1

INTRODUCTION

Unit Structure :

1.0 OBJECTIVES

After going through this unit you will be able

- To understand the important and basic concepts of computer and network security
- To study and practise various cryptography techniques
- To identify and analyse security attacks, vulnerabilities and threats
- To learn various security domains to protect assets

1.1 INTRODUCTION

Use of Internet is growing exponentially in today's world and with this global internet use, users are worried about various security parameters like what to protect, how to protect, what are various vulnerabilities, threats and attacks etc. In today's digital globalization, the need of information security is in high. Currently, data security professionals are in more demand and definitely its importance will be growing day by day. We will focus on security policies, services and mechanisms, various types of attacks on network, data and on assets of the organization. We will also discuss various measures to be taken to adapt and monitor preventive measures.

Information Security

- Information security is a process or method of protecting information and information systems from unauthorised access, interruption, data alteration, disclosure, recording or inspection of information.
- The information can be any form such as data over the cloud, social media data, data of any customer or supplier or any user, phone data etc.

Network Security

- Network security is a process or method of protecting networking infrastructure from an unauthorized access and risks and maintaining integrity and confidentiality of data and networks.

- The whole world has transformed into digitization and every business wants to provide security services which protect their stakeholder's data.
- Network security helps to protect sensitive information from various attacks and also protects your assets.

1.2 SECURITY TRENDS

- Implementation of digitalization in every aspect of life is not only making each one of us dependent on it but also warning us data security is a primary goal to maintain privacy.
- Constant changes in technologies have shifted network security trends as virus, worms, adware, phishing, data breaching are considered to be normal activities.

A. IOT

a. Internet of Things (IOT) refers to physical devices connected to internet and share data. Eg. Fitness trackers, smartwatches, voice assistants etc.

b. Open communication between various devices lead to unknown attacks or software bugs.

B. Ransomware

a. Ransomware attacks are rising.

b. Hackers use ransomwares to hide their malicious code into it and earn financial gains.

c. Exploits have been utilized by mobile malware with the help of social engineering tricks which can easily get access or control on infected device.

C. Rise of Artificial Intelligence

a. The use of Machine Learning and Artificial Intelligence is growing rapidly in the industries as it plays very important role in network security.

b. Though practical implementation of AI and ML is a paramount in developing automated security systems and threat detection, there is a risk as ML and AI may get exploited by attackers as attacker are getting smarter.

D. Mobile security

a. Mobile threats such as spyware, vulnerabilities with android operating system, DDOS attacks, spam SMS, stealing of data causing harm to individual as well as organization which is continuously evolving.

b. In this era of digital transformation intruders are constantly searching exposures and new ways to target assets of the organization.

E. Cloud computing

a. Though cloud offers scalability, efficiency and low cost but they could also be a prime target for cyber criminals.

b. Misconfigured cloud can cause unauthorised access, data hijacking, cloud maintenance, unsecured network, weak passwords and some accidental or intentional threats.

F. Social engineering and phishing

a. SMS phishing, likejacking, clickjacking, SIM jacking and duplicate websites are also the threats which is worrying to users as these tricks give access to private and sensitive information to attackers.

G. Social networking attacks

a. Attackers try to connect to target by using social media as their medium and by spreading social network.

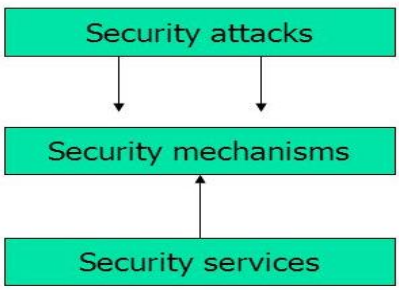
b. Usually, cyber criminals spend a lot of time in social networking creating fake links or offers and targeting people and networks and gaining access to personal information.

1.3 THE OSI SECURITY ARCHITECTURE

It is very essential to evaluate security needs of an organization effectively and to assess and select security policies accordingly. To set up parameters for information and network security appropriately for an organization, there is a need of defining security requirements properly and various methodologies towards satisfying it.

OSI Security Architecture was developed as an international standard. For Information and network security, organizations need to developed security features for their services and products related to these services and mechanisms.

OSI security architecture is a systematic way of defining and providing requirements of the security and it emphasises on following concepts:-



- **Security attacks : -**

  The actions which comprise information security of an organization

- **Security mechanisms :-**

  The practises that are designed to identify, prevent and recover from a security attacks.

- **Security services :-**

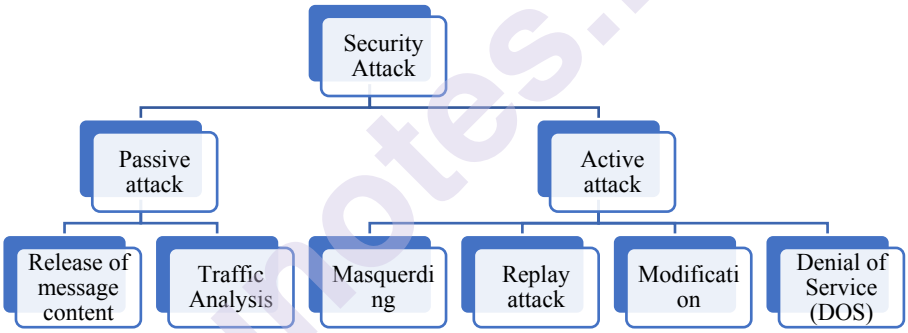
  Services which improve systems and data processing and transferring security and also planned to counter security attacks.

1.4 SECURITY ATTACKS

Security attack is an unauthorized access to sensitive data to expose, steal or damage it.

