



**Uttar Pradesh Rajarshi Tandon
Open University**

Master of Computer Science

MCS -119 INFORMATION AND NETWORK SECURITY

Block-1 **03-88**

UNIT-1	INTRODUCTION
UNIT-2	CLASSICAL ENCRYPTION TECHNIQUES
UNIT-3	BLOCK CIPHERS AND DES
UNIT-4	CONFIDENTIALITY USING SYMMETRIC CIPHERS

Block-2 **89-166**

UNIT-5	Introduction to Number Theory
UNIT-6	Public Key Cryptography
UNIT-7	Message Authentication and Hash Function
UNIT-8	Digital Signature

Block-3 **167-252**

UNIT-9	Authentication Applications
UNIT-10	Electronic Mail Security
UNIT-11	IP Security
UNIT-12	Web Security

Block-4 **253-312**

UNIT-13	Intruders
UNIT-14	Malicious Programs
UNIT-15	Firewall



**Uttar Pradesh Rajarshi Tandon
Open University**

Master of Computer Science

**MCS -119
INFORMATION AND NETWORK
SECURITY**

Block

1

INFORMATION SECURITY AND SYMMETRIC CIPHERS

Unit 1	07-22
---------------	--------------

Introduction

Unit 2	23-46
---------------	--------------

Classical Encryption Techniques
--

Unit 3	47-70
---------------	--------------

Block ciphers and DES

Unit 4	71-88
---------------	--------------

Confidentiality Using Symmetric Ciphers
--

Course Design Committee

(Prof.) Ashutosh Gupta **Chairman**
Director (In-charge)
School of Computer and Information Science, UPRTOU,
Prayagraj

Prof. R. S. Yadav **Member**
Department of Computer Science and Engineering
MNNIT-Allahabad, Prayagraj

Dr. Marisha **Member**
Assistant Professor (Computer Science),
School of Science, UPRTOU, Prayagraj

Dr. C. K. Singh **Member**
Lecturer
School of Computer and Information Science, UPRTOU, Prayagraj

Course Preparation Committee

Dr Rajeev Singh **Author**
Assistant Professor
Department of Computer Engineering
GB Pant University of Agriculture and Technology
Pantnagar, Uttarakhand

Prof. Abhaya Saxena **Editor**
Head, Dept. of Computer Science
Dev Sanskriti Vishwavidyalaya
Hardwar, Uttarakhand

Prof. Ashutosh Gupta
Director (In-charge),
School of Computer and Information Science
UPRTOU, Prayagraj

Dr. Marisha **Coordinator**
Assistant Professor (Computer Science)
School of Science
UPRTOU, Prayagraj

© UPRTOU, Prayagraj. 2023
ISBN : 978-93-83328-27-7

All Rights are reserved. No part of this work may be reproduced in any form, by mimeograph or any other means, without permission in writing from the Uttar Pradesh Rajarshi Tondon Open University, Prayagraj.

Printed and Published by Vinay Kumar, Registrar, Uttar Pradesh Rajarshi Tandon Open University, 2023.

Printed By: K.C.Printing & Allied Works, Pacnhwati, Mathura -281003.

COURSE INTRODUCTION

The objective of this course is to introduce the basic and theoretical aspect of Information and Network Security. The security concepts and techniques along with examples are provided. By learning the concepts and security techniques, you should be able to apply them in diversified environments like cloud, web, sensor and ad-hoc networks etc. The aim is to provide an organized and simplified course material covering extensively the topics on this subject. The course is organized into following blocks:

- Block-1** introduces the concept of information security and discusses few popular symmetric ciphers.
- Block -2** covers one of the effective security measure i.e., public key cryptography and a reliable integrity ensuring method i.e., hash functions.
- Block-3** describes the key security application areas like e-mail security, IP security and web security.
- Block-4** describes the functioning and threats by intruders and malwares. It also discusses firewalls.

UNIT-1

INTRODUCTION

Structure

- 1.0 Introduction
- 1.1 Objectives
- 1.2 History
- 1.3 Outlining Information Security
- 1.4 Widespread Information Security Model
- 1.5 Components of Information Security
- 1.6 Aspects of Information Security
- 1.7 Attacks Targeting Security
- 1.8 Security Mechanisms and Security Services (X.800)
- 1.9 Basic Network Security Model
- 1.10 Summary
- 1.11 Terminal Questions

1.0 INTRODUCTION

In this unit, an introduction of elementary concepts of information security and network security is given. The unit initiates historical background and definition of information security. Then, information security model is discussed. It is followed by identification of components and aspects of information security. Security attacks, accompanying security mechanisms and security services are elaborated, their relationship is also presented. Finally, model for network security is illustrated in this unit.

1.1 OBJECTIVES

After the end of this unit, you should be able to:

- define information security;
- know the historical evolution of information security;
- identify the components and aspects of information security
- understand the various kinds of security attacks
- learn about security mechanisms, services and their relationship

- explain information security and network security models

1.2 HISTORY

Security related activities were carried out in earlier time by utilizing signals of smoke/fire for conveying messages to distant partners. Traditional information security related activities refer to cryptography i.e., secret writing and ancient steganography i.e., covered writing. Historically, cryptography is quite old – at least about 4000 years. The oldest noticeable cryptographic algorithm (cipher) was developed by Julius Caesar (50 B.C.) and was named as Caesar cipher. In this cipher, each character in plain text is shifted by 3 letters. Another cipher termed as Atbash cipher was used by Palestinians (600 B.C.) that simply represents letters of the alphabet in reverse order. Figure 1.1 shows examples of ancient security techniques. The German used their popular code machine ‘Enigma’ for security purpose during 1930s. Though later the code was broken during World War-II by British and American people using a computer named ‘Colossus’ [1-3].

An example of early cryptography (Scytale: encryption method used by Spartans in 500 BC)	An example of Ancient steganography
	
A leather belt had characters written on it that appears meaningless (encryption). For reading the message, a piece of wood with correct diameter (key) was taken and the belt is winded around it.	A message in a wooden block was engraved followed by wax covering, so it looked like a blank wax tablet. For message retrieval wax was melted.

Figure 1.1: Some early methods used in securing information

Modern day information security started its journey side by side to the development of the newer computing resources like mainframes, PCs and high end servers. Initially such devices were developed by the military for meeting their objectives and hence were housed mainly in the military accomplishments. Securing hardware of military computing resources and controlling the physical access into the military locations hosting these resources was the main objective of

the security experts during this time. With the development of U.S. Department of Defense's – Advanced Research Project Agency Network (ARPANET), the network related issues and risk raised their hood. The first published paper 'Rand Report R-609' (year 1967) emphasized that the physical security of information systems should be extended to protecting the data, limiting unauthorized access to resources and involvement of persons for security from multiple levels of organization [2].

The security features were added into the then developed operating systems. For example, MULTICS (Multiplexed Information and Computing Service), a time sharing operating system developed for mainframes considered multiple levels and passwords in its core functionality. In 1970s, the password function was added into UNIX operating system that was based upon one way transformation. This transformation converts passwords into unusable and unreadable formats. In 1978, "Protection Analysis: Final Report" was published which identified the vulnerabilities in Operating Systems.

With the arrival of network resources (LAN) and Internet in 1990s, the picture changed dramatically and the information security started gaining priority. The early Internet deployment was based upon the idea of sharing and assuming that the users were trustworthy. Hence, low priority was given to security during this time. Later, as the Internet technology became universal, industry standards were developed considering information security. This resulted in a fundamental change i.e., from securing an individual networked computer to securing the information that is stored on a computer or in-route to other computer.

Start of 21st century saw evolution of new methods of communication like Email, cellular communication, secure web based transactions, digital cash etc. Along with this, thieves and attackers also evolve. The attacks and incidences in 2000s led to development of notion of national security. Hence, government came into picture and several documents and laws for Information security were designed. The Information Technology Act, 2000 became effective from 17 October, 2000. It is the second technology related legislation in India after the Indian Telegraph Act, 1885. It has given comprehensive legal definitions of terms like, computer, computer network, data, computer database etc. The IT Act, 2000 has granted legally an ultimate importance to ICT practices in India. It enables use of electronic records & digital signatures, facilitates e-Governance & e-commerce and imposes civil & criminal sanctions. Subsequent amendments took place in October 27, 2009. Department of Electronics and IT under Ministry of Communications & IT issued a notification regarding National Cyber Security Policy (NCSP-2013) on July 02, 2013 for strengthening the country's cyber ecosystem.

1.3 OUTLINING INFORMATION SECURITY

The Information Systems (IS) are developed with the objective to achieve the goal of enhancing business and productivity of an organization. An Information system is basically combination of

entire hardware, set of software, data, users, procedures and networks. Each of these components has its own strengths, weaknesses and security requirements.

Information Security is referred in short as 'InfoSec'. It is independent of the form of data i.e., it is applicable to data that is either in motion (transmitted on network or under processing) or in rest (stored on computer). Information security needs to be aligned with the overall objectives of an information system. Application of policy, education, training (awareness) and technology helps in accomplishing these objectives.

Committee on National Security Systems (CNSS) defines Information Security as:

"The protection of information and information systems from unauthorized access, use, disclosure, disruption, modification, or destruction." [4]

J. Anderson an executive consultant at Emagined Security, Inc. provides another definition of Information Security as:

"A well-informed sense of assurance that information risks and controls are in balance." [5]

Risk represents the probability that something unwanted will happen whereas control represents security mechanisms, policies, or procedures that can successfully counter attacks, reduce risk, resolve vulnerabilities, and otherwise improve the security within an organization [2].

One of the major drawbacks in the information system development is that the developers follow normal software development cycle ignoring the security issues. It is advisable to concentrate upon the security issues right from the beginning of the system development phase. Hence, issues such as the format into which data will be kept, how and where it will be kept needs to be addressed in early phases. This will ensure that the information risks and controls are in balance.

1.4 WIDESPREAD INFORMATION SECURITY MODEL

Information* is valued by its characteristics. The computer security industry developed a concept termed as C.I.A. triangle. It is based upon three major characteristics of information namely, confidentiality, integrity and availability. Later, the model was expanded due to increasing threats and constantly changing environment to include accuracy, authenticity, utility and possession. The characteristics of information as per the expanded model are:

Availability – Information is made available to an authorized user in the correct format as per the needs.

Accuracy – Information must correctly meet the end user expectations and free from errors.

Authenticity – Information must maintain genuinity or originality.

Confidentiality – Information must be protected from unauthorized access and exposure.

Integrity – Information must be protected from modifications and its state must remain intact or untouched.

Utility – Information must remain valuable for its users.

Possession – Information must maintain its state of ownership.

The notion of information security rotates around strengthening these characteristics. The CNSS (Committee on National Security Systems) presented a comprehensive security evaluation standard for securing information systems where a graphical model for information security was created by John McCumber (in 1991). This model is widely used three dimensional model, with each dimension lying along each axis. Thus, a cube of $3 \times 3 \times 3$ dimension with 27 cells is formed (Figure 1.2). Each cell represents an area to be addressed for securing Information Systems. For example, intersection of Technology, Transmission and Confidentiality means that a Technological control for securing Information during Transmission is required to be addressed. In this case, encrypting or hiding (safeguarding) the data of file or message during transmission will serve as the desired control [2][6].

* In field of security both data and information are important and hence both the terms are used interchangeably. This text also follows this convention.

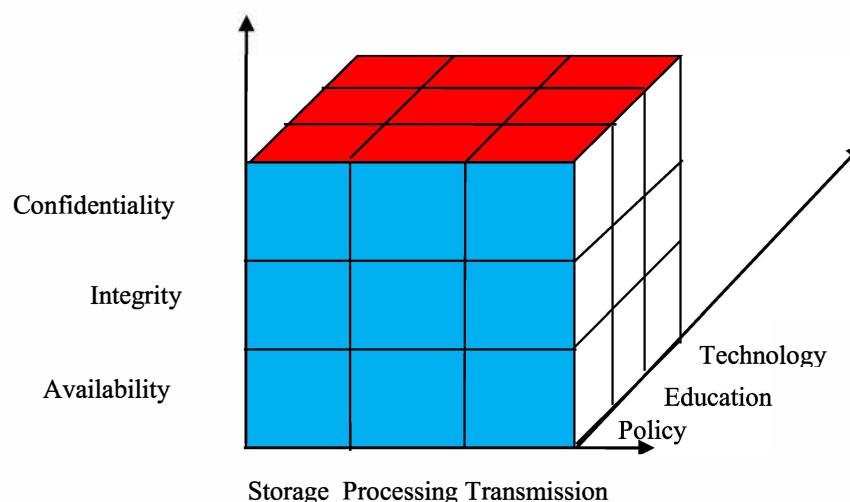


Figure 1.2: The McCumber Cube [6]

1.5 COMPONENTS OF INFORMATION SECURITY

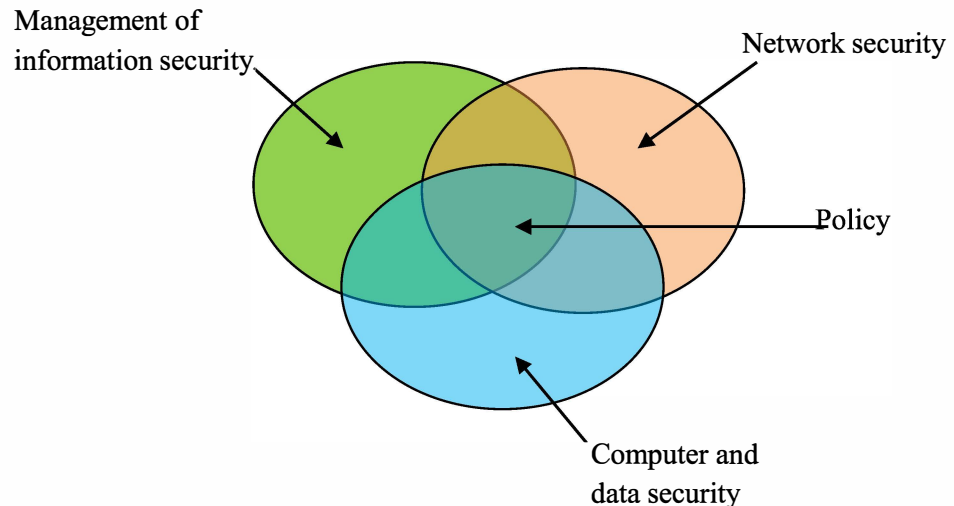


Figure 1.3: Components of Information Security

Information security components include information security management, computer security, data security, and network security as shown in Figure 1.3. Information security management is essential to an organization that must protect its critical assets, including data, infrastructure, and people. It may have sub functions, such as coordinating security services and mechanisms; distributing security-relevant information; reporting security-relevant events; controlling the distribution of cryptographic keying material; and authorizing subscriber access, rights, and privileges. Network security aims to protect networking components, connections, and contents. Computer and data security target to secure the hardware, software and data stored on a system. Policy is central to all the components and it represents set of organizational guidelines that dictates certain behavior within the organization. There are three kinds of policies:

First, security program policy (SPP) or general program policy which sets the strategic direction, scope, and character for all security efforts within an organization.

Second, an issue-specific security policy (ISSP) which addresses specific areas of technology and contains an issue statement on the organization's position on an issue. An ISSP is formalized as written agreed upon documents which are distributed to users.

Third, system-specific security policies (SSSPs) which are frequently codified as standards and procedures used when configuring or maintaining systems.

1.6 ASPECTS OF INFORMATION SECURITY

Every user of an information system wants fast information access. Information security though essential yet poses restriction on information access to some extent. For end user, even a small delay during access may be considered as annoying while for a security

person this time gap may be perceived as important because a required security enhancing task is completed within this time. The relationship between information system and access may be better understood by relation between lock on house and the resources kept inside it. A house with more valuable resources may require even more locks whereas house with less valuable resources may not require any lock at all.

A computer system may be compromised and selected as object of attack by an attacker. It may later be used as subject to attack other systems. For example, under distributed denial of service (DDoS) attack, an army (botnet) is first created by compromising several computer systems and then DDoS attack is conducted towards the target (subject).

Security efforts to protect information depend upon its value and circumstance. The value of information may be lost if it is delivered too late. Without security, information may lose its value whereas completely secure information system would not allow anyone access. Hundred percent absolute security is not feasible. It is a continuous evolving process. A classical paper titled “Why Cryptography is Harder than it looks” by Bruce Schneier, explains this fact. It says that most of the systems are designed and implemented by engineers who lack cryptography potential and usually develop the system considering cryptography as afterthought not from initial phases. Such implemented insecure products looks like secure because of their infancy stage but they may fall to crackers upon wide use. Hence, an experienced cryptographer or security professional needs to be consulted from initial conception. In future only the product having high security strength will prevail. Balances between money spend and security attained has to be maintained while ensuring high security [7].

Three prominent aspects of security are attacks, mechanisms and services.

Security attack: An action through which security of an organization’s information is compromised.

Security mechanism: A process or method that is designed to protect from a security attack.

Security service: A service (either processing or communication) that facilitates the security of a system along with secure information transfer. It is basically the services that protect against security attacks by utilizing one or more security mechanisms. Further, a mechanism can also be used in one or more services. This suggests that the security services and mechanisms are closely related to each other.

We discuss each of these aspects in the subsections ahead.

Check your progress 1

What happens to the value of information if it is delivered too late?

1.7 ATTACKS TARGETING SECURITY

A security attack* is an assault on a system that evades security services and violates the security policy of the system [8]. The three major goals of security i.e., confidentiality, integrity and availability are threatened by various attacks. Based upon this fact attacks are classified three categories. Attacks are also classified into two groups: passive and active.

* A threat is a possible danger that might exploit vulnerability

Passive attacks

The passive category contains all kinds of attacks where the attacker's intension is to just obtain the information but not to modify the data or to harm the system. In this case revealing the sensitive information affects only the sender and the receiver. Attacks in this category are difficult to detect till someone locates the information leakage. Application of cryptography prevents passive attacks.

Active attacks

Attacks in active category may change the data or harm the system. They threaten integrity and availability goals and easier to detect. Figure 1.4 shows a classification of security attacks as per the three security goals and active/passive groups.

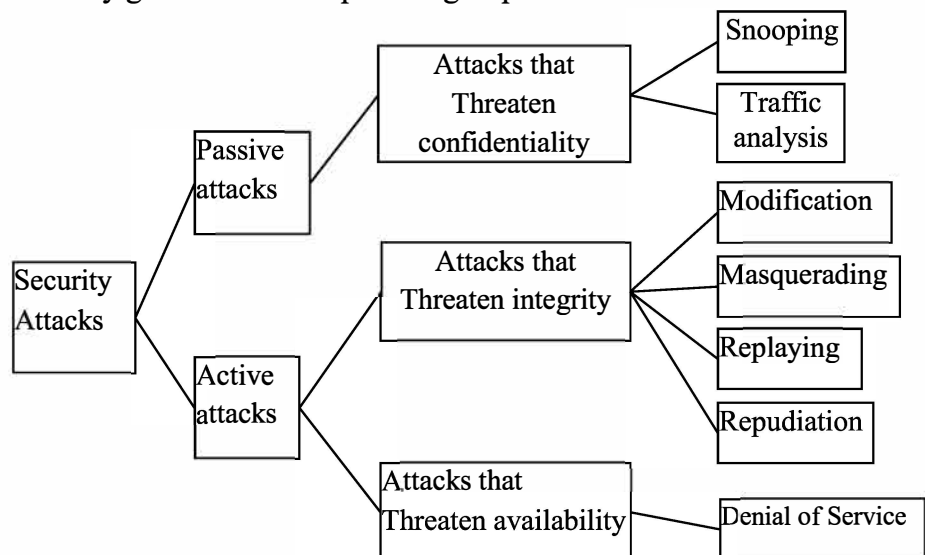


Figure 1.4: A classification of security attacks

Attacks that threaten confidentiality

Snooping: interception of data and its use for attacker's benefit is referred as snooping e.g. learning the unencrypted password on LAN by adversary. Snooping is shown pictorially in the Figure 1.5.

Here and in subsequent figures in this section it is assumed that the two parties named as Jai (sender) and Veeru (recipient) are participating in communication and an adversary named Gabbar is sitting in between and is having mal intentions.

Traffic analysis: monitoring of online traffic is referred as traffic

analysis e.g. behavioral analysis can be performed by studying the user's online activities (Figure 1.6).

Both snooping and traffic analysis fall under passive attack category

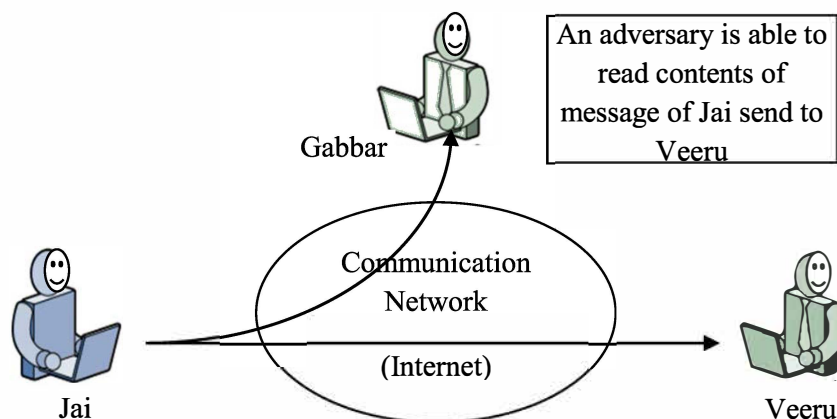


Figure 1.5: Passive attack - Learning the message contents (snooping)

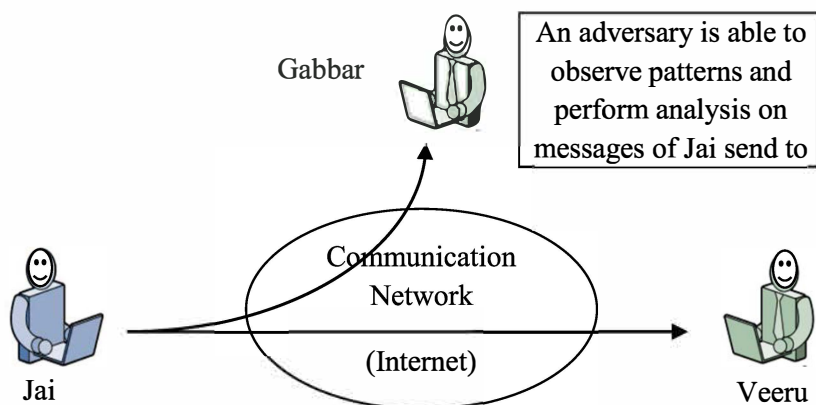


Figure 1.6: Passive attack - Traffic analysis

Attacks that threaten integrity

Under this category the attacker modifies the information in-route and use it for personal benefits e.g. the attacker may change amount or type of financial transaction. This category also covers deleting and delaying of information.

The following attacks are examples of active attacks.

Masquerading: an attack where adversary impersonates the sender and pretends as sender to the destination (Figure 1.7).

Replaying: implies capturing the sender's data and replaying it later by an adversary to achieve undesired effects (Figure 1.8).

Modification: implies that messages can be delayed or reordered or

altered by an adversary to achieve undesired effect (Figure 1.9).

Denial of service (DoS) attack: disruption, disabling, overloading or flooding a resource (of server) such that the client is denied appropriate services (Figure 1.10).

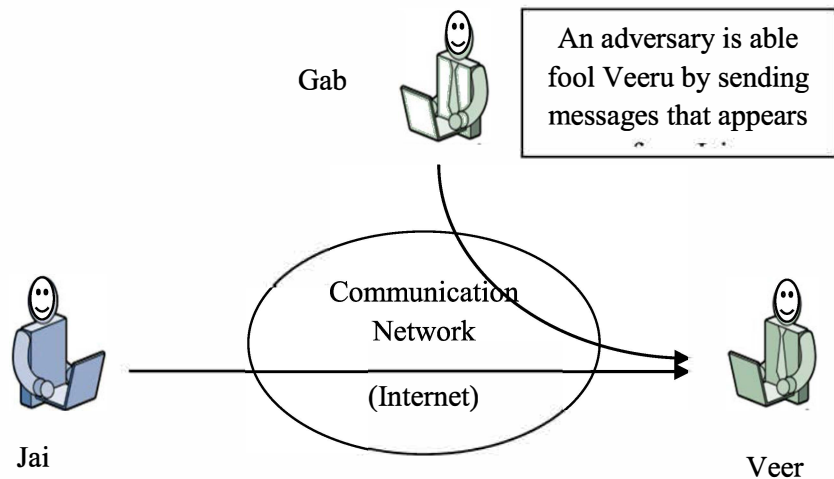


Figure 1.7: Active attack – Masquerading

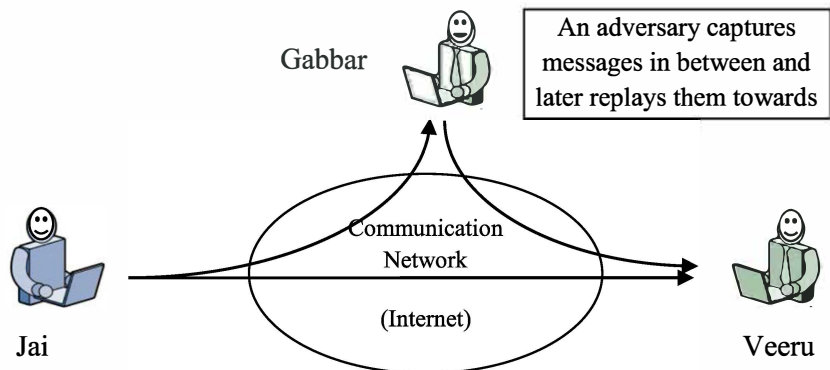


Figure 1.8: Active attack - Replay

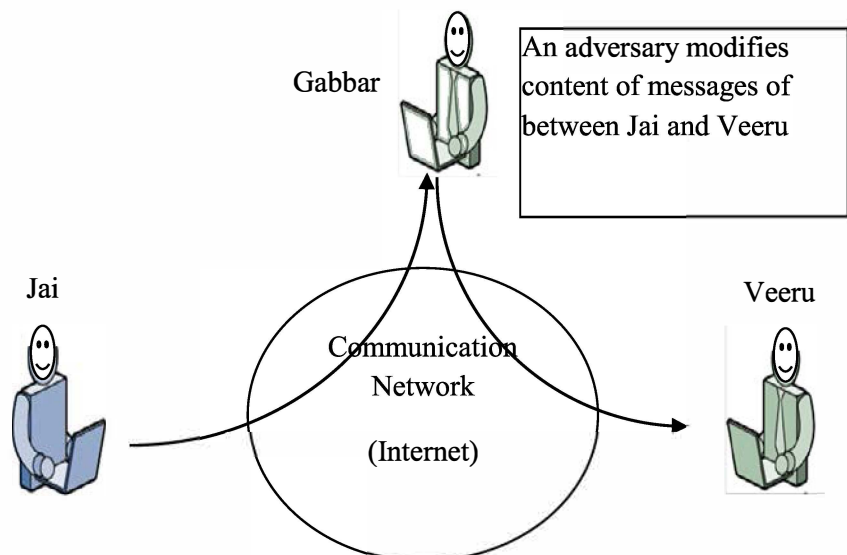


Figure 1.9: Active attack - message modification

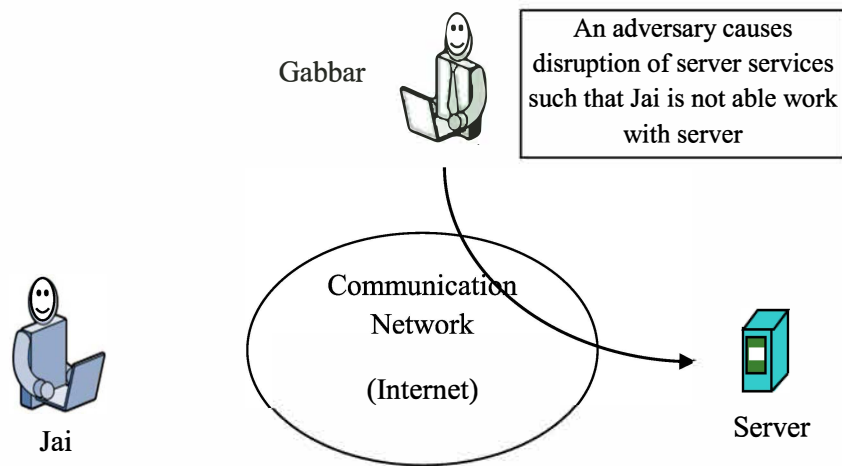


Figure 1.10: Active attack - Denial of service

Check your progress 2

Prevention of passive attack is easy while detection of passive attack is difficult as compared to active attacks. Why?

1.8 SECURITY MECHANISMS AND SECURITY SERVICES (X.800)

Security services and mechanism for implementing the security services are provided by International Telecommunication Union-Telecommunication Standardization Sector (ITU-T).

Security services

ITU-T (X.800) categorized 14 specific services into five categories (Figure 1.11) namely data confidentiality, data integrity, data authentication, non-repudiation and access control.

Authentication: provides assurance regarding the claim of communicating party. For connection oriented communication it authenticates the sender or receiver (peer entity authentication) whereas for connectionless communication it authenticates the source (data origin authentication).

Access control: protects information against unauthorized access. It defines who can access what and what can be done by him.

Data confidentiality: provides protection of message from disclosure to unauthorized users. It provides protection to data sent over a connection (connection confidentiality) or to single data block

(connectionless confidentiality) or to selected fields (selective-field confidentiality) or to particular traffic flows (traffic flow confidentiality).

Data Integrity: protects from message modification, insertion, deletion, or replays. Provides guarantee that the received data is one and same as that the send data. During connection oriented communication it detects that any modification, insertion, deletion or replay has taken place (connection integrity without recovery). If modification is detected then recovery may be attempted (connection integrity with recovery). It may provide data block integrity (connectionless integrity). Integrity may also be checked for selective field (selective field connection integrity or selective field connectionless integrity).

Non repudiation: provides protection against - denial by a user or entity (repudiation) i.e., it provides proof of identity of sender (non-repudiation origin) or that of receiver (non-repudiation destination).

Security Mechanisms

ITU-T (X.800) provides recommendation for security mechanisms that can provide security services.

Specific security mechanisms that may be incorporated into appropriate protocol layer:

Encipherment: provides confidentiality by either hiding (cryptography) or covering (steganography) data.

Data integrity mechanism: user evaluates value (hash) from message and appends this to message. Receiver recalculates hash and verifies that message is not tampered i.e. its integrity is preserved.

Digital signature: Electronic signature (a value) is calculated using secret known to sender only (private key of sender) and it is attached to message. Receiver can later verify the electronic signature via information known to him to him previously (public key). Digital signature verification also ensures that the message is not tampered as the hash value is also checked during this process.

Authentication exchange: message exchange between two parties over a communication medium for providing one's identity to other.

Traffic padding: inserting randomly generated data into actual data traffic to nullify the traffic analysis effect.

Routing control: changing route for preventing eavesdropping on a particular route.

Notarization: involving a trusted third party to guarantee certain properties during communication.

Access control: utilizes the password or PINs etc. to identify the

user's access rights.

Pervasive security mechanisms those are not specific to any particular protocol layer:

Trusted functionality: that which can be identified as correct and trusted.

Security labels: marking for naming or designating a security attribute of a resource.

Event detection: identifying security related events.

Security audit trail: data collection, independent review and examination of system from security view point (security audit).

Security recovery: recovery actions based upon event handler and management function request.

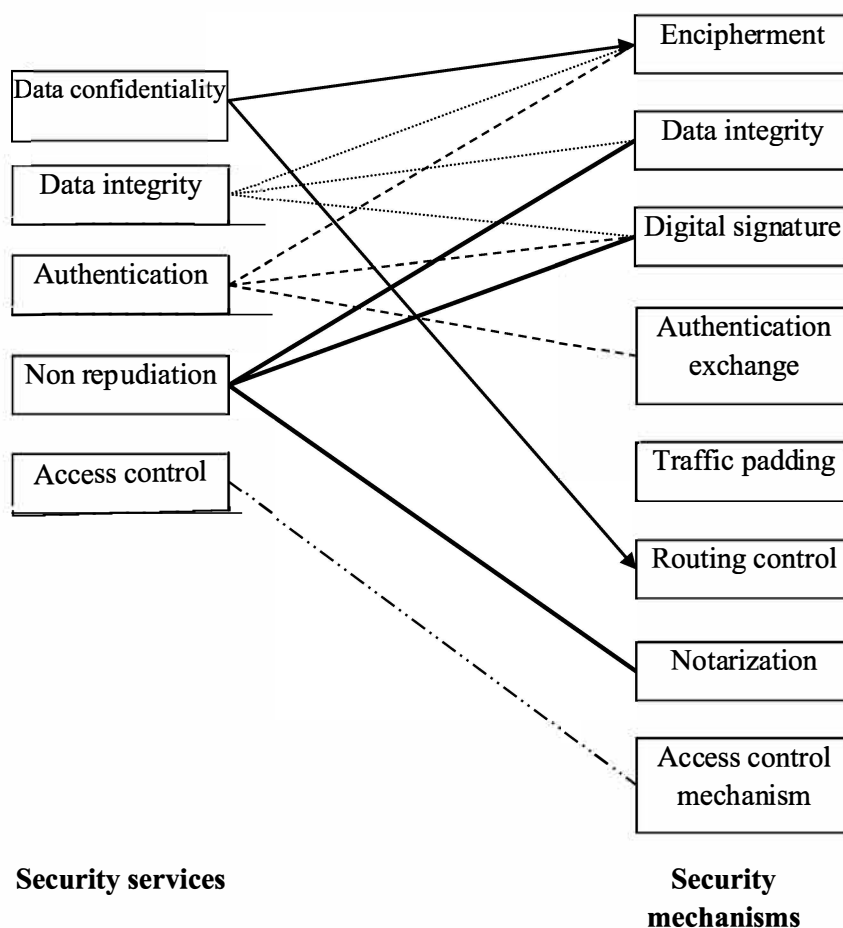


Figure 1.11: relationship between security services and mechanisms.

Figure 1.11 clearly indicates that relationship between services and mechanism is one-to-many and vice-versa.

1.9 BASIC NETWORK SECURITY MODEL

A generalized model for network security is shown in Figure 1.12. The two communicating parties co-operate each other for the secured transformation. The two parties create logical channel over insecure medium using communication protocol like TCP/IP. An adversary poses threat to ongoing communication. The security techniques that provide protection to ongoing communication should have following two components:

1. Application of transformation to the message so that message becomes unreadable. Appending code to the message for verifying the sender's identity.
2. Use of secret key by the parties in the transformation for coding/decoding information.

A trusted third party is involved who is responsible for secret key distribution and ensuring trust between the communicating parties.

Thus four basic tasks are involved in designing a particular security service are:

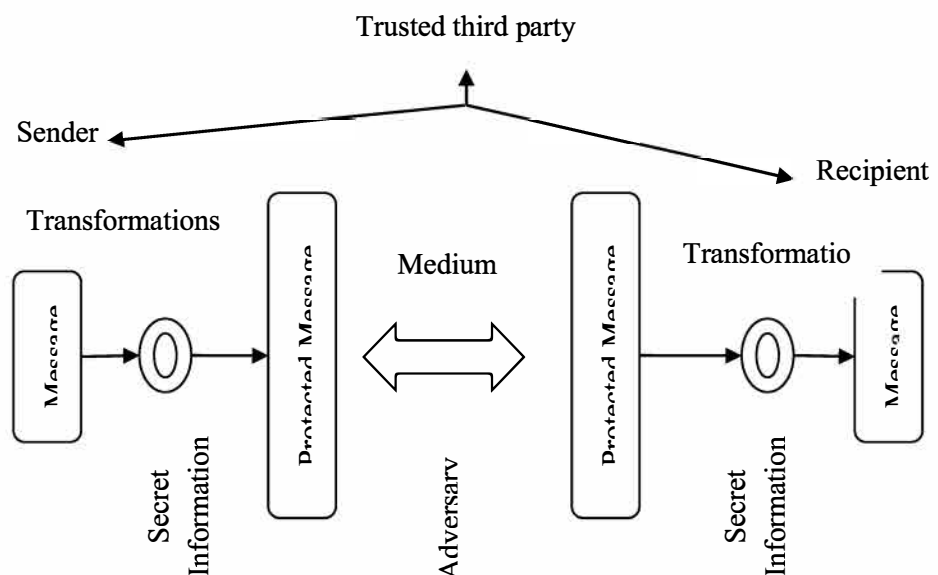


Figure 1.12: Network security model

1. Use of algorithm for transformation
2. Generation of secret key
3. Use of key distribution and sharing methods

4. Use of protocol for applying algorithm and key for transformation to achieve a particular security service.

Another model for protecting an information system from unwanted access is shown in Figure 1.13. Several adversaries including humans (hackers/disgruntled employees) and softwares (viruses/worms) try to gain unwanted access into an information system. The security mechanisms that protect against such access are of two types: (1) password based login procedures (gatekeeper) that provide access to only authorized users, (2) screening logic that identifies and filters viruses/worms. Screening logic searches the software codes that are able to bypass the password based login security. It constantly monitors system activity and analyze stored information for detecting presence of viruses/worms [8].

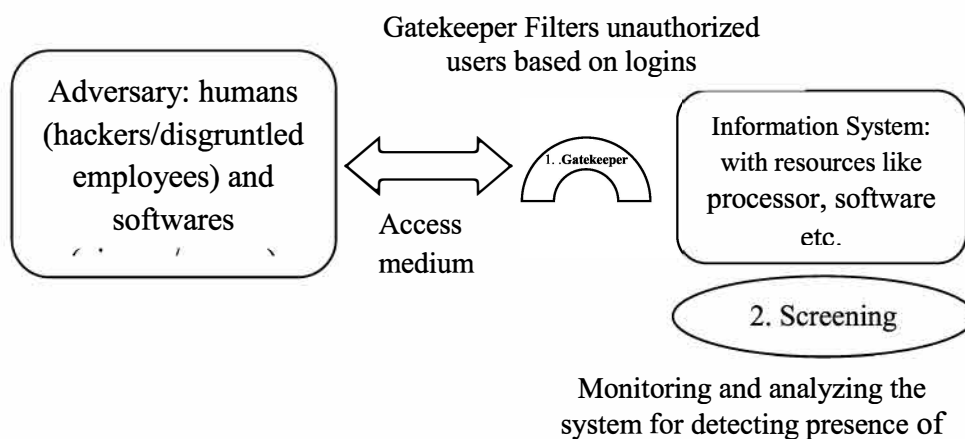


Figure 1.13: Network access security model

1.10 Summary

In this unit a brief introduction of information and network security is presented. Existing definitions of information security along with the prevailing information security model have been explained. The historical background along with the essential components and aspects of information security is also provided. The three essential things in security i.e., security attacks, security mechanisms and security services are elaborated. The relationship between security mechanisms and security services is also presented before the readers. At last, model for network security is covered.

Terminal Questions

1. *What are the overall goals of information security?*
2. *Differentiate between the following:*
 - a. *Authentication and Authorization*
 - b. *Active Attack and Passive Attack*

3. *Name and explain any four security services and security mechanisms*
4. *Show the relationship between security services and mechanisms in matrix form.*
5. *Information security is not always desired. Justify.*
6. *How denial of service attack causes danger to availability?*

References:

1. Overview of cryptography, PKI Outreach Programme (POP) Nationwide Awareness Programme, Centre for Development of Advanced Computing (C-DAC), Electronics City, Bangalore, 2012.
2. Principles of Information Security, Chapter 1 (Introduction of Information Security), 4th Edition, Michael E. Whitman, Herbert J. Mattord, Cengage Learning, 2012.
3. Information Security - Wikipedia, the free encyclopedia, accessed on 20/09/2016.
4. Committee on National Security Systems: National Information Assurance (IA) Glossary, CNSS Instruction No. 4009, 26 April 2010.
5. Anderson, J. M. (2003). "Why we need a new definition of information security". *Computers & Security*, 22(4), 308–313.
6. McCumber, John. "Information Systems Security: A Comprehensive Model." Proceedings of the 14th National Computer Security Conference, National Institute of Standards and Technology, Baltimore, MD, October 1991.
7. Schneier, Bruce (1997). "Why Cryptography is Harder than it looks", Counterpane Systems.
8. Cryptography and Network Security, Chapter 1 (Introduction), William Stallings, Pearson Education, Fourth Edition, 2010.

UNIT-2

Classical Encryption Techniques

Structure

- 2.0 Introduction
- 2.1 Objectives
- 2.2 Historical Background
- 2.3 Model for Symmetric Encryption
- 2.4 Classical Substitution Cipher Techniques
- 2.5 Transposition Cipher Techniques
- 2.6 Understanding Steganography
- 2.7 Summary
- 2.8 Terminal Questions

2.0 INTRODUCTION

In previous unit of this block, you learned about introductory definitions and concepts of information and network security. In this unit, the historical background of the classical encryption techniques is given. It is followed by the explanation of the symmetric cipher model. Two categories of traditional cipher i.e., substitution and transposition ciphers are explained. Towards end of the unit steganography is discussed. Readers are also advised to go through the appendix 2A for prominent sets used in the field of cryptography and their associated definitions.

2.1 OBJECTIVES

This unit deals with the classical encryption techniques. At the end of this unit, you will be able to

- define the terms and the concepts of symmetric key ciphers.
- learn the two categories of traditional ciphers: substitution and transposition ciphers.
- describe the categories of cryptanalysis used to break the symmetric ciphers.
- introduce the concepts of the stream ciphers and block ciphers.
- lay the foundation for study of modern cryptography.

2.2 HISTORICAL BACKGROUND

Some of the initial work related to cryptography in the ancient times was discussed in introductory part of unit I. Few other noticeable historical events related to information security are listed in Table 2.1. It is clear from the table that the journey of cryptography started with symmetric ciphers. An important issue raised by most of the symmetric key ciphers was – how the two communicating parties can share the key to be used further in symmetric ciphers. The Diffie-Hellman key exchange protocol was developed with the notion of strengthening the encryption process and it was realized that the protocol can contribute towards deriving shared key between the two communicating entities. This protocol is used till date for exchanging symmetric key between the two entities. The data encryption standard (DES) algorithm was adopted in the year 1977 by National Institute of Standards and Technology (NIST). It was shown through analysis that this algorithm may be broken easily [1]. Diffie's work contributed towards evolving a new notion in cryptography that is popularly known as asymmetric cryptography or public key cryptography. Encouraged by their work Ron Rivest, Adi Shamir and Len Adleman at MIT developed a new asymmetric scheme in 1977. This scheme was named as Rivest-Shamir-Adleman (RSA) scheme. It is one of the most widely accepted scheme in the field of cryptography. In 1999, NIST issued a document stating that DES is to be used with legacy systems instead a new variant termed as triple DES (3DES) is going to be used. The journey of 3DES was not very heartening and soon NIST asked for a proposal for new standard (termed as Advanced Encryption Standard) whose security and other features were not less than 3DES. Out of 15 proposals submitted, NIST selected Rijndael as the AES (Advanced Encryption Standard) standard in the year 2001. AES was supposed to replace DES and 3DES symmetric ciphers. But as of date DES and 3DES are still used in some applications and entities [2].

Table 2.1: Historical timeline for some of the ciphers [2][3]

Year	Event
1585 AD	Blaise de Vigenère discussed Vigenere cipher
1917 AD	American, Gilbert S. Vernam, develops the One-time-pad
1976 AD	Diffie-Hellman key exchange protocol was developed
1977 AD	Symmetric key cipher DES was developed by IBM
1977 AD	Asymmetric key cipher RSA was developed
2001 AD	AES was chosen as the successor to DES

2.3 MODEL FOR SYMMETRIC ENCRYPTION

Before moving into the functioning of symmetric cipher model, first let us understand the basic terminology prevailing in the field of cryptography [3].

- **Plaintext:** Also referred to as clear text. It is basically the original message that is required to be kept safe and secure i.e., one on which encryption is to be carried out.
- **Ciphertext:** the scrambled or unintelligent message generated after applying encryption.
- **Enciphering or encryption:** the process of scrambling the message or making message unintelligent.
- **Encryption algorithm:** algorithm that performs encryption, has two inputs: a plaintext and a secret key
- **Deciphering or decryption:** gaining back the plaintext from ciphertext.
- **Secret key:** a number or a value (say) that is selected for encryption and decryption. As same key is used for encrypting as well as decrypting, it is termed as a symmetric key
- **Cipher or cryptographic system:** The complete mechanism of encryption and decryption.
- **Cryptography:** science dealing with study and application of ciphers.
- **Cryptanalysis:** science of studying attacks against or breaking into cryptographic systems.
- **Cryptology:** combination of cryptography and cryptanalysis

Figure 2.1 shows a representative model for a traditional encryption mechanism. Here, the hiding process or user utilizes an encryption algorithm that takes plaintext and shared symmetric key as input, performs processing and gives output in terms of scrambled or unintelligent text (ciphertext). The opponent or attacker is not in possession of key and hence is unable to understand anything from it due to which message remains safe. The other process or user who is in possession of key can anytime perform the reverse mechanism i.e. utilize the decryption algorithm that takes ciphertext and key as input and generates/extracts back the plaintext.

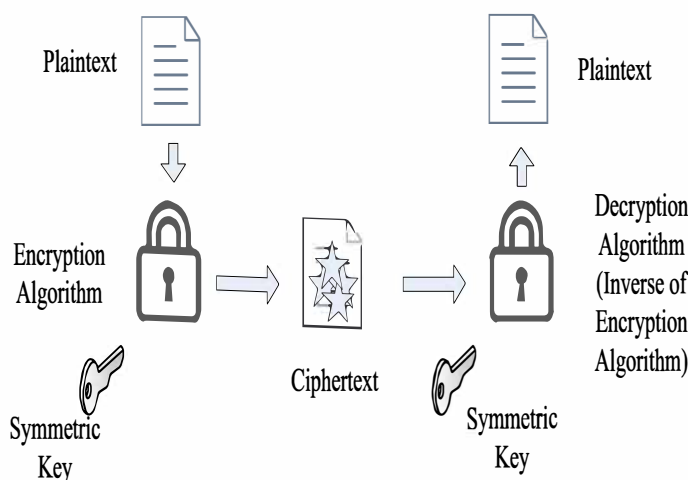


Figure 2.1: A representative model for traditional encryption mechanism

There are two requirements for secure encryption.

- (1) The encryption algorithm is known to everyone in public whereas key is hidden and is shared only by the communicating parties. It is almost impossible or impractical for anyone to gain plaintext message back from ciphertext.
- (2) The key* are obtained in secure manner and are kept secure either with the help of software or hardware.

* Keys are usually contained in data dongle or pen drives. Even some of the vendors have incorporated products with low cost chips for the data algorithms for making encryption efficient.

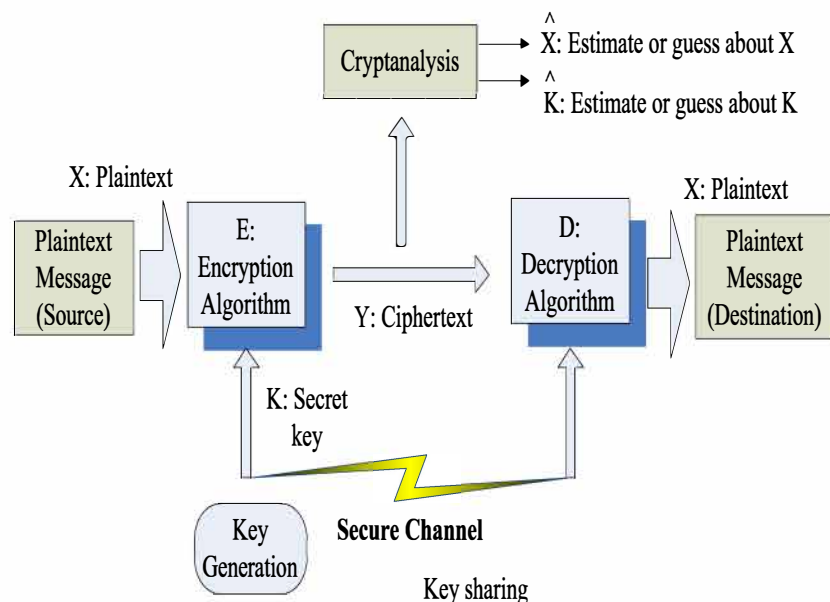


Figure 2.2: A complete symmetric cipher model

Figure 2.2 depicts the complete symmetric cipher model. It can be seen that plaintext X is encrypted at source i.e., converted to ciphertext Y by application of encryption algorithm E . This ciphertext is converted back into plaintext X by application of decryption algorithm D . The symmetric key K is first generated (either at source or by third party) and is later shared between the source and destination on a network via a secure channel. The attacker or the eavesdropper may perform cryptanalysis using ciphertext Y and encryption algorithm E and may try to evaluate key K or plaintext X .

Mathematically, the functionality of model depicted in figure 2.2 may be summed as:

$Y = E_K(X)$ [X encrypted using key K and resultant is ciphertext Y]

$X = D_K(Y)$ [Y decrypted using key K and resultant is plaintext X]

Where, both E (encryption algorithm) and D (decryption algorithm) are known to public.

Ciphers

There are two kinds of ciphers: symmetric cipher and asymmetric cipher. In former, two keys are utilized during encryption and decryption. Both are basically one and same (or common). Strength of this cipher rest upon the secrecy of these shared keys. All classical encryption algorithms discussed in this text material are symmetric. Symmetric cipher is further classified into two categories: First, block cipher that usually takes one block of data (typically 64 or 128 bits) at a time and produces encrypted output. Second, stream cipher that takes data one bit or one byte at a time and produces encrypted data in a continuous manner. In latter, the keys utilized during encryption and that during decryption are different. The key used during encryption is termed as public key while that used during decryption is termed as private key. The strength of this cipher rest upon the secrecy of private key and it is basically the responsibility of the owner to keep it safe.

Cryptanalysis

In cryptanalysis, the attacker tries to break into or tries to weaken the cryptographic system by either recovering the plaintext or key. According to Kerkhoff's principle the attacker knows all the details but not the secret key. Cryptanalysis has two general approaches: (a) brute-force attack (b) non-brute-force attack (cryptanalytic attack).

Brute-Force Attack is the most common and easy analysis performed by the attackers to find out the plaintext. In this, attacker tries all possible key for decrypting the ciphertext. The one that generates the meaningful text is identified as the possible key. Trying large number of combination means for large size key this attack is not much useful as the time required by the involved computations is almost impractical. For example, for 128 bit key length, $2^{128} = 3.4 \times 10^{38}$ keys are possible. Half of the key space needs to be tried on an average for getting the exact key match. If time required is 1 decryption/ μ s then applying brute force attack will mean that $2^{127} \mu\text{s} = 5.4 \times 10^{24}$ years will be required. For faster computer (10^6 decryptions/ μ s), it will take 5.4×10^{18} years.

Cryptanalytic Attacks

Cryptanalytic attacks may be classified [4] into the following:

- (a) Ciphertext-only attack
- (b) Known-plaintext attack
- (c) Chosen-plaintext attack
- (d) Chosen-ciphertext attack

Ciphertext-only attack

Finding out plaintext by an attacker on the basis of only ciphertext analysis is termed as ciphertext-only attack (Figure 2.3). If

such an attack is a possibility against any encryption mechanism then that mechanism is considered as completely insecure.

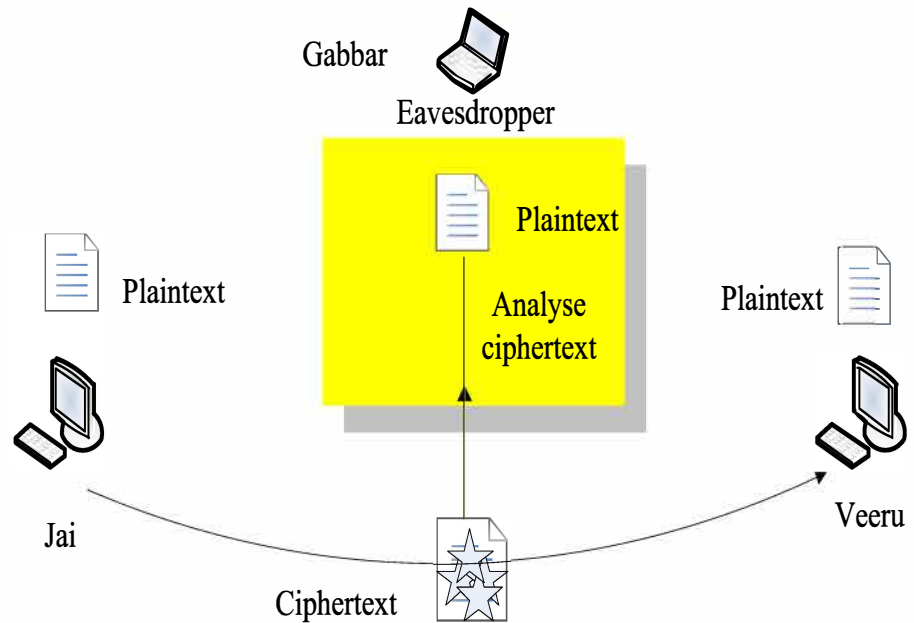


Figure 2.3: Cipher text only attack

Known-plaintext attack

Finding out plaintext and secret key by an attacker on the basis of analysis of new ciphertext and previous plaintext-ciphertext pairs is termed as known-plaintext attack (Figure 2.4).

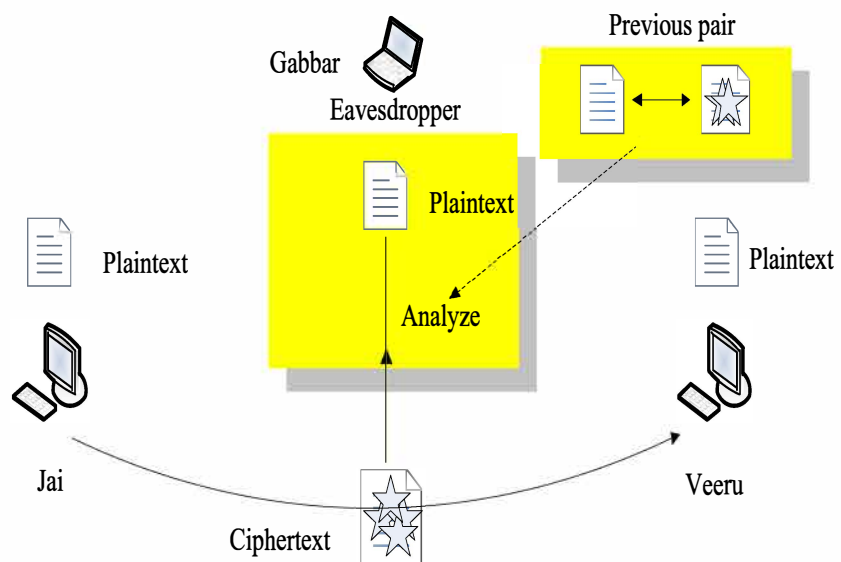


Figure 2.4: Known plaintext attack

Chosen-plaintext attack

Finding out plaintext and secret key by an attacker on the basis of analysis of chosen plaintexts (along with corresponding ciphertexts) and new ciphertext is termed as known-plaintext attack (Figure 2.5).

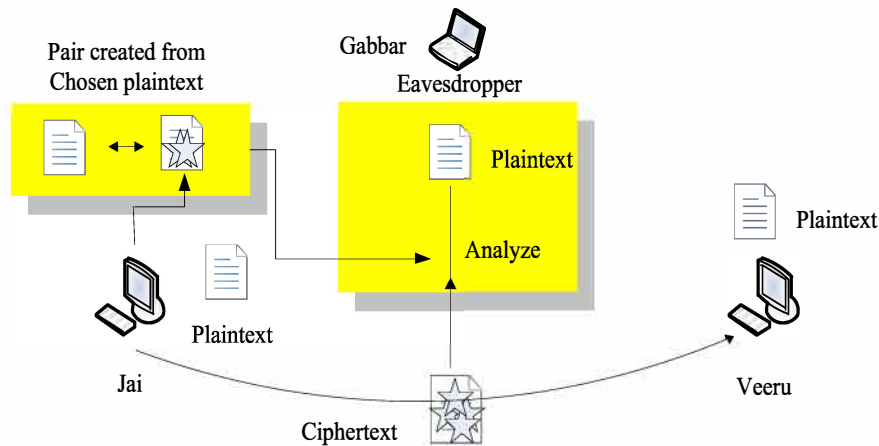


Figure 2.5: Chosen plaintext attack

Chosen-ciphertext attack

Finding out plaintext and secret key by an attacker on the basis of analysis of chosen ciphertexts whose plaintexts are known (i.e., pairs are created based upon chosen ciphertexts) and new ciphertext is termed as chosen-ciphertext attack (Figure 2.6).

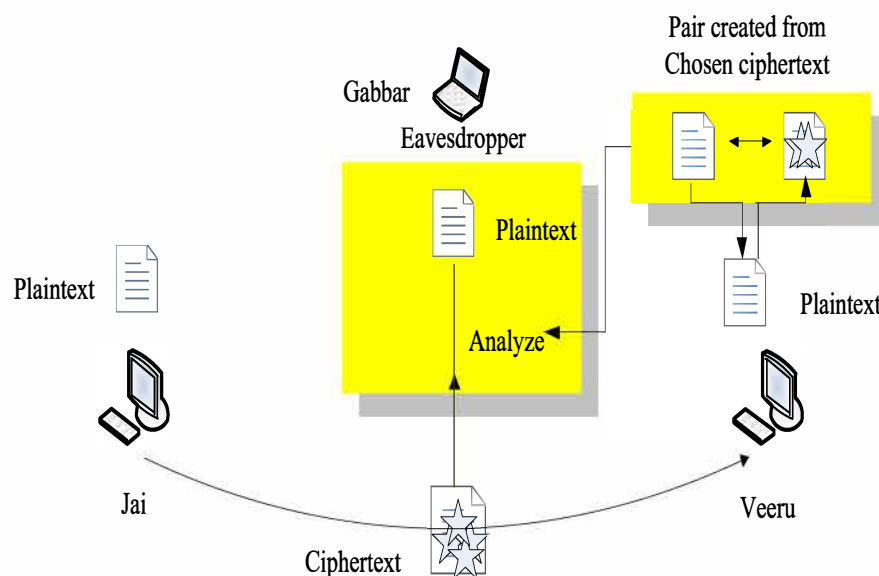


Figure 2.6: Chosen ciphertext attack

Classical Ciphers

In classical ciphers, the plaintext is considered as a sequence of elements like bits or characters. These ciphers fall in three categories: first - substitution cipher that replaces each element of the plaintext with another element, second - transposition cipher that deals with permutations i.e., rearrangements of the plaintext elements, third - product cipher that use several stages and combinations of substitutions and transpositions.

Check your progress 1

- a. What do we mean by encryption of text? When it is required?
- b. Define: cryptanalysis and Brute Force Attack.

2.4 CLASSICAL SUBSTITUTION CIPHER TECHNIQUES

The classical substitution ciphers are divided in two major categories: monoalphabetic and polyalphabetic ciphers. In monoalphabetic substitution, a plaintext character is replaced by single ciphertext character every time i.e., plaintext character to ciphertext character mapping is one-to-one. Whereas in polyalphabetic substitution, each occurrence of a character may be replaced by a different character i.e., plaintext character to ciphertext character mapping is one-to-many [3].

A. Monoalphabetic ciphers

Under the first category Caesar, multiplicative, affine and monoalphabetic substitution ciphers are presented.

1. Caesar Cipher

The Caesar cipher was developed by Julius Caesar for communicating with his officers and is considered as the oldest cipher. It provides a substitution method where each letter in the plaintext is substituted by another letter that lies three places apart (cyclically) in the English alphabet as shown below:

Plain letters : 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

To be substituted as : 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

Plaintext message: Hi this is demonstration of Classical Cryptography

Encrypted message: Kl wklv lv ghprqvwudwlrq ri Fodvvlfdofubswrjusdkb

As each letter is shifted by 3, the key in Caesar cipher is 3

A **general Caesar cipher** involves shifting by k letters (key = k) or addition of k letters. It is hence also referred to as **shift or additive cipher**. The plaintext, ciphertext, and key are integers in Z_{26} in Caesar cipher. This can be represented mathematically as:

Ciphertext c after applying encryption = $E_K(p) = (p + k) \bmod 26$

Plaintext p obtained after decryption = $D_K(c) = (c - k) \bmod 26$

Here, the mapping from letters to numbers is assumed with first letter 'a' mapped to 0 and last letter 'z' mapped to 25.

The Caesar cipher is prone to brute force attack. All the keys (starting from 0 – ending at 25) are tried one after the other and the key is said to be cracked (i.e., algorithm stops) when some meaningful text is gained from the brute force attack.

In this course material the encryption examples are studied and evaluated using CrypTool™ version 1.4.30 [5].

Multiplicative Ciphers

In a multiplicative cipher during encryption the plaintext is multiplied by key k and then mod is taken. While during decryption the ciphertext is multiplied by inverse of key k and then mod is taken.

This can be represented mathematically as:

Ciphertext c after applying encryption = $E_K(p) = (p \times k) \bmod 26$

Plaintext p obtained after decryption = $D_K(c) = (c \times k^{-1}) \bmod 26$

Here, plaintext and ciphertext are integers in Z_{26} whereas the key is an integer in Z_{26}^* (**Appendix 2A**). The set Z_{26}^* has only 12 members (1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25).

Affine Ciphers

In affine cipher a pair of keys are used where the first key (multiplicative key) lies in Z_{26}^* and while the second key (additive key) lies in Z_{26} . Hence, the key space is $12 \times 26 = 312$.

This can be represented mathematically as:

Ciphertext c after applying encryption = $E_K(p) = (p \times k_1 + k_2) \bmod 26$

Plaintext p obtained after decryption = $D_K(c) = ((c - k_2) \times k_1^{-1}) \bmod 26$

Monoalphabetic Substitution Cipher

The additive, multiplicative, and affine ciphers discussed till now have small key space and are hence very susceptible to brute-force attack. In monoalphabetic substitution cipher, each letter of the plaintext is replaced by a different random letter in the corresponding ciphertext. Thus, a total of $26! = 4 \times 10^{26}$ keys exists which makes it difficult to crack using brute force attack.

Examples (monoalphabetic substitution cipher)

Case 1 (Figure 2.7 (a))

key entry: remaining characters are filled in ascending order

Key: MYKEYMONOALPHABETIC

Information on the substitution encryption

Mapping of 26 letters:

From: ABCDEFGHIJKLMNOPQRSTUVWXYZ

To:

MYKEONALPHBTICDFGJQRSUVWXZ

Plaintext: Hi this is demonstration of Classical Cryptography

Ciphertext: Lp rlpq pq eoidcqrjmrpdc dn Ktmqqpkmt Kjxfrdajmflx

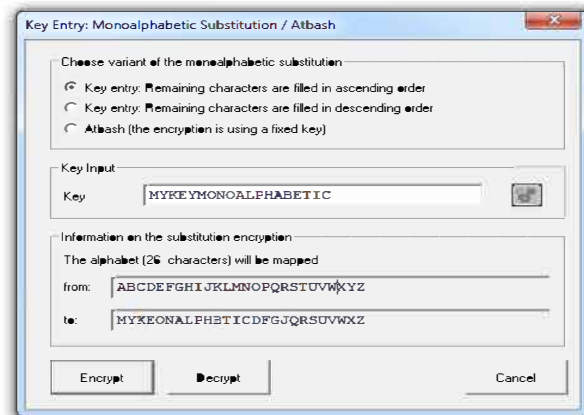


Figure 2.7 (a): Case 1

Case 2 (Figure 2.7 (b))

key entry: remaining characters are filled in descending order

Key: MYKEYMONOALPHABETIC

Information on the substitution encryption

Mapping of 26 letters:

From: ABCDEFGHIJKLMNOPQRSTUVWXYZ

To: MYKEONALPHBTICZXWVUSRQJGFD

Plaintext: Hi this is demonstration of Classical Cryptography

Ciphertext: Lp slpu pu eoizcu

Ciphertext: Lp rlpq pq eoidcqrjmrpdc dn Ktmqqpkmt Kjxfrdajmflx

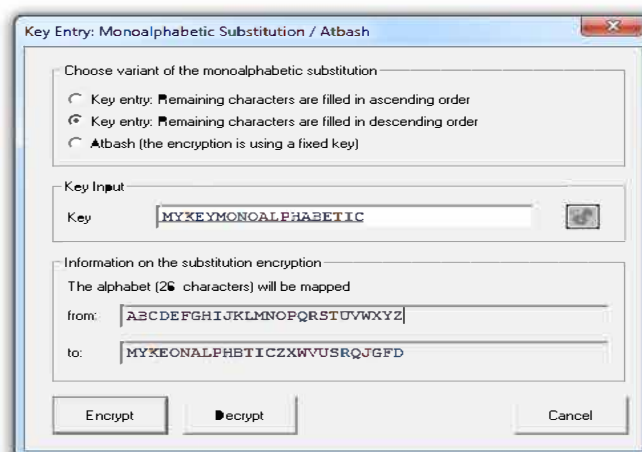


Figure 2.7 (b): Case 2

Case 3 (Figure 2.7 (c))

Atbash (the encryption is using a fixed key)

Key: ZYXWVUTSRQPONMLKJIHGFEDCBA

Information on the substitution encryption

Mapping of 26 letters:

From: ABCDEFGHIJKLMNOPQRSTUVWXYZ

To: ZYXWVUTSRQPONMLKJIHGFEDCBA

Plaintext: Hi this is demonstration of Classical Cryptography

Ciphertext: Sr gsrh rh wvnlmhgizgrlm lu Xozhhrxzo Xibkgltizksb

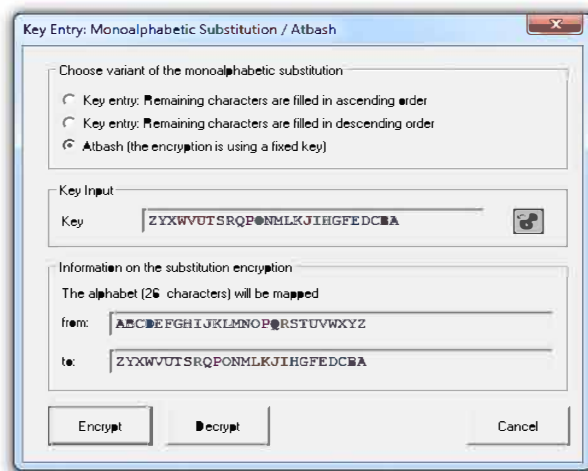


Figure 2.7 (c): Case 3

Letters in English language are not random rather they occur with some frequency distribution with E letter enjoying highest frequency followed by T, R, N, I, O, A, S. In comparison, other letters like Z, J, K, Q, X are fairly infrequent. Similar frequency behavior is observed for double & triple letters. In Double letters, the frequencies in decreasing order are as:

th he an in er re es on, ...

In triple letters, the frequencies in decreasing order are as:

the and ent ion tio for nde, ...

As shown in encryption cases above, monoalphabetic substitution does not change relative frequencies of letters. Hence, it is not secure against some other cryptanalytic attacks because of the fact that the attacker can calculate letter frequencies for ciphertext and compare with the distribution against the known frequencies.

In Monoalphabetic substitution Cryptanalysis is done utilizing the following steps:

STEP 1: In the given ciphertext count the relative letter frequencies.

STEP 2: Assume those letters as e, t (the most occurring plaintext letters) and similarly other ciphertext letters may be mapped.

STEP 3: Repeat the step 2 for double and triple letter combinations.

STEP 4: Proceed with trial and error finally get the plaintext.

B. Polyalphabetic Substitution Ciphers

Polyalphabetic substitution ciphers use a sequence of monoalphabetic ciphers ($M_1, M_2, M_3, \dots, M_k$) for encrypting characters. The key determines which sequence of ciphers to use. Each plaintext letter has multiple corresponding ciphertext letters e.g., a letter say, p is decoded as m using one key and next time it is decoded as some other character using other key. Thus, cryptanalysis becomes harder since the letter frequency distribution will now involve less variation.

As example of polyalphabetic ciphers Autokey, Playfair, Hill, Vigenère ciphers are discussed in this section.

Autokey Cipher

Autokey cipher is the simplest polyalphabetic cipher where plaintext is considered as part of key after the first key character. It is named as autokey as the subkeys are automatically created from plaintext. Hence, it is able to generate different characters at different time intervals.

It can be represented mathematically as:

Plaintext $p = p_1 p_2 p_3 \dots$

Ciphertext $c = c_1 c_2 c_3 \dots$

Key $k = (k_1, p_1, p_2, \dots)$

Ciphertext c after applying encryption = $E_K(p) = (p_i + k_i) \bmod 26$

Plaintext p obtained after decryption = $D_K(c) = (c_i - k_i) \bmod 26$

This scheme is also vulnerable to statistical cryptanalysis as plaintext and key has same letter frequency distribution.

Playfair Cipher

Monoalphabetic cipher does not provide much security even though key space is increased. Hence, Charles Wheatstone in 1854 invented a new technique for strengthening the security where multiple letters are encrypted at one time. This technique was named after his friend Baron Playfair. Playfair Key uses a 5×5 matrix (or table), the letters of key are filled into the matrix row wise from left to right. While filling the matrix, duplicates in the key are not considered and these duplicates are dropped. The remaining matrix is filled with other letters. Both "I" and "J" are placed in the same matrix cell. The matrix is considered as the cipher key.

For encrypting the message, groups of 2 letters are formed for the entire message e.g. "Hi this is demonstration of Classical Cryptography" becomes "HI TH IS IS DE MO NS TR AT IO NO FC LA SX IC AL CR YP TO GR AP HY". These are then converted to ciphertext by mapping on the key table. The mapping means application of the following rules, in order, to each pair of letters in the plaintext:

- (i) In case both letters of a pair are the same or only one letter is left in the end of pairing of plaintext, an "X" is added after the first letter. The newly pair is encrypted and the process is continued.

- (ii) In case the letters lie on the same row they are replaced with the letters to their immediate right cyclically i.e., wrapping around to the left side is done.
- (iii) In case the letters lie on the same column they are replaced with the letters immediately below them cyclically i.e., wrapping around to the top side is done.
- (iv) Cases not covered above in (ii) and (iii) i.e., the letters do not lie on the same row or column, replace them with the letters lying on the corners of the rectangle defined by the original pair. The first encrypted letter of the pair is the one that lies on the same **row** as the first plaintext letter.

Demonstration of the rules

Assume one wants to encrypt the digraph AT. There are three general cases:

- 1)

```
*****
* AMTN
*****
*****
*****
```

Hence, AT -> MN

2)

```
** A **
** N **
*****
** T **
** M **
```

Hence, AT -> NM

3)

```
N**A*
*****
*****
T**M*
*****
```

Hence, AT -> NM

For decryption, inverse of these 4 rules is implemented.

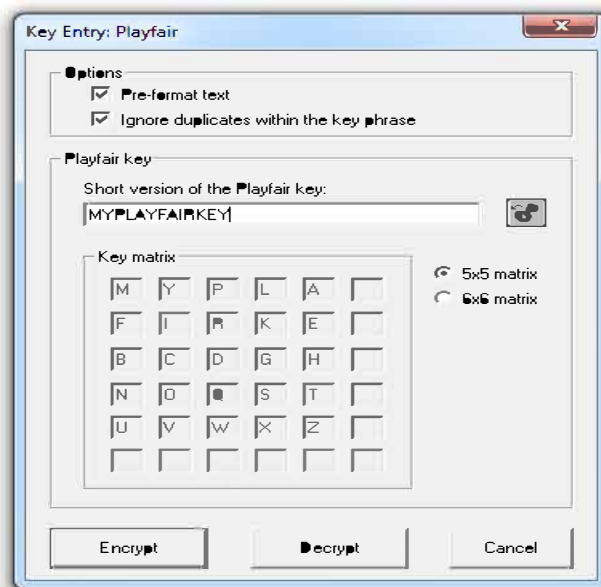


Figure 2.8: Key matrix playfair cipher

Example

Using "MYPLAYFAIRKEY" as the key, the table/matrix is as shown in Figure 2.8.

Plaintext: Hi this is demonstration of Classical Cryptography

Pairing: HI TH IS IS DE MO NS TR AT IO NO FC LA SX IC AL
CR YP TO GR AP HY

Ciphertext:

CEZTKOKOHRYNOTQEEZCVOQIBAMXLOKHYYGIPAQSCEP
ADLV

Security and cryptanalysis of Playfair Cipher

Playfair Cipher was one of the most widely used ciphers in world war I & II. It has better security in comparison to monoalphabetic cipher. It may be treated equivalent to a monoalphabetic cipher having an alphabet of $26 \times 26 = 676$ characters. It leaves some structure of plaintext unchanged and therefore, it can be easily broken if enough text is there. If both plaintext and ciphertext are known key may be obtained fairly easily. When only the ciphertext is known, brute force is required that involves searching through the key space for matches between the frequency of pairs of letters between plaintext and ciphertext.

Hill Cipher

The Hill cipher is a polygraphic substitution cipher based on linear algebra. A polygraphic cipher is the one that operates on groups of letters.

Hill encryption consists of the following parts:

1. Encoding the alphabet characters to numbers by encoding the first alphabet character as number 0.
2. Selecting Hill key matrix

A Hill key matrix is selected in a manner such that it has a multiplicative inverse.

$$K = \begin{bmatrix} k_{11} & k_{12} & \dots & k_{1m} \\ k_{21} & k_{22} & \dots & k_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ k_{m1} & k_{m2} & \dots & k_{mm} \end{bmatrix}$$

Assume that the 2x2 selected Hill matrix for encryption is:

$$H[t] = \begin{bmatrix} & Q & Z \\ & B & A \end{bmatrix}$$

Encoding $H[t]$ to numbers:

$$H[n] = \begin{bmatrix} & 16 & 25 \\ & 01 & 00 \end{bmatrix}$$

The Hill matrix $D[n]$ for decryption is inverse of the $H[n]$ matrix i.e., $H[n] * D[n] = I$ (Identity matrix with all 1's along diagonal otherwise 0). A matrix with no inverse cannot be used as key in Hill cipher.

$$D[t] = \begin{bmatrix} & A & B \\ & Z & Q \end{bmatrix}$$

Encoding $D[t]$ to numbers:

$$H[n] = \begin{bmatrix} & 16 & 25 \\ & 01 & 00 \end{bmatrix}$$

3. Hill encryption/decryption

For the Hill encryption and decryption the Hill matrix is multiplied by plaintext as shown below:

$$C_1 = P_1k_{11} + P_2k_{21} + \dots + P_mk_{m1}$$

$$C_2 = P_1k_{12} + P_2k_{22} + \dots + P_mk_{m2}$$

.....

$$C_m = P_1k_{1m} + P_2k_{2m} + \dots + P_mk_{mm}$$

In the example, plaintext is multiplied by a column vector from right.

Input plaintext - Hi this is demonstration of Classical Cryptography.

The plaintext consists of 50 characters. There are 6 non-alphabet characters (non-alphabet characters are ignored.)

Below is the demonstration of the Hill encryption and decryption on the first 2 characters of the given plaintext: "HI":

Hill encryption

The plaintext vector P is below encoded to numbers:

$$P[n] = [07 \quad 08]$$

Computation of the Hill encryption by matrix*vector product (the vector P[n] is a column vector):

$$A \quad <-- \quad 00 = 16 * 07 + 25 * 08 \pmod{26}$$

$$H \quad <-- \quad 07 = 01 * 07 + 00 * 08 \pmod{26}$$

The Hill ciphertext is: "AH".

Complete Cipher Text generated by Hill cipher: Ah ltgi gi
sdwminbthakim na Fulkswipa Pcfyebtglakh

Hill decryption

Decryption of the ciphertext: "AH":

Ciphertext vector C decoded to numbers:

$$C[n] = [\quad 00 \quad 07 \quad]$$

Computation of the Hill decryption by the row*column matrix*vector product (the vector C[n] is a column vector):

$$H \quad <-- \quad 07 = 00*00 + 01* 07 \pmod{26}$$

$$I \quad <-- \quad 08 = 25*00 + 16* 07 \pmod{26}$$

The Hill plaintext is: "HI"

Vigenère Cipher

Vigenère cipher is the simplest polyalphabetic substitution cipher that considers the set of all Caesar ciphers $\{ C_a, C_b, C_c, \dots, C_z \}$. Considering key as *mykeyvigenere*, each letter of the plaintext is substituted using each letter from $C_m, C_y, C_k, C_e, C_y, C_v, C_i, C_g, C_e, C_n, C_e, C_r, C_e$ in cyclic manner. Decryption simply works in reverse. As shown in Figure 2.9, Vigenere cipher can be visualized as

combinations of m additive ciphers. It may be interpreted easily that the additive cipher is a special case of Vigenere cipher in which $m = 1$.

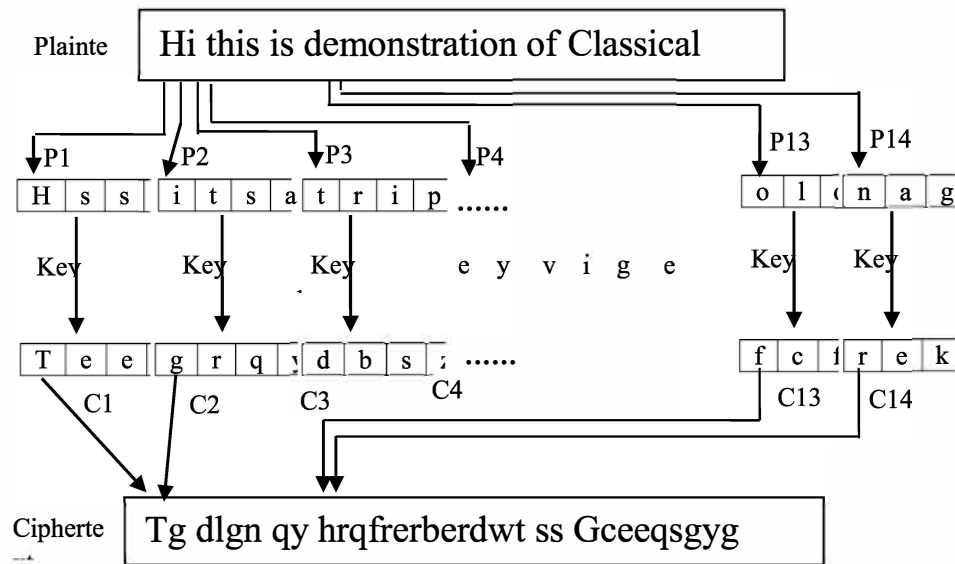


Figure 2.9: Vigenère Cipher

Example (Figure 2.9)

Key word: mykeyvigenere

key: mykeyvigenere mykeyvigenere mykeyvigenere mykey

plaintext: Hi this is demonstration of Classical Cryptography

ciphertext: Tg dlgn qy hrqfrerberdwt ss Gcee qsgyg Kxcxfkdyzlw

Security of Vigenère Ciphers

There are multiple ciphertext letters corresponding to each plaintext letter. So, letter frequencies are reduced but not eliminated. For breaking a Vigenere cipher, first the key length needs to be guessed. For key length m , the cipher consists of m Caesar ciphers. Plaintext letters at positions k , $m+k$, $2m+k$, $3m+k$, etc., are encoded using the same cipher.

Hence, each individual cipher may be attacked by forming their group.

The Plaintext words separated by multiples of the key length are encoded in the same way and this makes guessing the key length easy. For example, if plaintext = "...thexxxxxthe..." then "the" will be encrypted to the same ciphertext words. A look at the ciphertext for repeated pattern. e.g. repeated "LSO" in the following ciphertext example suggests the key length is 3 or 9. Such repetition could be by chance i.e., it may or may not be present in the selected case.

ciphertext: ZICLSOQN GRZGLSOAVZHCQYGLMGJ

One-Time Pad

Cryptography strives for perfect secrecy. Shannon's study points that perfect secrecy can be achieved if a randomly chosen key is chosen from the key domain. This means that there is no statistical relationship between plaintext and key. This idea is utilized by Gilbert Vernam in 1918. **Vernam cipher** (one time pad) considers binary data instead of character data. The ciphertext binary digit (C_i) for plaintext binary digit (P_i) may be calculated using i th binary key digit (B_i) as:

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

Decryption also uses XOR operation between C_i and K_i for getting plaintext back.

For increasing the key length Vernam suggested running loop of tapes which increases the strength of the scheme but it still fall under cryptanalysis.

An improvement (another one time pad) to Vernam cipher is suggested by Joseph Mauborne that considers new key each time for encrypting a new message and the length of key and message are considered equal. Thus, this method proves to be ultimate secure due to the randomness of the key. There are two difficulties in this method: one Generating large quantities of true random keys is problem. Second, key distribution securely among two parties is required.

Check your progress 2

- a. Use the affine cipher to decrypt the message "ZEBBW" with the key pair (7, 2) in modulus 26.
- b. Assume that Alice and Bob agreed to use an autokey cipher with initial key value $k_1 = 12$. Now Alice wants to encipher and send Bob the message "Attack is today". Enciphering is done character by character.

2.5 TRANSPOSITION CIPHER TECHNIQUES

A transposition cipher reorders symbols. It is of three types [4]:

- (i) Keyless Transposition Ciphers
- (ii) Keyed Transposition Ciphers
- (iii) Combining Two Approaches

The **keyless ciphers** permute the characters. In this permutation of characters, plaintext is written in one way and read in another way The permutation is applied on the whole plaintext. Transposition ciphers

initially started as keyless ciphers e.g., rail fence cipher which creates ciphertext by writing plaintext in two rows and then reading the pattern row by row. For example (Figure 2.10), to send the message “key is behind photo” to Bob, Alice writes

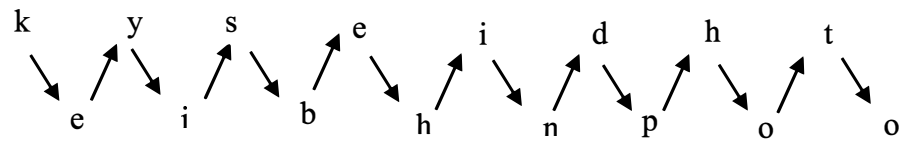


Figure 2.10: Rail fence cipher

Ciphertext in the above example is “KYSEIDHTEIBHNPOO”.

As another example shown below of keyless transposition cipher, the same plaintext is written row by row, in a table of four columns. The ciphertext is generated by reading columns one after the other.

K	E	Y	I
S	B	E	H
I	N	D	P
H	O	T	O

Ciphertext in the above example is “KSIHEBNOYEDTIHPO”.

The above cipher is a transposition cipher where each character in the plaintext is permuted into the ciphertext based on the positions. As shown in Figure 2.11 placement is done in such a manner that second goes to fifth position and third goes to the ninth position. The pattern after permuting is: (01, 05, 09, 13), (02, 06, 10, 14), (03, 07, 11, 15), and (04, 08, 12, 16). It can be easily interpreted that the difference between the two adjacent numbers is 4.

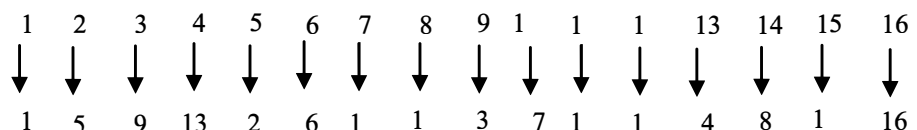


Figure 2.11: Ciphertext position permutations

In **keyed transposition ciphers**, plaintext is divided into groups of predetermined size (blocks) and then a key is used to permute the characters in each block separately.

The two approaches i.e., keyless and keyed transposition may be

combined as shown in Figure 2.12 [4].

