such that $d = e^{-1} \bmod \phi(n)$ or $ed \bmod \phi(n) = 1$.
6. Public key = {e, n}, private key = {d, n}.
7. Find out ciphertext using the formula,

   $C = P^e \bmod n$ where, $P < n$ and

   C = Ciphertext, P = Plaintext, e = Encryption key and n = Block size.
8. $P = C^d \bmod n$. Plaintext P can be obtain using the given formula.

   Where, d = decryption key.

Both sender and receiver know the value of n. In addition, the sender must know encryption key 'e' and receiver must know decryption key 'd'.

Use JDK latest version for compilation and running the java programs

```
import java.io.*;
import java.math.*;
```

Scanned by CamScanner

```
public class Sample
{
public static long p, q, n, phi, m, d, e, enc, dec;

public static long GCD(long phi)
    {

long a, c, b;

for(long i=4; i<phi; i++)
        {
          a=i; b=phi; c=b;
while(b!=0)
          {
           c=b;
           b=a%b;
           a=c;
          }
if(a==1)
        {
          e=i;
          i=phi;
        }
      }
return e;
    }
public static long Encryption(long n, long phi, long m)
    {

long x=m, y=m;

      e=GCD(phi);

for(long a=0;a<e-1;a++)
```

Scanned by CamScanner

```
    {
        x=x*y;
    }
enc=x % n;
return enc;
  }
public static BigInteger Decryption(long e, long enc, long n, long phi)
  {
long y, temp=phi+1,x;
BigInteger object2=null;

try
    {
for(long i=1;i<phi; i++)
      {
if(((i*e)%phi)==1)
        {
          d=i;
          i=phi;
        }
      }
igInteger object1=new BigInteger(Long.toString(enc));
igInteger object3=new BigInteger(Long.toString(n));

      object1=object1.pow((int)d);
      object2=object1.mod(object3);

    }
atch(Exception exception)
    {
ystem.out.print("\n Exception In DEC : " + exception);
    }
turn object2;
  }
```

Scanned by CamScanner

```
public static void main(String args[])
    {
try
        {
DataInputStream in=new DataInputStream(System.in);

System.out.print("\n Enter First Prime No : ");
            p=Integer.parseInt(in.readLine());
System.out.print("\n Enter Second Prime No : ");
            q=Integer.parseInt(in.readLine());

System.out.print("\n Please Enter Message Between ( 0 to 32):");
            m=Integer.parseInt(in.readLine());
            n=p*q;
phi=(p-1)*(q-1);

enc=Encryption(n,phi,m);
System.out.print("\n Encrypt. KEY : " + e);
System.out.print("\n Encrypted DATA : " + enc);

BigInteger Result=new BigInteger("123");
            Result=Decryption(e,enc,n,phi);

System.out.print("\n Decrypt. KEY : " + d);
System.out.print("\n Decrypted DATA : " + Result.toString());
        }
catch(Exception exception)
        {
System.out.print("\n Exception In Main: " + exception);
        }
    }
```

Save above program with program name as RSA.java (In java class name should be sam ; program name).

Scanned by CamScanner

Output

```
C:\Program Files (x86)\Java\jdk1.7.0_25\bin>javac RSA
It will display note but students can run it

C:\Program Files (x86)\Java\jdk1.7.0_25\bin> Java RSA

Enter First Prime No : 13
Enter Second Prime No : 11

Please Enter Message Between ( 0 to 32):13

Encrypt. KEY : 7
Encrypted DATA : 117
Decrypt. KEY : 103
Decrypted DATA : 13
```

Try out another example:-

```
C:\programfiles\jdk1.6\bin>java RSA

Enter First Prime No : 3
Enter Second Prime No : 7

Please Enter Message Between ( 0 to 32):11

Encrypt. KEY : 5
Encrypted DATA : 2
Decrypt. KEY : 5
Decrypted DATA : 11
```

Scanned by CamScanner

➢ Experiment 2 : Implementation of Diffie Hellman key exchange algorithm.

Aim : Our aim is write a program in Java to implement Diffie Hellman key exchange algorithm.

Objective

Diffie Hellman algorithm is used to generate same (symmetric) private cryptographic key at sender as well as receiver end so that there is no need to transfer this key from sender to receiver. Remember that Diffie Hellman algorithm is used only for key agreement not for encryption or decryption of message. If sender and receiver want to communicate with each other they first agree on the same key generated by Diffie Hellman Algorithm later on they can use this key for encryption or decryption.

Solution :

Following the important steps of Diffie Hellman algorithm :

1. The first step is that if Ramesh wants to communicate with Suresh they must agree on two large prime numbers p and q.
2. Ramesh selects another secret large random integer number a, and calculate R such that

$$R = q^a \bmod P$$

3. Ramesh sends this R to suresh.
4. Suresh independently selects another secret large random integer number b, and calculate S such that.

$$S = q^b \bmod P$$

5. Suresh sends the number S to Ramesh.
6. Now Ramesh is calculating his secret key by using $R_K = s^q \bmod P$
7. Suresh is calculating his secret key S_K by using

$$S_K = R^b \bmod P$$

8. If $R_K = S_K$ then Ramesh and Suresh can agree for future communication called as key agreement algorithm.
9. We have $R_K = S_K = K$ hence proved. (K is called symmetric key).

```
import java.util.*;
import java.math.BigInteger;
public class DiffieHellman {
    final static BigInteger one = new BigInteger("1");
```

Scanned by CamScanner

```
public static void main(String args[]) {
    Scanner stdin = new Scanner(System.in);
    BigInteger p;

    System.out.println("Enter the first prime number p of your choice");
    String ans = stdin.next();
    p = getNextPrime(ans);
    System.out.println("Enter another prime number q");
    BigInteger g = new BigInteger(stdin.next());
    System.out.println("Ramesh: select your secret number a.");
    BigInteger a = new BigInteger(stdin.next());
    BigIntegerresulta = g.modPow(a,p);
    System.out.println("Ramesh can sends the value of R to Suresh "+resulta+".");
    System.out.println("Suresh select your your secret number b");
    BigInteger b = new BigInteger(stdin.next());
    BigIntegerresultb = g.modPow(b,p);
    System.out.println("Suresh now sends value of S to Ramesh "+resultb+".");
    // Now calculate secret key of Ramesh & Suresh Rk&Sk
    BigIntegerKeyACalculates = resultb.modPow(a,p);
    BigIntegerKeyBCalculates = resulta.modPow(b,p);

    System.out.println("Ramesh takes "+resultb+" raise to mod p "+a+" mod "+p);
    System.out.println("The Secret key Rk  Ramesh calculates is "+KeyACalculates+".");
    System.out.println("Suresh takes "+resulta+" raises to mod p "+b+" mod "+p);
    System.out.println("The secret Key Sk Suresh calculates is "+KeyBCalculates+".");
}

    public static BigIntegergetNextPrime(String ans) {

    BigInteger test = new BigInteger(ans);
    while (!test.isProbablePrime(99))
    test = test.add(one);
    return test;
  }
}
```

Scanned by CamScanner

Save above program by using DiffieHellman.java in JDK bin directory or set path before compilation. Now compile the program using javac DiffieHellman.java and run using java DiffieHellman

Output

```
C:\Program Files (x86)\Java\jdk1.7.0_25\bin>java DiffieHellman
Enter the first prime number p of your choice17
Enter another prime number in q7
Ramesh: select your secret number a.5
Ramesh can sends the value of R to Suresh 11.
Suresh select youryour secret number b3
Suresh now sends value of S to Ramesh 3.
Ramesh takes 3 raise to the power 5 mod 17
The Secret key Rk  Ramesh calculates is 5.
Suresh takes 11 raises to the power 3 mod 17
The secret Key Sk Suresh calculates is 5.

C:\Program Files (x86)\Java\jdk1.7.0_25\bin>java DiffieHellman
Enter the first prime number p of your choice17
Enter another prime number in q11
Ramesh: select your secret number a.5
Ramesh can sends the value of R to Suresh 10.
Suresh select youryour secret number b3
Suresh now sends value of S to Ramesh 5.
Ramesh takes 5 raise to the power 5 mod 17
The Secret key Rk  Ramesh calculates is 14.
Suresh takes 10 raises to the power 3 mod 17
The secret Key Sk Suresh calculates is 14.
```

Scanned by CamScanner

Experiment 3 : For varying message sizes, test integrity of message using MD-5, SHA-1, and analyse the performance of the two protocols. Use crypt APIs.

Experiment 3(a) : Write program in Java to implement MD5 algorithm for key generation and cipher verification.

Aim : Our aim is to write java program to implement MD5 algorithm for key generation and cipher verification.

Objective

It was developed by Ron Rivest. This algorithm takes an input of arbitrary length and 128 - bit message digest is produced. The input message is produced in 512 - bit blocks. This algorithm takes an input of arbitrary length and 128 - bit message digest is produced. The input message is produced in 512 - bit blocks. Following steps explains the procedure of MD5.

Solution :

Refer Chapter 6 for complete steps of MD5 algorithm.

Following Program Demonstrate the MD5 algorithm in details :

```
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.PrintStream;
import java.io.UnsupportedEncodingException;

public class Md5
{
private static final int BUFFER_SIZE = 4096;

private static final int S11 = 7;
private static final int S12 = 12;
private static final int S13 = 17;
private static final int S14 = 22;
private static final int S21 = 5;
private static final int S22 = 9;
```

Scanned by CamScanner

```
private static final int S23 = 14;
private static final int S24 = 20;
private static final int S31 = 4;
private static final int S32 = 11;
private static final int S33 = 16;
private static final int S34 = 23;
private static final int S41 = 6;
private static final int S42 = 10;
private static final int S43 = 15;
private static final int S44 = 21;

private static byte padding[] =
{ (byte) 0x80, (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
    (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
    (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
    (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
    (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
    (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
    (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
    (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
    (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0, (byte) 0,
    (byte) 0, (byte) 0, (byte) 0, (byte) 0 };

private InputStream in = null;
private booleanstring p = false;
private int     state[] = null;
private long    count = 0;
private byte    buffer[] = null;
private byte    digest[] = null;

private static String stringify(byte buf[])
  {
StringBuffersb = new StringBuffer(2 * buf.length);
```

Scanned by CamScanner

```
for (int i = 0; i <buf.length; i++)
    {
int h = (buf[i] & 0xf0) >> 4;
int l = (buf[i] & 0x0f);

sb.append(new Character((char) ((h > 9) ? 'a' + h - 10 : '0' + h)));
sb.append(new Character((char) ((l > 9) ? 'a' + l - 10 : '0' + l)));
    }
return sb.toString();
  }


private final int F(int x, int y, int z)
  {
return ((x & y) | ((~x) & z));
  }

private final int G(int x, int y, int z)
  {
return ((x & z) | (y & (~z)));
  }

private final int H(int x, int y, int z)
  {
return (x ^ y ^ z);
  }

private final int I(int x, int y, int z)
  {
return (y ^ (x | (~z)));
  }

private final introtate_left(int x, int n)
```

Scanned by CamScanner

```
  {
return ((x << n) | (x >>> (32 - n)));
  }

private final int FF(int a, int b, int c, int d, int x, int s, int ac)
  {
    a += (F(b, c, d) + x + ac);
    a = rotate_left(a, s);
    a += b;
return a;
  }

private final int GG(int a, int b, int c, int d, int x, int s, int ac)
  {
    a += (G(b, c, d) + x + ac);
    a = rotate_left(a, s);
    a += b;
return a;
  }

private final int HH(int a, int b, int c, int d, int x, int s, int ac)
  {
    a += (H(b, c, d) + x + ac);
    a = rotate_left(a, s);
    a += b;
return a;
  }

private final int II(int a, int b, int c, int d, int x, int s, int ac)
  {
    a += (I(b, c, d) + x + ac);
    a = rotate_left(a, s);
    a += b;
return a;
```

Scanned by CamScanner

```
}
private final void decode(int output[], byte input[], int off, intlen)
{
int i = 0;
int j = 0;

for (; j <len; i++, j += 4)
    {
output[i] = (((int) (input[off + j] & 0xff))
            | (((int) (input[off + j + 1] & 0xff)) << 8)
            | (((int) (input[off + j + 2] & 0xff)) << 16)
            | (((int) (input[off + j + 3] & 0xff)) << 24));
    }
}

private final void transform(byte block[], int offset)
{
int a = state[0];
int b = state[1];
int c = state[2];
int d = state[3];
int x[] = new int[16];

decode(x, block, offset, 64);

  /* Round 1 */
  a = FF(a, b, c, d, x[ 0], S11, 0xd76aa478); /* 1 */
  d = FF(d, a, b, c, x[ 1], S12, 0xe8c7b756); /* 2 */
  c = FF(c, d, a, b, x[ 2], S13, 0x242070db); /* 3 */
  b = FF(b, c, d, a, x[ 3], S14, 0xc1bdceee); /* 4 */
  a = FF(a, b, c, d, x[ 4], S11, 0xf57c0faf); /* 5 */
  d = FF(d, a, b, c, x[ 5], S12, 0x4787c62a); /* 6 */
  c = FF(c, d, a, b, x[ 6], S13, 0xa8304613); /* 7 */
  b = FF(b, c, d, a, x[ 7], S14, 0xfd469501); /* 8 */
```

Scanned by CamScanner

```
a = FF(a, b, c, d, x[ 8], S11, 0x698098d8); /* 9 */
d = FF(d, a, b, c, x[ 9], S12, 0x8b44f7af); /* 10 */
c = FF(c, d, a, b, x[10], S13, 0xffff5bb1); /* 11 */
b = FF(b, c, d, a, x[11], S14, 0x895cd7be); /* 12 */
a = FF(a, b, c, d, x[12], S11, 0x6b901122); /* 13 */
d = FF(d, a, b, c, x[13], S12, 0xfd987193); /* 14 */
c = FF(c, d, a, b, x[14], S13, 0xa679438e); /* 15 */
b = FF(b, c, d, a, x[15], S14, 0x49b40821); /* 16 */

/* Round 2 */
a = GG(a, b, c, d, x[ 1], S21, 0xf61e2562); /* 17 */
d = GG(d, a, b, c, x[ 6], S22, 0xc040b340); /* 18 */
c = GG(c, d, a, b, x[11], S23, 0x265e5a51); /* 19 */
b = GG(b, c, d, a, x[ 0], S24, 0xe9b6c7aa); /* 20 */
a = GG(a, b, c, d, x[ 5], S21, 0xd62f105d); /* 21 */
d = GG(d, a, b, c, x[10], S22, 0x2441453); /* 22 */
c = GG(c, d, a, b, x[15], S23, 0xd8a1e681); /* 23 */
b = GG(b, c, d, a, x[ 4], S24, 0xe7d3fbc8); /* 24 */
a = GG(a, b, c, d, x[ 9], S21, 0x21e1cde6); /* 25 */
d = GG(d, a, b, c, x[14], S22, 0xc33707d6); /* 26 */
c = GG(c, d, a, b, x[ 3], S23, 0xf4d50d87); /* 27 */
b = GG(b, c, d, a, x[ 8], S24, 0x455a14ed); /* 28 */
a = GG(a, b, c, d, x[13], S21, 0xa9e3e905); /* 29 */
d = GG(d, a, b, c, x[ 2], S22, 0xfcefa3f8); /* 30 */
c = GG(c, d, a, b, x[ 7], S23, 0x676f02d9); /* 31 */
b = GG(b, c, d, a, x[12], S24, 0x8d2a4c8a); /* 32 */

/* Round 3 */
a = HH(a, b, c, d, x[ 5], S31, 0xfffa3942); /* 33 */
d = HH(d, a, b, c, x[ 8], S32, 0x8771f681); /* 34 */
c = HH(c, d, a, b, x[11], S33, 0x6d9d6122); /* 35 */
b = HH(b, c, d, a, x[14], S34, 0xfde5380c); /* 36 */
a = HH(a, b, c, d, x[ 1], S31, 0xa4beea44); /* 37 */
d = HH(d, a, b, c, x[ 4], S32, 0x4bdecfa9); /* 38 */
```