There are two types of security attacks :-

- Passive attack
- Active attack



1.3.1 Passive attack

- Attacks which attempt to observe or make use of information but don't affect resources of the system are called as passive attacks.
- Inspection of data transmission and gaining access to data
- These attacks do not modify contents of original message
- So, these attacks are very difficult to detect
- And that's why it is always better choice to prevent these kinds of attacks rather than identifying and correcting them
- These attacks are threats against data confidentiality

  Protective measures :-

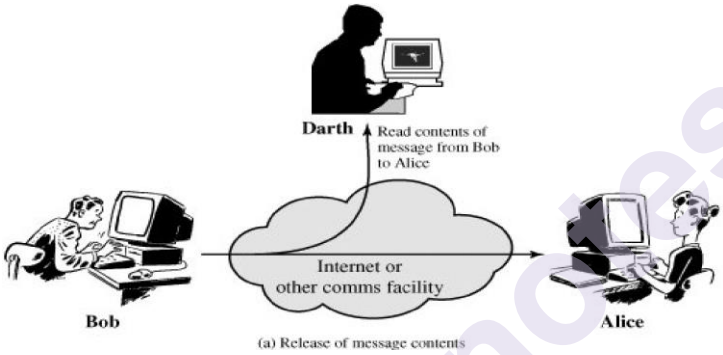
  - Avoid disclosing sensitive and personal information over social media or online as attackers make use of them for hacking the network

- Use encryption for data masking so that information is unreadable by an intruder

- These attacks are sub categorized into following two types :-
  - Release of message contents
  - Traffic analysis

Release of message content

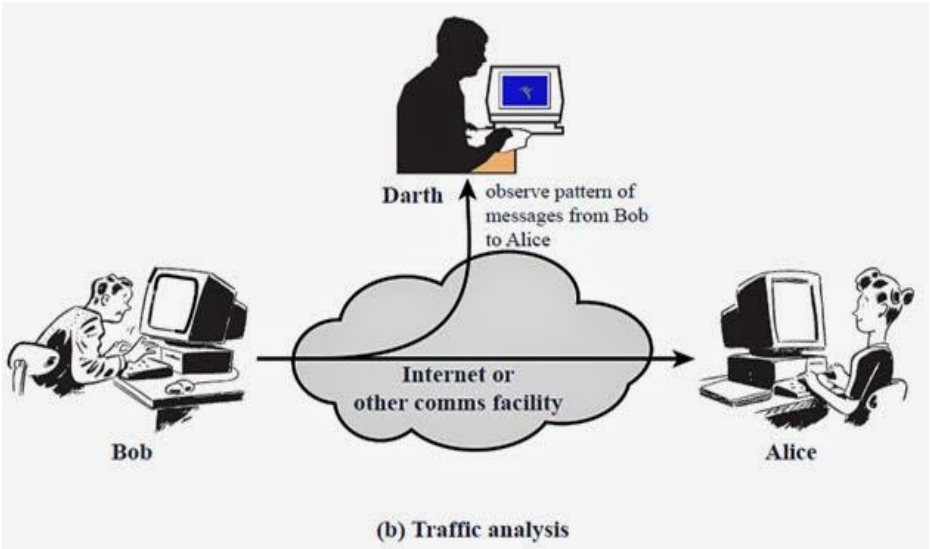
- Unauthorized access of data
- Interception of data
- Disclosure of confidential information
- Threat against confidentiality
- To prevent such attacks, encryption technique can be used for data masking.



(a) Release of message contents

Traffic analysis

- Observing online data traffic pattern
- Analysing frequency and length of data packets
- Threats against data confidentiality



(b) Traffic analysis

Active attack

- Attacks which attempt to modify resources of the system or their operations are called as active attack
- Data alteration or modification
- False stream creation
- More harmful as compared to passive attacks
- Threats against integrity and availability
- Preventing theses attacks are very difficult as it involves high scale of software and physical vulnerabilities
- The victim gets notified about this type of attack as it involves data modification

  Protective measures :-

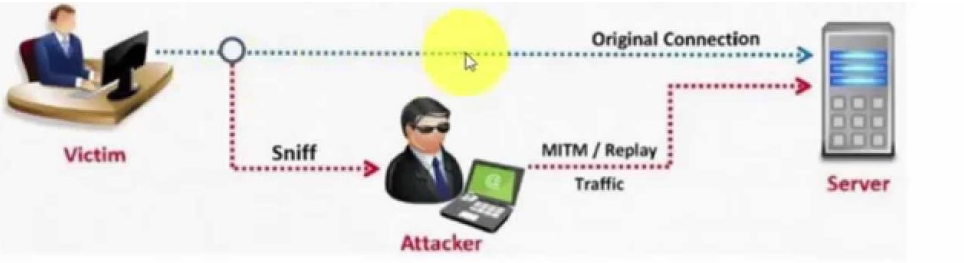
  - Use of strong password is recommended
  - Use of one time password for authentication during communication
  - Use of session keys to prevent access once session time out.
- Active attacks are classified into four types :-
  - Masquerading
  - Replay attack
  - Modification
  - Denial of Service (DOS)

Masquerading

- A system or a user posing as a false identity to gain access or to modify information
- Also known as spoofing
- One entity pretends to be other entity