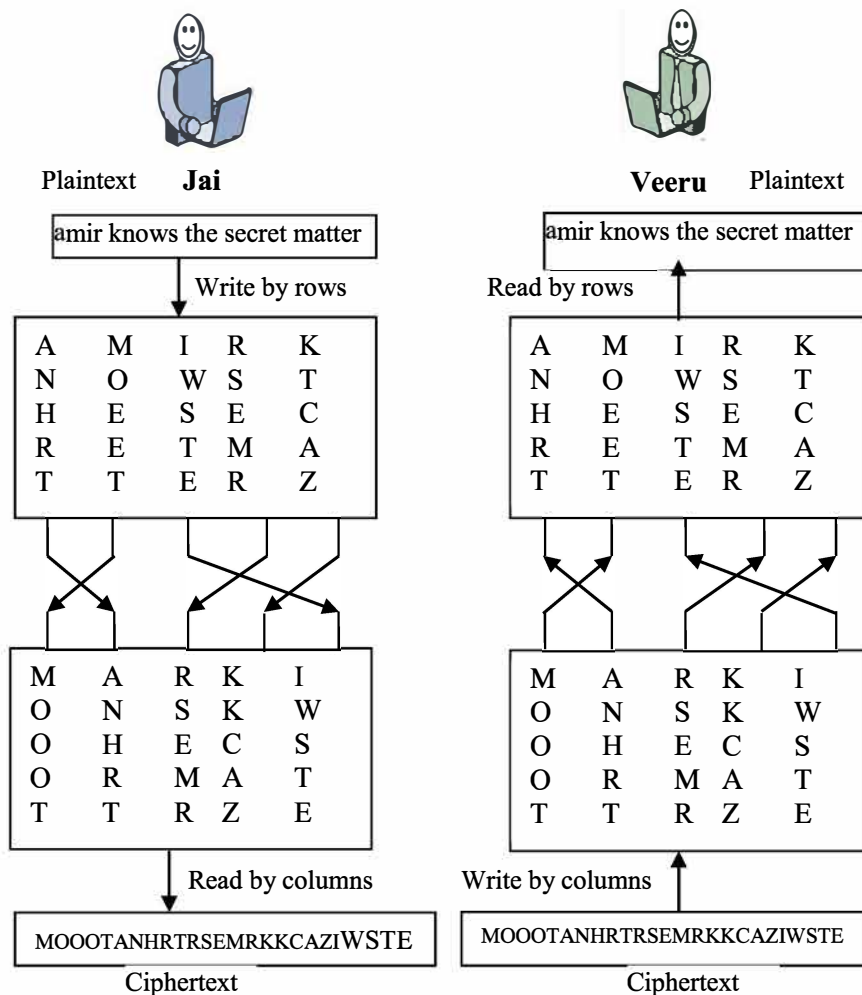


Figure 2.12: Combined approach Keys

In the above figure, a single key is used for encryption and for decryption. Formally, two key are required one for encryption and the other for decryption as shown below (Figure 2.13):

Encryption/decryption keys in transpositional ciphers

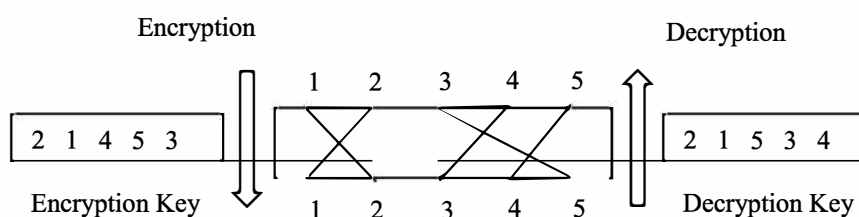


Figure 2.13: Encryption/Decryption key

The encryption/decryption process for a transposition cipher can also

be implemented using matrices (Figure 2.14).

Representation of the key as a matrix in the transposition cipher

$$\begin{array}{c}
 \boxed{\begin{matrix} 2 & 1 & 4 & 5 & 3 \end{matrix}} \\
 \Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow \quad \Downarrow \\
 \begin{bmatrix} 00 & 12 & 08 & 17 & 10 \\ 13 & 14 & 22 & 18 & 19 \\ 07 & 04 & 18 & 04 & 02 \\ 17 & 04 & 19 & 12 & 00 \\ 19 & 19 & 04 & 17 & 25 \end{bmatrix} \times \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 12 & 00 & 17 & 10 & 08 \\ 14 & 13 & 18 & 19 & 22 \\ 04 & 07 & 04 & 02 & 18 \\ 04 & 17 & 12 & 00 & 19 \\ 19 & 19 & 17 & 25 & 04 \end{bmatrix}
 \end{array}$$

Plaintext
Key
Ciphertext

Figure 2.14: Transposition cipher (matrix form)

Check your progress 3

- a. Apply the following operations in sequence on string – ABRACADABRA:
 - (i) apply permutation (31254).
 - (ii) apply substitution by 5 characters away from current character.
- b. Apply the following operations in sequence on string – PDLJDLXHVQC:
 - (i) apply permutation (34152).
 - (ii) apply substitution by 3 characters away from current character.

2.6 UNDERSTANDING STEGANOGRAPHY

Steganography literally means “covered writing”. The word itself is borrowed from the Greek language words: steganos and graphein. Former means - covered, concealed, or protected while latter means - writing.

Steganography refers to the process of hiding secret message in another message or media (termed as cover) in a manner such that existence of secret message remains unknown or unapparent to the attacker.

Digital media such as image, text, audio and video are generally used as cover for hiding purpose. Obviously, the cover is a pleasant looking cover media.

A major difference between cryptography and steganography is that the former changes the secret message but does not try to hide the

existence of the message while latter does not change the contents of the secret message but tries to conceal the existence of the message.

If cryptography is used along with steganography, the security of the system gets enhanced.

For such case when cryptography is used along with steganography two new terms: stego-key and stego are coined. Former refers to secret key used for encryption and decryption during the steganography while latter refers to the final steganography process output. Using this technique the message is first encrypted using stego key and then encoded into the cover. Stego-cover is then transmitted over an insecure channel.

Eavesdropper or attacker tries to decode and identify the secret message from the stego-cover during transmission but as the cover looks like a decent media and also the secret message is encrypted, this task becomes difficult.

Examples:

Figure 2.15 shows the data ‘01010010110’ hidden in a single line of text. The encoding is done by representing a single space as 0 while double spaces as 1.

This	text	deals	with	security	but	does	not	hide	anything	into	it.
0	1	0	1	0	0	1	0	1	1	0	

Figure 2.15: Covering secret data into text

A colour image is represented by pixel i.e. picture element and is a combination of three colours (red, blue and green). Each colour takes 8 bits which implies 24 bits are required for 1 pixel. The least significant bit (LSB) of each colour is usually 0. For representing a single bit of secret data, it may remain as such to represent 0 or may be inverted to represent 1 (Figure 2.16). This change is not detected by humans through their eyes [6].

100111 <u>0</u>	0101110 <u>1</u>	0001110 <u>1</u>
100101 <u>0</u>	0101011 <u>0</u>	1000101 <u>0</u>
100111 <u>1</u>	1110001 <u>0</u>	0011001 <u>1</u>

Figure 2.16: Color image steganography with hidden data marked in LSB

King Darius of Susa shaved the head of one of his prisoners and wrote secret information on his scalp. Soon, the prisoner’s hair grew back

and he was sent to the king's son in law Aristogoras in Miletus undetected.

Invisible ink was used by Romans for transferring the secret messages. In this process, a natural substance such as fruit juices, milk, lemon etc. was used for secret message writing. Later, heating up the message reveal the secret message.

World War I and II witnessed two new approaches: (1) null ciphers in which 3rd letter from each word is considered to form a hidden message and (2) microdots in which secret pictures were taken and then shrunk to the size of a dots before hiding in cover.

Since 1992 onwards, steganography is being used digitally with the introduction of most popular carriers such as digital images, digital signals, video files and communication protocols.

Several steganographic approaches and tools have been invented till date. A new steganographic technique is devised when an old technique has been understood by attackers. The modification is done based on the old methods and a new better technique is generated for data hiding.

Characteristics of Steganography

In steganography, the method of hiding the data should be such that it maintains the quality of the cover media. Some of the essential characteristics of a data hiding steganography approach are: robustness, capacity, undetectability, invisibility and security.

Robustness is the ability to resist against intentional (rescaling, rotation, resizing, cropping, random chopping, filtering, etc.) and unintentional attacks (lossy compression, digital-to-analog compression, re-quantization, re-sampling, etc).

Capacity refers to the amount of data that can be hidden in the cover file without affecting the quality of the cover.

Undetectability refers to inability of the computer to differentiate between covers and stego-objects by using statistics or computational methods.

Invisibility or Transparency measures how well secret information is hidden in the cover media without affecting the quality (degradation or loss) of the cover.

Security ensures that the information must be reached to intended recipient, not to other unintended attacker.

2.7 SUMMARY

In this unit classical cipher techniques and terminology is provided. Functioning of some of the major classical substitution methods like monoalphabetic substitution cipher, playfair cipher, polyalphabetic cipher is described. The cryptanalysis of these ciphers using letter frequencies and brute force attack is discussed. Another

category of cipher namely, transposition cipher is presented along with product ciphers. Towards end, steganography is elaborated with example.

Terminal Questions

1. *Mention the characteristics of symmetric encryption?*
2. *Differentiate between Cryptography and Staganography.*
3. *Encrypt “The Bell is ringing please Hurry” by using Playfair cipher with key “MEET ME AFTER THE CLASS”.*
4. *Encrypt “Encryption and Decryption requires key and its inverse” by using Playfair cipher with key “CIPHER”.*
5. *List all the possibilities of replacement of alphabet “E” in a Playfair cipher with key “WHERE ARE MY NOTES”.*
6. *Decrypt the following cipher using Playfair cipher with the key “royal new zealand navy”.*

QKWIWOUOONIKGLNWEHHDHZNBMOD

7. *Encrypt the message “Russians are Indian friends” using Hill Cipher with the following key.*

(11 05 07 08
13 01 12 06
04 09 14 17
18 03 19 02)
8. *Enrypt a message “WHEN WILL YOU START READING THESE NOTES” using transposition cipher with the help of key “cipher”.*

References:

1. Paul Van De Zande, “The Day DES Died”, 2001. available at: <https://www.sans.org/reading-room/whitepapers/vpns/day-des-died-722>, accessed on 21.05.2017.
2. Overview of cryptography, PKI Outreach Programme (POP) Nationwide Awareness Programme, Centre for Development of Advanced Computing (C-DAC), Electronics City, Bangalore, 2012.
3. Stallings, William, “Chapter 1: Introduction”, Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
4. Forouzan, Behrouz A., “Chapter 3: Traditional Symmetric-Key Ciphers”, Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
5. CryptTool™ Version 1.4.30, available at- <https://www.cryptool.org/en/download-ct1-en/215>, accessed on 30.11.2016.

6. Forouzan, Behrouz A., "Chapter 1: Introduction", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.

Appendix 2A

Table 2A.1 shows some of the important sets and definitions in integer arithmetic that create a background for modular arithmetic/cryptography and are used in this unit. They are discussed along with the examples later in Block2.

Table 2A.1

Set	Definition	Representation
Set of Integers	Contains all integral numbers from negative infinity to positive infinity	$Z = \{ \dots, -2, -1, 0, 1, 2, \dots \}$
Set of Residues	In modular arithmetic, the set obtained by applying modulo operation is referred as the set of least residues modulo n , or Z_n .	Some Z_n sets $Z_n = \{0, 1, 2, \dots, (n-1)\}$ $Z_6 = \{0, 1, 2, \dots, 5\}$
Additive Inverse	Two numbers a and b are additive inverses of each other if upon addition the remainder is evaluated as zero (0) in modular arithmetic	Additive inverse pairs in Z_{10} : $(0, 0), (1, 9), (2, 8), (3, 7), (4, 6),$ and $(5, 5)$
Multiplicative Inverse Z_n^*	In Z_n , two numbers a and b are the multiplicative inverse of each other if upon multiplication the remainder is evaluated as one (1) in modular arithmetic. In modular arithmetic, an integer may or may not have a multiplicative inverse.	Multiplicative inverses in Z_{10} : $(1, 1), (3, 7)$ and $(9, 9)$ Numbers 0, 2, 4, 5, 6, and 8 do not have a multiplicative inverse $Z_{10}^* = \{1, 3, 7, 9\}$

Z_n is used when additive inverses are required while Z_n^* is used when multiplicative inverses are required.

UNIT- 3

Block Ciphers and DES

Structure

- 3.0 Introduction
- 3.1 Objectives
- 3.2 Principles Related to Block Ciphers
- 3.3 Data Encryption Standard Cipher
- 3.4 DES Strength
- 3.5 DES Cryptanalysis
- 3.6 Principles Related to Design of Block Cipher
- 3.7 Multiple DES Scenarios
- 3.8 Block Cipher Operation Modes
- 3.9 Summary
- 3.10 Terminal Questions

3.0 INTRODUCTION

In the initial topics of this unit, basics pertaining to block and stream ciphers are provided followed by their comparisons. Then, Data Encryption Standard (DES) one of the most widely used symmetric ciphers, is presented. A detailed study of DES provides us an opportunity to learn the principles of symmetric ciphers. In comparison to public key algorithm like RSA, DES structure is much more complex. Hence, DES is described in details in this unit. These details include strengths, cryptanalysis of DES. It is followed by discussion of block cipher design principles and multiple DES. Towards end, different modes of operations for block ciphers are presented.

3.1 OBJECTIVES

Traditional cryptography started with sharing a secret between the two communicating parties. Hence, symmetric cryptography methods were evolved. DES is an important step in symmetric cryptography. Without proper understanding of DES algorithm, the functionality of symmetric algorithms cannot be understood properly. At the end of this unit, you will be able to:

- know about the Feistel cipher and an example of Feistel cipher i.e., DES.
- Understand the definition of P-Box, S-Box, compression box and other internal principles of good symmetric encryption methods.
- Know the concept of round key generation and its use in

various DES phases.

- Gain knowledge about cryptanalysis attacks on DES and block cipher design approaches.
- Apply Multiple DES i.e., using keys and cipher more than once.
- Understand different modes of operations for block ciphers.

3.2 PRINCIPLES RELATED TO BLOCK CIPHERS

In symmetric encryption two kinds of ciphers exist: One stream and other block cipher.

Stream ciphers

A Stream cipher operates on sequence of bits or bytes of plaintext. It generates a keystream independent of plaintext and ciphertext. The keystream usually depends upon chosen key and is evaluated by both sender and receiver synchronously. Hence, the cipher is also termed as synchronous stream cipher. The keystream may even sometimes depend upon data and its encryption (stream cipher is referred to as self-synchronizing in this case). The keystream is combined with plaintext using bitwise exclusive-or operation. Stream cipher encrypts each character in a plaintext using a different key from the key stream. Hence, the transformation of plaintext units will vary, depending on when they are encountered during the encryption process. The most common stream ciphers are Vigenere cipher and one-time pad cipher.

Block Ciphers

A Block cipher encrypts fixed length blocks and returns output having same length as that of input block. Typical length of block considered in cryptography is 64 or 128 bits.

In block cipher, encrypting a plaintext using same key will result in the same ciphertext.

A block cipher having block length as 1 is called *substitution cipher*.

Any block cipher can also be used as a stream cipher by transforming it into a keystream generator using certain modes of operations. In comparison to such transformed ciphers, stream ciphers with a dedicated design are much faster. Stream ciphers can be designed to work much faster than any block cipher whereas block ciphers operate on large blocks of data. Block ciphers are applicable to more range of applications in comparison to stream ciphers e.g. network based symmetric encryption applications make use of block ciphers [1][2].

Differences between block and stream cipher

Block Cipher	Stream Cipher
• All the bits of a block are encrypted at the same time. • Key used for encryption for all blocks remains the same. However, different modes of encryption (like ECB, CBC, CFB, OFB) can be applied to a number of blocks.	• Encryption is done bit by bit or on a small number of bits at a time. • Key used for encryption/decryption varies each time an operation is performed and is discarded after encryption / decryption.

Feistel Cipher:

Basis of the current day symmetric algorithms is Feistel structure. Feistel cipher is a block cipher. It is named after IBM cryptographer Horst Feistel.

It is sometimes also referred as Feistel network. Feistel network follows a definite structure based upon multiple repetitions. Each repetition is referred to as a round.

Feistel cipher is a product cipher and each round in it involves the following operations:

- (i) Shuffling of Bit referred to as permutation boxes (P-boxes).
- (ii) Simple non-linear functions referred to as substitution boxes (S-Boxes)
- (iii) Mixing using XORing.

The result of these operations is “**confusion and diffusion**”.

Diffusion is an essential step in the symmetric key based cryptography and is used mainly to prevent the attacker’s attempt. It refers to the process where a single digit in a plain text is responsible for affecting many digits in the ciphertext. This is required to hide the statistical nature of plaintext i.e., this hides the frequency of occurrence of plaintext characters. Thus it is not possible for the attacker to find out from the changes in the output about whether the change occurred in the input plaintext or the encryption key. Thus, changes in the output do not correlate with changes at a particular place in the plaintext or the encryption key.

By **Confusion** process, the cryptographic algorithm is made non-linear and complex. The deciphering of the plain text is easy in confusion whereas the attacker’s task to finding the cryptographic algorithm and its functionality becomes difficult.

The diffusion effect is caused by Bit shuffling while confusion results from substitution.

Basic operation of a Feistel cipher

There are several rounds and in each round, the plaintext block that is supposed to be encrypted is split into two halves. One half of the block takes a subkey and applies function f . The output produced by f is EXORed with the other half. The two halves are then swapped. Each round follows the same pattern except for the last round where there is no swap.

A nice feature of a Feistel cipher is that encryption and decryption are structurally identical, though the sub-keys used during encryption at each round are taken in reverse order during decryption, that is key schedule is reversed. Therefore, the size of the code or the circuitry required to implement such a cipher is nearly halved. Moreover, function F used while encryption, does not have to be invertible and can be very complex [2].

3.3 DATA ENCRYPTION STANDARD CIPHER

Data Encryption Standard (DES) is published by National Institute of Standards and Technology (NIST). It is a symmetric key based block cipher. It is basically a modification of the IBM project Lucifer and was submitted by IBM against the request for proposal for a national symmetric key cryptosystem in 1973. It was published in 1975 in federal register as a draft of the Federal Information Processing Standard (FIPS). Initially, the draft was opposed and criticized because of the following two reasons:

- 1) The original Lucifer algorithm had 128 bit key length whereas the drafted DES has small key length (Only 56 bits). This reduced key length may result in attacks like brute-force to succeed.
- 2) The internal structure of DES was hidden and was not explained in depth to public. It was thought that suspicious S boxes may contain trapdoor that does not require key for decryption under some condition.

In 1977, DES was finally published as FIPS 46. Later NIST recommended a new variant triple DES as FIPS 46-3 standard.

DES uses 64 bit block size which implies input to DES encryption is 64 bit plaintext while output by DES is 64 bit ciphertext. During decryption 64 bit ciphertext as input is given to DES while 64 bit plaintext is generated as output (**Figure 3.1**).

Encryption process includes two permutations (P –Boxes) and 16 Feistel rounds. One of the P box is termed as initial permutation box while other box is termed as final permutation box. A new round key of 48 bits is generated for each round from 56 bit cipher key. The general structure of DES is shown in **Figure 3.2**. The two P boxes - initial and final boxes permute the data block. These are keyless straight permutations and are inverse of each other. This can be understood from the **Figure 3.3**.

The two permutations are shown in tabular form in **Table 3.1**. In this, the element defines input port number while index defines the output port number. These two permutations are predetermined and bear no

cryptographic significance. They are probably included in DES design due to the fact that they may prevent the software simulation. The initial and final permutations are inverse of each other [3].

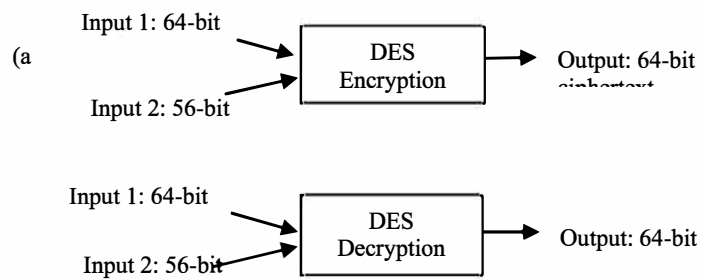


Figure 3.1: (a) DES - encryption and (b) DES – decryption

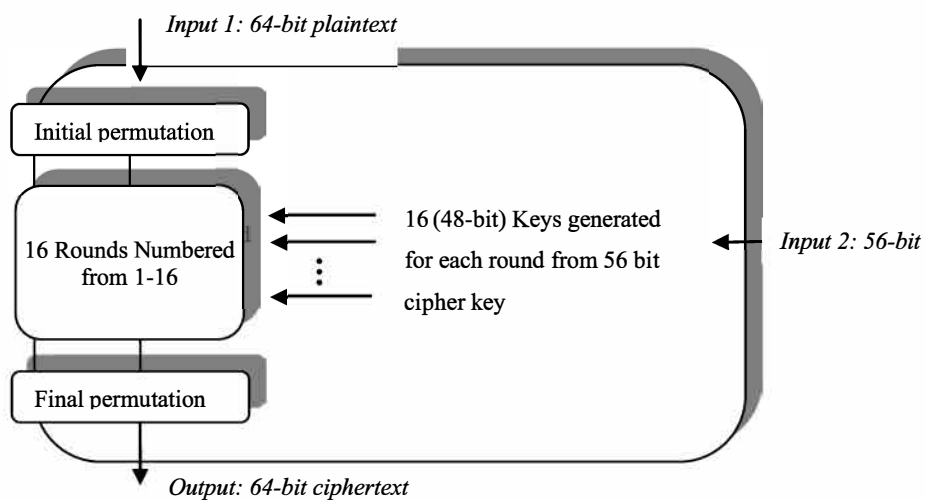


Figure 3.1: (a) DES - encryption and (b) DES - decryption

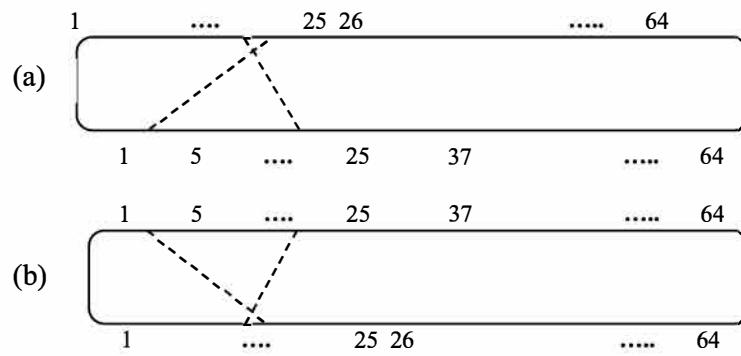


Figure 3.3: DES (a) Initial permutation and (b) Final permutation

Table 3.1: DES – Expansion P-box table for (a) initial and (b) final permutation

58	50	42	34	26	18	0	02	40	08	48	16	56	24	64	32
<i>1</i>	<i>2</i>	<i>3</i>	<i>4</i>	<i>5</i>	<i>6</i>	<i>7</i>	<i>8</i>	<i>1</i>	<i>2</i>	<i>3</i>	<i>4</i>	<i>5</i>	<i>6</i>	<i>7</i>	<i>8</i>
60	52	44	36	28	20	12	04	39	07	47	15	55	23	63	31
<i>9</i>	<i>10</i>	<i>11</i>	<i>12</i>	<i>13</i>	<i>14</i>	<i>15</i>	<i>16</i>	<i>9</i>	<i>10</i>	<i>11</i>	<i>12</i>	<i>13</i>	<i>14</i>	<i>15</i>	<i>16</i>
62	54	46	38	30	22	14	06	38	06	46	14	54	22	62	30
<i>17</i>	<i>18</i>	<i>19</i>	<i>20</i>	<i>21</i>	<i>22</i>	<i>23</i>	<i>24</i>	<i>17</i>	<i>18</i>	<i>19</i>	<i>20</i>	<i>21</i>	<i>22</i>	<i>23</i>	<i>24</i>
64	56	48	40	32	24	16	08	37	05	45	13	53	21	61	29
<i>25</i>	<i>26</i>	<i>27</i>	<i>28</i>	<i>29</i>	<i>30</i>	<i>31</i>	<i>32</i>	<i>25</i>	<i>26</i>	<i>27</i>	<i>28</i>	<i>29</i>	<i>30</i>	<i>31</i>	<i>32</i>
57	49	41	33	25	17	09	01	36	04	44	12	52	20	60	28
<i>33</i>	<i>34</i>	<i>35</i>	<i>36</i>	<i>37</i>	<i>38</i>	<i>39</i>	<i>40</i>	<i>33</i>	<i>34</i>	<i>35</i>	<i>36</i>	<i>37</i>	<i>38</i>	<i>39</i>	<i>40</i>
59	51	43	35	27	19	11	03	35	03	43	11	51	19	59	27
<i>41</i>	<i>42</i>	<i>43</i>	<i>44</i>	<i>45</i>	<i>46</i>	<i>47</i>	<i>48</i>	<i>41</i>	<i>42</i>	<i>43</i>	<i>44</i>	<i>45</i>	<i>46</i>	<i>47</i>	<i>48</i>
61	53	45	37	29	21	13	05	34	02	42	10	50	18	58	26
<i>49</i>	<i>50</i>	<i>51</i>	<i>52</i>	<i>53</i>	<i>54</i>	<i>55</i>	<i>56</i>	<i>49</i>	<i>50</i>	<i>51</i>	<i>52</i>	<i>53</i>	<i>54</i>	<i>55</i>	<i>56</i>
63	55	47	39	31	23	15	07	33	01	41	09	49	17	57	25
<i>57</i>	<i>58</i>	<i>59</i>	<i>60</i>	<i>61</i>	<i>62</i>	<i>63</i>	<i>64</i>	<i>57</i>	<i>58</i>	<i>59</i>	<i>60</i>	<i>61</i>	<i>62</i>	<i>63</i>	<i>64</i>

(a) (b)

NOTE: A cell has two values: First value represents the input port number of permutation box while second value written in small and italic fonts represents index/output port number of permutation box. Ports used in Figure 3.3 are marked in Table 3.1 with color.

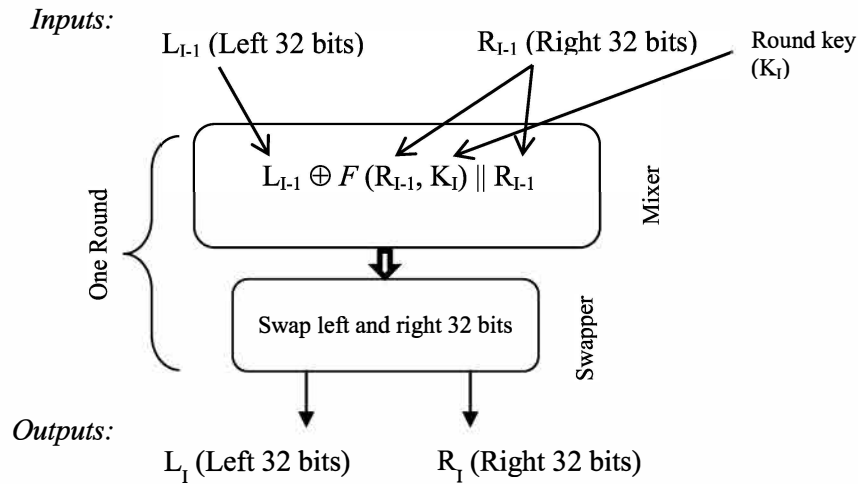


Figure 3.4: DES A single round

1. 32 bit input expanded to 48 bit using *expansion P-Box*.
2. Expanded 48 bits XORed with key (K_I). It is termed as *whitener*.
3. 48 XORed bits are compressed to 32 bits using *array of 8 S-Boxes*.
4. 32 bits are permuted using *Straight P-Box* to give final 32 bit output.

Figure 3.5: DES function ($F(R_{I-1}, K_I)$) steps

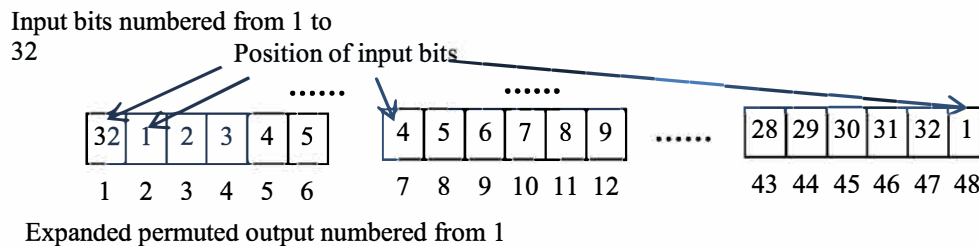


Figure 3.6: DES-48 bit expanded permuted output

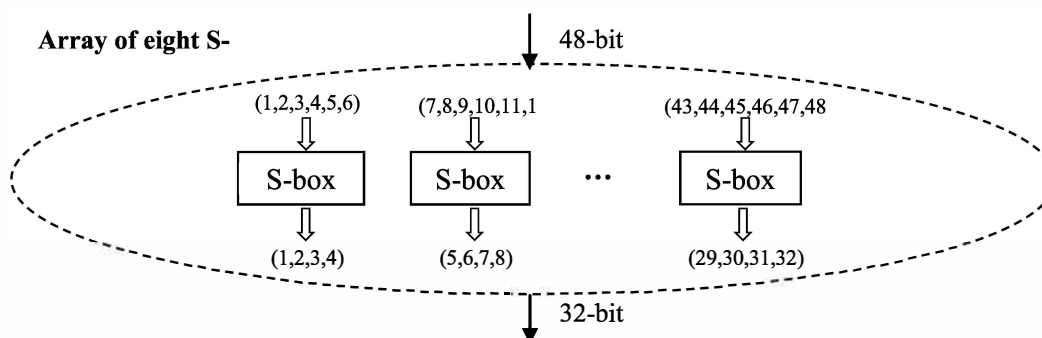
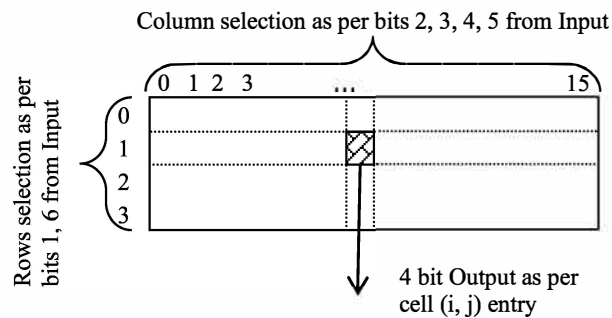


Table 3.: DES – S boxes

Table 3.2: DES – Expansion P-box table

32 1	01 2	02 3	03 4	04 5	05 6
04 7	05 8	06 9	07 10	08 11	09 12
08 13	09 14	10 15	11 16	12 17	13 18
12 19	13 20	14 21	15 22	16 23	17 24
16 25	17 26	18 27	19 28	20 29	21 30
20 31	21 32	22 33	23 34	24 35	25 36
24 37	25 38	26 39	27 40	28 41	29 42
28 43	29 44	31 45	31 46	32 47	01 48

**Figure 3.8: Functioning of S Box in DES**

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

Table 3.3* : DES S Box 1

* Due to space considerations, only one S-Box i.e., S-BoX 1 is shown. The values in the **Table 3.3** given in decimal are to be changed to binary.

DES has 16 rounds and each round is Feistel cipher. A single round (**Figure 3.4**) takes input as L_{I-1} and R_{I-1} and gives output as L_I and R_I for the next round. The first round takes input from the initial P box while the last round gives output for the final P box. Each round has two elements termed as mixer and swapper. Both are invertible. The swapper swaps left and right parts. The mixer has a non-invertible DES function $f(R_{I-1}, K_I)$ that takes

the right part of previous round and round subkey as input. The output of this DES function is XORed with the left side part of the previous round. This output along with the right hand side part of previous round is fed into swapper who swaps these.

The DES function (**Figure 3.5**) has four sections: an expansion P-box, a whitener, S-boxes and a straight P- box. DES function considers 48-bit key for operating on right side part i.e R_{I-1} . The output is of 32-bits.

Expansion P-box expands the right side (R_{I-1}) 32 bits to 48 bits. For expanding, 8 4 bit sections are considered and each is expanded to 6 bits. **Figure 3.6** shows the expansion process. The 4 bits i.e 1,2,3 and 4 are send directly to the output bits i.e 2,3,4 and 5. First and Sixth bits of the output comes from fourth bit of previous section and first bit of next section. This expansion is predetermined and hence shown in expansion P-box **Table 3.2**. In the table the number of output parts is 48 and the range of values is 1-32 which means that some inputs go to more than one output.

After expansion, the 48 bits (of the right side part (R_{I-1})) are XORed with the 48 bits of the round key. The round key is used in this part only in the entire symmetric cipher.

The S-Boxes provide confusion (substitution) in the DES algorithm. 8 S-boxes (**Figure 3.7**) are used in DES, each having 6 bit input and 4 bit output. The 48 bit output from whitener is divided into 8 groups of 6 bits. Each group is then supplied to S-box. The 8 S-boxes together generates 8×4 i.e. 32 bits. Each S-box (**Figure 3.8 and Table 3.3***) has 4 rows and 16 columns. First and Sixth input (2 bits) determines the row in the S-box where as bits from 2 to 5 (4bits) determines the column in the S-box. The value of a cell hence determined in S-box is returned as 4 bit output by S-box.

The final step in DES function is application of a predetermined straight permutation that considers 32 bit input and gives 32 bit output. It is shown in **Table 3.4**

Check your progress 1

- What is full form of DES? What are its main functions of this algorithm?
- What is the first step in DES? What role does substitution plays in DES?
- What is the output if input to the S-Box is 100101 ?
- What is the output if input to the S-Box is 001101 ?

Table 3.4: Straight permutation

16 1	07 2	20 3	21 4	29 5	12 6	28 7	17 8
01 9	15 10	23 11	26 12	05 13	18 14	31 15	10 16
02 17	08 18	24 19	14 20	32 21	27 22	03 23	09 24
19 25	13 26	30 27	06 28	22 29	11 30	04 31	25 32

Table 3.5: DES – compression P-Box (parity drop and permutation table)

57 <i>1</i>	49 <i>2</i>	41 <i>3</i>	33 <i>4</i>	25 <i>5</i>	17 <i>6</i>	09 <i>7</i>	01 <i>8</i>
58 <i>9</i>	50 <i>10</i>	42 <i>11</i>	34 <i>12</i>	26 <i>13</i>	18 <i>14</i>	10 <i>15</i>	02 <i>16</i>
59 <i>17</i>	51 <i>18</i>	43 <i>19</i>	35 <i>20</i>	27 <i>21</i>	19 <i>22</i>	11 <i>23</i>	03 <i>24</i>
60 <i>25</i>	52 <i>26</i>	44 <i>27</i>	36 <i>28</i>	63 <i>29</i>	55 <i>30</i>	47 <i>31</i>	39 <i>32</i>
31 <i>33</i>	23 <i>34</i>	15 <i>35</i>	07 <i>36</i>	62 <i>37</i>	54 <i>38</i>	46 <i>39</i>	38 <i>40</i>
30 <i>41</i>	22 <i>42</i>	14 <i>43</i>	06 <i>44</i>	61 <i>45</i>	53 <i>46</i>	45 <i>47</i>	37 <i>48</i>
29 <i>49</i>	21 <i>50</i>	13 <i>51</i>	05 <i>52</i>	28 <i>53</i>	20 <i>54</i>	12 <i>55</i>	04 <i>56</i>

Encryption/Decryption - Encryption and Decryption both have 16 rounds. There are two approaches for encryption/decryption [3]

- I. Round 16 (Last round) is different from other rounds i.e., it has only mixer and no swapper. The round keys are applied in reverse order during decryption i.e. K_{16} (Sub key for round 16) is applied first and K_0 at last.

In the encryption side L_0 (or R_0) is encrypted as L_{16} (or R_{16}) whereas in the decryption side L_{16} (or R_{16}) is decrypted as L_0 (or R_0). In this approach (Shown in **Figure 3.9**), the rounds are not aligned rather the mixer and swapper are aligned.

- II. An extra swapper is added after complete 16 rounds (of mixer and swappers). This addition nullifies the effect of last swapper.

Key Generation –

DES cipher key – DES key generation involves pre-process step where the initial 64 bit key is reduced to 56 by removing extra parity bits. This is referred as parity bit drop. The last bit from each octet is removed i.e. bits 8, 16, 32, 64 are removed. The parity drop table is shown as **Table 3.5**. The remaining 56 bits are used to

generate sixteen 48 bit round keys. The entire key generation process is shown in **Figure 3.10** [3].

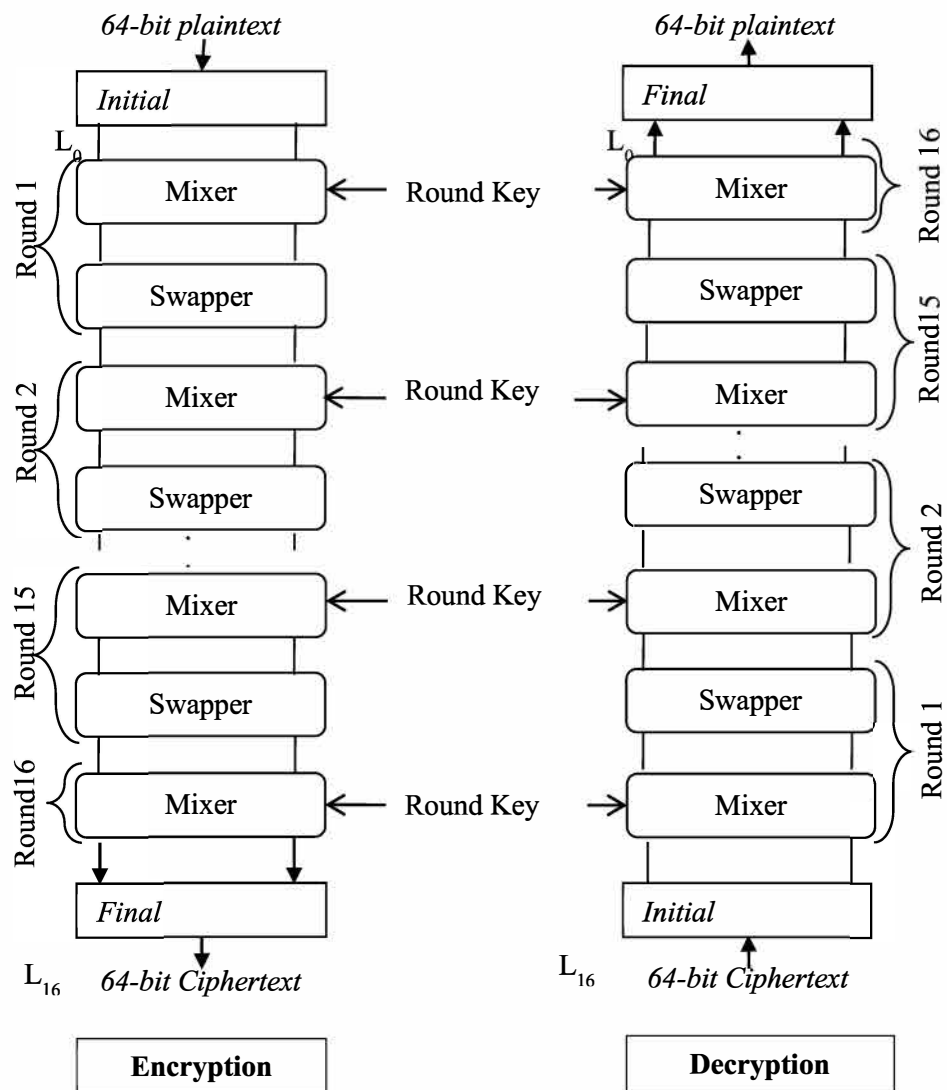
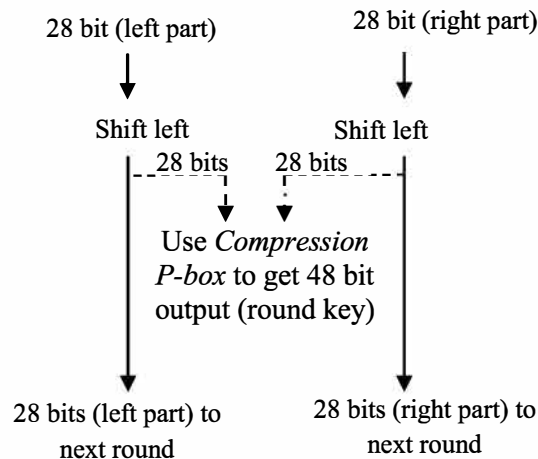


Figure 3.9: DES – cipher and reverse cipher approach

Table 3.6: Key compression (P-Box) table

14 1	17 2	11 3	24 4	01 5	05 6	03 7	28 8
15 9	06 10	21 11	10 12	23 13	19 14	12 15	04 16
26 17	08 18	16 19	07 20	27 21	20 22	13 23	02 24
41 25	52 26	31 27	37 28	47 29	55 30	30 31	40 32
51 33	45 34	33 35	48 36	44 37	49 38	39 39	56 40
34 41	53 42	46 43	42 44	50 45	36 46	29 47	32 48

* Drop parity bits from 64 bits key to get 56 bits key. Divide it into 2 parts, each of 28 bits.



Rule for shifting in a round:

For round number 1, 2, 9, 16 shift by one bit

For other rounds shift by two bits

Figure 3.10: Generating keys for individual rounds in DES

Left Shift - The 56 bit key is divided into 2 parts, each of 28 bits. Each part is shifted left by one bit (or two bits) in a circular fashion. Shifting of 1 bit is done in round number 1, 2, 9 and 16 while in other rounds 2 bit shifting is done. After shifting the two parts are combined.

Compression permutation - The compression permutations shown in **Table 3.6** are applied for changing 56 bits to 48 bits for use in a round [3].

Example DES

Key: 1111010101111111

Plaintext: Encrypting using DES

Ciphertext (Using Crypt Tool– Cipher Block Chaining (CBC mode)

(in Hex): 6B 35 0F 7C E9 35 0B E7 3A BE 1D E2 86 55 E2 DB 5B C1 18 67 A0 93 A4 DA

NOTE: Various cipher modes of operations like CBC are discussed later in section 3.8.

DES analysis

Properties - Below the two important block cipher properties are discussed.

- 1) **Avalanche effect** - This effect implies that upon change in plaintext or key by small amount should reflect tremendous changes in the cipher text. The results in the ex. show that half of ciphertext bits change upon changing a single bit in either plaintext or keys.

- This property has been incorporated in DES cipher.
 - The avalanche effect in DES is shown below by changing (a) plaintext (b) key.
- (a) Changing plaintext by 1 bit results in change in 45 percent (29 bits) in ciphertext.

Key: 1111010101111111 (Using Crypt Tool – CBC mode)

Plaintext: 0000000000000001

98 58 E1 77 C8 71 B3 28 CE 19 94 D9 F4 78 C1 7F 5F BE 06 85
D9 E6 D9 BA

Changed Plaintext: 0000000000000011

98 58 E1 77 C8 71 B3 28 **41 5B 95 3A EE 14 A4 32 60 0E 01 C0**
55 07 74 6F

- (b) Changing key by 1 bit results in change in 91.7 percent (44 bits) in ciphertext.

Plaintext: 0000000000000001 (Crypt Tool mode – CBC mode)

Key1: 1111010101111111

98 58 E1 77 C8 71 B3 28 CE 19 94 D9 F4 78 C1 7F 5F BE 06 85
D9 E6 D9 BA

Key 2: 0111010101111111

D4 **56** 19 4E DA 3D 1B B3 79 DD **54 A9** 0B 0E A3 **7A** 00 45 97
DF A3 FF ED 52

- 2) Completeness effect - This effect implies the dependency of ciphertext bit on many plaintext bits. Again, DES strongly implements this effect through its P-boxes and S-boxes.

3.4 DES STRENGTH

Use of 56 bit keys

- 2^{56} keys are possible, a brute-force attack on such large number of keys is impractical.
- For a machine executing one DES encryption per microsecond, it will take approximately 1000 years for breaking the key.
- For a machine that executes in parallel with approximately 1 million encryption devices each performing one encryption per microsecond it will take 10 hours to break the key.
- In 1998, the electronic frontier foundation built DES cracker machine that took less than 3 days to crack the DES cipher.
- The attack was based upon key search rather than checking all possible keys.

- In such attacks knowing the language of plain text, whether the text message was compressed before encryption and the type of data matters i.e. it is easy to apply the brute force attack if some information about expected plain text is available before applying decryptions.

Nature of DES algorithm

- Eight S-boxes were the issue of concern because their functionality was not known in public and it was suspected that cryptanalysis was possible by the person who knows the S-boxes weaknesses
- Till date no vulnerability or weakness has been discovered in S-boxes.

Timing attacks - In timing attacks the time taken by the cipher to perform decryptions on various cipher text is recorded and utilized for gaining information about the plain text or key.

- This mechanism does not help much in knowing the actual key but may prove to be first step towards doing so.
- DES is resistant to timing attacks.

3.4 DES CRYPTANALYSIS

Differential and Linear Cryptanalysis - The brute force attack is impractical on DES due to longer key length and use of 3 keys in triple DES. Hence, cryptanalysis is used to conduct attacks on DES. Two methods are discussed below that are used for cryptanalysis on DES.

(i) Differential Cryptanalysis - Differential cryptanalysis efforts on DES are summarized in [2]. The results suggest that using differential cryptanalysis DES can be cracked in less than 2^{47} encryptions. But this requires 2^{47} chosen plaintext by the attacker.

- Differential cryptanalysis does not work well against DES because the team that implemented DES was aware of the differential cryptanalysis at the design time and hence have taken measures against this attack.
- Differential cryptanalysis on eight round DES requires 2^{14} chosen plaintexts.
- In differential cryptanalysis the following procedure is followed:
 - Two plaintext messages m and m' are considered with a given difference for a single round.
 - After each round, the probable difference for ciphertext is traced.

- As DES cipher is divided into two parts (m_0, m_1), two probable differences one for each part is traced i.e, the difference after each round is equal to $\Delta m_{17} || \Delta m_{16}$.
- m and m' are submitted for actual encryption under unknown key and the actual different is recorded.
- The actual difference is compared with the probable difference as:

$$E_K(m) \oplus E_K(m') = (\Delta m_{17} || \Delta m_{16})$$
- For the difference that matches as per the above equation, it is assumed that all the intermediate probable patterns are correct.
- The above process is repeated several times for determining all the bits of key.

(ii) Linear Cryptanalysis:-

- This method finds the DES key in 2^{43} known plaintexts which is better in comparison to differential cryptanalysis. Also, it is easier to collect known plaintext than chosen plaintext.

- Linear cryptanalysis is feasible on DES.

- Principle of linear cryptanalysis:

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

holds with probability $p \neq 0$.

where, for message space: $1 \leq a, b \leq n$ and for key space: $1 \leq c \leq m$

- If proposed relation is established, left hand side of above equation is solved for large number of plaintext-ciphertext pairs.
- If result is zero (0) for more than 50%, K is assumed as

$$K[\gamma_1, \gamma_2, \dots, \gamma_c] = 0$$
- If result is one (1) for more than 50%, K is assumed as

$$K[\gamma_1, \gamma_2, \dots, \gamma_c] = 1$$

More such linear equations are obtained and then they are solved to obtain the key bits.

3.6 PRINCIPLES RELATED TO DESIGN OF BLOCK CIPHER

The three major aspects of block cipher design are: the number of rounds, function F design and key scheduling.

Number of rounds

MCS-119/61

- Greater numbers of rounds are chosen so that more cryptanalysis efforts are required irrespective of key strength.
- These efforts must be other than brute-force attack.
- As per Schneier's observations, in 16 round DES, differential cryptanalysis attack require more effort ($2^{55.1}$ operations) as compared to brute-force attack (2^{55} operations). In case DES has less than 15 rounds, differential cryptanalysis is more effective as compared to brute-force attack.

Design of cipher function F

- The substitution provided by F should be difficult to identify.
- F should be made more non-linear for making cryptanalysis difficult.
- More non-linear implies, it is difficult to approximate F by set of linear equations.
- F should follow strict avalanche criterion (SAC), i.e., upon inverting an input bit, the output bit should change with probability $\frac{1}{2}$.
- F should also follow bit independence criterion (BIC), which says that two output bits should change independently upon change in single bit
- Both SAC & BIC strengthen the confusion of DES cipher.

S-Box Design:-

- Larger S-Boxes are more resistant to differential and linear cryptanalysis.
- But on increasing the dimension the look up table also becomes larger.
- Hence, practically the input dimension of S-Box is considered as 8-10.

The following are some of the approaches suggested by Nyberg regarding S-box design [5].

- The entries of S-box should be generated using pseudo random generator.
- Selecting randomly the entries of S-box, testing them against some criteria and rejecting the one which do not pass the test.
- Manual approach using simple mathematics (used by DES design). Large S-boxes are difficult to design using this approach.

- Designing S-boxes using mathematical principle such that they are resistant against linear & differential cryptanalysis.
- S-boxes that use pseudo random number generated digits and then following alteration using keys are also effective (example - Blowfish S-boxes). In such cases the analysis of S-boxes is not possible as the value change depends upon the current key.

Key Scheduling Algorithm

Use of several subkeys in Feistel block cipher ensures that the main key remains safe.

Weakness in DES cipher key is Key Size.

Weak keys: Out of 2^{56} DES keys, 4 keys are termed as weak keys. A weak key consists of either all 0's, all 1's, or half 0s and half 1s.

For example 0101 0101 0101 0101 after dropping parity becomes 0000000 0000000 (56 bits). The problem with a weak key is that upon encrypting a block with weak key and then encrypting the result with same weak key derives the plain text. An adversary can intercept the cipher text and try the weak keys. If after decrypting the cipher text twice with the weak key results in the same ciphertext then the security key is found. This process is shown in **Figure 3.11 (a)**.

Semi weak keys: These occur in pairs. In all six pairs (first - second) are there (**Table 3.7**).

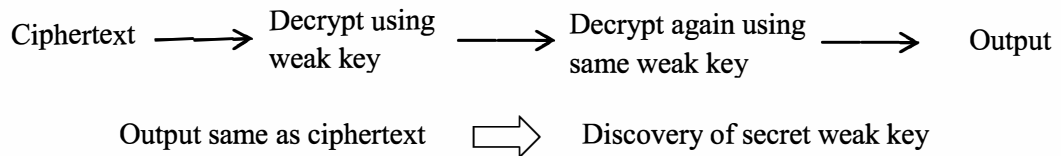
If these semi weak keys are used for generating round keys then only two different round keys are generated from them instead of 16 round keys. Each of the two generated round keys is repeated 8 times. The use of semi-weak keys is illustrated in **Figure 3.11 (b)**.

Possible weak keys: Only 4 distinct round keys are generated by a possible weak key. There are 48 such keys possible in DES.

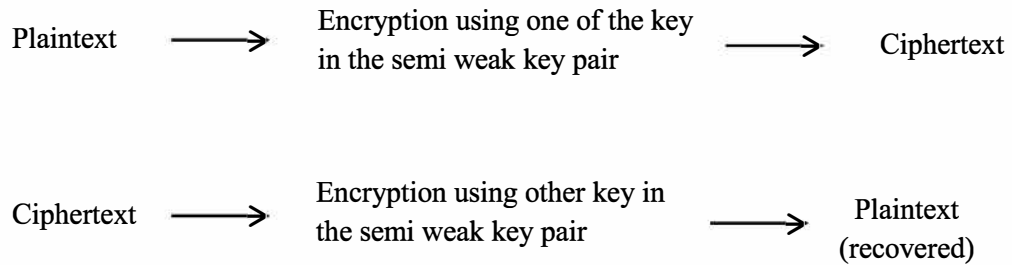
Key complement: Out of 2^{56} keys, half of the keys are complement as these keys can be obtained by inverting bits of the other half. Therefore, for performing brute force attack only half of 2^{56} keys are required to be tested.

Table 3.7: Semi-weak keys (in pairs)

First key	Second key
01FE 01FE 01FE 01FE	FE01 FE01 FE01 FE01
1FE0 1FE0 0EF1 0EF1	E01F E01F F10E F10E
01E0 01E1 01F1 01F1	E001 E001 F101 F101
1FFE 1FFE 0EFE 0EFE	FE1F FE1F FE0E FE0E
011F 011F 010E 010E	1F01 1F01 0E01 0E01
E0FE E0FE F1FE F1FE	FEE0 FEE0 FEF1 FEF1



(a)



(b)

Figure 3.11 (a): Drawback of using (a) a weak key (b) semi weak key pair

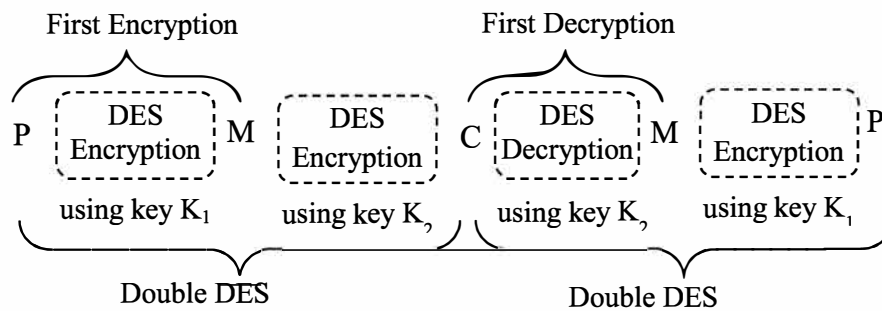
3.7 MULTIPLE DES SCENARIOS

The major concern in DES is the key length. As shown in the cryptanalysis section that DES can be broken by using high capacity machines. Thus, a new thought was given for utilizing the same encryption/decryption algorithm more than once so as to increase the key space. Initially, double DES (2 DES) was considered as shown in **Figure 3.12**. In this, 2 different keys (K_1 & K_2) were considered. It seemed that the key space was doubled (2^{112}) but due to meet-in-the-middle attack, it was proved that the key space was improved only slightly

as shown in Figure 3.12 The meet-in-the-middle attack is based upon principle that the middle text (M) should be same after first encryption and that after first decryption i.e.,

$$M = E_{K_1}(P) \text{ and } M = D_{K_2}(C)$$

An attack may intercept P and C (known – plaintext attack) and then record values in two tables. In one table results obtained after encrypting P by K_1 are kept while in other table results obtained after decrypting C by K_2 are kept. A search is then made in two tables for equal M. If such M is obtained then corresponding keys are also obtained. Hence, for improving security triple DES was proposed [3].



Double DES process: - Result of first encryption is same as first

Meet in Middle Attack

- Under known-plaintext attack, attacker manages (P and C) pair.
- He encrypts P using all possible key (K_1) combinations (2^6) & store result in table as M values.
- He also decrypts C using all possible key (K_2) combinations (2^6) & store result in table as M' values.
- Upon a match between M & M' values, the two keys K_1 & K_2 used for double DES are identified.

Figure 3.12: Double DES & a possible attack on it

Triple DES

In triple DES, the DES cipher is applied thrice. It has two variants. First, triple DES with 2 keys (K_1 and K_2) as shown in **Figure 3.13**. Second, Triple DES with 3 keys (K_1 , K_2 and K_3). Former is used widely by the banking industry while latter is used by applications like PGP. In triple DES, E-D-E combination is used at the encryption site while D-E-D combination is used at the decryption site. Here, E stands for encryption and D stands for decryption [3].

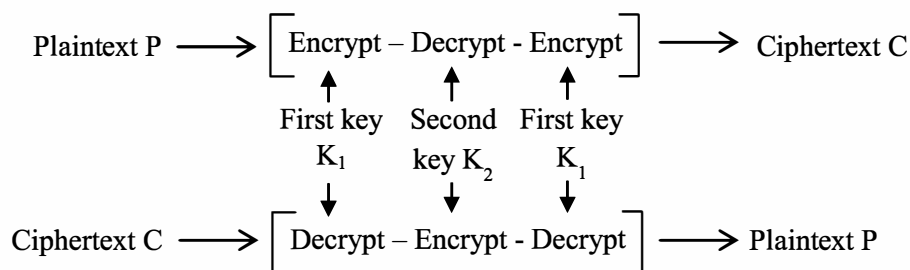
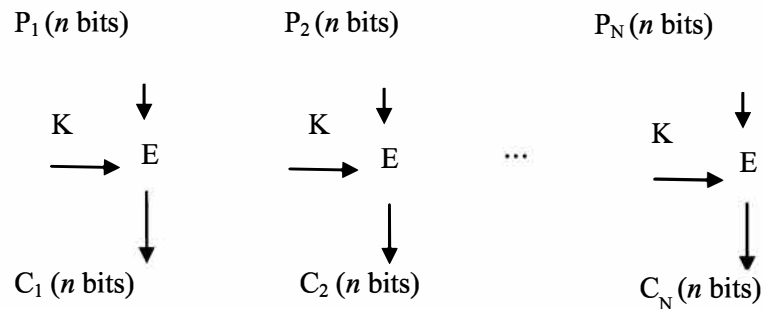


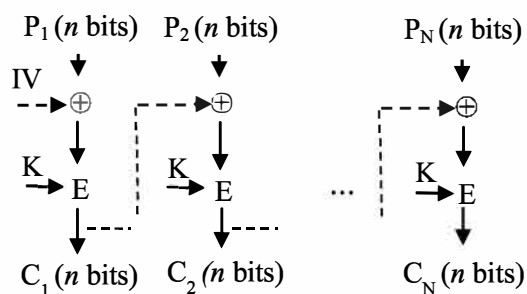
Figure 3.13: Triple DES using two keys (K_1 and K_2)

ECB:

- (1) If plaintext is not a multiple of block size n , it is padded to make it a multiple of n .
- (2) Same key is used to encrypt each block
- (3) Ciphertext and plaintext can be kept in a two column table (Codebook)

**(a)****CBC:**

- (1) IV (initial vector) is predetermined shared value between the sender and receiver and is used to provide the initial value to XOR function. It should be kept secure from changes.
- (2) For encrypting blocks other than first block, the ciphertext of previous block (C_{i-1}) is XORed with plaintext block (P_i) and then encrypted to produce the new ciphertext (C_i).

**(b)****Figure 3.14 (a, b): Modes of operations of block ciphers (ECB & CBC)**

3.8 BLOCK CIPHER OPERATION MODES

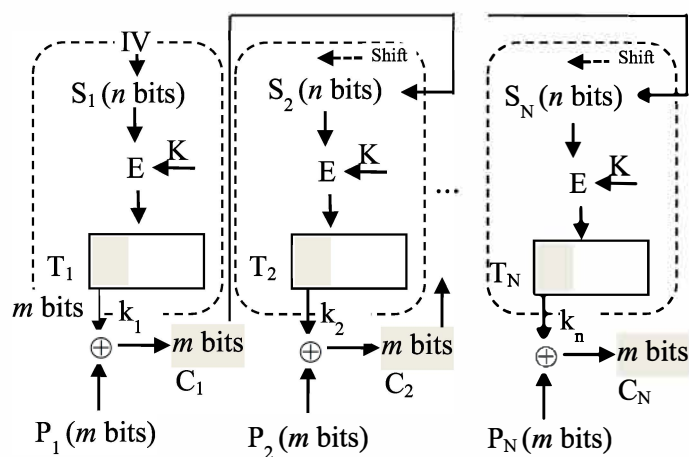
Modern ciphers usually consider block of fixed size (64 bits or 128 bits) as input. In case the length of plaintext to be encrypted is more than the block size i.e., plaintext is of variable size then five (05) modes of operations have been devised. These are:

- (1) Electronics codebook (ECB)
- (2) Cipher block chaining (CBC)
- (3) Cipher feedback (CFB)
- (4) Output feedback (OFB)
- (5) Counter (CTR)

In all modes, the reverse process is applied while decrypting. Due to space considerations, only the encryption in each of these is presented (**Figure 3.14**). Out of these five modes, (1), (2) and (5) are applicable to N blocks each of b bits whereas (3) and (4) consider small size ($m < n$) units of plaintext e.g., ASCII characters sequences. Cipher E may be any modern block cipher, K is the symmetric key, P_i is the plaintext unit and C_i is the ciphertext [4].

CFB:

- (1) It has feedback, m bits from cipher stream are taken for left shifting register S_i .
- (2) Contents of shift register (S_i) are encrypted not the plaintext (P_i)
- (3) m bits ($m < n$) of plaintext (P_i) are XORed with m bits of temp register to produce output (C_i).

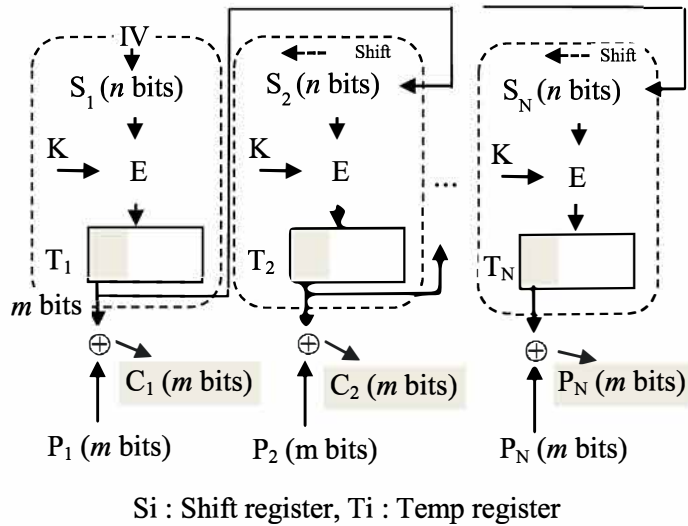


S_i : Shift register, T_i : Temp register, k_i : key stream

OFB:

- (1) The only difference between CFB and OFB mode is that m bit output from temp register is taken for left shifting register S_i (as feedback) instead from C_i .

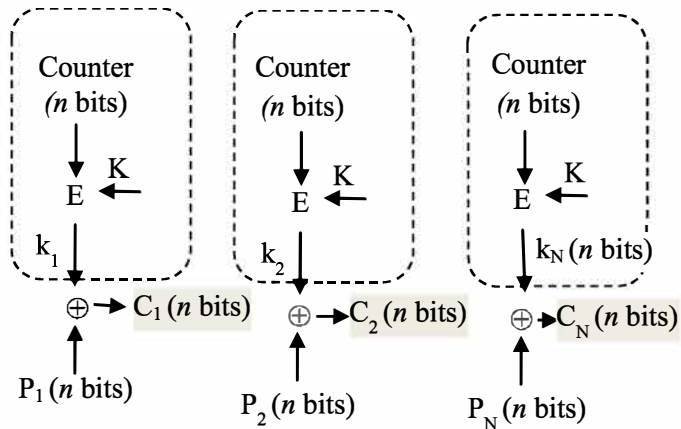
Advantage is that error propagation is avoided as contents of temp register are unaffected by transmission errors and can be calculated independent of ciphertext i.e., before ciphertext arrival at the receiver.



(d)

CTR:

- (1) No Feedback
- (2) A counter variable is initialized with pre-determined value (IV) and is incremented by one each time it is encrypted by E using key K.
- (3) n bit encrypted output is XORed with plaintext block (P_i) to give final output block (C_i).



Si : Shift register, k_i : key stream

(e)

Figure 3.14 (c-e): Modes of operations of block ciphers

3.9 SUMMARY

In this unit, you have learnt about one of the most commonly used symmetric cipher i.e., DES. DES structure is based upon Feistel structure and as such appears as complex (has several rounds involving shifting, XORing, permuting etc.) but the efficiency of DES over public key cipher makes it useful. The major concern in DES is the key length. As shown in the cryptanalysis section that DES can be broken by using high capacity computer machines. Thus, a new thought was given for utilizing the same encryption/decryption algorithm more than once so as to increase the key space. One such popular mechanism that we have followed in this unit is triple DES (3DES) that is stronger as compared with single DES. You are also able to learn the block design principles and functionalities of various block cipher modes of operations.

Terminal Questions

1. *Show the steps involved in DES round key generation for key: 01FE 01FE 01FE 01FE.*
2. *Show the output of DES for plaintext having all 1's (64 bits) and round key as all alternate 0's and 1's (64 bits).*
3. *Discuss and disadvantages and advantages of each mode of cipher block operations.*
4. *What does Triple DES use and what does triple DES guarantee?*
5. *Show the diagram for encryption in CFB and OFB mode for $r = n$.*
6. *Show the functioning of DES cipher in both ECB and CBC modes.*

References:

1. Forouzan, Behrouz A., "Chapter 3: Traditional Symmetric-Key Ciphers", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
2. Stallings, William, "Chapter 3: Block Ciphers and the Data Encryption Standard", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
3. Forouzan, Behrouz A., "Chapter 6: Data Encryption Standard (DES)", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
4. Forouzan, Behrouz A., "Chapter 8: Encipherment using Modern Symmetric-Key Ciphers", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
5. Robshaw, M., Block Ciphers, RSA Laboratories Technical Report TR-601, Aug 1995.
<http://www.rsasecurities.com/rsalabs>.

UNIT-4

Confidentiality Using Symmetric Ciphers

Structure

- 4.0 Introduction
- 4.1 Objectives
- 4.2 Understanding the Encryption Function Placement Options
- 4.3 Providing Confidentiality to Ongoing Traffic
- 4.4 Key Distribution Process
- 4.5 Methods for Generating Random Numbers
- 4.6 Summary
- 4.7 Terminal Questions

4.0 INTRODUCTION

In this unit, primarily the use of symmetric encryption function to provide confidentiality is discussed. An important issue regarding choice of placement of symmetric encryption function either at the link layer or over end to end communication is covered. Use of symmetric encryption function for countering the traffic analysis attacks and for solving key distribution is also illustrated. Towards end, functionality of random number generators along with examples is shown.

4.1 OBJECTIVES

This unit deals with the use of symmetric encryption functionality. At the end of this unit, you will be able to:

- learn the requirements and issues involved in placement of symmetric encryption ciphers.
- use of symmetric encryption for ensuring traffic confidentiality.
- describe the scenarios and methods for symmetric key distribution.
- understand various kinds of random number generators like Linear Congruential Generator (LCG), Pseudo Random Number Generator (PRNG), Blum Blum Shub Generator etc.

4.2 UNDERSTANDING THE ENCRYPTION FUNCTION PLACEMENT OPTION

Encryption is used against security attacks to ensure confidentiality. Encryption function may be located at different places in a communication system. Two major places where encryption may

be provided are (1) communication links (2) end to end communication. Communication entities are usually attached to Local Area Networks (LANs) through which they reach to other network or to other communicating nodes. LANs are broadcast networks and hence, transmission by a node on LAN is visible to all other nodes. An eavesdropper on LAN can monitor, capture and analyze the frames based upon addresses contained within the frame. The situation gets worsened in case the medium is wireless.

The outside world is reached via modem or router. The wire lines on the way to the other end are vulnerable to an intruder who taps into the wire. The attacker may also pick the radio (wireless) signals from nearby locations. Both active and passive attacks are possible in such environment. An attack or vulnerability may take place in hardware or software wherein memory may also be utilized. Thus, attacks can occur at several locations which are not under the control of end user (**Figure 4.1**).

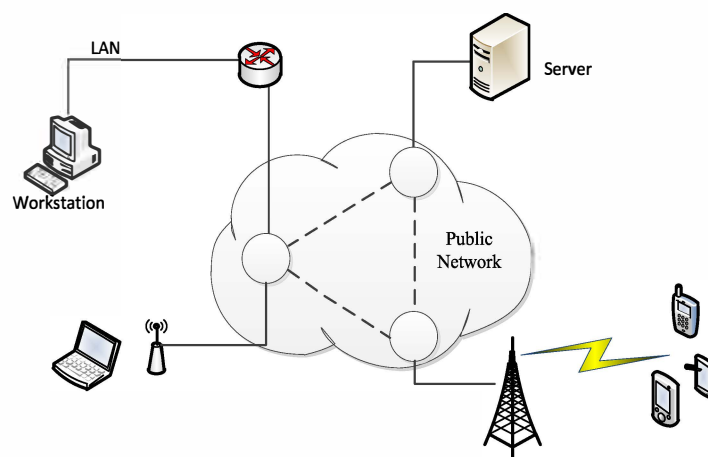


Figure 4.1: Vulnerability points in a network

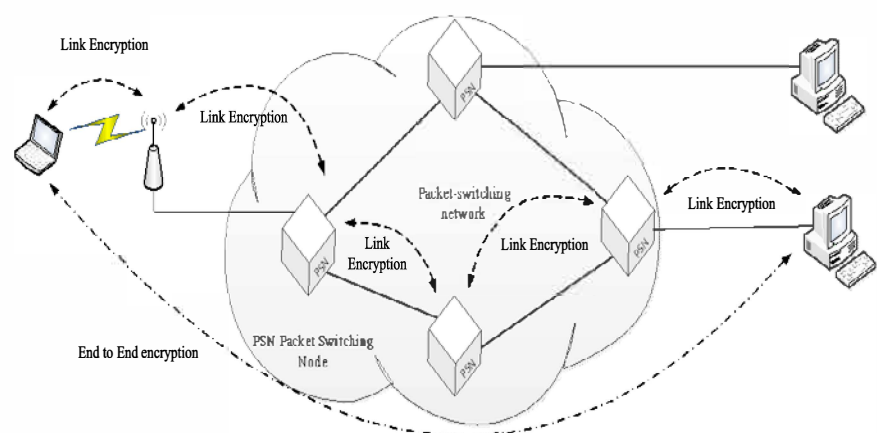


Figure 4.2: Encryption among nodes in a packet switched network

Comparative study between link and end to end encryption

Encryption tries to secure the points of vulnerability. Under link encryption, the link is protected by encryption devices at both the communication ends securing all the traffic on the link. If links are connected to switch then message needs to be decrypted upon its entry into the switch as address from the packet is required. This leaves the message vulnerable at switch. All the links from source to destination must use link encryption which means different keys per link are required whereas under end to end encryption, the data once encrypted at one end is transmitted as such till it reaches the other end where it is decrypted. End to end communication is also associated with weakness. The end host encrypts entire packet including header which means that the in-between switch will not be able to route the packet. Hence, only the user data in a packet must be encrypted leaving the header in clear i.e., unencrypted. Though the user data is secure still a lot may be learnt (by the eavesdropper) from the unencrypted headers. In this method authentication is inherently provided as the decryption at the receiver confirms that this was indeed sent by the user sharing encryption key. Link encryption method does not provide such authentication. Both, link and end to end encryption may be used together when high security is required. Under such circumstances, the sender encrypts the user data via end to end encryption while the packet is encrypted using the link encryption method. This encrypted packet undergoes encryption – decryption several times on switches before reaching destination. During this time i.e., when the packet lies in the memory of switch, the packet is insecure. **Table 4.1** provides a point wise comparison among link and end-to-end encryption [1].

Table 4.1. Comparison among Link and End-to-End Encryption [1][2]

S. No.	Link Encryption	End-to-End Encryption
1	Message gets exposed in sending hosts and in intermediate nodes	Message remains encrypted in sending hosts and in intermediate nodes
2	Link encryption is transparent to user and is applied by sending host who maintains encryption facility and this encryption facility is used by all users on a particular host	End-to-End encryption is applied by the sending process/user. The user selects encryption scheme and determines the algorithm
3	Either all the link layer messages are encrypted or none. (The encryption can be done in hardware)	User may choose to encrypt any message or no message. (The encryption is done in Software)
4	Involves one key per pair. This pair can be either: host-intermediate node pair or intermediate node-intermediate node pair	Involves one key per user pair (sender – receiver)
5	Considers host authentication	Considers user authentication

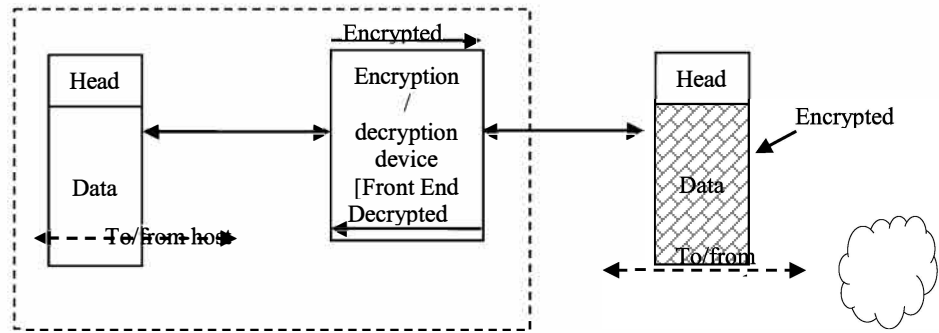
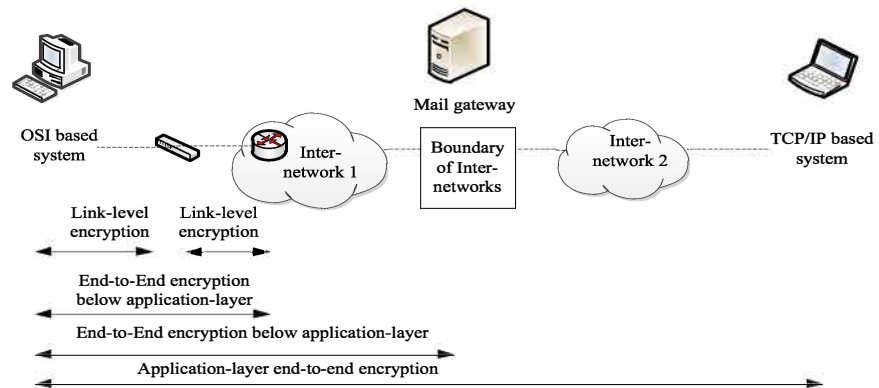


Figure 4.3: Front-End Processor (FED) Function



Application of Encryption	Unencrypted packet contents	Encrypted packet contents
(1) Application-level encryption (on links and at routers and gateway)	Link-H, Net-H, IP-H, TCP-H, Link-T	Data
(2) TCP-level encryption (a) On links and at routers (b) In gateways	Link-H, Net-H, IP-H, Link-T	TCP-H, Data
	Link-H, Net-H, IP-H, TCP-H, Link-T	Data
(3) Link-level encryption (a) On links (b) In routers and gateways	Link-H, Link-T	Net-H, IP-H, TCP-H, Data
	Link-H, Net-H, IP-H, TCP-H, Link-T	Data

Link-H	Net-H	IP-H	TCP-H	Data	Link-T
--------	-------	------	-------	------	--------

A general packet format

Here,

TCP-H = TCP header

IP-H = IP header

header)

Net-H = Network-level header (e.g., X.25 packet header, LLC

Link-H = Data link control protocol header

Link = Data link control protocol trailer

Figure 4.5: Application of encryption to different protocol units/packets

Placing the encryption function

Link encryption is done at lower layers (either physical or link layer) whereas end to end encryption is done at higher layers (network or transport or application layer). There are two possibilities for placement of end to end encryption: (1) Network layer/Transport layer and, (2) Application layer end to end protection. In first possibility of end to end encryption, each end host is in possession of shared encryption key which is used by all the users and applications on that system. On such system, the task of protection/encryption may be assigned to a front end processor (FEP) (**Figure 4.3**). FEP may lie in between host and network. On the host side, FEP receives the user packets (IP packets), encrypts them leaving the header part unencrypted. On the Network side, FEP decrypts the packets and delivers them to host. TCP can also be implemented in FEP but in such cases end to end protection is done only within an inter-network not across the boundary of an inter-network. **Figure 4.4** clears this concept. In this figure, an electronic mail gateway (Store and forward device) is used that inter connects two different inter networks i.e. on one side OSI system is followed while on the other side TCP/IP system is followed. Here, the packets from one host of an internetwork are first terminated at the gateway that sets up new connection to other host in different inter-network. During this period the user packets remain unencrypted (unsafe). Thus, for applications running at application layer of such systems, the only way to achieve protection is to apply end to end encryption at application layer. Usually, a network may support hundreds of hosts and thousands of processes/ users which imply that large number of secret keys are required. The packet structure and data encrypted in all the three schemes i.e. application level encryption, TCP (Network) level encryption and link level encryption are shown in **Figure 4.5**. The following points are observed:

1. In application level encryption only the user data is encrypted whereas TCP, IP and link layer headers are left unencrypted.
2. Under TCP (network) level encryption both the TCP data and headers are encrypted on links at routers whereas on gateways only the TCP data is encrypted. On gateways the packets are decrypted first and then again encrypted for further transmission to other network using a new connection. For this period on the gateways, the data remains unencrypted and unsafe.
3. Considering link level encryption, all units of a packet except link layer header and trailer are encrypted and protected on links whereas it is not so on routers and gateways. Decryption followed by encryption takes place every time a router or gateway is encountered and hence the data remains unsafe during this period on these devices.
4. Upon moving up in the protocol hierarchy (from link to application), less information is encrypted (as headers are not encrypted) but the security gets enhanced.

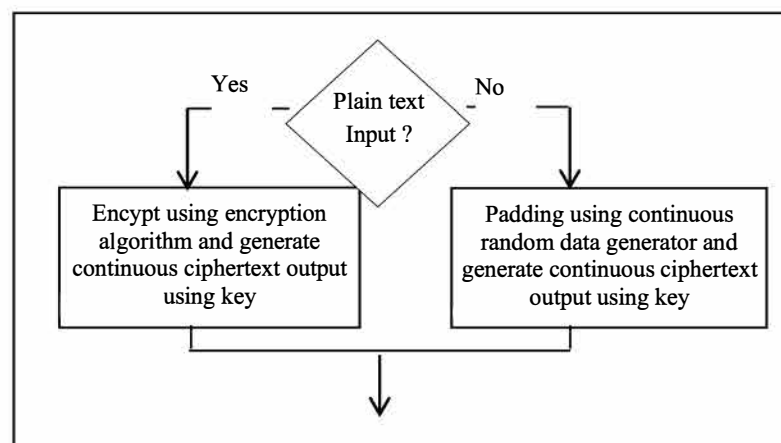
4.3 PROVIDING CONFIDENTIALITY TO ONGOING TRAFFIC

Attackers are usually interested in extracting information by analyzing the ongoing traffic. Such analysis often causes problems especially in military or commercial domains. The information that may be identified from traffic analysis is:

1. Identity of a communicating node
2. Frequency of communication
3. Importance of information may be deduced (by identifying specific patterns or length/quantity of messages).
4. Events may be identified (by correlation analysis of the communicated data).

A mal intended user of an organization may create a covert (stealthy) channel for communicating data to the outside world. This transfer occurs in such a way that the security possibly is violated in the organization and administrators are not able to detect it. For example, two users participating / communicating via covert channel may agree upon that legitimate message of certain length represents binary 0 whereas the messages greater than certain length represents binary one.

In link encryption approach, though the entire header is encrypted, still an eavesdropper may learn the amount of traffic moving into and out of system. This can be stopped by traffic padding (**Figure 4.6**) which restricts such learning by eaves dropper. In traffic padding, additional data from continuous random stream is added such that the output cipher text is produced continuously despite of absence of plain text data. The eavesdropper is unable to differentiate between actual and random data just by seeing the amount of traffic and hence it becomes impossible for him to find out the exact amount of traffic. Similar padding mechanism is also useful in end to end encryption where data parts are padded to uniform length at transport or application level.



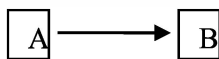
4.4 KEY DISTRIBUTION PROCESS

In symmetric key based encryption either link or at end to end protection, a key is required to be shared among users /processors. The following are the characteristics of keys.

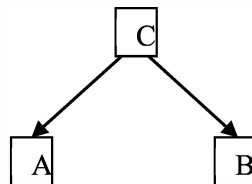
1. Keys must be kept safe from others.
2. Keys must be refreshed or changed at regular intervals so as to limit the amount of data that may be compromised if key is somehow known to the attacker.

A secure key distribution technique is required for exchanging keys between communicating parties (A and B). Following are the ways/cases for key distribution (**Figure 4.7**):

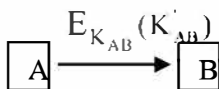
1. Physical (Manual) delivery of key by A to B.
2. Involving third party who selects the key and hand it physically (manually) to both A and B.
3. Utilization of old or existing key for new key distribution by encrypting new key with old key by any one communicating party.
4. Utilization of encryption between A & C and B & C for exchanging corresponding keys.



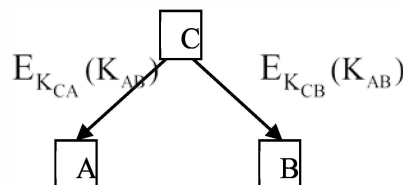
Case 1: Physical Transfer of key from A



Case 2: Third Party Physically



Case 3: A encrypts using old key and then transfers new



Case 4: Third Party encrypts using old key and then transfers new key

$E_K(m)$: Encryption of m using key k

K_{AB} : Old key shared between A and B

K'_{AB} : New key shared between A and B

K_{CA} : Key shared between C and A

K_{CB} : Key shared between C and B

K_{AB} : New communication key shared between A and B

Figure 4.7: Various cases of key distribution

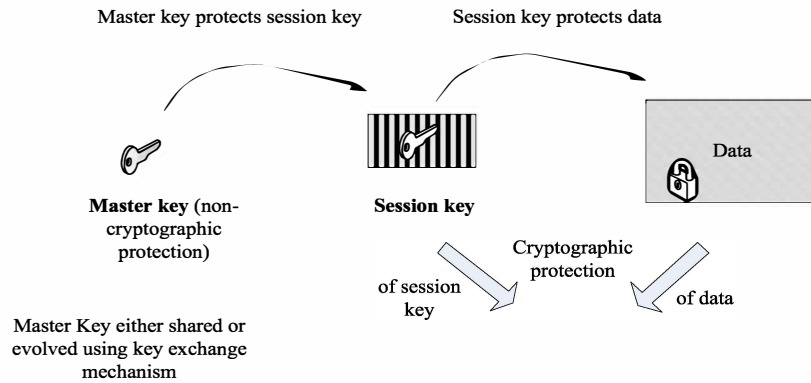


Figure 4.8 Key Hierarchy

For link encryption case 1 and 2 may be used as data exchange is done between directly connected nodes by link. These cases are not useful for end to end encryption as this means for a node communicating with N other nodes, number of keys required is $N(N-1)/2$. A network may have large number of processes as compared to number of nodes which means number of keys for process communication (end to end application encryption) is much greater than $N(N-1)/2$. Case 3 is useful to both link and end to end encryption, with almost same number of initial key distribution. This case also has a drawback that upon compromising a key all further keys generated will also be compromised. Case 4 is most

widely used method where a key distribution center (KDC) plays the role of third party and is responsible for handing over the communication keys to the two parties. It is assumed that the handing over of key to node takes place by encrypting it via key shared between KDC and node. Thus KDC uses two level hierarchies of keys (**Figure 4.8**). One between the two communicating nodes (termed as session or temporary key) other between KDC and a node (termed as master key). The master key is unique and is distributed by KDC which means for a network with N communicating nodes N master keys are there and at most $N(N-1)/2$ session keys are there. Delivery of master key is carried out physically (**Figure 4.9**).

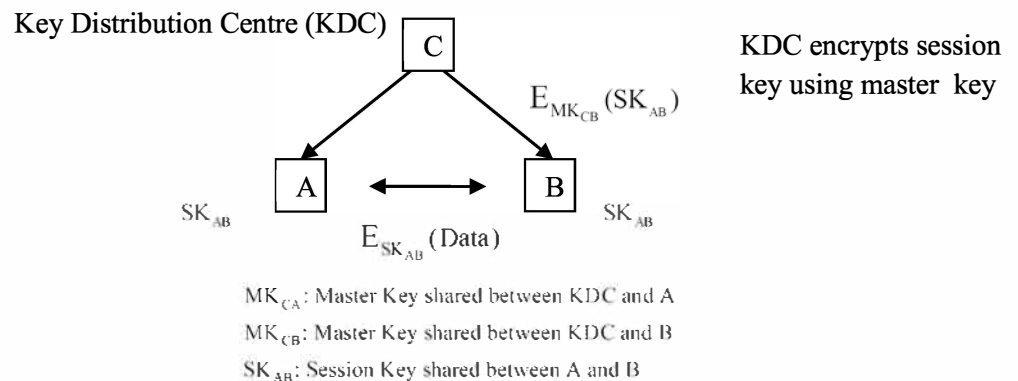


Figure 4.9 Session Key distribution by KDC

Scheme 1 a scenario for Key Distribution

As discussed in previous sub section a secret master key is shared between a user (node) and KDC. **Table 4.2** shows assumed simplified parameters for the system.

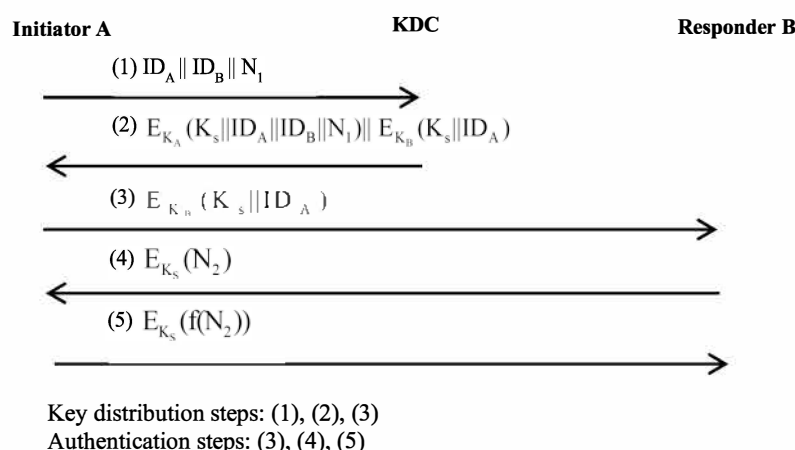
Table 4.2 Assumed simplified parameters for key distribution

K_A	Master Key of A shared with KDC
K_B	Master Key of B shared with KDC
A	Initiator
B	Responder
K_s	Session Key between A and B
$E_K(M)$	Message M encrypted with key K

The following steps are taken to establish the session key:

1. A sends request to KDC containing ID of A and B along with nonce N_1 . A nonce is a number that is used once and is basically used to prevent masquerading. This number cannot be guessed rather it is either time stamp or counter or random number which differs in each request.
2. Upon receiving the request KDC replies back. The reply has two parts, one for initiator node A other for responder node B. Both the parts are encrypted for protecting from eavesdropper. The initiator part is encrypted using K_A while responder part is encrypted using K_B . Thus, A and B can only read their parts. Initiator part contains session key (K_s) to be used in forthcoming session and the original contents of initiator's request i.e. $ID_A || ID_B || N_1$. The responder part contains session key and ID_A .
3. Upon receiving initiator part, node A extracts and stores session key and forwards the responder part to B. Upon receiving responder part, node B establishes the identity of A and extracts the session key.
4. B selects a nonce N_2 , encrypts using session key and sends the encrypted message to A.
5. A applies function $f()$ to N_2 , encrypts using session key and sends the encrypted message to B.