Scanned by CamScanner

```
c = HH(c, d, a, b, x[ 7], S33, 0xf6bb4b60); /* 39 */
b = HH(b, c, d, a, x[10], S34, 0xbebfbc70); /* 40 */
a = HH(a, b, c, d, x[13], S31, 0x289b7ec6); /* 41 */
d = HH(d, a, b, c, x[ 0], S32, 0xeaa127fa); /* 42 */
c = HH(c, d, a, b, x[ 3], S33, 0xd4ef3085); /* 43 */
b = HH(b, c, d, a, x[ 6], S34, 0x4881d05); /* 44 */
a = HH(a, b, c, d, x[ 9], S31, 0xd9d4d039); /* 45 */
d = HH(d, a, b, c, x[12], S32, 0xe6db99e5); /* 46 */
c = HH(c, d, a, b, x[15], S33, 0x1fa27cf8); /* 47 */
b = HH(b, c, d, a, x[ 2], S34, 0xc4ac5665); /* 48 */

/* Round 4 */
a = II(a, b, c, d, x[ 0], S41, 0xf4292244); /* 49 */
d = II(d, a, b, c, x[ 7], S42, 0x432aff97); /* 50 */
c = II(c, d, a, b, x[14], S43, 0xab9423a7); /* 51 */
b = II(b, c, d, a, x[ 5], S44, 0xfc93a039); /* 52 */
a = II(a, b, c, d, x[12], S41, 0x655b59c3); /* 53 */
d = II(d, a, b, c, x[ 3], S42, 0x8f0ccc92); /* 54 */
c = II(c, d, a, b, x[10], S43, 0xffeff47d); /* 55 */
b = II(b, c, d, a, x[ 1], S44, 0x85845dd1); /* 56 */
a = II(a, b, c, d, x[ 8], S41, 0x6fa87e4f); /* 57 */
d = II(d, a, b, c, x[15], S42, 0xfe2ce6e0); /* 58 */
c = II(c, d, a, b, x[ 6], S43, 0xa3014314); /* 59 */
b = II(b, c, d, a, x[13], S44, 0x4e0811a1); /* 60 */
a = II(a, b, c, d, x[ 4], S41, 0xf7537e82); /* 61 */
d = II(d, a, b, c, x[11], S42, 0xbd3af235); /* 62 */
c = II(c, d, a, b, x[ 2], S43, 0x2ad7d2bb); /* 63 */
b = II(b, c, d, a, x[ 9], S44, 0xeb86d391); /* 64 */

state[0] += a;
state[1] += b;
state[2] += c;
state[3] += d;
}
```

Scanned by CamScanner

```
private final void update(byte input[], intlen)
  {
int index = ((int) (count >> 3)) & 0x3f;

count += (len<< 3);
int partLen = 64 - index;
int i = 0;

if (len>= partLen)
    {
System.arraycopy(input, 0, buffer, index, partLen);
transform(buffer, 0);
for (i = partLen; i + 63 <len; i += 64)
transform(input, i);
index = 0;
    }
else
    {
      i = 0;
    }
System.arraycopy(input, i, buffer, index, len - i);
  }

private byte[] end()
  {
byte bits[] = new byte[8];

for (int i = 0; i < 8; i++)
bits[i] = (byte) ((count >>> (i * 8)) & 0xff);
int index = ((int) (count >> 3)) & 0x3f;
intpadlen = (index < 56) ? (56 - index) : (120 - index);

update(padding, padlen);
```

Scanned by CamScanner

```
update(bits, 8);
return encode(state, 16);
 }

 // Encode the content.state array into 16 bytes array

private byte[] encode(int input[], intlen)
 {
byte output[] = new byte[len];
int i = 0;
int j = 0;

for (; j <len; i++, j += 4)
   {
output[j] = (byte) ((input[i]) & 0xff);
output[j + 1] = (byte) ((input[i] >> 8) & 0xff);
output[j + 2] = (byte) ((input[i] >> 16) & 0xff);
output[j + 3] = (byte) ((input[i] >> 24) & 0xff);
   }
return output;
 }

 /**
  * Get the digest for our input stream.
  * This method constructs the input stream digest, and return it, as a
  * a String, following the MD5 (rfc1321) algorithm,
  * @return An instance of String, giving the message digest.
  * @exception IOException Thrown if the digestifier was unable to read the
  *   input stream.
  */

public byte[] getDigest()
throwsIOException
 {
```

Scanned by CamScanner

```
byte buffer[] = new byte[BUFFER_SIZE];
int got = -1;

if (digest != null)
return digest;
while ((got = in.read(buffer)) > 0)
update(buffer, got);
this.digest = end();
return digest;
  }

  /**
   * Get the digest, for this string digestifier.
   * This method doesn't throw any IOException, since it knows that the
   * underlying stream ws built from a String.
   */

public byte[] processString()
  {
if (!stringp)
throw new RuntimeException (this.getClass().getName()
            + "[processString]"
            + " not a string.");
try
    {
returngetDigest();
    } catch (IOException ex)
    {
    }

throw new RuntimeException (this.getClass().getName()
        + "[processString]"
        + ": implementation error.");
}
```

Scanned by CamScanner

```
/* Get the digest, as a proper string.    */

public String getStringDigest()
 {
if (digest == null)
throw new RuntimeException (this.getClass().getName()
            + "[getStringDigest]"
            + ": called before processing.");
return stringify(digest);
 }

/**
 * Construct a digestifier for the given string.
 * @param input The string to be digestified.
 * @param encoding the encoding name used (such as UTF8)
 */

public Md5 (String input, String enc)
 {
byte bytes[] = null;

try
    {
bytes = input.getBytes(enc);
    } catch (UnsupportedEncodingException e)
    {
throw new RuntimeException("no " + enc + " encoding!!!");
    }
this.stringp = true;
    this.in = new ByteArrayInputStream (bytes);
this.state = new int[4];
this.buffer = new byte[64];
this.count = 0;
```

Scanned by CamScanner

```
state[0] = 0x67452301;
state[1] = 0xefcdab89;
state[2] = 0x98badcfe;
state[3] = 0x10325476;
  }

  /**
   * Construct a digestifier for the given string.
   * @param input The string to be digestified.
   */

public Md5 (String input)
  {
this(input, "UTF8");
  }

  /**
   * Construct a digestifier for the given input stream.
   * @param in The input stream to be digestified.
   */

public Md5 (InputStream in)
  {
this.stringp = false;
    this.in = in;
this.state = new int[4];
this.buffer = new byte[64];
this.count = 0;
state[0] = 0x67452301;
state[1] = 0xefcdab89;
state[2] = 0x98badcfe;
state[3] = 0x10325476;
  }
```

Scanned by CamScanner

```
ublic static void main(String args[])
throwsIOException
{
if(args.length != 1)
  {
System.out.println("Md5 <file>");
System.exit(1);
  }
  Md5 md5 = new Md5 (new FileInputStream(new File(args[0])));
byte b[] = md5.getDigest();

System.out.println(stringify(b));
 }
}
```

- Save above program with program name Md5.java into the JDK bin directory and compile it using javac Md5.java and execute using java Md5 b1.txt.
- To get correct output please create one text file (Give any name to that text file here we have created one text file b1.txt) as an input to MD5 algorithm.

Output

To get correct output please create one text file (Give any name to that text file here I have created one text file b1.txt text provide in b1.txt is `Cryptography & System Security`) as an input to MD5 algorithm .

```
C:\Program Files (x86)\Java\jdk1.7.0_25\bin>javac Md5.java

C:\Program Files (x86)\Java\jdk1.7.0_25\bin>java Md5 b1.txt
036302dca5126270864f53563f66198a
```

MD5 encrypt the text `Cryptography & System Security` & produce the digest as 036302dca5126270864f53563f66198a

```
C:\Program Files (x86)\Java\jdk1.7.0_25\bin>
```

Scanned by CamScanner

➢ Experiment 3(b): Write a program in Java to implement SHA-1 algorithm using Libraries (API).

Objective

Aim of this assignment is to implement Secure Hash Algorithm – 1 using Librarie (API).

Refer Chapter 6 for theory of SHA-I

Solution :

```
import javax.crypto.*;
import java.io.*;
import java.security.*;
public class SHA
{
public static String Getdata()
    {
        String Message=null;
try
        {
DataInputStream in=new DataInputStream(System.in);
System.out.print("\n Please Enter One Message : ");
            Message=in.readLine();
        }
catch(Exception exception)
        {
        }
return Message;
    }
public static String Compute(String Message,Key key)
    {
byte[] digest=new byte[512];
        String store=null;
try
        {
MessageDigest digest1=MessageDigest.getInstance("SHA-1");
digest1.update(Message.getBytes());
```

Scanned by CamScanner

```
byte[] value=digest1.digest();
            Mac mac=Mac.getInstance("HmacMD5");
mac.init(key);
mac.update(value);
digest=mac.doFinal();
StringBuffer buffer=new StringBuffer();
for(int i=0;i<digest.length;i++)
          {
int value1=digest[i] & 0xff;
if(value1<16)
          {
buffer.append('0');
          }
buffer.append(Integer.toHexString(value1));
          }
store=buffer.toString();
      }
catch(Exception exception)
      {
System.out.print("\n Exception : " + exception);
      }
return store;
      }
public static void main(String args[])
      {
        String Message,holder1,holder2;
try
      {
KeyGenerator generator=KeyGenerator.getInstance("AES");
generator.init(128);
SecretKey key=generator.generateKey();
          Message=Getdata();
          holder1=Compute(Message,key);
System.out.print("\n Message Digest1 : [ " + holder1 + " ]");
```

Scanned by CamScanner

```
            Message=Getdata();
            holder2=Compute(Message,key);
System.out.print("\n Message Digest2 : [ " + holder2 + " ]");
if(holder1.equals(holder2))
            {
System.out.print("\n Message Is Same ..");
            }
else
            {
System.out.print("\n Message Is Not Same ..");
            }
}
catch(Exception exception)
        {
System.out.print("\n Exception In Main : " + exception);
        }
    }
}
```

Save above program with the program name SHA.java, Compile it using javac SHA.java and run it using java SHA

Output

```
C:\Program Files (x86)\Java\jdk1.7.0_25\bin>javac SHA.java
It will two note but students can run it

C:\Program Files (x86)\Java\jdk1.7.0_25\bin>java SHA

Please Enter One Message : Cyber Crime & Security

Message Digest1 : [e1b061ca3fc6017ob18799da1ffb006f]

Please Enter One Message : Cyber Crime and Security

Message Digest2 : [b346499838f2ad69e5473d507ae90a72]
```

Scanned by CamScanner

c SHA.java


```
Message Is Not Same..

Compile & Run it again with same message

:\Program Files (x86)\Java\jdk1.7.0_25\bin>javac SHA.java

:\Program Files (x86)\Java\jdk1.7.0_25\bin>java SHA

Please Enter One Message : Cyber Crime & Security

Message Digest1 : [e1b061ca3fc6017ob18799da1ffb006f]

Please Enter One Message : Cyber Crime & Security

Message Digest2 : [e1b061ca3fc6017ob18799da1ffb006f]

Message Is Same ..
```

Keep in mind that for every old/new message SHA-1 always generates new message digest

- **Experiment 4 : Study of packet sniffer tools - wireshark**
- **Experiment 4(a) : Download and install wireshark and capture icmp, tcp, and http packets in promiscuous mode.**

Aim : To install a packet analyzer & capture the network traffic. Make sure that packet analyzer only captures the live packets from different networks, it cannot detect whether the packet is malicious or not. This detection was done by using intrusion detection system (IDS), will discuss IDS in experiment number 9. The Students can select any packet analyzer tool here I have selected Wireshark as it is freely available on Internet.

Objective

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible. You could think of a network packet analyzer as a measuring device used to examine what's going on inside a network cable, just like a voltmeter is used by an electrician to examine what's going on inside

Scanned by CamScanner

an electric cable (but at a higher level, of course).In the past, such tools were either very expensive, proprietary, or both. However, with the advent of Wireshark, all that has changed. Wireshark is perhaps one of the best open source packet analyzers available today.

Solution :

A brief history of Wireshark

- In late 1997, Gerald Combs needed a tool for tracking down networking problems and wanted to learn more about networking, so he started writing Ethereal (the former name of the Wireshark project) as a way to solve both problems.
- Ethereal was initially released, after several pauses in development, in July 1998 as version 0.2.0.Within days, patches, bug reports, and words of encouragement started arriving, so Ethereal was on its way to success.
- Not long after that, Gilbert Ramirez saw its potential and contributed a low-level dissector to it.
- In October, 1998, Guy Harris of Network Appliance was looking for something better than tcp view, so he started applying patches and contributing dissectors to Ethereal.
- In late 1998, Richard Sharpe, who was giving TCP/IP courses, saw its potential on such courses, and started looking at it to see if it supported the protocols he needed. While it didn't at that point, new protocols could be easily added. So he started contributing dissectors and contributing patches.
- The list of people who have contributed to the project has become very long since then, and almost all of them started with a protocol that they needed that Wireshark or Ethereal did not already handle. So they copied an existing dissector and contributed the code back to the team.
- In 2006 the project moved house and re-emerged under a new name : Wireshark.
- In 2008, after ten years of development, Wireshark finally arrived at version 1.0. This release was the first deemed complete, with the minimum features implemented. Its release coincided with the first Wireshark Developer and User Conference, called SharkFest.

Some intended purposes of Wireshark

- Network administrators use it to **troubleshoot network problems.**
- Network security engineers use it to **examine security problems.**
- Developers use it to **debug protocol implementations.**

Scanned by CamScanner

People use it to **learn network protocol** internals.