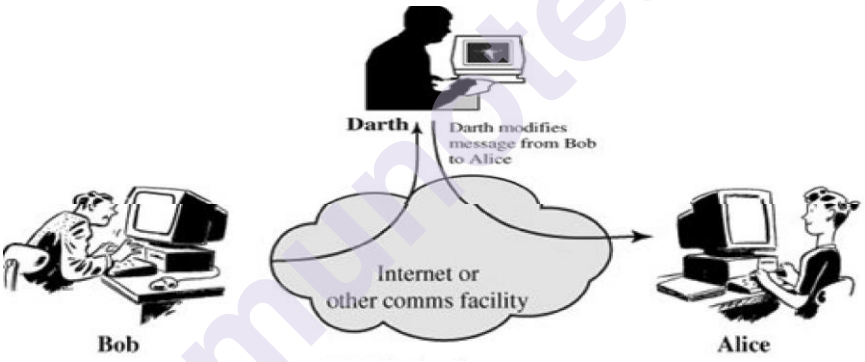
Replay attack

- Passive capturing of data packet and its transmission to produce authorized effect
- In order to get benefit, repeating some actions on data and reusing captured data. This action gets performed little later than the original time



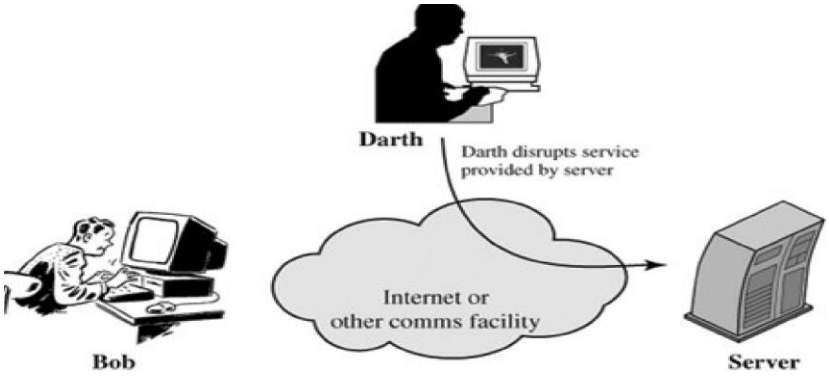
Data Modification

- Some part of the message is modified or the message is delayed or restructured to create unauthorized effect
- Alteration in packet header eg destination address field will redirect packet to some other destination



Denial of Service (DOS)

- This uses authorized users from using services
- Attacker intends to target on a specific system
- Interruption in entire network by disabling the network
- Flooding the messages on targeted system to lower down its performance
- Prevents regular use of services
- Incapacitating the server or targeted machine
- Sometimes targeted system will be completely collapsed or shut down due to packet over flooding
- Usually reported such attacks in internet related services



Difference between Active attack and passive attack

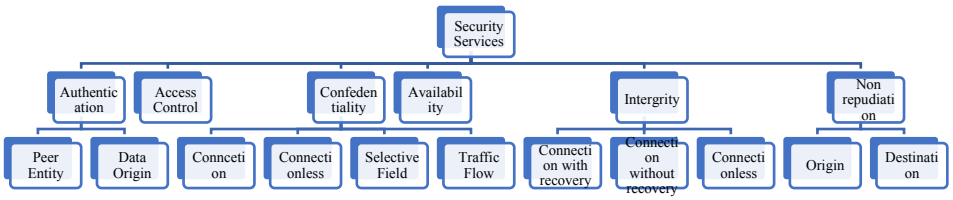
Active Attack	Passive Attack
Message modification	Message observation
Data packets changed	Data packets unchanged
Threats against integrity and availability	Threats against confidentiality
Detection of attacks are possible	Prevention of such attacks are possible
Resources can be changed	No effect on resources
Influences system resources	Information or data is reviled
With the help of passive attacks, data is gathered to make an attack	Information collection is possible with the help of chat, personal communication or through social media
Prohibition of such attacks are very difficult	Easy to prevent

1.5 SECURITY SERVICES

- Services which are provided to enhance security of data processing and information sharing of an organization is called as security services.
- These services are intended to mitigate security attacks.
- These services make use of one or many security mechanisms.
- These services are defined by x.800.
- These services are provided by communicating open systems' protocol layer
- Services ensures sufficient security for systems and data communication processes.

- Processing or communication services provides protection to system resources

Security services or principles of security are classified as follows :-



Authentication

- Providing the assurance of origin of identity
- Establishing proof of identity
- Guarantees identity proof of sender and receiver before carrying out any communication
- Violating authentication is fabrication
- Fabrication means failing to prove authentication
- Receiver needs to check for actual sender's identity as what is claimed by him
- There can a secret key shared between parties involved in communication
- Authentication can make use of digital signature
- Authentication can be verified by third party also.
    - **Peer entity**
        - In logical connection to assure the identity of the entities connected
    - **Data origin**
        - In connectionless transmission to assure identity of the sender as it is claimed
        - Not useful while duplication or modification of data
        - Useful in e-mail system

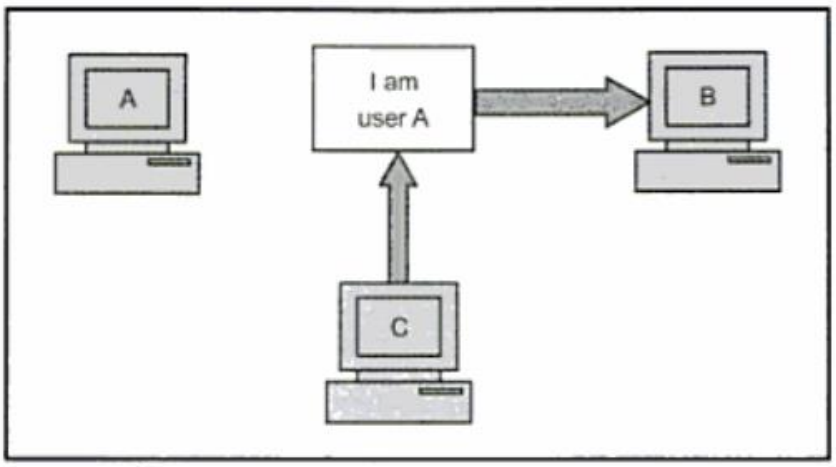


Fig. 1.5 *Absence of authentication*

Access Control

- Prevention from unauthorized access to a resource
- Policies that define who can access what
- Limiting and controlling access of host and its applications.
- If the users or services are authenticated then access controls are allotted to them.
    - **Role management**
        - Roll of a user
        - It defines which user can access what resources.
    - **Rule management**
        - It defines which resources can be accessible under what circumstances.
        - Access Control List (ACL) matrix needs to prepared which defines list of users, their roles and what they can access.