Steps 1 to 3 performs key distribution tasks while 4 to 5 (and also 3) performs authentication (**Figure 4.10**)

**Figure 4.10** A scenario having distribution of session key

Hierarchical KDC structure

A hierarchy of KDC is maintained: Local KDC's for local domain (who are responsible for key distribution in local domain) global KDC's for communication between local KDC's.

Hierarchical structure has two advantages 1. It reduces the master key distribution efforts 2. It limits the damage of compromised KDC to local area.

Lifetime of Keys

Ideally a key in cryptography should be changed every time with transfer of new packet. On the other side in the KDC based system sharing a new key for session is complex and involves time. Therefore a balance between these two must be maintained. Connection oriented protocols are inclined towards changing the key per new session. For longer sessions this changing is done periodically. Connectionless protocols do not involve connection initiation or termination and are inclined towards changing the key periodically

or with passage of fixed number of transactions. If more security is required new session key for each data unit may be used.

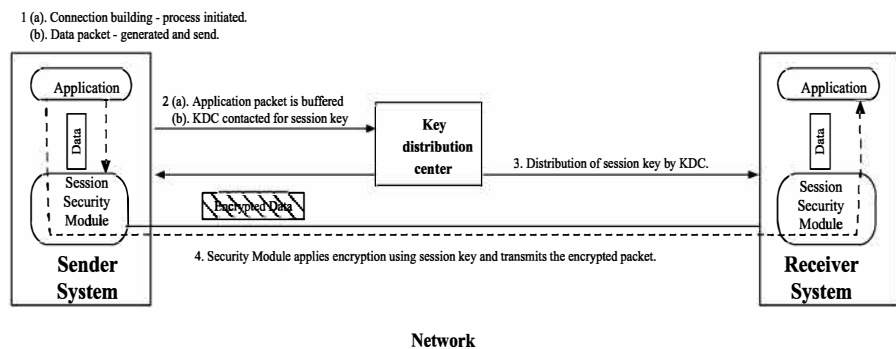


Figure 4.11 A scenario having transparent key control

Initiator A

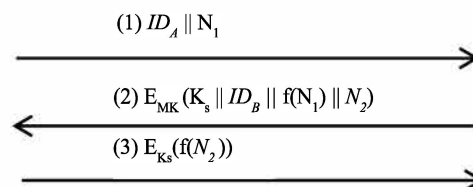


Figure 4.12 A scenario having decentralized key control

Scheme 2 Transparent key control

In this scheme (**Figure 4.11**) end to end encryption is provided in such a manner that it becomes transparent to the users. Session security module (SSM) executes end to end encryption and evolves new session keys. Four steps are involved:

1. Application requests the KDC. This request is stored by SSM.
2. Ask KDC for new session keys.
3. KDC replies back giving one time key.
4. This reply is verified at host B.

SSM can now release the connection as the new connection is established between A and B.

Scheme 3 Decentralized Key Control

Decentralized Key Control (**Figure 4.12**) requires $N(N-1)/2$ master keys for N system networks. Steps involved are:

1. A sends request to B containing nonce N_1 .
2. B's response is basically a message encrypted using master key. This message contains encrypted information. Other parts include session key selected by B, ID_B , value N_2 and $f(N_1)$.
3. A applies function f to N_2 and returns $f(N_2)$ to B.

Methods for controlling/limiting the Key Usage

Key Types

1. Master Key – used throughout
2. Session Key – Used for new sessions and divided in 3 categories on the basis of use:

Data encryption key- for general communication

PIN encryption key- for Personal Identity numbers (PINs)

File Encryption Key-for securing stored files

For controlling/limiting the way in which key is used, two methods exist:

1. A tag based method where a tag is associated with each key. For example in DES cipher this tag considers 8 bits in each 64 bit DES key. One bit of the tag identifies whether it is session or master key, one bit identifies whether it can be utilized for encryption other one bit identifies whether it can be utilized for decryption. Remaining 5 bits are left for future use. This tag is also encrypted along with the key during key distribution process. The demerits of the scheme are: (1) That it is less flexible (small fixed tag size) (2) Can be used in limited manner as it has to be decrypted before use.
2. Control vector scheme- A control vector is associated with each session key. It has field that tells the use and restrictions on session key. **Figure 4.13** shows how controlled vector is used coupled and de coupled with key. First controlled vector is hashed followed by EXORing with master key. The output key is used for encrypting the session key. Reverse mechanism is applied on the receiver side. The controlled vector scheme has two advantages as compared to tag based scheme:

1. Length of control vector may vary implying complex controls may be added.
2. Control vector is used in clear implying that it may be used in multiple encryption locations.

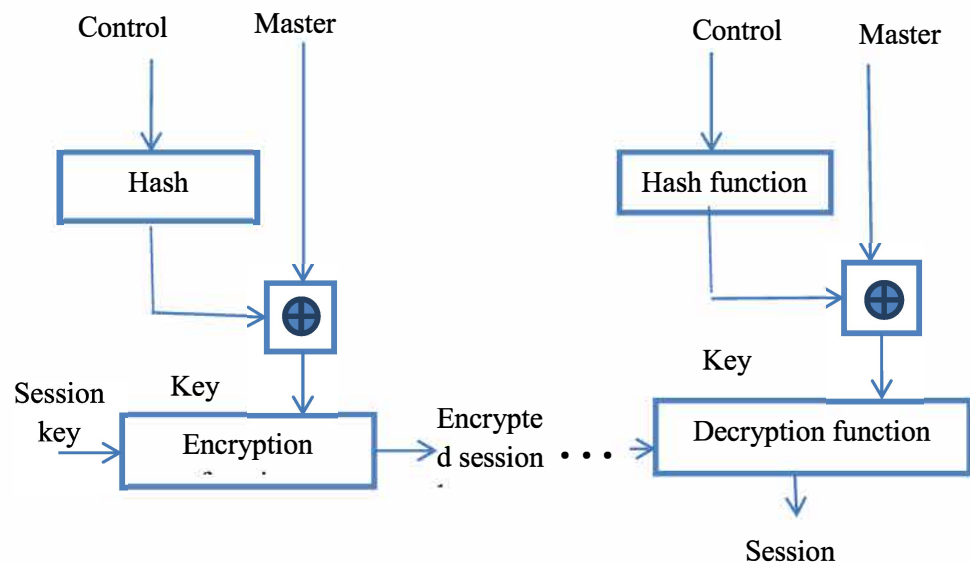


Figure 4.13 Use of Control Vector for encryption/decryption

Check your progress 1

- a. How keys are used? Why is key lifetime required?
- b. Define: Various keys used in cryptography and KDC.

4.5 METHODS FOR GENERATING RANDOM NUMBERS

Several security applications based on cryptography depends upon random numbers for their proper functioning. Some uses of random numbers are:

- (i) Key distribution schemes often utilize nonce for preventing replay attacks. Use of random numbers as nonce makes the attacker's task difficult.
- (ii) Each of the session key generated should be different from the previous ones. Hence, random number generations are often used in the session key generation process.
- (iii) Random numbers are also used as keys in public encryption algorithms like RSA.

The random numbers generated as sequence must conform to two properties namely, randomness and unpredictability.

Randomness: Statistically random sequences are often required in cryptography. The following criteria are used to test the randomness property of a sequence:

Uniform distribution: If frequency of occurrence of each number in a sequence is same, the sequence is said to be uniformly distributed.

Independence: This property states that there is no possibility of inferring a value from other values in the sequence i.e., each number is independent of the other.

Unpredictability: Talks about value that will be generated in future. It says that on the basis of existing values it is not possible to predict future values.

Note/Tip: True random numbers are not much used in cryptography instead pseudorandom numbers (numbers that are almost similar to random ones) are utilized in cryptography.

Pseudorandom Number Generators (PRNGs): Instead of pure random number sequences pseudorandom numbers that pass many reasonable random tests are used in cryptography. Cryptography utilizes algorithm for generating random numbers. These algorithms are deterministic and generate pseudorandom numbers. Thus due to practicality, pure random number concept is seldom used in cryptography [3].

Linear Congruential Generators (LCG): Most widely used PNG method [1][4]. It generates sequence of integral random numbers $\{X_n\}$ as per the iterative equation (1).

$$X_{n+1} = (a X_n + c) \bmod m \quad - (1)$$

where, m is the modulus ($m > 0$)

a is the multiplier ($0 < a < m$)

c is the increment ($0 \leq c < m$)

X_0 is the seed or initial value ($0 \leq X_0 < m$)

Produced integral sequence numbers lie in the range $0 \leq X_n < m$.

Few LCG Cases for finding period and sequence:

LCG case (i) $a = 7, c=0, m=32, X_0=1$

Sequence generated: $\{7, 17, 23, 1, 7 \dots \text{repeated values}\}$

This implies out of 32 possible values only 4 values (termed as period) are used. Hence, the sequence is not satisfactory.

LCG case (ii) $a = 5, c=0, m=32, X_0=1$

Sequence generated: $\{5, 25, 29, 17, 21, 9, 13, 1, 5 \dots \text{repeated values}\}$

This again implies that out of 32 possible values only 8 values are used i.e., period is increased to 8. Still 24 are unused. Hence, the sequence is still not satisfactory.

A random number generator is evaluated using the following three tests [5]:

- (i) The function should generate all the numbers between 0 and m , before any of the value is repeated i.e., it should be a full period generating function.
- (ii) Statistical tests should be used to analyze the degree of randomness.
- (iii) The function should be able to work with conventional 32 bit arithmetic.

LCG Case (iii) m is prime, $c = 0$, $m = 2^{31} - 1$ (for conventional 32 bit arithmetic)

The function may be rewritten as

$$X_{n+1} = (a X_n) \bmod (2^{31} - 1) \quad - (2)$$

Thus, value of a needs to be selected such that all 3 tests for randomness listed above may pass. One such widely used generator for statistical and simulation work considers $a = 7^5 = 16807$ (used with IBM 360 family of computers)

Some points regarding LCG:

- (1) LCG works fine if the multiplier and modules are properly chosen. It then generates sequence (considering no repetitions) that is similar to sequence drawn from 0 – ($m-1$) at random (without replacements).
- (2) The only value that change is X_0 and it is fed only once, the LCG sequence is followed deterministically. So, if an attacker knows a small part of sequence generated, the entire sequence may be determined.
- (3) For making LCG non reproducible, internal system clock use is suggested [6] in the following ways:
 - (i) Sequence is restarted after N numbers by selecting new seed as – current clock value (mod m)
 - (ii) Addition of current clock value to each random number (mod m).

Random number generators that use cryptographic mechanisms

Different random number generators that use cryptographic mechanisms are presented below:-

Cyclic Encryption: In cyclic encryption technique (**Figure 4.14**) secure master key is used for encrypting a counter, the encrypted value generated is the random value. The counter is incremented every time a random number is required. The technique is secure as master key is kept safe and is not available to attacker. Also, as each of the random numbers generated ($X_0, X_1, \dots, X_{N-1}$) utilizes different counter value (i.e., increased counter value), all random numbers are different from each other.

ANSI X9.17 PRNG: An ANSI specified PRNG (**Figure 4.15**) used mainly in financial security applications and E-mail security schemes (like PGP). It is one of the (cryptographically) strongest PRNGs. Two

key triple DES (3DES) algorithms are used in encrypt-decrypt-encrypt manner in this PRNG. Two keys are termed as K_1 and K_2 .

The following are some of the noticeable points regarding this PRNG:

- (1) Date/time value (DT_i) is provided as input at the start of i^{th} stage. This value keeps on changing and is updated at every stage.
- (2) A seed (V_i) is supplied. So the new seed (V_{i+1}) is generated as output by i^{th} stage

$$R_i = 3DES_{(K_1, K_2)}^{3DES_{(K_1, K_2)}} (V_i \oplus \oplus 3DES_{(K_1, K_2)}^{3DES_{(K_1, K_2)}} DT_i)$$

- (3) A pseudorandom number (R_i) is produced as output in the i^{th} stage.

$$V_{i+1} = 3DES_{(K_1, K_2)}^{3DES_{(K_1, K_2)}} (R_i \oplus \oplus 3DES_{(K_1, K_2)}^{3DES_{(K_1, K_2)}} DT_i)$$

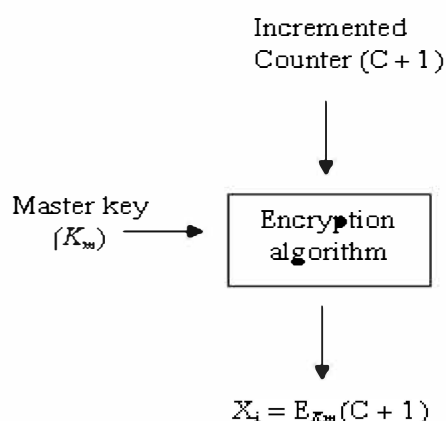
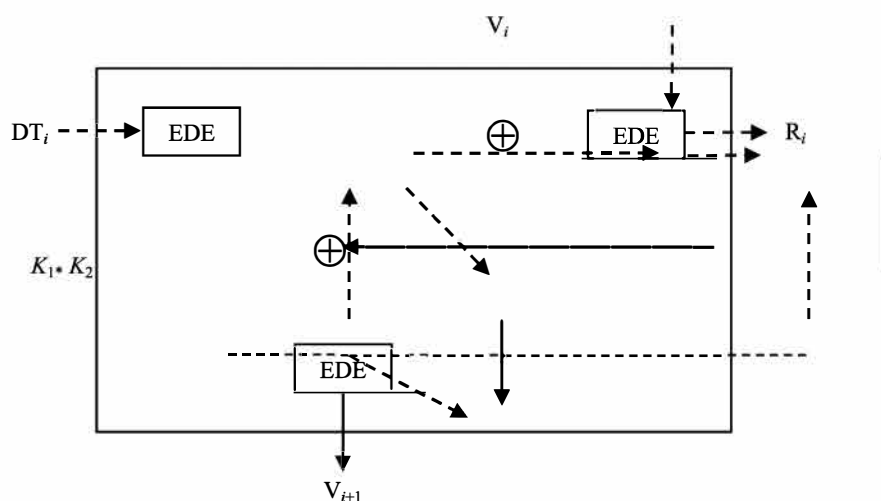


Figure 4.14 PRNG generation using counter [7]



EDE: Encrypt-Decrypt-Encrypt

Figure 4.15 ANSI X9.17 PRNG

- The technique is secure as 3DES is used 3 times in this technique, implying total of 9 single DES.

Also, different 3DES encryptions generated separate V_{i+1} and R_i so that if one is compromised, the system will still work.

- The amount of material required by attacker is too much i.e., 2 different pseudo random numbers (date/time and seed) are considered as input and the seed considered is different than pseudo random number generated by system.

Blum Blum Shub (BBS): Another cryptographic technique known as Blum Blum Shub (BBS) generator is named after name of its developers [8]. It consider two large prime numbers such that

$$p \equiv q \equiv 3 \pmod{4}$$

The above notation means that remainder after division of either p or q by 4 is 3. A number s is also selected such that it is relatively prime to n ($n = p \times q$) which means neither p nor q is factor of s .

BBS algorithm produces sequence of bits B_i as shown below:

$$\begin{aligned} X_0 &= s^2 \pmod{n} \\ \text{for } i &= 1 \text{ to } \infty \\ X_i &= (X_{i-1})^2 \pmod{n} \\ B_i &= X_i \pmod{2} \end{aligned}$$

BBS algorithm is treated secure because there is no algorithm through which predicting $(K + 1)^{\text{st}}$ bit from first k bits of output has probability greater than $\frac{1}{2}$ (This test is termed as next bit test). In other words, the sequence generated is unpredictable [9].

Also, breaking of algorithm BBS depends upon factoring n (i.e. finding some factors p and q) which is difficult.

The computer system security utilizes PRNGs. TRNG (True Random Number Generators) on the other side use nondeterministic source like natural processes (e.g. gas discharge tubes) for producing randomness. For such TRNG special chips/hardware is required. A TRNG may generates numbers that are biased in some way (skewness) and therefore may also require deskewing algorithms, like use of hash function on the generated bit stream.

Check your progress 2

- a. Why are random number generators used in cryptography and network security?
- b. Define: seed and period in random number generators.

4.6 SUMMARY

In this unit, you have learnt about the use and placement of symmetric encryption function for providing confidentiality. As per

the requirements, symmetric encryption may be positioned either at the link layer or at end to end communication layer. Role of symmetric encryption function for countering the traffic analysis attacks is discussed in this unit. A key hierarchy commonly used in network security is presented and notion of key distribution centre (KDC) is introduced. Issues related with key distribution and storage are also discussed along with. Functionality of various random number generators like ANSI X9.17 PRNG, BBS is explained towards end. It helps us to realize that the true random numbers are rare instead pseudo random numbers are more useful in security.

Terminal Questions

1. *Differentiate between:*
 - (i) *Link and End to end encryption*
 - (ii) *Session key and master key*
2. *Define:*
 - (i) *Nonce*
 - (ii) *Key distribution centre.*
 - (iii) *Pseudo random number generator*
3. *How traffic analysis helps the attacker?*
4. *State the purpose of traffic padding.*
5. *In how many ways can a symmetric key be distributed among two communicating partners?*
6. *Find the maximum period, and value of a in the following generator.*

$$X_{(n+1)} = (aX_n) \bmod 2^4$$
7. *Can XORing successfully be used in cryptography? Consider the following scenario where sender and receiver both XORs the key with random bit string and exchanges the result with each other. Find the issues in this scenario.*

References:

1. Stallings, William, "Chapter 7: Confidentiality Using Symmetric Encryption", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
2. Pfleeger, C. Security in Computing. Upper Saddle River, NJ: Prentice Hall, 2002.
3. Pseudo Random Number Generator.
https://en.wikipedia.org/wiki/Pseudorandom_number_generator . accessed on 16.12.2016.
4. Cryptographically secure pseudorandom number generator
https://en.wikipedia.org/wiki/Cryptographically_secure_pseudo_random_number_generator. accessed on 16.12.2016.
5. Park, S., and Miller, K. "Random Number Generators: Good Ones are Hard to Find." Communications of the ACM, October 1988.

6. Bright, H., and Enison, R. "Quasi-Random Number Sequences from Long-Period TLP Generator with Remarks on Application to Cryptography." *Computing Surveys*, December 1979.
7. Meyer, C., and Matyas, S. *Cryptography: A New Dimension in Computer Data Security*. New York: Wiley, 1982.
8. Blum, L.; Blum, M.; and Shub, M. "A Simple Unpredictable Pseudo-Random Number Generator." *SIAM Journal on Computing*, No. 2, 1986.
9. Menezes, A.; van Oorschot, P.; and Vanstone, S. *Handbook of Applied Cryptography*. Boca Raton, FL: CRC Press, 1997.



**Uttar Pradesh Rajarshi Tandon
Open University**

Master of Computer Science

**MCS -119
INFORMATION AND NETWORK
SECURITY**

Block

2

**PUBLIC KEY ENCRYPTION AND
HASH FUNCTIONS**

Unit 5	91-106
Introduction to Number Theory	
Unit 6	107-130
Public Key Cryptography	
Unit 7	131-150
Message Authentication and Hash Function	
Unit 8	151-166
Digital Signature	

Course Design Committee

(Prof.) Ashutosh Gupta Director (In-charge) School of Computer and Information Science, UPRTOU, Prayagraj	Chairman
Prof. R. S. Yadav Department of Computer Science and Engineering MNNIT-Allahabad, Prayagraj	Member
Dr. Marisha Assistant Professor (Computer Science), School of Science, UPRTOU, Prayagraj	Member
Dr. C. K. Singh Lecturer School of Computer and Information Science, UPRTOU, Prayagraj	Member

Course Preparation Committee

Dr Rajeev Singh Assistant Professor Department of Computer Engineering GB Pant University of Agriculture and Technology Pantnagar, Uttarakhand	Author
Prof. Abhaya Saxena Head, Dept. of Computer Science Dev Sanskriti Vishwavidyalaya Hardwar, Uttarakhand	Editor
Prof. Ashutosh Gupta Director (In-charge), School of Computer and Information Science UPRTOU, Prayagraj	
Dr. Marisha Assistant Professor (Computer Science) School of Science UPRTOU, Prayagraj	Coordinator

© UPRTOU, Prayagraj. 2023
ISBN : 978-93-83328-27-7

All Rights are reserved. No part of this work may be reproduced in any form, by mimeograph or any other means, without permission in writing from the Uttar Pradesh Rajarshi Tondon Open University, Prayagraj.

Printed and Published by Vinay Kumar, Registrar, Uttar Pradesh Rajarshi Tandon Open University, 2023.

Printed By: K.C.Printing & Allied Works, Panchwati, Mathura -281003.

UNIT - 5

Introduction to Number Theory

Structure

- 5.0 Introduction
- 5.1 Objectives
- 5.2 Concept of Prime Numbers
- 5.3 Functioning of Fermat's and Euler's theorem
- 5.4 Fundamentals of Discrete Logarithms
- 5.5 Summary
- 5.6 Terminal Questions

5.0 INTRODUCTION

In this unit the operations and functions performed in the cryptographic domain are presented. The basic understanding of the terminology and definitions of integer arithmetic e.g. set of integers ($\mathbb{Z}$), integer division, prime numbers, modular arithmetic, set of residues, additive inverse, multiplicative inverse etc. is provided. The prime numbers are introduced along with the notion of relatively prime (coprime). The important properties and use of prime numbers and coprimes are explained. Two sets $\mathbb{Z}_n^*$ and $\mathbb{Z}_p^*$ are frequently used in cryptography and security. The former, represents set whose members are relatively prime to modulus n whereas latter represents set whose members are relatively prime to modulus p .

Some other useful concepts discussed in this unit are: Euler's Phi Function, Fermat's little theorem and Euler's Theorem. Euler's Phi Function, $\Phi(n)$ (also known as totient function) finds out number of integers smaller than n and relatively prime to n i.e., $\Phi(n)$ calculates number of elements in $\mathbb{Z}_n^*$. Fermat's little theorem is useful in finding quickly some exponentiation and multiplicative inverses. Euler's Theorem is basically a generalization of Fermat's little theorem that considers modulus as integer instead of prime. Toward end, the exponentiation and logarithm are covered in brief. Exponentiation and Logarithm are inverses of each other. In cryptography, exponentiation is used with modular operation i.e., $y = a^x \bmod n$. A public key encryption system requires large exponents (i.e., x). Hence, for finding inverse i.e., logarithm in public key encryption system, exhaustive

search method is very inefficient. So another approach termed as discrete logarithm approach is used instead that includes finding primitive roots (generators). Primitive roots are the one that can generate entire set from themselves.

5.1 OBJECTIVES

After the end of this unit, you should be able to:

- Understand the terms and concepts like prime number, modular arithmetic, set of residues, additive inverse, multiplicative inverse etc.
- List the elements of Z_n^* and Z_p^* .
- Calculate number of elements in Z_n^* using Euler's Phi Function.
- Find out primitive roots and able to generate entire set from them.
- Quickly find out the exponentiation and multiplicative inverses using Fermat's little theorem and Euler's Theorem.

5.2 CONCEPT OF PRIME NUMBERS

Before discussing the operations and functions performed in the cryptographic domain, let us first understand basic terminology and definitions of integer arithmetic [1]. This will create the background for modular arithmetic used in cryptographic functions.

In this unit only non-negative integers are considered. The non-negative integers are grouped into three categories:

- (i) Number 1 that has exactly one divisor
- (ii) Prime numbers that have exactly 2 divisors (e.g., 7 is a prime number as it has two divisors 1 and 7 itself) and
- (iii) Composite numbers that have more than two divisors (e.g., 10 is a composite number as it has three divisors 1, 2 and 5).

The set of integers (Z): It contains all integers from negative infinity to positive infinity i.e.,

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

Integer Division: Operation - a divides b implies that the remainder on division is 0 (a is divisor of integer b). In other words, for any two integers a and n , if we divide a by n , we can get quotient (q) and remainder (r). q and r are also integers, the relationship between all four integers is as:

$$a = q \times n + r$$

$$(\text{dividend} = \text{quotient} \times \text{divisor} + \text{remainder})$$

The division process is shown in **Figure 5.1**

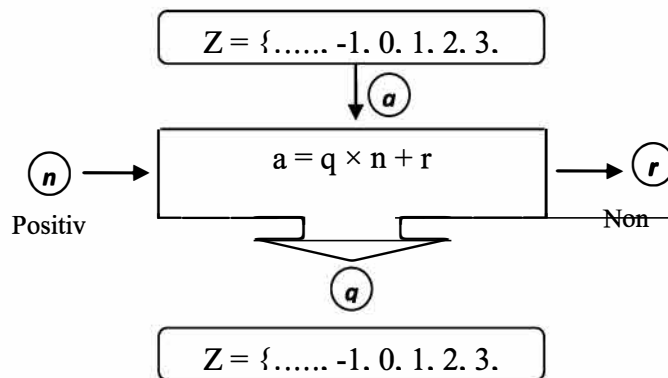


Figure 5.1 Integer division process

Example 1: Consider, $a = 255$ and $n = 11$. Find q and r using the integer division?

Answer: $q = 23$ and $r = 2$

Prime numbers are represented by letter p and they play an important role in cryptography.

Definition: An integer is prime number if and only if its divisors are 1 and the number itself.

Modular Arithmetic

In modular arithmetic, the operator is termed as mod (or modulo operator). The input n is termed as *modulus* and the output (remainder r) is termed as *residue*.

Thus, integer division operation ($a = q \times n + r$) involves two inputs (a and n) and two outputs (q and r) whereas mod operation involves only one input (a) and one output (the remainder r).

The comparison between the integer division and modular arithmetic is shown in **Figure 5.2 [1]**

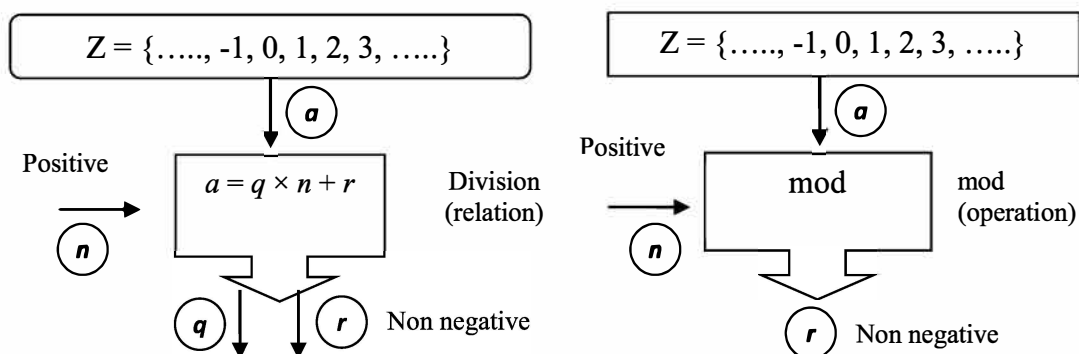


Figure 5.2 Comparison between the integer division and modular arithmetic

Set of Residues

The set created by modulo operation is referred to as the set of least residues modulo n , or Z_n .

$$Z_n = \{ 0, 1, 2, 3, \dots, (n-1) \}$$

Example 2: Write Z_2 , Z_4 , Z_8 and Z_{13}

Answer: $Z_2 = \{ 0, 1 \}$; $Z_4 = \{ 0, 1, 2, 3 \}$;
 $Z_8 = \{ 0, 1, 2, 3, 4, 5, 6, 7 \}$;
 $Z_{13} = \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 \}$

Additive Inverse

In modular arithmetic, each integer in set Z_n has an additive inverse.

It can be found using the following:

$$a + b \equiv 0 \pmod{n}$$

i.e., the two numbers a and b are additive inverses of each other if, sum of an integer and its additive inverse is congruent to 0 modulo n .

Example 3: How many additive inverse pairs lie in Z_{12} . Find them?

Answer: There are seven pairs of additive inverses that lie in Z_{12} . These are:

(0, 12), (1, 11), (2, 10), (3, 9), (4, 8), (5, 7), and (6, 6).

Multiplicative Inverse

In modular arithmetic, it is not necessary that an integer have a multiplicative inverse i.e., it may or may not have it.

If the multiplicative inverse exists, it follows the following relationship:

$$a \times b \equiv 1 \pmod{n}$$

i.e., the product of the integer and its multiplicative inverse is congruent to 1 modulo n .

An integer 'a' has multiplicative inverse in set 'Z_n' if and only if GCD (n, a) = 1 and in such case 'a' and 'n' are said to be relatively prime to each other [1].

Example 4: How many multiplicative inverse pairs lie in Z_{12} . Find them?

Answer: There are only four pairs. These are:

(1, 1), (5, 5), (7, 7) and (11, 11)

Thus, numbers 0, 2, 4, 6, 8, 9 and 10 in Z_{12} do not have a multiplicative inverse.

Example 5: How many integer numbers have inverse in Z_{13} ?

Answer: As in Z_{13} , GCD (13, a) is 1 for all values of a from 1-12 except 0 i.e., 13 is relatively prime to all numbers in set except 0. Hence, there are twelve integer numbers that have inverse in Z_{13} . They form pairs: (1, 1), (2, 7), (3, 9), (4, 10), (5, 8), (6, 11) and (12, 12).

NOTE:

This set containing multiplicative inverses is termed as Z_n^*

In simpler terms, all members of set Z_n have additive inverses but they may or may not have their multiplicative inverses whereas on the other side, all members of set Z_n^* have multiplicative inverse but they may or may not have their additive inverses.

Example 6: Write Z_6 , Z_{10} and Z_{11} . Also, write Z_6^* , Z_{10}^* and Z_{11}^* .

Answer:

$$Z_6 = \{0, 1, 2, 3, 4, 5\}$$

$$Z_6^* = \{1, 5\}$$

$$Z_{10} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$$

$$Z_{10}^* = \{1, 3, 7, 9\}$$

$$Z_{11} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

$$Z_{11}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$$

In short, we need to use Z_n when additive inverses are needed; we need to use Z_n^* when multiplicative inverses are needed.

In cryptography, $n = p$ (a prime number) i.e., Z_p is used instead of Z_n , The advantage of using this set is that Z_p^* set contains members that all have additive as well as multiplicative inverses.

$$Z_p = \{0 \text{ to } (p-1)\}$$

$$Z_p^* = \{1 \text{ to } (p-1)\}$$

$$\text{For } p = 13, Z_p^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$$

Let us now concentrate upon some more properties/use of prime numbers.

An integer a can be factored in a unique way using fundamental theorem of arithmetic that can be stated as:

$$a = p_1^{a_1} p_2^{a_2} \dots p_t^{a_t} \quad - (1)$$

where $p_1 < p_2 < \dots < p_t$ are prime numbers and each a_i is a positive integer. This can also be expressed as:

$$a = \prod_{p \in P} p^{a_p} \text{ where each } a_p \geq 0 \quad - (2)$$

Right hand side of the above equation is basically product of all possible prime numbers p .

$$\text{Ex, } 91 = 7 \times 13 = 7^1 \times 13^1 \text{ all other exponents } a_p \text{ are zero}$$

$$11011 = 7 \times 11^2 \times 13$$

The value of any positive integer is represented by listing all non-zero exponents. Thus, 91 is represented as $\{a_7 = 1, a_{13} = 1\}$

$$11011 \text{ is represented as } \{a_7 = 1, a_{11} = 2, a_{13} = 1\}$$

Thus, any number can be represented as product of primes.

Suppose $a = \prod_{p \in P} p^{a_p}$ and $b = \prod_{p \in P} p^{b_p}$ then $k = a.b = \prod_{p \in P} p^{k_p}$

such that, $k_p = a_p + b_p$, for all $p \in P$

Thus, multiplication of two numbers is equal to adding their corresponding exponents and followed by multiplication [2].

Example 7: Consider, $a = 24$, $b = 18$

$$a = 2^3 \times 3, b = 2 \times 3^2$$

$$k = a \times b = 2^{3+1} \times 3^{1+2} = 2^4 \times 3^3 = 16 \times 27 = 432$$

Similarly, if $a|b$, then $a_p \leq b_p$ for all p

Example 8: $a = 12$, $b = 48$

$$12 | 48$$

$$a = 12 = 2^2 \times 3 \Rightarrow a_2 = 2, a_3 = 1$$

$$b = 48 = 2^4 \times 3^1 \Rightarrow b_2 = 4, b_3 = 1$$

Thus, $a_2 < b_2$ and $a_3 = b_3$ which implies that $a_p \leq b_p$ is satisfied for all prime numbers.

GCD (Greatest Common Divisor) of two numbers is the largest number that divides both the numbers. For finding GCD, Euclidean and extended Euclidean algorithm is used [1].

The fact that a positive integer is represented as product of primes is also utilized to find gcd of two numbers a & b as:

$$\text{For } k = \text{gcd}(a, b) \text{ each } k_p = \min(a_p, b_p)$$

Example 9: Find Gcd (360, 756)

$$\text{Answer: } 360 = 2^3 \times 3^2 \times 5^1$$

$$756 = 2^2 \times 3^3 \times 7^1$$

$$\text{Gcd}(360, 756) = 2^2 \times 3^2 \times 5^0 \times 7^0 = 36$$

Definition: If GCD of two numbers is equal to one then the numbers are said to be coprime or relatively prime.

Summing up,

- For a prime number p , all integers from 1 to $p-1$ are relatively prime to p
- Z_n^* denotes set whose members are relatively prime to modulus n
- Z_p^* denotes set whose members are relatively prime to modulus p

The number of primes is infinite. Still it is difficult to find out number of primes smaller than or equal to a given number n (consider n as large number)

Next, we discuss two theorems namely, Fermet's theorem and Euler's theorem that are considered important in public key cryptography.

5.3 FUNCTIONING OF FERMET'S AND EULER'S THEOREM

Euler's Phi Function, $\Phi(n)$

It is also referred to as Euler's totient function. It finds out number of integers smaller than n and relatively prime to n i.e., $\Phi(n)$ calculates number of elements in Z_n^*

$\Phi(n)$ can be calculated as:

- (i) If $n = 1$, $\Phi(1) = 0$
- (ii) If p is prime, $\Phi(p) = p-1$
- (iii) If m and n are relatively prime, $\Phi(m \times n) = \Phi(m) \times \Phi(n)$
- (iv) If p is a prime number, $\Phi(n)$ is expressed exponentially as $\Phi(p^e) = p^e - p^{e-1}$

Suppose a number ' q ' can be factored as $p_1^{e_1} \times p_2^{e_2} \dots p_k^{e_k}$ then

(iii) and (iv) can be combined as $\Phi(q) = (p_1^{e_1} - p_1^{e_1-1}) \times (p_2^{e_2} - p_2^{e_2-1}) \times \dots (p_k^{e_k} - p_k^{e_k-1})$

Table 5.1 lists some of the cases of calculating $\Phi(n)$ [2] [3].

Table 5.1 some $\Phi(n)$

S.No.	Find	Solution
1.	$\Phi(7)$	$\Phi(7) = 7-1 = 6$
2.	$\Phi(8)$	$\Phi(8) = \Phi(2^3) = 8 - 4 = 4$
3.	$\Phi(10)$	$\Phi(10) = \Phi(2) \times \Phi(5) = 1 \times 4 = 4$
4.	$\Phi(240)$	$240 = 2^4 \times 3^1 \times 5^1 \Rightarrow \Phi(240) = (2^4 - 2^3) \times (3^1 - 3^0) \times (5^1 - 5^0) = 64$
5.	$\Phi(49)$	$\Phi(49) = 7^2 - 7^1 = 42$
6.	No of element in Z_{14}^*	$\Phi(14) = \Phi(7) \times \Phi(2) = 6 \times 1 = 6$. Therefore, 6 elements are there. These members (in Z_{14}^*) are: 1, 3, 5, 9, 11, 13

In second last example factoring $\Phi(49)$ as $\Phi(7) \times \Phi(7)$ is not correct as 7 is not relatively prime to 7. Hence, rule 3 is not applicable.

Difficulty of finding $\Phi(n)$ depends upon difficulty of finding factors of n

Fermat's little theorem

It is available in the following two versions:

I) $a^{p-1} \equiv 1 \pmod{p}$

where, a is an integer and p is a prime and p does not divides a i.e., a and p are relatively prime to each other.

- II) $a^p \equiv a \pmod{p}$, this version has no relatively prime constraint on a and n

Use – Fermat's little theorem is useful in finding quickly some exponentiation and multiplicative inverses.

Multiplicative Inverse can be evaluated as:

$$a^{-1} \pmod{p} = a^{p-2} \pmod{p}$$

Thus, use of extended Euclidean algorithm for finding an inverse is avoided.

Example 10: (a) Find exponentiation: $2^{402} \pmod{11}$?

(b) Find exponentiation: $3^{12} \pmod{11}$?

Answer: (a) $2^{10} \pmod{11} = 1 \pmod{11}$ [Using first version of Fermat's theorem]

$$\text{Thus, } 2^{402} \pmod{11} = (2^{10})^{40} * 2^2 \pmod{11} = 4 \pmod{11}.$$

$$(b) 3^{12} \pmod{11} = (3^{11} \times 3) \pmod{11} = (3^{11} \pmod{11}) \times (3 \pmod{11})$$

$$= 3 \times 3 = 9 \pmod{11} \text{ [Using second version of Fermat's theorem]}$$

Euler's Theorem

It is basically a generalization of Fermat's little theorem that considers modulus as integer instead of prime [2] [3].

- I) $a^{\Phi(n)} \equiv 1 \pmod{n}$ for two coprimes a and n .

- II) Second version removes the coprime condition between a and n . It is stated as

$$a^{K \times \Phi(n) + 1} \equiv a \pmod{n}$$

where, $n = p \times q$, $a < n$ and K is an integer

The second version of Euler's theorem is used in RSA cryptosystem.

Uses [1]:-

- (i) Finding of some exponentiations

$$\begin{aligned} \text{Example 11: } 20^{62} \pmod{77} &= (20 \pmod{77}) (20^{\Phi(77)+1} \pmod{77}) \\ &= (20) (20) \pmod{77} = 15 \end{aligned}$$

- (ii) Multiplicative inverses can be calculated as

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

where, a and n are coprime.

$$\text{Example 12: } 60^{-1} \pmod{187} = 60^{\Phi(187)-1} \pmod{187} = 60^{159} \pmod{187} = 53 \pmod{187}$$

Check your progress 1

a. Find:

- (i) $(-18) \bmod 3$;
- (ii) $(1,723,345 + 2,124,945) \bmod 11$
- (iii) all multiplicative inverses in Z_{10}
- (iv) multiplicative inverse of 23 in Z_{100}

b. How is Fermat's little theorem different from Euler's theorem?

c. How do we find multiplicative inverses for square matrices?
[Hint: use determinant]

5.4 FUNDAMENTALS OF DISCRETE LOGARITHM

Exponentiation and Logarithm

- Exponentiation and Logarithm are inverses of each other.
- Exponentiation is expressed as $y = a^x$ and logarithm is expressed as $x = \log_a y$
- Here, a is base of exponentiation or logarithm.
- In cryptography, exponentiation is used with modular operation i.e., $y = a^x \bmod n$
- Such operations utilizing large exponents are used by public key encryption system like RSA algorithm for encryption and decryption.
- Efficient algorithms are required for computing exponentiation operations with large exponents.
- The exhaustive search method for finding modular logarithm i.e., finding solution as $x = \log_a y \bmod n$ by continuously solving $y = a^x \bmod n$, is very inefficient.
- Hence, another approach termed as discrete logarithm is used.
- For understanding the discrete logarithm approach, multiplicative groups and their properties need to be discussed.
- In the following subsection, some of the important concepts related to multiplicative groups are discussed [2] [3].

Finite multiplicative group

- A multiplicative finite group $G = \langle Z_n^*, \times \rangle$ contains multiplicative operation.
- Set Z_n^* contains integers from 1 to $n-1$ that are relatively prime to n .
- The set has 1 as the identity element i.e., $e = 1$.
- As a special case, modulus n may be prime number and hence group becomes

$$G = \langle Z_p^*, \times \rangle$$

Order of the Group

Order of the group is defined as number of elements in that group and is calculated as

$$|G| = \Phi(n).$$

Example 13: What is the order of group $G = \langle \mathbb{Z}_{21}^*, \times \rangle$?

Answer: $|G| = \Phi(21) = \Phi(3) \times \Phi(7) = 2 \times 6 = 12$.

Corresponding 12 elements in this group are: 1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, and 20.

Order of an Element

Order of an element is represented as $\text{ord}(a)$ and is defined as the smallest integer i such that

$$a^i \equiv e \pmod{n}$$

where, e is identity element equal to 1.

Example 14: Find the order of all elements in $G = \langle \mathbb{Z}_8^*, \times \rangle$.

Answer: Order of group $\Phi(8) = 4$

Corresponding elements: 1, 3, 5, 7.

Order of each element (by trial and error) using above definition.

- a. $1^1 \equiv 1 \pmod{8} \rightarrow \text{ord}(1) = 1$.
- b. $3^2 \equiv 1 \pmod{8} \rightarrow \text{ord}(3) = 2$.
- c. $5^2 \equiv 1 \pmod{8} \rightarrow \text{ord}(5) = 2$.
- d. $7^2 \equiv 1 \pmod{8} \rightarrow \text{ord}(7) = 2$.

Euler's Theorem

States that for any member of group $G = \langle \mathbb{Z}_p^*, \times \rangle$ the following holds:

$$a^{\Phi(n)} \equiv e \pmod{n} \quad (3)$$

comparing equation (3) with order of group, it can be found that the relation $a^i \equiv 1 \pmod{n}$ holds when $i = \Phi(n)$. thus, it is ensured that atleast one order of element exists.

Example 15: Find order of all elements in the group $G = \langle \mathbb{Z}_6^*, \times \rangle$

Answer: Order of group $\Phi(6) = \Phi(2) \times \Phi(3) = 1 \times 2 = 2$,

Corresponding elements: 1, 5.

The table (Table 5.2) for $r = a^i \pmod{6}$ is drawn with shaded area identifying $r = 1$ for $i = \Phi(6) = 2$.

Similarly, $r = 1$ for other values of i for every member element of the group. Considering smallest i for every member element, we have their orders as:

$$\text{Ord}(1) = 1, \text{ord}(5) = 2$$

Table 5.2 calculations for $r = a^i \pmod n$

	$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$
$a = 1$	$r: 1$	$r: 1$	$r: 1$	$r: 1$	$r: 1$
$a = 5$	$r: 5$	$r: 1$	$r: 5$	$r: 1$	$r: 1$

Primitive Roots: In the group $G = \langle \mathbb{Z}_n^*, \times \rangle$, an element whose order is equal to $\Phi(n)$, is called the primitive root of the group.

Example:

1. In $G = \langle \mathbb{Z}_8^*, \times \rangle$ (example 14), no primitive roots are there because none of the element's order is equal to $\Phi(8) = 4$. Infact, the order of elements are all smaller than 4.
2. For group $G = \langle \mathbb{Z}_{10}^*, \times \rangle$ $\Phi(n) = 4$. The results of $a^i \equiv r \pmod{10}$ along with primitive roots are shown in **Table 5.3**

Table 5.3 Calculations for $a^i \equiv r \pmod{10}$

	$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$	$i = 6$	$i = 7$	$i = 8$	$i = 9$
$a = 1$	$r: 1$	$r: 1$	$r: 1$	$r: 1$	$r: 1$	$r: 1$	$r: 1$	$r: 1$	$r: 1$
$a = 3$	$r: 3$	$r: 9$	$r: 7$	$r: 1$	$r: 3$	$r: 9$	$r: 7$	$r: 1$	$r: 3$
$a = 7$	$r: 7$	$r: 9$	$r: 3$	$r: 1$	$r: 7$	$r: 9$	$r: 3$	$r: 1$	$r: 7$
$a = 9$	$r: 9$	$r: 1$	$r: 9$	$r: 1$	$r: 9$	$r: 1$	$r: 9$	$r: 1$	$r: 9$



= Primitive root

[ord (3) = 4 and ord (7) = 4]

- In general, a group $G = \langle \mathbb{Z}_n^*, \times \rangle$ has primitive roots only if n is 2, 4, p^t , or $2p^t$.

Example 16: Which of the following groups have primitive roots:

$$G = \langle \mathbb{Z}_{13}^*, \times \rangle ; \langle \mathbb{Z}_{40}^*, \times \rangle ; \langle \mathbb{Z}_{26}^*, \times \rangle ; G = \langle \mathbb{Z}_{98}^*, \times \rangle ?$$

Answer: $G = \langle \mathbb{Z}_{13}^*, \times \rangle ; \langle \mathbb{Z}_{26}^*, \times \rangle ;$ and $G = \langle \mathbb{Z}_{98}^*, \times \rangle$ have primitive roots as in first group $n=13$ is prime, in second group $n=26$ can be expressed as 2×13 and in third group $n = 98$ can be expressed as $98 = 2 \times 7^2$.

Group $\langle \mathbb{Z}_{40}^*, \times \rangle$ has no primitive roots.

Some important points regarding primitive roots:

- A group $G = \langle \mathbb{Z}_n^*, \times \rangle$ is cyclic if it has primitive roots.

- If any primitive root is present in the group $G = \langle Z_n^*, \times \rangle$, then there exists $\Phi(\Phi(n))$ number of primitive roots.
- Each primitive root is termed as generator (g) and is used to generate the entire set.
- Thus, set Z_n^* can be generated as $\{g^1, g^2, g^3, \dots, g^{\Phi(n)}\}$

Example 17:

Find the primitive roots in $G = \langle Z_7^*, \times \rangle$ and show that the entire set can be generated from them.

Answer: $\Phi(\Phi(n)) = \Phi(\Phi(7)) = \Phi(6) = 2$. Hence, two primitive roots 3 and 5 are possible. These can generate the entire set as shown below (Table 5.4):

Table 5.4 Set generated by primitive roots in $G = \langle Z_7^*, \times \rangle$

$g = 3$	$g = 5$
$g^1 \bmod 7 = 3$	$g^1 \bmod 7 = 5$
$g^2 \bmod 7 = 2$	$g^2 \bmod 7 = 7$
$g^3 \bmod 7 = 6$	$g^3 \bmod 7 = 6$
$g^4 \bmod 7 = 4$	$g^4 \bmod 7 = 2$
$g^5 \bmod 7 = 5$	$g^5 \bmod 7 = 3$
$g^6 \bmod 7 = 1$	$g^6 \bmod 7 = 1$

- If $n = p$ then group $G = \langle Z_p^*, \times \rangle$ is cyclic as it always has a primitive root and is hence considered in discrete logarithms.
- Elements in this group include all integers from 1 to $p - 1$.
- The elements can be created using g^x where, g are primitive roots or generator and x is an integer from 1 to $\Phi(n) = p - 1$.

In modular arithmetic discrete logarithms are used to solve the problems of the form $y = a^x \pmod{n}$ in which y is given and x is needed to be solved.

Discrete logarithms represented as L_g and considers primitive roots or generators (g) in a group $G = \langle Z_p^*, \times \rangle$ as the base of logarithm i.e., $\log_g y$.

A table is maintained (similar to log tables used to calculate logarithms base 10) that contains discrete logarithms with bases (primitive roots or generators) in rows while y lies in the columns.

In this table, a particular value of y and g (or base a) is searched to solving the x .

Example 18: Find x for the following:

$$3 = 2^x \bmod 5$$

Answer: The tabulation (Table 5.5) of discrete logarithm for the group $G = \langle \mathbb{Z}_5^*, \times \rangle$ is done below:

Table 5.5 discrete logarithm for the group $G = \langle \mathbb{Z}_5^*, \times \rangle$

Discrete logarithm	$y = 1$	$y = 2$	$y = 3$	$y = 4$
$x = L_2 y$	4	1	3	2

Here, $y = 3$, g (or a) = 2

Considering the above table

$$x = L_2 3 \bmod 5 = 3.$$

Some of the useful properties of discrete logarithms may be stated (Table 5.6) as:

Table 5.6 Some properties of discrete logarithms

Discrete logarithms properties
$L_g 1 \equiv 0 \pmod{\Phi(n)}$
$L_g (a \times b) \equiv (L_g a + L_g b) \pmod{\Phi(n)}$
$L_g b^k \equiv k \times L_g b \pmod{\Phi(n)}$

Two points to be considered while solving discrete logarithm problems:

- For large p , the discrete logarithm problem cannot be solved using tabulation and hence algorithms that make use of the basic properties of discrete logarithms are devised.
- The discrete logarithm problem has the same complexity as the factorization problem.

5.5 Summary

In this unit you have understood the terms and concepts used in the cryptographic domain like prime number, modular arithmetic, set of residues, additive inverse, and multiplicative inverse etc. The use and utility of prime and coprime numbers is presented in the unit. Hence forth, you should be to list the elements of $\mathbb{Z}_n^*$ & $\mathbb{Z}_p^*$ and also able to calculate number of elements in $\mathbb{Z}_n^*$ using Euler's Phi Function. To enhance your learning - application and use of Fermat's little theorem and Euler's Theorem for finding the exponentiation and multiplicative inverses is given. An understanding for finding the primitive roots and utilizing them for generating the entire set has

been developed. Towards end, discrete logarithm approach for finding inverse of exponentiation i.e., logarithm (useful in public key encryption system), is also provided.

Terminal Questions

1. *Define the following:*
 - (a) *Prime number*
 - (b) *Totient function*
 - (c) *A divides B*
 - (d) *Congruent (in modular arithmetic)*
 - (e) *Primitive root of a number*
2. *Differentiate between*
 - (a) Z and Z_n
 - (b) Z_n and Z_n^*
 - (c) *Index and discrete logarithm*
 - (d) *Euler and Fermet's theorem*
3. *Find gcd (2n+1, n)? Also find gcd (2740, 1760)?*
4. *Find additive and multiplicative inverses in modulus 20?*
5. *Use Fermat's theorem for finding:*
 - (a) $3^{201} \bmod 11$
 - (b) *a number between 0 and 72 such that it is congruent to $9794 \bmod 73$*
6. *Use Euler's theorem for finding a number between 0 and 9 such that it is congruent to $7^{1000} \bmod 10$.*
7. *Determine the following:*
 - (a) $\Phi(231)$
 - (b) $\Phi(440)$
8. *Prove the following for a prime number p*

$$\Phi(p) = p^i - p^{i-1}$$
9. *List the primitive roots of 25*
10. *Solve the following using discrete logarithm table. Assume 2 is a primitive root of 29:*
 - (a) $17 x^2 \equiv 10 \pmod{29}$
 - (b) $x^2 \equiv 17 \pmod{29}$

References:

1. Forouzan, Behrouz A., "Part 1 – Symmetric-Key Encipherment; Chapter 2: Mathematics of Cryptography", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.

2. Stallings, William, “Chapter 8: Introduction to Number Theory”, Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
3. Forouzan, Behrouz A., “Part 2 – Asymmetric-Key Encipherment; Chapter 9: Mathematics of Cryptography”, Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.

UNIT – 6

Public Key Cryptography

Structure

- 6.0 Introduction
- 6.1 Objectives
- 6.2 Principles Related with Public-key Cryptography
- 6.3 RSA – An Example Public Key Cryptosystem
- 6.4 Key Management Methods in Public-key Cryptosystem
- 6.5 Diffie-Hellman Key Exchange Algorithm
- 6.6 Summary
- 6.7 Terminal Questions

6.0 INTRODUCTION

In case when non-repudiation is primarily required symmetric key based cryptosystems (discussed in previous block) are used less. In the present day E-commerce oriented Internet world, non-repudiation is often required. In such situation, public key based cryptosystems are more useful. Hence, this unit rotates around the public key based cryptosystems targeting mainly the requirements, properties and algorithms of public key cryptosystems. It also describes RSA (Rivest-Shamir-Adleman) algorithm along with its working as an example of public key cryptosystem. In a public key based cryptosystems, the key pairs (public key and private key) are generated, shared and stored. Thus, key management tasks attain importance here. A subsection is devoted to enhance our understanding of such key management tasks. Sometimes, keys used in public key cryptosystems are shared using key exchange algorithm like Diffie-Hellman key exchange algorithm. The unit also covers this algorithm along with an example.

6.1 OBJECTIVES

This unit deals with the public key cryptosystems and the associated issues like public key based encryption/decryption algorithms, key exchange, certificate storage etc. At the end of this unit, you will be able to:

- Differentiate between symmetric key cryptosystems and public key cryptosystems.
- Understand the properties, characteristics and applications of public key based cryptosystems.

- Learn About the workings of RSA (Rivest-Shamir-Adleman) public key algorithm.
- Understand the key management and key exchange methods.

6.2 PRINCIPLES RELATED WITH PUBLIC-KEY CRYPTOGRAPHY

- The initial cryptographic tools were mainly based upon substitution and permutations.
- The public key cryptography uses mathematical principles for manipulating the numbers and revolutionized the cryptography radically.
- Public key cryptography uses two separate keys for encryption and decryption and hence is also sometimes referred to as asymmetric cryptography.
- Public key cryptography is used mainly to provide confidentially, key distribution, authentication and digital signatures.
- There are three misconceptions regarding public key cryptosystem [1]:
 - 1) It is more secure than symmetric cryptosystem. It is not so, security of any system depends upon (i) length of key, (ii) computational work required to be done for breaking a cipher.
 - 2) Public key cryptosystems have made symmetric key cryptosystems obsolete. It is not so, both of them are used equally in different segment of applications. Public key cryptography is universally used in key distribution and signature applications while symmetric cryptography is used for protecting data packets.
 - 3) Key distribution is simpler in public key cryptosystems as compared to symmetric cryptosystems. It is not so, the procedures involved in key distribution are neither simpler nor efficient in public key cryptosystems as compared to symmetric cryptosystems.

Motivation for public key cryptosystems

The following major factors contributed towards public key cryptosystem evolution.

- 1) How to share symmetric key used in symmetric encryption was a major issue. The use of Key Distribution Centre (KDC) was not very convincing as it defeats the very fundamental assumption of symmetric encryption that symmetric key is known only to participants, not to third party. In case KDC gets compromised, entire system is compromised.
- 2) With increasing electronic documents, a need to put signatures was imagined for authentication purpose. This leads to

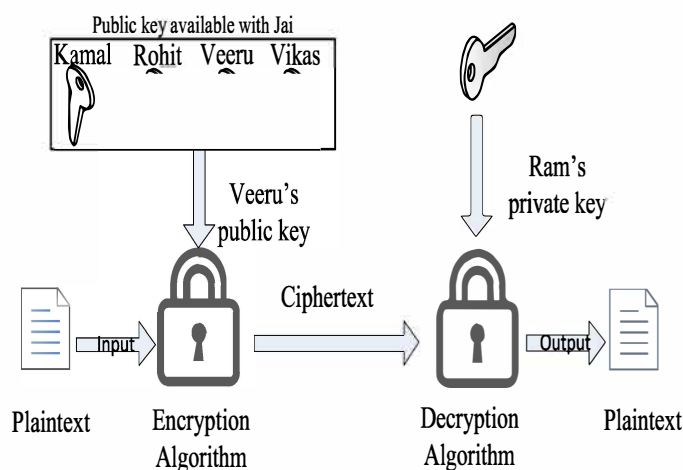
development of digital signature concept in public key cryptosystems.

Public key cryptosystems: An Introduction

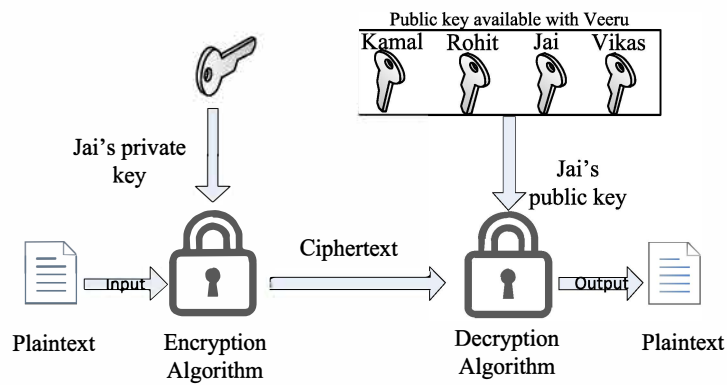
- These systems use two keys which are related to each other (usually inverse of each other). One key is used for encryption while the other is used for decryption.
- An important characteristic of these systems is that for a given cryptographic algorithm and encryption key it is computationally infeasible to obtain the decryption key.
- RSA, a popular public key algorithm (discussed later in this unit) considers that any one among the two keys can be used for encryption. The other key that is left can be used in decryption.

Any public key cryptosystem has following six elements (Figure 6.1) [2]:

- **Plaintext:** Input data or message.
- **Encryption algorithm:** The combination of various procedures that transform the plaintext into ciphertext.
- **Ciphertext:** An unintelligent message that was transformed by encryption algorithm. A ciphertext gets changed upon changing the key or message in the input to encryption algorithm.
- **Public and private keys:** Both are selected from several key pairs, one for encryption other for decryption.
- **Decryption algorithm:** Takes cipher text and decryption key for generating plaintext output.



(a) Encryption



(b) Authentication

Figure 6.1 Public-Key Cryptography

The steps involved in public key cryptography are:

1. Key pair generation by one of the user.
2. Selection of one key from the pair as public key and storing it in a publically accessible register. The other key is kept private and is not shared at all. Each user also keeps public keys of others.
3. For transferring a message securely (confidentially) to other party, first party encrypts using encryption algorithm and other party's public key.
4. The receiver decrypts using decryption algorithm and its own private key. Thus, only this receiver is able to decrypt the message encrypted previously using its public key.
5. Whenever key refreshing is required i.e., change of key is required, a new pair is generated. Private Key is kept secret while the old public key kept in public register is replaced.

A comparison between symmetric and public key cryptosystem is shown in **Table 6.1**.

Table 6.1 Comparison between symmetric and public key [2]

Conventional Encryption	Public-Key Encryption
Same key is used for encryption and decryption with same algorithm	Two different keys (in pair) are used: one for encryption and other for decryption with same algorithm
The single key needs to be kept secret	Only one of the keys (among pair) needs to be kept secret, other is known to public
The symmetric (secret) key cannot be derived via knowledge of algorithm and some samples of ciphertext	The private (secret) key cannot be derived via knowledge of algorithm, other (public) key and some samples of ciphertext
Does not guarantee non-repudiation	Does guarantee non-repudiation

- The security of the public-key cryptographic system remains intact till the private key is kept secure and safe.
- To avoid confusion of term 'secret key', the secret key in public cryptosystem is referred to as private key.
- The three cases of public key cryptosystem that ensures secrecy, authentication and both are now discussed.
- Like symmetric encryption (in Unit 1 and 4), the communicating users are named as Jai and Veeru while the eavesdropper is named as Gabber.
- The following is the nomenclature used for public key cryptosystem in this Course Material [1] [2]:

P: Public Key, K : Private Key

Key Pairs - Jai (K_J , P_J); K_J is kept secret while P_J is known to public

Key Pairs - Veeru (K_V , P_V); K_V is kept secret while P_V is known to public

Plaintext, $X = [X_1, X_2, \dots X_M]$

Ciphertext, $Y = [Y_1, Y_2, \dots Y_M]$

A public key cryptosystem may be used to provide confidentiality or authentication or both as discussed in the following cases [2]:

- I. Jai forms ciphertext as $Y = E_{P_J} (X)$

Veeru performs inverted transformation to gain the plaintext as

$$X = D_{K_V} (Y)$$

- The encryption algorithm (E) and decryption algorithm (D) are known to all users.
- An attacker may perform cryptanalysis for finding an estimate of plaintext ($\hat{X}$) or estimate of private key ($\hat{K}_V$)
- The cryptanalysis is based upon known quantities i.e., ciphertext (Y) and public key (P_V).
- If private key is cracked or estimated then all future and past correspondence may be decrypted.
- The above method is able to provide only confidentiality (shown in **Figure 6.2**) but not authentication.

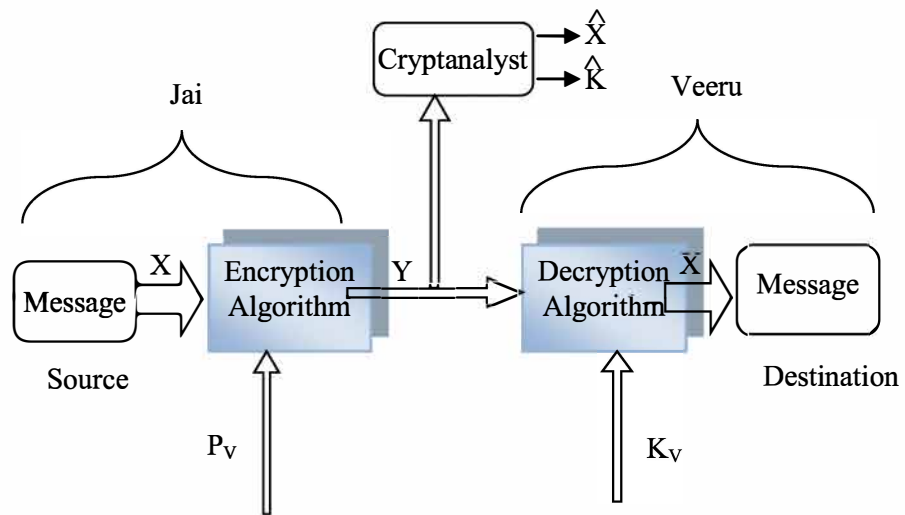


Figure 6.2 Public-Key Cryptosystem: Ensuring Secrecy

II. In case II, Jai encrypts the message X as

$$Y = E_{K_J}(X)$$

Veeru decrypts the plaintext as

$$X = D_{P_J}(Y)$$

- Here, the encrypted message of Jai serves the purpose of digital signature.
- The encrypted message ensures that modifications to plaintext are not possible without private key of Jai.
- The message is authenticated as Veeru is ensured that the message can only be encrypted by Jai who is in possession of private key.
- Though the method provides both data integrity and authentication, it has two issues:
 - (i) Both plaintext and its corresponding ciphertext need to be stored for use in case of dispute i.e., a mapping between original message and its corresponding ciphertext is required.
 - (ii) Sometimes, authentication of a document without its encryption is required.
- In both the issues above, an authenticator that is attached to document is required.

- Confidentiality is not ensured in this case (II) (shown in **Figure 6.3**) as any user can decrypt the message by utilizing the public key of sender.

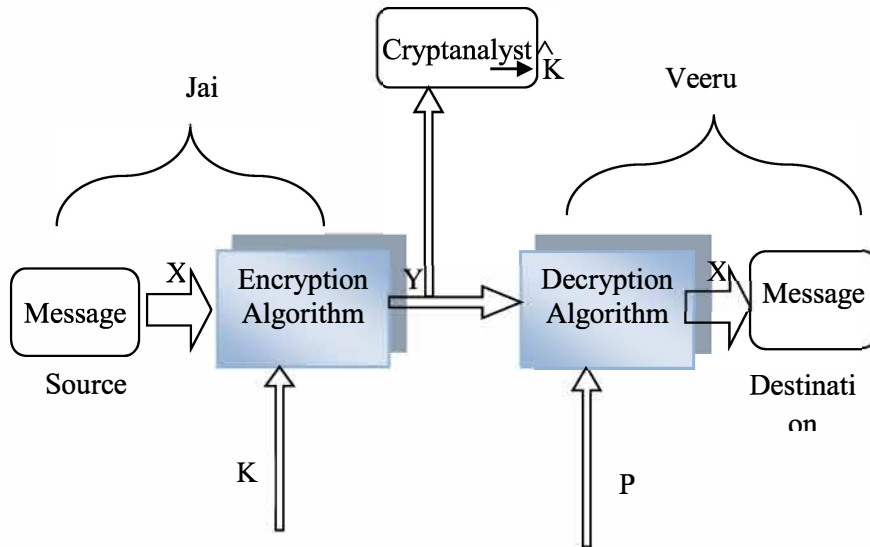


Figure 6.3 Public-Key Cryptosystem: Ensuring Authentication

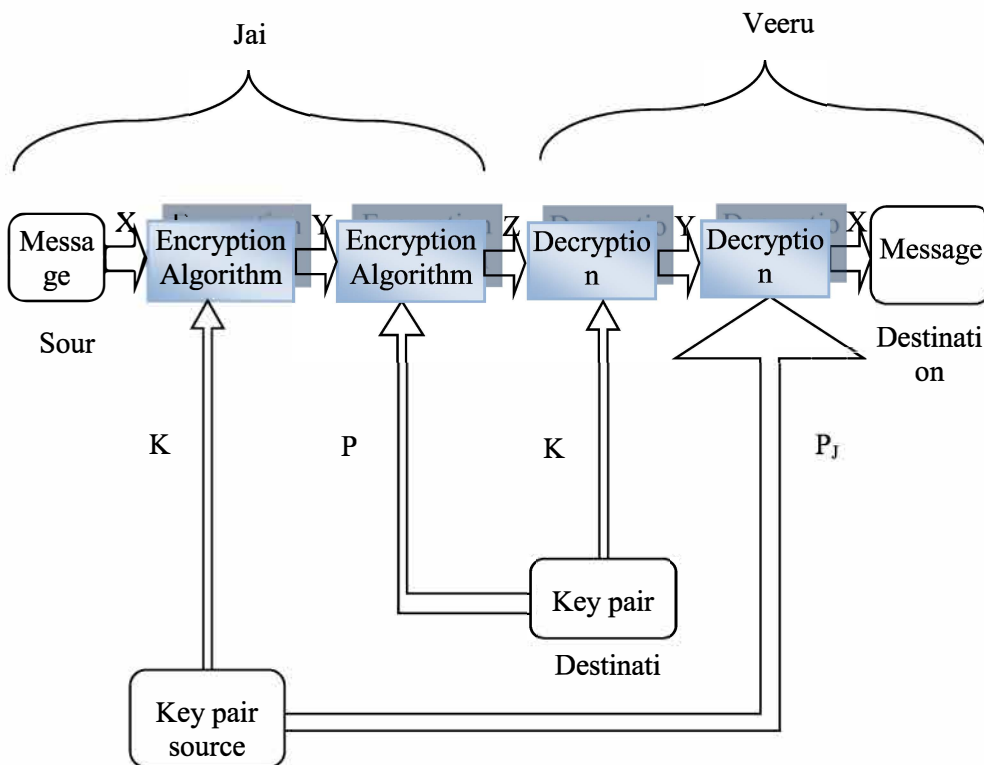


Figure 6.4 Public-Key Cryptosystem: Ensuring both Authentication and Secrecy

III. In this case (**Figure 6.4**), Jai and Veeru both executes encryption algorithm/function twice.

- Jai performs encryption as:

$$Z = E_{P_v}(E_{K_j}(X))$$

Veeru performs decryption as:

$$X = D_{P_j}(D_{K_v}(Z))$$

- The encryption done by Jai using its own private key (K_j) ensures authentication and again using public key of Veeru ensures that only Veeru will be able to retrieve it back i.e., confidentiality is also provided.
- The disadvantage of the above method is that public key algorithm is executed four time (2 times by Jai and 2 times by Veeru)

Applications

Applications (use) for public key cryptosystem falls under following three categories.

- Encryption/Decryption – Use of public key encryption for providing confidentiality. Encryption is done using receiver's public key while decryption is done using receiver's private key.
- Digital Signature – Use of public key encryption for providing message and user authentication. Usually a small sized data is derived as a function of message and then it is encrypted using sender's private key so as to represent signature.
- Key Exchange – Use of public key encryption for exchanging and deriving session keys. This may involve private keys of both parties (Diffie-Hellman method).

Table 6.2 – lists the algorithms of public key cryptography that uses all three, two or single of the above mentioned functionalities.

Table 6.2 Public-Key Cryptosystem – uses

Algorithm	Uses
RSA	Used for Encryption/Decryption, Digital Signature and Key Exchange
Diffie-Hellman	Used for Key Exchange
DSS	Used for Signature

Requirements

Public key cryptography utilizes two related keys. The public key cryptographic algorithms must fulfill the following conditions:-

- 1) It is computationally easy
 - (i) by a party (say Veeru) to generate a key pair (K_v , P_v)
 - (ii) to generate ciphertext (C) from plaintext (P) by other party (say Jai) using public key of first party as
$$C = E_{P_v}(M)$$
 - (iii) To decrypt the ciphertext to get back the plaintext as
$$M = D_{K_v}(C) = D_{K_v}(E_{P_v}(M))$$
- 2) The private and public keys can be applied in any order as
$$M = D_{K_v}(E_{P_v}(M)) = D_{P_v}(E_{K_v}(M))$$
- 3) It is computationally infeasible for an attacker (say Gabbar) to find (i) the private key (K_v), (ii) the plaintext (M) using ciphertext (C) and public key (P_v)
 - The above requirement reduces itself to the need for a trap-door one-way function.
 - A one way function is one that itself is easy to calculate but finding its inverse is infeasible.
i.e. $Y = f(x)$ is easy, while $X = f^{-1}(y)$ is infeasible.
where, X is domain of function and Y is its range.
 - A function is easy to calculate means computations can be done in polynomial time i.e., n^a , where n is input of n bits and a is a constant.
Here, infeasible to calculate means it requires time much larger than polynomial time e.g. of the order of 2^n .
 - A trap door one way function is one for which finding inverse is feasible (may be calculated in polynomial time) provided some additional information is known i.e., for a trapdoor K , calculating inverse
$$X = f_K^{-1}(y)$$
 is easy.
 - The traditional notion of best case/worst case complexity is not followed in case of cryptography. In cryptography infeasible to compute means it is infeasible for all input irrespective of best or worst case input.

Check your progress 1

- a. Why is Public key cryptography useful?
- b. What are major advantages of a using Public key encryption method in a system?
- c. In public key encryption system what steps does sender A takes?

6.3 RSA – AN EXAMPLE PUBLIC-KEY CRYPTOSYSTEM

Diffie and Hellman's introduction of Public key cryptography in 1976 paved the path for development of one of the most widely accepted and general purpose public key encryption method (in 1977) at MIT by Ron Rivest, Adi Shamir, and Len Adleman. This method is popularly known as Rivest-Shamir-Adleman (RSA) scheme. It is a block cipher in which the block size is taken as 1024 bits [2].

Description

The RSA algorithm has three components: key generation, encryption and decryption. Each of these is described below:

Key generation

1. Select two large prime numbers p and q ($p \neq q$)
2. Calculate $n = p \times q$
3. Calculate $\phi(n) = (p - 1)(q - 1)$
4. Select an integer e ($\leq \phi(n)$) such that it is relatively prime to $\phi(n)$ (Euler totient function) i.e., $\gcd(\phi(n), e) = 1$
5. Calculate multiplicative inverse of e (name it as d) as:

$$d \equiv e^{-1} \pmod{\phi(n)}$$

$\{e, n\}$ becomes the public key while $\{d, n\}$ becomes the private key

Encryption

Consider plaintext as M

Then ciphertext (C) is evaluated using public key (e, n) and exponentiation as:

$$C = M^e \pmod{n}$$

Decryption

Plaintext M is gained using private key (d, n) as:

$$M = C^d \pmod{n}$$

Figure 6.5 shows the complexity of encryption and decryption in RSA algorithm whereas **Figure 6.6** shows the encryption, decryption and key generation process [3].

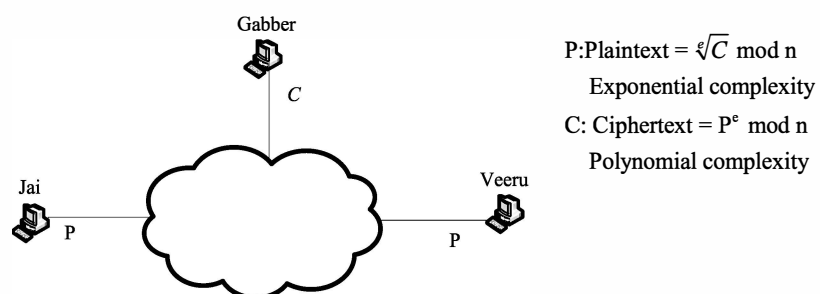


Figure 6.5 RSA complexity of encryption and decryption

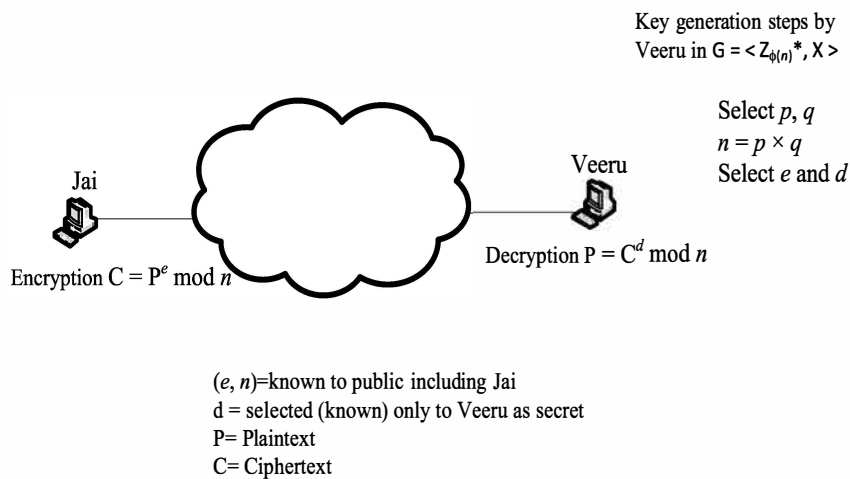


Figure 6.6 RSA: encryption, decryption and key generation

Example 1:

Apply RSA algorithm for encrypting message $m=10$ [1]

Consider primes $p=11$, $q=3$.

Answer:

$$n = p * q = 11 * 3 = 33$$

$$z = (p-1)(q-1) = 10 * 2 = 20$$

Choose $d=7$

Compute ' e ' such that $7 * e = 1 \bmod 20$

$$\Rightarrow e = 3$$

Public key = $(n, e) = (33, 3)$

Private key = $(n, d) = (33, 7)$.

Encryption of the message $m = 10$

$$c = m^e \bmod n = 10^3 \bmod 33 = 1000 \bmod 33$$

cipher text $c = 10$

Decryption

$$m' = c^d \bmod n = 10^7 \bmod 33 = 10000000 \bmod 33 = 10$$

(plaintext)

Figure 6.7 shows another example (self explanatory) where the sender wants to send message NO. For this purpose he maps alphabets to numbers 00 to 25 [3].

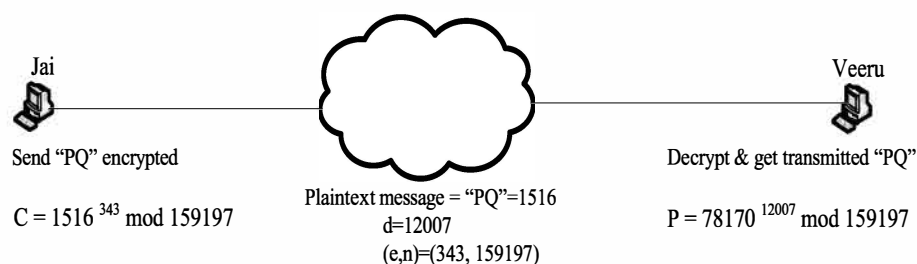


Figure 6.7 Example of RSA

Proof:

$$C = M^e \bmod n$$

Message can be obtained back as:

$$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n \quad (\because ed \bmod \phi(n) = 1)$$

- Following are to be remembered in RSA

p, q are privately chosen by a party

n is known to public

e is also known to public

d is privately calculated.

- Following requirements must be met in RSA
 - (i) n, e and d can be calculated such that $M^{ed} \bmod n = M$
 - (ii) Exponential calculations like $M^e \bmod n$ and $C^d \bmod n$ are easy to perform.
 - (iii) For a given c and n, d is infeasible to calculate.
- For calculating exponentiation in modular arithmetic the following property is used:

$$[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$$

e.g. calculating x^{16} means 15 multiplications are required.

This can also be attained by using 4 multiplications by squaring the partial results.

$$\text{Thus, } x^{16} = (((x^2)^2)^2)^2$$

$$\text{Similarly } x^{11} = x^{1+2+8} = (x)(x^2)(x^8)$$

$$x^{11} \bmod n = (x) \times (x^2) \times (x^8) \bmod n$$

$$= [(x \bmod n) \times (x^2 \bmod n) \times (x^8 \bmod n)] \bmod n$$

using above property

The above method is known as fast modular exponentiation algorithm

- The new key generation in RSA deals with selecting two large primes p and q .
- There is no deterministic technique that finds such primes. Instead there are probabilistic algorithms like Miller-Rabin algorithm that identifies such prime numbers after several trials.
- Extended Euclid's algorithm is used to find e such that $\gcd(\phi(n), e) = 1$, i.e. finding e relatively prime to $\phi(n)$.

Security

RSA may be targeted using any of the following four methods [2]:

1. **Brute force** – Under this category, attacker tries all possible combinations for private keys. Thus, RSA must use large key space i.e. increase key size. In RSA, a complex key generation method is used. Also encryption/decryption depends upon key size. Thus, upon increasing key size, the system will become slow i.e., processing with large key will consume extra time.
2. **Mathematical attacks** – There are following three mathematical approaches to counter RSA:
 - (i) If n factors i.e., p and q are known, then $\phi(n) = (p - 1)(q - 1)$ is known. This enables calculating the private key as $d = e^{-1} \pmod{\phi(n)}$
 - (ii) Determine $\phi(n)$ directly without finding factors, p and q
 - (iii) Determine d directly without finding totient function i.e., $\phi(n)$
 - Among these, the attacker primarily targeted factoring the product of two primes.
 - In 1994, a group claimed the prize for breaking a challenge initiated in 1977 by RSA inventors. The public key size in this challenge was 428 bits. The attempt shows that RSA can be broken by finding factors.
 - The growing computing power of systems and improved factoring algorithms (quadratic sieve, generalized number field sieve, special number field sieve) indicate and force to use large key size. In India, currently RSA variants with 1024 and 2048 bits is used.
3. **Timing attacks** – In these attacks, attacker observes the time taken by computer to decrypt the messages. The observations are then used to determine the private key [4].
 - These are ciphertext only attacks.
 - The attacker starts from left - bit by bit. If the particular iteration for decrypting a single bit is slow, the bit is assumed to be 1.
 - If the particular iteration for decrypting a single bit is fast, the bit is assumed to be 0.
 - Thus, variation in execution time is utilized for breaking RSA.
 - The following countermeasures are utilized to dampen the effect of timing attacks.
 - (i) Ensuring that all exponentiation steps take equal amount of time (constant exponentiation time method). This method degrades the performance.
 - (ii) Addition of random delay for confusing the attacker. This is not much effective as the attacker can take additional measures to compensate the added random delays.
 - (iii) Blinding – This method is used by RSA data security and in this method ciphertext is multiplied with a random number

before going for exponentiation. Blinding adds to computation and therefore has associated performance issues.

RSA data security considers finding method as follows:

a) A secret random number (r) is generated such that $0 < r < n-1$

b) Multiply ciphertext by r (Blinding) as

$$C' = C (r^e) \bmod n \quad \text{where, } e \text{ is public exponent}$$

c) Decrypt ciphertext as

$$M' = (C')^d \bmod n$$

d) Calculate plaintext as

$$M = M' r^{-1} \bmod n$$

Where r^{-1} is multiplicative inverse of $r \bmod n$

4. Chosen Ciphertext Attack (CCA)

Let us first consider an example of CCA

Assume, Ciphertext $C = M^e \bmod n$

Calculate $X = (C \times 2^e) \bmod n$

This X is now submitted as CCA to the receiver

The receiver decrypts as

$$Y = X^d \bmod n$$

$$= (C \times 2^e)^d \bmod n$$

$$= (M^e \times 2^e)^d \bmod n$$

$$= (2M)^{ed} \bmod n = (2M) \bmod n$$

$Y = 2M \bmod n$, hence M can easily be calculated.

Thus in CCA, attacker selects ciphertext and then submit it for decryption. Decryption yields the desired data for cryptanalysis.

For protecting against such attacks RSA security Inc. suggests use of padding on plaintext by using OAEP (Optimal Asymmetric Encryption Padding) method.

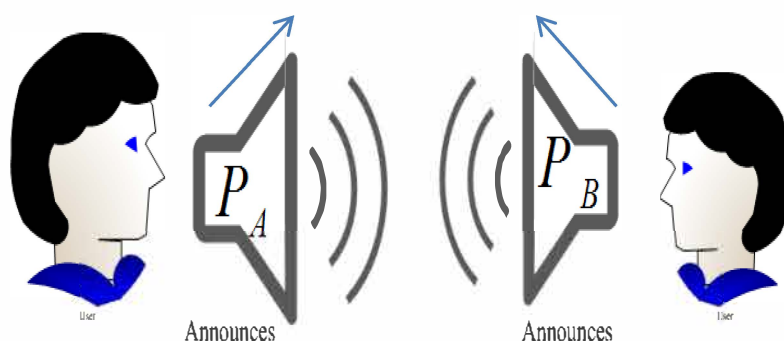


Figure 6.8 Public announcements of public keys

6.4 KEY MANAGEMENT METHODS IN PUBLIC-KEY CRYPTOSYSTEMS

Public key cryptography is used to distribute public keys and also to distribute secret symmetric key using public key encryption.

1. Public keys distribution approaches [5]

Major categories into which public key distributing schemes can be grouped are

(i) Announcement of public keys to public

In this category (**Figure 6.8**) the public keys are shared between different users using announcements or broadcasting. Pretty Good Privacy (PGP) that enhances security in E-mail applications by using RSA, appends user's public keys to messages. These messages are sent to public forums, news groups and mailing lists. The method is simple and convenient but is prone to forgery issues where a person may act on behalf of sender and send public key to participants. The participants start encrypted communication with this forged user. Later, actual sender may identify the fraud and try to alert others but till then the damage may have occurred.

(ii) **Use of Public Directory-** A third party maintains a public directory that keeps public keys of all users (**Figure 6.9**). This organization keeps entries of the form {name, public key} for each user. For keeping entry and for updating later, each user registers either in person or through secure authenticated communication. The entry may be changed dynamically later or upon getting the key compromised. For updation, authentic communication between user and public directory is required. In case the private key of the directory gets compromised, the attacker may impersonate any user, eavesdrop on messages and may even modify the directory records.

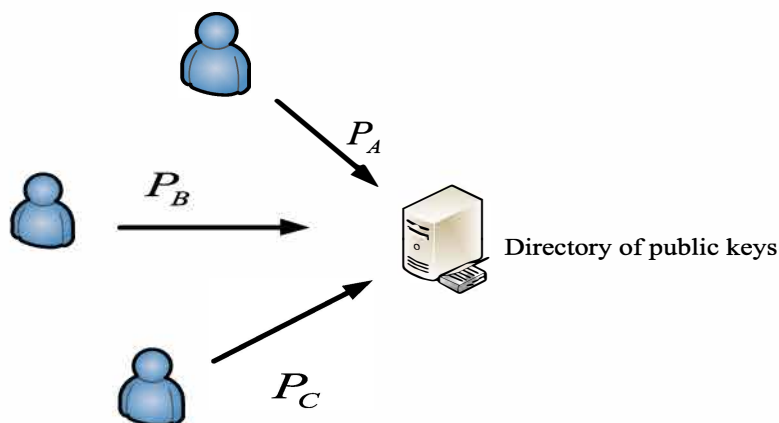


Figure 6.9 Public Directory for public keys

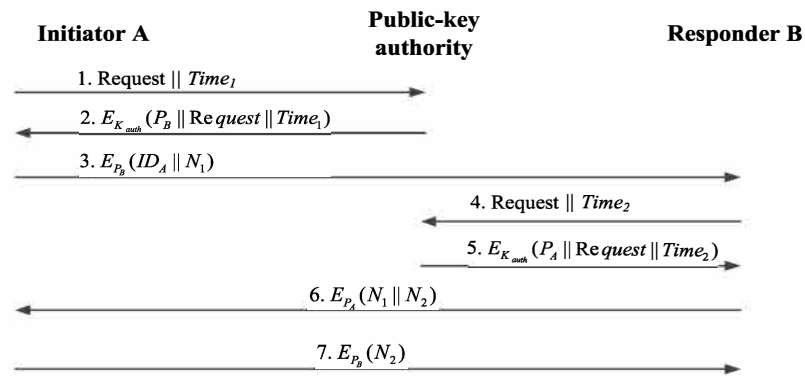


Figure 6.10 A Distribution Scenario for public keys

(iii) **Use of Public-key Authority-** It is similar to the Public Directory. In addition the authority keeps its private key with itself and dismounts its public key in public i.e., to all users. In the scenario shown in **Figure 6.10**, A is Initiator and it wants to communicate with B who is responder. The steps required to establish communication between A and B using Public-key authority are as:

- A sends request regarding public key of B to the authority. The request contains current time stamp.
- Authority's reply contains public key of B (P_B) and original request of A. The reply is encrypted using private key (K_{Auth}) of authority. Encryption is done so that A is assured that the reply is given by authority only. Time stamp ensures that this reply is not an old one.
- A extracts public key of B (P_B) and uses it to encrypt message for B. Message for B contains ID (ID_A) and nonce (N₁). Nonce identifies a particular message from A to B.
- As done in (a) and (b) above by A for getting public key of B, messages (4) and (5) are used to get public key of A from authority by B. After this step both A and B have public keys of each other.
- B now replies A. Reply contains Nonce (N₁) extracted in step (c) above and a new nonce of B (N₂). The message is encrypted using public key of A.
- Lastly, A acknowledges B by sending back Nonce (N₂). This ensures B that it is communicating with A only. Thus, a total of 7 messages are exchanged for sharing public keys of each other using public-

key authority. A and B can also cache the public keys. In this case first 4 steps (a - d) are not required.