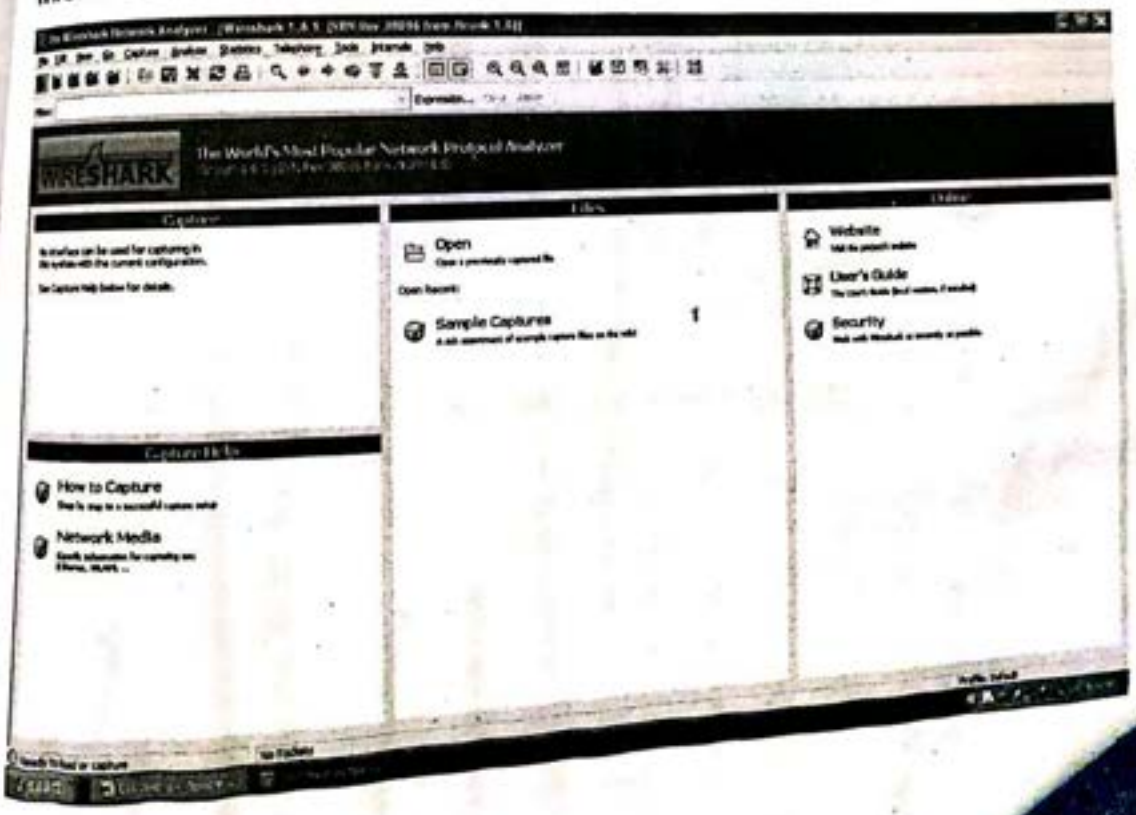
Beside these examples, Wireshark can be helpful in many other situations too.

Features

The following are some of the many features Wireshark provides :

- Available for **UNIX** and **Windows**.
- **Capture** live packet data from a network interface.
- Display packets with very detailed protocol information.
- **Open and Save** packet data captured.
- **Import and Export** packet data from and to a lot of other capture programs.
- **Filter packets** on many criteria.
- **Search** for packets on many criteria.
- **Colorize** packet display based on filters.
- Create various **statistics**.

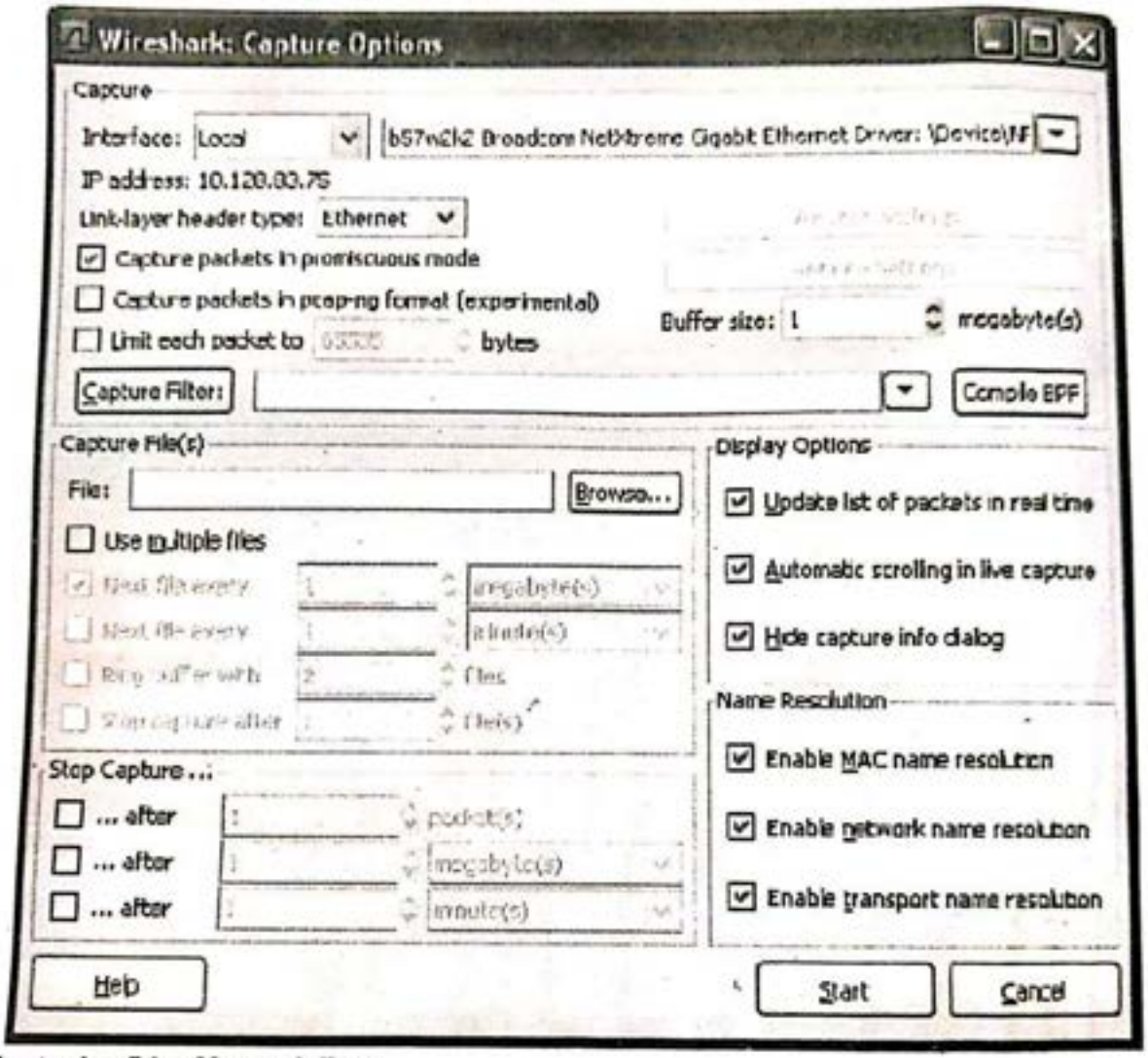
Wireshark Main Window



Scanned by CamScanner

Wireshark Main Window

Capture option dialog box

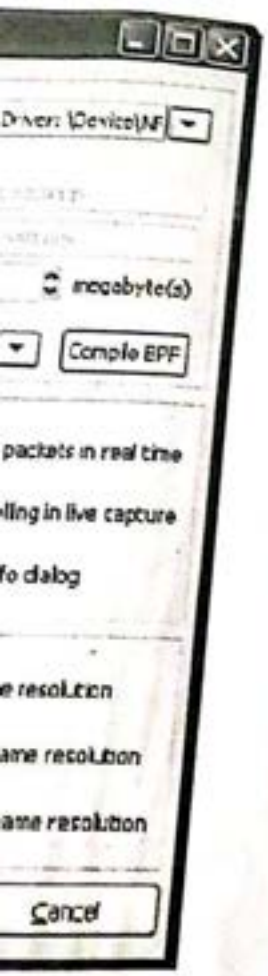


Capturing Live Network Data

Capturing live network data is one of the major features of Wireshark. The Wireshark capture engine provides the following features :

- Capture from different kinds of network hardware (Ethernet, Token Ring, ATM, ...).
- Stop the capture on different triggers like: amount of captured data, captured time, captured number of packets.
- Simultaneously show decoded packets while Wireshark keeps on capturing.
- Filter packets, reducing the amount of data to be captured.

Scanned by CamScanner



The Wireshark

TM, ...).

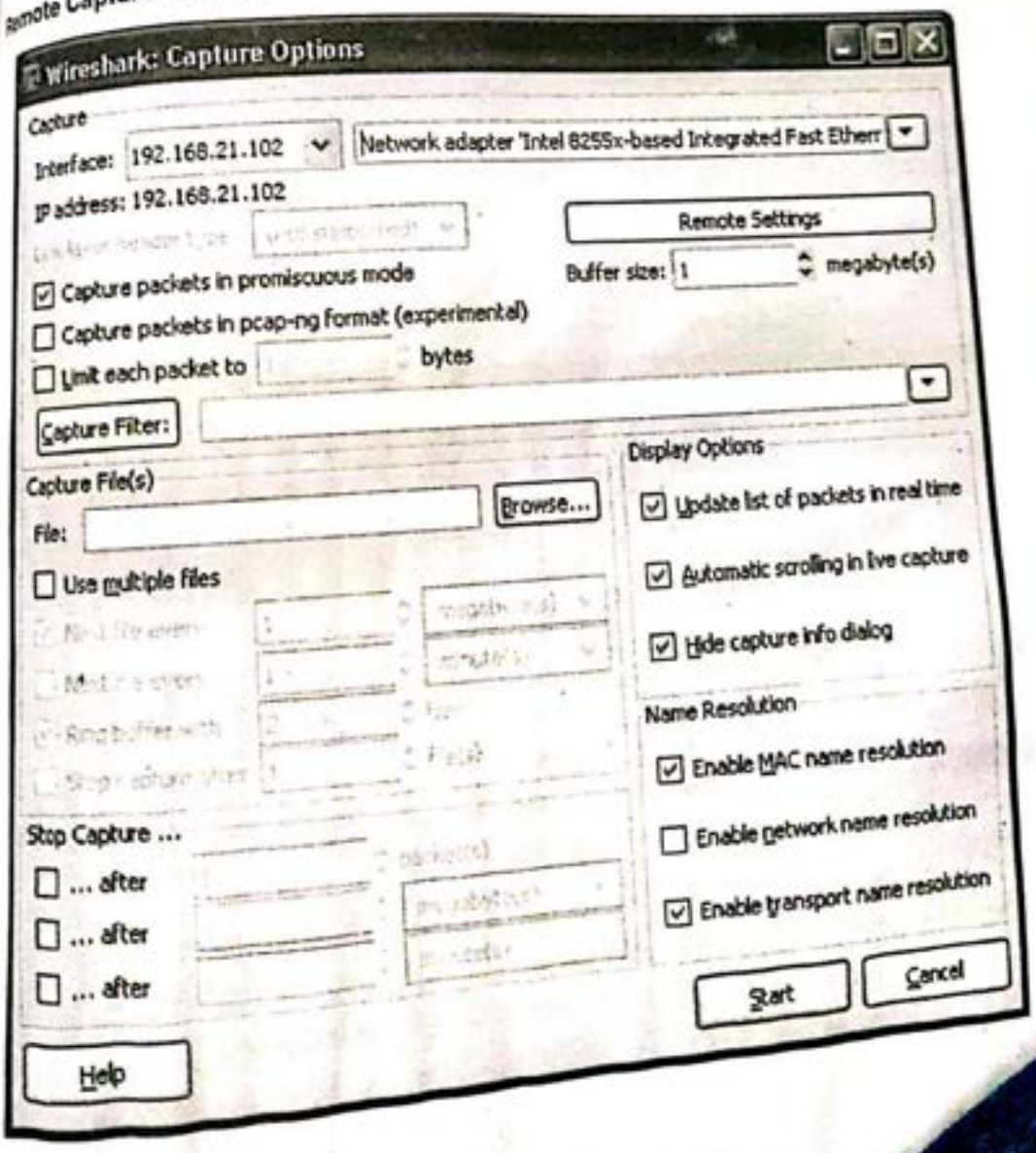
aptured time,

Capturing into multiple files while doing a long term capture, and in addition the option to form a ring buffer of these files, keeping only the last x files, useful for a "very long term" capture.

The capture engine still lacks the following features

- Simultaneous capturing from multiple network interfaces (however, you can start multiple instances of Wireshark and merge capture files later).
- Stop capturing (or doing some other action), depending on the captured data.

Remote Capture Interface



Scanned by CamScanner

Live capture from many different network media

Wireshark can capture traffic from many different network media types - and despite its name -including wireless LAN as well. Which media types are supported, depends on many things like the operating system you are using.

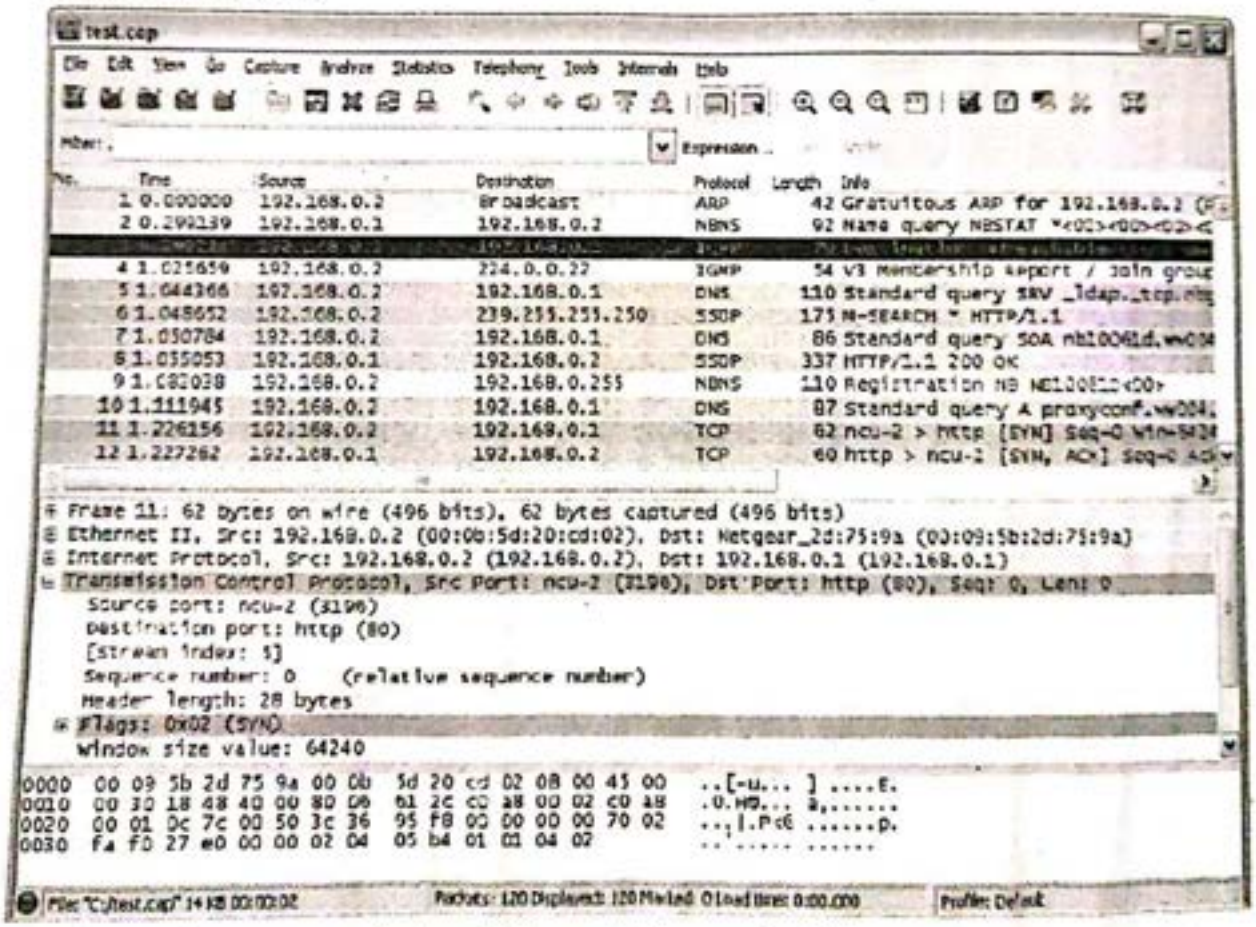
Import files from many other capture programs

Wireshark can open packets captured from a large number of other capture programs. For a list of input formats.