Confidentiality

- Assuring data privacy
- Protecting unauthorized data access
- Securing data transmission from passive attacks
- Only intended sender and receiver should be able to see contents of the message.
    - **Connection**
        - Protection for all user's data over a connection
    - **Connectionless**

- Protection for all user's data over a single data block

- **Selective field**
  - Confidentiality is made applicable on selective fields over a connection or on a single data block

- **Traffic flow**
  - Protecting analysis of traffic flow including source, destination length, frequency or any other parameters of traffic

- Data encryption is needed before data transmission so that intruder over the network won't be able to read contents of the message.
- Loss of confidentiality is known as interception.

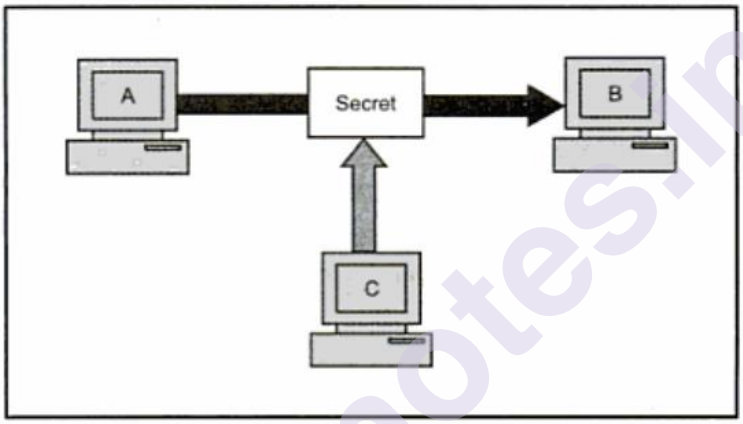


Fig. 1.4 *Loss of confidentiality*

Availability

- Availability of system and its resources as and when required by an authorized entity
- Depends upon proper management and system control of resources
- Attack against availability is interruption

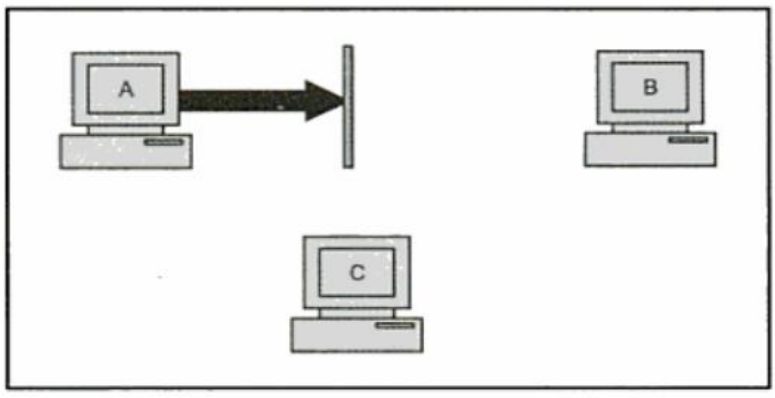


Fig. 1.8 *Attack on availability*

Integrity

- Data received by the receiver is exactly same as send by the sender
- Keeping data intact till it reaches to destination
- Data transmission do not allow data alteration, insertion, deletion or replay
- Related to active attacks so detection is more important rather than prevention
- Attack against integrity is modification or alteration
    - **Connection with recovery**
        - Integrity of user data on a connection and discovering data alternation, insertion, deletion or replay if any
        - Reports stream modification and denial of service
    - **Connection without recovery**
        - Integrity of user data on a connection and discovering data alternation, insertion, deletion or replay if any but without data recovery
    - **Connectionless**
        - Integrity of user data on a connectionless communication and discovering data alternation, insertion, deletion if any.
        - Replay detection is possible in some cases

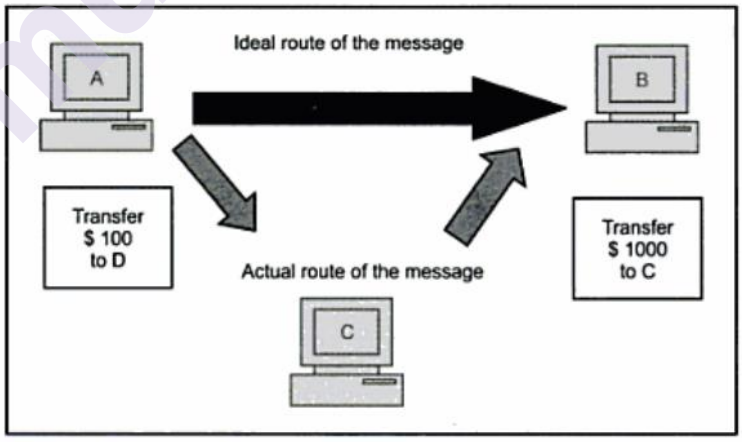


Fig. 1.6 *Loss of integrity*

Non repudiation

- Parties involved in communication can not refuse to sending or receiving of data