- (iv) **Use of certificates-** The previous two schemes have drawbacks. The public directory is vulnerable to tampering while public-key authority being central to the system must be able to handle large user requests for public keys. It may become bottleneck as whenever a communication is required it is contacted for user's public key.

The concept of public certificates issued by central authority proves useful under such conditions. A public key is extracted from certificate whenever communication is required. The authority need not be contacted every time for public key. This reduces load on authority. Also, as the certificate is signed by authority it is reliable and can be trusted. The certificate contains name & identifier of owner of public key, time of creation of certificate and public key itself. The certificate authority is a trusted third party (usually government organization). A user submits its public key through secure communication medium to authority, authority in turn creates and signs corresponding certificate. This certificate is published in a public repository or database. Anybody can obtain this certificate and verify the contents and authority's signature using public key of the authority.

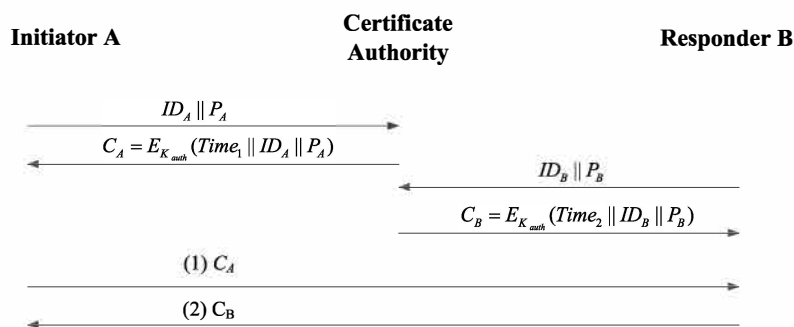


Figure 6.11 A Scenario for exchanging public key certificates

The following are the requirements of public key certificate scheme:

- (1) The certificates are created, maintained and signed only by the authority.
- (2) Any user can verify the contents of the certificate and signature of the authority.
- (3) After verification, a user is able to retrieve the details of the ID, name and public key contained within the certificate.

- (4) Using time stamp in certificate, it can be identified whether the certificate is current or old.

The entire process is shown in **Figure 6.11**. The scheme has following steps:

- (i) Each participating user submits the request for certificate to the authority. The request contains identifier and public key.
- (ii) The certificate authority issues a certificate as

$$C_A = E_{K_{Auth}} (T || ID_A || P_A)$$

Here, T is the Timestamp of generation of certificate

K_{Auth} is the private key of certificate issuing authority

- (iii) This certificate is given to the other corresponding communicating user. The user may verify the certificate and extract public key of user A as

$$D_{P_{Auth}} (C_A) = D_{P_{Auth}} (E_{K_{Auth}} (T || ID_A || P_A)) = T || ID_A || P_A$$

Thus, T ensures that the certificate is current and prevents replays of old certificates. The certificates with old timestamps are considered as expired. P_A is the desired public key of user with ID_A .

- As the certificate is decrypted using public key of authority, it is ensured that the certificate was infact generated by authority.
- X.509 is the standard for format of certificates and is used by applications like, Secure Socket Layer (SSL), Secure Electronic Transactions (SET) etc.

2. Utilization of public key cryptography for distributing symmetric secret keys [5]

- Although public key cryptography provides secure mechanism but it is not preferred exclusively or completely for communication. This is due to the fact that public key cryptography provides slow data rates as compared to symmetric key cryptography.

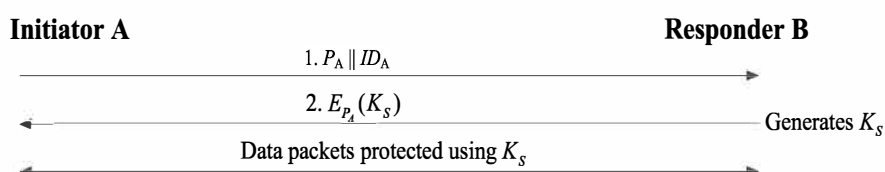


Figure 6.12 A simple Session Key establishment

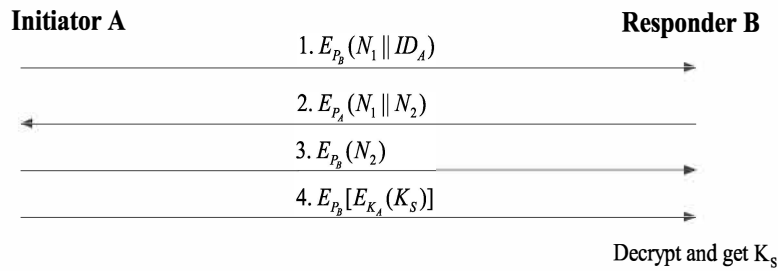


Figure 6.13 Distributing Secret Keys Using Public Key Method

First, let us discuss a simple key distribution mechanism shown in **Figure 6.12**. The communication is done between A and B using following steps.

- (i) A generates public key pair, keeps private key with itself and sends ID_A (identifier A) and P_A (public key of A) to B.
- (ii) B generates secret key for session (K_S), encrypts using A's public key and sends it back to A as

$$E_{P_A}(K_S)$$

- (iii) A decrypts the message as

$$D_{K_A}(E_{P_A}(K_S))$$

and obtains secret session key (K_S). Now, both A and B have shared and similar secret keys.

- (iv) A and B starts communicating using K_S while discarding the public keys.

- This simple mechanism of sharing symmetric encryption key is prone to man-in-the-middle attack.
- In this attack, the attacker (E) lies in between and intercepts the message send by A to B. Attacker removes the B's public key and adds its public key (P_E) and sends message containing $P_E || ID_A$ to B.
- B transmits encrypted key K_S as

$$E_{P_E}(K_S)$$

This can easily be decrypted by attacker as

$$D_{K_E}(E_{P_E}(K_S))$$

- The attacker than reconstructs the message to A as

$$E_{P_A}(K_S)$$

Thus, to A it seems that nothing wrong has occurred but as shown above the attacker is able to extract the symmetric key in between.

A second method that shares the symmetric encryption key and maintains confidentiality and authentication is proposed by Needham and Schroeder [6] (**Figure 6.13**). In this method, both the parties share random numbers (nonce N_1 and N_2). The following steps are involved in this method.

- (i) A sends ID_A and a nonce (N_1) encrypted using B's public key.
- (ii) B extracts N_1 and then reply by keeping both N_1 and a new nonce (N_2) into reply. The reply is encrypted using A's public key. The receipt of N_1 at A ensures that B has correctly received previous message and A is infact communicating with B.
- (iii) A decrypts and extracts N_2 . This N_2 is encrypted with public key of B and then send to B. This is to confirm B that it is infact communicating with A.
- (iv) A selects secret encryption key K_S and encrypts it twice: using public key of B (P_B) to ensure only B can retrieve it and using its own private key (K_A) as

$$C = E_{P_B}(E_{K_A}(K_S))$$

- (v) B decrypts the ciphertext message to gain the secret key.

A hybrid approach that uses Key Distribution Centre (KDC) for distributing secret key is also used. In this scheme, KDC shares secret master key with each of its users. KDC evolves secret session keys and send them encrypted with master keys to the users. The session keys are distributed frequently. Use of public key encryption/decryption for their distribution could cause computation issues. Hence, public key encryption is limited to update master key that is not changed so frequently. As this scheme uses KDC, it should be backward compatible with existing KDC mechanism and involve minimal changes.

Check your progress 2

- a. What is the use of Euler totient function in RSA algorithm?
- b. Why does the method of encrypting each message transmission by a different random key and sending the key to the receiver using his public key useful?
- c. How can DES and public key algorithm be combined for better results?

6.5 DIFFIE-HELLMAN KEY EXCHANGE ALGORITHM

- Diffie-Hellman is a popular key exchange algorithm. It enables two communicating parties to evolve a shared secret key based upon exchanging some value over insecure medium. The shared secret key is further used to encrypt the messages between these two parties.

- The effectiveness of Diffie-Hellman method depends upon difficulty in computing discrete logarithms.
- Diffie-Hellman method has some global publically known numbers considered by both the communicating parties. These are:

- (i) A prime number q
- (ii) A primitive root α of q

- Whenever, two users want to exchange a key using Diffie-Hellman method, they select their own secret numbers. Consider the two users as Jai and Veeru. Jai selects his secret number as X_J ($X_J < q$) and Veeru selects his secret number as X_V ($X_V < q$).

- Both the users calculate their public numbers as

$$\text{Jai: } Y_J = \alpha^{X_J} \bmod q$$

$$\text{Veeru: } Y_V = \alpha^{X_V} \bmod q$$

- These public numbers are exchanged over insecure medium. On receiving these public numbers each side calculates the shared common secret key as

$$\text{Jai: } K = (Y_V)^{X_J} \bmod q$$

$$\text{Veeru: } K = (Y_J)^{X_V} \bmod q$$

- Thus both users (Here, Jai and Veeru) are able to evolve same key as

$$\text{Jai: } K = (Y_V)^{X_J} \bmod q$$

Veeru:

$$K = (Y_J)^{X_V} \bmod q$$

$$= (\alpha^{X_V} \bmod q)^{X_J} \bmod q$$

$$= (\alpha^{X_J} \bmod q)^{X_V} \bmod q$$

$$= \alpha^{X_V X_J} \bmod q$$

$$= \alpha^{X_J X_V} \bmod q$$

- The entire process is shown in **Figure 6.14 [4] [7]**.
- As both Jai and Veeru keeps X_J and X_V secret, an attacker has only public numbers i.e., q , α , Y_J and Y_V to work with.

Thus, for cracking the Diffie-Hellman method, the attacker must determine the discrete logarithms. For large primes, calculating discrete logarithms is infeasible.

- An example as shown in **Figure 6.15 [1]**:

- Diffie-Hellman method is prone to Man-in-the-middle (MitM) attack as shown in **Figure 6.16** [1] [7].

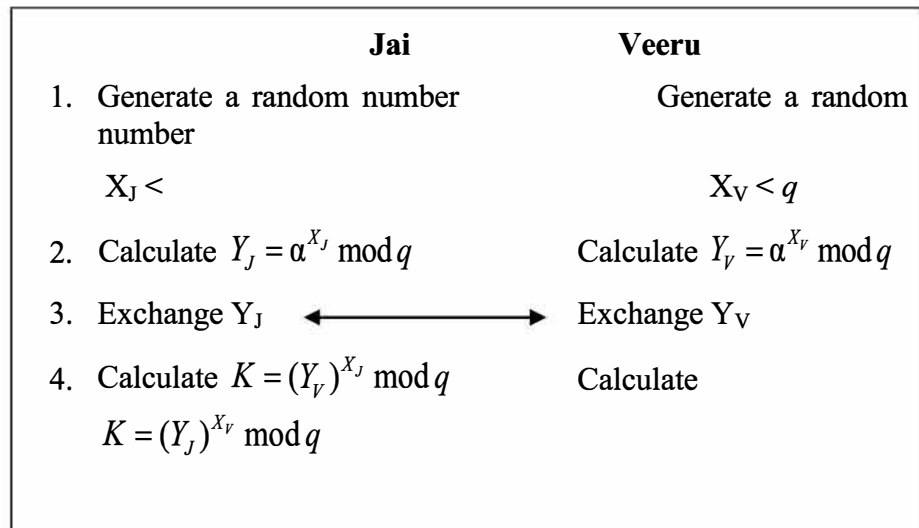


Figure 6.14 Diffie-Hellman key exchange

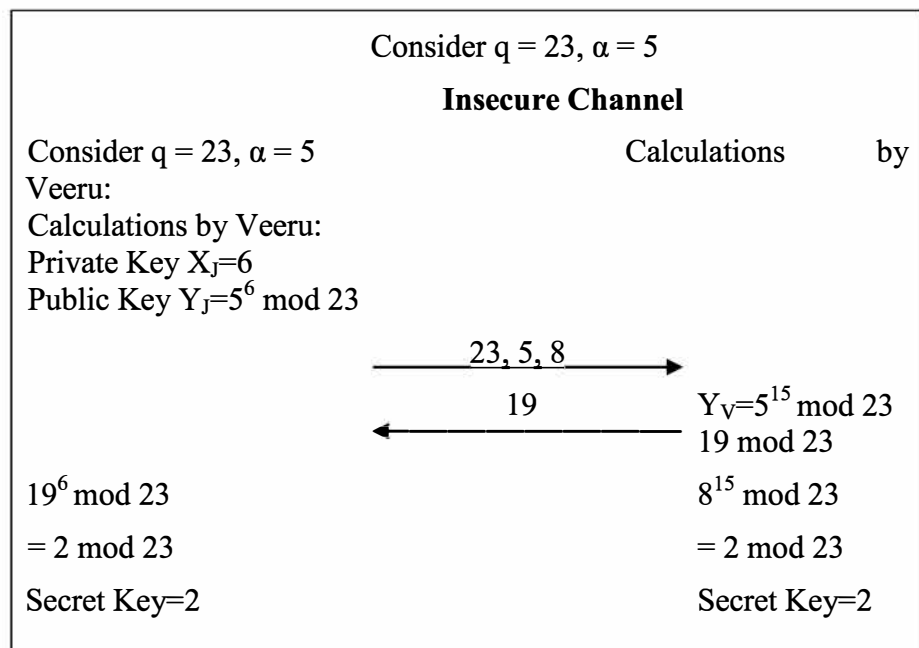


Figure 6.15 Diffie-Hellman Example

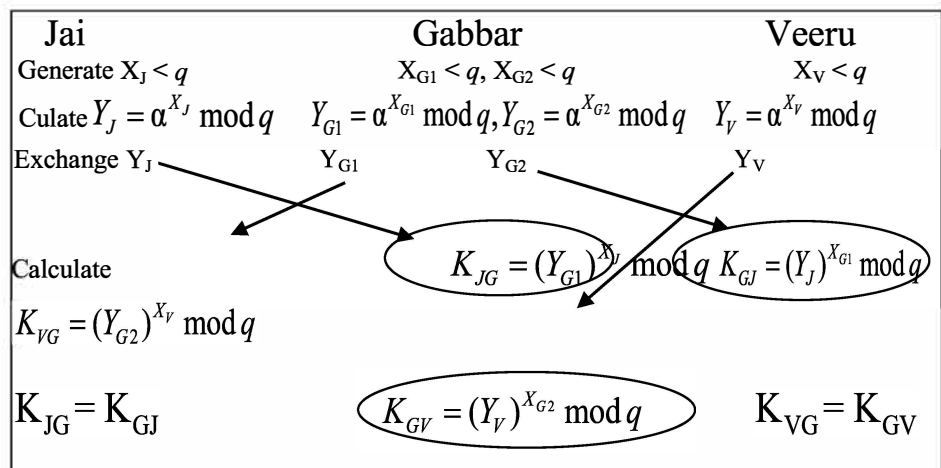


Figure 6.16 Effect of Man-in-the-middle attack

- It can be easily seen that $K_{JG} = K_{GJ}$ and $K_{GV} = K_{VG}$. Hence, whatever message (M) is sent encrypted by Jai (using K_{JG}) is intercepted and decrypted by Gabbar (using K_{GJ}). Gabbar reads the message and again encrypts it (using K_{GV}) before sending it to Veeru. Upon receipt, Veeru decrypts the message M (using K_{VG}).
- Thus, the middle man, Gabbar is able to read the encrypted message without letting this fact known to Jai or Veeru.
- The Diffie-Hellman key exchange protocol is prone to MitM attacks because it lacks user authentication. This can be overcome by using digital signatures and public-key certificates.

6.6 SUMMARY

In this unit, one of the most secure techniques used in present day information security i.e., public key cryptography is discussed. The requirements of public key cryptosystems are elaborated. One way functions plays an important role in such public key cryptosystems. How to derive the one way functionality for enhancing security using keys is realized. One of the most popular and an old algorithm in the public key cryptosystem domain i.e., RSA algorithm is described. Few examples of RSA are worked upon that aids in your understanding of the algorithm. The keys used in any public key cryptosystems are shared using key exchange algorithm. Diffie-Hellman is one such important algorithm used for key exchange between any two users. As it is prone to man-in-the-middle attack, it cannot be used to provide secrecy of messages.

Terminal Questions

1. *When is public key use required? How is it difference from symmetric key?*
2. *What are the requirements of public key cryptosystems? Explain from view point of security.*
3. *Why is one way function required? What are its important characteristics?*
4. *Discuss use of trapdoor function.*
5. *Apply the RSA algorithm for the following:*
 - (a) $p = 5; q = 11, e = 3; M = 9$
 - (b) $p = 11; q = 13, e = 11; M = 7$
6. *For the intercepted ciphertext $C = 10$ and public key $e = 5$ and $n = 35$, identify the plaintext M.*
7. *For public key RSA system having values $e = 31, n = 3599$, identify the private key.*
8. *Assess the following situation for safety - a person leaks his private key and decides to generate a new public key and new private key using same modulus as before.*

9. *How are public keys distributed? Discuss any 2 schemes.*
10. *What is the importance of Diffie-Hellman key exchange algorithm? Why is it prone to man-in-the-middle attack?*
11. *List and explain the components of a public key certificate.*
12. *Why is session key evolved? Differentiate between master key and session key.*

References:

1. Overview of cryptography, PKI Outreach Programme (POP) Nationwide Awareness Programme, Centre for Development of Advanced Computing (C-DAC), Electronics City, Bangalore, 2012.
2. Stallings, William, “Chapter 9: Public Key Cryptography and RSA”, Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
3. Forouzan, Behrouz A., “Chapter 10: Asymmetric-Key Cryptography”, Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
4. Kocher, P. “Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems.” Proceedings, Crypto '96, August 1996.
5. Stallings, William, “Chapter 10: Key Management; Other Public-Key Cryptosystems”, Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
6. Needham, R., and Schroeder, M. “Using Encryption for Authentication in Large Networks of Computers.” Communications of the ACM, December 1978.
7. Forouzan, Behrouz A., “Chapter 15: Key Management”, Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.

UNIT – 7

Message Authentication and HASH Functions

Structure

- 7.0 Introduction
- 7.1 Objectives
- 7.2 Requirements of Authentication
- 7.3 Authentication Functionality
- 7.4 Message Authentication codes (MACs)
- 7.5 Hash Functions: SHA-1, MD5
- 7.6 Secure Hash Algorithm (SHA)
- 7.7 Summary
- 7.8 Terminal Questions

7.0 INTRODUCTION

Authentication is an important measure in information and network security. It helps in answering questions like – Who own the message? Who generated the message? Who is communicating? Identity of sender? Is the message received only by the concern party? etc. In this unit, the authentication requirements are identified and then authentication functions are described. Traditionally, authentication is implicitly provided by encrypting the message. Two other prominent authentication methods exist. One depends upon secret key while other is independent of secret key. The former is termed as Message Authentication code (MAC) while latter is termed as Hash. Both of these methods are described here along with an example algorithm.

7.1 OBJECTIVES

This unit is devoted to learning authentication functions and finding hash of message either with or without using key - from security perspective. At the end of this unit, you will be able to:

- know about the various authentication requirements and functionalities.
- Understand how to calculate and work with hash for authentication of messages.
- Know the concept Message Authentication code (MAC), its properties and features.
- Learn about the structure of a secure hash algorithm.

7.2 REQUIREMENTS OF AUTHENTICATION

The following are the authentication requirements imposed during communication of message(s) between sender and receiver.

- (i) The received message should be sent only by the alleged sender i.e., message and the acknowledgements from fraudulent source are never accepted.
- (ii) The message has not been altered on the way i.e., insertion, deletion, transformation and modification of message contents is not possible.
- (iii) The sequencing of messages or its parts is maintained i.e., it is not possible for the attacker to change the sequencing.
- (iv) The delayed or replayed messages must not be accepted i.e., modification in timings of message is not possible. Thus, a message cannot be accepted beyond a particular time limit.
- (v) Non repudiation of source (sender) is ensured i.e., a source may never deny sending of message.
- (vi) Non repudiation of destination (receiver) is ensured i.e., destination cannot deny receipt of message.

7.3 AUTHENTICATION FUNCTIONALITY

Authentication of message involves two steps:

- (i) Creation of a value (termed as authenticator) that is used to authenticate the message.
- (ii) Verification of authenticator by receiver.

Ways of creating an authenticator:

- (i) The encrypted ciphertext of the message may serve the purpose of an authenticator.
- (ii) A fixed length value evaluated after applying a function that takes message and shared secret key as input. The fixed length value is termed as message authentication code (MAC) and may serve the purpose of an authenticator.
- (iii) A fixed length value that is evaluated as a function of message only. This value is termed as hash and may serve the purpose of an authenticator.

In the subsection, each of the above authenticators are discussed.

Use of Message Encryption for Authentication:

Both symmetric and asymmetric encryption may be used independently.

Application of Symmetric Encryption:

Symmetric encryption provides confidentiality and authentication. **Figure 7.1 (a)** shows the usage of symmetric encryption to ensure both confidentiality and authentication. Authentication is implicitly ensured along with confidentiality by encrypting selected message

using shared encryption key while only the receiver is able to decrypt back the data. As no other attacker knows the shared key, the alteration to the message is also not possible. The only issue involved in such kind of authentication is to identify the legitimacy of the message. To elaborate this point consider communication between sender A, receiver B, encryption function E, decryption function D and secret key K. The receiver B may accept any input X and decrypt it as $Y = D_K(X)$. It might happen that the decrypted Y is meaningless sequence of bits (in case X is provided by attacker). Thus, receiver needs to identify whether this Y has some meaning or not. Upon identifying that Y has some meaning the corresponding plaintext X is treated as legitimate and the receiver gets ensured that it must be send by the sender A only (**Figure 7.2**).

Out of all possible bit combinations only small subset may be considered as legitimate e.g., out of all possible combination only few qualify as per ordinary English language. Thus, the problem of determining that a given cipher text decrypts to intelligible plaintext is difficult. The difficulty level get raise if the plaintext considered for encryption is in binary format/file.

This provides ample opportunities to the attacker for disrupting or interfering decryption process at receiver by sending forged random messages that appears as encrypted messages from legitimate users.

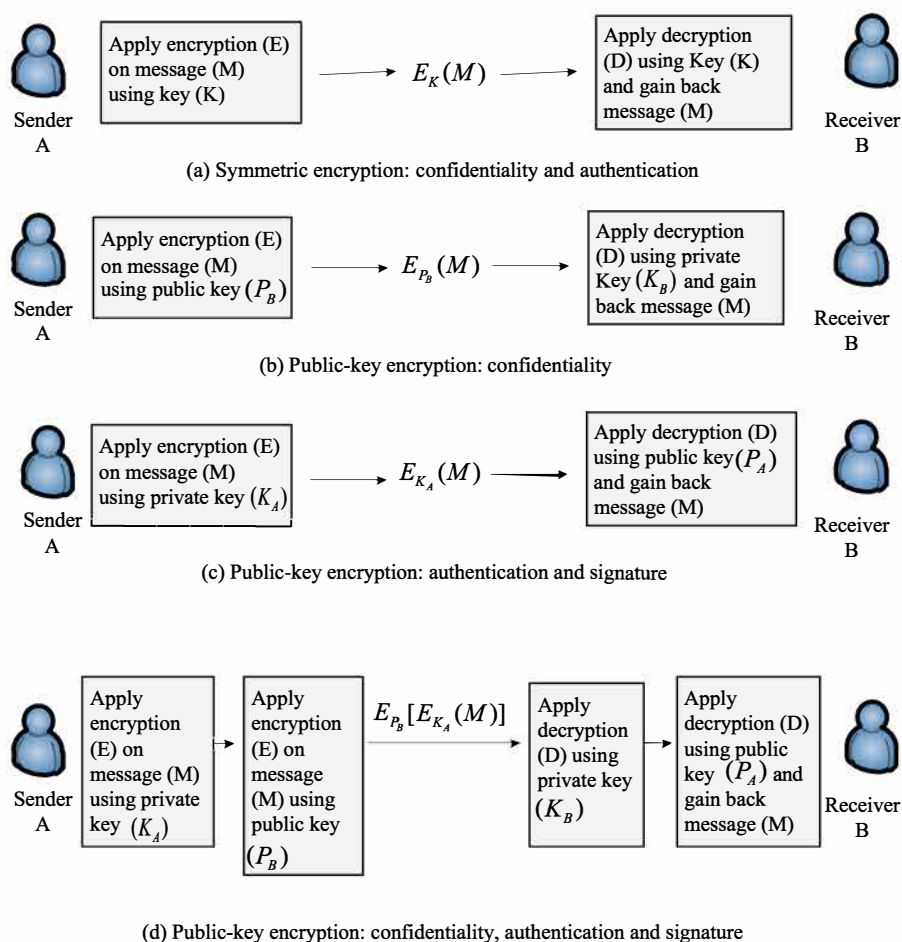


Figure 7.1 Uses of message encryption

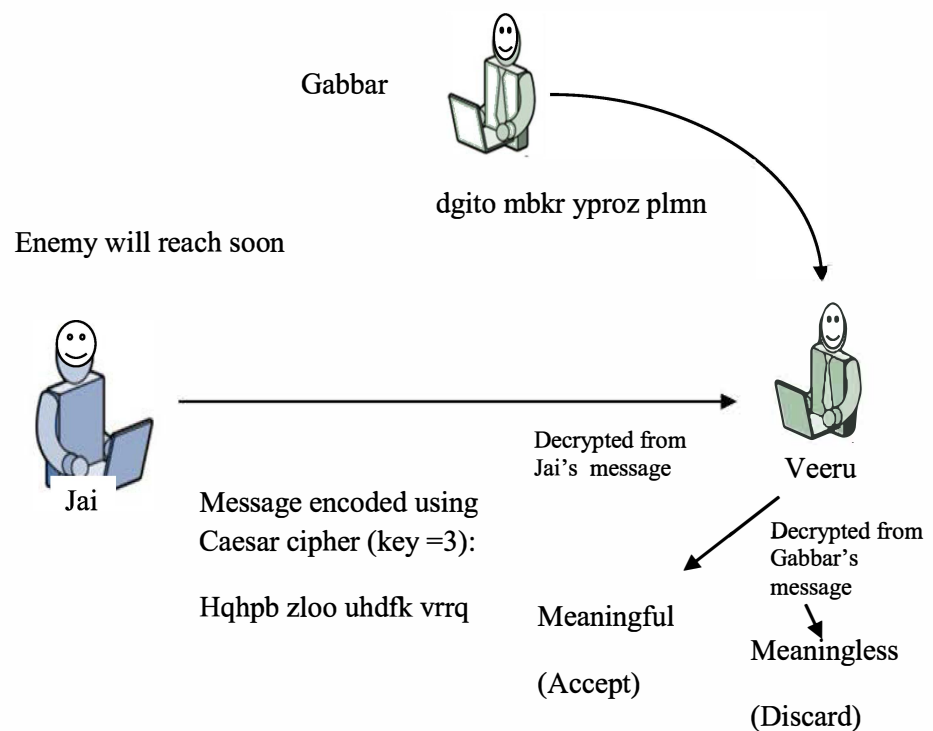


Figure 7.2 Symmetric encryption for providing authentication

Thus, a structure is required to be added to the transmitted message that enhances the authentication capabilities for example, addition of check sum or Frame Check Sequence (FCS) as error control measure to the message M and then encrypting the complete block (**Figure 7.3 (a)**) strengthens authentication. The receiver first decrypts the message using shared secret key and then calculates FCS from decrypted message.

If the calculated FCS matches with the FCS contained in the encrypted message, the message (plaintext) is considered as authentic. If the attacker tries to disrupt the decryption process by sending random message for decryption then it is unlikely that such relationship between decrypted message M and FCS prevail and hence attacker's messages are easily discarded. If instead of this above mentioned scheme (termed as internal error control) another scheme (termed as external error control) is used where the FCS is calculated by sender for the encrypted message and is concatenated with the encrypted message (**Figure 7.3 (b)**), the attacker again has opportunity to disrupt or interfere the decryption process. This is because the attacker can now create forged messages with valid FCS and may direct them towards receiver.

Another example where structuring adds towards enhancing the authentication capabilities is TCP segment encryption. The TCP segment is encapsulated in IP datagram and the part of datagram excepting IP header is encrypted. The TCP header information includes checksum, sequence number etc. In case the attacker tries to

create and send a forged datagram, the resulting encrypted text (containing TCP header information) will become meaningless and therefore leading to discarding of such forged datagrams [1][2].

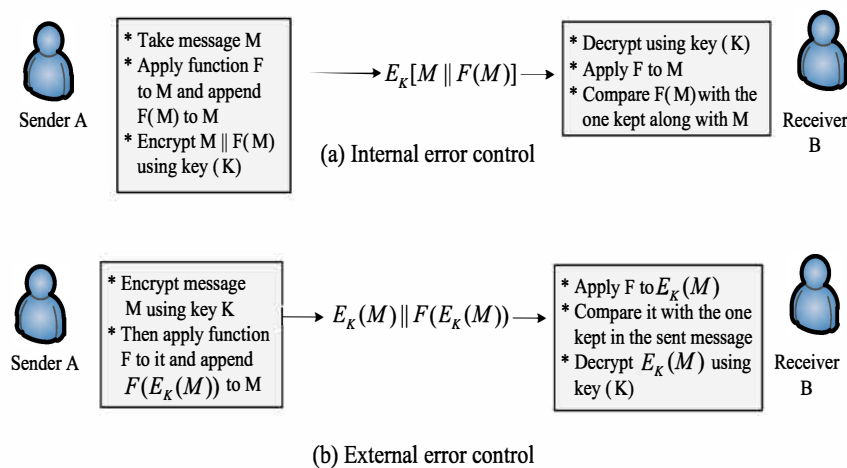


Figure 7.3 Two types of Error control using FCS (Function F)

Application of Public-key Encryption:

In Public-key Encryption each sender and receiver has two key pairs (private and public key). Thus, there are two possibilities (**Figure 7.1 (b)**, **Figure 7.1 (c)**). (i) Sender uses public key of the receiver for encryption of message M (ii) sender uses its own private key for encryption of message M. The former ensures only confidentiality while latter ensures authentication and signature. This is because in (i) above anybody can send encrypted message but only the intended receiver is able to decrypt the message as only he has his private key. In (ii) above only the sender can encrypt the message M using his private key while others are able to verify this by decrypting using corresponding public key. Here, the encrypted ciphertext becomes digital signature because only sender A owns his private key (K_A) not even receiver B has any idea about this. This scheme does not provide confidentiality as anyone who possesses public key of the sender can decrypt the encrypted message. **Figure 7.1 (d)** illustrates the public-key encryption scheme that provides both confidentiality and authentication. Hence, first sender encrypts M using his private key (K_A). This becomes its signature and used for authentication and is then encrypted using receiver's public key to ensure confidentiality. The major drawback in this scheme as compared to symmetric method is requirement of more calculations. Here, the public key encryption is applied 2 times and public key decryption is applied. 2 times making

count as four whereas in symmetric method only one encryption and one decryption is required [1][2].

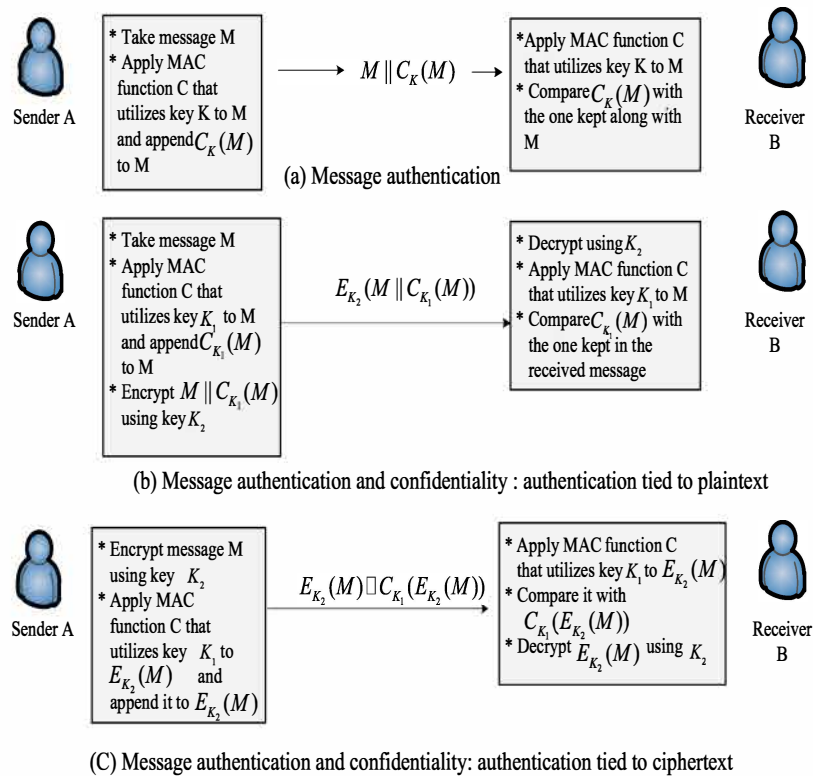


Figure 7.4 Uses of Message Authentication Code (MAC)

Use of Message Authentication Code for Authentication:

Message Authentication Code (MAC) is basically cryptographic check sum.

- It is evaluated using shared secret key and is of fixed size.
- The evaluated MAC is appended to the message.
- Considering communication between sender A and receiver B, message M and shared secret key as K, the MAC is evaluated as

$$\text{MAC} = C_K(M)$$

Where C is the MAC function that utilizes key K

- The receiver calculates MAC using same function on M and then compares the calculated MAC with the appended MAC. If the two matches the message is accepted otherwise rejected.
- The receiver ensures the following upon matching the MAC:
 - (i) The message is not modified on the way. If attacker tries to modify the message, the calculated MAC will not be same as that of appended MAC. The two MAC differs because attacker is unable to evaluate and append new MAC due to lack of secret key.

- (ii) The message is sent by A only as no other party knows the secret key and hence cannot create the message along with the appended MAC.
 - (iii) The sequence numbers corresponding to a message cannot be modified due to lack of secret key with any other party. Hence, sequencing remains unaltered.
- In comparison to encryption algorithm, the MAC algorithm is not reversible (unlike decryption). Also, MAC function is many-to-one e.g., considering 50-bit messages and 10-bit MAC, possible messages are 2^{50} while possible MACs are only 2^{10} . This means that single MAC is generated by $2^{50}/2^{10}=2^{40}$ different messages. Considering key of 6 bit size, $2^6=64$ different mapping from set of messages to set of MAC is possible.
- MAC can be used to provide either authentication or authentication and confidentiality both (**Figure 7.4 (a) – (c)**). In former, message is sent in clear and hence, only authentication is ensured (**Figure 7.4 (a)**). Latter, further has two sub cases: (i) where authentication is tied to plaintext (**Figure 7.4 (b)**) and, (ii) where authentication is tied to ciphertext (**Figure 7.4 (c)**). In case (i) encryption is performed after evaluating MAC. Two separate keys shared by sender and receiver are required under these cases. One for evaluating MAC, other for encrypting message. Out of these, case (i) is preferred as this reduces the attacker's chances.
- Authentication via MAC is preferred to authentication via encryption in some scenarios. These are:-
 - (i) Broadcasting of a message to more number of destinations e.g. broadcasting notifications or raising an alarm etc. Such messages are sent in plaintext along with MAC which can be verified and authenticated by destinations sharing secret keys.
 - (ii) For heavily loaded systems MAC can be verified for selective messages instead of decrypting all the incoming messages.
 - (iii) Computer programs may be provided with MAC instead of encrypting entire program. The MAC verification ensures program integrity.
 - (iv) Messages in certain applications like Simple Network Management Protocol (SNMP) need not be encrypted rather authenticated. Thus, a managed system working under the control of SNMP doesn't require commands in concealed form rather commands are required to be authenticated for taking necessary action as per the instructions.
- Separation of confidentiality and authentication functionalities may be helpful in some cases where confidentiality may be provided at lower level (transport or link layer) while

authenticity may be provided at higher level (application layer). This enhances the flexibility.

- MAC extends the message protection time. In case of encryption, once a message is decrypted upon receipt, its protection is lost. In case of MAC protection, the MAC is always associated with message not only in transit but otherwise also. So authenticity is guaranteed all the time.
- MAC itself does not provide repudiation feature like digital signature as sharing of key is involved between two parties i.e., sender and receiver. Digital signatures are created using secret known to only one party not to others.

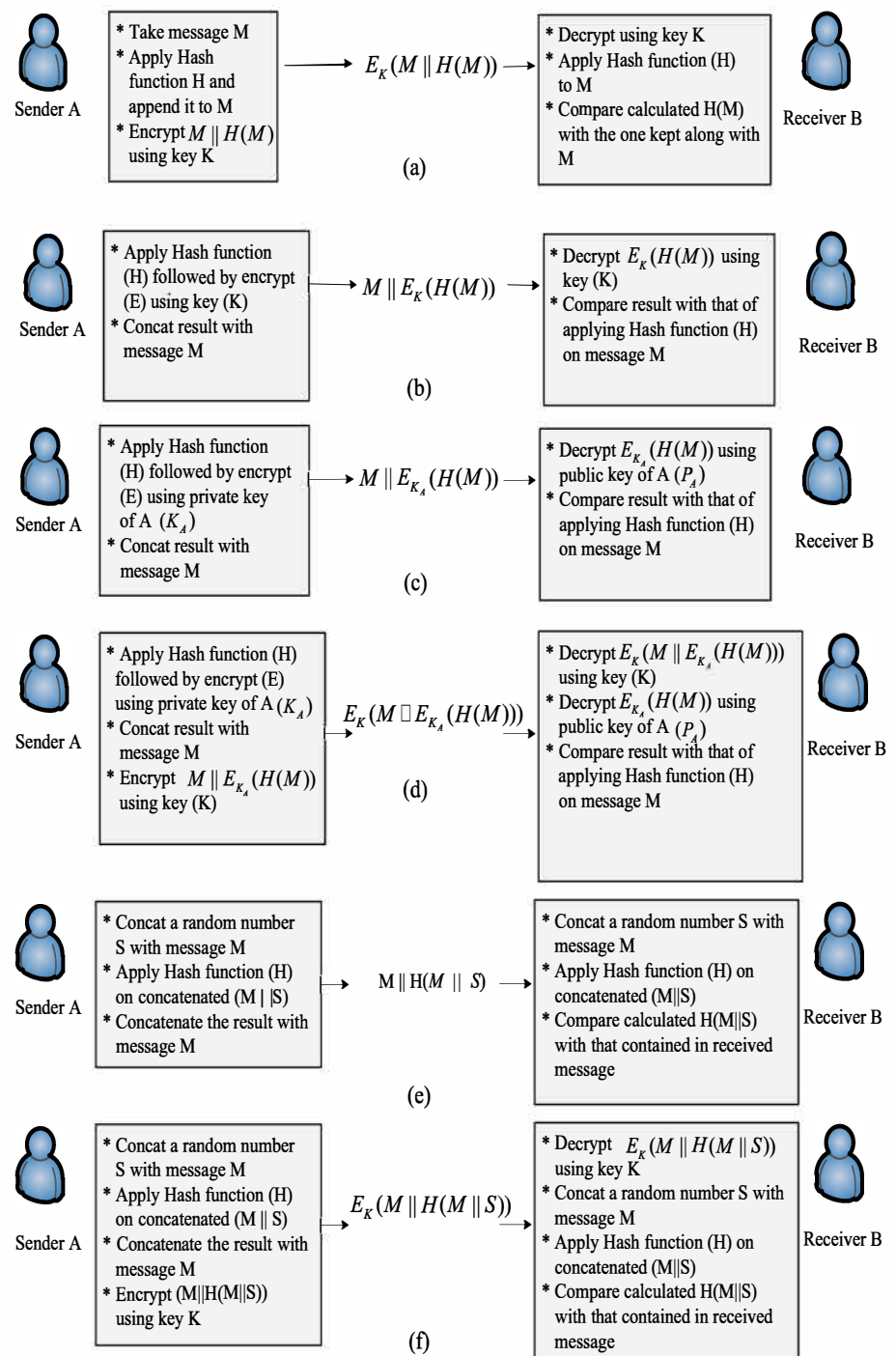


Figure 7.5 Uses of Hash function

Use of Hash Function for Authentication:

Hash function represents variation of MAC that doesn't use secret key rather only variable size message is used as input.

- Hash function is one way function and it is function of all message bits.
- It's output is fixed and changes on changing even a single message bit. The output is termed as message digest or hash value (code).
- A hash code is used for authentication in following ways:
 - (i) Message and Hash both are encrypted using shared secret key. This provides authentication and confidentiality both (**Figure 7.5 (a)**).
 - (ii) For application that do not require confidentiality. The calculated small sized hash is encrypted using symmetric key. This also reduces the processing overhead to encrypt/decrypt the entire message. The overall functionality of such hash use is equivalent to MAC functionality (**Figure 7.5 (b)**).
 - (iii) Private Key of sender is used to encrypt the calculated hash only, leaving message as such. This ensures authentication (also digital signatures) as only sender is able to perform such functionality because only he has the private key (**Figure 7.5 (c)**).
 - (iv) When digital signatures and confidentiality is required, the digital signature is created as in (iii) above and then both message and digital signatures are encrypted using symmetric shared secret key (**Figure 7.5 (d)**).
 - (v) Sender selects a random number S concatenates it with message M and then calculates hash. The calculated hash is appended to the message. No encryption is done in this method. The random number is shared between sender and receiver only. This implies no modification is possible by the attacker (**Figure 7.5 (e)**).
 - (vi) The message and hash calculated in (v) can be encrypted further to enhance confidentiality (**Figure 7.5 (f)**).

Among these above mentioned hash techniques, (ii) and (iii) are popular, though they do not involve encryption. This is due to following;

- Encryption using software is relatively slow as compared to encryption using hardware.
- Hardware used for encryption is usually optimized for large sized data. Hence, due to the initialization (startup) cost, small encryption is not effective on such hardware.
- Hardware cost for several systems cannot be considered as negligible.

- Algorithms for encryption (e.g. RSA algorithm) are usually patented and therefore cost is involved for attaining licenses.

7.4 MESSAGE AUTHENTICATION CODES (MACs)

- As discussed in the beginning of section 7.3, MAC is a fixed length authenticator.
- It is evaluated as $MAC = C_K(M)$
Where C is MAC function, M is variable length message, K is shared secret key and output from MAC is fixed length code.
- Any MAC function should meet the following requirements:-
 - (i) It is computationally infeasible to find or construct another message M such that $C_K(M') = C_K(M)$
 - (ii) The hash function should be uniformly distributed. i.e., for n bit size MAC the probability that $C_K(M') = C_K(M)$ is 2^{-n} .
 - (iii) It is not possible to apply some transformation on message M such that the hash of M' (transformed message) = hash of M. More specifically.
 $Pr = [C_K(M) = C_K(M')] = 2^{-n}$
- Consider message M that is formed by concatenating 64-bit blocks (X_i) as:

$$M = (X_1 || X_2 || \dots || X_m)$$

Define MAC as

$$\Delta(M) = X_1 \oplus X_2 \oplus \dots \oplus X_m$$

$$C_K(M) = E_K(\Delta(M)) \quad - (1)$$

Where, E is encryption algorithm (DES in electronic code book mode); K is secret Key; and $\oplus$ is Exclusive OR (XOR) operation.

Instead of Brute force attack, attacker opted for fraud insertion and created Y_m by replacing X_1 to X_{m-1} by new desired values Y_1 to Y_{m-1} .

$$Y_m = Y_1 \oplus Y_2 \oplus \dots \oplus Y_{m-1} \oplus \Delta(M) \quad - (2)$$

$$\begin{aligned} \text{Received } \Delta m &= Y_1 \oplus Y_2 \oplus \dots \oplus Y_m \text{ \{Substitute } Y_m \text{ from (2)\}} \\ &= \Delta M \end{aligned}$$

Thus, hash of Δm is same as per eqn. (1) above and therefore new message ($Y_1 || Y_2 || \dots || Y_{m-1}$) is inserted and still M accepted by receiver.

This should not happen and requirement (i) should be followed by MAC authenticator.

The requirement (ii) deals with protecting against brute-force attack. Consider, key size (K) > MAC size (n).

As MAC function is many-to-one, for applying brute-force attack, attacker must try different key values (K_i) such that

$$MAC_i = C_{K_i}(M) \text{ and } MAC_i = MAC_1$$

$$\text{where } MAC_1 = C_{K_1}(M)$$

In fact number of keys (set) will produce correct MAC (MAC_1). Thus attacker's task is now reduced to find actual key from this set and hence repetitive execution of this scheme is required.

Example 1: Key size 80 bit, MAC size = 32 bit

$$\text{I}^{\text{st}} \text{ round } 2^{80-32} = 2^{48} \text{ possible keys}$$

$$\text{II}^{\text{nd}} \text{ round } 2^{48-32} = 2^{16} \text{ possible keys}$$

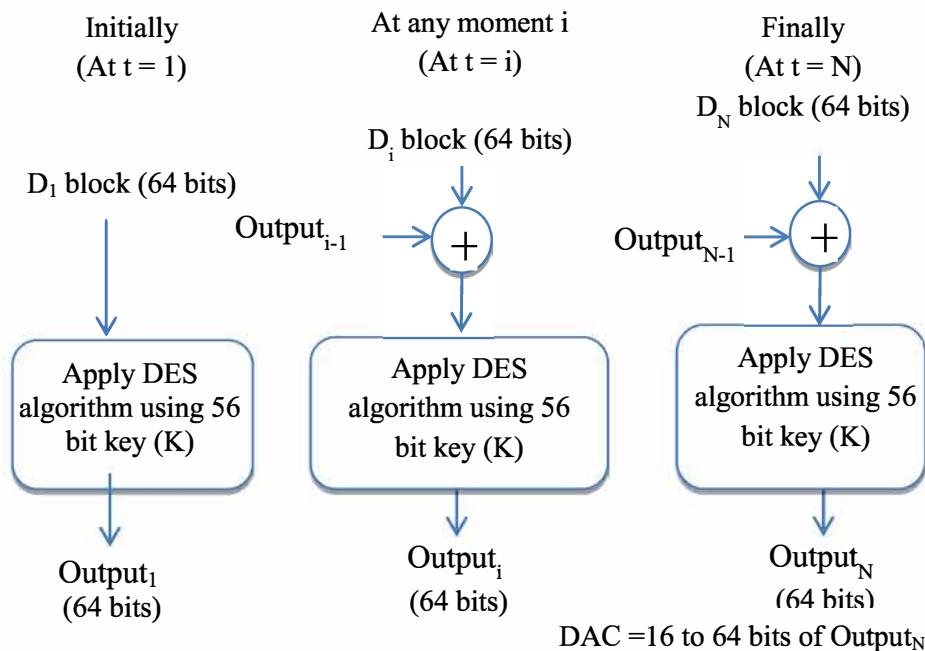
$$\text{III}^{\text{rd}} \text{ round } 2^{16} \Rightarrow \text{last round (1 possible key)}$$

In case $K \leq n$, the first round may produce a match. Thus, efforts required to find the authentication key is not less than the efforts required to find the decryption key and the key may be found in I^{st} round itself.

The requirement (iii) says that existence of weak spots (where algorithm proves to be weak) in M should be avoided otherwise the attacker may easily get early success in identifying new message (M') such that hash of M and M' are equal [2].

Example of MAC created:

- DES in Cipher Block Chaining (CBC) may be used to create/form MAC.
- Initial vector is initialized as zero.
- The data is grouped into 64-bit blocks ($D_1, D_2, \dots, D_N$).
- Final block may be padded on right side with zeros.
- The final output, Data Authentication Code (DAC) is of 16 to 64 size (Figure 7.6).



• Figure 7.6 Example of MAC based upon DES

7.5 HASH FUNCTIONS: SHA-1, MD5

- As discussed in section 7.3, a hash function H produces a hash value h as

$$h = H(M)$$

where, M is variable length message

- Hash value itself is not secret rather it needs to be protected by applying another function.
- The main purpose of hash function is to generate a fingerprint of data.
- The following are requirements for a hash function (H):
 - (i) Input is data block of any size.
 - (ii) Output is of fixed size.
 - (iii) Software and hardware implementations of H are practical.
 - (iv) Given input x , it is easy to calculate h as $H(x)$ but reverse calculations are computationally infeasible i.e., given h it is not possible to computationally find x in some time. This requirement is also referred to as one way property.
 - (v) Weak collision resistance: Given x , it is computationally insensible to find y ($\neq x$) such that $H(y) = H(x)$.
 - (vi) Strong collision resistance: No pair (x, y) can be found computationally such that $H(x) = H(y)$.
- (iv)th property ensures that if secret key is used during hash evaluation then it remains safe as reverse calculations are not possible.
- (v)th property ensures that an alternate message that generates some hash value is not possible.
- (vi)th property ensures resistance against birthday attacks.

In birthday attacks, attacker generates $2^{m/2}$ variations of message and compares them with actual set and then tries to find pair of messages that produces the same hash code.

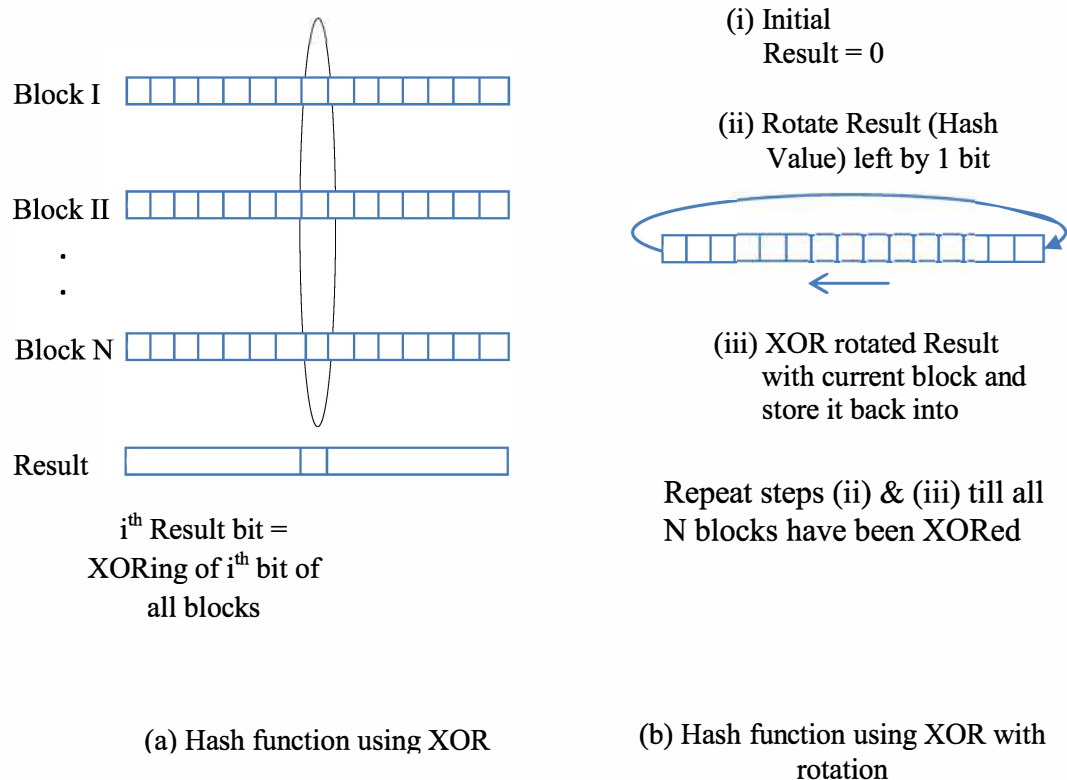


Figure 7.7 Two simple Hash function (one using XOR other using XOR with rotation)

Below two simple hash functions are presented to enhance the understandability [3].

- (1) Hash function (**Figure 7.7(a)**) that performs bit-by-bit XORing of every block to produce single bit of hash code (C_i) as:

$$C_i = b_{i1} \oplus b_{i2} \oplus \dots \oplus b_{im}$$

where

m = number blocks (each block is of n -bit size)

$b_{ij} = i^{\text{th}}$ bit of j^{th} block

- The above function is effective for data integrity check. It provides parity for each bit and is known as longitudinal redundancy check.
 - The function is less effective where data is predictably formatted e.g. in normal text files, the MSB in each octet is always zero. For 128 bit hash code effectiveness should be 2^{-128} but in the normal text file case it is 2^{-112} as 16 bit positions evaluates to zero hash code.
 - Thus a method to randomize the input is required.
- (2) Hash function that shifts one bit in circular fashion on hash code after each block is processed (**Figure 7.7(b)**)

- It has two steps
 - (i) n – bit hash code is initialized to zero.
 - (ii) For each n – bit block of data:

Rotate the hash code left then XoR it with block bits and store it back into hash code.
- This hash function is still of not much use even when encrypted hash code is used along with the message because it is easy to prepare alternate message which still evaluates same hash code.
- Another technique that encrypts message and hash code is as:

message M : sequence of 64 bit blocks $X_1, X_2, \dots, X_n$

$$\text{hash code } C = X_{N+1} = X_1 \oplus X_2 \oplus \dots \oplus X_n \quad - (1)$$

and append it to message

Encrypt message and hash code using CBC mode

Encrypted message $Y_1, Y_2, \dots, Y_{n+1}$

An example of manipulating message M while remaining undetectable via hash code was given by [5]

According to CBC definition

$$X_1 = IV \oplus D_K(Y_1)$$

$$X_i = Y_{i-1} \oplus D_K(Y_i)$$

$$X_{N+1} = Y_N \oplus D_K(Y_{N+1})$$

Substituting X_i 's into (1) above - results in hash code as:

$$X_{N+1} = [IV \oplus D_K(Y_1)] \oplus [Y_1 \oplus D_K(Y_2)] \oplus \dots \oplus [Y_{N-1} \oplus D_K(Y_N)]$$

This XoRing can be done irrespective of order of ciphertext block. Thus, hash code doesn't change upon permuting ciphertext blocks giving chances to attacker.

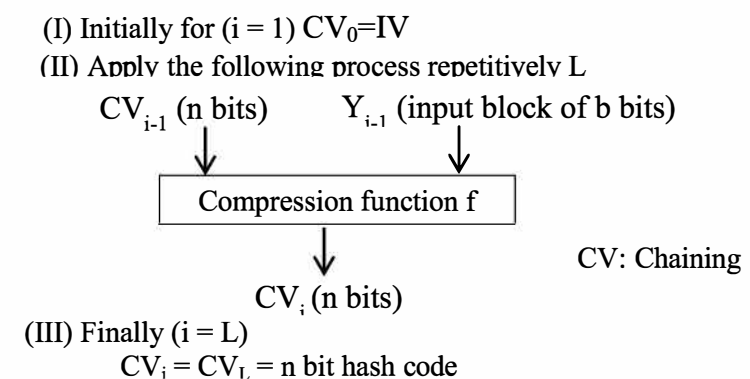


Figure 7.8 General Structure (Secure Hash Code)

Overall structure of a typical secure Hash function:

- The structure shown in **Figure 7.8** was proposed by Merkle [6].
- This structure is used by most hash functions including SHA (discussed in next section) and whirlpool.
- The input message is divided into L fixed sized blocks each of b bits.
- The last block is padded so that its size becomes b bits.
- The last block also contains length of message. The length is included to make attacker's task difficult.
- The hash function (H) repeatedly uses a compression function f . f takes two inputs: one b -bit block (Y_{i-1}) and other n bit chaining variable CV_{i-1} (previous step output). The initial value of CV_0 is IV . The final output (CV_L) is the hash code ($H(M)$). The intermediate CV_i (output of f) is calculated as
$$CV_i = f(CV_{i-1}, Y_{i-1}) \quad 1 \leq i \leq L$$
- The function f is termed as compression function because the block size (b) is often greater than hash code size (n).
- The presented general structure for secure hash code has an important property stated as:

If f is collision resistant then so is hash function H . This means that-

- (i) H can operate upon message of any size.
- (ii) The secure hash function designing is reduced to collision resistant f designing.
- (iii) Designing f is comparatively easy as it operates upon fixed sized inputs.
- Crypt analysis of hash function is directed to f and hence try to produce collisions for f .
- f itself consists of series of processing rounds.
- For any Hash function, collision does exist (as $b \geq n$) but these are computationally infeasible to identify.

Check your progress 1

- a. How MAC authentication is different from digital signature based authentication?
- b. Why requirements of MAC different from that of hash function?

7.6 SECURE HASH ALGORITHM (SHA)

- The basic structure provided in the previous section has proved fundamentally sound and hence most of the hash function design follows the refinement of this structure/compression function.
- Two approaches of hash algorithm are widespread: one that considers use of modular arithmetic and logic binary operations as the compression function (e.g., secure hash algorithm

(SHA)). Other that considers use of symmetric block cipher as the compression function (e.g., whirlpool).

- MAC approaches are also divided into two: one based upon SHA as the main part other that uses symmetric block cipher in cipher block chaining mode.
- Secure hash algorithm (SHA) was developed by National Institute of Standards and Technology (NIST).
- It is published as Federal Information Processing Standard (FIPS 180) in 1993 and subsequent revisions as (FIPS 180-1) in 1995 and (FIPS 180-2) in 2002.
- FIPS 180 is the initial secure hash standard, FIPS 180-1 describes SHA-1 while FIPS 180-2 defines three versions namely, SHA-256, SHA-384 and SHA-512.
- SHA-1 that produced 160 bit hash code is currently phased out.
- SHA-1, SHA-256, SHA-384 and SHA-512 uses modular arithmetic and logical binary operations as compression function.
- SHA-512 algorithmic logic and steps are presented next [3][4].
- Following are the parameters of SHA-512 algorithm.

Input: message having maximum length as 2^{128} bits i.e., $M < 2^{128}$

Output: 512 bit fixed

Block size: 1024 bits

Word size: 64 bits

No of steps: 80

- The overall structure of SHA-512 is shown in **Figure 7.9**
- The processing involves 5 steps
 - Step 1: A padding consisting of single 1-bit followed by variable number of 0-bits (as per the requirement) is added at the end (last block). The padding is always done even if the message fits completely in block or not. Padding is done such that the length becomes congruent to $896 \bmod 1024$ i.e. $\text{length} \equiv 896 \pmod{1024}$.
 - Step 2: The original message length (size) is treated as unsigned integer of 128 bit and is added in the last block such that the total length of message now becomes multiple of 1024 i.e., $\text{length message } (M_1, M_2, \dots, M_N) = N * 1024 \text{ bits}$.
 - Step 3: For holding hash code or value 512 bit buffer is used. It is considered as combination of eight 64-bit registers and these are named as a, b, c, d, e, f, g and h (**Figure 7.10**).

For initializing these registers square roots of first eight prime numbers is calculated and the first 64 bits corresponding to fractional part of each root is assigned to these registers in big endian format (where MSB corresponds to leftmost byte position). Thus initial

values (in hexadecimal) of these registers during first block processing are [3]:

$$a = 6A09E667F3BCC908$$

$$b = BB67AE8584CAA73B$$

$$c = 3C6EF372FE94F82B$$

$$d = A54FF53A5F1D36F1$$

$$e = 510E527FADE682D1$$

$$f = 9B05688C2B3E6C1F$$

$$g = 1F83D9ABFB41BD6B$$

$$h = 5BE0CDI9137E2179$$

- Step 4: processing (**Figure 7.10**) of each 1024-bit block of message is done in 80 rounds (Shown by F in **Figure 7.9**). Each round considers the buffer (abcdefgh) and updates its value. Initial value of this buffer is initialized as per step 3 above.

- ❖ For i^{th} block of message the buffer has intermediate hash value H_{i-1}

- ❖ Each round t considers following inputs:

- (i) A 64-bit value W_t derived from current block of message (M_i).

- (ii) An additive constant K_t ($0 \leq t \leq 79$). Each K_t for a round is evaluated from first 64 bits of fractional part of cube root of first t^{th} prime number. These K_t are used to remove regularities in the input data.

- (iii) Buffer values from previous round.

- ❖ The output of last round (80th round) is added with H_{i-1} to produce final H_i from this processing. Each of the eight registers (a, b, c, d, e, f, g and h) are added independently using modulo 2^{64} addition.

- Step 5: After processing all blocks, the output Message Digest (MD) (of 512 bits) from N^{th} block processing represents the hash code or value

Hiding the round complexities, entire algorithm may be summed as:

$$H_0 = IV$$

$$H_i = \text{SUM}_{64} (H_{i-1}, \text{abcdefgh}_i)$$

$$MD = H_N$$

where,

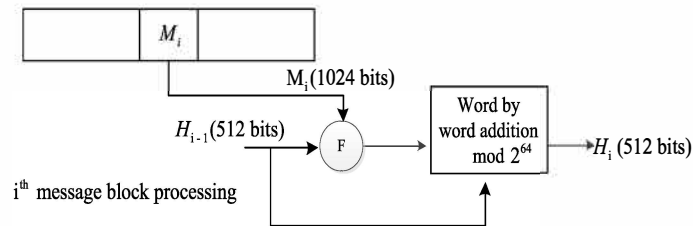
IV is the initial value of buffer (abcdefgh)

$abcdefgh_i$ represents out of last round processing for i^{th} message block

SUM64 represents addition in modulo 2^{64} for each word.

MD represents final message digest or hash value

- Message converted into N-blocks (each of 1024 bits).
- While doing so, if entire message fits in exactly N blocks then an extra block is added.
- Last block contains message data (if any), message length and is padded using single 1 followed by 0's (binary values).



For 1st block processing $H_0 = \text{IV}(512 \text{ bits})$

Total block processed = N

Hash code = H_N

Figure 7.9 Message Digest Generation (SHA-512)

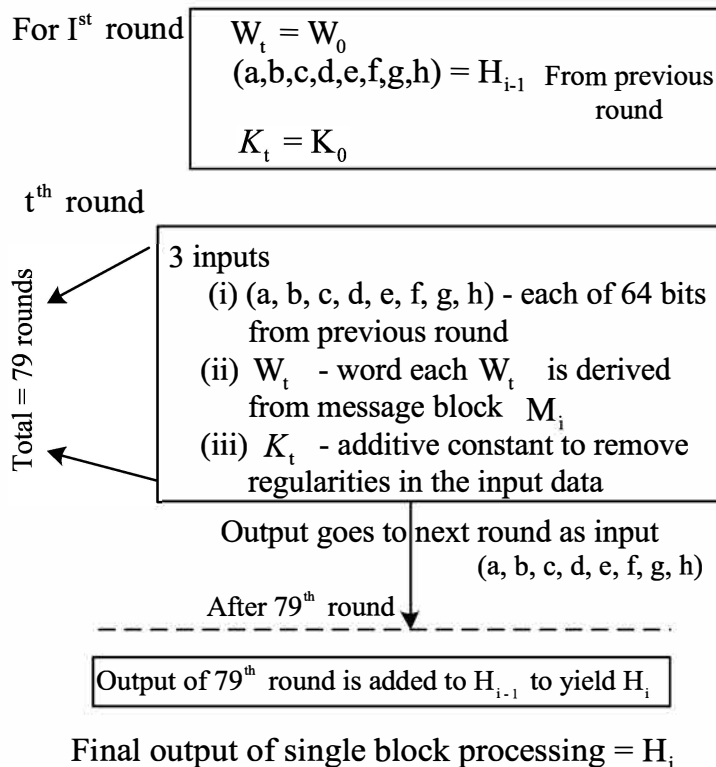


Figure 7.10 Single Block (1024 bit) Processing (SHA-512) corresponding to i^{th} block

Check your progress 2

- a. What is input and output of SHA-512?
- b. Why is padding done in SHA-512 algorithm?

7.SUMMARY

For message authentication, the sender evaluates a fixed length value (hash) after applying a function that takes message (and may or may not consider the shared secret key) as input. This evaluated fixed length value is then appended to the message and later verified at the receiver upon receipt. This unit covers message authentication in depth. The various authentication requirements and functionalities are discussed. The concept of Message Authentication code (MAC), its properties and features are dealt with sufficient considerations. Toward end, the structure of a secure hash algorithm is also provided to enhance your understanding.

Terminal Questions

1. *In what order symmetric encryption and error control be used to provide message authentication? Why?*
2. *How can a hash value be used such that it provides equivalent security as that of MAC?*
3. *In how many ways can message authentication be done?*
4. *Explain the use and utility of hash and MAC functions.*
5. *Differentiate between:*
 - (a) *hash and MAC.*
 - (b) *MAC and digital signature*
6. *What are the key features and characteristics of secure hash function?*
7. *What is meant by collision resistance? Differentiate between weak and strong collision resistance.*
8. *How is compression function useful in a hash function?*

References:

1. Forouzan, Behrouz A., "Chapter 11: Message Integrity and Message Authentication", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
2. Stallings, William, "Chapter 11: Message Authentication and Hash Functions", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
3. Stallings, William, "Chapter 12: Hash and MAC algorithms", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
4. Forouzan, Behrouz A., "Chapter 12: Cryptographic Hash Functions", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.

5. Jueneman, R., Matyas, S. and Meyer, C., "Message Authentication." IEEE Communications Magazine, September 1988.
6. Merkle, R. "One Way Hash Functions and DES." Proceedings, CRYPTO '89, 1989; published by Springer-Verlag.

UNIT – 8

Digital Signatures

Structure

- 8.0 Introduction
- 8.1 Objectives
- 8.2 Understanding the Digital Signatures
- 8.3 Protocols for Authentication
- 8.4 The Digital Signature Standard (DSS)
- 8.5 Summary
- 8.6 Terminal Questions

8.0 INTRODUCTION

A digital signature may be considered as the originator's stamp that is put on the message. It signifies to the recipient on a network that the message is owned by the originator and is not tampered by an adversary or attacker. Digital Signature utilizes public key cryptography. It alone does not ensure confidentiality rather ensures message authenticity and integrity. In this unit, the notion of digital signature is introduced and explained. It is then followed by mutual authentication and one way authentication. Towards end, NIST digital signature standard (DSS) is discussed.

8.1 OBJECTIVES

Digital signatures are associated with one's digital identity. On Internet, digital signatures are used to accomplish the individual user identities. Financial transactions may be carried in a safe and secure manner by using digital signatures of users. After the end of this unit, you should be able to:

- understand the concepts and requirements of digital signatures.
- comprehend the ways for providing mutual and one way authentication using digital signatures.
- realize the Key Distribution Centre (KDC) based algorithms for symmetric authentication & encryption and also asymmetric authentication & encryption.
- understand one of the digital signature standards i.e., Digital Signature Standard (DSS) published by National Institute of Standards and Technology (NIST).

8.2 UNDERSTANDING THE DIGITAL SIGNATURES

Message authentication solves one domain of problems in security. It is basically used to protect communication between two parties from third party i.e., unauthorized third party is kept out of communication. But it is unable to solve dispute between two parties i.e., two parties may themselves cause problem to each other. Let us consider communication between two parties – A and B. A can forge an unwanted message using the shared encryption key and claims that it was send by party B. In fact, B didn't send anything. It is also possible that party B may deny later after sending the message to A. This is possible because one cannot prove that message was indeed send by B. One may answer that this message might be a forged one created using shared encryption key by A himself. Such scenarios are a practical reality e.g. electronic fund transfer between two parties, a telephonic conversation between a stock broker and a stock holder regarding sale and purchase of shares etc. Thus, whenever there is a possibility of denial by any one party during communication that leads to distrust between them, something more than bare authentication is required. This something more actually authenticates the message in such a manner that the creator of message cannot later deny and is termed as Digital Signature [1] [2]. Digital Signature is nothing but a code calculated from the message and protected by using private key of the sender. It is attached to the message. Digital Signatures must meet the following:

- (i) It must enforce the creation time stamp along with message authorship.
- (ii) It must authenticate the contents of the message.
- (iii) In case of disputes it must be verifiable.

Thus, following are the requirements for Digital Signature:

- It must depend on the message.
- It must use sender's unique secret key.
- Digital signatures must be easy to: produce, recognize and verify.
- Forgery of digital signature must be impossible i.e., computationally infeasible.
- Copy of digital signature can be kept in storage.

The digital signature approaches are of two types: Direct and arbitrated.

Digital Signatures without involving an arbiter i.e., Direct Digital Signatures:

Assumptions

- (i) Direct digital signatures are applicable for communication of two parties where one acts as source and other acts as destination.

- (ii) Destination has the knowledge of source public key.

Direct digital signatures are created by encrypting the complete message by private key of sender. It may also be created by first calculating the message hash code and then encrypting the hash code by using private key of the sender.

If confidentiality is also required then the entire message and its digital signature are encrypted. The encryption may be done either using receiver's public key or shared symmetric key. The encryption in such case is performed after evaluating signatures because under dispute the third party may require message and its corresponding signatures. The receiver may keep message and its corresponding signatures for future use under dispute settlements. If the message is encrypted before evaluating signatures then under dispute the third party may require access to decryption key for reading and comprehending the message.

Each digitally signed message may include time stamp. The direct signature method depends heavily upon sender's private key and hence it must be kept safe. The sender must report immediate to administrative authority if the private key gets stolen. The direct signature method is still prone to some threats where the attacker who gets the private key of the sender somehow may tamper the message by putting old time stamps in fraud messages.

Digital Signature involving an Arbiter:-

Owing to the issues in direct signature method, an arbiter may become functional and solve the issues pertaining to sender. An arbiter is a trusted party or system and both sender & receiver trust him for secure communication. The communication between sender S, and Receiver R takes place through arbiter A. The sender's message M goes to arbiter who tests the message and its signature. Upon verification of origin and content, the message is forwarded to receiver.

During verification, arbiter may or may not see the message contents. Hence, there are three possibilities (**Figure 8.1**) under arbitrated digital signature methods: (1) use of symmetric encryption such that arbiter sees message (2) use of symmetric encryption such that arbiter does not see message (3) use of public key encryption such that arbiter does not see message.

In possibility (1), the message M and associated signatures are sent to A. The signature contains identifier of S i.e. ID_S along with hash of message i.e., $H(m)$. A decrypts the signature of S using the shared symmetric key between himself and S i.e. K_{SA} . A adds ID_S and time stamp to the message and signature of S. This newly created message is encrypted using shared symmetric key of A and R. Upon receiving the message from A, R can verify the message and keep the message along with the signature for future use under disputes. R can also verify the time stamp which implies that the message is not the replayed one. In this method, the receiver is not able to verify the signature of S but considers it authentic because the message was routed through A. In case of disputes, R sends message to arbiter who can decrypt and verify the signature of S.

In this method, the arbiter is able to see the message, still it is trusted highly by both S and R. S and R trust that A does not reveal the shared keys and also that under disputes, A behaves ideally [2].

In (2), the arbitration is done similar to that of (1) with a difference that it also ensures confidentiality. Here, the arbiter is not able to see the contents of the message send by S. S encrypts the message M using key shared with R i.e. K_{SR} then it evaluates hash ($H(E_{K_{SA}}(M))$) of the encrypted message. This hash along with ID_s is encrypted using K_{SA} and send to A. A decrypts and sends a newly created message to R. Thus, A is not able to read the messages between S and R. The new message created by A contains all the data send by S and a time stamp T. This message is encrypted by using K_{AR} . Thus, this method is able to prevent the fraud on behalf of S or R but still there is a possibility that A might form an alliance with either S (to fool R) or with R (to fool S).

In (3), the double encryption utilizing public key is done that removes the threats and issues raised in method (1) and (2). The message M is encrypted by S using its private key (K_S) followed by another encryption using public key of R (P_R). This doubly encrypted message behaves as signature. The ID_s and signature is encrypted using private key K_S and send along with unencrypted ID_s to A. A verifies using public key of S (P_S) and extracts the signature. A now creates a new message containing extracted signature and time stamp. This new message is send by A after encrypting with its private key i.e., K_A . The R verifies the time stamp after decrypting using public key of A (P_A) and signature of S by using public key of S and its own private key. Though this method involves encryption twice yet it enjoys following advantages:

- (i) Contents of M are hidden from A.
- (ii) No information is shared between S and A or S and R, hence it is not possible to form an alliance among S and A or S and R.
- (iii) Even if K_S gets compromised, the adversary is not able to create a message with incorrect time stamp.

- | |
|--|
| <p>(1) $S \rightarrow A : M E_{K_{SA}}(ID_s H(M))$</p> <p>(2) $A \rightarrow R : E_{K_{AR}}(ID_s M E_{K_{SA}}(ID_s H(M))) T$</p> <p>(a) Use of symmetric encryption for digital signature such that arbiter sees message.</p> <p>(1) $S \rightarrow A : ID_s E_{K_{SR}}(M) E_{K_{AS}}(ID_s H(E_{K_{SR}}(M)))$</p> <p>(2) $A \rightarrow R : E_{K_{AR}}(ID_s E_{K_{SR}}(M) E_{K_{SA}}(ID_s H(E_{K_{SR}}(M)))) T$</p> <p>(b) Use of symmetric encryption for digital signature such that arbiter does not see message.</p> <p>(1) $S \rightarrow A : ID_s E_{K_S}(ID_s E_{P_R}(E_{K_S}(M)))$</p> <p>(2) $A \rightarrow R : E_{K_A}(ID_s E_{P_R}(E_{K_S}(M))) T$</p> <p>(c) Use of public key encryption for digital signature such that arbiter does not see message.</p> |
|--|

Figure 8.1 Arbitrated Digital Signature Possibilities

Check your progress 1

- a. What are characteristics of digital signature? Why is digital signature required?
- b. Why is hashing done during evaluation of digital signature?

8.3 PROTOCOLS FOR AUTHENTICATION

Digital signatures provide authenticity to message and to its originator. Considering two party communication, authentication is either mutual or one-way. Therefore, discussion in this section pertains to two categories: (1) Mutual authentication and (2) One-way authentication.

I. Authentication of each other (mutual authentication):

Mutual authentication protocols assume that the two communicating parties possess secret key or private/public key pairs. In mutual authentication, the two communicating parties prove each other identity and are able to evolve session key for securing the communication. The authenticated key exchange should be performed such that the (i) confidentiality requirement is met i.e., information exchange must use encryption technique and (ii) message replays doesn't affect the ongoing process. The replays may lead to key compromise, other party impersonation and key exchange disruption. **Table 8.1** lists the various replay attacks [2][3].

S.No.	Type	Comments
1.	Simple replay attack	A copy of the original message is replayed later by the attacker.
2.	Logged repetitions	A timestamped message may be used for replays within valid time limits.
3.	Repetition that cannot be detected	This can happen when the original message is suppressed while the replayed message is communicated.
4.	Backward replay without modification	Under symmetric encryption, the encrypted message is replayed back to the sender such that he is unable to differentiate between the actually send and the replayed messages.

Table 8.1 Replay Attacks

There 3 possibilities against replay attacks:

- (i) Sequence numbers are used in each message.
 - (ii) Timestamps are used in each message.
 - (iii) First party gives a challenge (nonce) and expects it back from other in response.
- First approach requires tracking of sequence numbers for each participant and due to this overhead, sequence numbers are seldom used in authentication and key exchange.

- Second approach based on timestamp is useful in connectionless applications. It cannot be used for connection oriented applications due to following reasons:
 - (i) Another protocol that requires synchronization among various processor clocks is required. This protocol should be fault tolerant and secure.
 - (ii) If one of the party's clock faults, there are chances of attack.
 - (iii) Precise synchronization is difficult due to variable and unpredictable network delays.

The acceptable window of time in this approach should be large enough to accommodate the network delays. On the other hand, it should be small enough so that chances of attacks are less.
- The third approach that of challenge/response requires overhead of handshake before transmission and is hence not suitable for connectionless applications. Connectionless applications are basically faster applications that usually avoid overhead of handshakes.

Symmetric encryption based mutual authentication approaches:

- The symmetric encryption methods are used to provide mutual authentication to the two users.
- They usually involve a trusted third party termed as key distribution centre (KDC) who is responsible for key distribution.
- It is assumed that each participating user or communicating party shares a master secret key with KDC (**Figure 8.2**).
- The secure communication between any two users or parties is done by encrypting the data packets using shared symmetric key that has short lifetime. It is termed as session key (**Figure 8.2**).
- The session key is usually generated by KDC. It is encrypted by master key (shared between KDC and communicating party) and is distributed to both the communicating parties.
- In this section, three such schemes namely, Needham and Schroeder [4], Denning [5-6] and KEHN [7] are discussed. This discussion also serves as initial platform for a well known KDC based authentication protocol known as Kerberos.

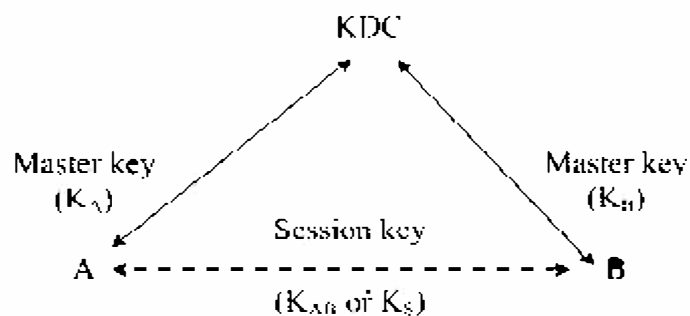


Figure 8.2 A two level hierarchy of symmetric keys.

Note: In this unit K_A (or K_B) refers to shared key between A (or B) and KDC whereas in public key encryption methods in other units K_A (or K_B) refers to private key of A (or B).

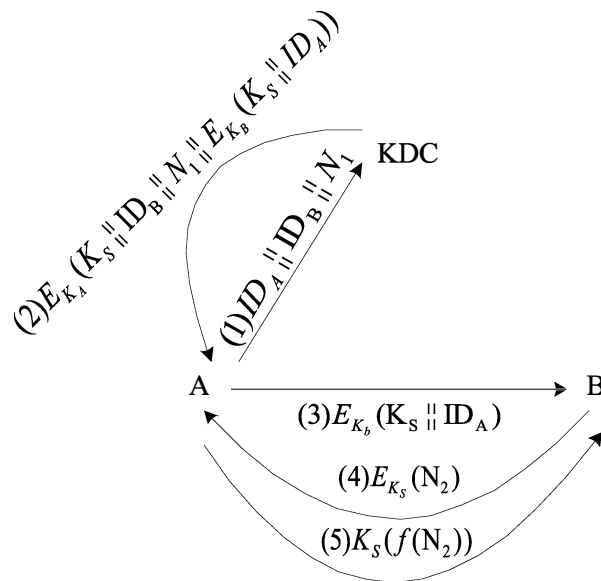


Figure 8.3 Needham and Schroeder method

Needham and Schroeder method [4]:

The steps (1 – 5) in Needham and Schroeder method (**Figure 8.3**) are as follows:

- A requests KDC for a new session key between A & B (indicated via their identifiers ID_A & ID_B). A nonce is kept in request to prevent reply of request and to indicate that this is fresh request.
- A decrypts the response given by KDC using shared master key (K_A). From the response, A is able to get session key (K_S) generated by KDC and is also able to verify nonce N_1 (which means this is response against its own request). A is unable to decrypt the part of response encrypted using K_B . It hence forwards this part to B.
- B decrypts the message sent by A (using master key K_B) and is able to extract the session key (K_S). Thus, now both A & B possess some secret key (K_S). Presence of ID_A , ensures that the message/request is originated by A.
- B selects a new nonce and encrypts it using recently acquired session key (K_S). [Step 4]. A decrypts the message and is able to get nonce. A applies function f to this nonce and encrypts it using K_S and send it to B. [Step 5]. Step 4 and 5 assures A and B mutually that they now possess session key and are now ready for secure communication using this session key.

- Needham and Schroeder protected key generation and sharing has an issue. If an adversary is able to compromise an old session key between A and B then he impersonates A and can replay message 3 to B. B as usual, extracts the session key and puts a new nonce encrypted using this old session key. Adversary decrypts message 4 and sends message 5 to B. B verifies the message and enters into communication with adversary using this old session key.

Denning's improvement:

- Denning [5][6] suggested incorporating timestamp in steps 2 and 3 using distributed clock mechanism to overcome the weakness of Needham and Schroeder protocol. The entire communication is shown in **Figure 8.4**:

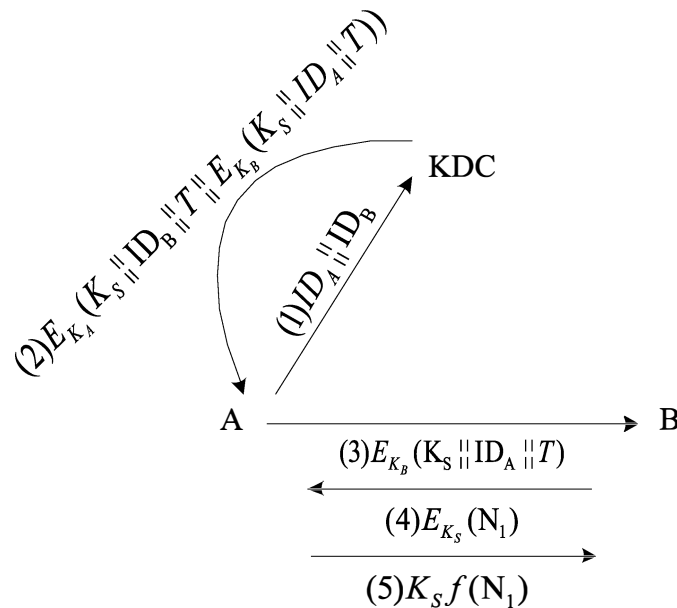


Figure 8.4 Denning's improvement

- Time stamps in this protocol ensures:
 - (i) Fresh message exchange
 - (ii) As the time stamps are communicated using encryption, the adversary who is in possession of old session key is not able to forge the time stamp and hence the replay of message 3 will be treated as useless due to wrong time stamp.
- The Denning protocol may face problem if the distributed clocks become unsynchronized.
- To overcome this problem, the communicating parties may regularly check their clock with that of KDC or the hand shaking messages may involve nonce along with time stamps [5] [6].

An improvement to both of the above protocols is given by **KEHN** [7] is shown in **Figure 8.5**.

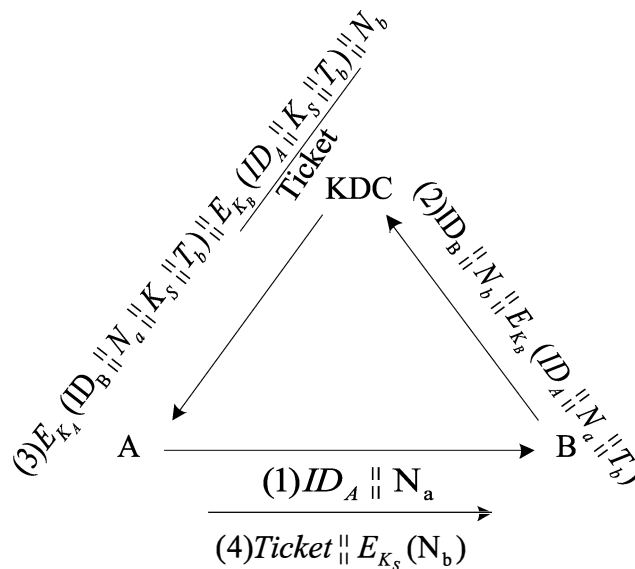
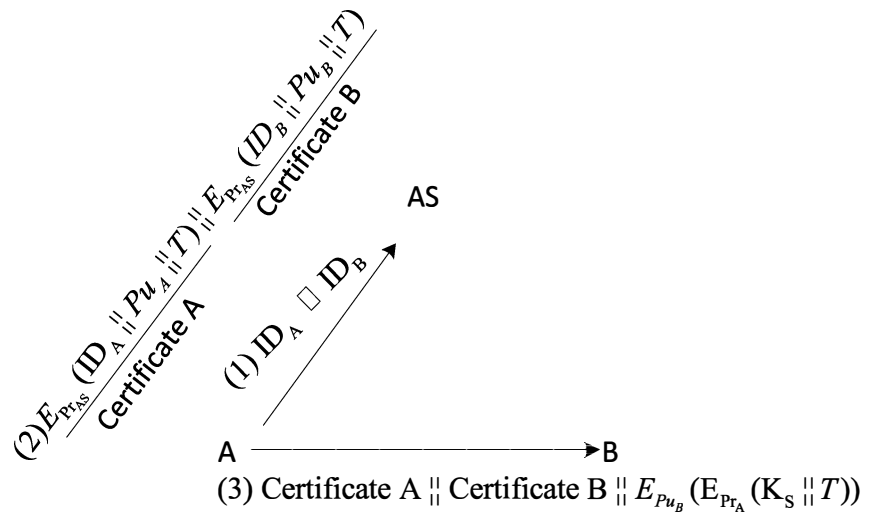


Figure 8.5 KEHN's improvement

- A initiates the communication and sends message containing selected nonce N_a to B. This nonce is returned to A via KDC which ensures that communication is fresh.
- B appends timestamp to A's message and forwards it in encrypted form to KDC indicating that a session key between A and B is required.
- KDC creates a ticket (encrypted ID_A , session key (K_s) and timestamp (T_b)) and adds other information for A (ID_B , N_a , K_s and T_b).
- Upon receipt of message 3, A extracts session key (K_s), confirms that this is fresh and creates message 4 containing ticket and encrypted nonce of B (N_b).
- Message 4 provides securely the session key to B via associated ticket. It also ensures B that A is in possession of session key.
- An important aspect of this protocol is that the unexpired ticket may still be used again by A to derive a new session key directly without involving KDC.
- The expiry of ticket is checked by B (for T_B) using its own clock and hence no synchronization of clocks is required.

Public encryption key based mutual authentication approaches:

- A compact public key based protocol that uses timestamps is shown in **Figure 8.6**.
- The protocol uses central Authentication Server (AS) for distributing public key certificates not for key distribution.
- As the session key is chosen by A, AS has no idea about it and this ensures that no other party except A and B has key.
- This protocol requires synchronization of clocks.