Export files for many other capture programs

Wireshark can save packets captured in a large number of formats of other capture programs. For a list of output formats.

Capturing Packets



Wireshark captures packets and allows you to examine their content.

Scanned by CamScanner

Open Source Software

- Wireshark is an open source software project, and is released under the GNU General Public License (GPL). You can freely use Wireshark on any number of computers you like, without worrying about license keys or fees or such. In addition, all source code is freely available under the GPL.
- Because of that, it is very easy for people to add new protocols to Wireshark, either as plugins, or built into the source, and they often do.

Development and maintenance of Wireshark

- Wireshark was initially developed by Gerald Combs. Ongoing development and maintenance of Wireshark is handled by the Wireshark team, a loose group of individuals who fix bugs and provide new functionality.
- There have also been a large number of people who have contributed protocol dissectors to Wireshark, and it is expected that this will continue.
- You can find a list of the people who have contributed code to Wireshark by checking the about dialog box of Wireshark, or at the authors page on the Wireshark web site.
- Wireshark is an open source software project, and is released under the GNU General Public License (GPL). All source code is freely available under the GPL. You are welcome to modify Wireshark to suit your own needs, and it would be appreciated if you contribute your improvements back to the Wireshark team.
- You gain three benefits by contributing your improvements back to the community:
- Other people who find your contributions useful will appreciate them, and you will know that you have helped people in the same way that the developers of Wireshark have helped people.
- The developers of Wireshark might improve your changes even more, as there's always room for improvement. Or they may implement some advanced things on top of your code, which can be useful for yourself too.
- The maintainers and developers of Wireshark will maintain your code as well, fixing it when API changes or other changes are made, and generally keeping it in tune with what is happening with Wireshark. So if Wireshark is updated (which is done often), you can get a new Wireshark version from the website and your changes will already be included without any effort for you.

Related command lineTools

- **Tshark :** Terminal-based Wireshark.

Scanned by CamScanner

- **Tcpdump :** Capturing with tcpdump for viewing with Wireshark.
- **Dumpcap :** Capturing with dumpcap for viewing with Wireshark.
- **Capinfos :** Print information about capture files.
- **Rawshark :** Dump and analyze network traffic.
- **Editcap :** Edit capture files.
- **Mergecap :** Merging multiple capture files into one.
- **Text2pcap :** Converting ASCII hexdumps to network captures.

Keyboard Navigation

Accelerator	Description
Tab, Shift + Tab	Move between screen elements, e.g. from the toolbars to the packet list to the packet detail.
Down	Move to the next packet or detail item.
Up	Move to the previous packet or detail item.
Ctrl + Down, F8	Move to the next packet, even if the packet list isn't focused.
Ctrl + Up, F7	Move to the previous packet, even if the packet list isn't focused.
Ctrl+.	Move to the next packet of the conversation (TCP, UDP or IP)
Ctrl+,	Move to the previous packet of the conversation (TCP, UDP or IP)
Left	In the packet detail, closes the selected tree item. If it's already closed, jumps to the parent node.
Right	In the packet detail, opens the selected tree item.
Shift + Right	In the packet detail, opens the selected tree item and all of its subtrees.
Ctrl + Right	In the packet detail, opens all tree items.
Ctrl + Left	In the packet detail, closes all tree items.
Backspace	In the packet detail, jumps to the parent node.
Return, Enter	In the packet detail, toggles the selected tree item.

Conclusion

Thus we have studied a network packet analyzer i.e the Wireshark.

Scanned by CamScanner

Experiment 4(b): Explore how the packets can be traced based on different filters.

Aim : Aim of this experiment is to download & install freely available vulnerability tool to scan the entire network and find any vulnerability in the network or not.

Solution :

Nessus

In computer, **Nessus** is a proprietary comprehensive vulnerability scanning program. It is free of charge for personal use in a non-enterprise environment. Its goal is to detect potential vulner abilities on the tested systems.

For example

- Vulnerabilities that allow a remote cracker to control or access sensitive data on a system.
- Misconfiguration (e.g. open mail relay, missing patches, etc).
- Default passwords, a few common passwords, and blank/absent passwords on some system accounts. Nessus can also call Hydra (an external tool) to launch a dictionary attack.
- Denials of service against the TCP/IP stack by using mangled packets

On UNIX(including Mac OS X), it consists of **nessusd**, the Nessus daemon, which does the scanning, and *nessus*, the client, which controls scans and presents the vulnerability results to the user.

According to surveys done by sectools.org, Nessus is the world's most popular vulnerability scanner, taking first place in the 2000, 2003, and 2006 security tools survey. Tenable estimates that it is used by over 75,000 organizations worldwide.

Nessus Is also :

- A "Free, Powerful, up-to-date, easy to use, remote security scanner".
- Open-Source, free to use, modify, etc.
- Vulnerability definitions, called plugins, are free as well.
- Easy is a matter of perspective.

Scanned by CamScanner

Operation of Nessus

- In typical operation, Nessus begins by doing a port scan with one of its four internal port scanners (or it can optionally use AmaporNmap) to determine which ports are open on the target and then tries various exploits on the open ports. The vulnerability tests, available as subscriptions, are written in **NASL** (Nessus Attack Scripting Language), a scripting language optimized for custom network interaction.
- Tenable Network Security produces several dozen new vulnerability checks (called plugins) each week, usually on a daily basis. These checks are available for free to the general public; commercial customers are not allowed to use this Home Feed any more. The Professional Feed (which is not free) also give access to support and additional scripts (audit and compliance tests...).
- Optionally, the results of the scan can be reported in various formats, such as plain text, XML, HTML and LaTeX. The results can also be saved in a knowledge base for debugging. On UNIX, scanning can be automated through the use of a command-line client. There exist many different commercial, free and open source tools for both UNIX and Windows to manage individual or distributed Nessus scanners. If the user chooses to do so (by disabling the option 'safe checks'), some of Nessus's vulnerability tests may try to cause vulnerable services or operating systems to crash. This lets a user test the resistance of a device before putting it in production. Nessus provides additional functionality beyond testing for known network vulnerabilities.
- For instance, it can use Windows credentials to examine patch levels on computers running the Windows operating system, and can perform password auditing using dictionary and brute force methods. Nessus 3 and later can also audit systems to make sure they have been configured per a specific policy, such as the NSA's guide for hardening Windows servers.

Brief History of Nessus

- The "Nessus" Project was started by Renaud Deraison in 1998 to provide to the **Internet** community a free remote security scanner. On October 5, 2005, Tenable Network Security, the company Renaud Deraison co-founded, changed Nessus 3 to a proprietary (closed source) license.

- The Nessus 3 eng
scanner for the ab
configuration stand
checks and patch
Nessus to perform
and many other ty
- In July of2008, T
users full access
The Nessus 2 eng
source projects ba
- Tenable Network
several times sin
and Windows sy
need for an agen
Nessus 4.0.0.

Scanned by CamScanner

The Nessus 3 engine is still free of charge, though Tenable charges $100/month per scanner for the ability to perform configuration audits for PCI, CIS, FDCC and other configuration standards, technical support, SCADA vulnerability audits, the latest network checks and patch audits, the ability to audit anti-virus configurations and the ability for Nessus to perform sensitive data searches to look for credit card, social security number and many other types of corporate data.

In July of2008, Tenable sent out a revision of the feed license which will allow home users full access to plugin feeds. A professional license is available for commercial use. The Nessus 2 engine and a minority of the plugins are still **GPL**, leading to **forked** open source projects based on Nessus like **OpenVAS** and **Porz-Wahn**.

Tenable Network Security has still maintained the Nessus 2 engine and has updated it several times since the release of Nessus 3.Nessus 3 is available for many different UNIX and Windows systems, offers patch auditing of UNIX and Windows hosts without the need for an agent and is 2-5 times faster than Nessus 2.On April 9, 2009, Tenable released Nessus 4.0.0.

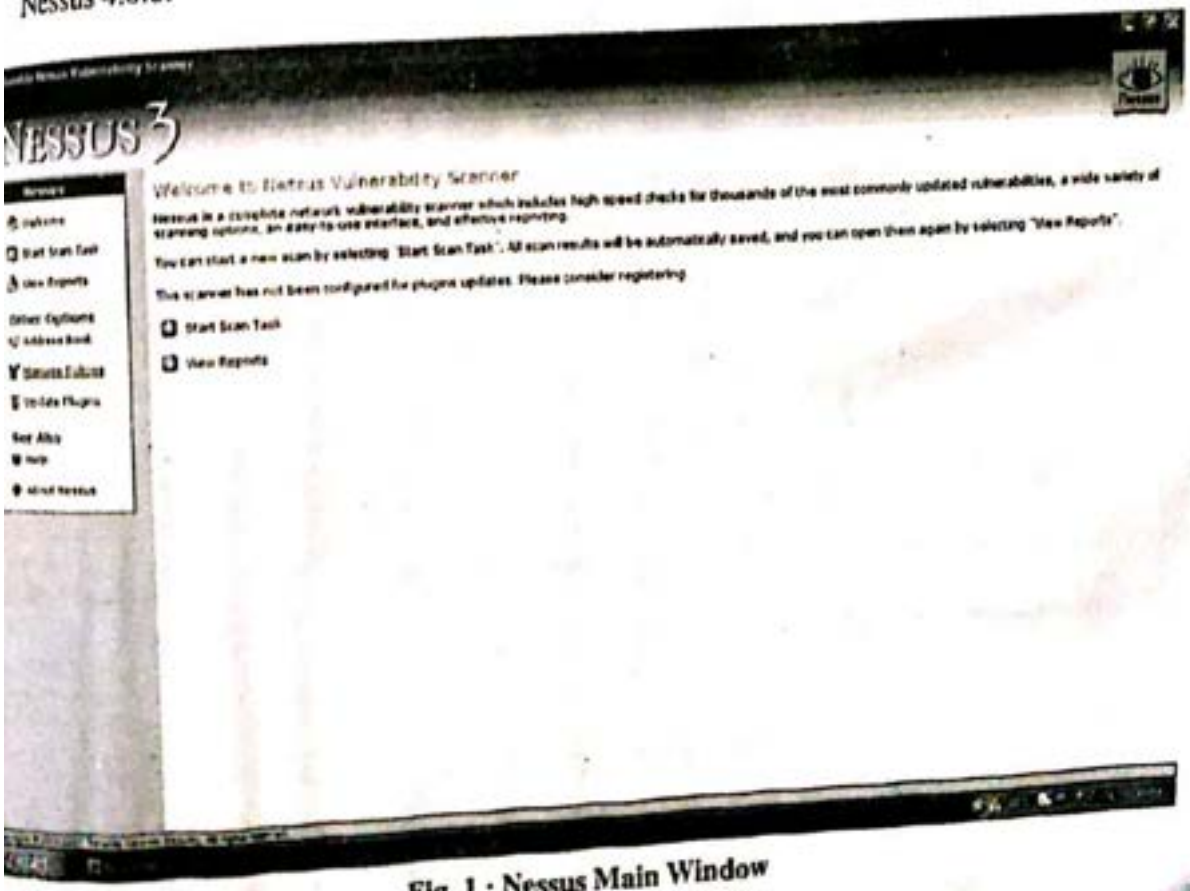


Fig. 1 : Nessus Main Window

Scanned by CamScanner

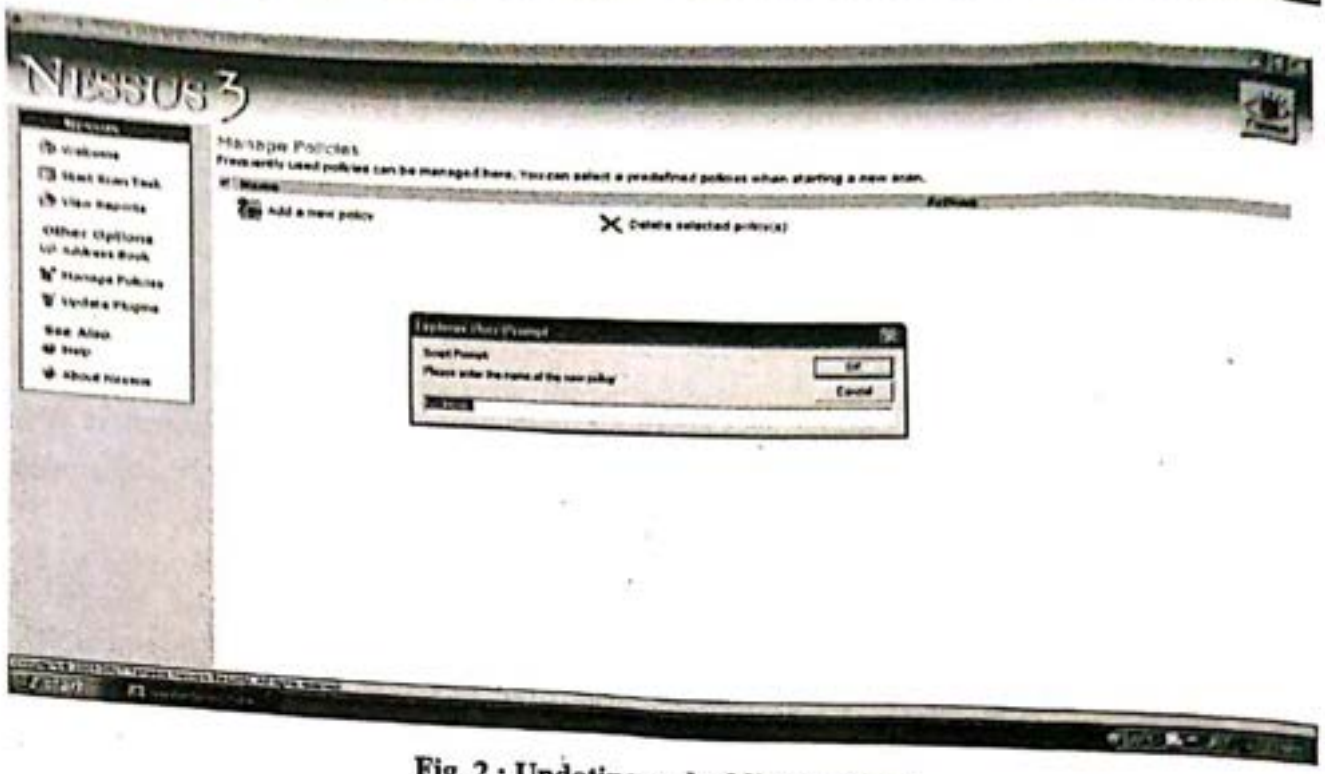


Fig. 2 : Updating and adding a policy

Nessus Features

- **Plugins :** Use its own scripting language(NASL) to define how it tests for vulnerabilities.
- **Client/Server architecture :** Client and server can be anywhere on the network
- **Protocol aware :** i.e. It will detect FTP running on port 31337
- **Application Aware :** Tests web servers running on the same port
- **Intelligent scanning :** Anonymous FTP Reports provide vulnerability listings and a good number of resolutions.
- Client/Server uses SSL to protect report results
- Much better about not crashing targets.

Nessus Architecture : Nessus Client and target systems

- Native Unix GTK Client (Linux, Solaris, and others).
- Windows Client (NessusWX).
- Windows Client is preferred, more report options, better interface.