- Protection against refusion of identities by sender or receiver after transmitting a message
- This principle of security is with respect to ownership of the message
- Non repudiation can be proved with the help of authentication and integrity
- Attack against non repudiation is repudiation
    - **Origin**
        - Message identity proof shared by the sender and after sending denying it
    - **Destination**
        - Message identity proof shared by the receiver and after receiving denying it

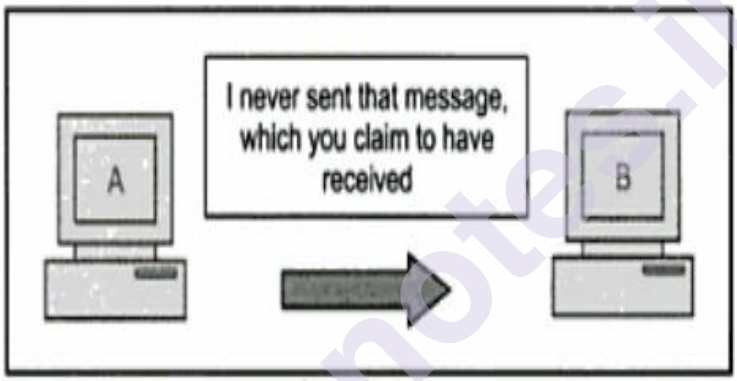
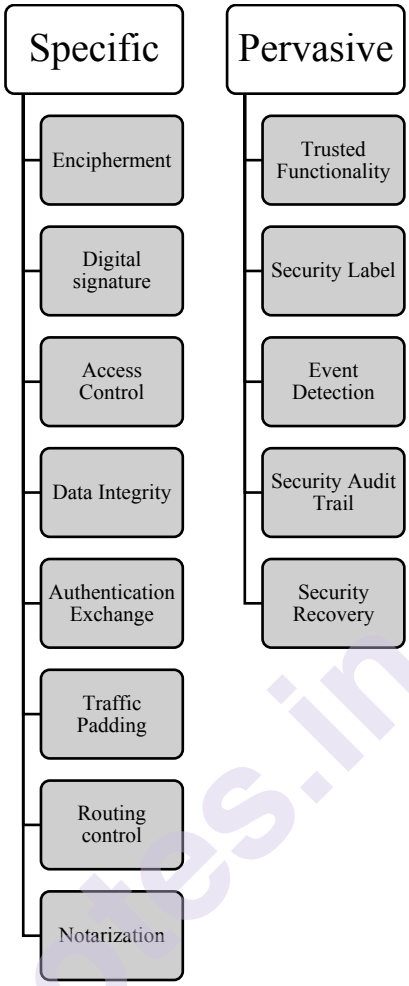


Fig. 1.7 *Establishing non-repudiation*

1.6 SECURITY MECHANISMS

- Security mechanisms are the techniques or tools to implement security principles or security services in an organization
- These mechanisms can be operated by itself, oneself, administrators or others to provide specific services
- Security mechanisms are set of processes designed to detect, prevent and recover from various types of threats and attacks
- Dependency on protocol layers or security services have divided security mechanisms into two categories
    - Specific security mechanism
    - Pervasive security mechanism



Specific Security Mechanism

- Mechanisms that are specific to any protocol layer or security services
  - **Encipherment**
    - Converting data to unreadable format through encryption to maintain confidentiality
      - For data transformation various mathematical algorithms are used
      - Data transformation and recovery depends on encryption keys
      - Difficulty level of encryption is dependent on mathematical algorithm and no of keys used
      - Techniques which are used in encipherment are steganography and cryptography
  - **Digital Signature**
    - Security achievement by appending original data with invisible digital data
    - Form of an electronic signature

- Digital signature is added to document by sender and verified by receiver
- Authenticating sender is been done by using this mechanism
- Sender uses his own private key and corresponding public key will be share with receiver. Receiver on the other side uses this public key to assure authenticity of the sender who has claimed data has been sent by him
- Authenticity and integrity of the data is been proved
- Data forgery is protected

- **Access Control**
  - Enforcing access permission for resources
  - Controls unauthorized access of data
  - Example is passwords, Pin numbers, usage of firewall
  - Defining role of the users and their permissions
- **Data Integrity**
  - Assurance of keeping data intact during transmission
  - Original data is appended with a code which needs to be same during transmission and verified by sender and receiver to prove data integrity
  - Original data contents and appended check value is shared with receiver.
  - Receiver at the other end also computes new check value based on some predefined algorithm.
  - This newly created check value and the value which shared by sender is then compared.
  - If these both are same then data integrity is maintained
  - If these both values do not match then there is considered to be data modification while transmission of data
  - Detects unauthorized modification of data
- **Authentication Exchange**
  - This mechanism involves identity of parties should be proved who involved in communication
  - Example is two-way handshaking mechanism used to establish connection
  - Entities involved in communication proves their identity to each other

- **Traffic padding**
  - Insertion of bits into data gaps
  - Prevent against traffic analysis attack
  - Generation of fake instance of data, false communication
- **Routing control**
  - Selection of specific secure routes for data transmission
  - Enables routing changes when security breach is suspected
  - Choosing and continuously changing various available routes between communicating entities
  - Preventing traffic analysis attack on a specific route
- **Notarization**
  - Involvement of trusted third party
  - Interference of third party to ensure certain data exchange properties
  - Mediator between sender and receiver to reduce any chances of data conflict
  - Record maintenance of request from sender and receiver in case later denied
  - Prevents repudiation