Pu = Public key

Pr = Private key

Figure 8.6 Public key encryption protocol that uses timestamps

- Another protocol (shown in Figure 8.7) proposed by Woo and Lam [8], does away with clock synchronization issues and uses only nonces.

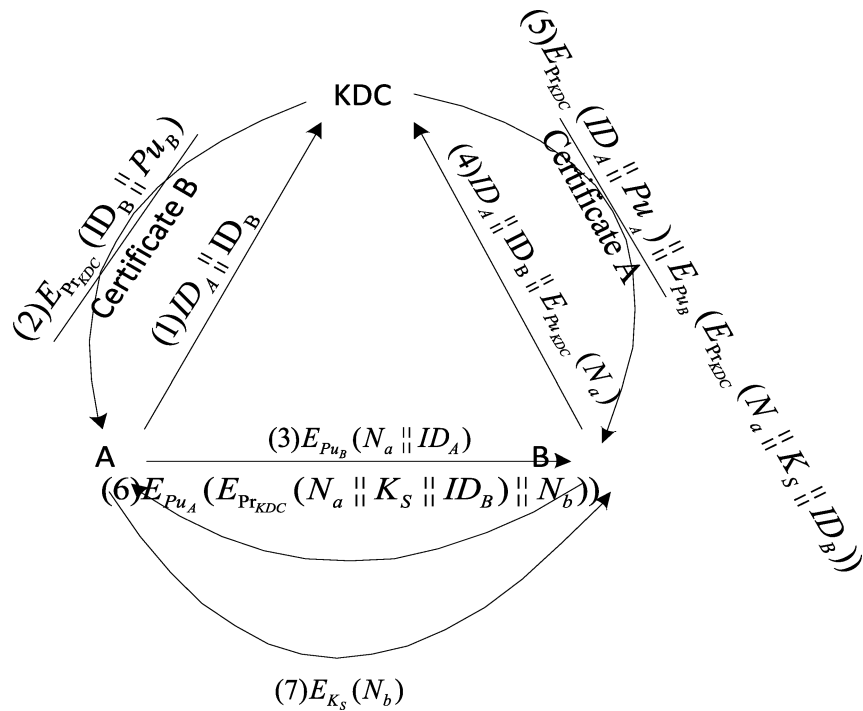


Figure 8.7 Woo and Lam protocol for Sharing Session Keys [8]

- Steps involved are briefly explained as:
 - (1) Request of A to KDC.
 - (2) KDC gives B's public certificate to A, A extracts public key of B.
 - (3) A sends nonce N_a encrypted using B's public key.
 - (4) B encrypts this nonce using KDC's public key and forwards A's request to KDC.
 - (5) KDC provides A's certificate to B along with encrypted session key bound to N_a . The encryption with Pu_B ensures that only B is able to extract the encrypted session key.
 - (6) The encrypted session key bound with N_a is forwarded by B to A. B also puts its nonce (N_b) and encrypts the entire message.
 - (7) A retrieves session key and acknowledge B by encrypting N_b with this session key.
- The authors later found a flaw in this protocol and suggested use of ID_A by KDC in step 5 i.e., session is bound not only to nonce, N_A but also to user ID_A [8][9].
- Thus, pair $\{ID_A, N_a\}$ uniquely identifies request of A for session key.

II. **Authentication of only one side i.e., one way authentication:**

- In some cases authentication is required in only one direction i.e., receiver requires guarantee that the message is from the particular sender.
- One such system of that requires one way authentication is E-mail system which has following issues:
 - (i) Sender and receiver are not necessarily online at the same time.
 - (ii) The header of e-mail is not encrypted because the e-mail is forwarded by intermediate systems in store and forward manner (using SMTP) after reading and understanding the information contained in header.
 - (iii) The message contained in e-mail can be encrypted for ensuring confidentiality and privacy.
- Both symmetric and asymmetric encryption is used for enhancing security in e-mail system.

Use of symmetric encryption for one way authentication

Assumptions:

- (1) Presence of key distribution centre (KDC) who shares symmetric secret key with each participant.
- (2) Session key is generated by KDC and is supplied to communicating participants.
 - In this method, KDC encrypts the session key along with ID of the communicating party.
 - Upon decryption, each party gets the session key and is able to verify the other participant using ID.

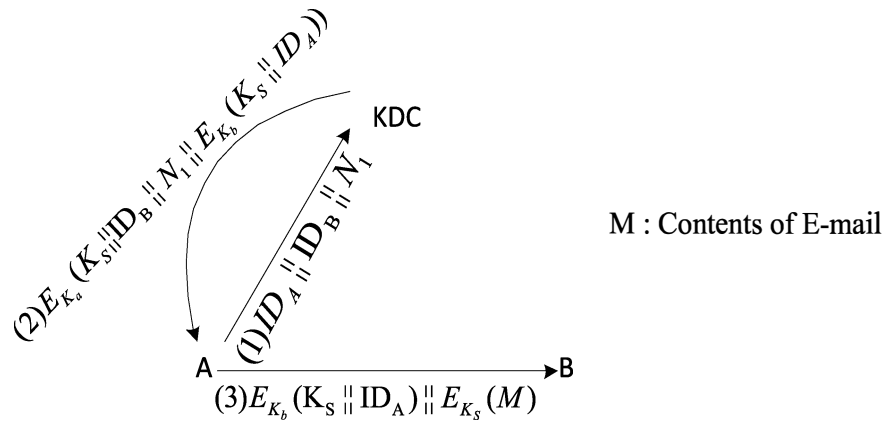


Figure 8.8 Use of symmetric encryption

- This method does not protect against replays and use of timestamp may improve the situation.

Use of public key encryption for one way authentication:

Using public key encryption, either confidentiality or authentication or both may be provided as shown in **Table 8.2**. It is assumed that A is sender while B is receiver and B knows A's public key.

S.No.	Property attained	Method
1.	Confidentiality	$E_{Pu_B}(K_S E_{K_S}(M))$
2.	Authentication	$M E_{Pr_A}(H(M))$
3.	Authentication and confidentiality	(i) $E_{Pu_B}(M E_{Pr_A}(H(M)))$ (ii) $M E_{Pr_A}(H(M)) E_{Pr_{AS}}(T ID_A P_{u_A})$

Digital Certificate

Table 8.2 Use of public key encryption

- In (2), another user (adversary) may capture the message, calculate new hash and encrypt message using his private key and deliver it to receiver. This should not be permitted. Hence message M should be encrypted (also in 3 (ii)).
- For encrypting message, one time session key may be used. This key is itself encrypted using B's public key before transmitting.

Check your progress 2

- How does KDC action differ in case of mutual authentication and one way authentication?
- What role does digital certificate play during authentication?

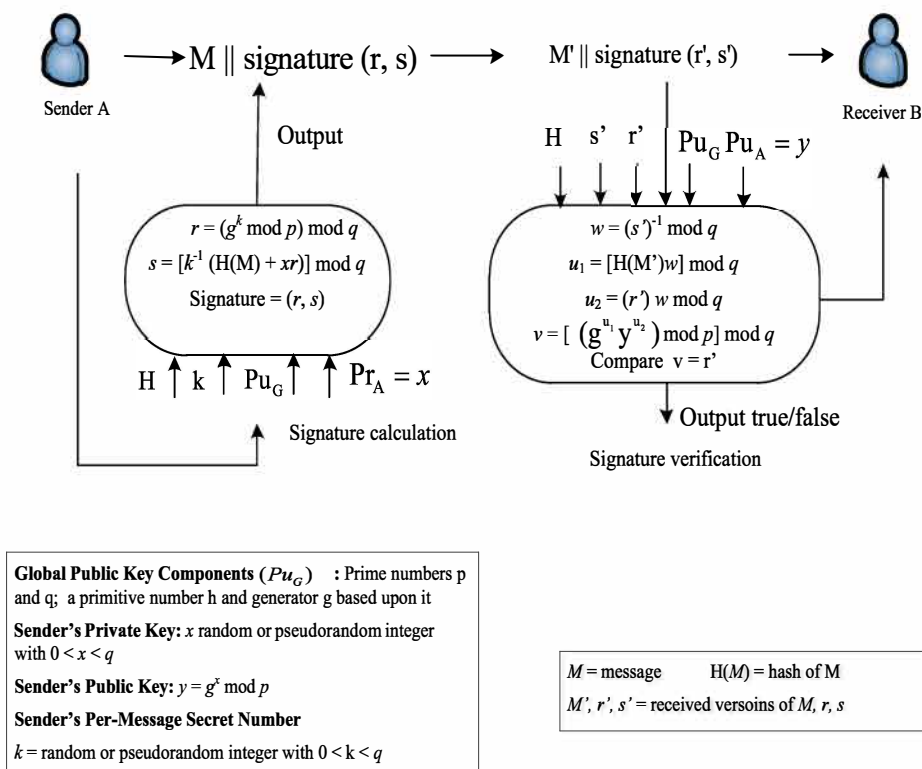


Figure 8.9 DSS approach for digital signing

8.4 THE DIGITAL SIGNATURE STANDARD (DSS)

- Digital Signature Standard (DSS) is published by National Institute of Standards and Technology (NIST) as Federal Information Processing Standard (FIPS 186).
- DSS involves hash calculation using Secure Hash Algorithm (SHA) while digital signature using Digital Signature Algorithm (DSA).
- Though several improvements have been made to DSS, the original DSS algorithm is discussed here.
- In comparison to RSA: DSS is used only for signing whereas RSA can also be used for encryption or key exchange. For some prime number p , DSS outperforms RSA signatures.
- In RSA method, a hash of message is evaluated and is encrypted using private key of sender. The encrypted message is sent along with the message. Receiver is able to decrypt the hash using public key and comparison of the decrypted hash with calculated hash verifies the message authenticity and integrity.

- In DSS method, the hash is still calculated and is provided as input to the signature function. The other inputs to the signature function are: (i) random number k generated for this particular signature, (ii) sender's private key (Pr_A) and (iii) set of global parameters termed as Pu_G . The output of signature function consists of two values termed as s and r . At receiver side, signature is verified by verification function. The hash of received message is calculated and is provided as input to verification function. The other inputs to verification function are: (i) Signature (s and r), (ii) sender's public key (Pu_A) and (iii) Pu_G . If output produced by verification function is equal to r , the signature of sender is verified. As only the sender is in possession of private key it is guaranteed that only he has put this signature [2] [3].
- The digital signature algorithm (**Figure 8.9**) has three major components:

(1) Key Generation

The signing party (say A) generates key pairs and announces the public key along with other global parameters (Pu_G)

(i) The global parameters are selected as:

- Prime number p is selected such that its length lies between 512 to 1024 bits and is expressed in increments of 64 bits.
- Prime number q that divides $(p-1)$ and is of 160 bit length.
- A primitive element in Z_p is selected as h ($1 < h < (p-1)$)

Then g is evaluated as:

$$g = h^{(p-1)/q} \bmod p$$

(ii) Private key x is selected such $0 < x < q$.

Public key is evaluated as:

$$y = g^x \bmod p$$

(iii) A random per message secret number k is selected such that $0 < k < q$.

(2) Signing

Signature consists of calculating r and s , generated as:

$$r = (g^k \bmod p) \bmod q$$

$$s = [k^{-1} (H(M) + x r)] \bmod q$$

(3) Verifying

(i) r and s are checked by receiver i.e. $0 < r < q$ and $0 < s < q$.

(ii) Digest/hash of message is calculated (Assuming receiver knows the hashing algorithm used by sender).

(iii) Receiver calculates V as

$$V = [(g^{h(M)s^{-1}} y^{rs^{-1}}) \bmod p] \bmod q$$

- (iv) If r is congruent to V , the sign is verified, else rejected.

8.5 SUMMARY

In this unit, an important tool i.e., digital signature for providing authentication, integrity and non-repudiation is dealt with detail. A person using digital signature over a message is authenticated at the receiver via digital signature verification. He cannot later deny that he has not sent the message. To ensure message integrity the message authentication code (MAC) is calculated. The message may additionally be encrypted to ensure secrecy and hence, overall security. This unit provides an understanding of the concepts and requirements of digital signatures. The ways for providing mutual and one way authentication using digital signatures are discussed. The Key Distribution Centre (KDC) based algorithms for symmetric authentication & encryption and also asymmetric authentication & encryption are explained. Towards end you also learn functionality of one of the digital signature standards i.e., Digital Signature Standard (DSS) published by National Institute of Standards and Technology (NIST).

Terminal Questions

1. *Identify the issues in the message authentication.*
2. *Mention properties and requirements of digital signatures.*
3. *Differentiate between:*
 - (a) *direct and arbitrated digital signatures*
 - (b) *use of timestamps and nonces*
4. *Identify the threats against direct arbitrated digital signature schemes.*
5. *What is a replay attack? Show various approaches against it.*
6. *Modify the arbitrated digital signature scheme (Figure 8.1) such that receiver can verify the digital signature.*
7. *If instead of seven steps the above problem 6 has following 5 steps:*
 - (a) $A \rightarrow B$
 - (b) $B \rightarrow KDC$
 - (c) $KDC \rightarrow B$
 - (d) $B \rightarrow A$
 - (e) $A \rightarrow B$

Then show the message at every step (a-e).

8. *Nonces can be used in the following three ways. Discuss the suitability of each.*

- (i) (1) $A \rightarrow B: N_A$
 (2) $B \rightarrow A: E_K(N_A)$
- (ii) (1) $A \rightarrow B: E_K(N_A)$
 (2) $B \rightarrow A: N_A$
- (iii) (1) $A \rightarrow B: E_K(N_A)$
 (2) $B \rightarrow A: E_K(f(N_A))$

References:

1. Overview of cryptography, PKI Outreach Programme (POP) Nationwide Awareness Programme, Centre for Development of Advanced Computing (C-DAC), Electronics City, Bangalore, 2012.
2. Stallings, William, "Chapter 13: Digital Signatures and Authentication Protocols", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
3. Forouzan, Behrouz A., "Chapter 13: Digital Signature", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
4. Needham, R., and Schroeder, M. "Using Encryption for Authentication in Large Networks of Computers." Communications of the ACM, December 1978.
5. Denning, D. "Timestamps in Key Distribution Protocols." Communications of the ACM, August 1981.
6. Denning, D. Cryptography and Data Security. Reading, MA: Addison-Wesley, 1982.
7. Kehne, A.; Schonwalder, J.; and Langendorfer, H. "A Nonce-Based Protocol for Multiple Authentications" Operating Systems Review, October 1992.
8. Woo, T., and Lam, S. "Authentication for Distributed Systems." Computer, January 1992.
9. Woo, T., and Lam, S. "Authentication' Revisited." Computer, April 1992.



**Uttar Pradesh Rajarshi Tandon
Open University**

Master of Computer Science

**MCS -119
INFORMATION AND NETWORK
SECURITY**

Block

3

NETWORK SECURITY APPLICATIONS

Unit 9	169-192
Authentication Application	
Unit 10	193-212
Electronic Mail Security	
Unit 11	213-228
IP Security	
Unit 12	229-252
Web Security	

Course Design Committee

(Prof.) Ashutosh Gupta **Chairman**
Director (In-charge)
School of Computer and Information Science, UPRTOU,
Prayagraj

Prof. R. S. Yadav **Member**
Department of Computer Science and Engineering
MNNIT-Allahabad, Prayagraj

Dr. Marisha **Member**
Assistant Professor (Computer Science),
School of Science, UPRTOU, Prayagraj

Dr. C. K. Singh **Member**
Lecturer
School of Computer and Information Science, UPRTOU, Prayagraj

Course Preparation Committee

Dr Rajeev Singh **Author**
Assistant Professor
Department of Computer Engineering
GB Pant University of Agriculture and Technology
Pantnagar, Uttarakhand

Prof. Abhaya Saxena **Editor**
Head, Dept. of Computer Science
Dev Sanskriti Vishwavidyalaya
Hardwar, Uttarakhand

Prof. Ashutosh Gupta
Director (In-charge),
School of Computer and Information Science
UPRTOU, Prayagraj

Dr. Marisha **Coordinator**
Assistant Professor (Computer Science)
School of Science
UPRTOU, Prayagraj

© UPRTOU, Prayagraj. 2023
ISBN : 978-93-83328-27-7

All Rights are reserved. No part of this work may be reproduced in any form, by mimeograph or any other means, without permission in writing from the Uttar Pradesh Rajarshi Tandon Open University, Prayagraj.

Printed and Published by Vinay Kumar, Registrar, Uttar Pradesh Rajarshi Tandon Open University, 2023.

Printed By: K.C.Printing & Allied Works, Panchwati, Mathura -281003.

UNIT - 9

Authentication Applications

Structure

- 9.0 Introduction
- 9.1 Objectives
- 9.2 Kerberos Motivation and Overview
- 9.3 Authentication Service provided by X.509 Standard
- 9.4 An Introduction of Public Key-Infrastructure (PKI)
- 9.5 Summary
- 9.6 Terminal Questions

9.0 INTRODUCTION

In this unit, two authentication services are described. One - Kerberos and second - X.509 directory authentication service. Former is one of the widely used services for distributed environment that facilitates authenticated communication between clients and servers. Latter provides a uniform digital certificate format (X.509) for containing public keys. For realizing the security environment, public key infrastructure (PKI) is required. PKI is based upon public keys, set of hardware, software, people, policies and procedures. PKI is an important component in asymmetric cryptography and deals with key and digital certificate management. The key PKI components are discussed towards end of the unit.

9.1 OBJECTIVES

This unit deals with two major concepts related with security: Kerberos authentication and X.509 certificate structure. At the end of this unit, you will be able to:

- list the requirements for Kerberos
- understand the full service Kerberos environment
- know the concept of realm in Kerberos
- differentiate between the two versions of Kerberos i.e., version 4 and 5
- know the purpose and format of X.509 certificate structure
- learn about operations related to digital certificates like obtaining of certificate, certificate revocation etc.
- identify the key elements of Public Key Infrastructure (PKI).

9.2 KERBEROS MOTIVATION AND OVERVIEW

- Kerberos draws its name from Greek mythology where a three headed dog safeguards the entry of a gate.
- In similar lines, the Kerberos service is supposed to have three components namely authentication, accounting and audit. It is intended to protect the network gate of an organization [1] [2].
- Out of three components only the authentication is primarily implemented and hence, Kerberos is mainly treated as authentication service.
- Kerberos is developed at MIT (as part of project Athena).
- In Kerberos, an Authentication Server (AS) authenticates users and servers. Thus, users upon authentication by AS can utilize the services of server or server can provide services to authenticated users.
- Kerberos, though a complex protocol, relies on symmetric encryption instead of asymmetric encryption.
- It treats user and system used by user (PC or work station or client machine) differently. A user authentication is considered not system authentication because of the following:
 - A user may pretend as another user while functioning from a system.
 - Impersonation of system is possible by changing network or MAC address.
 - Eavesdropping of messages exchanged between system and server is a possibility.
- Kerberos considers distributed client/server environment where users prove their identity to server by entering correct password and then invokes desired service.
- Servers in turn also prove their identity to communicating clients.
- Authentication service in Kerberos is provided by capable Kerberos servers.
- Kerberos imposes following requirements [1] [2]:
 - It should be secure enough such that finding information for impersonation or any other potential weakness is not possible.
 - Kerberos system should be highly reliable such that upon failure of Kerberos server, another backup server may be used.
 - A user should not bother about the authentication complexities. He should only enter the password on client system.
 - The Kerberos system should be highly scalable and hence, large number of clients and servers can participate.
- The following points are worth mentioning:
 - Kerberos is based upon Needham and Schroeder protocol [3].
 - Both client and server trust Kerberos for mutual authentication.

- The trusted authentication service will be secure provided Kerberos server remains physically secure.
 - Kerberos is also used as Key Distribution Centre (KDC) for distributing keys securely to users.
 - Kerberos is available in two versions i.e. version 4 and version 5.
- In this text Kerberos version 4 is initially discussed (as it is most popular version like IPv4) to build basic understanding and then specifics of version 5 are presented.
- Version 4 utilizes DES as the underlying symmetric encryption protocol.
- Before understanding the message exchanges under Kerberos, let us first understand some of the elements and concepts in Kerberos.
- Key Distribution Centre (KDC): trusted third party who shares secret key with all users (Fig 9.1). This helps in reducing the number of shared keys. Otherwise, for N user communication, keys required are of the $O(N^2)$ i.e., for 6 users $(6(6-1)/2)=15$ shared keys are required whereas using KDC only 6 keys are required.

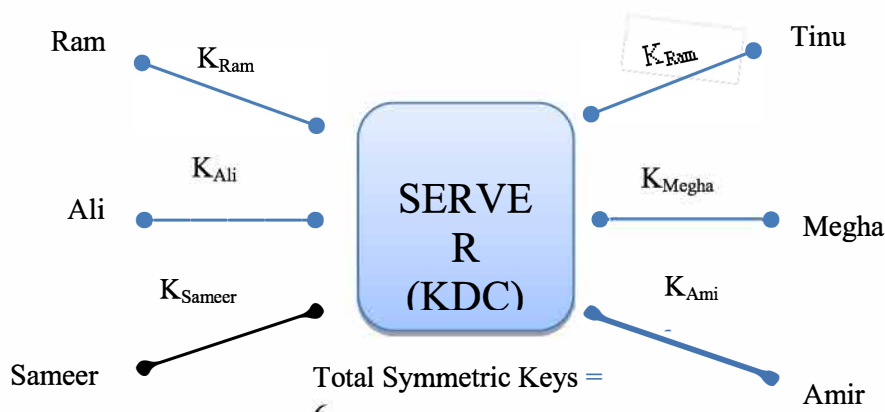


Figure 9.1 Sharing of secret key by KDC with all users

- Session Key: KDC shares secret keys with all users. For communication between any two users, KDC uses their shared keys to authenticate each and to derive a new key (session key) to be used for securing communication between the two users (Figure 9.2). The session key is used for a particular session between the two users and becomes unusable after session expiry.
- Servers: The following three servers are used in Kerberos:
 - a) Authentication Server (AS):

AS plays the role of KDC. All the users register with AS who keep a database of user identities and corresponding passwords. Main functionality of AS is to verify the users via their passwords and to provide them session key to be used ahead between user and Ticket-Granting Server. AS also gives ticket granting tickets to the user.

b) Ticket Granting Server (TGS):

TGS issues service-granting tickets and session key to the users upon request. Service-granting tickets are used to request service from real server while session key is used to protect session between user and real server. In case user wants to contact more than one server for different services, he has to contact TGS more than once for different tickets for different services.

c) Real Server:

Real server provides actual services like Print, FTP, File etc. to the users.

For enhancing clarity, the user is named as Jai while real server is named as Veeru [4]. Entire process involving user (Jai), AS, TGS, and Server (Veeru) is shown in Figure 9.3 [1][2].

- Tickets:

Kerberos offer two kinds of tickets-

(1) Ticket-granting ticket:

Provided by AS to user (Jai) for request and authentication at TGS.

(2) Service-granting ticket:

Provided by TGS to user for request and authentication at real server.

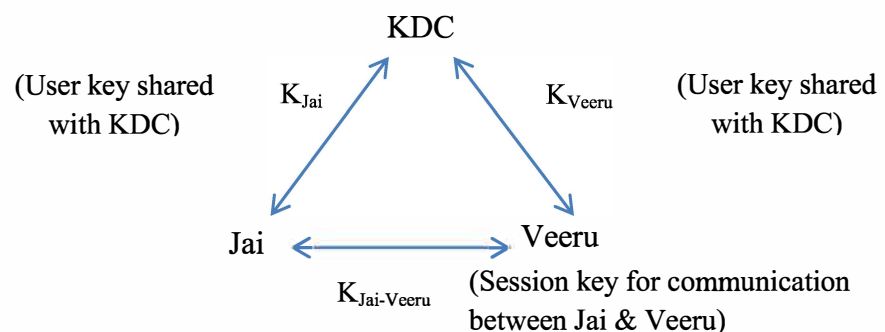
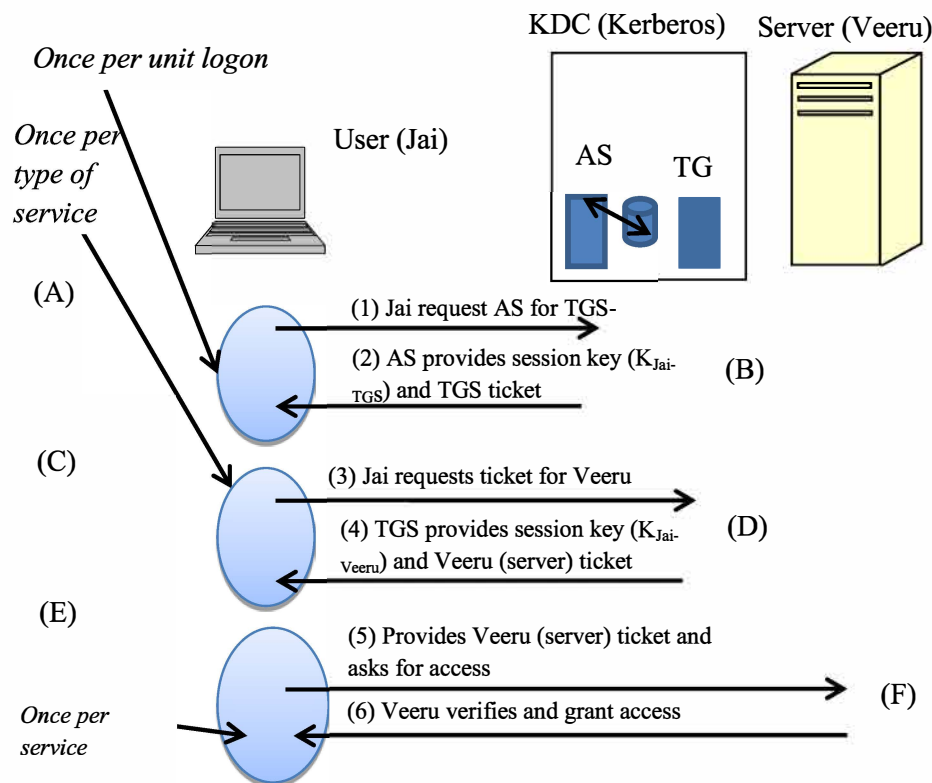


Figure 9.2 Sharing and evolving session key between two users (Jai and Veeru) via KDC



- (A) User logs on and request service
- (B) (i) AS verifies the user request via database.
(ii) It creates session key ($K_{Jai-TGS}$) and ticket granting ticket.
(iii) Derives the encryption key using password and use the key for encrypting data in (ii).
- (C) (i) Client system prompts user for password.
(ii) Derives the decryption key from password & decrypts (2).
(iii) Creates (3) containing ticket-granting ticket & authenticator.
(iv) Sends (3) to TGS
- (D) (i) TGS verifies ticket-granting ticket and authenticator.
(ii) Provides service-granting ticket in reply.
- (E) Client system extracts service granting ticket and sends it to server in (5) along with authenticator.
- (F) (i) Server (Veeru) verifies service-granting ticket and authenticator and accordingly grants access to user.
(ii) In case user (Jai) also requires server authentication, server creates and sends an authenticator.

Figure 9.3 Overview of Kerberos

- Kerberos realm:

Kerberos environment lying under single administrative domain having a Kerberos server (containing Kerberos database), client systems and service providing servers. Thus, all nodes in a realm share the Kerberos database. A copy of read only database exists on backup Kerberos server. A Kerberos

principal with proper privileges is able to make changes or manage the Kerberos database. Nodes under different administrative control (organizations) form different realms. Kerberos provides inter-realm authentication i.e. authenticated users in one realm may demand for service by servers lying in other realm or vice-versa.

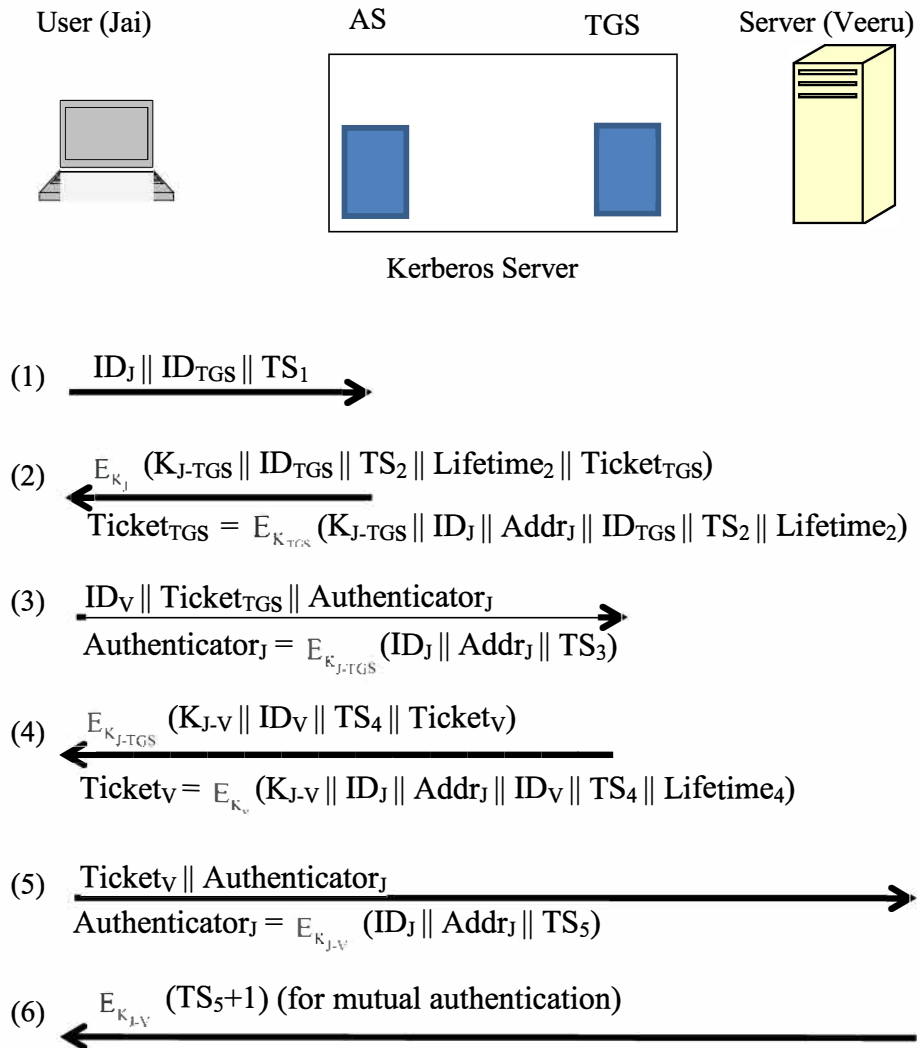


Figure 9.4 Message Exchanges in Kerberos

The functioning of Kerberos protocol is shown in Figure 9.4. The messages exchanged are elaborated in sequence as:

- (1) User (Jai) requests AS by putting his ID and TGS server ID in the request. This means that session key between this user and TGS server needs to be generated. A time stamp (TS_1) is also provided to AS for synchronizing the user clock (Jai's clock) and to prevent replays by attacker.
- (2) AS performs following tasks:
 - i. Verifies user and then searches his password followed by generating a secret key (K_J) based upon this password.
 - ii. Generates a session key between user and TGS server ($K_{J\text{-TGS}}$).

- iii. Creates a ticket ($\text{Ticket}_{\text{TGS}}$) for accessing the ticket granting server.
 - iv. Encrypts the message (2) using K_J . Message contains session key ($K_{J\text{-TGS}}$), ID of TGS server, a fresh time stamp, a fresh lifetime and Ticket ($\text{Ticket}_{\text{TGS}}$). $\text{Ticket}_{\text{TGS}}$ is already encrypted by K_{TGS} and hence can only be decrypted by TGS server. It contains session key for TGS server ($K_{J\text{-TGS}}$), ID & address of Jai, ID of TGS server, timestamp when ticket was created (TS_2) and lifetime indicating validity period of this ticket.
- (3) Jai (client) extracts $\text{Ticket}_{\text{TGS}}$ and sends it along with ID (of server) and an authenticator. The authenticator is evaluated by encrypting ID of Jai, his address and a timestamp (TS_3) using session key $K_{J\text{-TGS}}$. ID_V (ID of Server) is kept so that TGS can issue a certificate to access service of the real server (Veeru).
- (4) TGS server first decrypts ticket using key (K_{TGS}) and validity of ticket is checked via timestamp (TS_2) and lifetime (lifetime_2). From the decrypted contents following is assured:
- a. The communicating peer/user identity. This confirms that this ticket is send by Jai.
 - b. TGS ID presence confirms that the ticket was issued for TGS server.
 - c. Extracted symmetric session key ($K_{J\text{-TGS}}$) is used further to decrypt the Jai's authenticator.

TGS server decrypts the authenticator using $K_{J\text{-TGS}}$ and extracts ID_J , Addr_J and timestamp (TS_3). A match between this extracted ID (ID_J) and ID extracted from ticket at (a) above means that the ticket is indeed send by Jai to whom the ticket ($\text{Ticket}_{\text{TGS}}$) was issued, not by an attacker.

After verification, TGS server generates service granting ticket (Ticket_V) for the user (Jai). The Ticket_V is encrypted using real server key (K_V) and it contains session key between Jai and real server ($K_{J\text{-V}}$), ID & address of Jai (ID_J & Addr_J), ID of real server (for whom Ticket_V is generated), fresh TGS server timestamp (TS_4) and lifetime (lifetime_4) for ticket validity. TGS keeps same $K_{J\text{-V}}$, ID_V , TS_4 along with service-granting ticket (Ticket_V) in message (4). The message (4) is then encrypted using session key ($K_{J\text{-TGS}}$).

- (5) Jai decrypts using session key ($K_{J\text{-TGS}}$), verifies ID_V and timestamp (TS_4), extracts ticket (Ticket_V) and creates message (5) containing Ticket_V and authenticator. Message (5) is send to real server (Veeru). The authenticator is evaluated using session key ($K_{J\text{-V}}$). Upon receipt of message (5), real server (V) authenticates user (Jai) and also ensures that ticket is indeed send by Jai in a similar manner as done by TGS (at (4) above).
- (6) Veeru (server) increments timestamp (TS_5) by one and sends it to Jai (Client) for mutual authentication. The incremented

timestamp (TS_5) is encrypted using session key (K_{J-V}). Upon receipt of message (6), Jai easily authenticates server (Veeru).

Justification/Rationale for various ID_s used:

The role and use of various ID_s used in Kerberos for each message (1-5) is explained in Figure 9.5

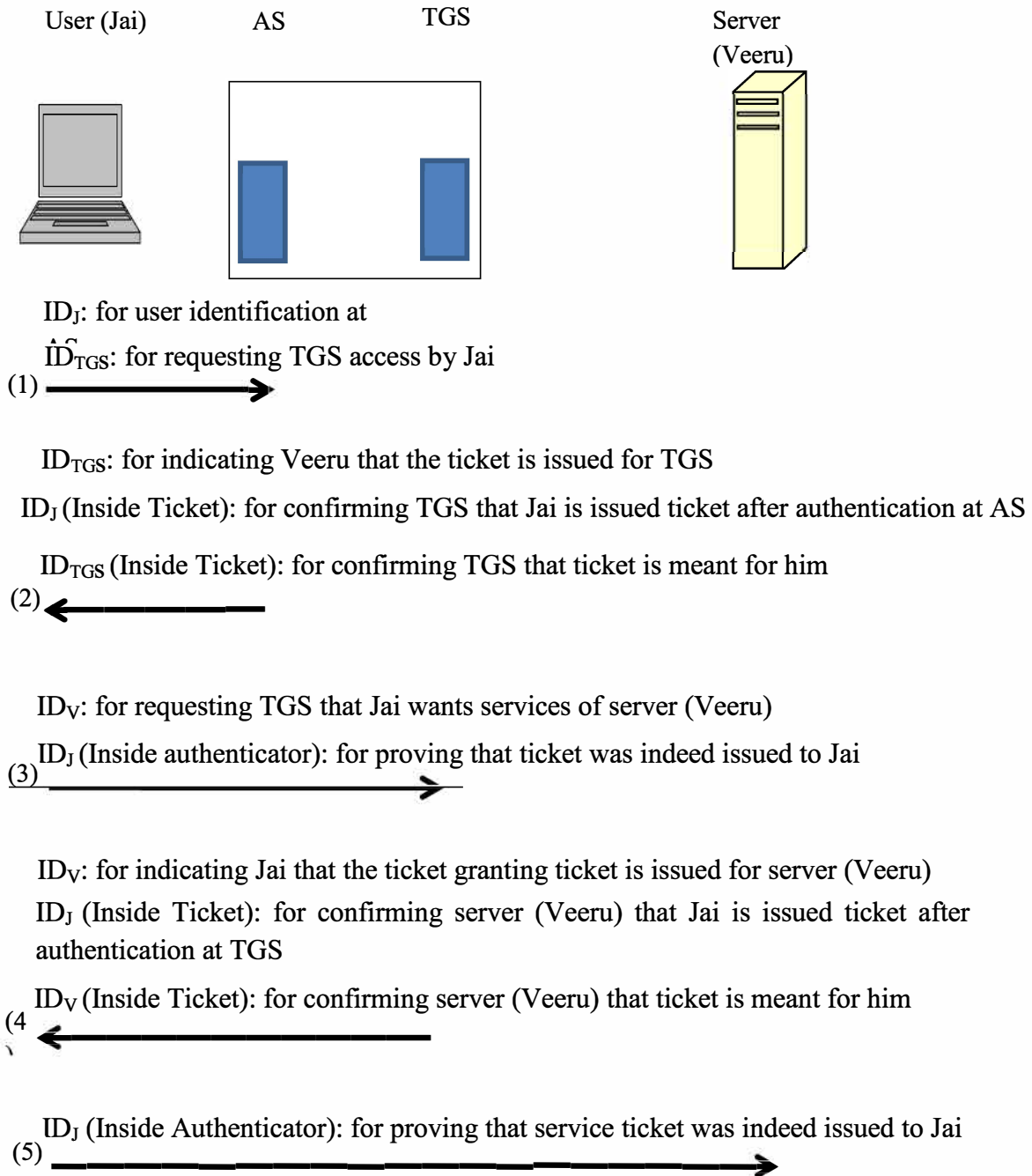
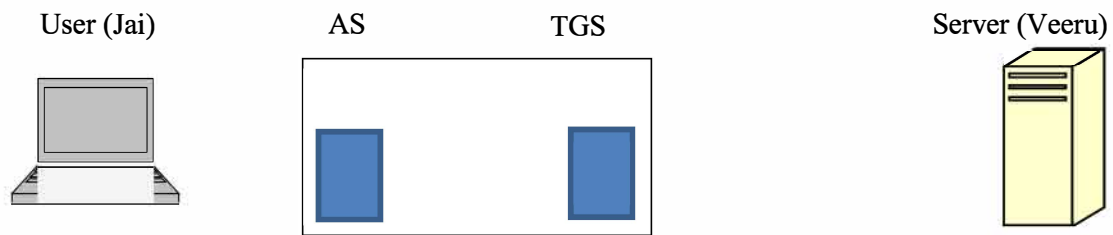


Fig 9.5 Role/Use of various ID_s used in Kerberos

Justification/Rationale for various timestamps and lifetimes

The role and use of various timestamps and lifetimes used in Kerberos for each message (1-6) is explained in Figure 9.6



- (1) TS_1 : for clock synchronization of Jai with AS
- (2) TS_2 : for indicating time of issue of ticket
 $Lifetime_2$: for indicating validity of ticket issued
- (3) TS_3 : for indicating that authenticator is recently evaluated and is not an old one
- (4) TS_4 : for indicating time of issue of ticket to Jai and later to Veeru
- (5) TS_5 : for indicating that authenticator is recently evaluated and is not an old one
- (6) TS_5+1 (Increment TS_5): for indicating that server (Veeru) has received & processed the previous message successfully and this reply is not a replayed one

Fig 9.6 Role/Use of various Timestamps and lifetimes used in Kerberos

Justification/Rationale for various tickets and authenticators

The role and use of various tickets and authenticators used in Kerberos for each message (1-4) is explained in Figure 9.7

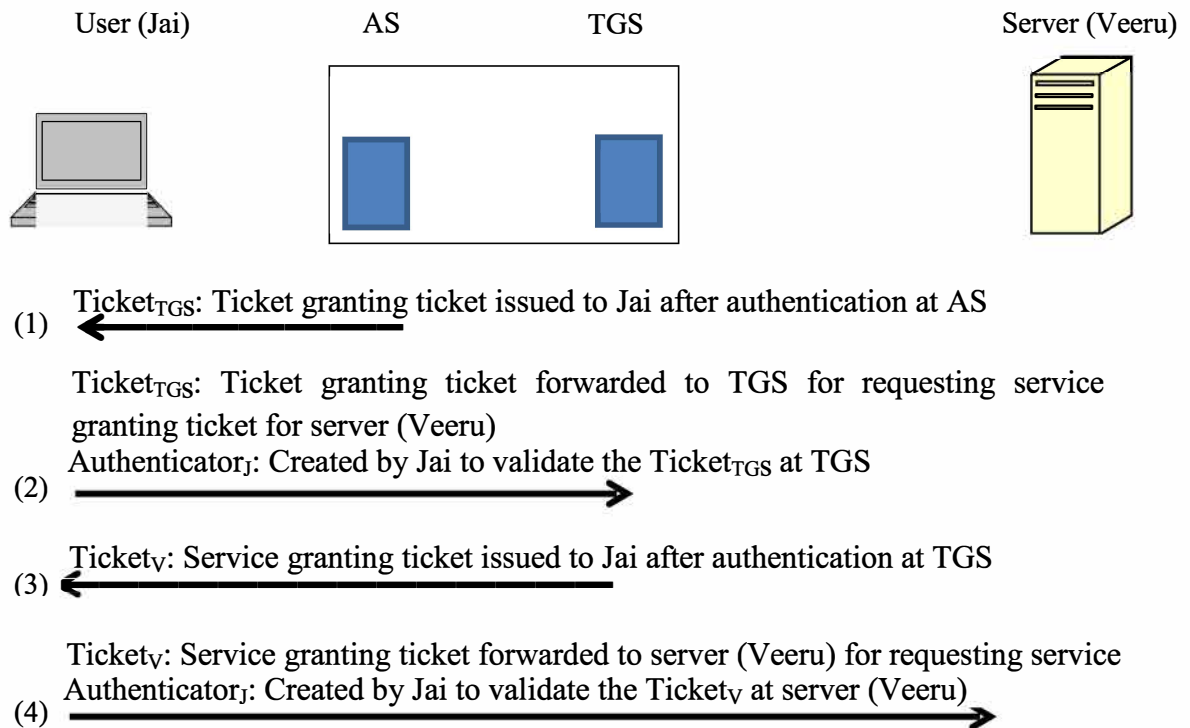


Fig 9.7 Role/Use of various tickets and authenticators used in Kerberos

Justification/Rationale for various keys used:

The role and use of various keys used in Kerberos for each message (1-5) is explained in Figure 9.8

- | | |
|-----|--|
| (1) | K_J : derived using Jai's password and used to protect message (2) send to Jai. |
| (2) | K_{J-TGS} : session key generated by AS and transferred to Jai in message (2) and to TGS via message (3) ($\text{Ticket}_{\text{TGS}}$). Thus, K_{J-TGS} is used to protect message (3 & 4) exchanges between Jai & TGS without requiring a permanent shared secret key. |
| (3) | K_{TGS} : secret key of TGS kept at AS used for securing $\text{Ticket}_{\text{TGS}}$. |
| (4) | K_{J-v} : session key generated by TGS and transferred to Jai in message (4) and to server (Veeru) via message (5) (Ticket_v). Thus, K_{J-v} is used to protect message exchanges (5 & 6) between Jai & server (Veeru) without requiring a permanent shared secret key. |
| (5) | K_v : secret key of server (Veeru) kept at TGS used for securing Ticket_v |

Fig 9.8 Various keys used in Kerberos

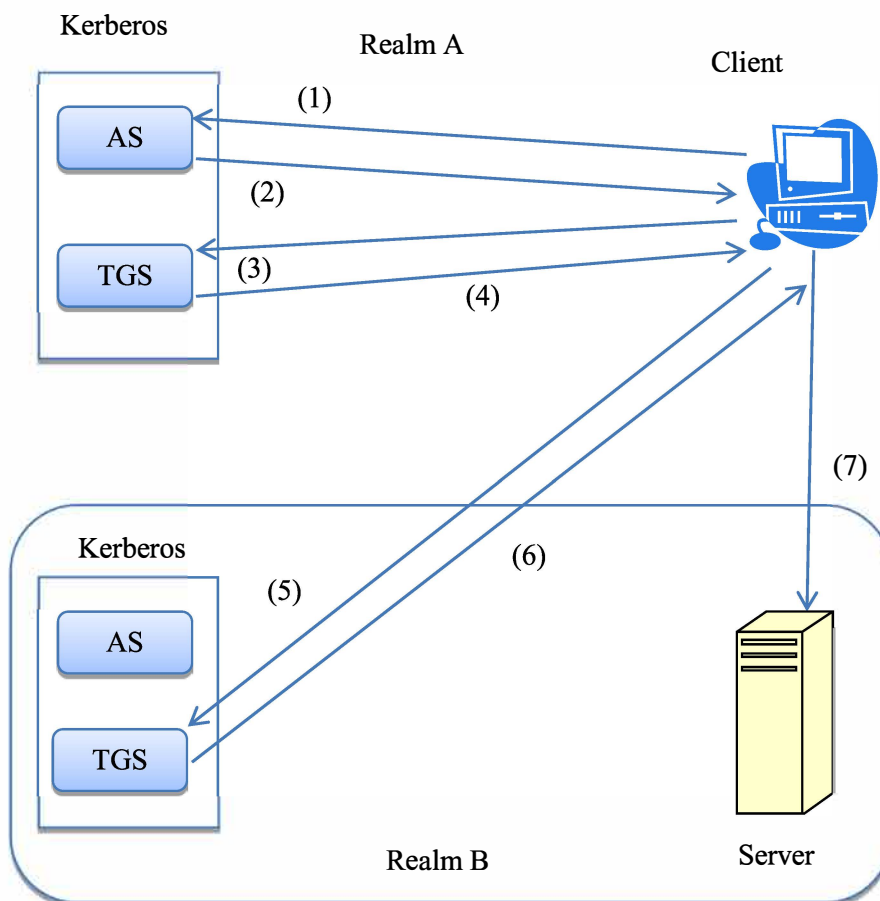
Full service Kerberos environment for inter-realm authentication [1]

The following are the requirements for full service Kerberos environment for inter-realm authentication:

- (1) All users are registered and their ID and hashed passwords are maintained in the database at the Kerberos server.
- (2) All servers are registered and their shared symmetric secret keys are kept both at the Kerberos server and at themselves.
- (3) For supporting inter-realm authentication, the servers lying in different realm are registered with each other and hence share symmetric keys i.e., they both trust each other for authenticating users lying in each other's realm.

The process when a user in one realm tries to access services in other realm is shown in Figure 9.9. The entire process is divided into 4 steps:

1. User in realm A contact AS server for authentication who in turn issues ticket-granting ticket for local TGS server (message 1 & 2).
2. User contacts local TGS server and gets ticket-granting ticket for remote TGS server (message 3 & 4).
3. User contacts remote TGS server who in turn issues service-granting ticket for remote real server (message 5 & 6).
4. User requests remote server using service-granting ticket issued by remote TGS server (message 7).



- (1) Client requests local TGS ticket from Authentication server of Realm A
- (2) Authentication server provides local TGS ticket to client
- (3) Client requests remote TGS ticket from local TGS server of Realm A
- (4) Local TGS server provides remote TGS ticket
- (5) Client requests service granting ticket for remote server from TGS server of Realm B
- (6) TGS server of Realm B provides service granting ticket for remote server
- (7) Client requests remote service using service granting ticket

Fig 9.9 Full service Kerberos environment for inter-realm authentication

Till now Kerberos version 4 is considered, the following subsection highlights features of version 5. Some of the major shortcomings in version 4 are:

- (i) Version 4 makes DES as compulsory element whereas version 5 facilitates use of any encryption method.
- (ii) IPv4 addresses are used in version 4 other address types are not supported. In version 5, this constraint is lifted and hence, any network addresses type can be used.
- (iii) In version 4, byte ordering method (little endian or big endian) is left to the choice of sender. In version 5, unambiguous byte ordering method like Abstract Syntax Notation One (ASN.1) and Basic Encoding Rules (BER) is used.
- (iv) Version 4 has 8 bit lifetime field in unit of 5 minutes which means maximum lifetime can be around $2^8 \times 5 = 1280$ minutes (>21 hours). In version 5, variable lifetime is possible by explicit inclusion of start time and end time.
- (v) Forwarding of credentials of one client to other clients is not done in version 4 whereas it is done in version 5.
- (vi) In version 4, for maintaining interoperability among N realms, N^2 key exchanges are required where as it is less in version 5.
- (vii) Tickets conveyed to users are encrypted twice, first time with secret key of other verifying server and second with user's secret key. This waste computation time.

(a) Authentication Server message interactions for ticket-granting ticket

(b) Ticket-Granting Server message interactions for service-granting ticket

(c) Client/Server Authentication message interactions regarding service

- (2) The AS 'reply' after authenticating user is send as message (2). Message (2) contains some unencrypted information, some encrypted information and a ticket. The ticket in version 5 is encrypted only once and it contains additional information as compared with version 4 i.e., Flags, realm and times. Flags depend upon status of ticket and request options set by user.
- (3) Message (3) is similar to that of version 4 (Message (3)), in addition it has- options, nonce and times field, with each having same functionality as message (1) above.
- (4) Message (4) is same as message (2) with exception that key used for encrypting user information is session key i.e., $K_{Jai-TGS}$.
- (5) The addition in message (5) (version 5) are
 - (i) Option whether mutual authentication is required or not. If required, message (6) is returned.
 - (ii) Sub key in authenticator for indicating choice of encryption key.
 - (iii) Sequence numbers put by server to protect against replays.
- (6) In message (6) unlike version 4, incrementing of time stamp is not done rather sub key & sequence numbers are used. As attacker does not know the encryption key, he cannot forge message (6).

Detailing of few flags used in version 5

- (i) In version 5, AS can directly issue the service-granting ticket. Hence, flag INITIAL is used to indicate the fact that the ticket was issued by AS not by TGS using ticket-granting ticket.
- (ii) PRE-AUTHENT flag implies authentication of client by Kerberos server before issuing ticket e.g., encrypted timestamp based pre authentication.
- (iii) HW-AUTHENT flag require hardware processing by user e.g., continually changing password method used for pre authentication by client for smart card based system.
- (iv) RENEWABLE flag in a ticket means that this ticket can be renewed with a new session time and expiration time.
- (v) A user may have tickets marked as POSTDATED and INVALID. He may request the validation of such tickets by setting the flag MAY-POSTDATE. This method is useful for tasks that needs server for longer times and hence requires tickets periodically.
- (vi) A server can act as proxy for client provided PROXIABLE flag is set by client in request. Under this mechanism the tickets issued by TGS contains different network address and PROXY flag set.

In same lines as in (vi) proxy method, FORWARDABLE flag can be set by client in request such that ticket issued contains different network address and FORWARDED flag set. Thus, a user may

present such tickets directly to a remote TGS server without requiring key sharing between different TGS servers lying in different realm.

Check your progress 1

- a. List the servers and tickets involved in Kerberos?
- b. What role does KDC play in Kerberos?
- c. How many messages are exchanged in Kerberos?

9.3 AUTHENTICATION SERVICE PROVIDED BY X.509 STANDARD

- ITU-T (International Telecommunication Union) recommends X.500 series for keeping and maintaining user's information in a directory (or server).
- X.509 is part of X.500 recommendations. It considers use of public key (or asymmetric) cryptography and digital signatures.
- X.509 assumes presence of X.500 directory (repository) that keeps public key certificates of users.

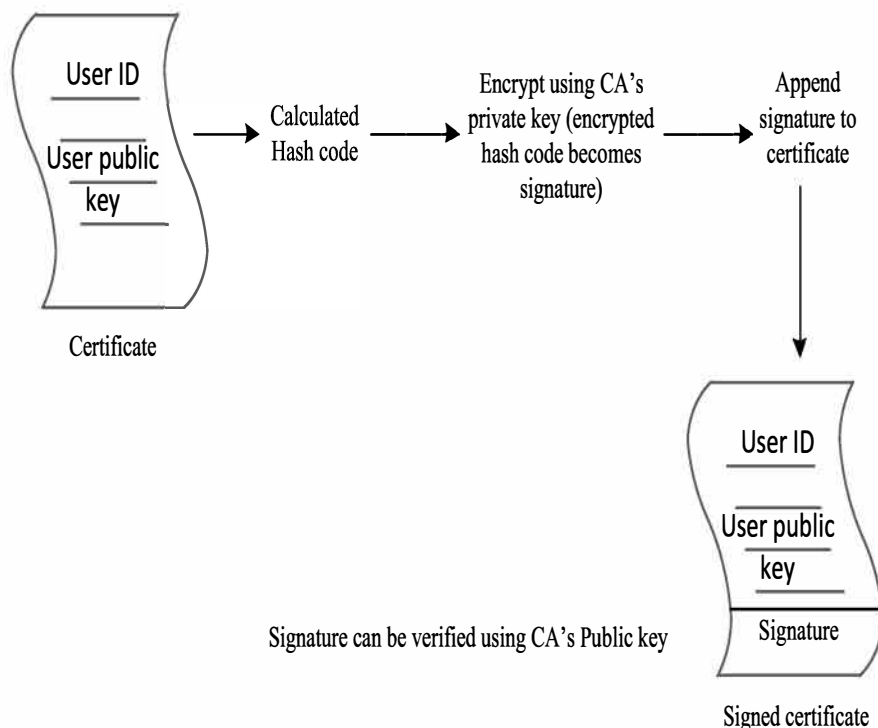


Fig 9.11 Public key certificate generation

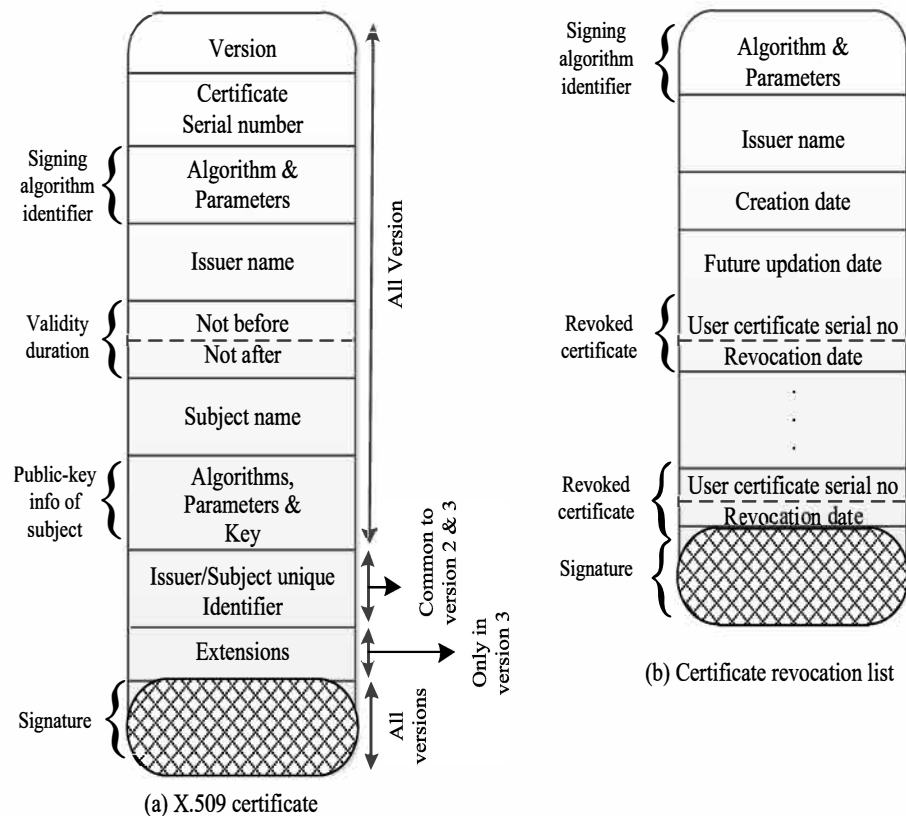


Fig 9.12 X.509 Certificate Format

- A public key certificate of a user contains public key of user digitally signed by private key of trusted third party (termed as certification authority (CA)).
- Based upon public key certificates, X.509 standard (framework) provides authentication protocols for ensuring authentication services.
- The X.509 based authentication protocols are used in many situations and protocols like S/MIME, IP security, SET etc.
- Among several public key algorithms, X.509 recommends RSA.
- The X.509 framework rest heavily upon public key certificates. These certificates are created by Certifying Authority (CA) and placed in the directory. Thus, for sharing a certificate either directory service is used or certificate is distributed by the owner to the communicating user.
- Figure 9.11 – shows steps involved in generation of a public key certificate by CA. The certificate containing user (subject) details along with public key is digitally signed by CA i.e., the hash of certificate is calculated and then encrypted using private key of CA. The digital signatures are appended to certificate (Figure 9.12 shows the general format and

components of X.509 certificate). The components constituting a certificate are as follows [1][2][5]:

- Version – Indicates the version of certificate format. Three versions 1, 2 and 3 are there with version 1 as the default. Two unique identifiers namely issuer unique identifier and subject unique identifier were added in version 2 while extensions were added in version 3.
- Serial number- A unique integer value assigned by CA identifying the certificate. This value remains unique within a CA.
- Signature algorithm identifier- Indicates the signing algorithm along with associated parameters.
- Issuer name- Name of CA that issues the certificate.
- Period of validity – Indicates the certificate validity period (in terms of two dates: Not Before and Not After) in which it can be used.
- Subject name – Certificate owner's name whose public key is contained in the certificate and who has corresponding private key.
- Subject public key information - Owner's public key along with algorithm identifier and associated parameters indicating that this public key can be used with corresponding algorithm.
- Optional unique identifiers for issuer and subject – To uniquely identify the issuer and subject (version 2). Though these are rarely used, they provide reuse of issuer and subject identifiers over time.
- Extensions – Provided in version 3, consists of number of optional extensions where each extension has an identifier, criticality indicator and value for that extension. A criticality indicator with TRUE value implies that the extension cannot be ignored and if its recognition is not possible then the entire certificate is treated as invalid.

These extensions correspond to the following three categories:

- (i) Extensions conveying additional information regarding keys (of both issuer and subject) and certificate policy. Such information includes public key usage indication (e.g., digital signature, key encryption, data encryption etc.), private key

validity period indication, certificate policy listing and mapping.

- (ii) Extensions indicating alternative names and formats of subject and issuer.
 - (iii) Extensions implying constraint specifications for issuing certificates by CA for other CAs i.e., for certificates lying in certification chain.
- Signature – Digital signature (i.e., hash encrypted using CA's private key) created for certificate.
- A digital certificate ensures the following:
 - (i) A user who possesses the CA's public key can verify the certificate and hence public key of the user contained within.
 - (ii) Modification to certificate by a party other than CA is not possible.

Both (i) & (ii) mean that certificates are unforgeable and can be kept in directory as such without protection.

- For obtaining other user's certificate by a user, two cases are there.
 - (i) Case 1: Users lie under same CA – in this case both of them (one who requires certificate, other whose certificate is required) trust same CA and hence can share their certificates directly. They can verify each other's certificate using public key of CA.
 - (ii) Case 2: Users lie under different CAs- in this case each user trust its own CA and is able to verify the certificates using public key of CA.

In case (ii) above, A is able to read the certificate_B (issued by CA₂ to B) but cannot verify it directly due to lack of public key of CA₂.

Thus, A follows following procedure:

- (1) A obtains certificate_{CA2} (CA₁<<CA₂>>) i.e., certificate of CA₂ issued by CA₁ from the directory. As A has public key of CA₁, from this certificate A can verify and extract public key of CA₂.
- (2) Using this public key A can now verify the certificate_B (CA₂<>) issued by CA₂ and can extract public key of B.

X.509 expresses this chain as

CA₁<<CA₂>>CA₂<>**

Note: ** notation $Y\langle\langle X \rangle\rangle$ implies certificate X issued by CA Y

Similarly, B can obtain public key of A (step 3 & 4 (Figure 9.13))

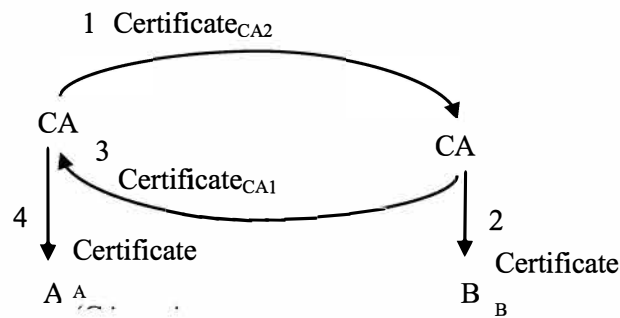


Figure 9.13 Obtaining of digital certificate by users

- In general, for N CAs certifying chain can be expressed as:
 $CA_1\langle\langle CA_2 \rangle\rangle CA_2\langle\langle CA_3 \rangle\rangle \dots CA_N\langle\langle B \rangle\rangle$
- Each of CA (CA_i) creates certificates for each other and these certificates lie in the directory from where user can learn the linkages.
- X.509 favors hierarchical arrangement of CAs for better navigation in certifying chain.
- In Figure 9.14 user R can obtain the certificates from directory *** to build the certifying chain (path) to U as:
 $P\langle\langle L \rangle\rangle L\langle\langle K \rangle\rangle K\langle\langle M \rangle\rangle M\langle\langle Q \rangle\rangle Q\langle\langle U \rangle\rangle$
 Thus, R can unwind this sequence to obtain public key of U

*** U can also provide these certificates to R in initial message exchanges.

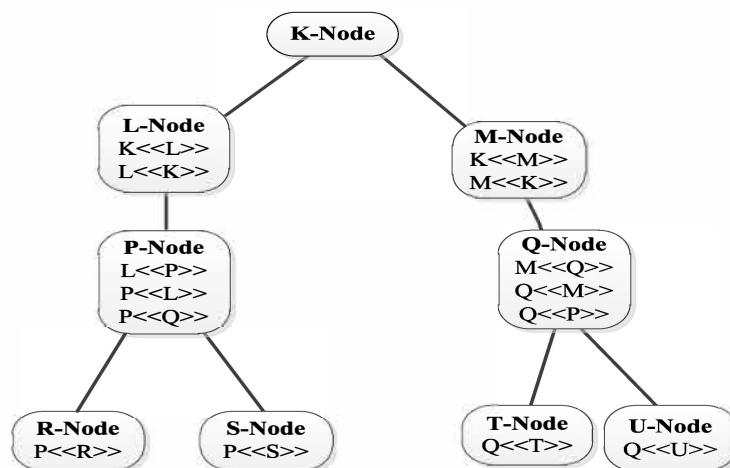


Fig 9.14 An example X.509 Certificate Hierarchy

- Certificates may be revoked before their expiry by the issuing CA on the following grounds:
 - (i) Compromise of private key of a user has been identified.
 - (ii) The CA no more certifies the user
 - (iii) CA certificate itself is compromised
- For all such revoked certificates, CA maintains a list termed as Certificate Revocation List (CRL)
- Such lists are maintained CA wise in the directory.
- Each CRL contains (Figure 9.12 (b)) name of issuer, list creation time, next scheduled date for posting CRL and revoked certificate details (as in this list entry).
- The revoked certificate details include certificate serial number and revocation date.
- Thus, a user trying to obtain a user's certificate, first looks it into CRLs in the directory for identifying whether it is valid or revoked. The user may also maintain certificates and CRLs into cache for decreasing the search and access time.

Authentication procedures

- X.509 authentication procedures utilize digital signatures.
- For creating digital signature each party obtains public key of other from the certificate.
- The certificates are shared either via initial messages or via shared directory.
- Three authentication procedures exist in X.509 standard; either of these may be used depending upon requirements.
- These authentication procedures are termed as one way, two way and three way authentication (Figure 9.15).
- One way authentication involves only one message (from sender to receiver). This message ensures:
 - (i) Message generation was done by sender only.
 - (ii) Message is intended for receiver.
 - (iii) Message is not a replay of previous message and its contents are unforgeable.
- This message proves sender identity at the receiver. It has following elements.
 - (i) Timestamp (t_S): generation time and expiration time of message; used for preventing delivery of old messages.
 - (ii) Nonce (r_S): a unique random value (can be stored at receiver during a particular message expiration period) used to prevent replay attacks.
 - (iii) Receiver Identity (ID_R): Indicates that message is intended for R
 - (iv) Sigs: digital signature of S, ensuring integrity of message and authenticity of sender.

- (v) Encrypted session key (K_{SR}): message may also be used to transfer the shared secret session key (K_{SR}) to R after encrypting using public key of R (P_R)
- Two way authentication has same message from sender to receiver and in addition has reply message containing the nonce received from sender (r_S), timestamp & nonce generated by R, signature of R and encrypted session key. Thus, in two way authentication R is also authenticated at S using reply and hence both S&R are able to verify each others identify.
- Three way authentication involves a third message (in addition to first and second message) that is send by S to R and contains signed copy of nonce r_R . This ensures echo of receiver nonce back to R and proves useful under replay attacks on R.

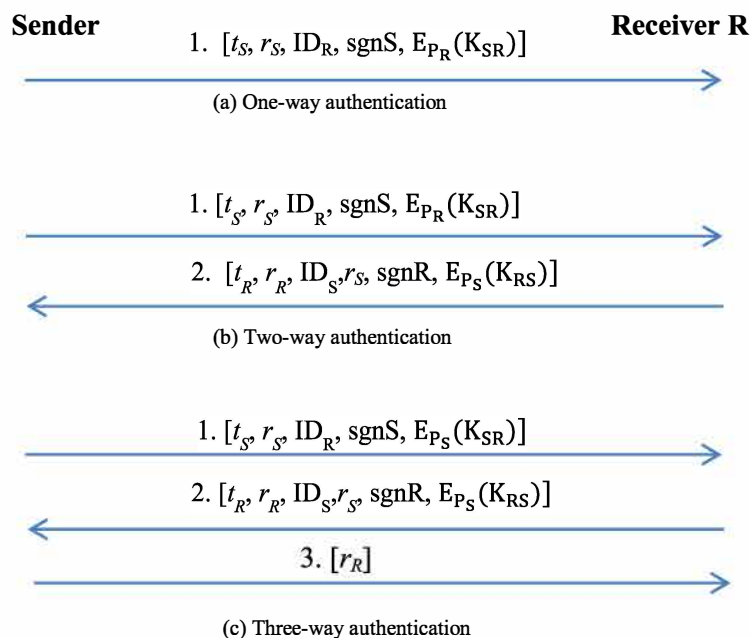


Fig 9.15 X.509 Authentication Procedures

9.4 AN INTRODUCTION OF PUBLIC KEY INFRASTRUCTURE (PKI)

Public key infrastructure (PKI) refers to set of hardware, software, people, policies and procedures. Together they are used for managing and storing digital certificates. It is defined by Internet Engineering Task Force (IETF) working group referred as Public Key Infrastructure X.509 (PKIX). PKIX (Figure 9.16) is based upon X.509 format. The key elements of this model are [1][5]:

- (1) End users & devices (end entities) that support PKI related services.
- (2) Certificate Authority (CA) who issues the certificates & certificate revocation lists (CRLs) along with supporting other administrative functions.
- (3) Registration authority (optional component) who registers an end entity.

- (4) CRL issuer (optional component) who may be delegated by CA to publish CRLs.
- (5) Repository: that supports certificates and CRLs storage and retrieval.

The following are management functions supported by PKIX model:

- Registration: refers to mutual (initial) authentication between user and CA. it is often refer to as enrollment where end entity is issued secret key for subsequent authentication.
- Initialization: installation of keying material on client system.
- Certification: issuing of public key/digital certificate to user by CA.
- Cross certification: issuing of digital certificate by one CA to another CA. the issued certificates contains signature key to be used by CA for further issuing certificates.

Apart from these functions key pair recovery for restoring the key pair from back facility, key pair updation and new certificate issuing for any change, revoking certificate under compromise are also supported PKIX.

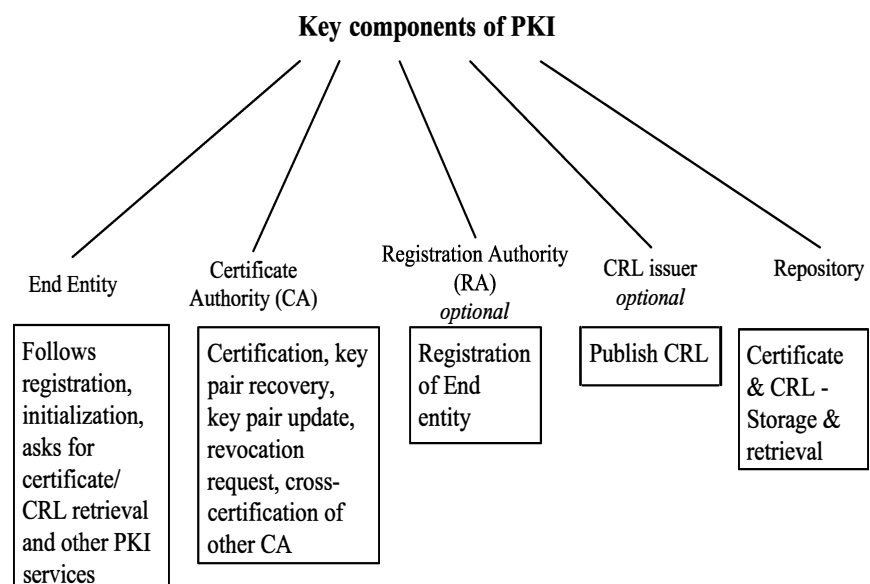


Fig 9.16 X.509 PKI Infrastructure model

Check your progress 2

- a. What is the responsibility of a certification authority?
- b. Why is certification of Digital signature by an independent authority required?

9.5 Summary

In this unit, two important public key based authentication functions i.e., Kerberos authentication and X.509 are presented. Kerberos, an authentication service for distributed environment ensures authenticated communication between client and server via third party authentication server. Kerberos working environment along with the notion of realm is discussed. Public Key Infrastructure (PKI) is a very popular framework for working with digital signatures. PKI uses digital certificates containing public key of users. The digital certificates are structured as per X.509 certificate standard. The various fields of X.509 certificate along with operations like certificate acquisition, certificate revocation etc. are elaborated. This enhances your skills set and knowledge in the domain of public keys and certificates. The X.509 certificate standard is also used by other applications like S/MIME.

Terminal Questions

1. *State the four requirements for Kerberos.*
2. *What is realm? When is it used?*
3. *List the major differences between Kerberos version 4 and 5*
4. *Why is X.509 used?*
5. *What do you understand by certificate hierarchy?*
6. *Discuss the process of obtaining and revoking a digital certificate.*

References:

1. Stallings, William, “Chapter 14: Authentication Applications”, Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
2. Forouzan, Behrouz A., “Chapter 15: Key Management”, Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
3. Needham, R., and Schroeder, M. “Using Encryption for Authentication in Large Networks of Computers.” Communications of the ACM, December 1978.
4. Overview of cryptography, PKI Outreach Programme (POP) Nationwide Awareness Programme, Centre for Development of Advanced Computing (C-DAC), Electronics City, Bangalore, 2012.
5. Public Key Infrastructure Components, PKI Outreach Programme (POP) Nationwide Awareness Programme, Centre for Development of Advanced Computing (C-DAC), Electronics City, Bangalore, 2012.

UNIT - 10

Electronic Mail Security

Structure

- 10.0 Introduction
- 10.1 Objectives
- 10.2 Notations Related with PGP
- 10.3 PGP Detailed Study
- 10.4 PGP Functionality
- 10.5 Fundamentals of S/MIME
- 10.6 Summary
- 10.7 Terminal Questions

10.0 INTRODUCTION

Electronic Mail (E-mail) is one of the most popular and widely used Internet applications. Users using E-mail want to ensure authentication, confidentiality and non-repudiation services. In this unit two mechanisms that provide such services are discussed in details. First is Pretty Good Privacy (PGP) mechanism that is implemented in an open source software package and is available freely to the users/programmers/developers. PGP uses digital signatures to provide authentication; symmetric encryption to provide confidentiality; ZIP mechanism to provide compression functionality; radix-64 encoding to provide E-mail compatibility; segmentation to manage long messages. Second is Secure Multipurpose Internet Mail Extension (S/MIME), an Internet standard which provides security services to MIME based mail exchanges.

10.1 OBJECTIVES

A lot of demand regarding securing E-mails is raised by the Internet users. This Unit deals with two methods namely, Pretty Good Privacy (PGP) and Secure Multipurpose Internet Mail Extension (S/MIME) for providing security to E-mails. The objectives are to:

- Define the terms and the concepts of Pretty Good Privacy (PGP)
- Introduce the notion of public and private key rings in PGP
- Learn the intricacies of PGP trust model
- Lay the foundation for study of Secure Multipurpose Internet Mail Extension (S/MIME)

- Describe the various S/MIME message types

10.2 NOTATIONS RELATED WITH PGP

The PGP developer, Phil Zimmermann considered utilization and integration of best available cryptographic methods and functions like 'RSA, DSS, Diffie-Hellman' for public key operations, 'CAST-128, IDEA, 3DES' for symmetric operations and 'SHA-1' for hashing - together to provide confidentiality and authentication for E-mail system. The developed system and its documentation along with source code are available freely and can run on different platforms like Windows, UNIX, Macintosh etc. PGP is available as RFC 3156. Table 10.1 shows notations and symbols used in PGP [1-5].

Table 10.1

Symbol	Meaning
K_S	Symmetric Encryption Key
K_A	Private Key User A
P_A	Public Key User A
E_P	Public key/Asymmetric Encryption (done either by private key or public Key)
D_P	Public key/Asymmetric Decryption (done either by private key or public Key)
E_C	Encryption Using Symmetric Key
D_C	Decryption Using Symmetric Key
 	Concatenation
Z	ZIP algorithm used for compression
R64	Radix 64 ASCII format conversion

In the following subsections PGP details related to services provided, requirements, message formats, operations and key management are discussed.

10.3 PGP DETAILED STUDY

PGP provides 5 services, these are [3-5]:

- I. Authentication service
 - ❖ Provided by evaluating and verifying the digital signatures of the message.
 - ❖ DSS or RSA signature algorithm is used, first hash of message is evaluated (using SHA-1) which is then encrypted (by RSA algorithm) using private key of sender. The resulting code becomes digital signature and is appended to the message. The entire process is shown in Figure 10.1 a.
 - ❖ The recipient uses sender's public key to decrypt the hash code. This hash code is verified with that calculated from message. Upon verification, message authentication by sender is ensured.

- ❖ The PGP authentication strength depends upon RSA/SHA-1 or DSS/SHA-1. Both provide effective PGP digital signatures.
- ❖ PGP also supports detached signatures which are stored and maintained independent of message. Such signatures are useful in following cases:
 - a) For maintaining a separate signature log.
 - b) For detecting virus infection of an executable program.
 - c) For signing a document (e.g. legal contract) by more than one party. In this case signature of each party has to be applied separately to the document i.e. signing of signatures of other parties (nesting of signatures) is not required.

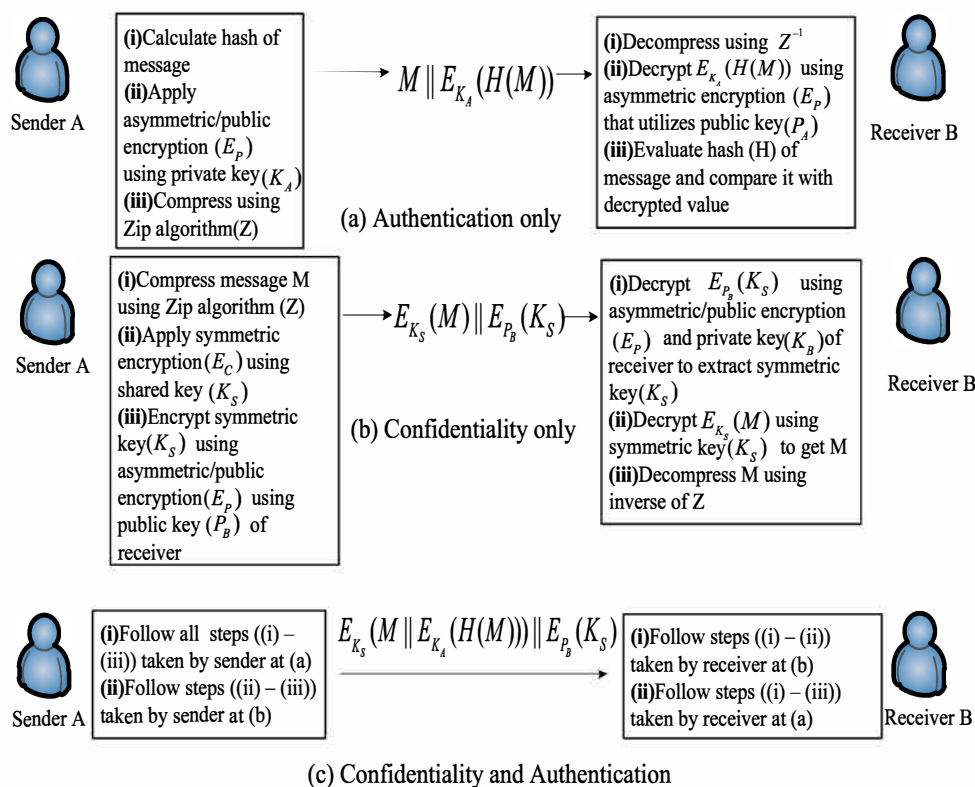


Figure 10.1 Cryptographic Functionality in PGP

II. Confidentiality service

- Provided by encrypting message (Figure 10.1b) using symmetric algorithms like CAST-128, IDEA, 3DES.
- A 128 bit random session key is used for encrypting the message. A fresh new and different session key (one time key) is generated for each message. Thus, session key is bound with the message and is send along with the message in encrypted form. Use of separate random session keys for messages ensures that small amount of content is encrypted with each key and also no relationship exists between different keys.

- The session key is sent after encrypting it (protecting it) with public key of the receiver.
- Upon receipt, first the session key is decrypted using private key of the receiver. The extracted session key is then used to decrypt the message.
- For encrypting/decrypting session key ElGamal Diffie-Hellman method is used.
- Use of public key encryption implies elimination of separate session key exchange mechanism and handshake mechanism.
- PGP uses combination of symmetric encryption/decryption for message and asymmetric encryption/decryption for session key. This tends to enhance the speed in comparison with asymmetric encryption/decryption for the message.

III. Confidentiality and authentication services

For providing both these services together (Figure 10.1 c) PGP adopts the following:

- a) Digital signature of the message is evaluated by sender using its private key. It is then added to the message. Hence, message and corresponding signatures may be stored for future use.
- b) The message and signature are later encrypted using symmetric shared session key.
- c) The session key itself is encrypted using receiver's public key and is appended to the message. For verifying, first session key is decrypted which is then followed by message decryption and signature verification.

IV. Compression mechanism

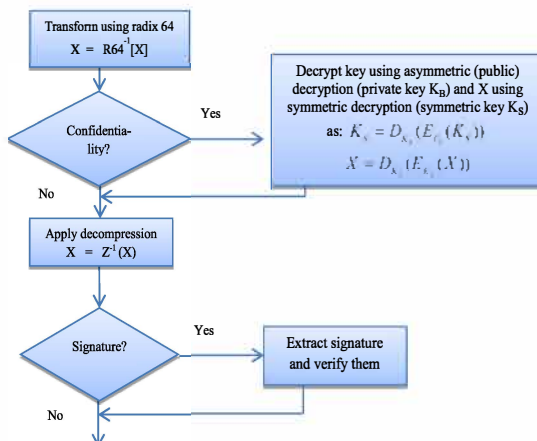
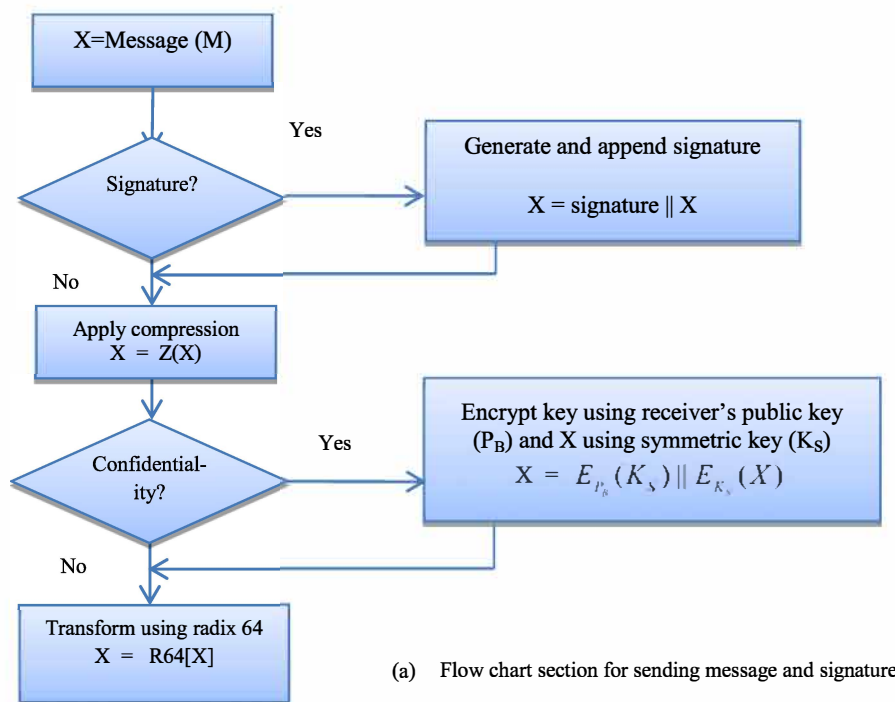
PGP uses compression mechanism for space efficiency reasons i.e., saving file storage cost and space during communication. The compression is applied after signature attachment and before message encryption. The compression algorithm is ZIP and is shown in Figure 10.2 as Z (decompression as Z^{-1}).

- Compression is done after signature as this ensures storing of plaintext message along with signature for future use and also eliminates the need of recompressing the message again at time of verification.
- The PGP compression algorithm is not deterministic and hence different implementations of it produce different results. Hence, if digital signature is applied after compression (by a particular compression algorithm) then it restricts use of same version of algorithm for decompression.
- Message compression produces less redundant message, the cryptanalysis of such compressed message becomes tougher. This supports the application of encryption in PGP after application of compression.

V. Compatibility regarding E-mail

- ❖ PGP uses radix-64 conversion scheme to map binary data (in groups of three) to ASCII characters (four characters).
- ❖ Though it seems that radix-64 will expand the message in size but keeping in view the effect of compression before application of radix-64 ensures that the overall effect is that of message compression.
- ❖ Application of radix-64 makes the result as unreadable and hence provides some preliminary level of confidentiality.

Figure 10.2 shows application of signature, compression, confidentiality and radix-64 to a message in sequence. It can be seen that signature & confidentiality are optional whereas compression and radix-64 are compulsory in PGP oriented message transmission and reception.



(b) Flow chart section for message and signature verification

Figure 10.2 Sending and Receiving of PGP Messages

VI. Segmentation and reassembly service

In internet based E-mail systems, maximum permissible message length is 50,000 octets and therefore the messages larger than this limit are broken into smaller segments and later reassembled at the destination (after stripping the headers). The data pertaining to session key and digital signature are always kept in the first segment at the start.

10.3.2 PGP requirements for keys

PGP involves following keys:

- Symmetric key for sessions
- Asymmetric key for sessions
- Passphrase based symmetric keys

Accordingly, PGP imposes following requirements on these keys:

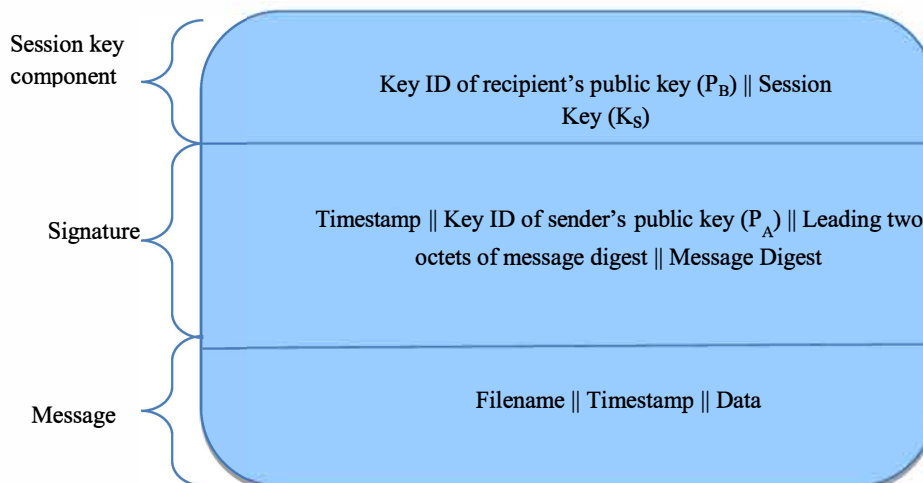
- I. The session keys generated are neither related (i.e. random) and not predictable. For meeting this requirement PGP uses CAST-128 encryption algorithm for generating random keys. Input to this algorithm is taken as 128 bit previous session key and two 64- bit random plaintext blocks generated by 128 bit random stream.
- II. A user may have multiple asymmetric key pair. This may be due to user requirement of changing the key with time or due to group interaction requirement. Thus, when a user protects the session key with the selected public key of receiver, he must indicate this to the receiver. This can be ensured by either transferring the public key itself with the message or by using an identifier associated with public key. Former is impractical as public key occupies large space. Latter is practical and therefore adopted by PGP. In this, a unique key ID (64 bits selected as least significant 64 bits of public key) is assigned per user ID by PGP. Mathematically, key ID for a public key P is $P \bmod 2^{64}$. A similar key ID (64-bits) is also required while signing a message using private key and an intimation of the corresponding public key via this key ID is provided to the receiver.
- III. The keys along with their IDs are required to be stored and managed for efficient use. For meeting this requirement, PGP defines two data structures namely private-key ring and public-key ring for use by all users. These key rings are maintained as table where each row contains information regarding public/private key pairs. Private-key ring of a user keeps public/private key pairs of this user only while public-key ring keeps public/private key pairs of other users that interact with this user.