Scanned by CamScanner

vulnerabilities.

twork

ngs and a good

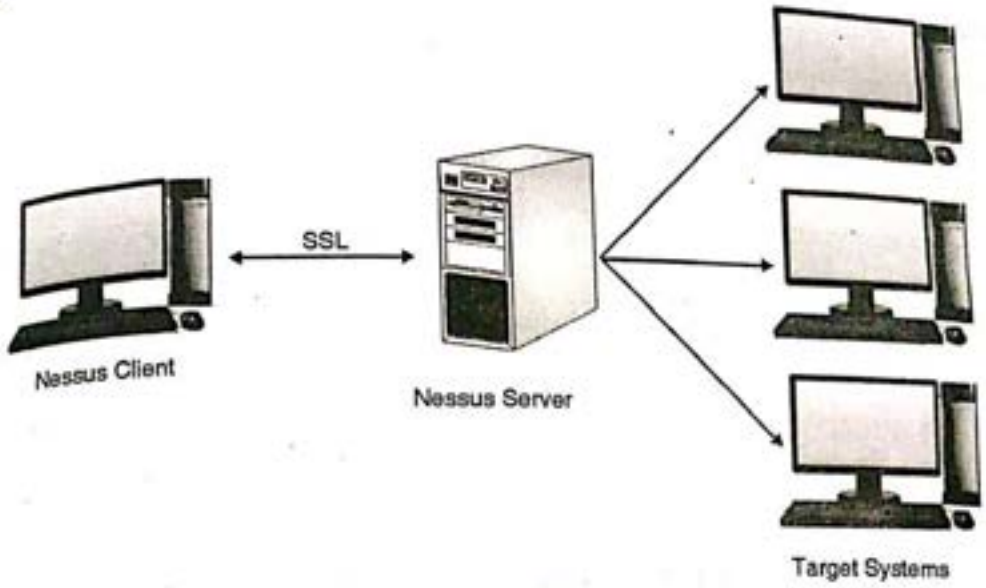


Fig. 3 : Nessus Architecture

Nessus Client on Windows

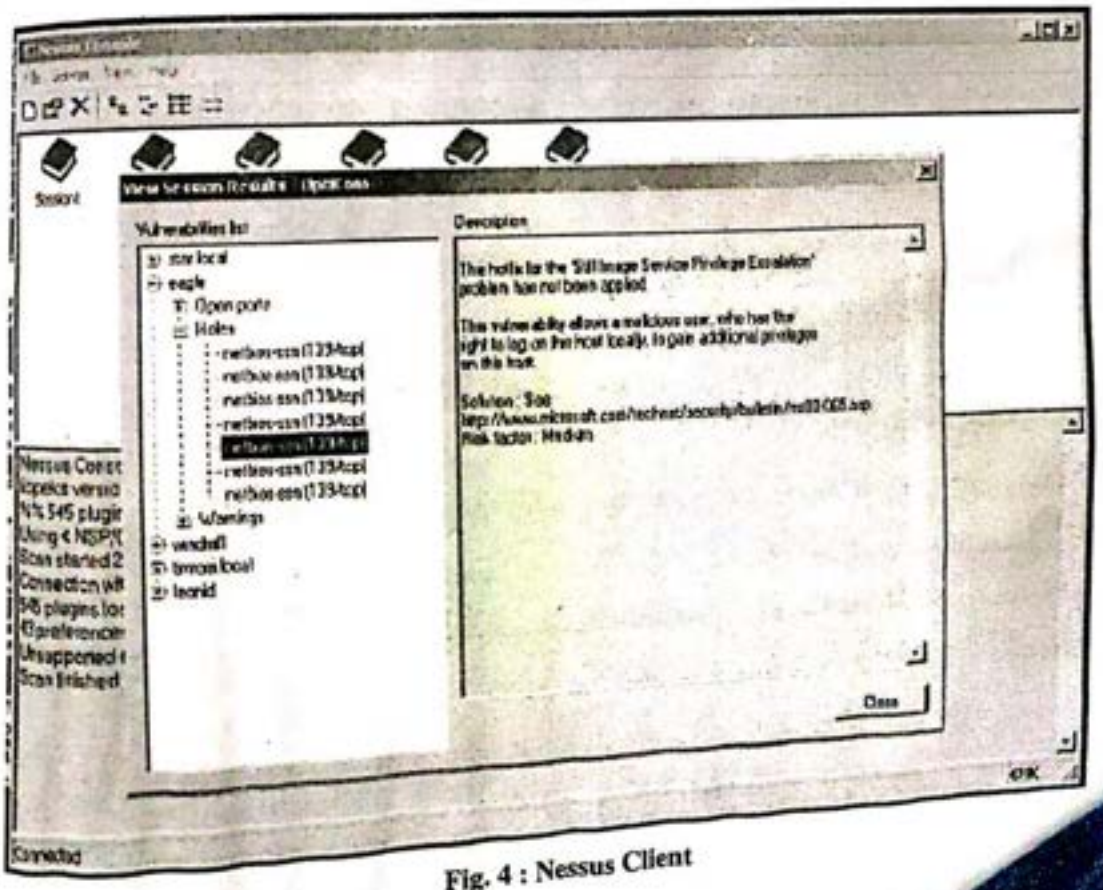


Fig. 4 : Nessus Client

Scanned by CamScanner

Nessus Server

- Runs on most Unixflavors (Unix, Linux, *BSD).
- I find it runs best on Linux, your mileage may vary.
- Performs all scanning functions, sends results back to client.
- Includes a plugin update facility.

Nessus Reports

- Numerous different formats.
- Problem – How to get the reports to the user securely.
- Answers include.
- Commercial Products.
- Write your own Perl or PHP application.

Scanning Methodologies

- Someone scans your system(s) and makes the report available to you.
- The end user requests a scan directly from the server, the machine is scanned, and report is sent automatically.
- When the user connects to the network the system is scanned automatically (Popular with wireless and VPN).
- Servers are scanned on a regular basis (maybe weekly) and results are compared.
- Network Perimeter is scanned on a regular basis.
- Which ones should I do.

Challenges – False Positives

- Must verify to some degree the vulnerabilities Nessus has found.
- This is time consuming and sometimes quite difficult.
- Nessus is getting better, but still a ways to go.

Challenges – Crash and Burn

- Nessus will crash systems, routers, firewalls, and any other devices on the network!
- Happens no matter how careful you are Monitor your configuration closely, test new plugins first.

Scanned by CamScanner

.

scanned, and report is

atically (Popular with

compared.

network!

osely, test new

- Prepare for the worst.

Challenges – What about the application

- Nessus does some application level vulnerability assessment.
- Tools from SPI Dynamics, EEye, and ISS are better.
- Make sure you have at least one other tool to test the application!

Challenges – Scan What ? When ?

- Getting permission to scan is half the battle.
- There is no guarantee that it will not crash the system.
- As you know, people don't like it when you find things wrong with their systems.

Challenges – How long will it take ?

- Depends.
- Number of hosts.
- Number of open ports.
- Number of services running on those ports.
- What kind of host (Windows, Unix, Mac).
- How many hosts have firewalls.
- Speed of the network.
- Other network traffic.
- How many vulnerabilities are found.
- If the host crashes after the first plugin or just before the last.

Challenges – How long does it usually take ?

- One host = A morning or afternoon.
- More than one host = 1 Day.
- Entire Class C subnet = 2-3 Days.
- Entire Class B = Weeks.

Conclusion

Thus we have studied the vulnerability too like Nessus for network Security.

Scanned by CamScanner

➢ **Experiment 5 : Download and install nmap. Use it with different options to scan open ports, perform OS fingerprinting, do a ping scan, tcp port scan, udp port scan, xmas scan etc.**

Aim : Aim of this experiment is to download & install port scanning tool to identify the number of computers on a network & to find the port open on one or more target computers.

Solution :

What is NMAP?

Nmap : "Network Mapper" It was developed by Fyodor.

- An open source tool for network exploration and security auditing.
- N-map uses raw IP packets in novel ways for information gathering

N-MAP features

Ping Sweeping : Identifying computers on a network.

Port Scanning: Enumerating the open Ports on one or more target computers.

OS Detection : Remotely determining the operating system and some hardware characteristics of network devices.

Ping Sweeping

- Ping Sweeping allows the hackers to automatically map out the entire target network and
- Pinpoint all alive systems within a particular range of IP addresses.

Operations on PING Sweeping

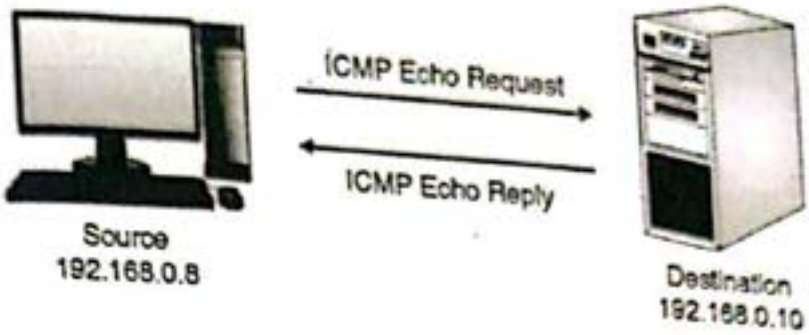
When Host is not Alive

- If the station isn't available on the network or a packet filter is preventing ICMP packets from passing, there will be no response to the echo frame.

Scanned by CamScanner

Host is Alive ?

A response from an active host will return an ICMP echo reply, unless the IP address is not available on the network or ICMP is filtered.



- Port Scanning is the process of connecting to TCP and UDP port for the purpose of finding which services and applications are open on the Target Machine.
- TCP establishes a connection by using what is called a Three way handshake. The TCP header contains one byte field for the flags.
- These flags include the following :
  - **ACK :** The receiver will send an Ack to acknowledge data.
  - **SYN :** Setup to begin communication on initial sequence number.
  - **FIN :** Inform the other host that the sender has no more data to send.
  - **RST :** Abort operation.
  - **PSH :** Force data delivery without waiting for buffers to fill.
  - **URG :** Indicate priority data.
- The port numbers are unique only within a computer system. Port numbers are 16-bit unsigned numbers. The port numbers are divided into three ranges: the Well Known Ports (0-1023), the Registered Ports (1024-49151), and the Dynamic and/or Private Ports (49152-65535).
- All the operating systems now honor the tradition of permitting only the super-user open the ports numbered 0 to 1023.
- Some are listed below :

echo 7/tcp	Echo
ftp-data 20/udp	File Transfer [Default Data]
ftp 21/tcp	File Transfer [Control]

Scanned by CamScanner

ssh 22/tcp SSH Remote Login Protocol

telnet 23/tcp Telnet

domain 53/udp Domain Name Server

www-http 80/tcp World Wide Web HTTP

- Nmap ("Network Mapper") is a free and open source utility for network exploration or security auditing.
- The FIVE port states recognized by Nmap such as :

  1. Closed 2. Filtered 3. UnFiltered
  4. Open-Filtered 5. Closed-Filter

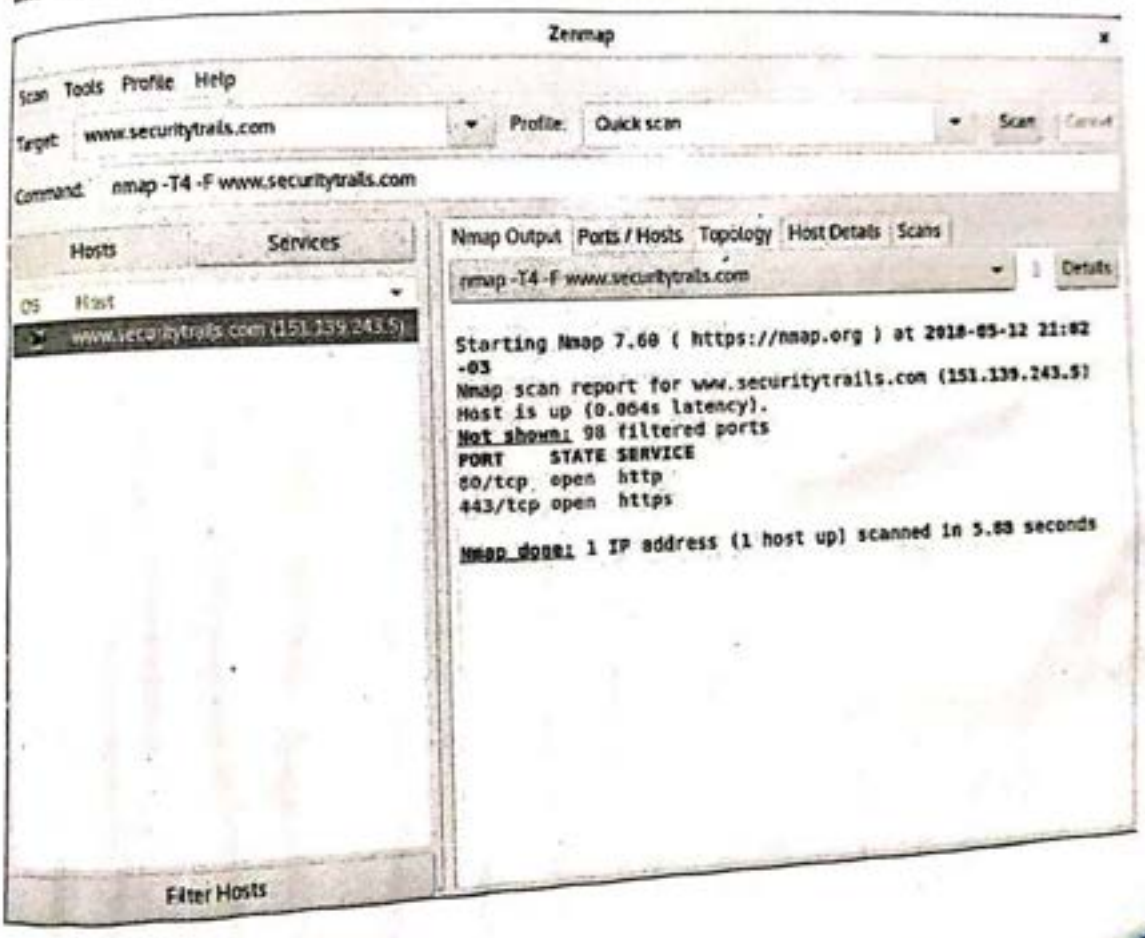
1. Download Nmap from www.nmap.org and install the Nmap Software with WinPcap Driver utility.
2. Execute the Nmap-Zenmap GUI tool from Program Menu or Desktop Icon.
3. Type the Target Machine IP Address(i.e. Guest OS or any website Address)
4. Perform the profiles shown in the utility.

Scan Type	Switch	Description
TCP connect() scan	-sT	Opens a connection to every potentially interesting port on the target machine.
TCP SYN scan	-sS	This is a "half-open" scan.
TCP FIN	-sF	This scan attempts to pass through packet filters by sending a TCP FIN packet.
Xmas Tree	-sX	Sends a packet with FIN, URG and push flags set.
Null	-sN	Sends a packet without any flags turned on.
Scan Type	Switch	Description
ACK scan	-sA	An ACK packet with random acknowledgment and sequence numbers is sent.
UDP scan	-sU	This sends 0 byte UDP packets to each port on the target machine(s).
List scan	-sL	Simply lists targets to scan.