Pervasive Security Mechanism

- Mechanisms that are not specific to any protocol layer or security services
  - **Trusted Functionality**
    - Supposed to be correct with respect to some criteria like as per security policies
    - Extending the scope or effectiveness of security mechanisms
    - Functionalities providing access to or directly access security mechanisms should be trustworthy
  - **Security Label**
    - Marking for a resource
    - Naming or assigning security attributes to the resource
    - Supplementary data associated with transferred data
  - **Event Detection**
    - Security related events detection
    - Detecting events associated with security violation

- **Security Audit Trail**
  - Independent review and monitoring of system records and activities
  - Analysis capability of system controls
  - Ensuring compliance with existing policies and operational procedures
  - Discovering security breaches
  - Suggests changes in policies, controls and procedures if needed
  - Collection of data and its potential use to facilitate security audit
- **Security Recovery**
  - Handles requests from mechanisms like management functions and event handling
  - Takes necessary recovery actions for applying set of rules

1.7 QUESTIONS

1. What is computer security and discuss its objectives?
2. Write a note on OSI security architecture.
3. State and explain types of active attacks.
4. Explain various principles of security.
5. List and briefly define categories of security mechanisms.

REFERENCE FOR FURTHER READING

- Cryptography and Network Security, Atul Kahate, Tata McGraw-Hill, 2013.
- Cryptography and Network Security: Principles and Practice 5th Edition, William Stallings, Pearson,2010
- Syngress.The.Basics.of.Hacking.and.Penetration.Testing.Aug.2011
- Cryptography and Network, Behrouz A Fourouzan, Debdeep Mukhopadhyay, 2nd Edition,TMH,2011

❄❄❄❄❄❄❄

2

CLASSICAL ENCRYPTION TECHNIQUES

Unit Structure :

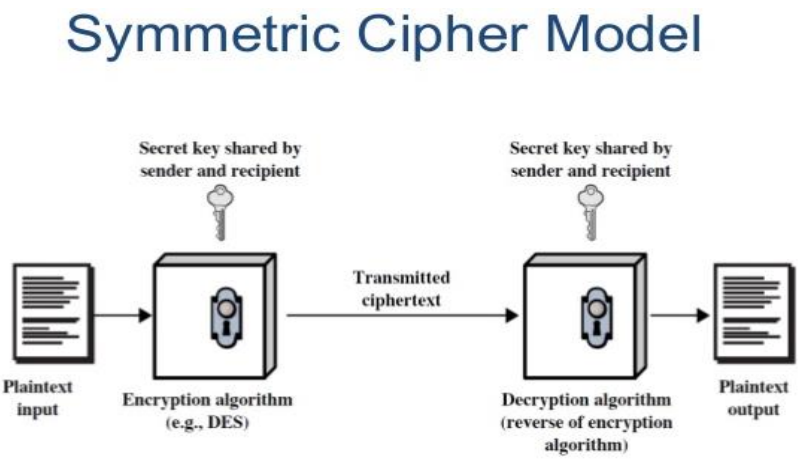
2.0 OBJECTIVES

Symmetric encryption or one key encryption or conventional encryption is a type of cryptosystem where encryption and decryption process is done using same key.

- An original message is called **plain text.**
- Converted form of message is called **cipher text**.
- Conversion of plain text to cipher text is called as **encryption**.
- Conversion of cipher text back to plain text is called as **decryption**.
- Study of techniques used for encryption is called as **cryptography**.
- Schemes used for encryption is called as **cryptographic system**.
- Tools or techniques used for message deciphering without knowing the details of enciphering is called as **cryptanalysis**.
- Study of cryptography and cryptanalysis together is called as **cryptology**.

2.1 SYMMETRIC CIPHER MODEL

Symmetric Cipher Model contains five components.



- **Plain text** – Original data that is to be feed as input to the algorithm
- **Encryption algorithm** – Various algorithms used to convert plain text into cipher text. These algorithms are classified as substitution and transposition algorithms.
- **Secret Key** – The value which is input to an algorithm and independent of plain text and that algorithm.
- **Cipher text** – Converted data that is received as output. Output is based on plain text and secret key used for algorithm.
- **Decryption algorithm** – Reverse of an encryption algorithm. It used cipher text and secret key and generates plain text.

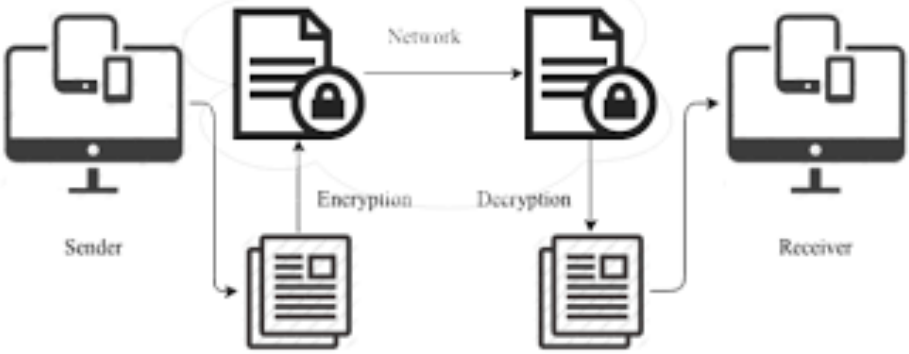
Requirements of symmetric key cryptography

- Use of strong encryption algorithm
- Secret key should be kept private and secure and not shared to anyone else who are not involved in communication

Characteristics of cryptography

- **Techniques used in encryption**
  - **Substitution** – Each character of plain text is replaced by other character or symbol
  - **Transposition** – Rearrangement of characters of plain text
  - Multiple rounds of substitution and transposition is possible
- **Number of Keys used**
  - If same key is used for encryption and decryption process by sender and receiver then process is called as
    - Symmetric key cryptography
    - Single key cryptography
    - Secret key cryptography
    - Conventional cryptography
  - If different keys are used for encryption and decryption process by sender and receiver then process is called as
    - Asymmetric key cryptography
    - Two key cryptography
    - Public key cryptography
- **Processing of plain text**
  - Processing one block of input at a time and producing corresponding output block for each input block is called as **block cipher**.