10.3.3 PGP message format and key ring structure

Figure 10.3 shows the general format of a PGP message transmitted. The message has three parts - for session key, signature and message itself. The session key part and signature part are optional [3].

- Message part contains filename, time of signature (timestamp) and data.
- Signature part contains encrypted message digest of 160 bits (encrypted using private key of sender), public key ID corresponding to private key used for encryption (this public key ID helps receiver to find the correct public key for decryption) and leading two octets of message digest. These octets are compared with two octets of message digest decrypted by the receiver and hence to verify that the receiver used the correct public key for decryption.
- Session key part has session key encrypted using receiver's public key along with the public key ID of the receiver used for message encryption.
- The entire message is encoded with radix-64 encoding.

Content



Asymmetric/Public encryption is applied on K_S using B's public key

Asymmetric/Public encryption is applied on Message Digest using A's private key

Message + Signature is encrypted using K_S

Zip compression is applied on Message + Signature

Radix-64 conversion is applied on Message + Signature + Session key component

Private and public key ring

Figure 10.4 shows values corresponding to a single row of private key ring. It has timestamp indicating the time of generation of user's key pair, key ID for public key, public key, private key (encrypted) and user ID (E-mail ID, say). The private key ring has several such rows. The entire private key ring/table is indexed using user ID or key ID. The user's key is kept by encrypted using CAST-DS/IDEA/3DES. The encryption key used in encryption is hash of the passphrase selected secretly by the user. The passphrase is not stored rather

discarded after use. During private key retrieval from ring, the user is again asked for passphrase. The hash of this passphrase is used as decryption key to decrypt the private key. Figure 10.5 shows values corresponding to a single row of the public key ring. It contains timestamp, key ID of public key, public key, user ID of owner of key. Other fields are explained in the next section [3][4].

Attribute	Value
Timestamp	T_i
Key ID	$P_i \bmod 2^{64}$
Public Key	P_i
Encrypted Private Key	$E(K_i H(\text{PassPhrase}_i))$
User ID	User i

Figure 10.4 Values corresponding to a single row of Private-Key Ring

Attribute	Value
Timestamp	T_i
Key ID	$P_i \bmod 2^{64}$
Public Key	P_i
Owner Trust	Trust_flag $_i$
User ID	User i
Key Legitimacy	Trust_flag $_i$
Signature (S)	User_Signature
Signature Trust (S)	Signature_Trust_Value

Note: Key ID, User ID – Indexing fields

Figure 10.5 Values corresponding to a single row of Public-Key Ring

10.4 PGP FUNCTIONALITY

PGP uses private and public key rings during message generation and message verification. Ignoring compression and radix-64 conversions, the steps by sender and by receiver are performed as:

PGP Sender-

PGP sender signs (steps (I) & (II)) and encrypts (steps (III)-(V)) the message and session key (Figure 10.6).

- I. For signing, the sender retrieves his private key from private-key ring using user's passphrase.
- II. Sender calculates hash and digital signature of the message.
- III. Session key is generated and using it, message along with signature is encrypted and stored in PGP message.
- IV. PGP retrieves the receiver's public key ID based upon user ID as index from the public key ring.
- V. Using the corresponding public key, the session key is encrypted and encrypted session key is also stored in the PGP message along with public key ID (together termed as session part of PGP message).

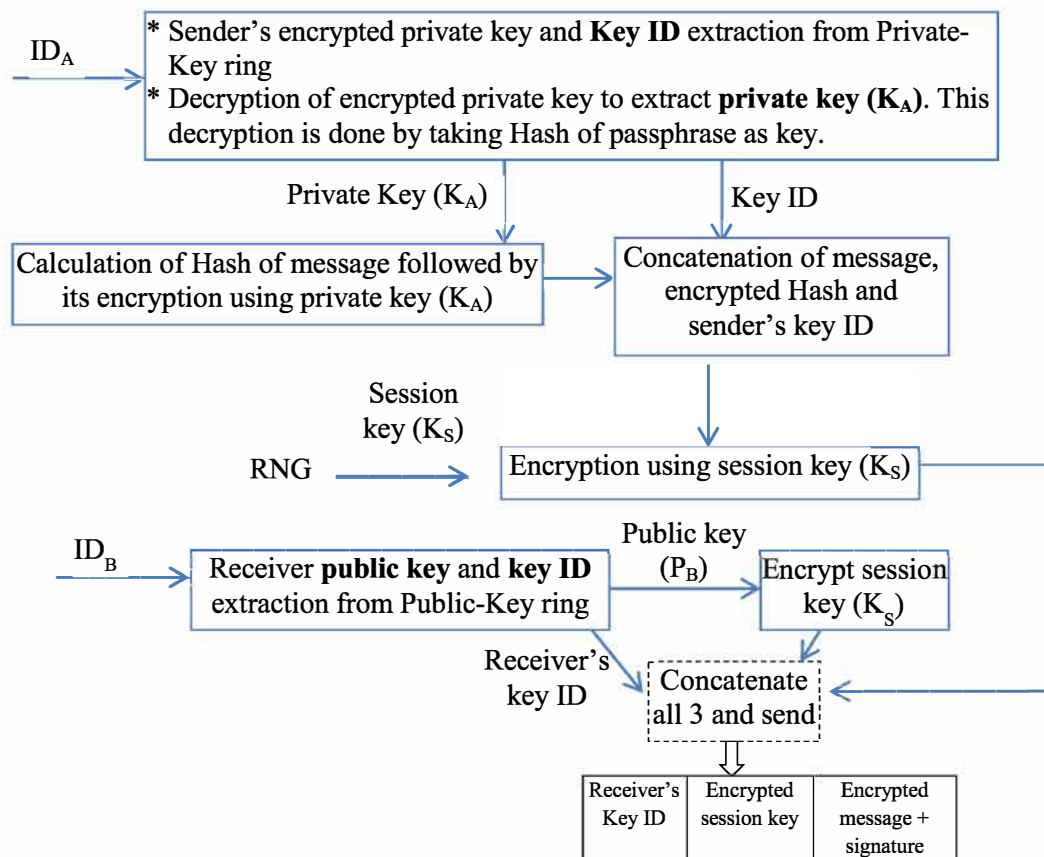


Figure 10.6 Generation of PGP Message at sender A without using compression or radix 64 conversion

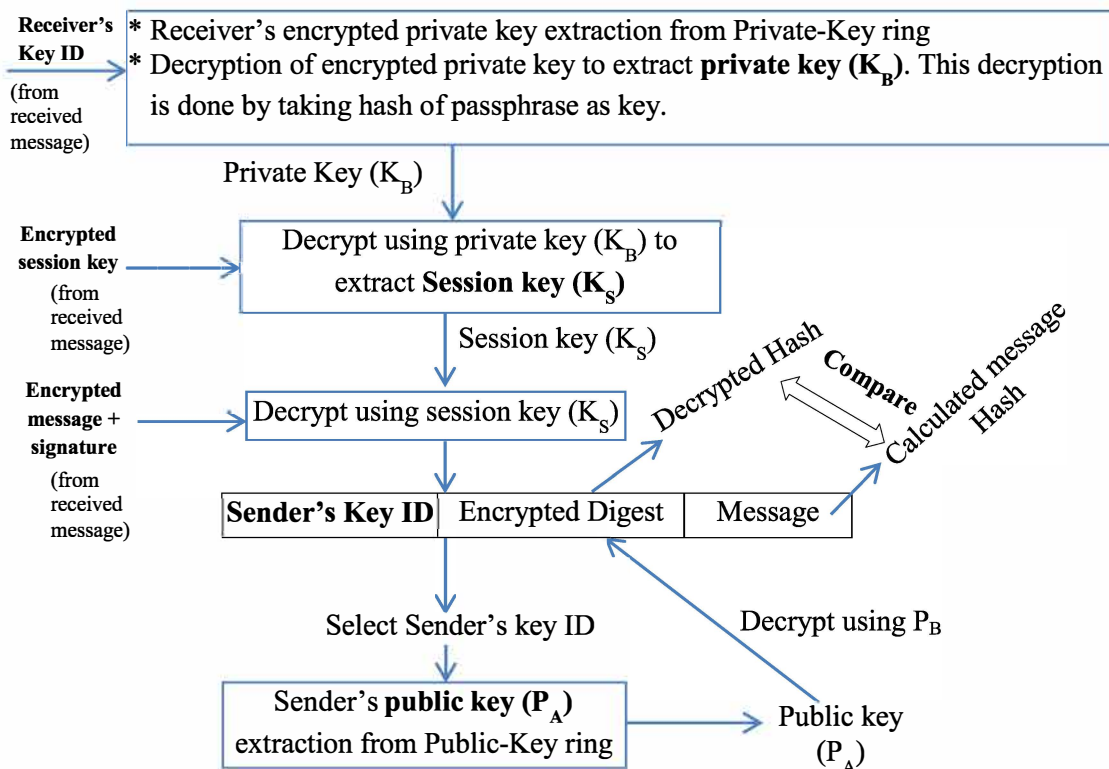


Figure 10.7 Processing of PGP Message at receiver B without using compression or radix 64 conversion

PGP Receiver-

PGP receiver decrypts (steps (I) - (III)) and verifies (steps (IV) - (VI)) the PGP message. (Figure 10.7)

- I. For decrypting, the PGP receiver extracts key ID field from session key part of the received message. Based upon this key ID, receiver finds the encrypted private key in its private-key ring.
- II. The encrypted private key is decrypted using hash code of the passphrase as key in the encryption algorithm.
- III. Using private key, session key is obtained after decryption.
- IV. Receiver extracts the key ID from signature key part of the message. Accordingly searches the public key ring for public key of the sender.
- V. Receiver decrypts the message digest using sender's public key.
- VI. Receiver compares the extracted message digest with the calculated message digest.

Managing Public Key

The user's public ring should have genuine or valid public keys. If a public key kept in ring gets compromised, the associated risk gets increased. Encrypting a message by such key implies that it can be read by an attacker. Also, chance of forgery of signature gets increased. Thus, the public keys should have an associated trust value that indicates that key is still valid and can be used. For obtaining a public key reliably either transfer via electronic means or verification methods like telephone or E-mail verification may be used. As numbers of such users may be large, use of such systems for key distribution are not much practical. PGP supports following two methods:

- I. Obtain other user's public key through mutual trusted individual who creates and signs the certificate.
- II. Obtain other user's public key through trusted third party (Certifying Authority (CA)).

The user contacts CA for public key certificate; created and signed by CA.

PGP Trust Associations

- I. **Key legitimate field (Calculated by PGP)** => Indicates binding between user ID and public key. Higher value means valid public key for this particular user ID.
- II. **Signature trust field (Calculated by PGP)** => Associated with each signature. Zero or more signatures are possible. Indicates PGP user's trust for signer to certify the public keys.
- III. **Owner trust field (Assigned by this public-key ring user)** => Degree to which this public key can be trusted to sign other certificates.

For new ring entry trust values are calculated as follows:

First, owner trust value is assigned to trust flag byte (OWNER TRUST) in the public key ring. The following trust values are possible:

Ultimate, unknown, untrusted, marginally trusted or completely trusted. Ultimate trust is assigned by PGP to public keys that appear in the node's private key ring, other trust values regarding owner are specified by user.

Second, signature trust value (SIGTRUST flag byte) is assigned for each signatures added per entry for each public key. PGP searches the signature's author in the public key ring. Upon finding an entry, the SIGTRUST field is assigned value of OWNERTRUST field.

Third, key legitimate value (KEYLEGIT flag byte) is calculated (by PGP) by analyzing the cached signature trust fields for this entry in the following manner:

- ❖ If at least one SIGTRUST value is ultimate, KEYLEGIT is assigned complete.
- ❖ Otherwise weighted sum of the trust values is assigned to KEYLEGIT field.

PGP also periodically performs the ring processing tasks for maintaining its consistent state.

Task 1: For each OWNERTRUST field, PGP scans ring for all signatures of this author and accordingly updates all SIGTRUST values to be equal to OWNERTRUST value.

Task 2: All KEYLEGIT fields are updated based upon SIGTRUST values.

Figure 10.8 shows an example public key ring for user "Sameer". This user has acquired public keys from owners (directly) or from third party. The user "Sameer" enjoys ultimate trust. This user always trusts users C, D to sign other keys while partially trusts user A, B to sign other keys. The arrows (linking some other nodes) in the trust model show signatory whereas arrows not linking any other nodes indicate unknown signatory. Partial trust is shown by half shaded circle whereas complete trust is shown by full shaded circles. The dot (.) at the centre indicates legitimate keys identified by Sameer.

It can be easily identified from trust model that two partially trusted signatures are sufficient for certifying another key (e.g. Node F). A key signed by fully trusted signatory or by two partial trusted signatories are treated as legitimate (e.g. Node F, J, K, L). Node M represents an orphan node with two unknown signatures.

Check your progress 1

- a. Where is Pretty good privacy (PGP) applicable?
- b. Which block cipher is used by PGP for encrypting data?

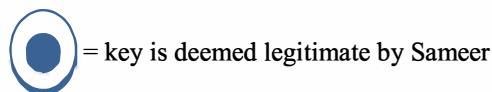
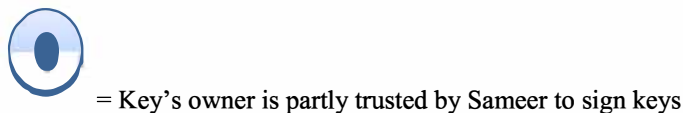
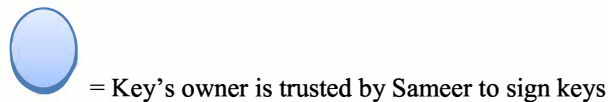
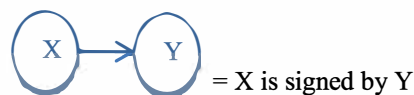
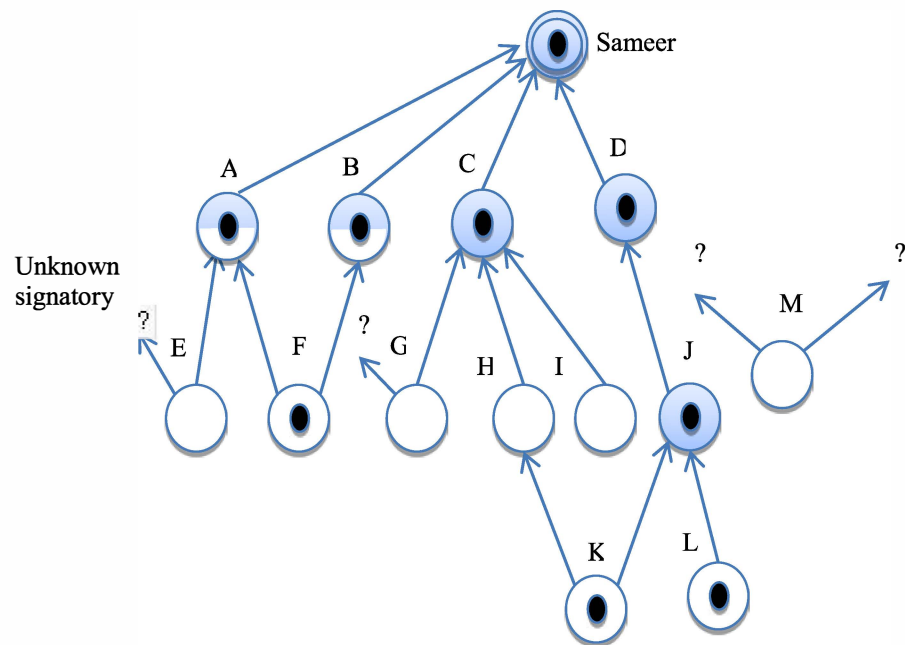


Figure 10.8 An example showing PGP Trust Model

10.5 FUNDAMENTALS OF S/MIME

For text based E-mail messages RFC 822 is used as the underlying standard. It defines the basic E-mail text message format. This format has two parts: (i) Envelope - containing information required for message transmission and delivery and (ii) Contents - the information to be delivered to receiver. RFC 822 message has header (header lines) and body unrestricted (ASCII text). Header and body are separated by a blank line. The header lines are used by user agent (browser) and also used to form the envelope. Header lines consist of keywords like From, To, Subject, Date etc. Each keyword is followed by colon and its arguments. An example 822 message is shown below [3].

Example 1 (conforming to RFC 822 standard)

Date: Tue, 16 Jan 2016, 06 : 55 : 45 (EST)

From: "Harish Shah" <hs@college.edu>

Subject: RFC 822 mail example

To: karan@rediffmail.com

Cc: rs@college.edu

Message body containing text goes here after leaving a blank line after message heading. An example message- Hi all, today an important discussion is required; Please assemble in Seminar Hall.

The RFC 822 based mails utilize SMTP store and forwarding approach/systems to forward the mails. Such systems have various shortcomings like inability to transfer executables/binary objects, inability to transmit national language, character (8 bit) based text messages, inability to transfer mail messages larger than particular size, translation problems at SMTP servers etc. Hence, RFC 822 is extended as Multipurpose Internet Mail Extension (MIME) via RFC 2045 to 2049. Major inclusions in MIME are:

- I. Addition of following 5 header fields that convey information pertaining to message body. These header fields are included in 822 headers.
 - ❖ **MIME Version:** For indicating that the message is in accordance with MIME (RFC-2045 & 2046).
 - ❖ **Content-Type:** For describing the data in body so that user agent can take action accordingly.
 - ❖ **Content-Transfer-Encoding:** For indicating the transformation/encoding applied on message for mail transport.
 - ❖ **Content-ID:** MIME entity identifier.
 - ❖ **Content-Description:** For describing the object in the message body.

Out of these 5 header fields, Content-ID & Content-Description fields may be ignored as these are optional while other fields are compulsory therefore needed by any MIME compliant entity.

II. Multimedia supporting content formats

Listed in RFC 2046, these are categorized into 07 major types for general type of data and 15 subtypes for specifying the format for particular type of data. Table 10.2 presents the MIME types and subtypes along with their description. Two examples for clarity purpose for multipart type are shown below:

Example 2

From: Sameer <sm@gmail.com>

To: Rajesh <rj@yahoo.com>

Subject: Conference invitation

MIME-Version: 1.0

Content-type: multipart / mixed; boundary="multipart mail boundary"

Preamble comes first and is to be ignored. Purpose - used as explanatory note to non-MIME readers.

--multipart mail boundary

Implicit plain ASCII text should lie here.

--multipart mail boundary

Explicit plain ASCII text should lie here.

--multipart mail boundary--

Epilogue comes at the end and is also to be ignored.

Table 10.2 MIME Content Types [3]

Type	Subtype	Description
Text (Basic text, no special software is required)	Plain	String of ASCII character
	Enriched	Greater formatting feature support and flexibility
Multipart (Body contains multiple, independent parts; content-type header field includes a parameter* named as boundary for delimiting the independent parts)	Mixed	Different parts are transmitted together and presented in order at the receiver.
	Parallel	Different parts may be presented without any order i.e., in parallel e.g., picture, text and commentary may be sent in parallel
	Alternative	Same information in different parts is represented in different manner. Different parts are ordered in increasing preference**
	Digest	Different parts are individual messages (as RFC 822 messages with headers)
Message	RFC 822	Body is RFC 822 messages
	Partial	A transparent fragmentation of large mail items. Three parameters id, Sequence number and total number of fragments are mentioned for this subtype in the content-type. Id is same for all fragments whereas sequence number is unique for each fragment
	External-body	Body has pointer (information required to access actual data object) instead of actual data object
Image	Jpeg	JPEF image format file/image
	Gif	GIF image format file/image
Video	mpeg	MPEG format video file
Audio	Basic	Single-channel 8 bit encoding with sample rate 8 khz
Application	Postscript	Adobe postscript data object
	Octet-stream	Binary data (8-bit type format)

* Boundary does not occur in the message parts and starts a fresh on new line. It is prefixed with two hyphens. Final boundary is suffixed with two hyphens.

** See example 2 and example 3 for more clarity.

Example 3

From: Sahil <sl@gmail.com>

To: Jai <jai@yahoo.com>

Subject: Text mail containing formatting

MIME-Version: 1.0

Content-Type: multipart/alternative;

boundary=boundary123

--boundary123

Content-Type: text/plain; charset=us-ascii

...text version of message without any formatting should lie here ...

--boundary123

Content-Type: text/enriched

...formatted text/enriched alternative of same message should lie here

--boundary123--

III. Transfer encodings govern inter-conversion of any form to a standard content for message body with aim of providing reliable delivery across different platforms and environments.

- ❖ A total of six values are possible for Content-Transfer-Encoding field namely 7bit, 8 bit, binary, x-token, quoted-printable and base64. The first three represents no encoding rather information about data. 4th (x-token) is non-standard encoding for which name (vendor or application specific) is to be provided. Quoted-printable encoding refers to human readable encoding whereas base64 is safe and compact encoding. It is also known as radix 64 encoding and is used in PGP as well.

Working of Secure Multipurpose Internet Mail Extension (S/MIME)

Secure MIME enhances E-mail security by making use of encryption/decryption and signing/verifying of the MIME messages. It is an IETF standard and is basically used for commercial and organizational use (In contrast PGP is also an IETF standard but is used for personal E-mail security) [3][5-7].

- S/MIME provides
 - Encryption of contents (Enveloped data)
 - Digital signing of data followed by base64 encoding (signed data). Due to encoding it is required that receiver should also have S/MIME capabilities to understand the message.
 - Application of base64 encoding only to signatures (clear signed data) leaving contents in unencrypted and un-

encoded. The data can be viewed by user not having S/MIME capability.

- Either signing the encrypted data or encrypting the signed data (signed and enveloped data).
- Following cryptographic algorithms are used in S/MIME
 - (i) Preferred methods for message digest & digital signature - SHA-I and DSS; other options - MD5, RSA.
 - (ii) Preferred methods for session key encryption - RSA; other options - Diffie-Hellman.
 - (iii) Preferred methods for (symmetric) encryption of message with session key - 3DES; other options - AES, RC2/40.
 - (iv) Preferred methods for creating message authentication code - HMAC with SHA-1.
- S/MIME has procedure for selecting the encryption algorithm by the sender and receiver. Sender may announce its decrypting capabilities which may be stored by the receiver for future use. Sender follows the following rules in order:
 - (i) Sender should choose highest preference capability from the receiver decryption capability list.
 - (ii) In absence of receiver in encryption list, sender should use in its outgoing messages the last encryption algorithm used by receiver for encryption and signing.
 - (iii) If no previous knowledge regarding decryption capability is present with the sender and sender may take risk regarding decryption capabilities of receiver, sender should use 3DES algorithm.
 - (iv) Same as (iii) above but now the sender is not willing to take risk regarding receiver's decryption capabilities and therefore use RC2/40.

Different types of messages in S/MIME

S/MIME adds new MIME content types. Among these, the newly added application types use the keyword PKCS (Public Key Cryptographic Specifications) [3][5-7].

- S/MIME is used mainly for securing the MIME entities (entire message or multiparts).
- S/MIME adds and processes some security related data like algorithm id, certificate etc. to produce PKCS object. The PKCS object is then added with MIME headers.
- The S/MIME content types (categories) are discussed next. Four S/MIME categories are represented by S/MIME parameters: signedData (signed S/MIME entity), envelopedData (encrypted S/MIME entity), degenerate signedData (S/MIME entity containing public key certificates) and compressedData (compressed S/MIME entity). The S/MIME entity in all these categories is an object consisting of arbitrary octet strings (binary data). This object is then encoded with base64 method.

(1) EnvelopedData

Type: Application; Sub Type: pkcs7-mime; smime parameter: envelopedData

Steps performed by sender for preparing envelopedData are:

- (i) Session key generation depending upon symmetric encryption algorithm selected.
- (ii) Encrypt the session key using receiver's public key.
- (iii) Prepare security related data block (termed as RecipientInfo) containing: Receiver public-key certificate id, algorithm id (for encrypting session key) and session key (encrypted).
- (iv) Use session key to encrypt message content.
- (v) Encode-RecipientInfo block and encrypted contents using base64. The encoded data becomes enveloped data.

Steps performed by receiver:

- (i) Decode/apply base64 encoding.
- (ii) Using public-key certificate id obtains the private key and decrypt to obtain session key.
- (iii) Decrypt the message using session key.

(2) SignedData

Type: Application; Sub Type: pkcs7-mime; smime parameter: signedData

Steps performed by sender for preparing signed data are:

- (i) Select algorithm for calculating hash of message.
- (ii) Apply the algorithm and evaluate hash.
- (iii) Encrypt hash using private key of sender.
- (iv) Prepare security related data block (termed as SignerInfo) containing: sender's public-key certificate hash algorithm id, encryption algorithm id, encrypted hash (digital signature).
- (v) Encode message and SignerInfo block using base64 and transmit the signed Data.

Steps performed by receiver for verification:

- (i) Decode/apply base64 encoding.
- (ii) Decrypt using sender's public key extracted from its public key certificate to obtain hash value.
- (iii) Calculate hash from message using hash algorithm as per algorithm id.
- (iv) Compare calculated hash and obtained hash values to verify the signature.

(3) Clear Signing

Type: Multipart; Sub Type: signed

First part - Any MIME type e.g. Text /plain

Second part – Type: Application; Sub Type: pkcs7-signature; smime parameter: signedData

Steps performed by sender for clear signing are:

- (i) Create two part MIME message. The first part is the message to be signed i.e. message to be transferred “in clear” without any transformations.
- (ii) Create a signedData object from first part i.e. of type Application/pkcs7-signature. This object has empty message content field and is like detached signature. Encode this object using base64 method and embed it as second part of the multipart MIME message.

Steps performed by receiver:

- (i) Evaluate the hash of the first part of the multipart MIME message.
- (ii) Decrypt the signatures from second part to extract the hash.
- (iii) Compare the evaluated hash and extracted hash to verify the message.

(4) Registration request

Type: Application/pkcs10

- Used for requesting public key certificate by an application or user from Certification Authority (CA).
- Contents of this message are:
 - CertificationRequestInfo (security related) block containing - subject (user) name and public key of subject.
 - Public key encryption algorithm identifier.
 - Signature of certificationRequestInfo block created using sender’s private key.

(5) Certificates-Only message

Type: Application; Sub Type: pkcs7-mime; smime parameter-degenerate

- This message is sent in response to registration request.
- It contains only certificates or Certificate Revocation List (CRL).
- The entire message is signed but has no message content and signer Info field is empty.

Check your progress 2

- a. What identifies the MIME entities uniquely with reference to multiple contexts?
- b. What does S/MIME stands for?

10.6 SUMMARY

This unit is devoted to E-mail security. In this unit, the two prominent methods of providing security to E-mails are discussed. First is Pretty

Good Privacy (PGP), an open source digital signature based E-mail security method. The terms and concepts related with PGP are covered. The public and private key rings are introduced followed by exemplifying the PGP trust model. Second is Secure Multipurpose Internet Mail Extension (S/MIME) method which is directed towards providing security services to MIME based mail exchanges. Various S/MIME message types, subtypes and contents are discussed.

Terminal Questions

1. *State the four principal services provided by PGP? Why does PGP generate signature before compression?*
2. *What is a detached signature? What is its need?*
3. *How PGP utilizes the concept of Trust?*
4. *List the major differences between MIME and S/MIME*
5. *Explain various subtypes of multipart type*
6. *Differentiate between enveloped data and signed data*

References:

1. https://en.wikipedia.org/wiki/Pretty_Good_Privacy accessed on 15/12/2016.
2. <https://ssd.eff.org/en/module/introduction-public-key-cryptography-and-gpg> accessed on 15/12/2016.
3. Stallings, William, “Chapter 15: Electronic Mail Security”, Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
4. <http://www.facweb.iitkgp.ernet.in/~sourav/PGP.pdf> accessed on 15/12/2016.
5. Forouzan, Behrouz A., “Chapter 16 - Security at Application Layer: PGP and S/MIME”, Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
6. <http://web.fe.up.pt/~jmcruez/ssi/ssi.0910/trabs-als/apres.22-morena.simatic.pdf> ‘S/MIME protocol’ 10/02/2017.
7. <https://tools.ietf.org/search/rfc5751> ‘Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification’ accessed on 10/01/2017.

UNIT - 11

IP Security

Structure

- 11.0 Introduction
- 11.1 Objectives
- 11.2 IP Security Application Scenarios
- 11.3 Architectural Description of IP Security
- 11.4 Authentication Header (AH) Protocol
- 11.5 Summary
- 11.6 Terminal Questions

11.0 INTRODUCTION

IP Security (IPSec) is a security enhancement to the present day Internet protocol (IP). Two version of IP i.e., IPv4 and IPv6 exist. IPSec places additional header information in both IPv4 and IPv6 for providing security features. IPSec talks about three security features: (i) authentication, (ii) confidentiality and (iii) key management. It encompasses two different modes: (1) transport mode, (2) tunnel mode. In transport mode, a security feature (authentication or confidentiality) is applied to an IP packet excepting its header whereas in tunnel mode, a security feature (authentication or confidentiality) is applied to entire IP packet including header and then the packet is encapsulated into another packet for further transfer. In this unit, IPSec overview, architecture, modes and authentication header is provided.

IPSec is basically a packet level security wherein an IP packet leaving an organization network is encrypted so that it remains secure while traversing on insecure and untrusted network. Also, a packet entering an organization network is verified for its authenticity and only those packets that are authenticated are given entry. Thus, if higher level application specific security mechanisms like S/MIME, PGP (E-mail security mechanisms); Kerberos (Client-Server security mechanism); secure socket layer (web access mechanism) exist, IP Security provides addition security measure.

If higher level application lacks any security measure, IP Security provides essential security measure. Among three security features supported by IPSec, authentication ensures that the received packet indeed belongs to the source mentioned in header and also ensures that the packet is not altered in route. Confidentiality via encryption ensures secrecy and privacy of data kept inside the packet. Key management ensures secure key exchange between two parties.

11.1 OBJECTIVES

At the end of this unit, you will be able to:

- get an overview of IP Security.
- have an elaborative idea of IPSec architecture.
- learn the various security features and functional areas supported by IPSec
- enhance the understandability of terms like security association, security policy etc.
- know about the different operation modes of IPSec.
- explain the role of authentication header and functionality of anti-replay service provided by IPSec.

11.2 IP SECURITY APPLICATION SCENARIOS

Fig 11.1 shows TCP/IP protocol suite used in Internet where layer 3 (network layer) is referred to as Internet Protocol (IP). IP Security (IPSec) provides authentication and confidentiality to packets at this layer. IPSec is useful as security provided at higher layers i.e., application and transport is not sufficient in few cases e.g.

- (i) PGP and S/MIME provides security only to E-mail application not to all client/server applications. Thus, IPSec is useful to such applications.
- (ii) Though applications that utilize TCP as underlying protocol are protected by SSL/TLS but not all applications use TCP some use UDP as well.
- (iii) Routing protocols works at layer 3 (IP layer) and hence require security at this layer only.

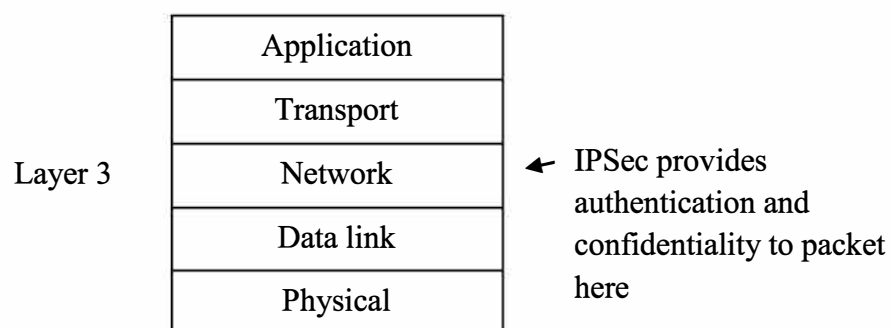


Fig 11.1

- IPSec can be implemented in firewall or router (i.e. devices connecting two internetworks). It can also be implemented in the end hosts/systems.
- IPSec provides security to packets moving outwards and verifies the packets coming inwards at the point of its implementation.
- Fig 11.2 shows the cases of use of IPSec.

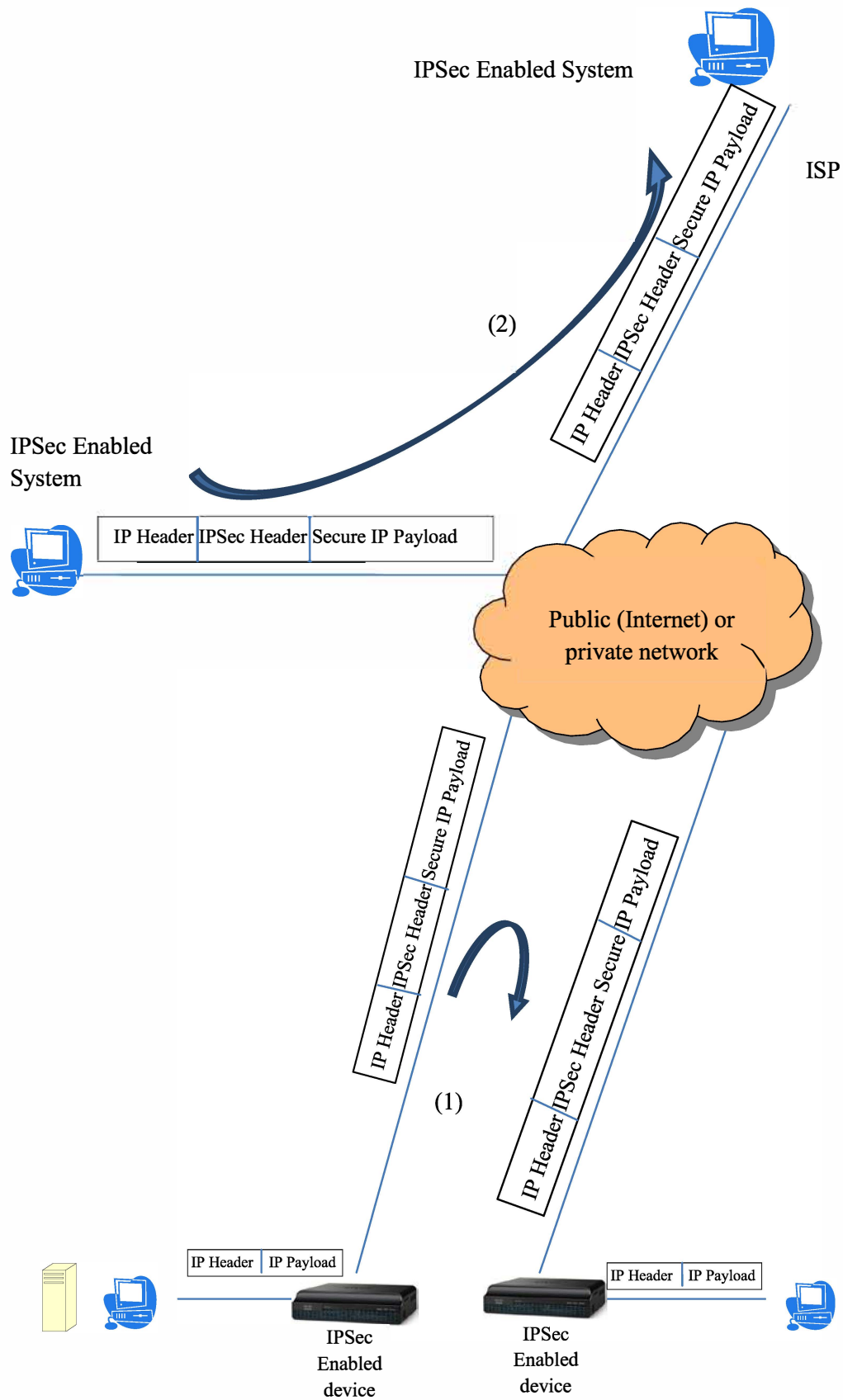


Figure 11.2 Use of IP Security in an example network

Case 1: Securing two different company offices (LANs) located at different locations and connected through public WANs (via routers). In this case, the traffic (IP packets) moving out via router1 are protected by IPSec while it is verified at router2. The IPSec header is added to outgoing packets at router1 while IPSec header is stripped off at router2. Thus, company's data remains safe despite of using public network.

Case2: Secure remote access to end user. The user's end system (system1) uses IPSec to send packets safely to access a distant network or Internet Service Provider (ISP) to utilize their resources and thus saving physical commuting or travelling cost to distant network or ISP.

Apart from these cases, IPSec enhances security for existing E-commerce applications and also establishes secure communication between partners over extranets and intranets.

- IPSec offers following benefits
 - (i) IPSec secures all the traffic that leaves the periphery of an organization through firewall or router.
 - (ii) IPSec analyses all the incoming traffic to an organization from outside world and can filter the undesired ones.
 - (iii) IPSec is transparent to applications as it operates below the transport layer (TCP/UDP) and involves no up-gradation of existing client or server software.
 - (iv) IPSec operates independent of users and hence, remains unaffected by users joining or leaving an organization. Also, it does not require any user training to educate the users and operators.
 - (v) Individual user level security can be provided by IPSec, if desired by setting up virtual secure subnetwork in an organization.
 - (vi) IPSec protects routing mechanism providing associations between routers such that an attacker can not cause any harm to routing mechanism. IPSec ensures following in routing:
 - (a) Provides authenticity to new router advertisements by which a router realizes its presence.
 - (b) Provides authenticity to advertisements between any two neighboring routers.
 - (c) Non forging of routing update packet.
 - (d) Reply of receipt (redirect messages) from the router to whom request was made.

Check your progress 1

- a. IPSec work at which layer? When it should be preferred?
- b. IP Sec works independent of user. Justify?

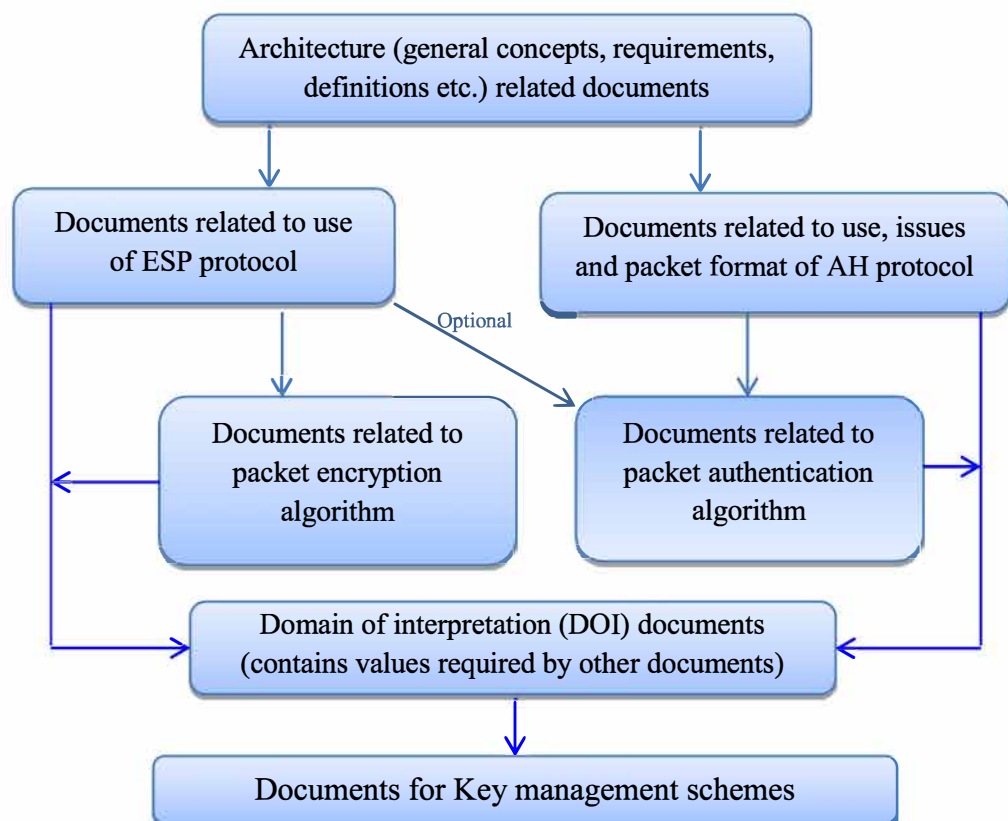
11.3 ARCHITECTURAL DESCRIPTION OF IP SECURITY

- IPSec specification mentions several documents (RFCs), worth mentioning are the following ones:
RFC 2401 that gives overview of entire IPSec architecture;
RFC 2402 provides authentication extension details of IPSec;
RFC 2406 provides encryption extension details of IPSec and
RFC 2408 that provides key management description.
- Features provided by IPSec are optional in IPv4 while compulsory in IPv6.
- These features are added in the form of headers placed after the main IP headers.
- A classification of documents pertaining to IPSec, published by IP Security protocol working group (of IETF) is shown in Fig 11.3.
- Under this classification, basic definitions, requirements and mechanism is presented under the Architecture head; Encapsulating Security Payload (ESP) format, issues are covered under ESP protocol head, while that of Authentication Header (AH) are covered under AH protocol head; Encryption algorithm and their use is presented under Encryption Algorithm head while Authentication Algorithm and their use is presented under Authentication Algorithm head; Key management issues and schemes are covered in Key management document; Domain of Interpretation (DOI) contains values and terms required by other documents for relating with each other like algorithm identifies, key lifetime etc [1].
- IPSec has two protocols:
 - (i) Authentication Header (AH) protocol
 - (ii) Encapsulating Security Payload (ESP) protocol
- IPSec provides following services via these protocols:
 - (i) Access control
 - (ii) Message integrity independent of connection
 - (iii) Origin authenticity
 - (iv) Protection against replays
 - (v) Confidentiality
 - (vi) Traffic flow (confidentiality) management
- An IP enabled system selects:
 - (i) Security protocol among AH and ESP
 - (ii) Algorithm(s) for providing desired service
 - (iii) Cryptographic keys as per the selected protocol
- Table 11.1 shows the services provided by each of AH and ESP.
- ESP has two options: with authentication and without authentication.

- AH does not provide confidentiality and traffic flow (confidentiality) management.
- ESP (without authentication) does not provide message integrity and origin authenticity.
- ESP (with authentication) provides all services.
- IPSec (Both AH and ESP) provides access control by checking the Security Association (SA) from the Security Association Database (SAD). If no SA is found in the database, packet is dropped.
- IPSec (Both AH and ESP) provides replay protection using sequence numbers and sliding receiver window.
- SA, SAD and sliding receiver window are discussed next.

Table 11.1 services provided by each of AH and ESP

	Access Control	Message Integrity	Origin Authenticity	Replay Protection	Confidentiality	Traffic flow (confidentiality) Management
AH	Y	Y	Y	Y		
ESP (without authentication- only encryption)	Y			Y	Y	Y
ESP (with authentication)	Y	Y	Y	Y	Y	Y



Notion of Security Association

- A security association (SA) is a logical relationship or contract between two communicating hosts. An SA helps in creating a secure channel between these hosts.
- It is one way only i.e. from sender to receiver. For securing traffic in opposite direction another security association is required. Thus, for a two-way secure exchange, two SAs are required.
- Fig 11.4 shows an example where confidentiality between sender S and destination D is required. Thus, S defines an outbound SA having encryption algorithm and symmetric shared key for encrypting the message send to D. Similarly, D defines an inbound SA having same encryption algorithm and corresponding symmetric shared key for decrypting the message.

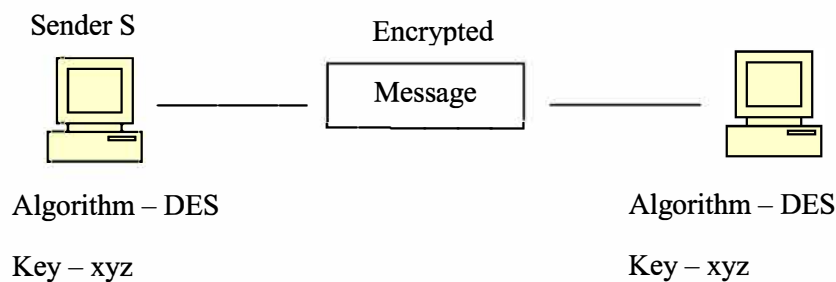


Fig 11.4 An example showing confidentiality between sender S and destination D

- An SA may be complex depending upon whether:
 - (i) SA for integrity, authentication along with encryption is required.
 - (ii) Different algorithms and parameters that exist for different protocols.
 - (iii) Communication is required between multiple senders and receiver.

In case (iii), each host should have both inbound and outbound SAs.

- Set of SAs are put together in a database termed as Security Association Database (SAD).
- SAD is basically a matrix with each row having a single SA.
- There are two SADs: inbound SAD and outbound SAD for keeping inbound and outbound SAs.
- Fig 11.5 shows one such SAD.
- An SA kept in SAD is uniquely identified by:

- (i) Security Parameter Index (SPI)
 - SPI is a 32 bit number or string assigned to the SA.
 - It is determined during SA negotiation on the hosts.
 - It is placed in the IPSec header (AH or ESP) so that receiver may identify the SA for this packet from SAD.
 - SPI remains same in all the packets tied to same SA.
 - (ii) Destination Address (DA) - Destination address of host (end user system/firewall/router). Thus, SAs are unique per DA.
 - (iii) Protocol (P) - IPSec protocols: either AH or ESP.
- Hence, An SA is uniquely identified by SPI, DA, P combination.
 - Each SA has associated parameters (remaining entries of a row in SAD) [2]. These are:
 - (i) **Sequence Number Counter:** A 32 bit counter used for generating sequence number to be kept in sequence number field of IPSec header.
 - (ii) **Sequence Number Overflow:** Flag used to identify the sequence number overflow and define the action taken by host in such circumstances.
 - (iii) **Anti-Reply Window:** A window of size n used to identify the replayed IPSec packet at the receiver.
 - (iv) **AH Information:** Information for AH protocol like algorithm, keys, key lifetime etc.
 - (v) **ESP Information:** Information for ESP protocol like encryption algorithm, authentication algorithm, keys, key lifetime, Initial Vectors (IV) etc.
 - (vi) **SA Lifetime:** Defines lifetime of the associated SA.
 - (vii) **IPSec Mode:** Either transport or tunnel mode of IPSec.
 - (viii) **Path MTU:** Defines maximum packet size (without fragmentation) and aging variables.
 - A key distribution mechanism is linked to authentication and encryption mechanisms via SPI

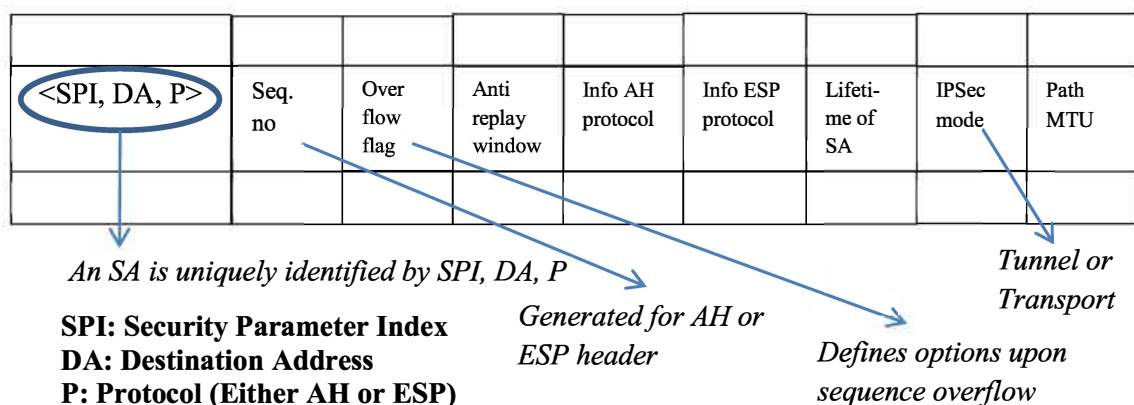


Figure 11.5 Attributes for a single SA in a Security Association Database (SAD)

Security Policy Description

- A security policy (SP) in IPSec implies the type of security to be applied to IP packet upon arrival or departure.
- The SP is kept in a Security Policy Database (SPD).
- Each entry in SPD is indexed based upon sextuple (termed as selectors): source address, destination address, name, protocol, source port and destination port.
- Here, source and destination address are unicast or multicast IP addresses; name refers to DNS entity; protocol refers to AH or ESP; source and destination port addresses refer to port addresses of processes running on source and destination systems.
- An example SPD is shown in Fig 11.6.
- An inbound and outbound SPD is created on IPSec enabled host.
- An outbound SPD is consulted (SPD procedure is called) when a user packet is send by transport layer to network layer (Fig 11.7).
- In this case outbound SPD is searched based upon sextuple index.
- Depending upon security policy either the packet is dropped (if it cannot be send further as per policy) or bypassed (if no policy exists for this index) or IPSec is applied.
- IPSec application results in two cases:
 - (i) If outbound SA already exists then based upon SA index (triplet), the corresponding SA is selected from out bound SAD. It is then followed by formation of AH or ESP header (utilizing encryption and/or authentication) and packet transmission.
 - (ii) If outbound SA does not exists then Internet Key Exchange (IKE) protocol is invoked so as to create new outbound and inbound SA for outbound SAD and inbound SAD respectively for this traffic.
- Upon packet arrival at the receiver, the inbound SPD is searched using sextuple index (Fig 11.8).
- Depending upon security policy either the received packet is dropped (if it cannot be send further as defined by policy) or bypassed (if no policy exists for this index) or IPSec is applied.
- IPSec application results in two cases:
 - (i) If inbound SA exists then based upon SA index, the corresponding SA is selected from inbound SAD. It is then followed by verification of AH or ESP header (using decryption and/or authentication) and removal of AH or ESP header. The extracted packet is then delivered to transport layer.
 - (ii) If no inbound SA exists, packet is discarded [2].

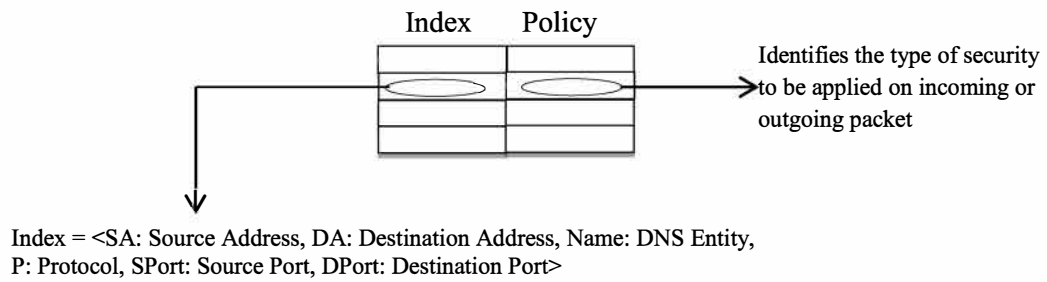


Figure 11.6 Security Policy Database (SPD)

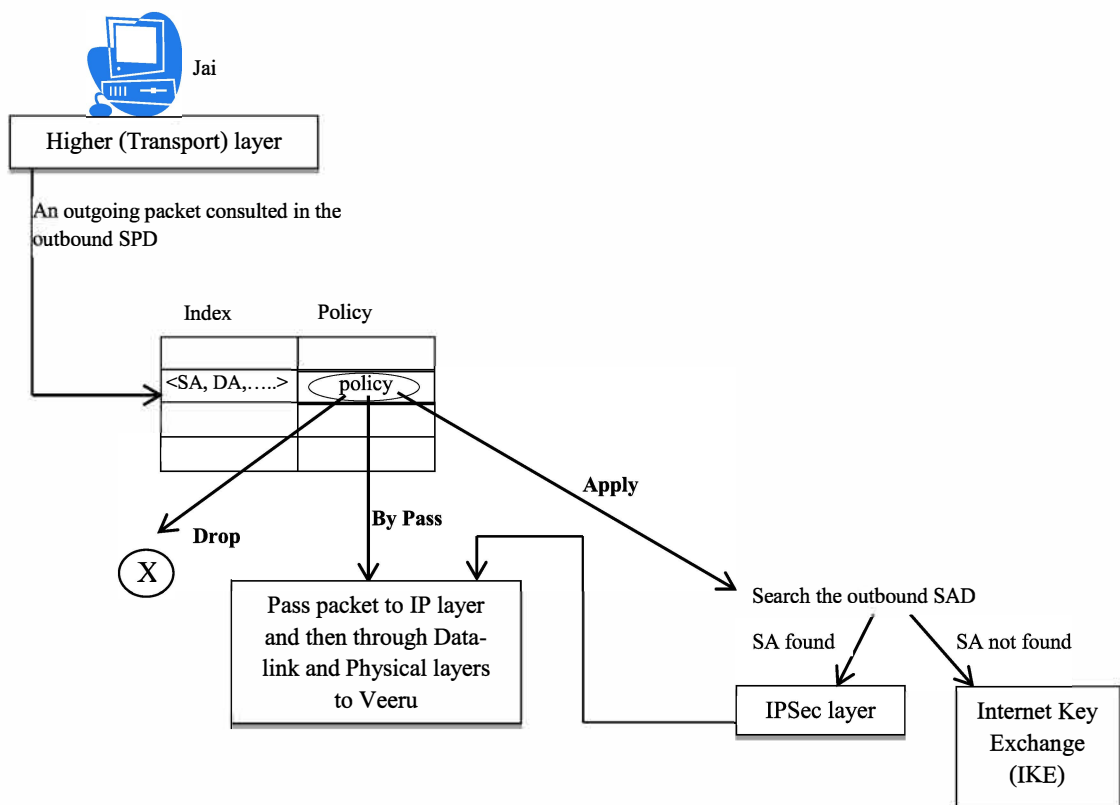


Figure 11.7 Outbound processing

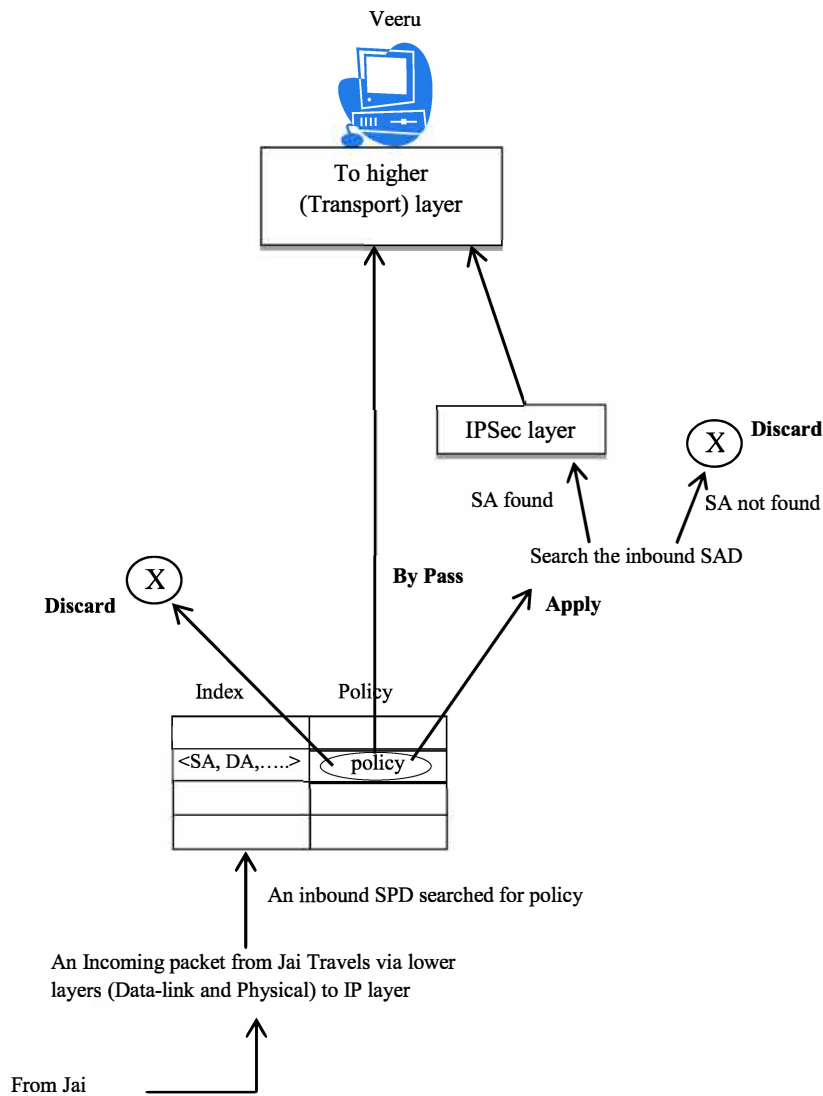


Figure 11.8 Inbound processing

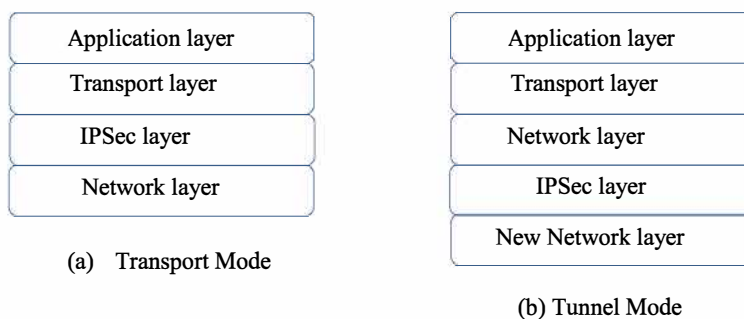


Figure 11.9 Transport mode versus tunnel mode

Operational Modes of IPsec

- IPsec operates in: transport mode or tunnel mode.
- In transport mode, transport layer packet (UDP or TCP segment or ICMP packet) is encapsulated by IPsec via addition of IPsec header and trailer. This encapsulated data is then added to the network layer packet as IP payload (Fig 11.9(a)).

- IPSec does not protect the IP header in transport mode, only the transport layer packet/information (IP payload) is protected [2].
- This transport mode is used to protect end to end (host to host e.g. client-server or workstation-workstation) data (Fig 11.9 (a) and 11.10) via encryption/decryption or authentication.
- In tunnel mode, the entire IP packet including the IP header (old) is protected (Fig 11.9 (b)). IPSec adds header/trailer to network layer packet and then adds a new IP header to it. Thus, tunnel mode is used to protect end to intermediate data (Fig 11.10).
- The new IP header and old IP header contains different information.
- The tunnel mode is used between.
 - (i) Two routers
 - (ii) A host and a router
 - (iii) A router and a host
- Thus, it appears that the protected packet is going through an imaginary tunnel (safe channel) on insecure network.
- In comparison, IP Sec lies in between transport layer & network layer in transport mode where as in between network and new network layer in tunnel mode (Fig 11.9).
- ESP in transport mode protects (encrypts and optionally authenticates) the IP payload only where as in tunnel mode protects the entire inner IP packet including the inner IP header.
- AH in transport mode authenticates IP payload along with selected parts of IP header where as in tunnel mode authenticates the entire inner IP packet and selected parts of outer IP header.

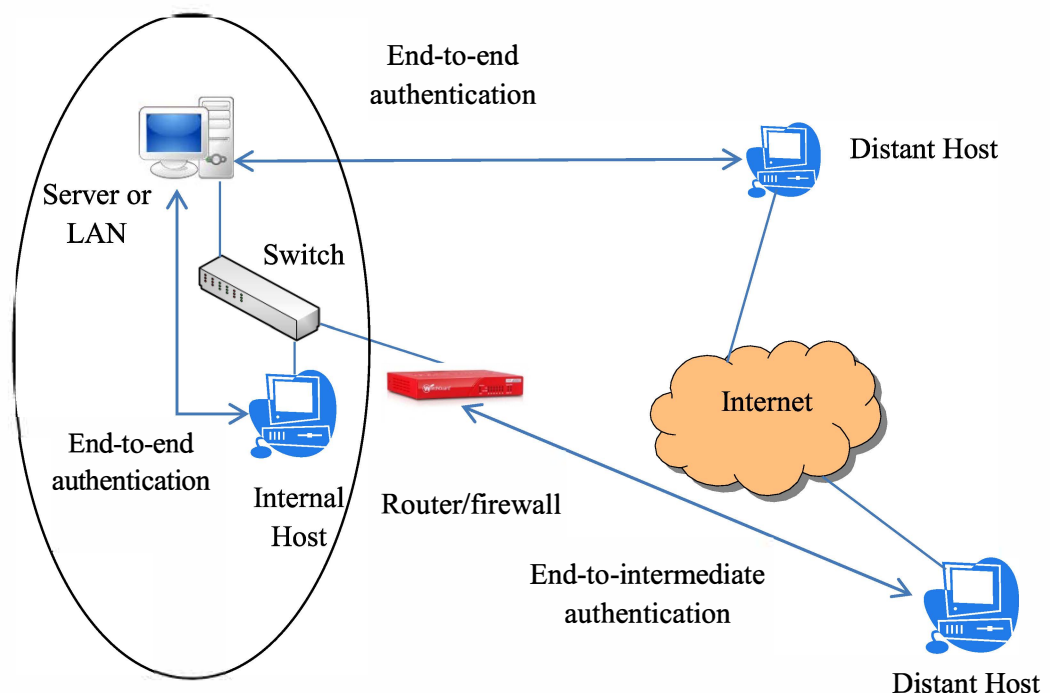


Figure 11.10 Two different authentication types (End-to-End and End-to-Intermediate)

(a) IPv4



(i) Before applying AH

*Authenticated except for mutable fields**



(ii) Transport mode

Authenticated except for mutable fields in the new IP header



(iii) Tunnel mode

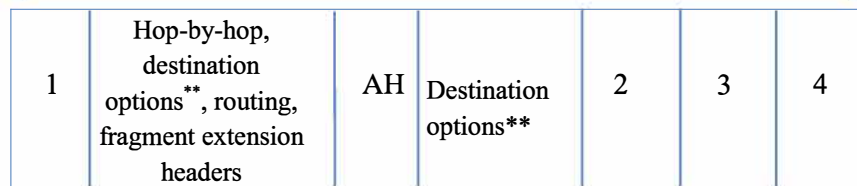
- 1: Original IP header
2: TCP header
3: Data

(b) IPv6



(i) Before applying AH

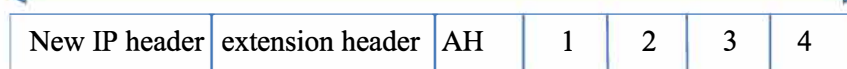
*Authenticated except for mutable fields**



(ii) Transport mode

- 1: Original IP header
2: Extension header (if present)
3: TCP header
4: Data

Authenticated except for mutable fields in the new IP header



(iii) Tunnel mode

Figure 11.11 AH Authentication in IPv4 and IPv6

* Mutable fields are set to zero for MAC calculation during authentication.

** Destination options may lie before or after the AH in IPv6 using IPsec.

Check your progress 2

- a. What is security policy? Where it is kept?
b. What does IPsec protects in the tunnel mode?

11.4 AUTHENTICATION HEADER (AH) PROTOCOL

- Authentication Header (AH) protocol enhances security by addition of Authentication Header that provides source authentication and IP payload integrity.
- For this, AH protocol calculates Message Digest (MD) or hash using symmetric key and hash function.

- The calculated MD is embedded into Authentication Header and then the IP packet is transferred to receiver.
- Receiver verifies the authenticity and integrity of the packet.
- The position of AH depends upon the version (IPv4 or IPv6) and mode (transport or tunnel).
- AH protocol does not provide privacy.
- Fig 11.11 shows IPv4 and IPv6 before applying AH and after applying AH (for transport and tunnel mode) [1].
- The IPSec Authentication Header (including fields) is shown in Fig 11.12. It consists of:
 - (i) **Next Header** - type of IP packet payload (TCP/UDP/ICMP/OSPF). IPSec copies the protocol field value of original IP packet header into next header field. The protocol field value in new IP header is changed to 51 to indicate that the packet contains AH.
 - (ii) **Payload Length** - it represents AH length in 4- byte multiples excluding first 8 bytes.
 - (iii) **Reserved**– 16 bits kept aside for future use.
 - (iv) **Security Parameter Index (SPI)** - required for Security Association (SA) and is like virtual circuit identifier and remains same for a particular SA connection.
 - (v) **Sequence Number** – provides sequencing of IPSec packets and prevents replays using anti-replay service. The sequence number changes for retransmitted packet and does wrap around. Upon reaching its maximum value, a new connection is initiated.
 - (vi) **Authentication Data (variable size - Integral of 32 bit words)** - used for keeping Integrity Check Value (ICV) for a packet.

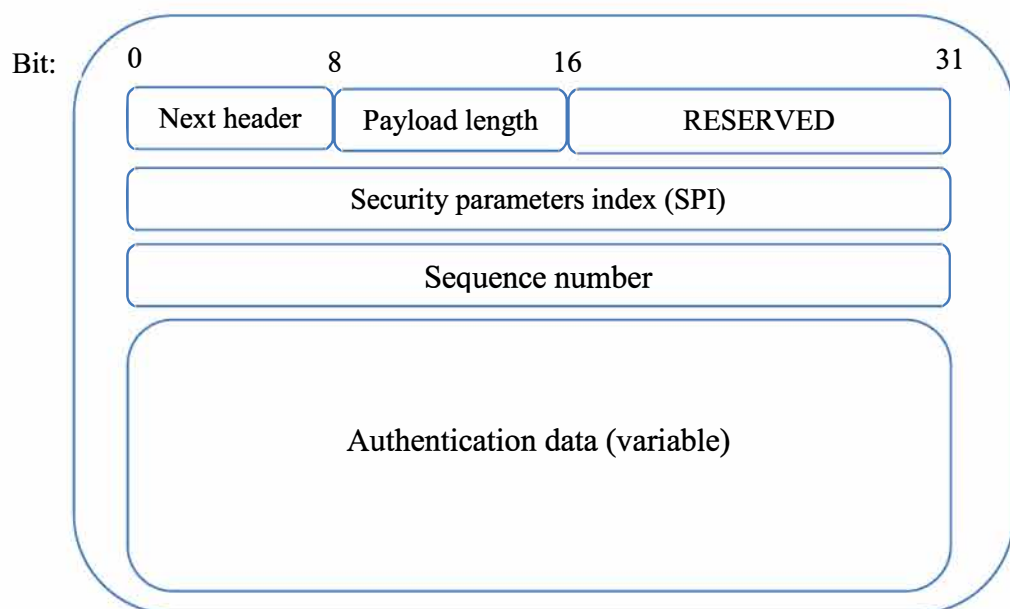


Figure 11.12 IPSec Authentication Header

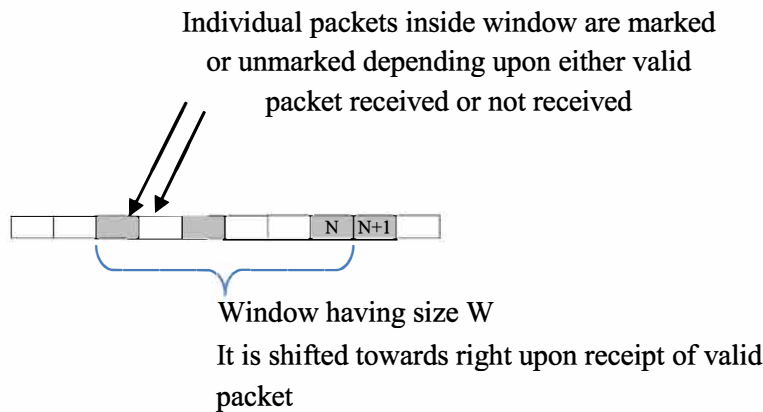


Figure 11.13 Anti-Replay Mechanism adopted by IPSec

AH protocol service against Replays (Fig 11.13): For implementing successful anti-replay service sender and receiver must take following actions:

Actions taken by the Sender:

- Sender initializes the sequence number to zero at the time of establishing a new Security Association (SA).
- This value is incremented and put in the IPSec header whenever a new packet is send.
- Thus, initial value of any assigned sequence number is 1 and its maximum value is $2^{32}-1$.
- When max value is reached; The SA is terminated and new SA in established.

Actions taken by Receiver:

- Receiver implements a window of size W (default W = 64).
- Note:** As packet may be received out of order due to connectionless and unreliable nature of IP, the notion of window is used.
- For a successfully received largest sequence number N, the left and right edges of window are represented as $(N - W + 1)$ and (N) respectively.
 - Any received packet whose MAC is verified and has -
 - (i) Sequence number that lies in the window i.e., within left and right edges of window, is treated as valid and hence, the corresponding window slot is marked.
 - (ii) Sequence number that lies outside window and just after right edge, the window is advanced (new right edge = previous right edge + 1) and the corresponding window slot is marked.
 - Packet is discarded when:
 - (i) Authentication of packet fails (due to wrong MAC).
 - (ii) Sequence number of received packet lies towards left of the window i.e. before $(N - W + 1)$.

Use of Integrity Check Value (ICV) by AH protocol

- The MAC or truncated codes produced by MAC algorithm (e.g. MD5/SHA-1 produced 96 bit truncated code) are kept in authentication data field of IPSec header and termed as ICV.
- The default length of authentication data field is 96 bits.
- The MAC in AH is calculated using:
 - (i) IP header fields that do not change during transmission (e.g. source and destination address) while ignoring the fields that may change (mutable fields) e.g. TTL field (these fields are set to zero while calculating MAC).
 - (ii) Authentication data field in AH with value zero at both sender and receiver.
 - (iii) Upper level protocol data (IP payload, may also include inner IP packet in tunnel mode).

11.5 SUMMARY

In this unit first an overview of IP Security is provided, followed by an elaborative description of IPSec architecture. You should be able to understand the three security features and functional areas supported by IPSec like authentication, confidentiality and key management. The unit also introduces to you new terms like security association, security policy etc. Two modes of operations of IPSec namely, transport and tunnel modes are described in details.

Terminal Questions

1. *Give some of the examples where IPSec is beneficial.*
2. *What is a Security Association (SA)? What parameters are used to identify a particular SA?*
3. *Define: SAD and SPI.*
4. *Differentiate between transport and tunnel modes*
5. *How replay attack is defended by IPSec?*
6. *In a tunnel mode, a new outer IP header is constructed. What is relationship between inner and outer header?*

References:

1. Stallings, William, "Chapter 16: IP Security", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
2. Forouzan, Behrouz A., "Chapter 18 - Security at Network Layer: IPSec", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.

UNIT - 12

Web Security

Structure

- 12.0 Introduction
- 12.1 Objectives
- 12.2 Security Threats Faced by Web
- 12.3 Web Traffic Protection Methods
- 12.4 Overview of Secure Socket Layer Protocol
- 12.5 Overview of Secure Electronic Transaction Protocol
- 12.4 Summary
- 12.5 Terminal Questions

12.0 INTRODUCTION

Internet based applications are very popular and widely used in E-commerce activities. These are extremely vulnerable to various threats and attacks. This chapter presents two important security protocols used in the Internet for facilitating commerce i.e., these protocols are mainly related with E-commerce related activities for example, smooth payment to the merchant over web, railway reservation, online payment of telephone bills, use of credit/debit cards etc. The first protocol is Secure Socket Layer (SSL) and the second one is Secure Electronic Transaction (SET). The former is concerned with providing security services to the applications that use TCP as the underlying protocol and hence functions between TCP and application layer. Its Internet standard version is referred to as Transport Layer Security (TLS). Therefore, the protocol is commonly referred as SSL/TLS. SSL/TLS protects messages of two users (sender and receiver) via symmetric encryption and message authentication code. The latter protocol i.e., SET is used for securing credit card transactions. Both these protocols are discussed and presented in details in this unit.

12.1 OBJECTIVES

With increased E-commerce and digital payment methods over Internet, an obvious and implicit requirement is that of security. This unit deals with providing security to end user applications. After going through the contents of this unit, you will be able to:

- have an idea of some of the major threats and attacks that target Internet applications.
- learn the two prominent protocols that are mainly related with E-

commerce related activities.

- have an elaborative idea of Secure Socket Layer (SSL).
- have an overview of credit card security protocol i.e., Secure Electronic Transaction (SET).

12.2 SECURITY THREATS FACED BY WEB

On web, the user uses client software (browser) to access the server software (web server). An important issue in the web is to understand the notion that the web security requirements are different from the normal computer and network security requirements. This can be understood from the following [1]:

- The traditional publishing environment is not prone to severe security threats. In comparison, the electronic publishing on the web (via web servers) is exposed to severe threats and attacks.
- The corporate try to keep product information and business transactions safe while providing the controlled information outflow via website but in the unsafe web environment their reputation and money can be affected via compromise of Web servers by the attackers.
- Web browsers, web servers and web contents are easy to manage and their functionality is well understood by the less technical users. This is not the case with the underlying code which is usually complex (not easily understood by the users who operate upon) and as such may hide many potential security flaws at several locations. Thus, many a times, new, properly installed and safe systems are easily subverted by the attackers who find flaws in code to break or intrude the systems.
- A compromised web server may be used by the attacker to intrude or gain access to data and systems not part of the web itself but connected to the server locally.
- Novice and untrained users who interact with servers using browsers are usually unaware of the security risks, security tools and effective countermeasures.

Thus, web related threats and attacks are to be analyzed separately and therefore, this section is dedicated to overview of web security threats.

As classified in unit 1, web related threats are classified into: active and passive.

Active attacks target - website alteration, message alteration between client and server, user impersonation etc.

Passive attacks target - eavesdropping on web traffic, restricted website information leakage etc.

The web threats target mainly *integrity, confidentiality, availability and authentication*.

Integrity threats use modification strategies to endanger user data, memory contents and message traffic in transit. These may result in loss of information, compromise of machine and vulnerability to other threats. Such threats are nullified by applying cryptographic checksums.

Confidentiality threats use sniffing, eavesdropping and theft strategies to endanger server information, client information (regarding which client/user is talking to server), network configuration details and client's communicated data. These may result in loss of information and loss of privacy. Such threats are nullified by applying encryption and using web proxies.

Denial of Service threats consider killing of user threads, flooding of a machine with bogus requests, filling up disk or memory, isolation of machine by DNS attacks to hamper availability. These may disrupt the user's work and may cause user annoyance. These threats can be detected but are difficult to nullify and prevent.

Authentication threats use spoofing, impersonation and forgery to endanger the authenticity. These may result in acceptance of false information and misrepresentation of user. Such threats are nullified by applying cryptographic checksums.

12.3 WEB TRAFFIC PROTECTION METHODS

Web related threats are also classified depending upon location of the threat: Web server, Web browser, and web traffic between browser and server. First two falls under computer system security.

Web Traffic Security approaches differ depending upon their relative position in the protocol stack. The approaches fall under following three categories [2]:

- (1) Using IP security (IPSec) below TCP (Figure 12.1(a)). This has following advantages:
 - ❖ IPsec is a general-purpose solution.
 - ❖ It is transparent to end users and applications.
 - ❖ Use of filtering in IPSec implies less processing overhead is required.

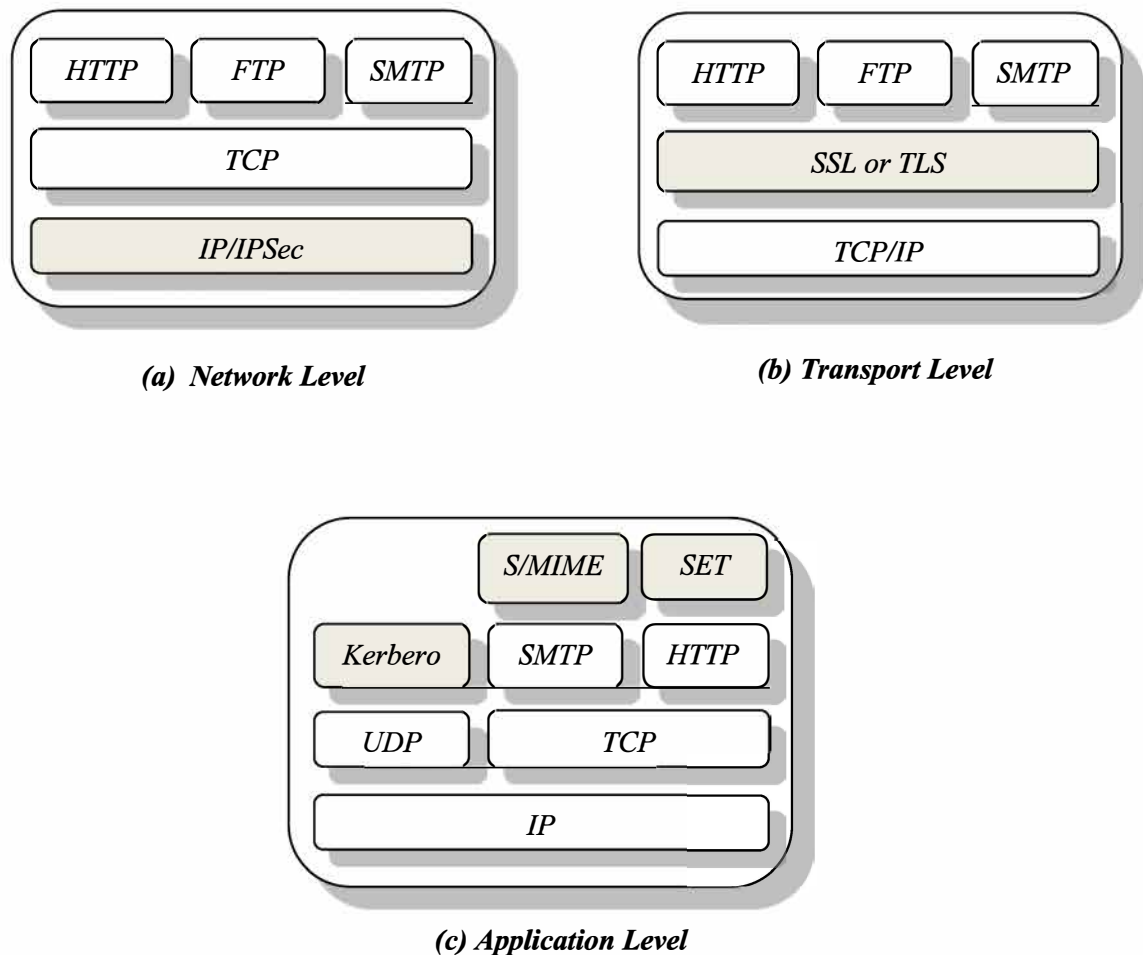


Figure 12.1 Various methods for web security and their relative position in the TCP/IP Protocol Stack

- (2) Using Secure Sockets Layer (SSL)/Transport Layer Security (TLS) above TCP (Figure 12.1b). SSL/TLS can be implemented as part of the underlying protocol suite or it can be embedded in specific packages e.g. web browsers and servers can embed SSL. In former implementation, SSL remains transparent to the applications.
- (3) Embedding the security methods within the particular application (Figure 12.1c). In this case application-specific security method can be tailored as per the application's requirements. For example, SET* on top of HTTP for securing credit card transactions, S/MIME on top of SMTP for securing E-mail services.

* This is a common implementation; it is also possible that SET makes use of TCP directly in some other implementation.

12.4 OVERVIEW OF SECURE SOCKET LAYER PROTOCOL

Secure Socket Layer (SSL) is one of the most popular Web security mechanisms, following are some of the interesting facts regarding SSL protocol:

- SSL is implemented as a transport layer security service.
- It was originally developed by Netscape.
- It's version 3 (SSLv3) was designed by taking public and industry input.
- Upon reaching a consensus, TLS working group was formed in IETF for developing the Internet standard and hence it was subsequently named as TLS (Transport Layer Security). The first TLS version is essentially represented as SSLv3.1
- It uses TCP to provide a reliable end-to-end service.
- SSL does not represent a single protocol but various protocols at two layers as shown in the SSL Protocol stack (Figure 12.2).

The SSL Record Protocol lies above TCP and provides basic security services to various higher-layer protocols like Hypertext Transfer Protocol (HTTP).

Three higher-layer protocols that manage SSL exchanges exist as part of SSL. These are: the Handshake Protocol, Change Cipher Spec Protocol, and Alert Protocol [2-4].

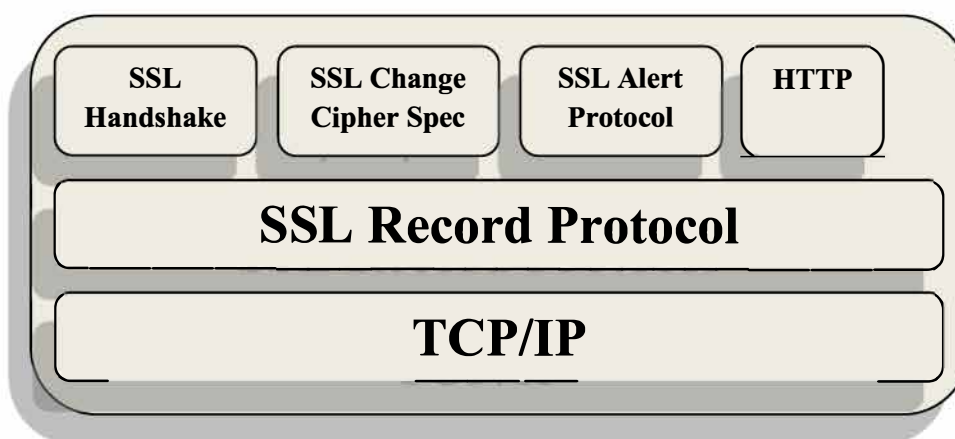


Figure 12.2 SSL Protocol stack

SSL Architectural Design

SSL maintains connection and session:

- **SSL Connection:** A connection is built to provide a particular type of service. It is a transient, peer-to-peer relationship. It is associated with a session.
- **SSL Session:** The Handshake Protocol creates an association between a client and a server which is termed as SSL session. A Session defines a set of cryptographic security parameters. These security parameters can be shared among multiple connections thereby avoiding negotiation of new security parameters for each connection.

A pair of parties (e.g. HTTP client application and server) may have multiple secure connections.

There are a number of states associated with each session. Each state has set of parameters for both receive (read) and send (write). The Handshake Protocol creates pending read and write states during its execution and after its successful completion, the pending states are copied to the current states.

Record Protocol Services in SSL

SSL Record Protocol is mainly used to provide confidentiality and message integrity. These services are provided for each SSL connections. The secret keys required for these services are exchanged and evolved during Handshake Protocol. Confidentiality is provided through encryption of SSL payload by using various ciphers while message integrity is provided through message authentication code (MAC).

The entire operation of SSL Record Protocol is shown in Figure 12.3. The Record Protocol fragments the application message into manageable blocks of fragments. The fragments may optionally be compressed. The protocol then computes MAC (similar to HMAC), concatenates MAC with the fragment. It is then followed by message encryption using shared symmetric secret key. The header (Fig 12.4) containing SSL details, major/minor version, and compressed length is added to the message before putting it into the TCP segment. Receiver performs decryption, MAC verification, decompression, and reassembling of fragments before delivering the message to application layer.