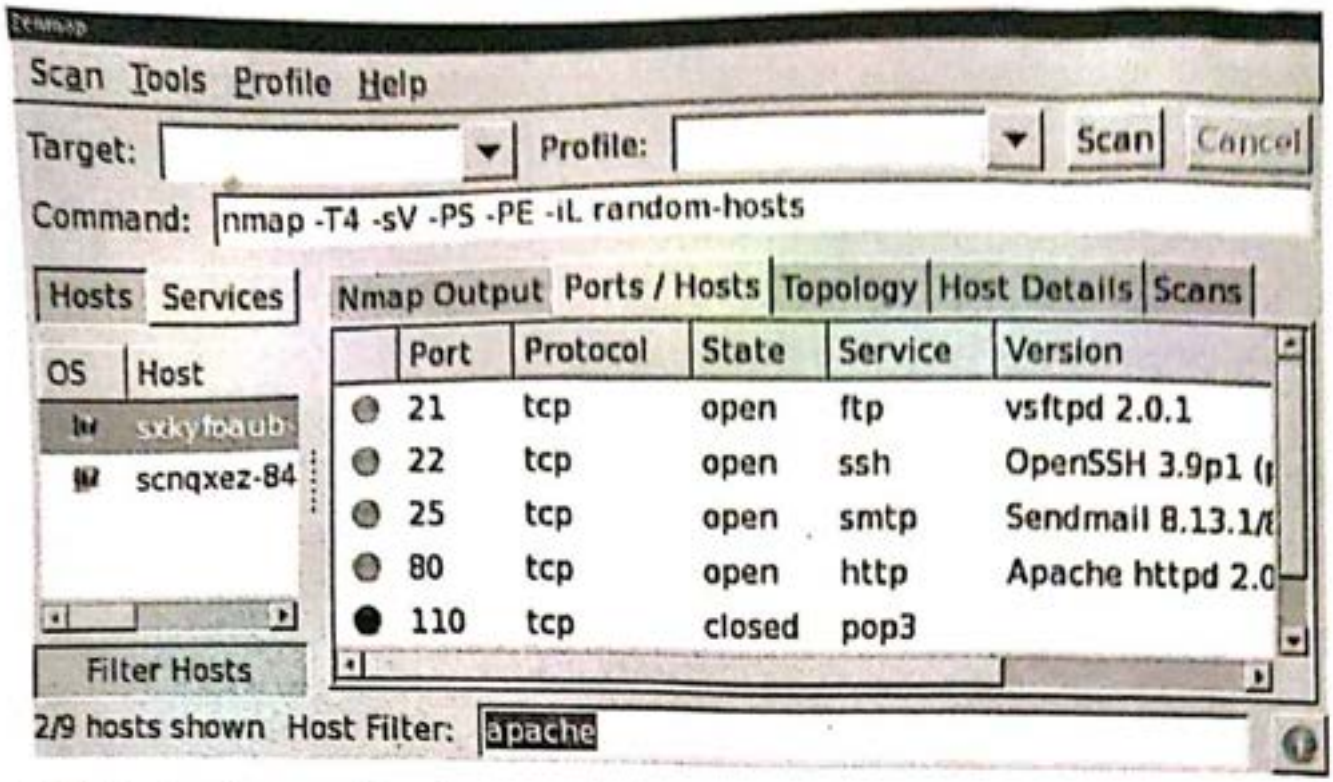
Scanned by CamScanner

NMAP installation steps along with output in Linux environment are listed below

```
sana@linux:~$ sudo apt install nmap
[sudo] password for sana:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libblas3 liblinear3
Suggested packages:
  liblinear-tools liblinear-dev ndiff
The following NEW packages will be installed:
  libblas3 liblinear3 nmap
0 upgraded, 3 newly installed, 0 to remove and 3 not upgraded.
Need to get 5,353 kB of archives.
After this operation, 24.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```



Scanned by CamScanner



Students can also compile and execute the below program

```
/*****************************************************************************
TITLE: C Program of PORT-SCANNING compile & run by using cross compiler of Linux environment
*****************************************************************************/

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <netdb.h>
#include <netinet/in.h>
#include <sys/socket.h>

FILE *ps;

int main(intargc, char *argv[])
{
        int sock, port,i;
        structhostent *h;
        time_tcurtime;
```

Scanned by CamScanner

```
    struct tm *loctime;
    char *t;

    ps=fopen("/etc/bbb/portscan.txt","w");
curtime = time (NULL);
loctime = localtime (&curtime);
    t=asctime (loctime);
    fprintf(ps,"\n*****************************************************\n");
    fprintf(ps,"\nPort Scan Results : %s",t);
    fprintf(ps,"\nFollowing ports are open:\n");

if ((h=gethostbyname(argv[1])) == NULL)
    {
printf("Gethostbynameka error!!");
exit(1);
    }
    for(port=0; port<=65000; port++)
    {
        structsockaddr_inaddr;

    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) == -1)
        {
        printf("Socket ka error!!");
        exit(1);
    }
    addr.sin_family = AF_INET;   // host byte order
    addr.sin_port = htons(port);  // short, network byte order
    addr.sin_addr = *((structin_addr *)h->h_addr);
        addr.sin_zero[0]='\0';
        i=0;
        while(1)
        {
            addr.sin_zero[i]='\0';
            i++;
```

Scanned by CamScanner

```
            if(i>8)
                    break;
        }
if (connect(sock, (structsockaddr *)&addr, sizeof(structsockaddr)) == -1)
        {
//              printf("%d closed\n", port);
                close(sock);
    }
        else
        {
                printf("%d open\n", port);
                fprintf(ps,"\n%d",port);
                close(sock);
        }
    }
    fprintf(ps,"\n*****************************************************\n");
    fclose(ps);
}
/*
```

Output

```
[root@localhost]# cc port_scan.c
[root@localhost]# ./a.outlocalhost
22 open
25 open
111 open
631 open
5335 open
32769 open
[root@localhost]#
*/
```

Scanned by CamScanner

Experiment 6 : Simulate DOS attack using Hping, hping3 and other tools.

This can be done in two ways one is using hping3 tool and another using IDS tool.

Experiment 6(a): DoS using hping3

Solution :

- Denial of service attack means making the network unavailable for the user to communicate securely.
- It is generally done by interrupting in the network connection between the users or making some services unavailable for user or disrupts the entire network by overloading with unwanted messages, so that network becomes slow and unavailable.
- DoS attack attempt to shut down the network, computer services and deny the use of resources or services to authorized users.
- Once attacker got entire access of network or server he can do the following things :
  - Flood the entire network or server with traffic until shutdown occurs because of overload.
  - Block ongoing traffic which results in a loss of access to network resources to the authorized users. Different security policies like firewall, Intrusion detection system helps to protect such type of attacks.
  - Different security policies like firewall, Intrusion detection system helps to protect such type of attacks.

What is hping3 ?

- hping3 is a free packet generator and analyzer for the TCP/IP protocol. Hping is one of the de-facto tools for security auditing and testing of firewalls and networks, and was used to exploit the Idle Scan scanning technique now implemented in the Nmap port scanner.
- The new version of hping, hping3, is scriptable using the Tcl language and implements an engine for string based, human readable description of TCP/IP packets, so that the programmer can write scripts related to low level TCP/IP packet manipulation and analysis in a very short time.

hping3 also used to ....

- Traceroute/ping/probe hosts behind a firewall that blocks attempts using the standard utilities.

Scanned by CamScanner

- Perform the idle scan (now implemented in nmap with an easy user interface).
- Test firewalling rules.
- Test IDSes.
- Exploit known vulnerabilties of TCP/IP stacks.
- Networking research.
- Learn TCP/IP (hping was used in networking courses AFAIK).
- Write real applications related to TCP/IP testing and security.
- Automated firewalling tests.
- Proof of concept exploits.
- Networking and security research when there is the need to emulate complex TCP/IP behaviour.
- Prototype IDS systems.
- Simple to use networking utilities with Tk interface.

DoS using hping3 with random source IP

you installed Kali Linux to learn how to DoS. You only need to run a single line command as shown below :

```
santoshdarade:~# hping3 -c 10000 -d 120 -S -w 64 -p 21 --flood --rand-source www.hping3testsite.com
HPING www.hping3testsite.com (lo 127.0.0.1): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown ^C
--- www.hping3testsite.com hping statistic ---
1189112 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
santoshdarade:~#
```

1. hping3 = Name of the application binary.
2. -c 100000 = Number of packets to send.
3. -d 120 = Size of each packet that was sent to target machine.
4. -S = I am sending SYN packets only.
5. -w 64 = TCP window size.
6. -p 21 = Destination port (21 being FTP port). You can use any port here.
7. --flood = Sending packets as fast as possible, without taking care to show incoming replies. Flood mode.

Scanned by CamScanner

er interface).

e complex TCP/IP

un a single line

ng3testsite.com

ncoming

--rand-source = Using Random Source IP Addresses. You can also use -a or --spoof to hide bostnames. See MAN page below.

www.hping3testsite.com = Destination IP address or target machines IP address. You can also use a website name here. In my case resolves to 127.0.0.1 (as entered in /etc/hosts file)

Simple command to DoS using hping3 and nping

- Simple SYN flood – DoS using HPING3

```
santoshdarade:~# hping3 -S --flood -V www.hping3testsite.com
using lo, addr: 127.0.0.1, MTU: 65536
HPING www.hping3testsite.com (lo 127.0.0.1): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown ^C
--- www.hping3testsite.com hping statistic ---
746021 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
santoshdarade:~#
```

- Simple SYN flood with spoofed IP – DoS using HPING3

```
santoshdarade:~# hping3 -S -P -U --flood -V --rand-source www.hping3testsite.com
using lo, addr: 127.0.0.1, MTU: 65536
HPING www.hping3testsite.com (lo 127.0.0.1): SPU set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown ^C
--- www.hping3testsite.com hping statistic ---
554220 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
santoshdarade:~#
```

- TCP connect flood – DoS using NPING