- Processing input in continuous format and producing corresponding output one element at a time as long as it goes is called as **stream cipher**.



Approaches for symmetric key cryptography attack

- **Cryptanalysis**
  - Depends on nature of the algorithm used
  - Knowledge of basic features of plain text
  - Exploits features of an algorithm
  - Attempt to find plain text or secret key used
  - Finds weaknesses in cryptography system
- **Brute force attack**
  - Trying every possible key for decryption process
  - Use of large keys fails this attack

Basic approaches of cryptography are substitution and transposition techniques. Various methods involved under these techniques are as follows:-

Cryptography Techniques	
Substitution Techniques	**Transposition Techniques**
1. Caesar cipher 2. Monoalphabetic cipher 3. Playfair cipher 4. Hill cipher 5. Vigenere cipher/ Polyalphabetic cipher 6. Vernam cipher 7. Additive cipher	1. Rail Fence 2. Columnar

2.2 SUBSTITUTION TECHNIQUES

- The techniques in which letters of plain text are replaced by other characters, symbols or numbers are called as substitution techniques.
- Plain text is considered as sequence of bits and substitution involves substitution of plain text bit pattern with cipher text bit pattern.

Caesar cipher

- Simplest substitution technique
- Invented by Julius Caesar
- Earlier known as substitution cipher
- Replacement of each character of plain text by three positions down in the alphabet
- Example :- A become D, B become E and so on
- First row below indicates plain text and second row indicates cipher text

A	B	C	D	E	...	...	X	Y	Z
D	E	F	G	H	...	...	A	B	C

Example

- Plain Text :- WELL DONE TYCS
- Cipher Text :- ZHOO GRQH WBFV

Monoalphabetic Cipher

- Mono means one i.e. one to one mapping between characters
- Each plain text character will be replaced by different cipher text character
- A become D, B become K, Z become I and so on
- These replacing characters can be any set of random characters
- But only one to one mapping is allowed
- Plain: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
- Cipher: D K V Q F B J W P E S C X H T M Y A U O L R G Z N I

Example

- Plaintext: IF WE WISH TO REPLACE LETTERS
- Cipher text: WI RF RWAJ UH YFTSDVF SFUUFYA

Playfair cipher

- This technique uses keyword
- Characters in keyword are arranged in 5x5 matrix row wise from left to right and from top to bottom
- Repeated characters from keyword have to be written only once in matrix
- Fill the rest spaces in matrix with remaining characters from A – Z which are not a part of keyword.
- I and J can not be written separately. They need to write in same cell of matrix.

  For example :- Keyword BALLOON

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

Encryption process

- Break plain text into group of 2 characters
  - Eg Plain text :- MAROON
  - MA RO ON
- If both alphabets are same add x after first alphabet
  - Eg Plain text :- BALLY
  - BA LX LY
- If only one character is left add x at end
  - Eg. Plain text :- MORNING
  - MO RN IN GX
- **RULE 1**
- If both characters from a group appears in the same row, replace with immediate right characters respectively.

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

  - Consider above matrix
  - Plain text :- ON

- Cipher text :- NB
- Next character of O is N and Next character of N is B is same row

- **RULE 2**
- If both characters from a group appears in the same column, replace with immediate below characters respectively.

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

- Consider above matrix
- Plain text :- FM
- Cipher text :- MT
- Character below F is M and character below M is T is same column

- **RULE 3**
- If both characters forming a group are not in the same column or row, replace them with character on corners of rectangle. First character must be present on the same row.

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

- Consider above matrix
- Plain text :- CU
- Cipher text :- GQ
- Characters forming rectangle with C and U are G and Q respectively. We have to consider row wise. So G is in same row as C and Q is in same row as U.

- Example

- Keyword :- PLAYFAIR EXAMPLE

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

Plain text – BALLOON

Group of 2 characters

BA – DP (Rule 3)

LX (same characters can not be repeated) – YR (Rule 3)

LO – AN (Rule 3)

ON – QO (Rule 1)

Hill Cipher

- Invented by L. S. Hill in 1929
- Divide input string into matrix of size n
- Replace alphabets with numbers as below
- Divide plain text characters into block size of 2 or 3 (preferable)
- According to this block size consider matrix of 2x2 or 3x3 as secret key.
- Secret key matrix can consist any numbers and should be same for each block
- Matrix multiplication of secret key and plain text character conversion matrix
- Resultant matrix mod 26
- Find the corresponding alphabets

Example

- Plain text: "LOVE", Secret Key: $\begin{bmatrix} 20 & 3 \\ 15 & 7 \end{bmatrix}$
- "LO" → $\begin{bmatrix} 20 & 3 \\ 15 & 7 \end{bmatrix} \begin{bmatrix} 11 \\ 14 \end{bmatrix} = \begin{bmatrix} 262 \\ 263 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$ mod 26
- "VE" → $\begin{bmatrix} 20 & 3 \\ 15 & 7 \end{bmatrix} \begin{bmatrix} 21 \\ 4 \end{bmatrix} = \begin{bmatrix} 432 \\ 343 \end{bmatrix} = \begin{bmatrix} 16 \\ 5 \end{bmatrix}$ mod 26
- 2, 3, 16, 5 are transformed to cipher text "CDQF"