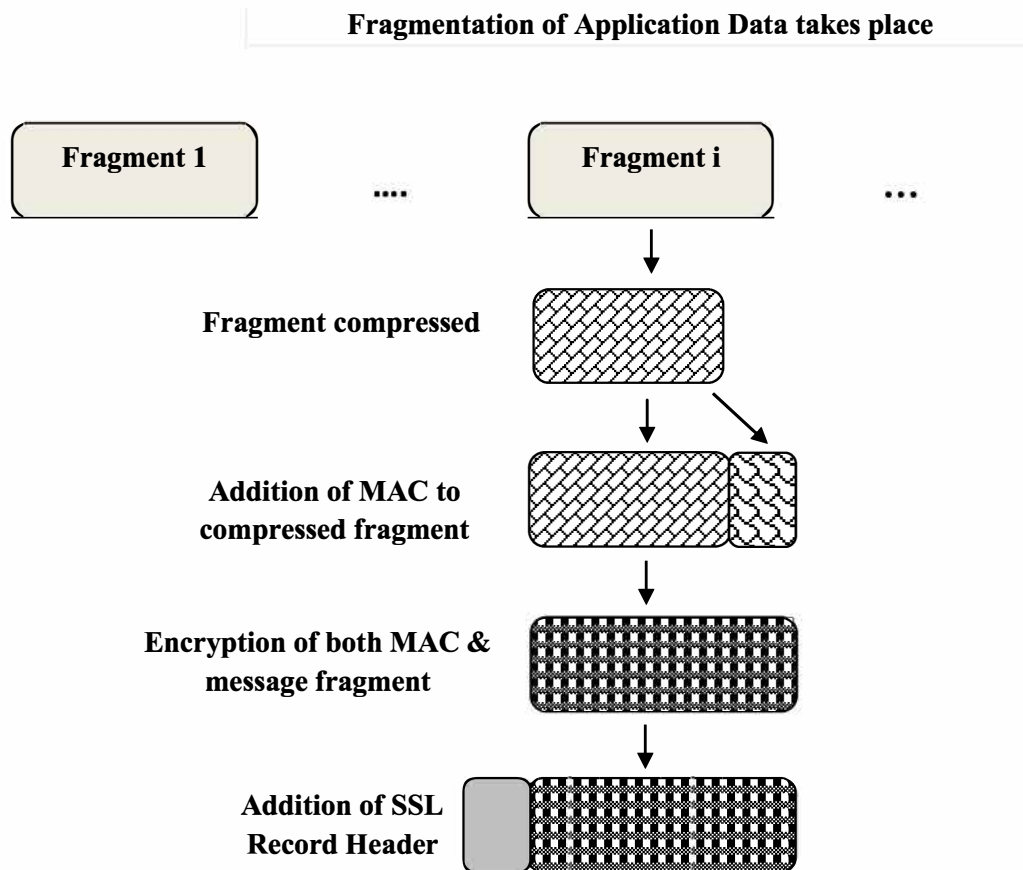


Fig 12.3 Operations (in sequences) of SSL Record Protocol

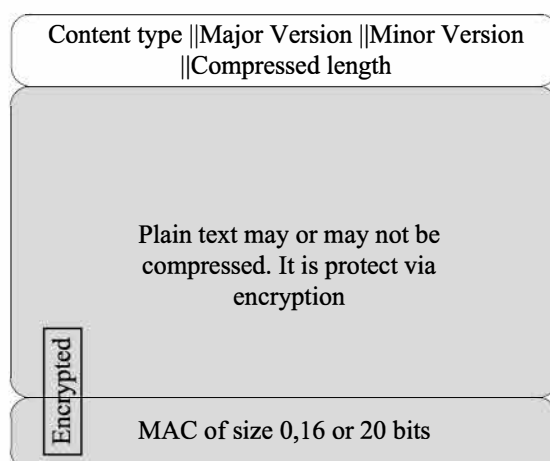


Fig 12.4 Record Format of SSL Protocol

The three SSL-specific protocols utilize the SSL Record Protocol. These protocols are SSL Change Cipher Spec Protocol, SSL Alert Protocol, SSL Handshake Protocol. These are discussed next.

(a) Change Cipher Spec Protocol of SSL

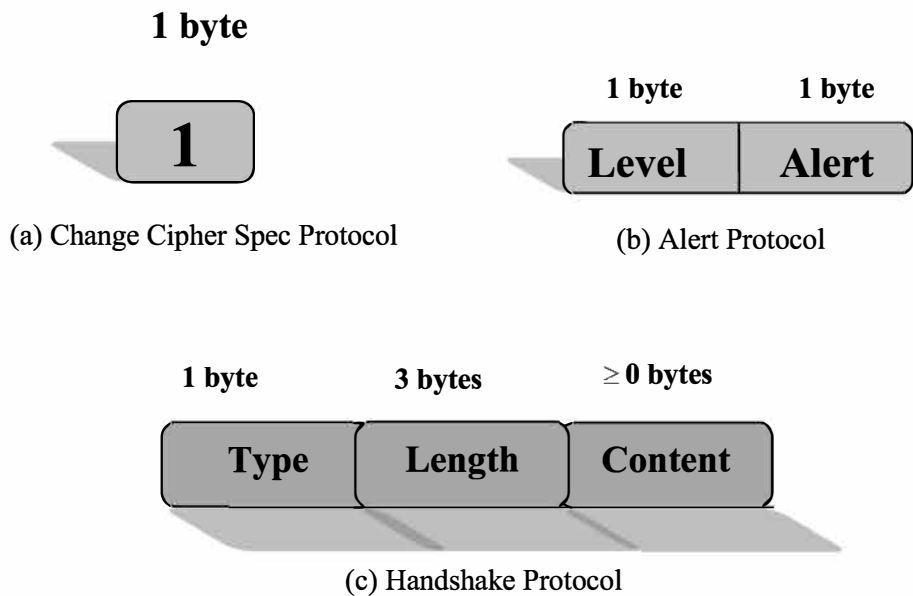


Figure 12.5 SSL Record protocol

The Change Cipher Spec Protocol is the simplest among the three SSL-specific protocols utilizing SSL Record Protocol. It has a single message having a single byte (Fig 12.5 (a)) which is initialized to a value of 1. The message causes copying of pending state into the current state. Thus, main intension of this protocol is to update the cipher suite of the connection.

(b) Alert Protocol of SSL

This protocol is meant for conveying the alerts (related with SSL) to the peers. The alert messages are compressed and encrypted in accordance with the current connection state.

The protocol message (Fig 12.5 (b)) has two bytes, the first byte indicates level of alert i.e., it conveys the severity of the message. The permitted values in first byte are: warning (1) or fatal (2). The second byte is used for indicating the specific alert via codes. These can be listed as:

- Fatal alerts: unexpected message, bad record MAC, decompression failure, handshake failure, illegal parameter.
- Warnings: close notify, no certificate, bad certificate, unsupported certificate, certificate revoked, certificate expired, certificate unknown.

(c) Handshake Protocol of SSL

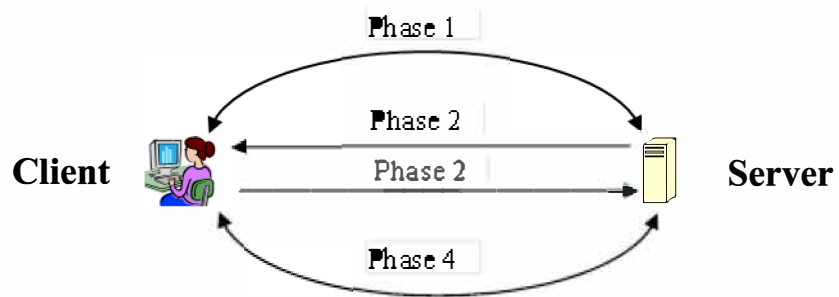
SSL Handshake Protocol is used between client and server before sending any data for the following purposes:

- Server & client authentication to each other.
- Negotiation between server & client for deciding encryption & MAC algorithms.
- Negotiation of cryptographic keys to be used for encryption & MAC calculations.

The protocol's message consists of 3 fields: type (1 byte), length (3 bytes) and content (≥ 0 byte).

The Handshake Protocol has 4 phases (Fig 12.6) during which several messages are exchanged between client and server. These can be briefly described as:

- Establishment of Security Capabilities (Phase 1) - The client initiates a logical connection with server. In this phase, client sends client_hello whereas server replies with server_hello for exchanging the security capabilities to be associated with SSL connection.
- Server Authentication and Key Exchange (Phase 2) - The server sends negotiation parameters (key exchange values) to the client and server certificate for authentication.
- Client Authentication and Key Exchange (Phase 3) - The client verifies the server certificate and checks that the server parameters are acceptable.
- Finish (Phase 4) – Both client and server exchange 'finish' and 'change_cipher_spec' messages. Both copies the pending CipherSpec into the current CipherSpec. This completes the handshake and setting up of a secure connection. Now onwards, the client and server may engage themselves in secure data transfer.



Phase 1

Initialization and Establishment of security capabilities between Client and Server. This includes sharing of protocol version, session ID, cipher suite, compression method and initial random numbers with each other

Phase 2

After establishing initial capabilities, Server may send (i) Its own certificate (ii) Key exchange parameters and (iii) Its request for Client certificate ((i), (ii) and (iii) are optional or situation-dependent messages). Towards end, Server signals end of hello

Phase 3

Depending upon Server's request for certificate, Client sends its certificate (optional or situation-dependent message) along with its own key exchange parameters. Client may also send certificate verification status (optional or situation-dependent message) to indicate that the sever certificate is verified

Phase 4

Both Client and Server exchange their cipher suite specifications to confirm initially exchanged cipher suite specifications. Both update their state including keys updates. Finished messages exchanged in the end indicate (to each other) regarding finishing of handshake protocol.

Fig 12.6 Handshake Protocol

Cryptographic Computations

During SSL Handshake, a shared master secret is evaluated via key exchange. Based upon the shared master secret, other cryptographic parameters like keys for encryption and MAC are also evaluated.

Using secure key exchange, a one-time 48-byte value (384 bits) termed as shared master secret is calculated. The creation is in two stages. Using either RSA or Diffie-Hellman key exchange, first - a `pre_master_secret` is exchanged, second - based upon this `pre_master_secret` a `master_secret` (shared master secret key) is calculated as:

```
master_secret=MD5(pre_master_secret || SHA('A' || pre_master_secret
|| ClientHello.random || ServerHello.random)) ||
MD5(pre_master_secret || SHA('BB' || pre_master_secret ||
ClientHello.random || ServerHello.random)) ||
MD5(pre_master_secret ||
SHA('CCC' || pre_master_secret || ClientHello.random ||
ServerHello.random))
```

Here, `ClientHello.random` and `ServerHello.random` represents the client and server nonce values exchanged via initial hello messages.

A, BB, CCC represents constant values.

This shared master secret key is calculated by both the peers.

CipherSpecs require client's and server's: MAC secret, symmetric encryption key and initial vector (IV). All of these are generated from the master secret by hashing the master secret and then selecting the sequence of sufficient length.

TLS (Transport Layer Security)

TLS is standardization initiative for SSL. It is defined in RFC 2246. RFC 2246 is similar to SSLv3, with small differences like change in record format version number, use of HMAC for MAC calculation, use of pseudo-random function for expanding secrets, use of additional alert codes, some changes in supported ciphers, changes in certificate types & negotiations and changes in crypto computations & padding [2-5].

Check your progress 1

- a. How can we ensure non-fraudulent transactions on the web?
- b. Where is SSL placed? What it encrypts and what does it provides?

12.5 OVERVIEW OF SECURE ELECTRONIC TRANSACTION PROTOCOL

Secure Electronic Transaction (SET) provides security specifications describing set of security protocols and formats for credit card transactions over the Internet.

The original specification was developed by group of companies like IBM, Microsoft, Netscape, RSA, Terisa, and Verisign upon a call from MasterCard and Visa in the mid 90s. This specification (summing to a total of 971 pages) was written in three books (in 1997) namely, Business Description, Programmer's Guide and Formal Protocol Definition. By the year 1998, the SET compliant products were available.

SET provides three services: secure communication, trust and privacy. As many parties are involved during communication, SET uses secure communication channel between them for securing the data exchange. It uses X.509v3 digital certificates for providing trust. It provides information to only the trusted parties involved in the transaction for enhancing privacy [2][6][7].

SET defines business requirements, features, and the participants for secure credit card transactions and processing. These are presented next.

SET Business Requirements

The following business requirements are imposed on SET protocol:

- (i) Provide confidentiality to both: payment and ordering information.
- (ii) Provide data integrity to each and every data packet.
- (iii) Linkage between user of card and his account such that it is ensured that user of the card is legitimate and that he own the corresponding account.
- (iv) Linkage between merchant and his financial institution such that it is ensured that merchant can accept transaction via credit card.
- (v) Use of best available security practices and system design techniques for protecting the users and their transactions.
- (vi) The protocol should function separately independent of any existing security methods.
- (vii) The protocol should assist and support the interoperability issues.

For meeting (i) and (ii) SET uses encryption and digital signatures thereby ensuring:

All information between the communicating parties is safe, authentic and is accessible only by the authorized users. This also reduces the chances of fraud by either sender or receiver or by any third party.

For meeting (iii) and (iv) digital signatures and digital certificates are used. Thus, card holders only identify the merchants for transactions while the merchants have confidence that the card holder is not fake and has an associated account.

For meeting requirement (v), SET uses well tested, reliable and highly secure algorithms. For meeting requirement (vi), SET functions independently i.e., without interfering with IPsec and SSL/TLS. For

meeting requirement (vii), SET defines formats and methods that are independent of any hardware or software.

Striking Features of SET

The striking features of SET can be listed as:

- **Information Confidentiality:** In SET, DES symmetric encryption is used to protect the following information from access by others.
 - (i) Cardholder account and other details.
 - (ii) Payment information.
 - (iii) Cardholder's credit card number; even merchant is prevented from learning it. The number is supplied only to the issuing bank.
- **Data Integrity:** For protecting the payment information sent from cardholders to merchants by any alterations, SET uses either RSA digital signatures with SHA-1 or HMAC using SHA-1.
- **Cardholder account authentication:** SET enables merchants to verify that a cardholder is a legitimate user of a valid card account number. SET uses X.509v3 digital certificates with RSA signatures for this purpose.
- **Merchant authentication:** SET enables cardholders to verify that a merchant has a relationship with a financial institution allowing it to accept payment cards. Again, SET uses X.509v3 digital certificates with RSA signatures for this purpose.

In comparison to IPSec and SSL/TLS, SET does not give any options for selecting among various cryptographic algorithms. This is due to the fact that SET is a single application and is specifically used for one purpose that of securing credit card based transactions whereas IPSec and SSL/TLS support a range of applications and provides general security to applications.

SET Participant Details

SET has 06 participants (Figure 12.7) [2]. These are:

- **Cardholder** (also termed as consumer or purchaser) - owns a credit card (e.g., MasterCard, Visa) issued by an issuer and is responsible for interaction with merchant via Internet enabled devices.
- **Merchant** (may be a person or an organization) - enjoys relationship with an acquirer and who has goods or services to sell on Internet (via website or by E- mail). He accepts payment cards of the cardholders.
- **Issuer** (financial institution - a bank) - who is responsible for the payment of the debt of the cardholder. Issuer opens the account and issues the payment card to the cardholder.
- **Acquirer** (financial institution) - establishes an account with a merchant and eases the merchant's work by dealing with multiple bankcard associations or with multiple individual

issuers. Thus, merchant is not supposed to interact and involve in transaction with different individual issuers.

The functions/processing provided by the acquirer are:

- (1) Inform the merchant that the (i) given card account is active and (ii) proposed purchase does not exceed the credit limit.
 - (2) Provides electronic transfer of payments to the merchant's account. These electronic funds transfers are later reimbursed by the issuer using payment network.
- Payment gateway (either acquirer or a designated third party fits in this role) - processes merchant payment messages. The merchant interacts with the payment gateway (via SET message) over the Internet. The payment gateway links (directly or indirectly) to the financial processing system of the acquirer.
 - Certification authority (CA) (or a hierarchy of CAs) - who is a trusted third party that issues public-key certificates (X.509v3) for all other major parties (cardholders, merchants, and payment gateways) in SET.

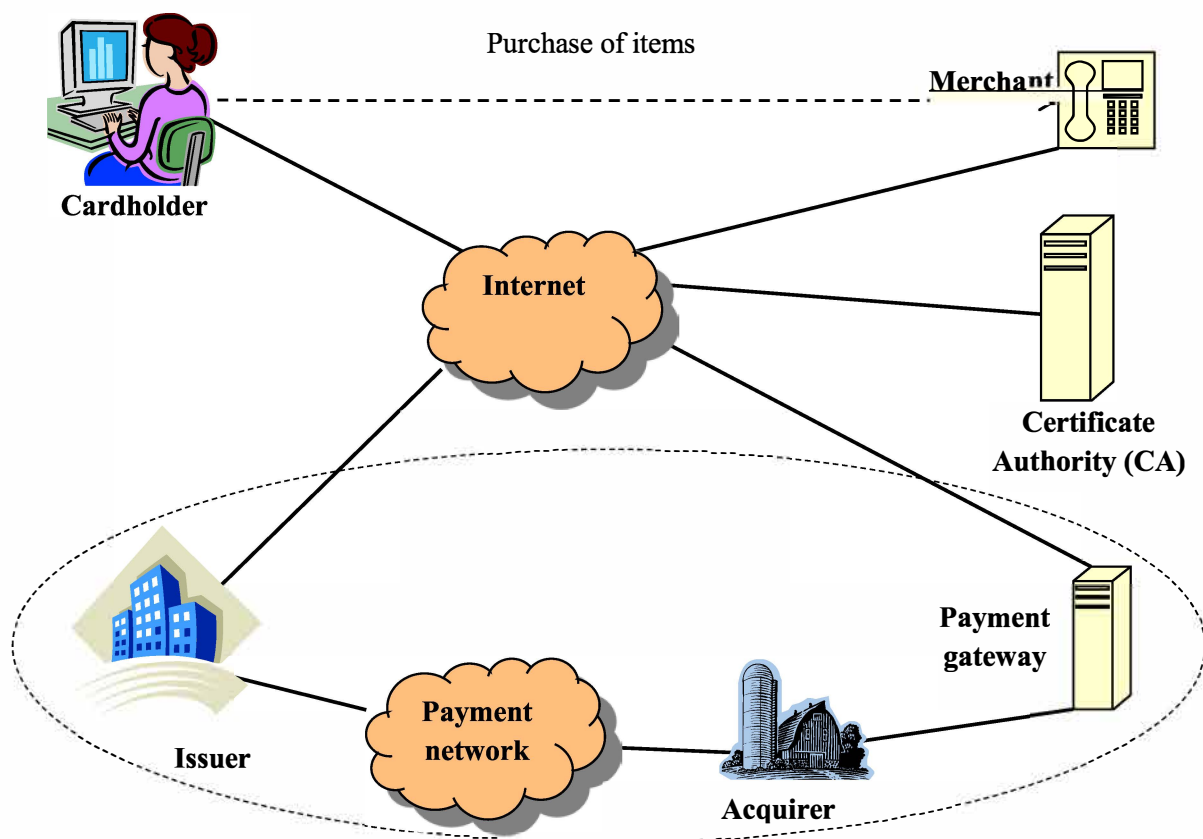


Figure 12.7 Participants (Components) in a Secure Electronic Commerce

STEPS involved in SET use are:

1. The first step is opening of customer account in a bank that supports electronic payment/SET and obtaining credit card (MasterCard, Visa etc.) for payment.
2. Bank issues a digital certificate (X.509v3) to the customer after verifying the customer's identity. The certificate is signed by the bank and it contains customer's RSA public key along with the expiration date. It establishes a relationship between the customer's key pair and credit card.
3. Merchants own two certificates for two different public keys: one for signing messages and other for key exchange. They also keep payment gateway's public-key certificate.
4. The customer browses the merchant's Web site and selects items. The selected list of items is send to the merchant or submitted online. The merchant replies with an order form containing order number & associated details (list of items, their price and total price) along with its digital certificate.
5. The customer verifies the merchant using merchant's certificate.
6. The customer sends certificate and order form to the merchant for confirming the purchase of the items. The merchant confirms the customer via certificate and considers the order. Customer also sends payment information (containing credit card details) in encrypted form to the merchant. The encryption is done so that merchant cannot read payment related information.
7. The merchant forwards the payment information to the payment gateway for verifying - whether the customer has available credit to make the current purchase or not.
8. Upon verification of the customer's request at the payment gateway, merchant sends order confirmation.
9. The merchant provides the demanded goods or service to the customer.
10. The merchant requests to the payment gateway regarding payment processing.

Dual Signature (DS) = Encryption of payment order message digest (POMD) using customer's private signature key (K_C)
 $= E_{K_C}(POMD)$

POMD = Hash of (PI message digest (PIMD) concatenated with OI message digest (OIMD))
 $= H(PIMD || OIMD)$

PIMD = Hash of PI = $H(PI)$

OIMD = Hash of OI = $H(OI)$

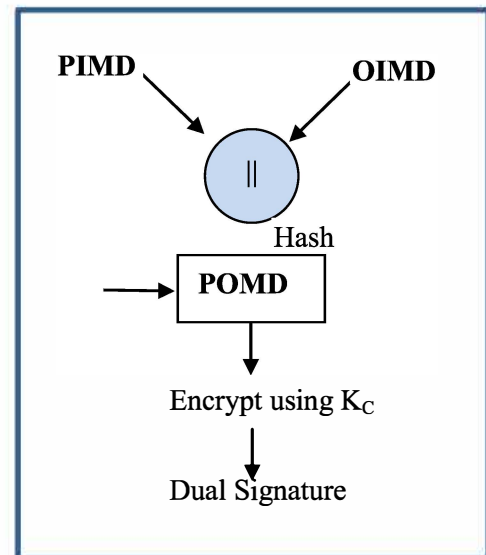


Figure 12.8 Process of creating a Dual Signature

Concept of Dual Signature (DS)

One of the major requirements of SET is that the customer's order information should not be visible to the bank and the payment information (e.g., customer's credit card number) should not be visible to merchant. For this requirement, SET introduces the concept of dual signature (Figure 12.8). The dual signature is used to link two messages of two different recipients (Bank and merchant). The message corresponding to order information (OI) is sent to the merchant whereas message corresponding to the payment information (PI) is sent to the bank. The following is ensured:

- (1) The merchant remains unaware of the PI and the bank remains unaware of details of the customer's order (OI).
- (2) The customer privacy is enhanced by keeping PI and OI separate.
- (3) OI and PI are linked with each other and hence can be used under dispute cases. Using this linkage the customer can prove that PI corresponds to this OI and not for some other order. Thus, linkage eliminates the confusion that a particular PI forwarded by merchant to bank corresponds to which of the OI's (in case different orders are given by same customer to this merchant).

As shown in Figure 12.8, the process of purchase of goods and services is initiated by the customer who takes individual hashes of PI and OI, then concatenates them and again takes the hash of the resultant. The final hash is encrypted with customer's private key and this finally encrypted hash is termed as dual signature (DS). Considering K_C as the customer's private key, DS is calculated as:

$$DS = E_{K_C} [H(H(PI)||H(OI))]$$

Processing of Payment in SET

SET supports many transaction types, major among them are:

Purchase request: Customer's message to merchant. It contains order and purchase information.

Payment authorization: Refers to merchant and payment gateway interactions for authorizing a purchase via a given credit card account for payment amount.

Payment capture: Refers to merchant's request to the payment gateway regarding payment.

These are considered next in details.

Purchase Request Transaction in SET

Steps taken in the preliminary phase between customer and merchant without involving SET:

- (i) Customer performs browsing, selecting, and ordering of goods/services to merchant.
- (ii) Merchant returns a completed order form.

The entire purchase request (SET enabled) interaction involves four messages. The customer having credit card also requires public key certificates of the merchant and the payment gateway. For this first two messages i.e., Initiate Request and Initiate Response are used.

Initiate Request: Customer's message requesting the merchant's certificate. It includes:

The brand of the credit card – required to link at the time of payment with the card issuer.

An ID - to identify the request and its corresponding response.

A nonce - to ensure timeliness and must be returned by merchant in the response.

Initiate Response: Merchant's reply message containing signed response (signed by merchant using his private key) and certificates. It includes:

The customer's nonce – to indicate that this response corresponds to previous request.

A fresh nonce - to ensure timeliness and must be returned by customer in the next message.

A transaction ID – to identify the purchase transaction.

This message also includes merchant and payment gateway certificates.

Purchase Request: Customer's Purchase Request message given to merchant.

Preprocessing:

- (i) Customer verifies the merchant and gateway certificates using CA signatures
- (ii) Creates the OI and PI containing transaction ID (send by the merchant in Initiate Response)
- (iii) An order reference generated in the preliminary phase before starting the SET enabled interaction.

After completing preprocessing tasks, the customer/cardholder generates a one-time symmetric encryption key, K_s and creates Purchase Request message (Fig 12.9) containing:

1. Purchase-related information to be forwarded to the payment gateway by the merchant and it contains the following:

- (i) The PI.
- (ii) The Dual Signature (DS).
- (iii) The OI Message Digest (OIMD).

(i), (ii) and (iii) are encrypted using one time secret key K_s . The OIMD is used to verify the DS. The bank possesses DS, PI, OIMD along with the customer's public key and can calculate:

- (i) $H(H[PI]||OIMD)$
- (ii) $D_{P_c}(DS)$ where, P_c is the customer's public key and D stands for decryption.

If (i) and (ii) are equal, the DS is verified by the bank.

The key K_s is encrypted using payment gateway's public key and is hence, hidden from the merchant. Thus, merchant cannot read PI. The encrypted K_s is termed as digital envelope as it must be decrypted for reading other items in the message.

2. Order-related information required by the merchant and it consists of the following:

- (i) The OI, send in clear without encryption.
- (ii) The DS.
- (iii) The PI Message Digest (PIMD).

The PIMD is used to verify the DS. The merchant possesses DS, OI, PIMD along with the customer's public key and can calculate:

- (i) $H(PIMD||H[OI])$
- (ii) $D_{P_c}(DS)$ where, P_c is the customer's public key and D stands for decryption.

If (i) and (ii) are equal, the DS is verified by the merchant.

3. Cardholder public key certificate required by merchant and payment gateway.

From (1) – (3), it is clear that:

1. The merchant receives OI and verifies DS.
2. The bank receives PI and verifies DS.
3. The customer ensures the linkage between OI and PI and can prove it later. Also, neither merchant nor bank can change this linkage by changing either OI or PI

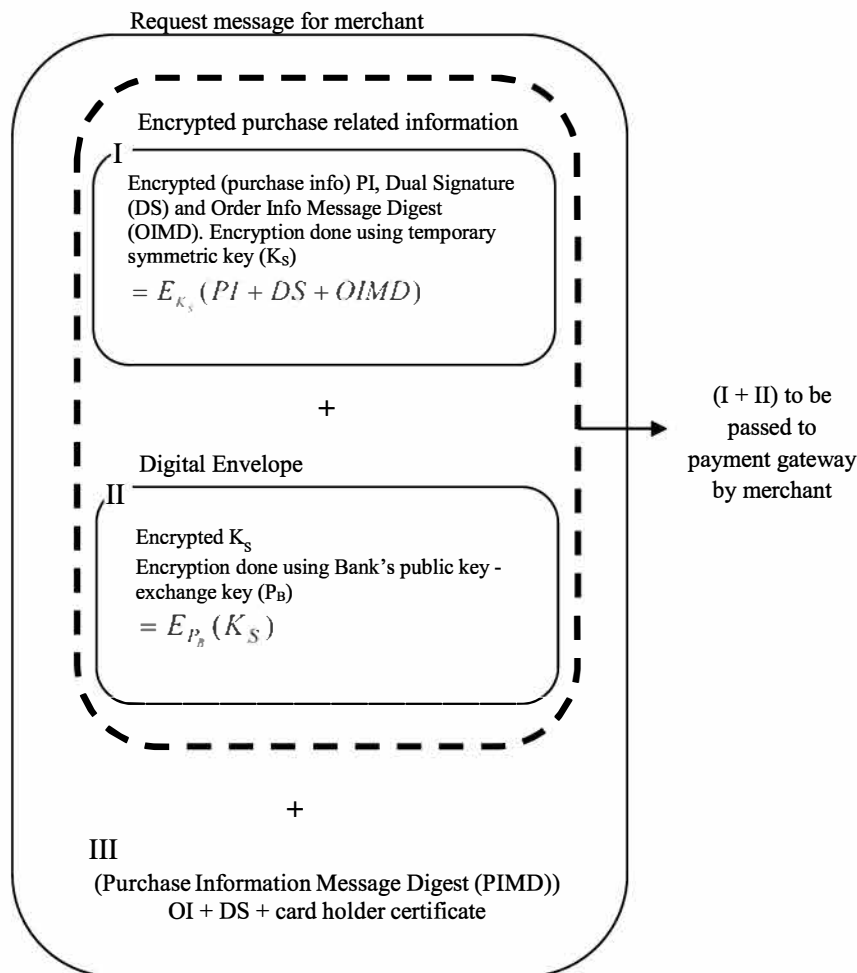


Figure 12.9 A Cardholder Purchase Request details

Processing steps executed by the merchant upon receiving the Purchase Request message (Fig 12.10):

1. Verification of the cardholder's certificates by verifying CA signatures and extraction of customer's public key.
2. Verification of the DS (using the customer's public key), indicating that the message was indeed sent by the customer and is not tampered in between.
3. OI processing and forwarding of the PI to the payment gateway for authorization.
4. Reply the customer via purchase response.

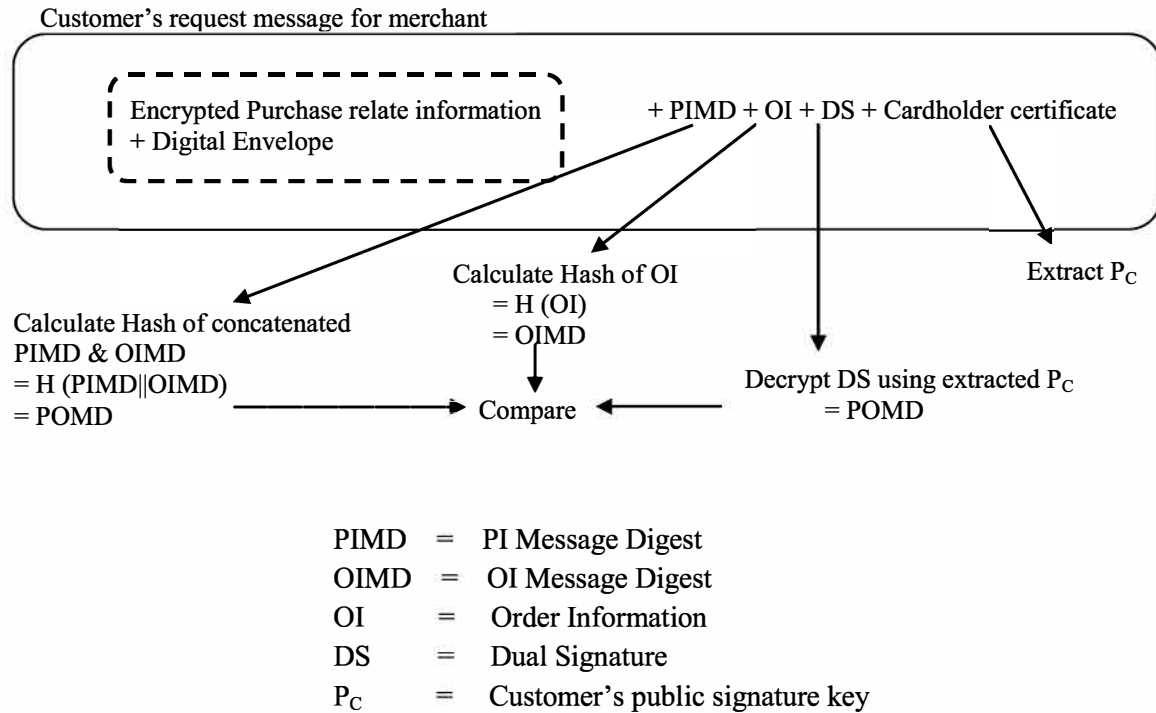


Figure 12.10 Verification of Customer Purchase Request by Merchant

Purchase Response: Merchant sends purchase response message to customer which contains:

(i) a response block and (ii) merchant's signature certificate. Former is signed by merchant using his private key and is used to acknowledge the purchase order. It also contains the transaction number. Latter contains the merchant's key which is used to verify the signature.

Upon receiving the purchase response message, customer verifies the merchant's certificate. It is then followed by verification of the merchant signature on the response block. Customer may display the message or update the order status based on the response conveyed via response block.

Payment Authorization Transaction in SET

The payment authorization is a process governed by payment gateway and it ensures (i) approval of the transaction by the issuer and (ii) surety regarding payment to the merchant.

Because the transaction is approved by the gateway, the merchant can supply services or goods to the customer. The payment authorization involves: Authorization Request message and Authorization Response message. Former is send by merchant to the payment gateway while latter is send by payment gateway to the merchant.

The Authorization Request message contains:

1. Purchase-related information received from the customer containing PI, DS, OI message digest (OIMD) and digital envelope.

2. Authorization-related information generated by the merchant having an authorization block and a digital envelope. Former has the signed transaction ID encrypted with a one-time symmetric key generated by the merchant. The transaction ID is signed using merchant's private signature key. Latter contains one-time symmetric key encrypted by payment gateway's public key-exchange key.
3. Three Certificates: customer's/cardholder's signature key certificate (for verifying DS), the merchant's signature key certificate (for verifying merchant's signature), and the merchant's key-exchange certificate (to be used in payment gateway's response).

Upon receiving Authorization Request, the payment gateway executes following operation sequence:

1. All three certificates are verified.
2. Decrypts the digital envelope payment gateway's private key-exchange key.
3. Obtains the symmetric key from digital envelope and then decrypts the authorization block to obtain signed transaction ID.
4. Verifies the merchant's signature from the signed transaction ID.
5. As done in step 3 extract the symmetric key from digital envelope of the payment block and use it to decrypt the payment block.
6. Verifies the DS from payment block
7. Compares the transaction ID received from the merchant with that received from the customer via PI.
8. Upon successful comparison, requests an authorization from the issuer.
9. Upon receiving authorization from the issuer, gateway creates and sends an Authorization Response message to the merchant.

The Authorization Response includes the following elements:

1. Authorization-related information generated by the payment gateway having the signed authorization block and digital envelope. Signature on authorization block done using gateway's private signature key while encryption is done using a one-time symmetric key generated by the gateway. The one-time symmetric key is encrypted by merchant's public key-exchange key to form the digital envelope.
2. Capture token information having a signed, encrypted capture token and a digital envelope. This token is returned as such by the merchant for payment in the payment request.
3. Gateway's signature key certificate.

After being authorized from the gateway, the process of goods or service delivery to the customer can proceed ahead.

Payment Capture Transaction in SET

For receiving payment, the merchant and gateway exchanges - capture request and a capture response message (together termed as Payment Capture Transaction).

The Capture Request message is created by the merchant and it contains: (i) Capture request block having payment amount and the transaction ID. This capture request block is signed and encrypted, (ii) Encrypted capture token received via Authorization Response from the gateway and, (iii) Merchant's signature key and key-exchange key certificates.

Upon receiving Capture Request message, the payment gateway - decrypts and verifies (i) the capture request block and (ii) the capture token block. Both capture request and capture token are checked for consistency of information. A clearing request is sent depending upon success of checking to the issuer over the private payment network for transferring the funds to the merchant's account.

Finally, the gateway sends payment notification to the merchant using Capture Response message which contains (i) a signed and encrypted capture response block and, (ii) gateway's signature key certificate. The capture response is stored by the merchant (software) for future reconciliation with the acquirer's details regarding payment [2].

Check your progress 2

- a. How does customer encrypt credit card number in SET protocol?
- b. In SET protocol how does a customer send a purchase order?
- c. How does SET provide authentication?
- d. Why are the PIMD and OIMD combined? What is its name given to the combined form?
- e. How the payment information is hidden from the merchant.

12.6 Summary

Need for web security is constantly increasing. In this unit prime focus is on two major security protocols that have concentrated primarily on enhancing the E-commerce related activities. First one is Secure Socket Layer (SSL)/Transport Layer Security (TLS) protocol that provides security above TCP. Architectural details of SSL/TLS including SSL Record Protocol which provides basic security services to various higher-layer protocols like HTTP are presented. SSL-specific protocols namely, SSL Change Cipher Spec Protocol, SSL Alert Protocol and SSL Handshake Protocol that also utilize the SSL Record Protocol are also discussed.

Second security protocol considered in the unit is Secure Electronic Transaction (SET). It describes set of security protocols and formats for credit card transactions over the Internet. Requirements, Key features and participants pertaining to SET are discussed. Specific

concepts like dual signatures, payment processing, authorization etc. are also covered here.

Terminal Questions

1. *State the conditions under which IPsec or SSL is used?*
2. *Differentiate between SSL connection and session.*
3. *Explain the functioning of SSL specific protocols.*
4. *What is purpose and utility of SSL Record Protocol?*
5. *List and define the SET participants.*
6. *State the importance of dual signature in SET protocol.*
7. *Write the steps involved in SET authorization.*
8. *What role is played by digital signature in SET and SSL?*

References:

1. Garfinkel, S., and Spafford, G. Web Security & Commerce. Cambridge, MA: O'Reilly and Associated, 1997.
2. Stallings, William, "Chapter 17: Web Security", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
3. Forouzan, Behrouz A., "Chapter 17 - security at the Transport Layer: SSL and TLS", Cryptography & Network Security, Tata McGraw Hill, Special Indian Edition, 2007.
4. <https://www.sans.org/reading-room/whitepapers/protocols/ssl-tls-beginners-guide-1029>, accessed on 10.01.2016.
5. https://en.wikipedia.org/wiki/Transport_Layer_Security accessed on 09.01.2017.
6. PKI Case studies, PKI Outreach Programme (POP) Nationwide Awareness Programme, Centre for Development of Advanced Computing (C-DAC), Electronics City, Bangalore, 2012.
7. https://en.wikipedia.org/wiki/Secure_Electronic_Transaction accessed on 23.01.2017



**Uttar Pradesh Rajarshi Tandon
Open University**

Master of Computer Science

**MCS -119
INFORMATION AND NETWORK
SECURITY**

Block

4

INTRUDERS AND VIRUSES

Unit 13	255-270
----------------	----------------

Intruders

Unit 14	271-288
----------------	----------------

Malicious Programs

Unit 15	289-312
----------------	----------------

Firewall

Course Design Committee

(Prof.) Ashutosh Gupta **Chairman**
Director (In-charge)
School of Computer and Information Science, UPRTOU,
Prayagraj

Prof. R. S. Yadav **Member**
Department of Computer Science and Engineering
MNNIT-Allahabad, Prayagraj

Dr. Marisha **Member**
Assistant Professor (Computer Science),
School of Science, UPRTOU, Prayagraj

Dr. C. K. Singh **Member**
Lecturer
School of Computer and Information Science, UPRTOU, Prayagraj

Course Preparation Committee

Dr Rajeev Singh **Author**
Assistant Professor
Department of Computer Engineering
GB Pant University of Agriculture and Technology
Pantnagar, Uttarakhand

Prof. Abhaya Saxena **Editor**
Head, Dept. of Computer Science
Dev Sanskriti Vishwavidyalaya
Hardwar, Uttarakhand

Prof. Ashutosh Gupta
Director (In-charge),
School of Computer and Information Science
UPRTOU, Prayagraj

Dr. Marisha **Coordinator**
Assistant Professor (Computer Science)
School of Science
UPRTOU, Prayagraj

© UPRTOU, Prayagraj. 2023
ISBN : 978-93-83328-27-7

All Rights are reserved. No part of this work may be reproduced in any form, by mimeograph or any other means, without permission in writing from the Uttar Pradesh Rajarshi Tondon Open University, Prayagraj.

Printed and Published by Vinay Kumar, Registrar, Uttar Pradesh Rajarshi Tandon Open University, 2023.

Printed By: K.C.Printing & Allied Works, Panchwati, Mathura -281003.

UNIT - 13

Intruders and Viruses

Structure

- 13.0 Introduction
- 13.1 Objectives
- 13.2 Realizing an Intruder
- 13.3 Techniques against Intrusion
- 13.4 Protecting the Password
- 13.5 Ways for Password Selection
- 13.6 Intrusion Detection Methodologies
- 13.7 Summary
- 13.8 Terminal Questions

13.0 INTRODUCTION

The Information systems and networks are usually targeted by unauthorized users in an unauthorized manner. This is referred to as intrusion. Two strategies are used to protect against intrusion attacks: Prevention and Detection. Prevention is very difficult and it mainly rests upon traditional techniques like use of passwords to differentiate between authorized and unauthorized users. Once intrusion has occurred in a system, it is advisable to reduce its impact. An intrusion detection system (IDS) is one such method for reducing the impact of intrusion. It is like early warning system that provides warnings and alerts against unauthorized access. An IDS is based upon detecting unusual usage patterns and activities. In nutshell, this unit deals with the topics related with intruders, intrusion prevention, intrusion detection and password management.

13.1 OBJECTIVES

After the end of this unit, you should be able to:

- learn about the intruder's behavior, characteristics and types.
- ponder upon the available solutions in case intrusion takes place.
- understand the fundamentals of Intrusion Detection System (IDS).
- apprehend the importance of password techniques and learn

various password management strategies.

13.2 REALIZING AN INTRUDER

As identifies by Anderson [1], there are three classes of intruders (users):

- Masquerader (an outsider) is the one who gains unauthorized access to the system.
- Misfeasor (an insider) is the one who gains unauthorized access to data, programs, or resources of an organization.
- Clandestine user (may be outsider or insider) is the one who seizes supervisory control of the system and utilizes the control to bypass the audit and access control mechanisms.

Intruder attacks lie among: simple exploration, reading of privileged data, performing unauthorized modifications, system malfunction or disruption etc.

Some of the intrusion cases as per Grans *et al.* (2004) [2] can be listed as:

- Compromising an e-mail server.
- Vandalizing a Web server.
- Password identification.
- Stealing credit card numbers by copying a part of database.
- Viewing personal details and information pertaining to employees and colleagues without authorization
- Sniffing for usernames and passwords on a network.
- Gaining internal network access via insecure communication devices.
- Pretending on behalf of someone for resetting password and hence, knowing the valid user's password.
- Use of logged-in system without permission.

The intruders may also be classified into: Benign (tolerable) and Malign (Serious).

Benign intruders are usually motivated by the thrill or prestige; look for some opportunities or security holes in the systems and then the discovered information is shared among their community. The quality discovery raises the intruder's status among the hacker community. Such activities may consume some resources and may slow performance of the system, still might be tolerable to some extent. In practice, it is really difficult to identify the intensions of the intruders and hence to know whether they are benign or malign.

For restricting Intruder's activities the following two methods are commonly used:

1. Intrusion Prevention Systems (IPSs)

2. Intrusion Detection Systems (IDSs)

Some countermeasures other than these IPSs and IDSs may be utilized by organizations like use of virtual private networks (VPNs) and restriction of remote logins communicating with specific IP addresses.

Due to growing incidences related with system intrusion, collaborative system having a centralized computer emergency response team (CERT) is established in almost every country. CERTs are attaining popularity. The main functionality any CERT is to find and collect system vulnerabilities. The information gathered is disseminated to systems managers & administrators so as to reduce the impact of the attack.

Once an attack or an intrusion technique is discovered, the system is strengthened against it by application of software patches. The intruder also learns this through the public CERT reports. Thus, the intruder is forced to change his techniques and intrusion patterns i.e., the intruder is now supposed to find out new vulnerabilities that are not noticed until recently. Hence, the techniques & behavior patterns of intruders are constantly changing to bypass the existing detection and security countermeasures. However, it is still observed that an intruder's behavior varies typically from that of normal users. One such example representing the intruder's behavior and actions during breaking into a financial organization [3] is shown (in steps) below:

A sequence indicating the behavior of a typical intruder:

1. Listing of target via IP lookup tools
2. Identification of accessible services on target
3. Identification of potentially vulnerable services on target
4. Application of password guessing method (may be brute force method)
5. Installation of remote administration tool
6. Capture Admin password by observing his login credentials
7. Access remainder of network using the captured password

The following points are to be noted in the above example:

The organization's network was running unprotected services even though some of them were not even needed.

A remote control application like 'pcAnywhere' (by Symantec Inc.) that provides secure connection to remote devices may be used to break-in. Thus, a tool used for providing security may itself be used during attack.

The organization's network does not support intrusion detection system. The intrusion in the network was only detected when someone saw the cursor moving files on the system.

Criminal Enterprise Behavior

A Criminal Enterprise is formed from loosely affiliated gangs of hackers/intruders. The participating hackers/intruders are usually young persons who do business on the Web and use underground forums like DarkMarket.org, theftservices.com etc. for communication, coordination and trading. Such Enterprise has following typical features:

- The hackers/intruders involved often target credit cards and after using these cards for crime or purchasing, post them for access and use by others. This is done to make investigation and tracing difficult.
- Unlike traditional hackers/intruders, criminal hackers usually have specific targets.
- After penetration, the hacker/intruder acts quickly for gaining desired information and immediately moves out of scene.
- Though IDS / IPS may be used in such situations but these have proved less effective due to quick actions of the attacker.

For protecting sensitive data from such attacks, database encryption should be used for credit card and other important information of the user. The site hosting should be done on dedicated server instead of shared server and monitoring of the provider's security services should be done closely and periodically.

The behavior of the criminal intruder can be summarized as:

- Criminal intruder activities are harder to identify and check due to their quick action.
- They usually exploit vulnerable ports and installs Trojan horses for back door re-entry.
- They often use capturing of passwords via sniffers.
- They do not stick around until noticed.
- They commit very few or negligible amount of mistakes.

Insider Attack Behavior

An insider attack is one of the most challenging attacks and is very difficult to detect and prevent. It has following typical features:

- The organization's employees themselves are involved in attacks. They have access, details & knowledge of organization's data & resources.
- These are mainly motivated by revenge (by terminated employee) or by sense of entitlement (where employees felt entitled to take extra office supplies and data for home use) while moving to competitor.
- In the insider attacks, IDS / IPS have restricted utility. The following are much helpful:

Enforcing least privilege, monitoring of logs, protecting sensitive resources via strong authentication, deletion of employee's computer and network access upon termination and creating a mirror image of employee's hard drive before it can be reissued.

The behavior of the insider can be summarized as:

- Creating network accounts (even for friends).
- Accessing the accounts and applications not used for daily activities.
- Searching new options via e-mailing former and prospective employers.
- Secret instant-messaging chats and conversations.
- Hovering around on sites related to disgruntled employees.
- Involved in larger downloads and copying.
- Access and using the network during odd hours.

13.3 TECHNIQUES AGAINST INTRUSION

The intruder mainly targets to gain access to system or targets to increase privileges for himself on the targeted system.

For achieving these goals, the intruder identifies and uses vulnerabilities that allow a user to execute code and acquire protected information such that entry into the system is granted. One of the common techniques by which attackers try to gain access into the system is buffer overflow attack. In this attack a program running with privileges is exploited.

The protected information is usually a user password. It is usually targeted by the attackers/intruders. Having obtained the user's password, an intruder can enter into the system and use all the privileges & power delegated to the legitimate user.

Two methods exist for acquiring the passwords: guessing and capturing.

Password Guessing

Password guessing is applicable mainly to password files containing hashed passwords. The attacker first copies the password file from the system such that target is unaware about it and then utilizes off-line password guessing for gaining the passwords. Intruders can also guess passwords online through actual logins but such login attempts are usually restricted by the system's operating system and upon abnormal number of failed logins, the operating system can trigger security mechanism like deactivating logins for some predetermined amount of time. This online method is valid when the user's passwords are weakly/poorly chosen.

Password Capture

Password capture involves: watching password entry over shoulder of the user by attacker, using a Trojan horse program for information collection, monitoring network login over insecure protocols like telnet, FTP etc., extracting recorded login information (from web history/cache or last number dialed from memory of the phone). By captured valid login/password a user can be impersonated and hence, users need to be educated for installing & using proper precautions and countermeasures. Such educated users may act carefully while at work and may check: regarding presence of who is around them, that they interact with the trusted systems only, that unknown softwares (major source of Trojan horse) are not used by them, secure network connections/protocols like HTTPS, SSH, SSL are used and also that the browser/phone histories are flushed periodically.

13.4 PROTECTING THE PASSWORD

Passwords provide primary authentication to users i.e., users who enter correct combination of login identifier (or name) and password credentials are considered authentic and are permitted to access the services of the system. In short, the login/password represents first line of defense in any system. The login id serves the purpose of determining the permissible privileges to the user while the password serves the purpose of id authentication.

The Passwords are stored in a dedicated and separate file. They are either kept in encrypted form using encryption algorithm like multiple DES or in hashed form using a cryptographic hash function like MD5. Thus, passwords are never stored in clear. An example password scheme of UNIX system is shown in **Figure 13.1**. The scheme uses a password file, indexed by user id. The password entered by user along with the selected 12 bit salt value is provided as input to the crypt(3) function. crypt(3) function is based upon DES algorithm and it produces a fixed size hash value. This hash value is stored along with clear text salt value in the password file. Whenever user login into the system, UNIX system searches the password file based upon the user's id as an index into the password file and retrieves user's salt value. System uses this salt value and password entered as input to crypt(3) function. The output of crypt(3) is matched against the corresponding stored hash value in password file. A match will mean that password must be accepted by the system [4][5].

The file containing these passwords hashes should be protected from intruders. Hence, it needs access control protections so that its copying becomes difficult for the intruders.

Password Studies

Two of the studies done at Purdue University in 1992 and by Klein in 1990 respectively demonstrated that users generally tend to choose passwords that are: short, poor in strength and guessable.

The users usually constrained by memory limits, try to select memorable passwords like their own name, their street name, common dictionary words etc. Due to this constraint, the password cracking becomes straight forward by an intruder. Thus, a strategy that forces users to select

passwords that cannot be guessed easily is required. The goal of such strategy is to eliminate guessable passwords on the other hand permitting users to have a memorable password.

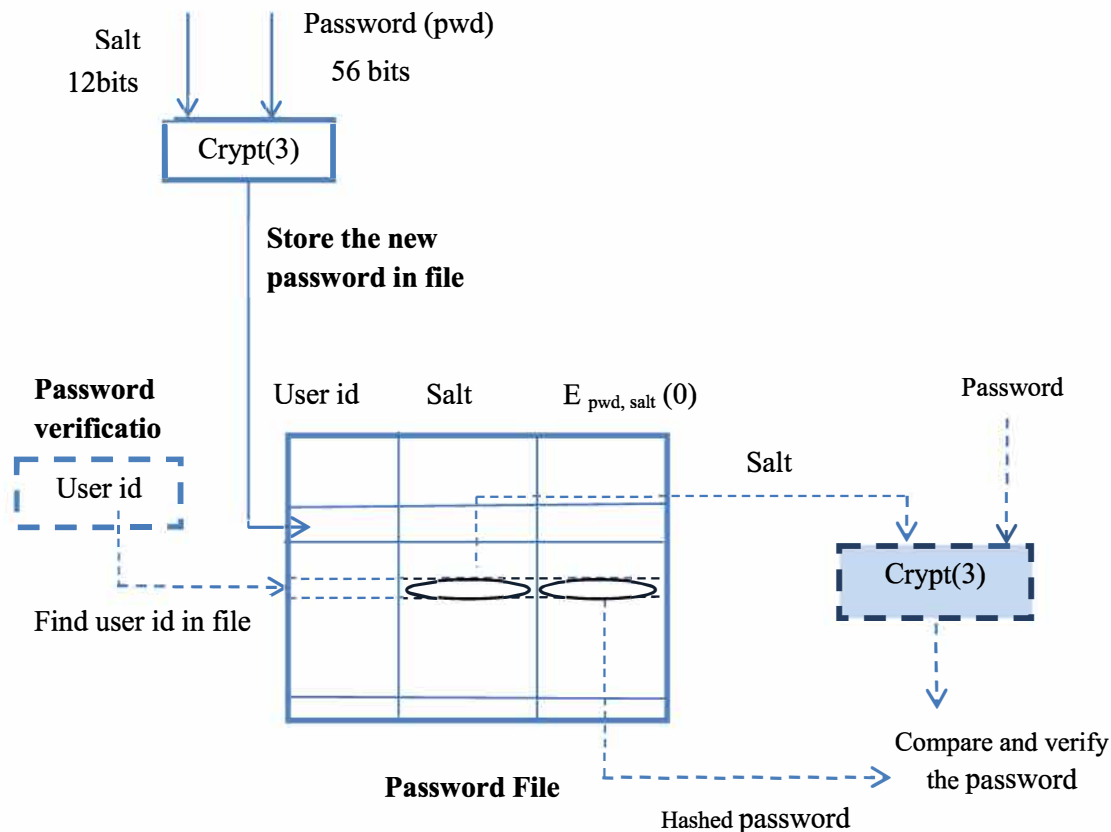


Figure 13.1 A password scheme employed by UNIX

13.5 WAYS FOR PASSWORD SELECTION

Four basic techniques for passwords are in use. These are: education, computer generation, reactive checking and proactive checking.

An education strategy spreads literacy among users regarding creation of safe, hard-to-guess and memorable passwords. It provides password setting guidelines and policies. One such guideline may list the following points to be followed for good passwords:

- minimum length of password should be greater than six.
- It should have mix of upper & lower case letters, numbers, punctuation and special symbols.
- no dictionary words should be used.

A sample policy may state that passwords: should be changed upon first login, should be refreshed frequently and should not be stored on

system.

But the issue here is that the users usually tend to ignore these guidelines and policies.

Computer software tools are used to automatically generate passwords for the users. Though safe and hard-to-guess passwords are possible by this method but it has following problems:

- Passwords generated are usually random in nature hence only possibility to remember them is to write down. (sticky label syndrome).
- It may be pronounceable but still not memorable.
- Such system generated passwords have poor user acceptance.

FIPS PUB 181 is one such automated password generation standard. It provides description of the approach along with listing of the C source code. The passwords are generated by forming a random set of pronounceable syllables and later on concatenating them to form a password.

A reactive password checking scheme is one where a password cracker tool is run by system itself against existing passwords. It is based upon the assumption that good dictionaries exist for almost all the languages. The password checking is done on periodic basis to identify guessable passwords. Once identified, these passwords are cancelled or disabled by the system and the user is notified. This scheme has two major drawbacks:

- The reactive password checking is resource intensive and consumes both time and memory.
- If a password is bad and is detected later during password checking, it remains vulnerable until detected.

Proactive checking is the most promising approach for providing password security. It allows users to select their own passwords. It is supported by system that verifies if the selected password is acceptable or not by:

- Using a system of rule enforcement, enforcing user guidelines for passwords.
- Comparing the selected password against dictionary of bad passwords. For this, a large dictionary of possible “bad” passwords needs to be created and maintained. This method has drawbacks: size of the dictionary may increase, computing cost may raise and search may be slow.
- Using algorithms like markov model of guessable passwords or bloom filter to find whether the choice is poor or not. This method doesn’t require a password dictionary.

The proactive checking method tends to maintain a balance between user acceptability and password strength.

Check your progress 1

- a. Why are Insider Attacks more devastating as compared to malicious software?
- b. How are passwords useful in security? What attention or precaution must be taken while designing the passwords?

13.6 INTRUSION DETECTION METHODOLOGIES

The intrusions are very difficult to prevent and as such the prevention methods doesn't prove much useful. In this situation, intrusion detection methods are more useful as these methods can quickly block system unauthorized access and minimize damage; can act as deterrent; or can collect information regarding intruders for future security.

Figure 13.2 shows that intruder's behavior and authorized user's behavior can be quantified. It is important to realize from the figure that though intruder's behavior and authorized user's behavior differs from each other, there is no sharp distinction between two. In other words, the two behaviors overlap in the shaded region [4][5].

If we give relaxation in the behavior of intruder (loose interpretation of intruder behavior) keeping in mind to catch more and more intruders, some of the authorized users in the overlap area are also caught e.g. portion identified as 'A' in Figure 13.2. Thus, here authorized users are identified as intruders and this is referred to case of "false positives".

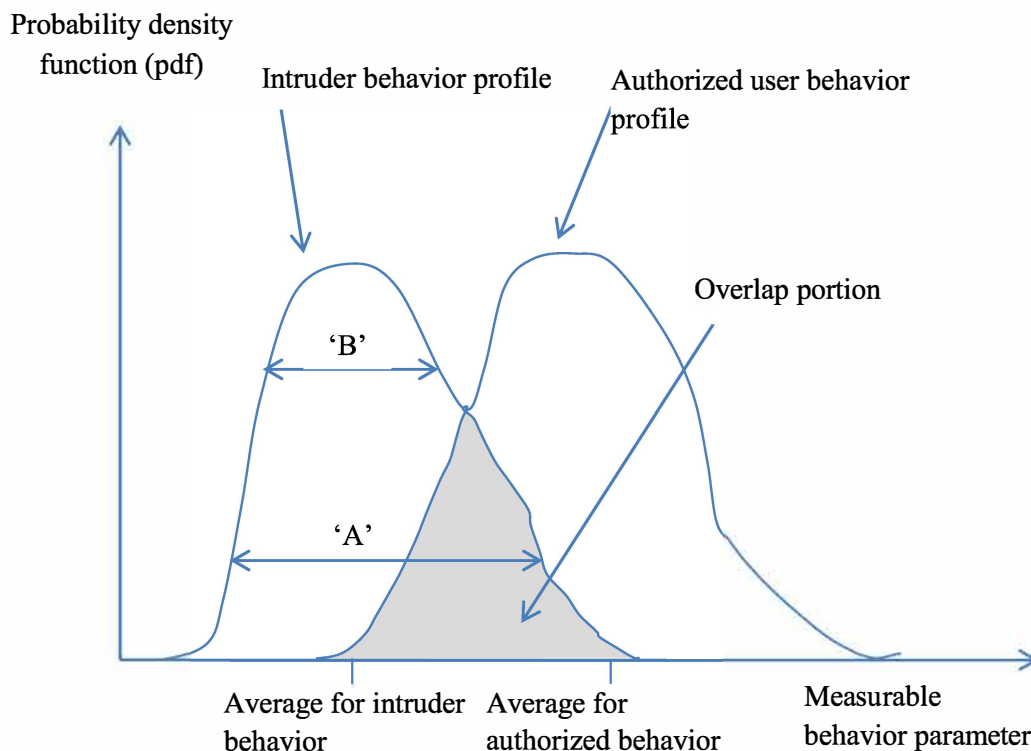


Figure 13.2 Approximate Quantification of Behavior of Intruders and Authorized Users*

* only an approximate curve, for exact curve readers may refer [4] and [5]

portion identified as 'B' in Figure 13.2. Thus, a compromise is required in between reducing either false positive or false negative cases during intrusion detection.

Approaches to Intrusion Detection are divided into: statistical approach for anomaly detection and rule based intrusion detection approach.

1. **Statistical approach for anomaly detection:** This refers to data collection and application of statistical tests so as to identify (with a high level of confidence) that the new behavior is legitimate or not. Statistical anomaly detection techniques are of two types: threshold detection and profile-based.
 - a. **Threshold based detection:** In this method, thresholds are defined for the frequency of occurrence of events over time. For example, number of occurrences of a specific event type may be counted in an interval of time and if the count increases beyond a particular number, then the system can assume that an intrusion has occurred. In this method, identification of appropriate threshold is usually done in a very crude manner and as such this method is not effective against latest and sophisticated attacks.
 - b. **Profile based detection:** In this method, profile of the user is created and then current activity of the user is checked against the profile to detect changes in the behavior. Thus, main emphasis is on characterizing past behavior of users and then detecting significant deviations. A profile is usually multi-parameter i.e., set of parameters and therefore more effective than threshold based method. It does not signal an alert based upon deviation on a single parameter rather the effect on all parameter is analyzed before identifying an anomaly. For this purpose analysis of audit records is done in profile based method.

Utilization of Audit Records

For detecting intrusion, audit record based scheme is one of the fundamental mechanisms. In this, record of ongoing activity by users is kept and is provided as input to an intrusion detection system. There are two audit record technologies:

- **Native (audit records):** These are used by all major operating systems to collect user activity information. As native audit record is already kept in the system hence no extra efforts are required but as the information kept in native audit record is fixed it may sometime does not provide the desired information.
- **Detection-specific (audit records):** These audit records may contain customized desired information, can be vendor independent and portable. On the contrary, it involves extra overhead on system.

An example of detection specific audit record is developed by Denning [6]. It has following fields:

Subject – Action initiator (terminal user/process) who issues the

Subject	Action	Object	Exception	Resource Usage	Timestamp
Sameer	execute	Copy.exe	0	CPU=00003	23111610120
Sameer	read	<old path> Figure1.jpg	0	RECORDS=0	23111610121
Sameer	execute	<new path> Copy.exe	Write-violation	RECORDS=0	23111610122

commands and are grouped as per the access class.

Action – Operations among login, read, write, execute etc. issued by a subject.

Object – upon which action is performed e.g. file, program, message, record, printer etc. The objects are grouped by type and have different granularities e.g. database actions are audited either as whole or at record level.

Exception condition – Exception triggered, upon execution of action.

Resource usage – A quantitative list of resource utilization. This may have:

- Number of lines printed/displayed
- Number of record accessed (read/write)
- Processor time utilization
- Number of Input/Output units used
- Time elapsed in a particular session

Timestamp – Time and date of action.

Example

Audit record generation for the command by Sameer:

Copy <old path> Figure1.jpg **To** <new path>Figure1.jpg

This involves: access validation, call to Copy.exe, read from one file and write to another file.

It may generate following detection-specific audit record.

As Sameer does not have write permission to <new path>, the execution is aborted.

An analysis of audit records is done which determine the activity profile of the average user. Then current audit records are used as input to detect intrusion, by analyzing incoming audit records to determine deviation from average behavior. Examples of metrics that are useful for profile-based intrusion detection are:

During audit record analysis, the programmer selects the quantitative metrics for measuring user behavior. Some of metrics useful for profile-based intrusion detection are:

Counter – A non-negative integer that is either incremented or reset but never decremented. It may be used for counting: number of user logins

per unit of time, number of times a command is executed per session, number of password failures per unit of time etc.

Gauge – It is also a non-negative integer that is either incremented or decremented. Its present value indicates the current value of an entity e.g. number of logical connections, number of outgoing messages etc.

Interval timer – Time interval between two events e.g. time elapsed between two logins done by same account etc.

Resource utilization – Amount of resource utilized in a period e.g. number of printed pages/time, processor time consumed etc.

Once the above general metrics are provisioned, following tests/models can be used to identify if the current activity is acceptable or not:

- (i) Mean and standard deviation – reflects average behavior and its variability. It is applicable to counters, timers and resource utilization.
- (ii) Multivariate – Characterize the intruder's behavior by considering correlation between two variables e.g. processor time and resource usage, login frequency and elapsed time etc.
- (iii) Markov process – Establishes transition probabilities among various states and commands.
- (iv) Operational model – It doesn't consider analysis of past audit records rather judgment or limits are defined for identifying abnormal behavior.

The major advantages of the use of statistical profiles are:

- (i) Prior knowledge of security flaws/vulnerabilities is not required.
- (ii) Normal behavior of a user is identified and deviations are easily searched.
- (iii) The mechanism is not dependent upon system properties and is therefore portable among different systems.

2. Rule-based intrusion detection approach: Set of rules are defined and used for identifying the intruder's behavior. A rule based detection method observes events and apply rules to decide if the user activity is suspicious or not. It involves two approaches: anomaly detection and penetration identification. These two are not entirely separated rather there is overlap between the two.

- a. Anomaly detection via rules: It involves identifying the anomaly via deviation from previous usage patterns as per the defined rules. Rule-based anomaly detection is similar to statistical anomaly detection. Both have almost similar methodology and strengths. Both statistical anomaly detection and rule-based anomaly detection do not require knowledge of any prior security vulnerabilities present in the system. Based upon the usage patterns as per the audit records, rules describing these patterns are automatically generated in a rule based anomaly detection method. Matching of present behavior is done against the

generated set of rules. Any deviation is treated as anomalous.

- b. Penetration identification via rules: It involves a different approach that of identifying suspicious behavior or penetrations via expert system methodology that involves rule generation by experts, system administrators and security analysts through views, interviews, interactions and codes. Thus, the skills of the person involved in rule generation determine the strength of the approach. The rules used in the penetration identification are specific to machine and operating system.

Statistical approaches are useful when the an outsider attacker is unable to imitate the behavior patterns of normal users and hence are easily caught via statistical analysis while rule-based approaches are useful in the case of inside attacker where the events and sequences may reveal penetration via rules. A combination of both approaches can be used for protection against broad range of attacks.

Notion of Base-Rate Fallacy

It is a practical problem that says that a system having high rate of detection of intrusions along with few false alarms is quite difficult to develop.

Current intrusion detection systems have not overcome this problem. They either:

- (i) Detects small amount of intrusions and are therefore considered less effective (less secure)
- (ii) Raises false alerts (alarms) frequently i.e., triggers alarms even when there is no intrusion. In such case, either the alarms are ignored or administrator's time is wasted.

Fundamentals of Distributed Intrusion Detection

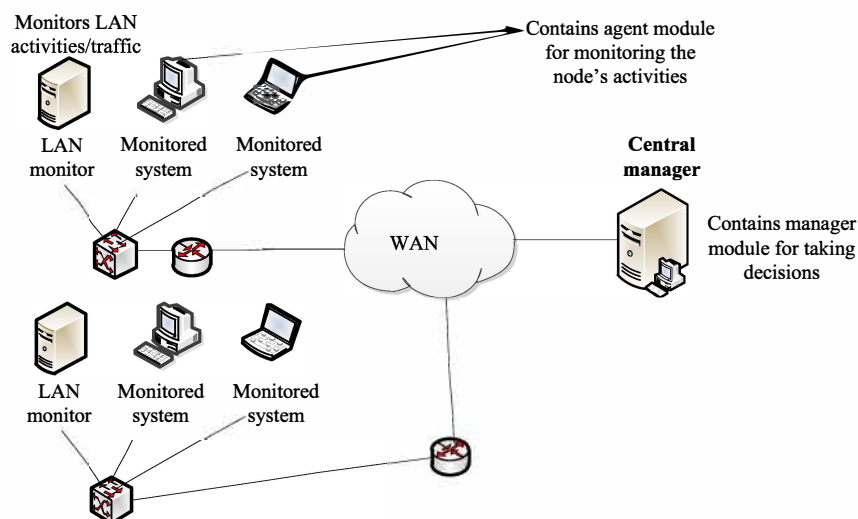


Figure 13.3 Intrusion Detecting Distributed Architecture

The Distributed Intrusion Detection System is supposed to protect and defend an organization having distributed collection of hosts supported by a LAN or internetwork. It provides an effective defense by coordination and cooperation among intrusion detection systems across the network.

Some of the major issues in the design of distributed IDS [7] are:

- Existence of different audit record formats; distributed IDS needs to convert and adjust among them.
- For data collection and analysis one or more nodes are responsible; data needs to be securely transmitted to them.
- Among centralized or decentralized distributed IDS architecture which one to use. Centralized distributed IDS provide single point IDS, is easier to manage but sometimes may become bottleneck due to overheads. In decentralized distributed IDS multiple centers exists and they must coordinate with each other.

The overall architecture of distributed IDS is shown in **Figure 13.3**. It was developed at the University of California at Davis. It has three main components. These are:

- Host agent module: This module monitors the system on which it is installed. It is used for audit collection and operates as a background process on a monitored system

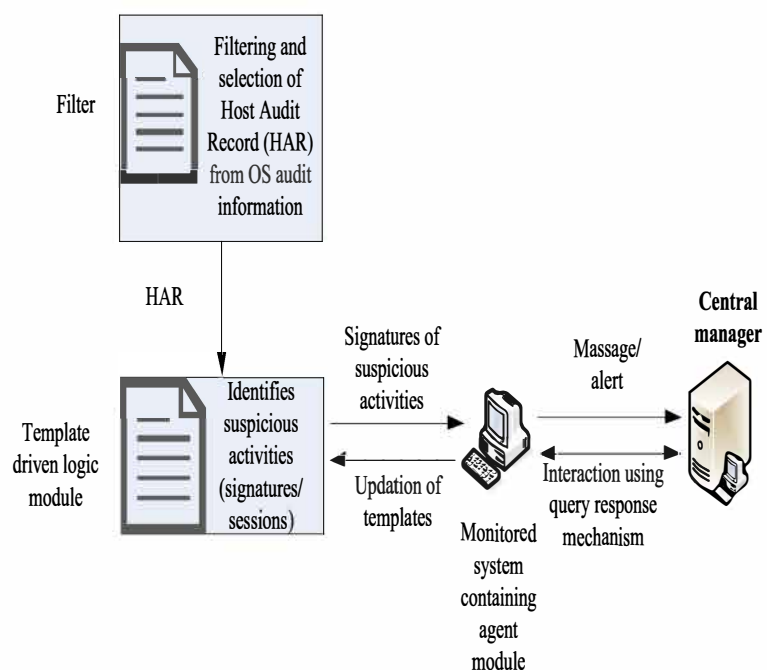


Figure 13.4 Functioning of an Agent in Distributed IDS

- LAN monitor agent module: This module functions similar to host agent module but instead it analyzes LAN traffic.

- **Central manager module:** Responsible for taking decision regarding intrusion. It receives reports from both LAN monitor and host agents. Accordingly, it processes, correlates and analyzes these reports.

The functioning and general approach of Agent in distributed IDS is shown in **Figure 13.4**. The agent captures records and filters each native operating system audit record according to its security value. The identified records are then converted into a Host Audit Record (HAR). HAR is a standard audit format. A logic module analyzes the records for suspicious activity. The logic module is template-driven. Whenever logic module finds something as suspicious it sends an alert or the notable signatures to the central manager. The central manager implements an expert system for drawing inferences from the received data regarding intrusion. The central manager may also query individual systems directly for HAR copies for interpretation.

Honeypot Systems

Honeypot systems are designed to attract, identify, trap and trace potential attackers. The identified or trapped attackers are kept away (diverted) from critical resources and systems. Thus, Honeypots not only divert the attackers from gaining access into critical systems and networks but also collect and maintain attacker's activity details. A Honeypot also tries to keep attacker engaged for a period such that administrators gets time to react to the attack.

The Honeypot systems have fabricated information which appears valuable for the attacker. The legitimate users of the system do not access the fabricated information. Thus, it is only the attacker who gets trapped into the Honeypot system. Honeypots have sensitive monitors and event loggers for detecting and collecting attacker's activities details.

Check your progress 2

- a. What does an audit record contains? How audit record analysis is beneficial?
- b. What are components of a Distributed Intrusion Detection System?
- c. Why is "Honeypot" named so?

13.7 Summary

In this unit the problem of intrusion which is caused by intruders is considered. Intruders are the unauthorized users who usually have mal-intensions and want to gain benefit from successful intrusion. The intrusion behavior and techniques are dealt with sufficient depth here. Password represents the first line of defense against intrusion. A description regarding password management including password creation issues is provided. Once the intrusion

takes place, the only thing that can be done is to reduce its impact. For this, intrusion needs to be identified and detected via some detection system. Hence, an insight about intrusion detection techniques including statistical and rule-based methods is provided to you.

Terminal Questions

1. *Classify the intruder classes. Also describe their features?*
2. *How password file is protected?*
3. *How an intruder may learn about passwords? List few methods*
4. *Differentiate between:*
 - (i) *Statistical anomaly detection and rule-based intrusion detection.*
 - (ii) *Rule-based anomaly detection and rule-based penetration identification.*
5. *List the benefits provided by an IDS.*
6. *Why is honey pot required in a network? Where it is located? What features are required in a honey pot?*
7. *How can passwords be strengthened?*
8. *Can we state that UNIX password scheme is hash code rather than encryption of a password. Justify?*
9. *If password is created from 26 alphabet characters and total password size is four characters then how much time will an adversary take to crack the password?*
10. *How the use of salt enhances the password security though it is stored unencrypted in the password file?*

References:

1. Anderson, J. Computer Security Threat Monitoring and Surveillance. Fort Washington, PA: James P. Anderson Co., April 1980.
2. Grance, T.; Kent, K.; and Kim, B. Computer Security Incident Handling Guide.
NIST Special Publication SP 800-61, January 2004.
3. Radcliff, D. "What Are They Thinking?" Network World, March 1, 2004.
4. Stallings, William, "Chapter 18: Intruders", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
5. Stallings, William and Lawrie Brown, "Chapter 8: Intrusion Detection", Computer Security: Principles and Practice, Pearson Education, Second Edition, 2012.
6. Denning, D. "An Intrusion-Detection Model." IEEE Transactions on Software Engineering, February 1987.
7. Porras, P. STAT: A State Transition Analysis Tool for Intrusion Detection. Master's Thesis, University of California at Santa Barbara, July 1992.

UNIT - 14

Malicious Programs

Structure

- 14.0 Introduction
- 14.1 Objectives
- 14.2 Various Malicious Programs
- 14.3 Nature and Functioning of Virus
- 14.4 Significant Virus Types
- 14.5 Notion of Macro Virus
- 14.6 Anti-Virus Techniques
- 14.7 Summary
- 14.8 Terminal Questions

14.0 INTRODUCTION

In previous unit of this block, you learned about intruders and ways to detect intrusion. This unit focuses on two prominent threats against computer systems or information systems, namely viruses and worms. Viruses and worms enjoy lots of publicity and are focused in many news, reports, fictions and movies. This is due to the fact that they may cause harm within few seconds and hence measures along with awareness are required against them. Technically speaking, viruses and worms are computer programs or malwares that pose serious threat and problem to the computer system. They try to take advantage of the vulnerabilities present in the system. The computer system needs protection against these computer programs. This unit provides an overview and awareness regarding these malwares. They represent the negative side of the computer programs i.e., they may be utilized for non-creative or non-productive purpose. First of all, various types of malware are described. It is then followed by detailed examination of nature of viruses and worms. Finally, some of the characteristics of anti-virus have been discussed.

14.1 OBJECTIVES

This unit deals with malicious programs (virus, worm etc.). At the end of this unit, you will be able to:

- Understand the classification of malicious software.
- Explain the basic operations of viruses and worms including their propagation mechanism.
- Learn about the types and nature of viruses.
- Describe the various countermeasures against viruses

including some of the advanced anti-virus approaches like Digital Immune System.

14.2 VARIOUS MALICIOUS PROGRAMS

Malicious software or program is a piece of code that is written by developer in such a manner so as when included or inserted in a system may cause harm to the system or to users or to the resources of the system. It is very difficult to consider one type of classification for classifying all types of viruses. It is because of a lack of universal standardization and because of overlap among various categories. A classification by Stallings is followed and is extended here as shown in Figure 14.1.

According to this taxonomy the malwares can be divided into three categories: those that require a host program, those that do not require other program i.e., are independent programs with particular malware functionality and lastly those that do not fall in either of these categories but pave the path to rooting malwares. Example of first category is virus who usually attaches itself to some program. Example of second category is worm that does not require any other program and multiplies on its own. Example of third category is a kit that automatically generates new viruses. The malwares are also sometimes classified into those that do not replicate and those that do replicate. Former are activated by a trigger while latter produce copies of themselves. A brief description of the malwares listed in **Figure 14.1** is as:

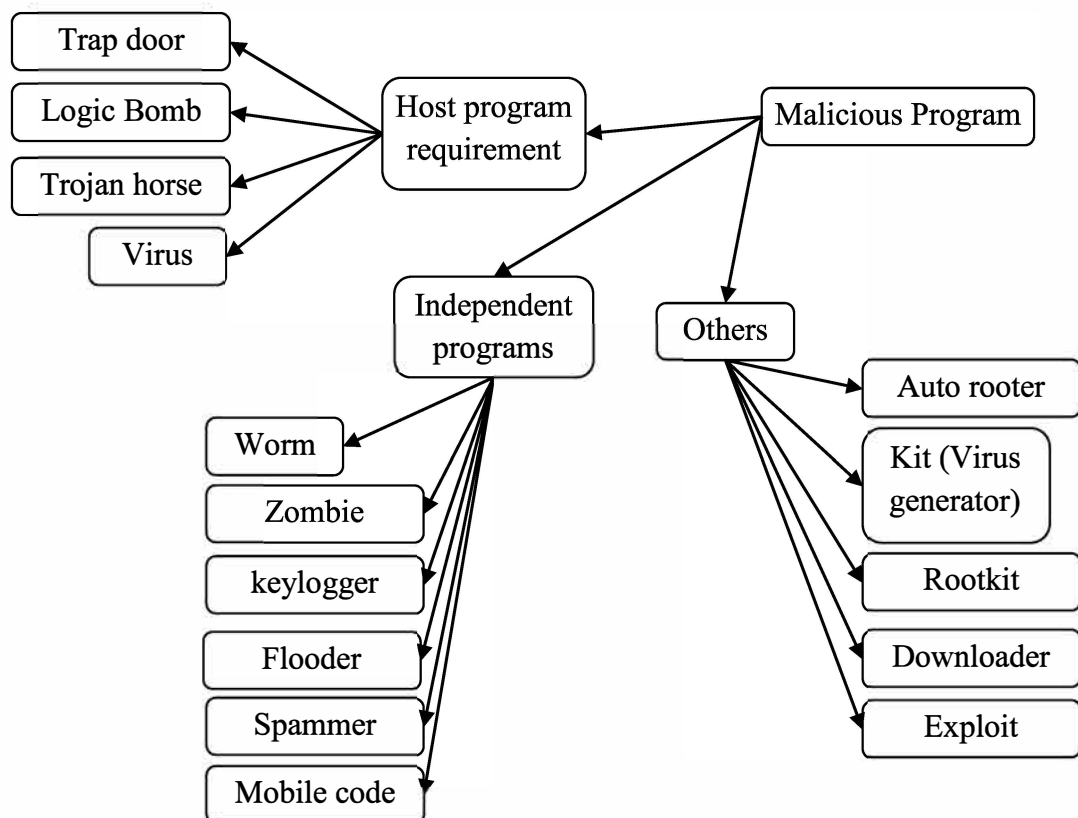


Figure 14.1 Classification of Malicious programs.

- **Backdoor or Trap door:** An unauthorized entry point in any software module that is known to person who created it.
- **Logic Bomb:** Condition (may be time) based malicious action or activity (termed as explosion) by a code.
- **Trojan horse:** An additional functionality exhibition by a program that differs from expectations.
- **Virus:** A malicious program that propagates via other programs and cause harm to system and users.
- **Worm:** A malicious program that propagates itself to other systems.
- **Zombie:** A malicious program installed on the infected system and is control by someone else for launching an attack. Mainly used in Distributed Denial of Service (DDoS) attacks.
- **Keylogger:** The program that captures keystrokes on the infected system.
- **Flooder:** The programs that either alone or in combination with other such programs on the infected systems are used to target a particular computer on Internet via large volumes of packets are termed as flooders. The traffic is too much for the target to handle and hence, it either crashes or restarts resulting in Denial of Service (DoS) to the connected users.
- **Spammer:** The programs that are used to send large volumes of unsolicited E-mails to users.
- **Auto rooter:** Hacking tool. It is used to break into another remote system.
- **Kit (Virus generator):** Tools/Software used for producing new viruses.
- **Rootkit:** Hacking tool. It is required for gaining root level privileges by the hacker.
- **Downloader:** It is a program that itself is spread via E-mails. It tries to download other malicious programs on the target machine.
- **Exploits:** code specific to vulnerabilities.
- **Mobile codes:** Portable code transferred from remote system (generally server) to user's local system where it executes and carries the desired harm.

A detailed description [1][2] of few important ones is as follows:

Backdoor or Trapdoor

A backdoor (sometimes also referred as trapdoor) is used mainly for debugging and testing purpose. During debugging and testing, it sometimes becomes essential to avoid unnecessary authentications. It may also happen that authentication procedure itself go wrong. Under all such cases the backdoor proves to be an effective measure. The backdoor provides entry into the system without going through the

normal authorization or passing the security checks. It recognizes some user ID or sequence of inputs or a particular sequence of events. Thus, backdoor provides a secret entry into a program by someone who is aware of it. It creates problem when some intruder or attacker learns about it and uses it for unauthorized access. This threat is very difficult to notice at the operating system level. The only ways to protect against such threat are: (1) to follow the proper software development that accompany security measures and (2) utilization of regular software updates.

Logic Bomb

A logic bomb is a software threat that is embedded into some other legitimate program and hence remains unnoticed. It gets activated on meeting some conditions like presence or absence of certain files, a particular day of the week or date, or a particular user who is running the application. Upon activation the logic bomb tries to meet its intended purpose which may range from alteration or deletion of data or entire files to halting/rebooting a machine.

Trojan horse

A Trojan horse derives its functionality from the ancient story in which a Trojan horse gift was given by Greeks to Spartans that led to the fall of Sparta. The soldiers hidden in the horse came out in the night when Spartans were sleeping and opened the city gates for the enemy army entry. Similarly a software Trojan horse comes through game, utility or software upgrade and usually seems useful. But as hidden side effects, it contains hidden code that performs some unwanted or harmful function. Thus, it is a method through which an unauthorized user could accomplish task that otherwise could not be performed directly by him. The most common tasks accomplished by Trojan horse includes making user's files readable, propagation of a virus or worm, backdoor installations and deletion of data.

Mobile Code

Mobile code refers to portable code e.g., script, macro, applet etc. that is transferred from remote system (generally server) to user's local system. It is executed on local system where it causes the desired effect. No user intervention is required for its execution and execution may take place over heterogeneous or homogeneous platforms. Mobile code often serves the role of (1) carrier for a virus, worm, or Trojan horse via Java applets, ActiveX controls, Java & VBScripts and (2) may also take advantage of vulnerabilities for providing unauthorized data access or root compromise. Cross-site scripting, interactive and dynamic web sites and downloading from untrusted sites or of untrusted software are some of the ways through which mobile codes performs their actions. For protection against mobile codes, major contemporary browsers ask for user permissions before any script or macro is executed on local system.

Multiple-Threat Malware

Upon mixing the selected advantages and features from different piece malicious codes (like mixing of features of worms, virus and mobile codes), the resulting code may prove to be more devastating. Such

blending utilizing multiple methods of infection and transmission helps enhancing spread and severity of the code. Examples include:
(1) Nimda that includes worm, virus, and mobile code characteristics
(2) multipartite virus that infects in multiple ways e.g., infecting multiple types of files.

Worms

Like virus, a worm is a program but unlike virus it does not require another program for propagation i.e., it replicates itself and sends its replicas on a network connection. For sending replicas on network, capabilities like email, remote execution or remote login may be utilized.

Upon settling on a host computer, the worm like virus may cause some undesired results and may propagate further on network. A worm may even implant Trojan horse programs.

Characteristics of a worm are similar to that of virus i.e., it has a dormant phase, a propagation phase, a triggering phase, and an execution phase. For propagation, a worm tries to search for addresses of other systems that can be infected. It examines host tables or repositories of these systems and then tries to establish a connection with the identified or vulnerable systems. Identification is followed by copying itself onto that system and then executing that copy. If a system has previously been infected then it is not selected for infection.

A worm may conceal itself by changing its name and properties e.g., it may change itself as a system process. Hence, worms are difficult to deal with.

Worm Propagation Model

As per analysis done by Zou *et al.* (2005) [3] regarding worm propagation, several factors are responsible for spreading the virus. These factors include mode of worm propagation, vulnerabilities exploited and similarities with the previous attacks. A new worm with almost negligible similarities with the previous worms has more chances of success. During propagation first a single host gets infected, then two, then four and the infection proceeds so on (exponential growth). This initial phase of propagation is termed as slow start phase as very few systems are infected in it. The middle phase of propagation is termed as fast spread phase as large number of systems get infected in this period. In this attack period, some of the time is invested in identifying that the visited host is already infected (linear growth). In the last phase, most of the systems have already been infected and hence it becomes difficult to find out the uninfected systems (slow finish phase).

Examples of worm attacks:

John Brunner's in 1975 introduced the term worm in its novel "The Shockwave Rider". The functionality of first worm was tested in Xerox Palo Alto Labs (early 1980s). The worm searches idle systems for running computationally intensive tasks.

Morris, a popular worm was released by Robert Morris in 1988. It attacked the UNIX systems. It spreaded via examining a variety of lists and tables; looking into users' mail forwarding files; utilizing the programs that reported the status of network connections etc. Upon identifying a host, access was tried by attempting the following: (1) cracking passwords from password file and then using them to logon to other systems, (2) exploiting finger protocol bug, and (3) exploiting a trapdoor in remote sendmail process. Once access is granted, the worm communicated with the operating system and loaded a bootstrap program on the infected system that called back the parent program for downloading the remaining part of worm. After downloading completely on the infected system, the worm is executed.

Code Red, released in July 2001. It exploits security hole in the Microsoft Internet Information Server (IIS) to penetrate. It probes random IP addresses for spreading and then initiates a distributed denial-of-service attack against a Web site via flooding from numerous infected systems. Code Red II is a variant that includes backdoor through which a hacker can control activities of victim computers.

SQL Slammer, a rapidly spreading worm appeared in 2003 and it exploited buffer overflow vulnerability in Microsoft SQL Server. In the same year, Sobig.f worm also showed its presence by exploiting open proxy server vulnerability and turning these systems into spam engines.

Mydoom (2004) is an e-mail worm that is used to send spam and installing a backdoor for gaining access to passwords and credit card numbers. Mydoom spread at the rate of 1000 times/minute choked the Internet with 100 million infected messages in one and half day.

Mobile phone worms that mainly targets smart phones started appearing in 2004. One such worm named CommWarrior was launched in 2005. The main methods of replication are Bluetooth sharing, or via MMS (multimedia messaging service) utilizing address-book numbers, or via removable memory cards. The major effects of mobile worms were: disabling of phone, deleting data and sending premium-priced messages.

Worm Technology

The worm technology includes the following variants:

- **Multiplatform:** A worm that targets variety of platforms like Windows and UNIX.
- **Multi-exploit:** A worm that utilizes many exploits e.g., utilizes vulnerabilities against web servers, browsers, e-mail, file sharing and other network-based applications.
- **Ultrafast spreading:** accelerating methods used by worms e.g. a prior Internet scan which keeps on accumulating Internet addresses of vulnerable machines.
- **Polymorphic:** Similar to viruses, worms also utilize polymorphic techniques for bypassing filters and detectors. In

these techniques, worms evolve a different code but with same functionality as the original.

- Metamorphic: Not only the appearance but behavior patterns are also changed by the metamorphic worms.
- Transport vehicles: Worms propagate quickly and compromise the target very fast, hence becomes vehicle for other attacks like distributed denial of service attacks.
- Zero-day exploit: An exploit that is unknown to everyone and is discovered only upon launching of worm. As the users are unaware of this vulnerability previously, the exploit is able to cause maximum surprise and affect.