```
Santoshdarade:~# nping --tcp-connect -rate=90000 -c 900000 -q www.hping3testsite.com
Starting Nping 0.6.46 ( http://nmap.org/nping ) at 2014-08-21 16:20 EST
^CMaxrtt: 7.220ms | Min rtt: 0.004ms | Avgrtt: 1.684ms
TCP connection attempts: 21880 | Successful connections: 5537 | Failed: 16343 (74.69%)
Nping done: 1 IP address pinged in 3.09 seconds
santoshdarade:~#
```

Conclusion : Hence we have studied how to simulate DOS attack using hping3.

Scanned by CamScanner

Experiment 6(b): DoS attack detection using Intrusion Detection System.

Aim : To install intrusion system and detect whether any malicious activities detected or not by capturing the live network packets. IDS detect IP Spoofing (fake IP address), Denial of service attacks.

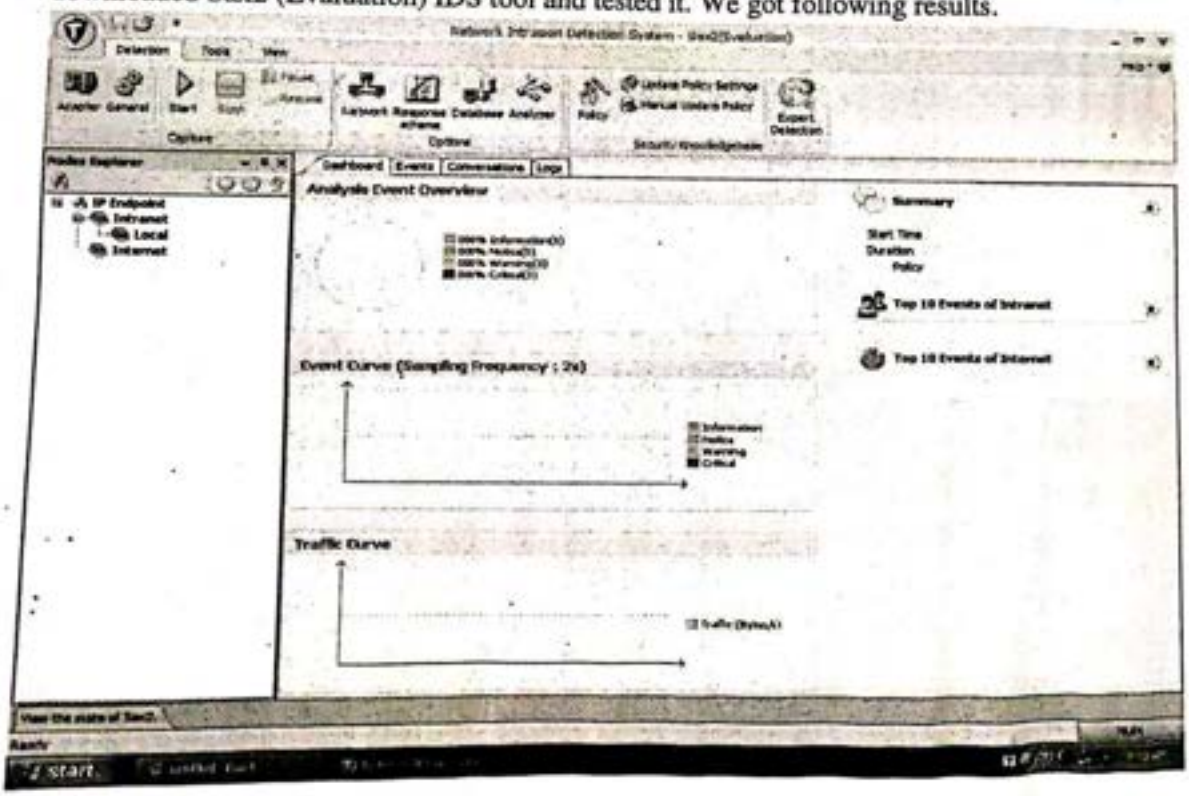
Objective

- "An Intrusion Detection System is software that monitors the events occur in a computer systems or networks, analyzing what happens during an execution and tries to find out indications that the computer has been misused in order to achieve confidentiality, integrity and availability of a resource or data".
- The IDS will continuously run on our system in the background and only generate the alert when it detects something suspicious as per its own rules and regulation or attack signature present into it and taking some immediate action to prevent damage.
- **An Intrusion detection :** System examines or monitors system or network activity to find possible attacks on the system or network. Signs of violation of system security policies, standard security practices are analyzed. Intrusion Prevention is the process of detecting intruders and preventing them from intrusive effort to system.

Solution :

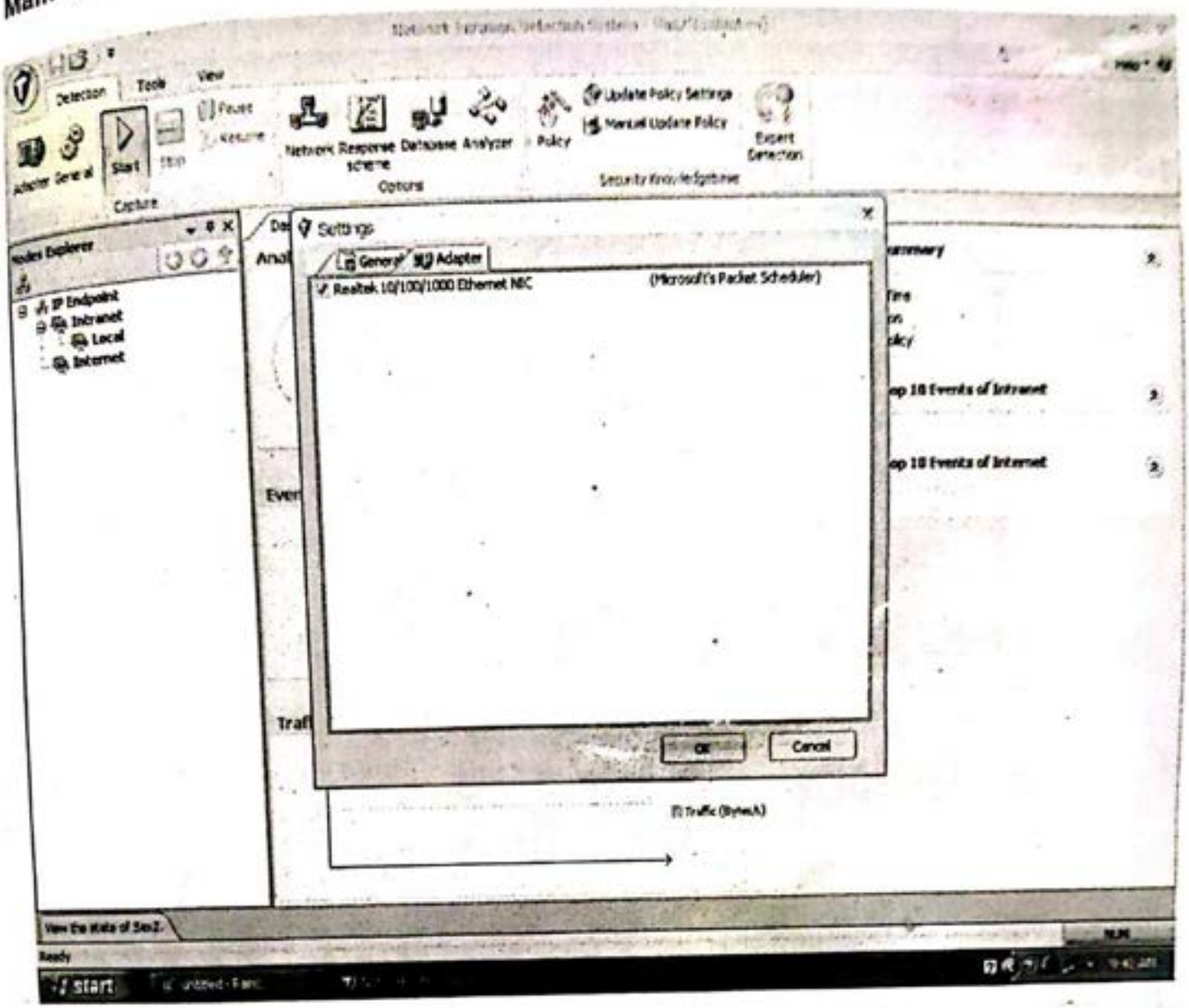
Refer Chapter 12 - section 12.5 for detail on IDS

For demonstration you can download any freely available IDS tool here we have downloaded Sax2 (Evaluation) IDS tool and tested it. We got following results.



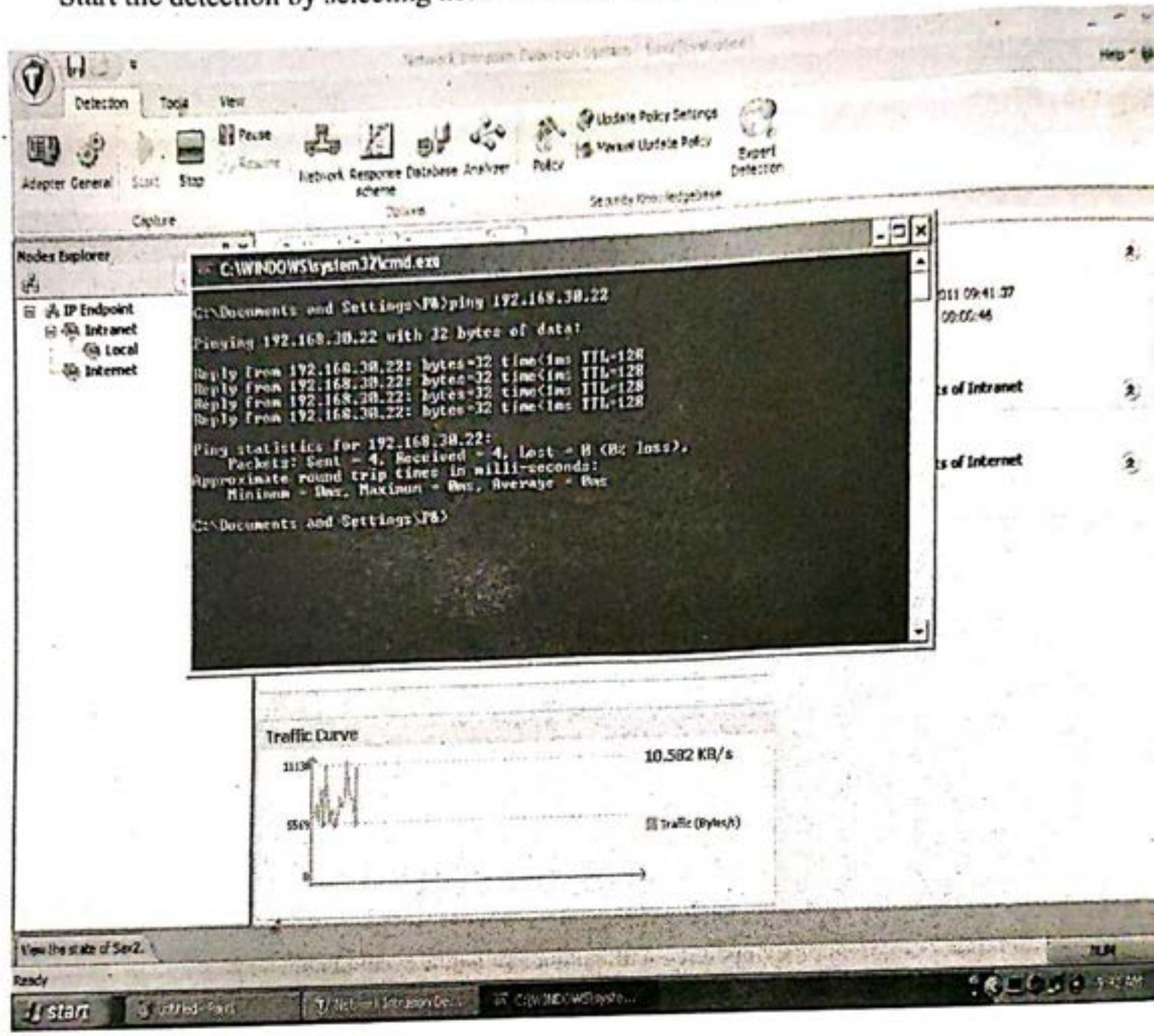
Scanned by CamScanner

Main window of Sax2 IDS tool



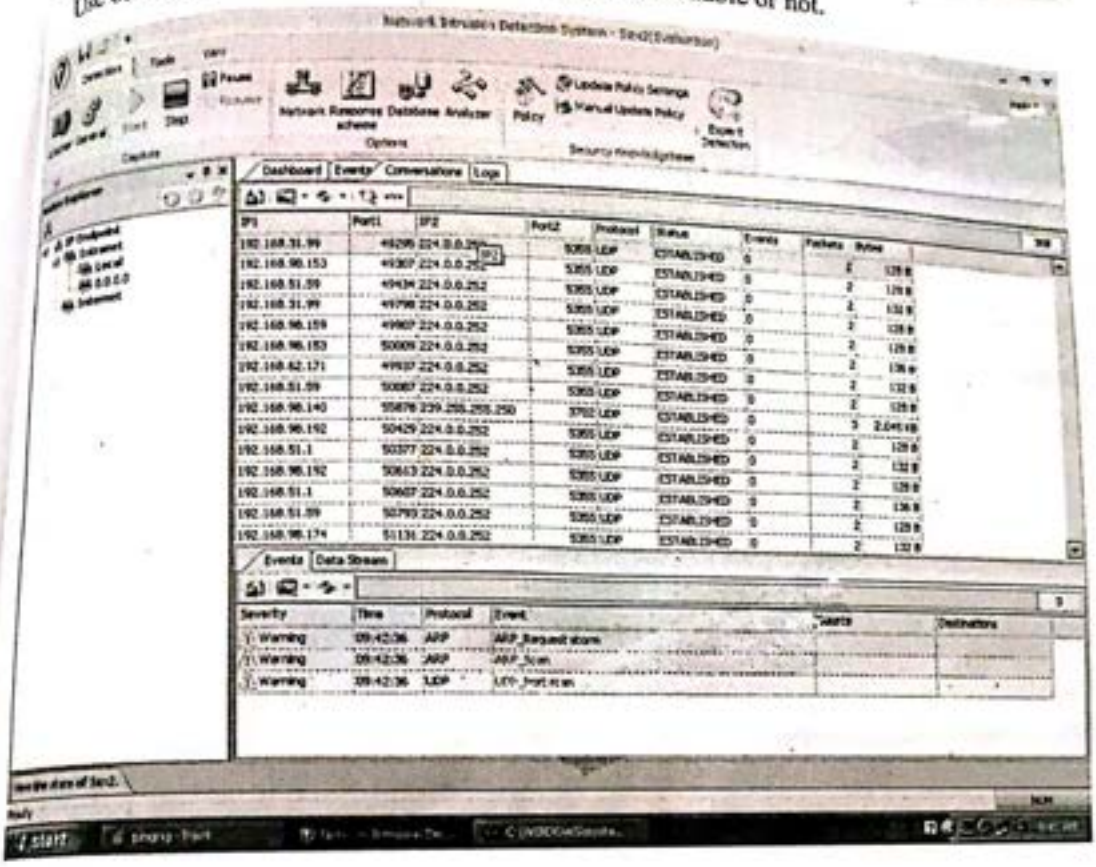
Scanned by CamScanner

Start the detection by selecting network interface card (Adaptor)



Scanned by CamScanner

Use of Ping command to check destination host is reachable or not.



Detection of different attacks

Experiment 7 : Setting up personal Firewall using iptables.

Aim : Setting Firewall Using iptables on Ubuntu 14.04

Solution :

- Setting up a good firewall is an essential step to take in securing any modern operating system. Most Linux distributions ship with a few different firewall tools that we can use to configure our firewalls. In this guide, we'll.be covering the iptables firewall.
- iptables is a standard firewall included in most Linux distributions by default (a modern variant called nftables will begin to replace it). It is actually a front end to the kernel-level netfilter hooks that can manipulate the Linux network stack. It works by matching each packet that crosses the networking interface against a set of rules to decide what to do.

Scanned by CamScanner

- In the previous guide, we learned how iptables rules work to block unwanted traffic. In this guide, we'll move on to a practical example to demonstrate how to create a basic rule set for an Ubuntu 14.04 server. The resulting firewall will allow SSH and HTTP traffic.

Basic Iptables Commands

First, you should be aware that iptables commands must be run with root privileges.

A good starting point is to list the current rules that are configured for iptables.

```
$sudoiptables– L
```

Output

```
Chain INPUT (policy ACCEPT)
targetprot opt source          destination

Chain FORWARD (policy ACCEPT)
targetprot opt source          destination

Chain OUTPUT (policy ACCEPT)
targetprot opt source          destination
```

```
$sudoiptables –S
```

Output

```
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
```

- Once again, the default policy is important here, because, while all of the rules are deleted from your chains, the default policy will *not* change with this command. That means that if you are connected remotely, you should ensure that the default policy on your INPUT and OUTPUT chains are set to ACCEPT prior to flushing your rules.

```
$ sudoiptables –F
$ sudoiptables -P INPUT ACCEPT
$ sudoiptables -P OUTPUT ACCEPT
$ sudoiptables -F
```

Scanned by CamScanner

Make your First Rule

- We're going to start to build our firewall policies. As we said above, we're going to be working with the INPUT chain since that is the funnel that incoming traffic will be sent through. We are going to start with the rule that we've talked about a bit above : the rule that explicitly accepts your current SSH connection.
- The full rule we need is this :

```
$ sudoiptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
```

- This may look incredibly complicated, but most of it will make sense when we go over the components :
  - **-A INPUT :** The -A flag appends a rule to the end of a chain. This is the portion of the command that tells iptables that we wish to add a new rule, that we want that rule added to the end of the chain, and that the chain we want to operate on is the INPUT chain.
  - **-m conntrack :** iptables has a set of core functionality, but also has a set of extensions or modules that provide extra capabilities.
  - In this portion of the command, we're stating that we wish to have access to the functionality provided by the conntrack module. This module gives access to commands that can be used to make decisions based on the packet's relationship to previous connections.
  - **–ctstate :** This is one of the commands made available by calling the conntrack module. This command allows us to match packets based on how they are related to packets we've seen before.
  - We pass it the value of ESTABLISHED to allow packets that are part of an existing connection. We pass it the value of RELATED to allow packets that are associated with an established connection. This is the portion of the rule that matches our current SSH session.
  - **-j ACCEPT :** This specifies the target of matching packets. Here, we tell iptables that packets that match the preceding criteria should be accepted and allowed through.
- We put this rule at the beginning because we want to make sure the connections we are already using are matched, accepted, and pulled out of the chain before reaching any DROP rules.

```
$ sudoiptables –L
```

Scanned by CamScanner

Output

```
Chain INPUT (policy ACCEPT)
targetprot opt source          destination
ACCEPT     all -- anywhere          anywherectstate RELATED,ESTABLISHED

Chain FORWARD (policy ACCEPT)
targetprot opt source          destination

Chain OUTPUT (policy ACCEPT)
targetprot opt source          destination
```

Accept Other Necessary Connections

- We have told iptables to keep open any connections that are already open and to allow new connections related to those connections. However, we need to create some rules to establish when we want to accept new connections that don't meet those criteria.
- We want to keep two ports open specifically. We want to keep our SSH port open (we're going to assume in this guide that this is the default 22. If you've changed this in your SSH configuration, modify your value here). We are also going to assume that this computer is running a web server on the default port 80. If this is not the case for you, you don't have to add that rule.
- The two lines we're going to use to add these rules are :

```
$sudoiptables -A INPUT -p tcp --dport 22 -j ACCEPT
$sudoiptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

- The new options are :
  - **-p tcp :** This option matches packets if the protocol being used is TCP. This is a connection-based protocol that will be used by most applications because it allows for reliable communication.
  - **--dport :** This option is available if the -p tcp flag is given. It gives a further requirement of matching the destination port for the matching packet. The first rule matches for TCP packets destined for port 22, while the second rule matches TCP traffic pointed towards port 80.
- There is one more accept rule that we need to ensure that our server can function correctly. Often, services on the computer communicate with each other by sending

Scanned by CamScanner

network packets to each other. They do this by utilizing a pseudo network interface called the loopback device, which directs traffic back to itself rather than to other computers.

- So if one service wants to communicate with another service that is listening for connections on port 4555, it can send a packet to port 4555 of the loopback device. We want this type of behavior to be allowed, because it is essential for the correct operation of many programs.

```
$ sudoiptables -I INPUT 1 -i lo -j ACCEPT
```

- **-I INPUT 1 :** The -I flag tells iptables to insert a rule. This is different than the -A flag which appends a rule to the end. The -I flag takes a chain and the rule position where you want to insert the new rule.
- In this case, we're adding this rule as the very first rule of the INPUT chain. This will bump the rest of the rules down. We want this at the top because it is fundamental and should not be affected by subsequent rules.
- **-i lo :** This component of the rule matches if the interface that the packet is using is the "lo" interface. The "lo" interface is another name for the loopback device. This means that any packet using that interface to communicate (packets generated on our server, for our server) should be accepted.
- To see our current rules, we should use the -S flag. This is because the -L flag doesn't include some information, like the interface that a rule is tied to, which is an important part of the rule we just added :

```
$ sudoiptables –S
```

Output

```
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
```

Implementing a Drop Rule

- We now have four separate rules that explicitly accept packets based on certain criteria. However, our firewall currently is not blocking anything.

Scanned by CamScanner

- If a packet enters the INPUT chain and doesn't match one of the four rules that we made, it is being passed to our default policy, which is to accept the packet anyways. We need to change this.
- There are two different ways that we can do this, with some pretty important differences.
- The first way we could do this is to modify the default policy of our INPUT chain. We can do this by typing :

```
$ sudoiptables -P INPUT DROP
```

- This will catch any packets that fall through our INPUT chain, and drop them. This is what we call a default drop policy. One of the implications of this type of a design is that it falls back on dropping packets if the rules are flushed.
- This may be more secure, but also can have serious consequences if you don't have another way of accessing your server. With DigitalOcean, you can log in through our web console to get access to your server if this happens. The web console acts as a virtual local connection, so iptables rules will not affect it.
- You may like your server to automatically drop all connections in the event that the rules are dumped. This would prevent your server from being left wide open. This also means that you can easily append rules to the bottom of the chain easily while still dropping packets as you'd like.
- The alternative approach is to keep the default policy for the chain as accept and add a rule that drops every remaining packet to the bottom of the chain itself.
- If you changed the default policy for the INPUT chain above, you can set it back to follow along by typing:

```
$ sudoiptables -P INPUT ACCEPT
```

- Now, you can add a rule to the bottom of the chain that will drop any remaining packets :

```
$ sudoiptables -A INPUT -j DROP
```

- The result under normal operating conditions is exactly the same as a default drop policy. This rule works by matching every remaining packet that reaches it. This prevents a packet from ever dropping all of the way through the chain to reach the default policy.
- Basically, this is used to keep the default policy to accept traffic. That way, if there are any problems and the rules are flushed, you will still be able to access the machine over the network. This is a way of implementing a default action without altering the policy that will be applied to an empty chain.

Scanned by CamScanner

- Of course, this also means that any rule that any additional rule that you wish to add to the end of the chain will have to be added before the drop rule. You can do this either by temporarily removing the drop rule :

```
$sudoiptables -D INPUT -j DROP
$sudoiptables -A INPUT new_rule_here
$ sudoiptables -A INPUT -j DROP
```

- Or, you can insert rules that you need at the end of the chain (but prior to the drop) by specifying the line number. To insert a rule at line number 4, you could type :

```
$sudoiptables -I INPUT 4 new_rule_here,
$sudoiptables -L --line-numbers
```

Output

```
Chain INPUT (policy DROP)
num  targetprot opt source          destination
1   ACCEPT    all -- anywhere      anywhere
2   ACCEPT    all -- anywhere      anywherectstate RELATED,ESTABLISHED
3   ACCEPT    tcp -- anywhere      anywheretcpdpt:ssh
4   ACCEPT    tcp -- anywhere      anywheretcpdpt:http

Chain FORWARD (policy ACCEPT)
num  targetprot opt source          destination

Chain OUTPUT (policy ACCEPT)
num  targetprot opt source          destination
```

Saving your Iptables Configuration

```
$sudo apt-get update
$sudo apt-get install iptables-persistent
```

Conclusion

Hence we have studied how to configure the firewall using iptables.

Scanned by CamScanner

➢ Experiment 8 : Set up Snort and study the logs.

Aim : Intrusion detection has become an extremely important feature of the defense-in-depth strategy. Snort is free network intrusion detection software. It can perform protocol analysis, content searching/matching, and can be used to detect a variety of attacks and probes, such as buffer overflow, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more. The main aim of the experiment is to implement snort on windows or Linux platform.

Objective

To Study what is snort, Implementation of snort.

Solution :

Theory

Snort is an open source network intrusion prevention and detection system (IDS/IPS) developed by source fire. Vombining the benefits of signature, protocol, and anomaly-based inspection, snort is the most widely deployed IDS/IPS technology worldwide.

Hardware and software requirements

- Hardware requirements for this system are dependent upon the size of your network and volume of traffic. The minimum hardware required is 1 GB RAM, a core processor and at least 2 GB free space on the hard drive. Snort can be implemented on any Linux platform or on the latest windows systems.
- There are three main modes in which Snort can be configured: sniffer, packet logger, and network intrusion detection system.
- Sniffer mode simply reads the packets off of the network and displays them for you in a continuous stream on the console. Packet logger mode logs the packets to the disk. Network intrusion detection mode is the most complex and configurable configuration, allowing Snort to analyze network traffic for matches against a user defined rule set and performs several actions based upon what it sees.
- Snort's open source network-based intrusion detection system (NIDS) has the ability to perform real-time traffic analysis and packet logging on Internet Protocol (IP) networks. Snort performs protocol analysis, content searching, and content matching. The program can also be used to detect probes or attacks, including, but not limited to, operating system fingerprinting attempts, common gateway interface, buffer overflows, server message block probes, and stealth port scans.

Scanned by CamScanner

Implementation on windows platform

Installing the base Snort system requires two components : The WinPcap packet capture library, and the Snort IDS program itself. In the following sections we configure and install both WinPcap and Snort.

WinPcap

WinPcap (Windows Packet Capture Library) is a packet-capture driver. Functionally, this means that WinPcap grabs packets from the network wire and pitches them to Snort. WinPcap is a Windows version of libpcap, which is used for running Snort with Linux.

Functions

The WinPcap driver performs these functions for Snort :

1. Obtains a list of operational network adapters and retrieves information about the adapters.
2. Sniffs packets using one of the adapters that you select.
3. Saves packets to the hard drive (or more importantly for us, pitches them to Snort).

Installation

The installation and configuration of WinPcap is explained as below :

1. Download the latest installation file from http://winpcap.polito.it/install/default.htm

   The installation file is generally called something like WinPcap_3_0.exe.
2. Double-click the executable installation file and follow the prompts.

   WinPcap installs itself where it belongs.

   Snort calls WinPcap directly on any of the functions to grab and analyze network packets. If the driver did not install properly, Snort does not function

snort.org distributes a convenient install package for Windows available at its Web site:

http://www.snort.org/dl/binaries/win32/

Download this package (generally called snort-2_1_0.exe) and perform the following

Steps to install Snort

1. Double-click the executable installation file. The GNU Public License appears.
2. Click the I Agree button. Installation Options window appears.

Scanned by CamScanner

3. In the Installation Options dialog box, click the appropriate boxes to select from among these options :
   - I do not plan to log to a database, or I am planning to log to one of the databases listed above. Choose this option if you are not using a database or if you are using MySQL or ODBC databases.
   - Snort has built-in support for these databases, and for our example, we chose this option.
   - I need support for logging to Microsoft SQL Server. Only click this radio button if you already have SQL Server client software installed on this computer, and you plan to use MSSQL as your logging database.
   - I need support for logging to oracle. Only choose this option if you have the Oracle client software installed on this computer, and you plan to use Oracle as your logging database server.
4. Click the Next button. The choose components window appears.
5. In the choose components window, select the components you want to install and then click the Next button. The install location window appears.
6. Choose a directory to install to.
7. Click the Install button.
8. When the installation is complete, click the Close button. An information window appears.
9. Click the OK button.

A new Snort installation requires a few configuration points. Conveniently, one file has all the configuration settings required.

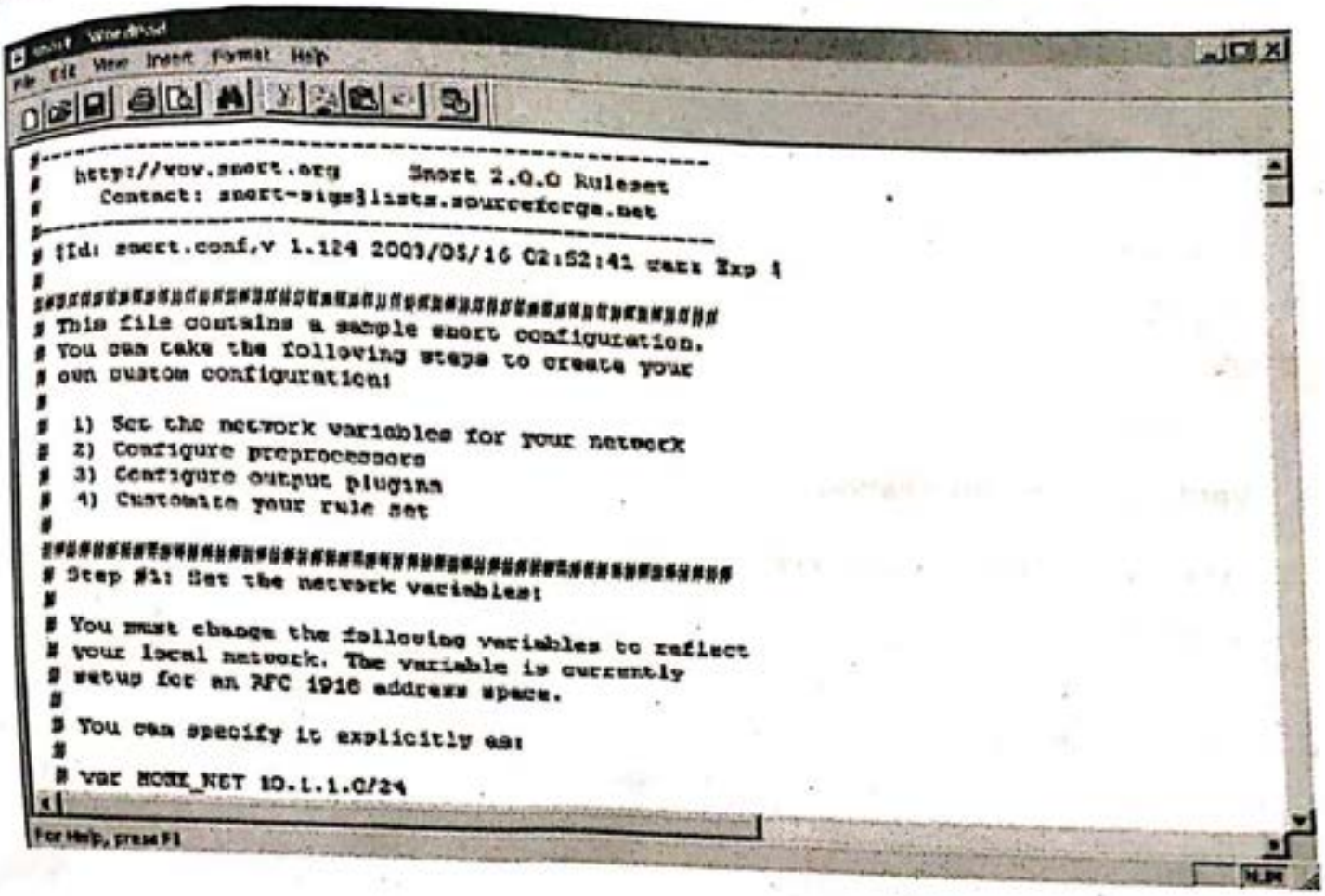
Snortpath/etc/snort.conf

When you're ready to configure Snort, open snort.conf in a text editor.

The following configuration options in the snort.conf file are essential to a properly Functioning Snort installation

1. Network settings
2. Rules settings
3. Output settings
4. Include settings

Scanned by CamScanner



```
#--------------------------------------------------
#   http://www.snort.org      Snort 2.0.0 Ruleset
#     Contact: snort-sigs@lists.sourceforge.net
#--------------------------------------------------
# $Id: snort.conf,v 1.124 2003/05/16 02:52:41 max Exp $
#
# This file contains a sample snort configuration.
# You can take the following steps to create your
# own custom configuration:
#
#  1) Set the network variables for your network
#  2) Configure preprocessors
#  3) Configure output plugins
#  4) Customize your rule set
#
# Step #1: Set the network variables:
#
# You must change the following variables to reflect
# your local network. The variable is currently
# setup for an RFC 1918 address space.
#
# You can specify it explicitly as:
#
# var HOME_NET 10.1.1.0/24
```

Implementing snort on a Linux platform

Download and Extract Snort

1. **Download the latest snort free version from snort website. Extract the snort source code to the /usr/src directory as shown below :**

```
# cd /usr/src
# wget -O snort-2.8.6.1.tar.gz http://www.snort.org/downloads/116
# tar xvzf snort-2.8.6.1.tar.gz
```

2. **Install Snort**

Before installing snort, make sure you have dev packages of libpcap and libpcre.

```
# apt-cache policy libpcap0.8-dev
libpcap0.8-dev:
Installed: 1.0.0-2ubuntu1
Candidate: 1.0.0-2ubuntu1
```

Scanned by CamScanner

```
# apt-cache policy libpcre3-dev
libpcre3-dev:
 Installed: 7.8-3
Candidate: 7.8-3
# cd snort-2.8.6.1
# ./configure
# make
# make install
```

3. Verify the Snort Installation

Verify the installation as shown below :

```
# snort --version
  ,,_     -*> Snort! <*-
o" )~   Version 2.8.6.1 (Build 39)
  '' By Martin Roeschand The Snort Team: http://www.snort.org/snort/snort-team
        Copyright (C) 1998-2010 Sourcefire, Inc., et al.
        Using PCRE version: 7.8 2008-09-05
```

4. Create the required files and directory

You have to create the configuration file, rule file and the log directory.

Create the following directories

```
# mkdir /etc/snort
# mkdir /etc/snort/rules
# mkdir /var/log/snort
```

Create the following snort.conf and icmp.rulesfiles

```
# cat /etc/snort/snort.conf
include /etc/snort/rules/icmp.rules
# cat /etc/snort/rules/icmp.rules
alerticmp any any -> any any (msg:"ICMP Packet"; sid:477; rev:3;)
```

The above basic rule does alerting when there is an ICMP packet (ping).

Following is the structure of the alert:

<Rule Actions> <Protocol> <Source IP Address> <Source Port> <Direction Operator> <Destination IP Address> <Destination > (rule options).

Scanned by CamScanner

Table 1 : Rule structure and example

Structure	Example
Rule Actions	Alert
Protocol	Icmp
Source IP Address	Any
Source Port	Any
Direction Operator	->
Destination IP Address	Any
Destination Port	Any
(rule options)	(msg:"ICMP Packet"; sid:477; rev:3;)

5. Execute snort

Execute snort from command line, as mentioned below :

```
# snort -c /etc/snort/snort.conf -l /var/log/snort/
```

Try pinging some IP from your machine, to check our ping rule. Following is the example of a snort alert for this ICMP rule.

```
# head /var/log/snort/alert
[**] [1:477:3] ICMP Packet [**]
[Priority: 0]
07/27-20:41:57.230345 > I/I len: 0 I/I type: 0x200 0:0:0:0:0:0
pkt type:0x4 proto: 0x800 len:0x64
209.85.231.102 -> 209.85.231.104 ICMP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:84 DF
Type:8 Code:0 ID:24905 Seq:1 ECHO
```

Alert Explanation

A couple of lines are added for each alert, which includes the following :

- Message is printed in the first line.
- Source IP
- Destination IP

Scanned by CamScanner

– Type of packet, and header information.

If you have a different interface for the network connection, then use -dev -i option. In this example my network interface is ppp0.

```
# snort -dev -i ppp0 -c /etc/snort/snort.conf -l /var/log/snort/
```

Execute snort as Daemon.

Add -D option to run snort as a daemon.

```
# snort -D -c /etc/snort/snort.conf -l /var/log/snort/
```

Lab Ends...

□□□

Scanned by CamScanner