Worm Countermeasures

The countermeasures for viruses and worms overlap; this is because of the fact that both behave similarly on the target system. Hence, antivirus software can be used to detect both viruses and worms. A point that is considered for protecting against worms is that worm propagation generates considerable network activity. Some of the countermeasures are:

- Signature-based worm scan filtering: generates a worm signature based upon which an entry or exit of worm is protected.
- Filter-based worm containment: focuses and filters on worm content in a message.
- Payload-classification-based worm containment: examine packets for worms using anomaly detection techniques.
- Threshold random walk (TRW) scan detection: exploits randomness in picking/scanning destinations to connect by an infected host during its operation.
- Rate limiting: limits the infected host traffic rate.
- Rate halting: immediately blocks outgoing traffic upon exceeding a threshold.

Rate limiting and rate halting techniques are not suitable for slow, stealthy worms.

Check your progress 1

- a. Write some of the characteristics of a Trojan horse and worm.
- b. Why are Rootkits difficult to detect?

14.3 NATURE AND FUNCTIONING OF VIRUS

A virus is a software component that causes infection to other programs (termed as host programs) and inserts its replica during

modification of host program. The infected program further repeats this process for propagating the virus and modifying other programs. The virus infected program performs its normal functioning upon execution while the secretly hidden virus code causes troubles like deleting files and data of the system. The viruses are spread usually by normal users who cannot be suspected. A Virus bears an instruction code for creating its replicas and usually takes advantage of the weakness of the system during its lifetime. Virus is itself a system dependent entity.

There are four phases of virus life cycle:

Dormant: This phase refers to the period for which virus remains idle and shows no activity. When stimulated by some trigger like start of a date, presence of corrupted file etc., the virus gets activated and enters the next phase. It is not necessary for all the virus to have this phase.

Propagation: In this phase, the virus makes replicas of itself and insert them into correct files or programs on the disk. The virus in the infected files and programs further enters into propagation phase and replicates.

Triggering: In this phase, the virus gets activated via some trigger for action and aims its intended purpose. Triggers here include various events like count of the number of copies of a virus becoming greater than some threshold value etc.

Execution: Virus completes its function or life cycle in this phase. During execution depending upon its functionality a virus may or may not damage the files or programs.

A computer virus has three parts or components [4]:

- Infection mechanism: process through which a virus spreads or replicates (also referred to as the infection vector).
- Trigger: event or condition for the payload to activate.
- Payload: the main virus functionality. The payload may involve benign or harmful activity.

A virus is embedded in some fashion (may be prepended or postponed) in a program such that upon activation first the virus code execution takes place followed by the execution of the original program code.

Once a single program gets infected, the virus is in a position to cause infection to other executable files on that system. Thus, the virus should be protected from gaining entry into the system. But this task is very difficult due to the fact that virus originated from outside the system. Thus, usually the system and application programs running on it are always vulnerable. If the system lacks access controls then viruses spread rapidly on them while for UNIX and related systems, the viruses are practically absent due to the existence of effective access control measures.

Virus Organization

Pseudo code example depicting general virus organization/structure is shown in **Figure 14.2**. During infection phase, a virus does two

things: one, it inserts its own code. In Figure 14.2 example, a code is prepended in the original file by the virus and a call to this virus code is inserted in the main function. Obviously, the virus actions are too rapid in nature. Second, virus does not change the functionality of the original file i.e., infected file remains as such. Thus, upon original file execution the user is not able to find out that the file is infected.

Whenever the infected program is called, the virus code is executed. The virus code has two functionalities: one, it tries to infect other files by random selection. During infection, virus checks that it is infecting already infected file. If so, then virus leaves this file and selects another uninfected file. Second, virus executes the instructions for harming the system. The trigger based mechanism is also possible here i.e., the harmful action is considered only upon meeting a condition (e.g. logic bomb).

As it can be noted that addition of virus instructions lead to increase in length of the original program and therefore the virus may be easily detected. Hence, the original program is first compressed and then virus code is added such that resulting length remains same as that of the original program. The pseudocode depicting compression virus is shown in **Figure 14.3**. **Figure 14.4** shows the functionality of the compressed virus. The virus code first compresses the original program file (P2) to be infected and then prepends the compression virus code (CV) to it to give rise to new infected file (P2'). Upon executing the infected file (P1'), the infection phase is executed first so as to infect other files. This is then followed by uncompressing of the compressed program file. Ultimately the uncompressed program file (P1) is executed. Thus, an infected file is able to infect other file while executing the original program file [1][2].

Virus.c

// Added virus-signature identifier

Infection(){

//code of virus that infects another executable file

//execute a loop

// within loop select a file randomly

// check for presence of virus signature

// If the selected file is already infected then select another file for

infection

// infect the selected file by prepending the virus code to file

 }

Damage(){

 ----- **//code of virus that do the required damage**

 }

main()

 {Infection();

**Figure 14.2: Pseudo code showing a file infected by virus
(virus structure)**

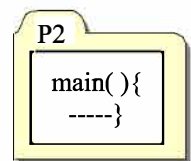
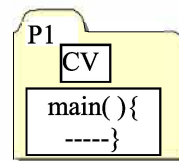
```

Compressed_virus.c
// Added virus-signature identifier
=====
Infection(){
    //code of virus that infects another executable file
    //execute a loop
    =====
    // within loop select a file randomly
    // check for presence of virus signature
    // If the selected file is already infected then select another file for infection
    =====
    // compress and then infect the selected file by prepending the compressed
    // virus code to file
}
main()
{Infection();
// uncompress and run the remaining code

```

Figure 14.3: Pseudo code showing a file infected by a compression virus

**Initially: P1- infected
P2- uninfected**



Original size P1=10KB
Compressed Virus (CV) =1KB

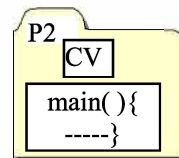
Upon infection P1 compressed to 9KB
Size of P1' = (9+1)KB=10KB

Original size P2=9KB

Intermediate: Infection spreading

1. Uncompress P1
2. Execute P1
3. Compress P2
4. Prepend compressed virus code

Finally: P2- also get infected



Upon infection P2 compressed to 8KB
Size of P2' = (8+1)KB=9KB

Figure 14.4: Functioning of a compression virus

14.4 SIGNIFICANT VIRUS TYPES

virus developers is often noticed. It sometimes appears that the viruses target specially the anti-virus softwares so that the virus actions remain hidden. Also, there are various thoughts regarding the categories of type of viruses i.e., classification may be on the basis of target affected or concealment strategy used. Therefore regardless of any classification, some of the significant types of viruses are presented here.

- (1) **Parasitic virus:** most commonly found virus and requires a host program. It gets attached to the running files and then makes copies of itself which then attacks other on-going programs.
- (2) **Memory resident virus:** It resides in main memory of system. From there it attacks almost every running program which exists in the main memory.
- (3) **Boot sector virus:** This virus resides in the boot sector of any bootable disk and spreads whenever such infected disk is used to boot a system.
- (4) **Stealth virus:** Stealth is basically a technique in which the infected virus is designed to fool the anti-virus software i.e., designed in such a way that it is not detected by anti-virus software. Examples include the compression virus and virus that corrupts the disk I/O routines in such a manner that the user does not suspect the length of the program (in former) or disk I/O functionality (in latter).
- (5) **Polymorphic virus:** polymorphic means in more than one form. Such viruses change its appearance in every infection due to which their detection becomes almost impossible. A polymorphic virus makes its replicas or copies during infection/replication that are similar in functions but have different bit patterns. This phenomenon is termed as mutation. Thus, such viruses are able to change the signature by either randomly changing the instructions or by changing the order of idempotent instructions. An example of polymorphic virus is encrypted virus that hides its own functionality by applying encryption. Each encrypted virus maintains its own secret encryption key, uses it for encryption and stores it within itself. This key is generated randomly in the infection phase by the infection causing virus.
- (6) **Metamorphic virus:** like polymorphic virus, metamorphic virus also changes its structure with every infection i.e., exhibits mutation. The major difference between the two types is that metamorphic virus not only changes the appearance but also changes their behavior by completely rewriting itself.
- (7) **File infector:** Infects selected files which are considered as executable by an operating system.

(8) Macro virus: A macro virus injects code that is able to execute a particular sequence of instructions and is interpreted by an application. These are discussed in the next section.

(9) E-Mail Viruses:

With the increased use of Internet, the virus developers took more interest in increasing the pace of spreading viruses. Hence, E-mail started gaining popularity as medium of transporting viruses. One such rapidly spreading e-mail virus is Melissa that utilizes Microsoft Word macro feature and embeds the macro in the E-mail attachments. The Word macro gets activated as soon as it is opened and then it sends itself to everyone on the mailing list in the user's e-mail account. The virus also does local damage to the user system.

The E-mail viruses get strengthened by time and in 1999 a method evolved through which activation is achieved just by opening an e-mail instead of opening the attachment. This helped a lot in virus propagation across the Internet within hours. The advantage gained out of this increase speed is that before any response or reaction or countermeasure, the virus is able to do a lot of damage. This means that security must be provided to each and every system and user on the Internet i.e., the breach in security of a single user or system may affect many systems on Internet.

14.5 NOTION OF MACRO VIRUS

A macro is a feature available in Microsoft Office tools like Word, Excel etc. It helps in recording of several steps (e.g. saving of keystrokes) and executing them together as one unit later. This feature is being utilized by virus developers to meet their desires in the mid-1990s. The virus developers define a sequence of keystrokes in a macro and then attach it to a word file. The macro gets triggered upon particular key selection. Specialty of these macro viruses is that they targets only documents stored in the system instead of program executables. This makes them platform independent i.e., hardware platform and operating system independent. As large amount of user data is stored in document, these proved devastating. They spread easily via E-mail attachments (in documents).

Adobe's PDF documents can also act as host for the macro oriented malware. Both Adobe and Microsoft realized the threat posed by macro viruses, tried to reduce the severity and released several tools like secure pdf viewer, optional Macro Virus Protection tool etc. that aim towards detecting and alerting the users regarding the file contamination with macros. Latest Anti-virus softwares also identify

these macro viruses and hence the malicious activities of macro viruses are now somewhat controlled.

14.6 ANTI-VIRUS TECHNIQUES

The primary aim of any anti-virus software or virus countermeasure is to control the malware activities such that the entire system remains safe and secure. Towards meeting this aim, the best possible solution is prevention against viruses. But practically it is not possible as viruses keep on evolving with better and sophisticated mechanism. Thus as an alternative the following technically feasible methods are adopted:

- **Detection:** Identifying the virus that is causing infection and finding out its source & place of action. Detection is possible only after the infection has taken place.
- **Identification:** determination of the particular virus that has resulted in infection.
- **Removal:** curing all traces of the identified virus, delineating them and restore the infected file to their original state. Ideally, the virus should be removed from all infected systems for preventing its further spread.

As virus may involve new signature its identification may take some time and also, as the virus utilize more advanced mechanisms its removal is sometimes not possible. Hence, under these failure situations, it is suggestive to discard the infected program and reload the clean program again [5].

Anti-Virus Evolution

With time not only the viruses but anti-viruses have become more advanced. Four generations of anti-viruses have evolved. Each signifies a new concept development e.g., first generation is signature scanner, second utilizes heuristics, third identifies virus actions, fourth uses combination of various anti-virus techniques.

A first-generation of anti-viruses is mainly scanner based. It tries to identify the known or existing virus signatures by either drive or system scan. They assumed that the virus structure and pattern remains same in all the infected copies. Obviously, scope of this generation of anti-viruses was limited due to the fact that viruses like polymorphic, metamorphic change their signatures frequently.

In second generation of anti-viruses, heuristic rules are used to find probable infections. A heuristic rule helps in finding fragments of code that are often associated with viruses and thus second generation is able to find more viruses as compared to first generation. Integrity checking was one of the major second generation technique used for identifying the infection.

Third-generation of anti-viruses targets to identify the actions instead of stored virus structure (signature). Viruses resident in the main memory execute some instructions i.e., take some actions. Based upon these small set of actions, a virus activity is identified and then reactionary measure is taken by anti-virus software. Thus, in this

generation, developing and identifying signatures / heuristics is not required.

Fourth-generation anti-virus softwares are packages that combine several antivirus techniques together like scanning and activity analysis. They try to control virus's ability of penetration into the system or files by access control feature. Thus, viruses are not able to update files unnoticed and are not able to contaminate other files [5].

Advanced Techniques against virus:

In the following section three advanced anti-Virus techniques namely Generic Decryption, Digital Immune System and Behavior-Blocking Software are discussed.

Generic Decryption Technique

Generic decryption (GD) methodology is used by anti-viruses to catch polymorphic viruses like encryption virus. In this methodology, a GD scanner is used to scan executable files containing encrypted viruses. GD has following three components:

- CPU emulator: It provides virtual environment for interpreting the infected file. The advantage of this virtual environment is that the virus is not given the actual processor and the instructions in an executable file are interpreted on virtual system. Hence, the virus is not able to do harm to the system but the anti-virus system is able to analyze it.
- Virus signature scanner: This part of GD keeps on scanning the virus code for known virus signatures or patterns
- Emulation control module: Controls the execution of malicious virus code and periodically interrupts the interpretation process for finding the virus signatures.

Each probable virus code is executed on CPU emulator. The emulator interprets lines of code one after the other. The virus gets exposed upon identification of decryption routine that tries to decrypts the original program code. The virus code is not able to harm the system or able to spread as it is being interpreted in a completely controlled environment under the control of emulation control module. The identification of signature marks or steps may consume some time and during this time the user of the system may get impatient. Hence, a balance between the 'amount of time taken in interpretation & resource reservations' and 'user response time' needs to be maintained by any such GD software module.

Technique that Enhances Digital Immunity

The digital immune system (**Figure 14.5**) is an effort by IBM and Symantec Corporation. It is a comprehensive anti-virus approach that covers not only users but organizations as well. The main aim of such system is to provide quick response leading to complete eradication or making virus useless. The immune system is responsible for capturing, analyzing, detecting, shielding, and removing a virus from an organization before it creates problems. The system also passes information to other non-infected systems about that virus so that it can be detected before the virus establishes its roots. The sequence of

steps (also marked in **Figure 14.5**) taken by digital immune system is as:

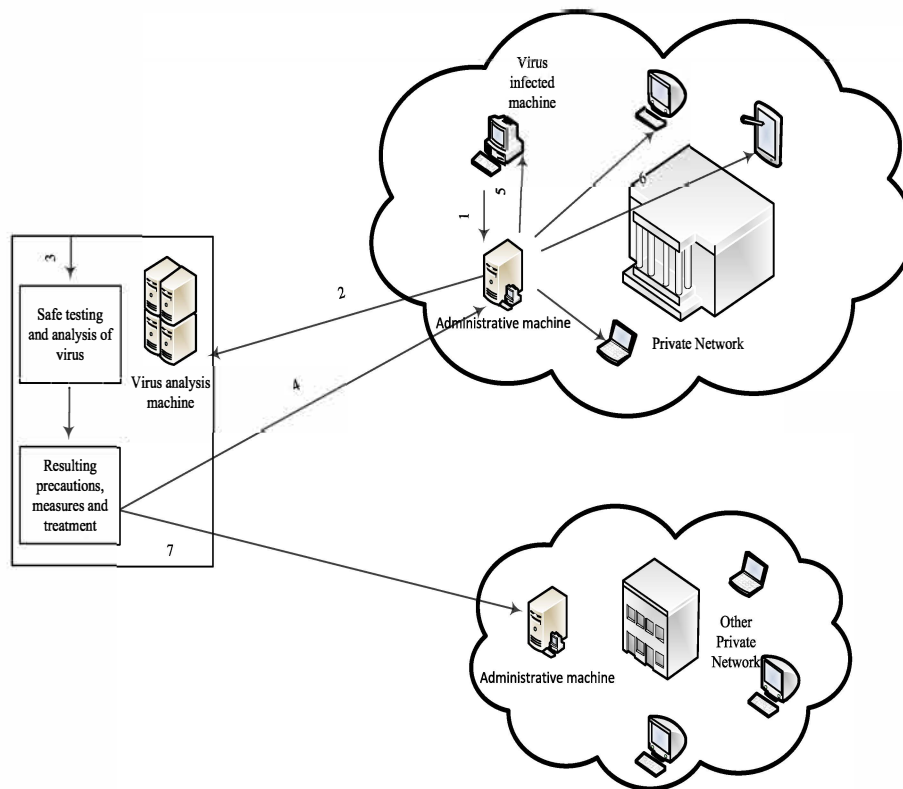


Figure 14.5: A system incorporating Digital Immunity [7]

1. Each system in an organization runs a local copy of monitoring program that uses heuristics to smell the presence of a virus. As soon as some suspicious code is detected, the copy of code is transferred to organization's administrative machine.
2. Administrative machine forwards the suspicious code to central virus analysis machine. It also encrypts the code so that no modification or eavesdropping is possible on the communication link.
3. The central virus analysis machine establishes a setup where suspicious programs can be tested. This setup is basically a virtual environment in which the code is safely run and analysis can be performed. The result of analysis is a treatment for identifying and removing the virus.
4. This treatment is then installed on administrative machine as countermeasure for virus.
5. The administrative machine is responsible for distributing the treatment within the organization including the infected client.
6. The prescription is also forwarded to other clients in the organization.
7. The central virus analysis machine updates all the anti-virus subscribers for protection against this newly discovered virus.

The virus analysis machine is like central nervous system to digital immune system. It constantly analyzes and monitors viruses' activities and constantly updates itself regarding new and innovative viruses so that it can counter the new viruses as soon as possible.

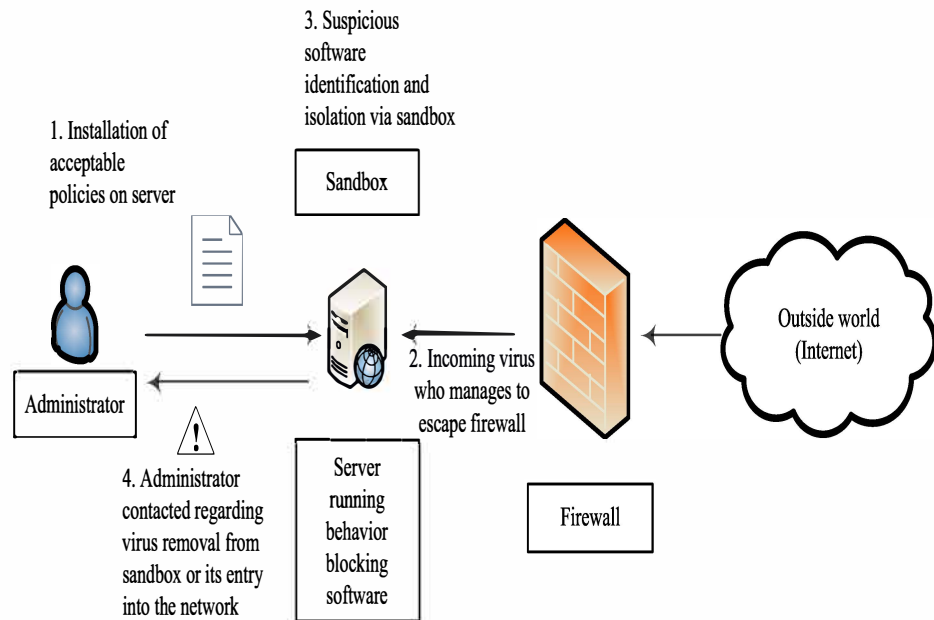


Figure 14.6: Functioning of Behavior blocking software

Technique that applies Behavior-Blocking

Behavior-blocking software monitors malicious program behavior and actions in real-time. It usually functions by integrating itself with system's operating system. Monitored behaviors can include attempts by malicious program regarding - opening, modifying, deleting files; formatting drives; other sensitive disk operations; changing macros or logic of executable files; changing critical system settings; script addition to e-mails and chatting modules; and establishing network communications. The identified malicious program is then blocked.

The entire functionality is shown in **Figure 14.6**. Network administrator set policies on Behavior-blocking software running on server. For those programs that are able to pass through the firewall but seems suspicious, the blocking software module blocks them i.e., they are prevented from executing and are kept isolated in a sandbox. This isolation protects the OS resources and applications from malicious program. The blocking software module then sends an alert to admin and depending upon the admin's reply it either removes or blocks or executes the blocked program. The major advantage of Behavior-

blocking software is that the blocking of the suspicious programs take place in real-time.

Check your progress 2

- a. Write some of the characteristics of a computer virus.
- b. What is the primary advantage of the use of workstation-based anti-virus?

14.7 SUMMARY

In this unit, you learned about malwares that prominently pose threat to computer security. Viruses and worms are the major malwares that need attention by the computer fraternity. A classification of these malwares and their nature is discussed here. You also learned about the basic functionality of viruses and worms; including their propagation mechanisms. In security, it is felt that it is very difficult to prevent any computer systems against these malware threats [6]. Hence, another practical alternative methodology involving malware detection, identification and removal is followed. In this methodology, sometimes only the malware detection is possible and in such cases the restoration of actual from backup of data and files is often required to ensure security. Obviously, taking appropriate actions in timely manner will not allow the malware to take advantage of vulnerabilities present in the system. This will indeed curb the negative aspects of these malicious programs. Towards end you learnt about some of the anti-virus approaches including some of the advanced concepts like Digital Immune System.

Terminal Questions

1. *Why compression is used during virus operation?*
2. *Why encryption is used during virus operation?*
3. *Discuss various types of virus.*
4. *Compare and contrast logic bomb and Trojan horses.*
5. *Justify the role of back door?*
6. *Define: keylogger, zombie, spammer and rootkit.*
7. *Describe the functioning of a typical virus.*
8. *What are worms? How are they propagated?*
9. *Compare an antivirus with digital immune system.*
10. *Describe the functioning of behavior blocking software.*

References:

1. Stallings, William, "Chapter 19: Malicious Software", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.

2. Stallings, William and Lawrie Brown, "Chapter 8: Intrusion Detection", Computer Security: Principles and Practice, Pearson Education, Second Edition, 2012.
3. Zou, C., et al. "The Monitoring and Early Detection of Internet Worms." IEEE/ACM Transactions on Networking , October 2005.
4. Aycock, J. Computer Viruses and Malware. New York: Springer, 2006.
5. https://en.wikipedia.org/wiki/Antivirus_software
6. Williams J. "What Is Anti-Virus?," SANS - OUCH!. The monthly security awareness newsletter for computer users. December 2014
7. Kephart, J., Sorkin, G., Chess, D., and White, S., "Fighting Computer Viruses," Scientific American, November 1997.

UNIT - 15

Firewall

Structure

- 15.0 Introduction
- 15.1 Objectives
- 15.2 Characteristics of Firewall
- 15.3 Various Firewall Types
- 15.4 Practical Firewall Configurations
- 15.5 Summary
- 15.6 Terminal Questions

15.0 INTRODUCTION

Through Internet an intruder or malicious code can creep easily into the Information systems and networks of an organization. In such case two mechanisms are possible: one, to protect individual system and two, to protect the entire network and systems in an organization. Usually the former is provided primarily using anti-virus software whereas latter is provided using firewalls. Anti-viruses generally follow signature and pattern matching whereas the Firewalls act like a barrier and filter in the path of traffic flowing in both inward and outward direction i.e., between public Internet and local LAN. It is also possible to use software firewalls on individual systems but it provides only a secondary defense. Dangerous and harmful viruses easily bypass the system firewall and cause damage to the systems. Thus, anti-virus software proves to be more effective in such situations. It is also not practical to equip each and every workstation and server in an organization with strong security features.

Most of the digital works of any organization is performed over networks (either intranet or Internet). The working nodes not only connect to the servers or workstations in near proximity (via LAN connection) but also in geographically distant location (via WAN or Internet connection). The organization and individual users gets benefitted upon reaching and accessing information from distant location. Though this is beneficial, it also opens doors of the organization's information systems and networks to the outsiders. An outsider (may be an intruder) may reach, access and interact with internal users & assets of an organization. Thus, it brings risk and threat to organization and its assets. Firewall proves to be an effective measure under such circumstances and provides a perimeter defense. On one hand it protects an organization's assets, systems and internal network from outside originated network-based security threats; on the other hand it provides access to the outside world. However, it

cannot be denied that for comprehensive security in an organization both firewall and host security should work hand in hand. In this Unit, Firewall design principles including characteristics are first presented. It is then followed by describing the various types of firewalls. Towards end, existing firewall configurations are explained.

15.1 OBJECTIVES

As evident from introduction, this unit is devoted to protecting organization and individual systems by firewalls. At the end of this unit, you will be able to:

- learn the design goals of a firewall along with realizing how a firewall enforces a security policy.
- understand the types of firewalls and their characteristics.
- use and configure rules for Windows firewall to protect individual systems.
- know about the various configurations of firewalls.
- know the concepts of Demilitarized zone (DMZ), Virtual Private Network (VPN) and distributed firewalls.

15.2 CHARACTERISTICS OF FIREWALL

A firewall usually has a policy installed that may identify and filter the unauthorized network traffic entry or exit i.e., it act like a control and monitoring point that applies rules for either passing or dropping the traffic between inside (local LANs) and outside systems (WANs and Internet). It acts like an erected virtual security wall that protects against the intruder's traffic by inspecting the packets.

A firewall has following characteristics [1-5]:

- It is implemented between the external untrusted world (Internet) and the internal more trusted network (organization network). **Figure 15.1** show general model of firewall use as a barrier or choke point between Internet and organizational network.
- It serves to keeps unauthorized users out of the protected/trusted network.
- It prohibits potentially vulnerable services from entering or leaving the network.
- It provides protection from various kinds of routing attacks.
- It provides a place where security-related events can be monitored as a whole on a network. It alarms when an abnormal behavior is detected.
- A firewall functions in a manner so as to support security mechanisms like IPSec and virtual private networks.

- It provides platform for non-security related Internet functions like NAT (Network Address Translation) and Internet usage audits or logs.
- The firewall itself should be strong enough and not fell prey to attacks conducted by intruders.

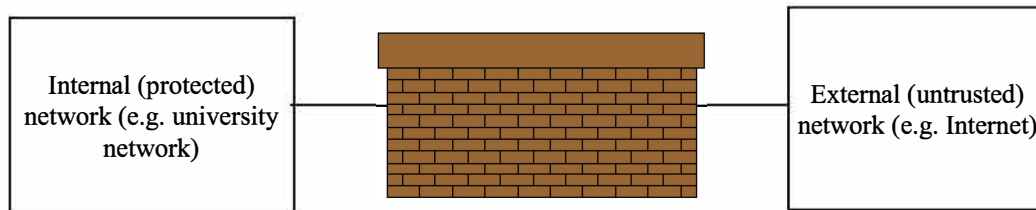


Figure 15.1 General model of firewall used for controlling the traffic at the entry

Firewall Limitations

Firewall do have limitations, some of them can be listed as:

1. It fails to safeguard against attacks that have the capability to bypass the firewall.
2. It does not protect against internal threats where a disgruntled employee or colluding employee (one who cooperates with an attacker) is responsible for the attack.
3. A wireless connection like wireless LAN (WLAN) is vulnerable to threats (due to inherent properties of wireless medium) irrespective of their protection via firewall i.e., firewall is unable to guard systems using wireless communications.
4. Firewall cannot protect portable computing devices that get infection while working or being used from outside the organizational network, and then brought and used internally.

15.3 VARIOUS FIREWALL TYPES

There are four types of firewalls: (1) packet filters, (2) stateful packet filters, (3) application-level gateways and (4) circuit-level gateways [1-3].

(1) Packet Filters

A packet-filtering firewall is simple and fast. It is transparent in functioning and uses a set of rules. The firewall rules are derived from information like source & destination IP addresses, ports (transport layer port number which defines applications like SNMP or TELNET), transport protocol and router interface (from which the packet came or towards which the packet is destined) etc. contained in a network packet. The rules are applied in both the directions i.e., for incoming and outgoing traffic.

As shown in **Figure 15.2**, a packet filter firewall utilizes information from the transport, network & data link layers (marked via grey shading) for making decisions on allowable traffic flows. It is placed in the border router (**Figure 15.3**) between and private (internal) and public (external) network and works toward securing internal more trusted private network from the external less-trusted Internet.

When a packet arrives at packet firewall, it is matched against the rules. Upon finding no match in the rule set for a packet, the default policies are used. There are two default policies.

- Conservative policy – wherein the default action is to discard the packet. It is more secure but may cause hindrance to users.
- Permissive policy – wherein the default action is to forward the packet. It is less secure.

Thus, each incoming and outgoing IP packet is examined by packet filter. Depending upon the rules and default policy, a packet is either forwarded or discarded (filtered).

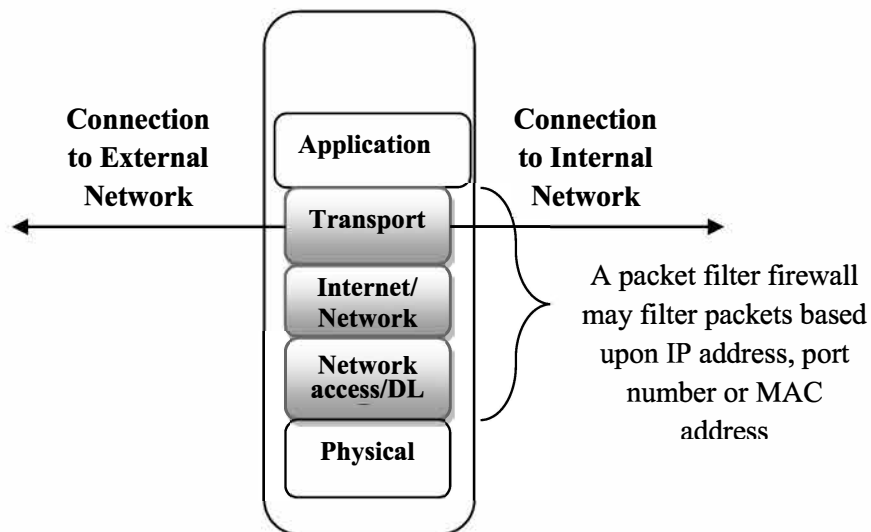


Figure 15.2 Scope of a packet filter firewall

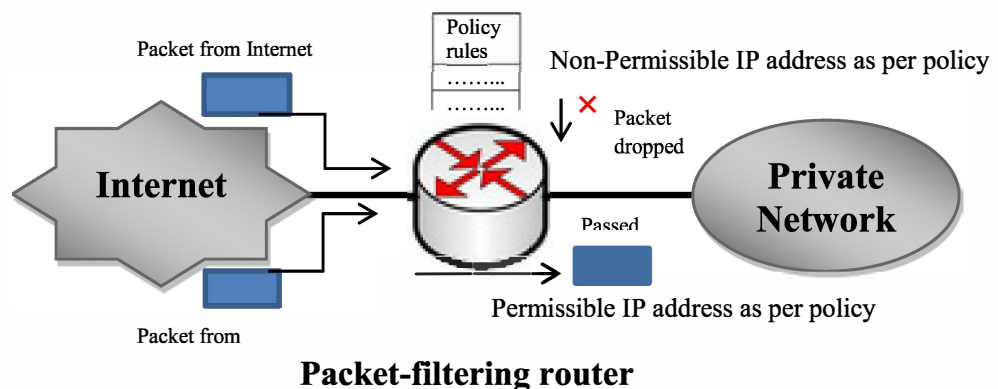


Figure 15.3 Functioning of a packet-filtering router

Table 15.1 shows example rules sets (set I-V) for a packet filtering firewall. The following points are worth for understanding the rule sets:

- Within a particular set, the rules are applied from top to bottom.
- ‘*’ represents wildcard character – It matches with everything.
- In these rules, default is the discard policy i.e., if packet does not match with a rule in a particular set, it is discarded.

A	action	ourhost	port	theirhost	post	comment
	block	*	*	EXTHOST	*	Any packets form external system EXTHOST are dropped
	allow	INT_GH	25	*	*	Incoming SMTP packets (E-mail) fro Internal Gateway Host named INT_GH are permitted

B	action	ourhost	port	theirhost	post	comment
	block	*	*	*	*	Default rule to be included as last rule in the policy

C	action	ourhost	Port	theirhost	post	comment
	allow	*	*	*	25	connection for sending and receiving E-mails via SMTP port

D	action	src	port	dest	port	flags	comment
	allow	List of internal hosts	*	*	25		Internal SMTP packets destined to SMTP port are bypassed
	allow	*	25	*	*	ACK	SMTP server responses are allowed

E	action	src	port	dest	port	flags	comment
	allow	List of internal hosts	*	*	*		Internal requests to any outside host
	allow	*	*	*	*	ACK	replies against previous requests
	allow	*	*	*	*	>1024	Data communication with nonserver

Table 15.1 Example rules sets for a packet filtering firewall [6]

- The action to be taken after matching is either ‘block’ or ‘allow’.
- The set has information pertaining to internal trusted network i.e., name of hosts (named as ourhost in the table) and the corresponding ports used. It also has information pertaining to outer untrusted network i.e., the names of hosts on these networks (named as theirhost in the table) and their corresponding ports used.

Explanation of individual sets:

Set I – It has two rules: (1) Packet from external host named as EXTHOST are blocked as this host is considered untrusted may be due to fact that it is sending large spam. (2) Incomming mails (SMTP port = 25) are permitted for the gateway host named as INT_GH.

Set II – This set has one rule only which can be treated as default rule. The default rule is to be included as last rule in any rule set.

Set III – This set also has one rule only that specify that mail from any inside host can send mail to outside mail server (default port = 25 => SMTP server). Thus, a TCP packet having 25 as a destination port is forwarded to mail server.

Issue: An attacker using an outside machine may configure itself to port 25 and may communicate with any inside host i.e., may send TCP packets to any inside host having source port as 25.

Set IV – This set has two rules: (1) An outgoing IP packet originated from internal hosts having destination port as 25 is forwarded (allowed). (2) Incoming IP packet originated from outside host having source port as 25 and ACK flag in the TCP segment as TRUE, is also allowed. Thus, TCP connection is established between internal hosts and outside server.

Set V – This set has three rules and is required for handling FTP connections. In FTP, two connections are required: one for setting up of file transfer (control connection), other for actual file transfer (data connection). Rule 1 permits internal requests to any outside host. Rule 2 allows reply against previous requests (ACK flag of TCP segment = TRUE). Rule 3 is for data connection where port greater than 1024 (for non-server) is used for communication.

Weaknesses of packet filter firewalls:

- Packet filter firewalls are unable to examine the upper layer data and unable to prevent the application level attacks and vulnerabilities. Thus, once an application is permitted by packet filter firewall then all the specific functions of that application may be performed.
- Packet filter firewalls provides limited logging i.e., a packet filter firewall log has only source & destination addresses and traffic type.
- Packet filter firewalls does not support advanced user authentication methods due to lack of support to application's data.
- It is unable to detect the packet spoofing i.e., when the information contained in the packet is altered.
- It is susceptible to security breaches caused by improper configurations e.g. a firewall may be wrongly or accidentally configured to bypass the traffic that otherwise is denied as per the organizational policy.

Packet Filters are prone to threats. Some common threats are:

- IP address spoofing
 - A packet originated on outside network is fabricated by an intruder so as to bear IP of an internal node as source IP.
 - This attack can be counteracted via enabling filters that identify such spoofed packets by noticing that this packet bearing an internal IP is received on external interface and hence, may be discarded.
- Source routing attacks
 - In this, attacker/ source specify the route that a packet should take to bypass the security measures.

- This attack can be counteracted by firewall discarding all source routed packets.

➤ Tiny fragment attacks

- Packet Filter takes an assumption that among the different fragments corresponding to one packet, it is sufficient to check the first fragment. If first fragment passes the rules, then this fragment and the remaining fragments are forwarded else, they are all dropped. Attacker splits the TCP header information over several tiny fragments. In this case, first fragment clears due to lack of information which implies that other fragments also clear the filtering rule and hence are bypassed by the firewall.
- This attack can be counteracted by enforcing that the first fragment has minimum fragment size to include full TCP header. If this contains full header, it can be checked properly. If this first one containing full TCP header is now discarded, all others are discarded whereas if it is selected all other fragments are now selected.

(2) Stateful Packet Filters

The packet filter firewall takes decisions for filtering based on an individual packet basis. Sometimes filtering is required based upon higher layer context, in such cases packet filters proves ineffective.

For understanding the notion of higher layer context, consider **Table 15.2** showing connection state table. In this table, source address & port, destination address & port and connection state is mentioned for each of the connection. This directory of outbound TCP connections is created by stateful inspection firewall and it has entry for each of the present connection.

It is assumed that each connection is established over TCP/IP and follows a client and server model. Usually, the client requests server and has associated port lying in the range 1024 and 65535 whereas server serves the client request and has associated port less than 1024. The client port is temporary and is created for client for the duration of session only whereas the server port is static and is a well-defined port.

A simple packet filtering firewall does not consider the directory of outbound TCP connections. Therefore, it permits inbound network traffic for TCP-based traffic without considering high-numbered ports. Thus, the attacker's remains unaffected and also gets entry into the network.

A stateful inspection firewall does consider the directory (**Table 15.2**) of outbound TCP connections. It permits incoming traffic to a high-numbered port only when at least one of the entries in this directory has this high numbered port present as source port. Otherwise, the packet is filtered. This means that the inbound packet is a response given, in the existing connection. A stateful firewall may also keep track of

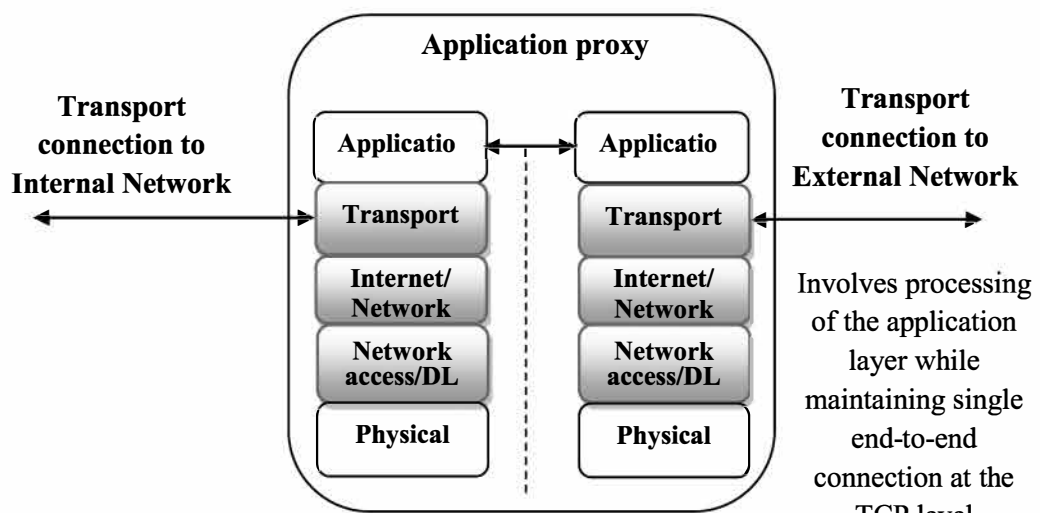
TCP sequence numbers to prevent attacks that depend on the sequence number i.e., a sequence number not lying in the expected range may be filtered. It may even inspect limited application data for protocols like FTP, IM etc. Thus, by inspecting state (context) expressed via port, sequence number, application data etc., the stateful inspection firewall checks that the packet validly belongs to one of the sessions. Hence, the stateful firewall is able to detect and filter bogus packets sent out of context.

Source IP	Source port	Destination IP	Destination port	Connection State
10.7.2.55	1030	177.200.15.1	80	Established
19.16.1.7	1048	10.1.1.2	25	Established

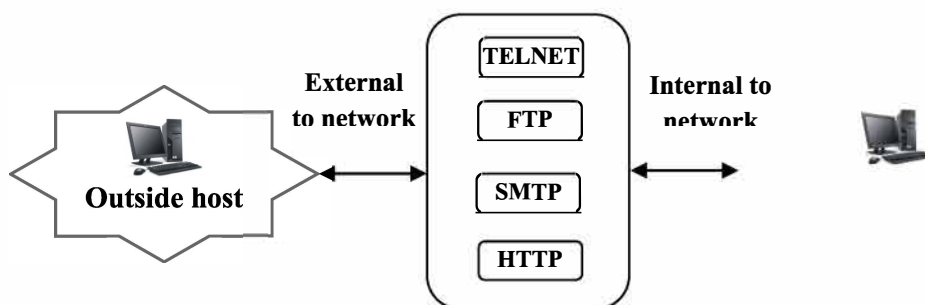
Table 15.2 Connection state table of stateful packet filter firewall

(3) Application Level Gateway

- An application-level gateway firewall (**Figure 15.4 (a)**) is supposed to be more secure as compared to packet-level firewalls. It works like a proxy and relays the application-level traffic of applications like TCP or FTP.



(a) Scope of an application-level gateway firewall



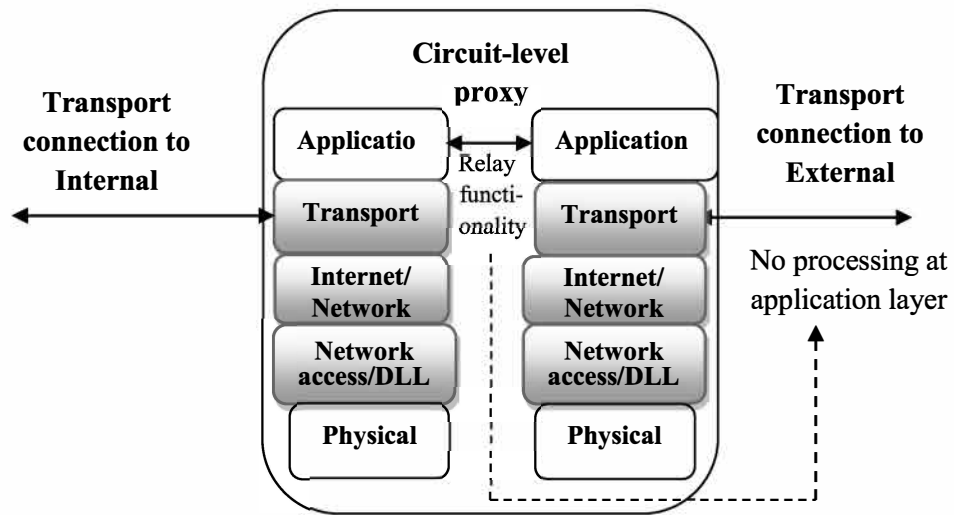
(b) Functioning of an application-level gateway

Figure 15.4 Application-level gateway

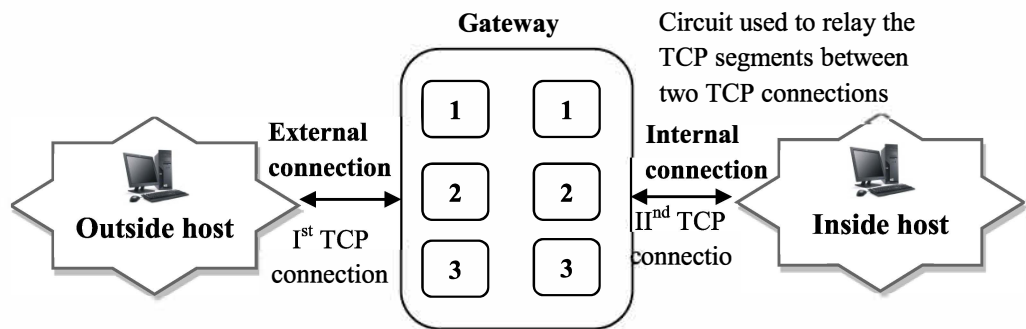
- Unlike packet-level firewalls that use several combinations of ‘allow’ and ‘deny’ for TCP and IP packets, the application-level gateway firewall examines only few applications (**Figure 15.4 (b)** lists these as Telnet, FTP, SMTP, HTTP).
- Application-level gateway firewall can log and audit traffic at application level.
- It lies in between the two communicating users resulting in two spliced connections.
- One spliced connection gateway interacts with user regarding remote host name, valid user id and authentication information.
- On the other spliced connection, gateway contacts the remote application and also relays the application data to the remote host.
- Separate proxies are required for each service.
- For getting service, the gateway must support proxy code for the specified application, so that the application’s traffic can be relayed between hosts and users. In absence of such proxy code, applications cannot get the services of the application-level firewall.
- One of the major concerns with this kind of firewall is the processing overhead required per connection.

(4) Circuit Level Gateway

- A circuit-level gateway firewall can be implemented as single standalone system or as an application-level gateway supporting specialized functions for selected applications.
- Unlike application-level gateway firewall where one end to end TCP connection is maintained, two TCP connections are setup in circuit-level gateway firewall.
- The circuit-level gateway firewall relays and controls the TCP segments from one connection to other, thereby providing the security feature. It does not examine the segment content while relaying and hence involves less processing as compared to application level firewall.



(a) Scope of a circuit-level proxy firewall



(b) Functioning of a circuit-level gateway

Figure 15.5 Circuit-level gateway

- First TCP connection is established between circuit-level gateway firewall and internal host while second TCP connection is established between circuit-level gateway firewall and external host.
- Circuit-level gateway firewall does not examine the contents of TCP segment while relaying them between two connections (**Figure 15.5 (a)**).
- In a typical example, both application-level gateway firewall and circuit-level gateway firewall can be used. In such situations, outgoing traffic is from trusted internal network and hence, does not require any processing so circuit-level gateway firewall can be used effectively whereas incoming traffic is from untrusted external network and hence, may involve processing overheads so application-level gateway firewall can be used effectively.
- SOCKS (RFC 1928) package is one of the most commonly implemented circuit-level gateways.

Concept of Bastion Host

- Bastion host strengthens or enhances the security of entire internal network. It refers to a system on which an application

level or circuit level gateway firewall can be installed by firewall administrator.

- Selection of such host is based upon assumption that this host is strongly protected or has strong security.
- The bastion host has following characteristics:
 - (i) It is trusted system that has secure operating system installed.
 - (ii) It installs upon itself only the essential proxy services like telnet, DNS, FTP, SMTP, user authentication etc.
 - (iii) It requires two levels of authentication: One by bastion host itself and other by proxy services running over it. Thus, user requires two logins.
 - (iv) It may have two or greater than that number of interfaces for relaying traffic according to the policy. The bastion must be trusted for enforcing the separation between the two connecting networks.
 - (v) Each proxy on bastion host
 - (a) is configured to support limited application command set/features.
 - (b) is configured for providing access to subset of systems on the trusted network.
 - (c) maintains detailed log audit information (traffic log, connection duration etc.) which can be used under intruder attacks.
 - (d) is a simple and small piece of code (software) that can be checked easily for security flaws.
 - (e) is independent of other proxies on same bastion host. Thus, a problem (or vulnerability) in one does not affect other. Also, one proxy can be installed/uninstalled independently of the other.
 - (f) Performs no disk access except initial reading of configuration file. This protects the proxy from unwanted installation of Trojans/dangerous files.
 - (g) Runs in a private and secure directory as non-privileged user.

Apart from the major firewalls discussed, two more kinds of firewall exists, one host based firewall and two, personal firewall [1].

Notion of Host-Based Firewalls

It refers to a software firewall used for securing individual hosts e.g. servers or workstations in an organization. It can be provided in operating systems or as an add-on package.

A host-based firewall is responsible for filtering and restricting the flow of packets on a host.

The major advantages using server-based or workstation-based firewalls are:

The host-based firewall provides an additional layer of protection along with the stand-alone firewalls like packet-filters, stateful packet filters, application-level gateways filters and circuit-level gateways. It functions independently of any stand-alone firewall and upon addition into the network does not require alterations into the existing stand-alone firewall configuration.

Depending upon the organization policies, tailoring of the filtering rules on host firewall is possible. It is also possible to have different filters on servers for different applications.

The host based firewall is located inside the organizational network and hence, both internal and external traffic must pass through the firewall. Thus, it functions independent of the topology of the network.

Understanding the Personal Firewalls

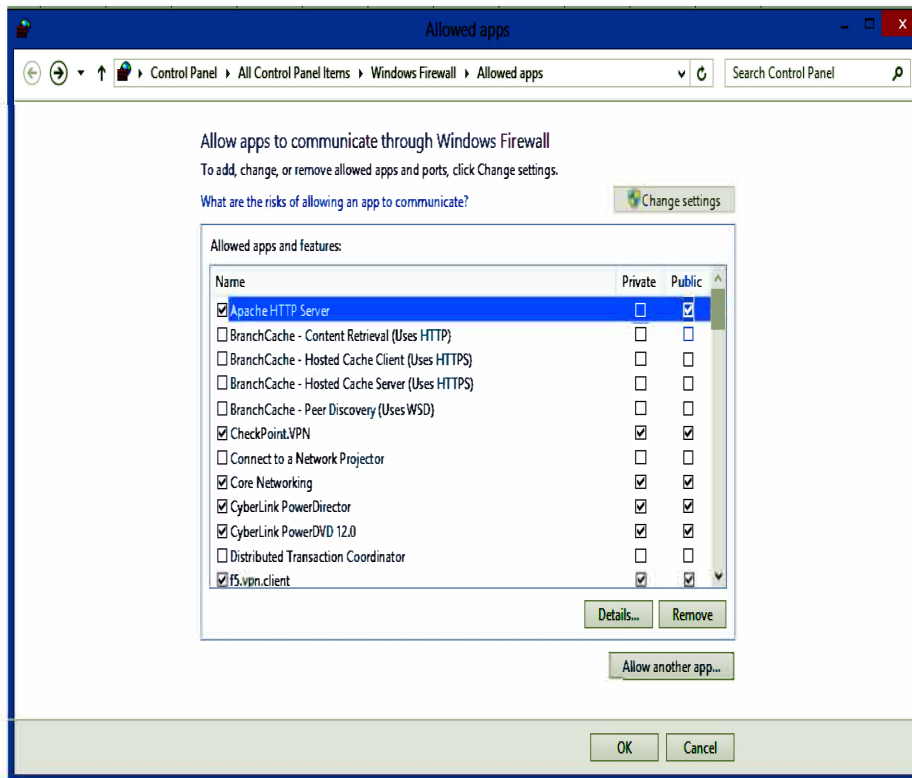
A personal firewall is used to protect the personal computer or workstation from the attacks and intrusions from Internet or organizational network. Though, personal firewall can be used in either home environment or in organizational intranets, it is typically used in home environment on the personal computers. If the home environment has multiple computers accessing the Internet, firewall functionality can also be incorporated in a router that connects all of the home computers to the Internet [7][8].

Personal firewalls are simpler than server-based firewalls or stand-alone firewalls. They deny unauthorized remote access to the computer and can monitor outgoing activity. The monitored activity details helps in detecting and blocking malware.

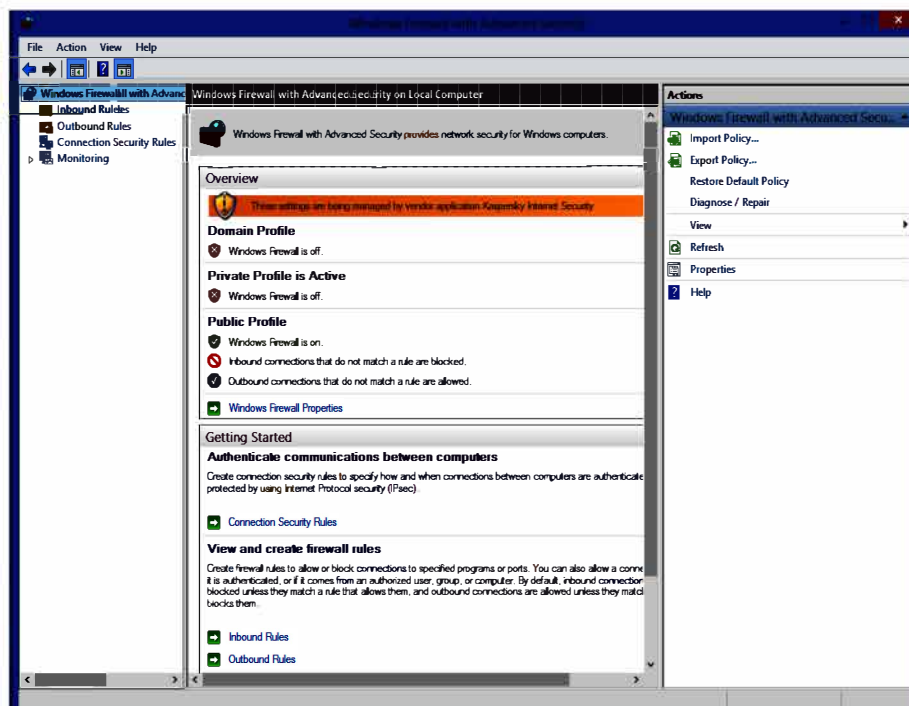
Let us consider a Windows based personal firewall. The personal firewall denies all inbound traffic excepting the one permitted by the user explicitly (**Figure 15.6 (a)**). The interface of personal firewall is shown in **Figure 15.6 (b)**. The following features are present in a Windows based personal firewall (**Figure 15.6 (c) - (h)**):

- The interface is simple and uses easy-to-configure checkboxes.
- A list of inbound services can be selectively enabled/disabled using their port numbers.
- For increased protection, advanced firewall features are available.
- UDP packets are mostly used for attacks and as such can be blocked by personal firewall. In such case network traffic belonging to TCP category linking to open ports is only permitted.
- For checking unwanted activity, it supports logging.

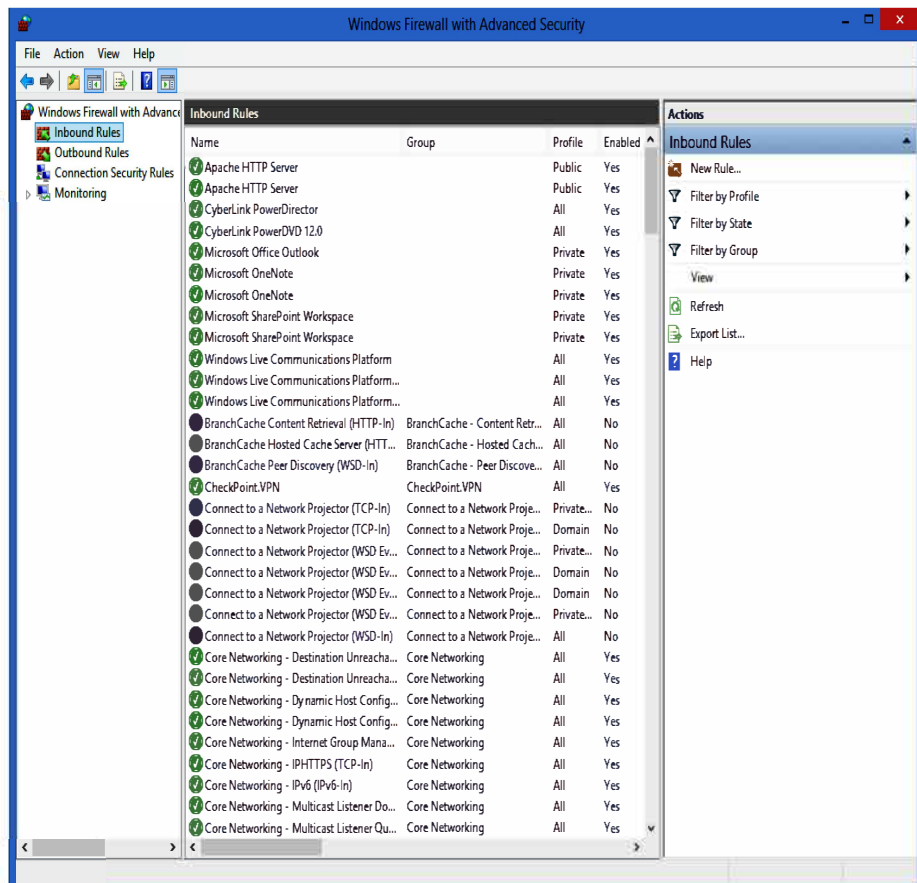
Similar features are provided by the firewalls provided in Mac OS and Linux systems.



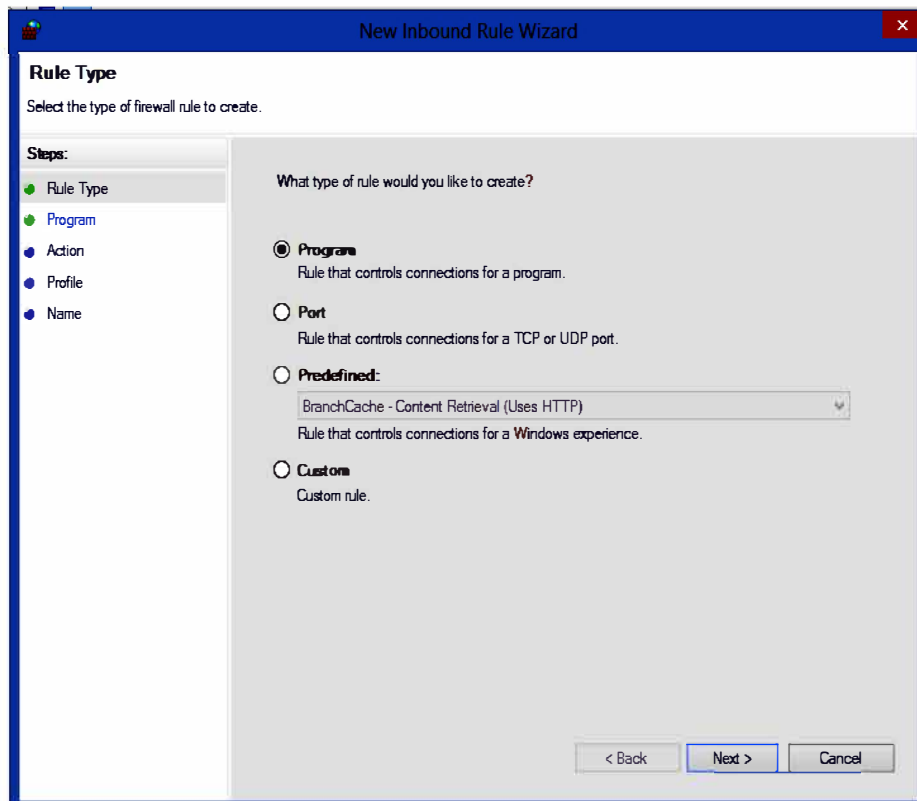
(a) Screenshot of windows firewall showing permitted and non permitted applications



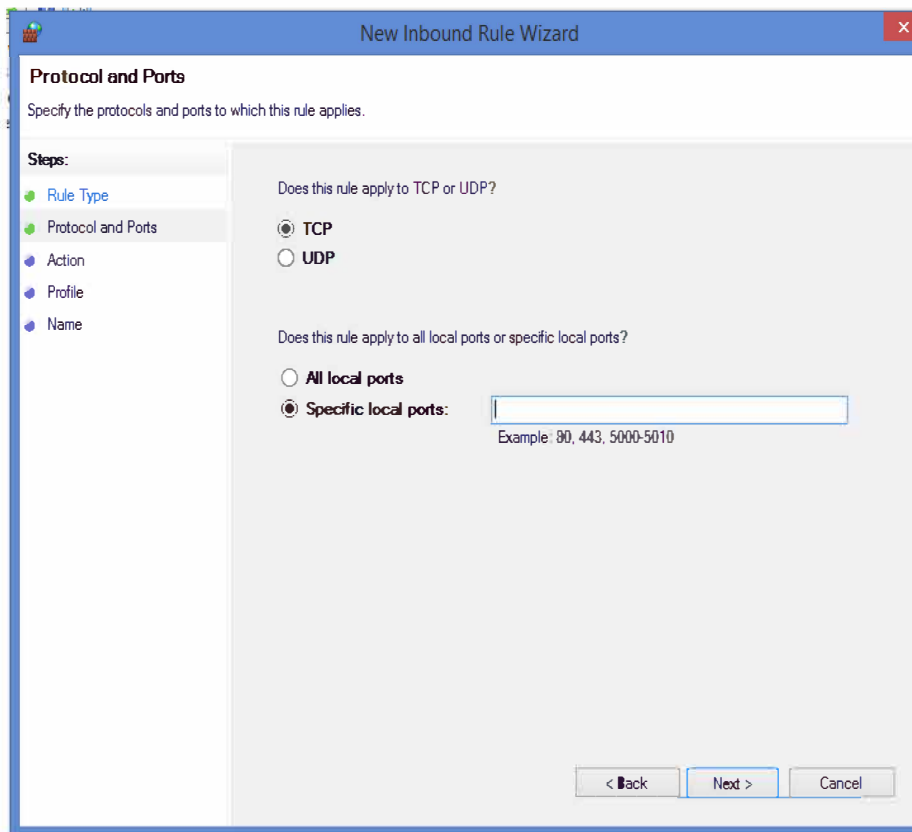
(b) Interface of a windows firewall



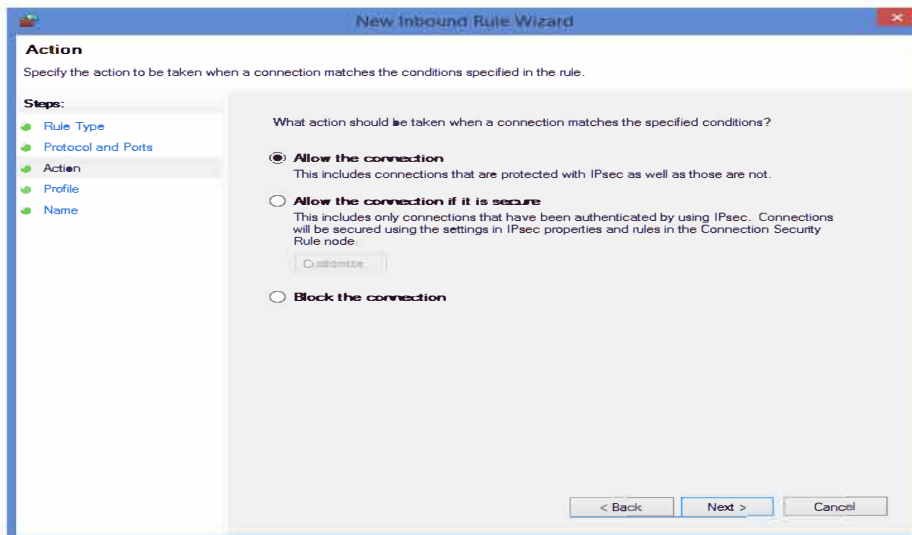
(c) Configured inbound rules on a windows firewall



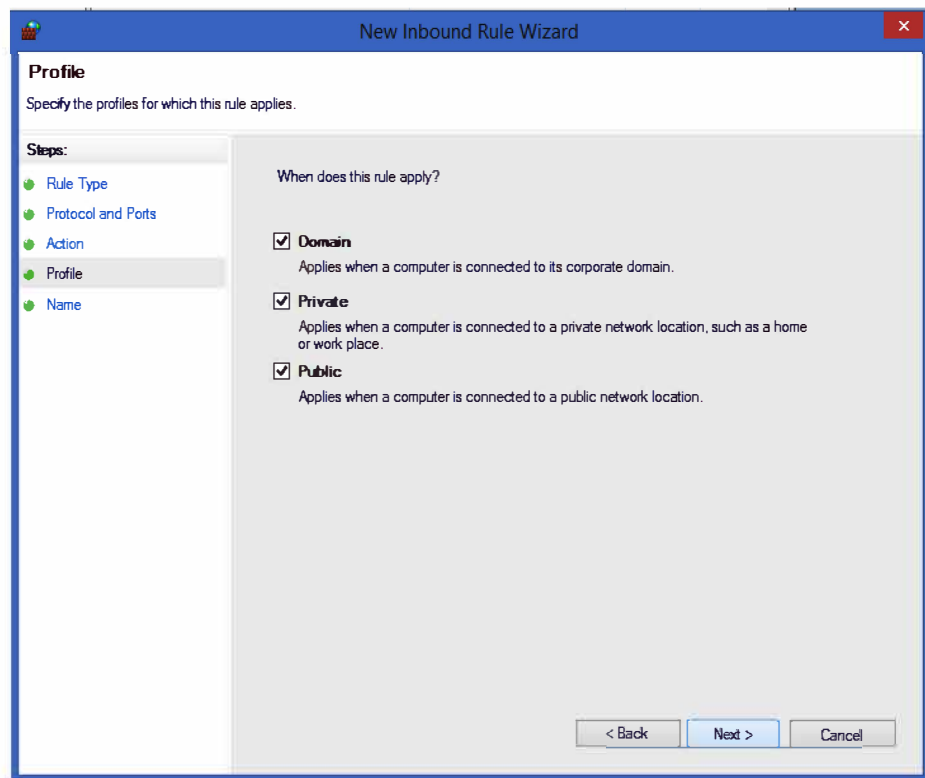
(d) Selecting a 'rule type' for creating an inbound rule on a windows firewall



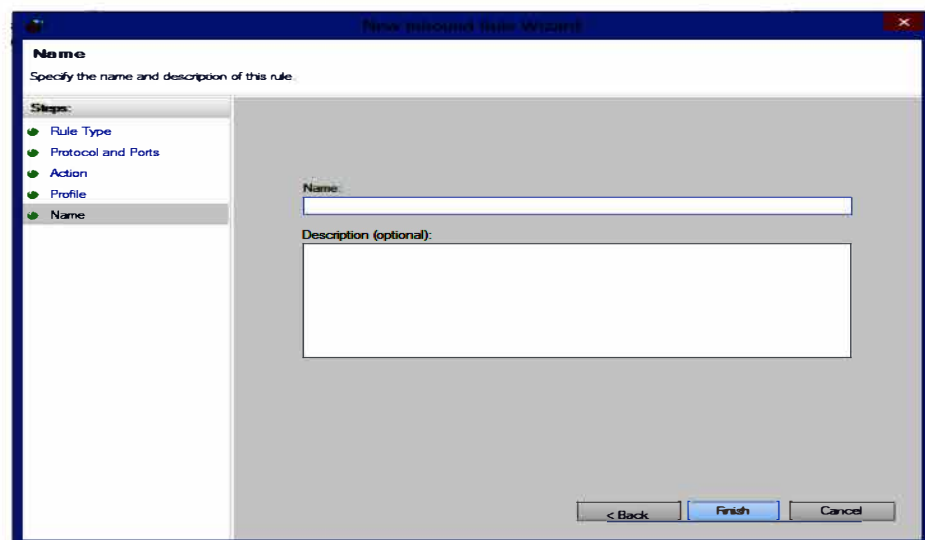
(e) Selecting protocol and ports for creating an inbound rule on a windows firewall



(f) Specifying action to be taken upon matching in an inbound rule on a windows firewall



(g) Specifying profile for which this inbound rule apply



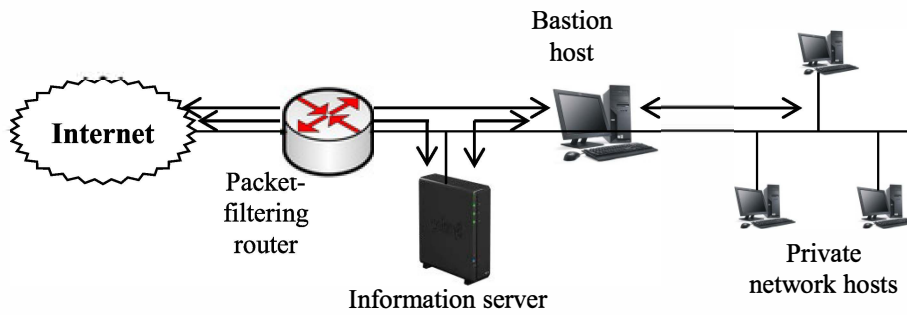
(h) Specifying name and description for an inbound rule

Figure 15.6 personal firewall

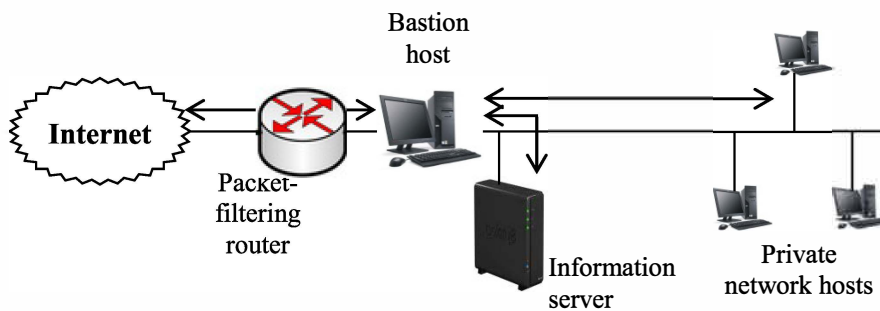
Check your progress 1

- What is firewall? Where it is implemented?
- What is the primary purpose of a firewall?
- What does firewall do if it exists as part of organization router?

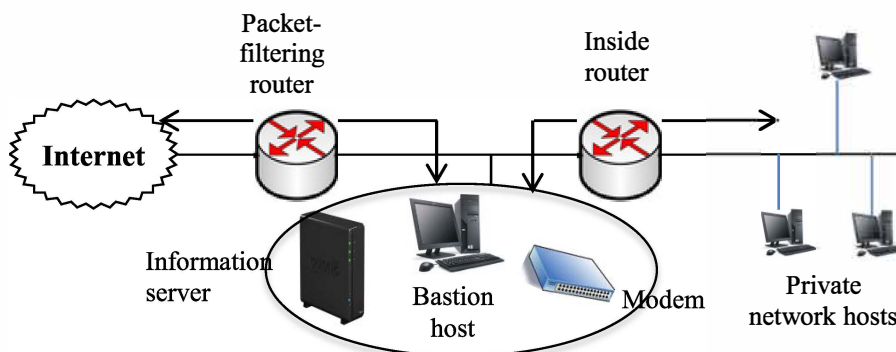
15.4 PRACTICAL FIREWALL CONFIGURATIONS



(a) A single-homed bastion host firewall system



(b) A dual-homed bastion host firewall system



(c) Firewall system having screened-subnet

Figure 15.7 Firewall Configuration

Practically, complex firewall configurations with more than one gateway or proxies on a network are common. These are classified into following three cases [1]:

- (1) Screened host firewall, single homed firewall (**Figure 15.7 (a)**). It has two systems:

➤ A router performing packed filtering functions. It allows:

- Inbound traffic destined for bastion host only.
- Outbound traffic from bastion host only.
- Bastion host running application level proxies. It performs authentication and proxy functions.

The combined use of two systems makes screened host, single homed firewall more secure than single packet-level or application-level firewall. Also, implementation of both single packet-level and application-level firewall make security policy more flexible. This also makes intruder's task difficult as two systems are to be penetrated now instead of one.

A public web server (information server) can also be deployed and router may be configured to provide direct Internet access by web server. This configuration has a problem that in case the first system i.e., packet filtering router is compromised, direct flow of traffic between Internet and internal host may result.

- (2) Screened host firewall, dual homed firewall (**Figure 15.7 (b)**). It provides dual layer security similar to that of Screened host firewall single homed bastion. It has advantage over single-homed bastion that in case the first system i.e., packet filtering router is compromised, direct flow of traffic between Internet and internal host is not possible.
- (3) Screened subnet firewall (**Figure 15.7 (c)**). It has following characteristics:
 - It is most secure among the three firewall configuration.
 - Two packet-level filtering routers are used: (i) Between Internet and Bastion host. (ii) Between Bastion host and internal network.
 - An isolated subnetwork between two packet-level filtering routers is formed. It has Bastion host, information server, modems etc.
 - An important characteristic is that traffic across screened subnet is blocked. Though both outside (Internet) and inside host can access the (common) screened subnet.

The advantages of this configuration are:

- It provides three levels (two routers and one Bastion host) of protection against intruders.
- Outside network (Internet) cannot directly access internal network that is internal network is invisible to Internet.
- Inside network cannot directly access and construct direct routes to outside network (Internet).

Realizing DMZ Networks

Demilitarized zone (DMZ) refers to a "screened subnet" that is located between an external and an internal firewall. The external firewall is placed inside the external boundary router and is used to provide

protection to the DMZ systems whereas internal firewall(s) is placed after the DMZ systems and is used to protect the organizational systems. Internet connectivity is maintained via external firewall.

The systems that lie in the DMZ are the ones that constantly maintain and promote the external connectivity. These may include a web server, an e-mail server, or a domain Name Server (DNS) server. Depending upon the policy, the external firewall provides access control and protection to the DMZ systems. Thus, systems lying in the DMZ are externally accessible and enjoy firewall protection. The external firewall may also provide some basic protection for the organizational network. As another option (or common practice), the DMZ may also be placed on another network interface of external firewall which is different from the interface used to connect the internal networks [1][2].

The organizational network may have one or more internal firewalls for protection. The internal firewall(s) serve the following purposes:

1. As compared to external filtering, the internal firewall provides more strict protection and filtering to the organizational servers and workstations from external attacks.
2. The internal firewall protects the organizational network from attacks launched from DMZ systems and, DMZ systems from attack launched from internal hosts.
3. Organizational network may be divided into multiple segments and each segment may have internal firewall. Thus, network segments are protected from each other as well via their respective internal firewalls.

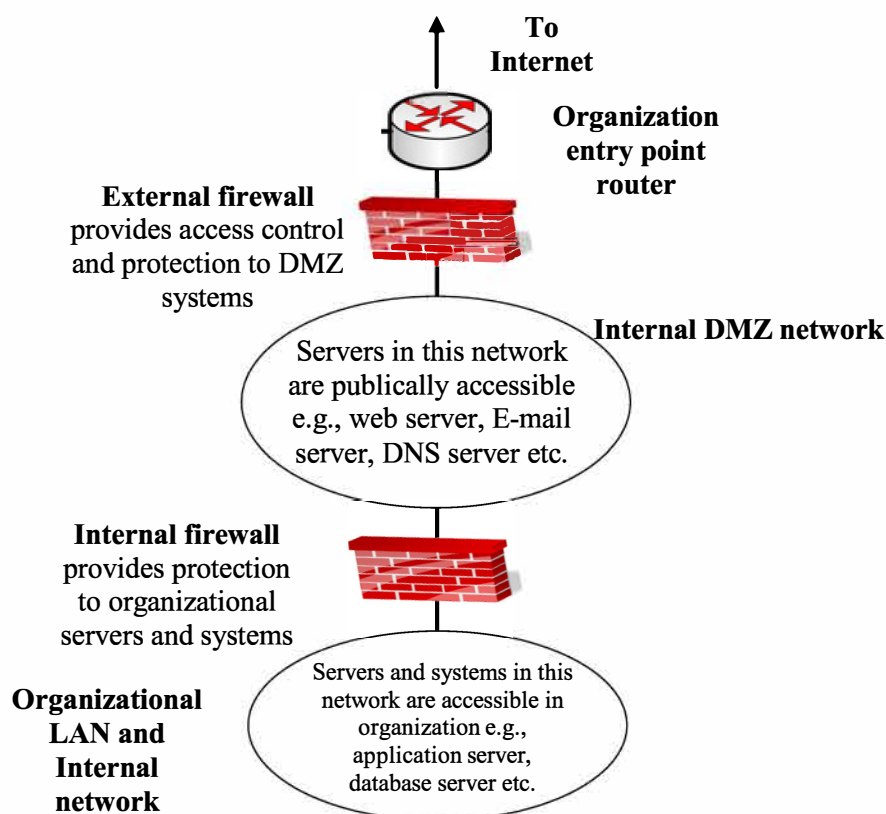


Figure 15.8 A network having Demilitarized Zone (DMZ)

Utility of Virtual Private Networks

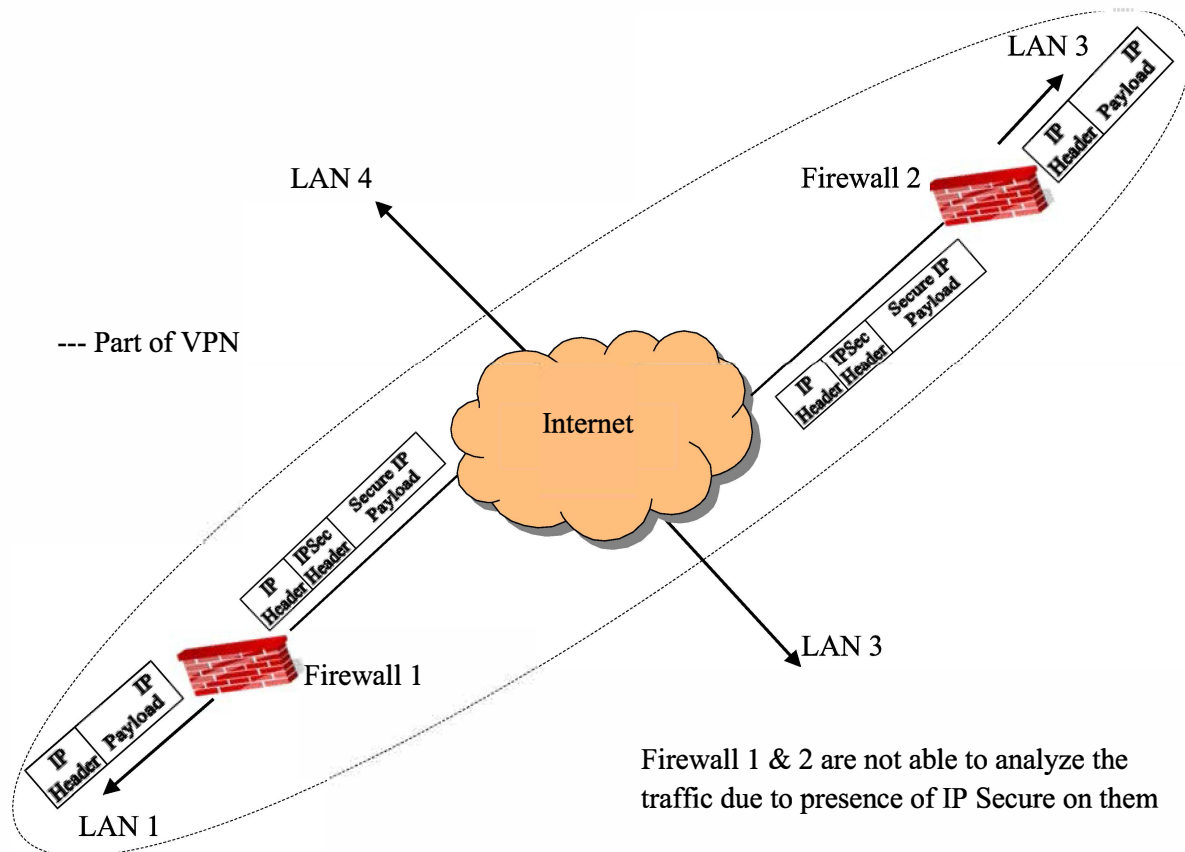


Figure 15.9 Use of VPN to enhance security in Distributed Environment

A virtual private network (VPN) consists of a set of interconnected computers (workstations, servers, and databases etc.) over a relatively unsecure network. It uses encryption and special protocols to provide security.

In VPN, the existing public network (Internet) can be used to interconnect sites (LANs) in safe manner. This provides cost savings in comparison to establishing and using a dedicated private network. This also offloads network management tasks to the public network provider. In VPN, the same public network provides an access path for login into the organizational systems from remote sites.

Network managers may use these VPN and firewalls, to cater the security in distributed computing environment in a better way. In this method, IPsec is implemented internal to the firewall (**Figure 15.9**). Thus, (VPN) traffic passing through the firewall remains invisible to the firewall due to encryption (by IPsec) and hence, firewall is unable to perform functions like access control, logging, or scanning for viruses.

Incorporating Distributed Firewalls

A distributed firewall system (**Figure 15.10**) has both host-based

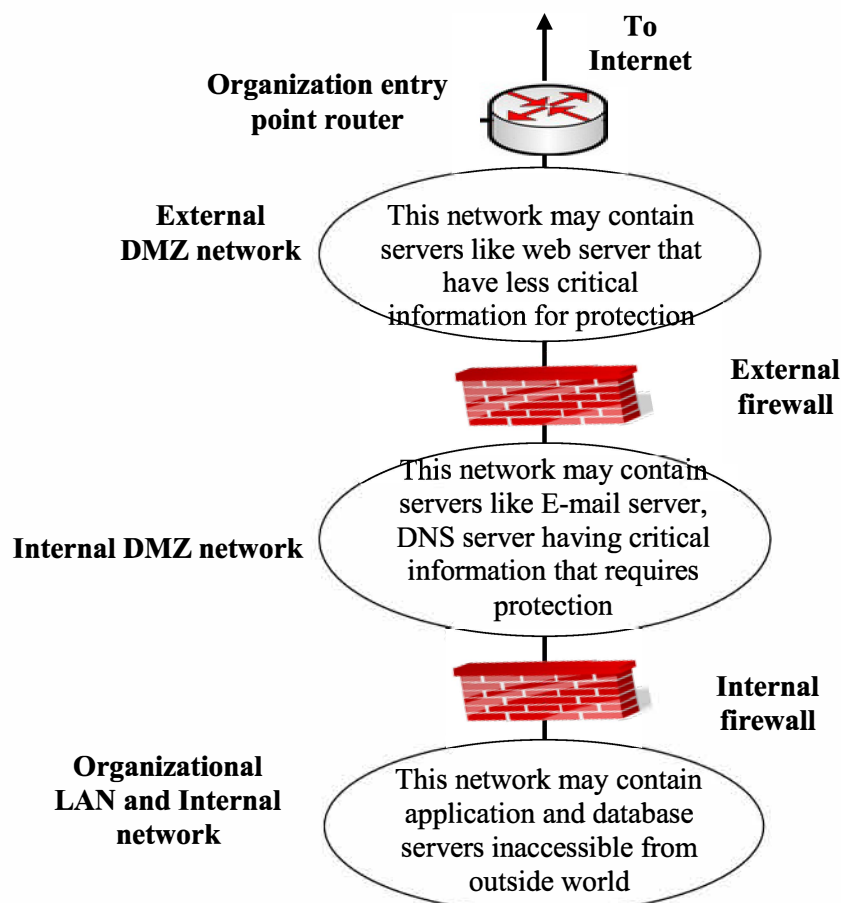
firewalls and stand-alone firewall devices. It works under a central administrative control. The configuration shown in figure may be used by large businesses and government organizations.

Host-resident firewalls are installed and used on servers and workstations. Personal firewalls are installed and used on local and remote user systems. Administrator set policies on these firewalls and monitor security of entire network. These firewalls protect specific machines and applications against internal attacks [1][2].

Two stand-alone firewalls namely, external firewall and internal firewall are used in such distributed system. They provide global protection.

Distributed firewalls may support two DMZs: an external DMZ and an internal DMZ. External DMZ is established outside the external firewall. Internal DMZ is established in between external and internal DMZ. An external DMZ may contain web servers. The web servers have less critical information on them and therefore they may need less protection and are hence placed in external DMZ. These servers are protected by host-based firewalls.

A distributed firewall configuration involves monitoring tasks like log aggregation and their analysis, firewall statistics collection and remote monitoring of individual hosts.



All hosts and servers in this distributed firewall system have host-resident firewalls installed on them

Figure 15.10 An elaborate view of Distributed Firewall System

Check your progress 2

- a. What are advantages and disadvantages of screened host firewall, single homed firewall?
- b. What is Demilitarized zone (DMZ)? How it is useful?

15.5 SUMMARY

Firewall is an important network security measure. It implements security policy and may be deployed on individual system or at the network perimeter where entire traffic can be monitored and filtered in an organization. In this unit, you should have learnt the characteristics of firewall and associated weaknesses. Various types of firewalls like packet-filter, stateful inspection, application proxy, circuit-level exists. Firewalls have three possible configurations (i) Screened host firewall, single homed firewall, (ii) Screened host firewall, dual homed firewall and (iii) Screened subnet firewall. The unit also provides merits and demerits of these configurations. Concepts of Demilitarized zone (DMZ), Virtual Private Network (VPN) and distributed firewalls are discussed towards end of this unit.

Terminal Questions

1. *How can we configure personal firewall on a window system?*
2. *Under what conditions firewall proves to be beneficial.*
3. *What are the design goals for a firewall?*
4. *How does a firewall enforce a security policy?*
5. *What is a packet filtering router? Mention some of its weaknesses.*
6. *Differentiate between:*
 - (i) *A packet-filtering router and a stateful inspection firewall?*
 - (ii) *Default discard and default forward policy in firewall.*
7. *Explain the following gateways:*
 - (i) *Application-level gateway*
 - (ii) *Circuit-level gateway*
8. *How can tiny fragment attack be overcome?*
9. *Explain various firewall configurations.*
10. *How many type of spoofing is there? Why does it create problem to the firewall?*

References:

1. Stallings, William, "Chapter 20: Firewalls", Cryptography and Network Security, Pearson Education, Fourth Edition, 2010.
2. Stallings, William and Lawrie Brown, "Chapter 9: Firewalls and Intrusion Prevention Systems", Computer Security:

Principles and Practice, Pearson Education, Second Edition, 2012.

3. [https://en.wikipedia.org/wiki/Firewall_\(computing\)](https://en.wikipedia.org/wiki/Firewall_(computing)) accessed on 12.12.2016.
4. <https://www.microsoft.com/en-us/safety/pc-security/firewalls-what-is.aspx> accessed on 12.12.2016.
5. http://www.cisco.com/c/en_in/products/security/firewalls/what-is-a-firewall.html accessed on 12.12.2016.
6. Bellovin, S., and Cheswick, W. "Network Firewalls," IEEE Communications Magazine, September 1994.
7. <https://answers.syr.edu/display/os/Enable+Firewall+in+Windows+8+and+8.1> accessed on 24.12.2016
8. <http://www.online-tech-tips.com/windows-10/adjust-windows-10-firewall-settings/> accessed on 24.12.2016