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

Vigenere Cipher / Polyalphabetic cipher

- Consider plain text and keyword
- If number of characters in keyword is less than number of characters in plain text then repeat keyword character in sequence.
- Write a matrix of all alphabets 26x26 in sequence of characters
- Find the mapping of one character of keyword to one character of plain text

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

- Plain text : STAY HOME
- Keyword : TYCS

  STAY HOME

  TYCS TYCS

  Intersection of S to T from above matrix which is L

  Intersection of T to Y from above matrix which is R

  Cipher text :- LRCQ AMOW

Vernam Cipher (One Time Pad)

- One time pad (OTP) is one time shared key.
- Covert plain text alpha bets into number like A= 0, B=1 and so on.
- One Time Pad is set of random characters
- OTP also has to be converted into numbers

- Add Plain text values with OTP values
- Find corresponding characters
- If addition is more than 25, subtract 26 from it

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

```
     H       E       L       L       O  message
   7 (H)   4 (E)  11 (L)  11 (L)  14 (O) message
+ 23 (X)  12 (M)   2 (C)  10 (K)  11 (L) key
= 30      16      13      21      25     message + key
=  4 (E)  16 (Q)  13 (N)  21 (V)  25 (Z) (message + key) mod 26
      E       Q       N       V       Z  → ciphertext
```

Additive cipher

- Also called as shift cipher
- Similar to modified version of Caesar cipher
- Key is considered in numeric format
- Plain text characters need to be shifted to the right according to specific positions
- If key = 7, A → H, B → I and so on
- Plain text :- S E C U R I T Y
- Cipher text :- X J H Z Z N Y A

2.3 TRANSPOSITION TECHNIQUES

- The techniques in which letters of plain text are rearranged or jumbled are called as transposition techniques.

Rain fence cipher

- Plain text is represented as sequence of diagonals
- And then read as rows
- Plaint text :- GOOD MORNING

G		O		M		R		I		G
	O		D		O		N		N	

- Cipher text :- GOMRIG ODONN

Columnar cipher

- Plain text is written in columns
- Order of the columns to read characters need to be finalized
- Read columns accordingly

  - **Simple columnar transposition technique with four columns order 4,2,1,3** :

Column 1	Column 2	Column 3	Column 4
C	O	M	E
H	O	M	E
T	O	M	O
R	R	O	W

  - Order of column is 4, 2, 1, 3
  - Eeowooorchtrmmmo

Multicolumnar cipher

- Plain text is written in columns
- Order of the columns to read characters need to be finalized
- Read columns accordingly
- Write cipher text again in columns
- And repeat above process multiple times by keeping order of columns same

  - **Multi columnar transposition technique with four columns order 4,2,1,3**:

Column 1	Column 2	Column 3	Column 4
E	E	O	W
O	O	O	R
C	H	T	R
M	M	M	O

  - Order of column is 4, 2, 1, 3
  - WRROEOHMEOCMOOTM

- In another version one keyword will be given.
- Number of characters in a keyword will be number of columns

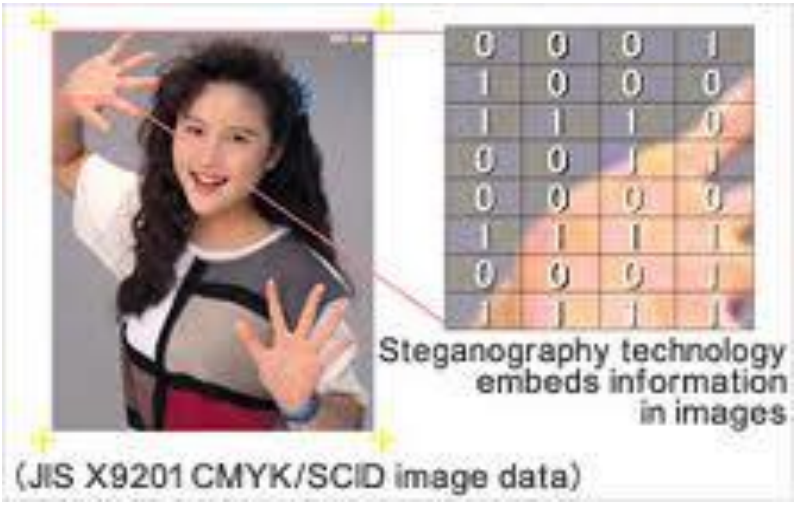
- Order of reading columns is alphabetical order of keyword
  - Keyword :- GERMAN.
  - PT : defend the east wall of the castle

G	E	R	M	A	N
d	e	f	e	n	d
t	h	e	e	a	s
t	w	a	l	l	o
f	t	h	e	c	a
s	t	l	e	x	x

  - CT :- nalcxehwttdttfseeleedsoax

2.4 STEGANOGRAPHY

- The technique of hiding message to keep it secret into some image is called as steganography.
- This technique provides secrecy.
- Special softwares are available to perform this process.
- It is a complicated process as it requires many steps to hide few bits of information into an image.
- If the process or image is discovered, then this method is useless.
- Example: people hide their secret messages with graphics images by replacing last two rightmost bits of each byte of the image with the two bits of the secret message.
- The resulting image will not look so different and will carry secret message.
- The receiver would perform the opposite by reading last two bits of each byte of that image file and reconstruct the secret message.



2.5 QUESTIONS

1. What are substitution techniques? Explain any one with the help of an example.
2. Explain Mono-alphabetic cipher with an example.
3. Explain Hill cipher with an example.
4. What is Polyalphabetic cipher? Explain with an example.
5. Explain Rail Fence cipher with an example.
6. Briefly define Caesar cipher with an example.
7. What are transposition techniques? Explain any one with the help of an example.
8. Define the following terms:- Brute force attack, cryptanalysis
9. Write a note on steganography.
10. Explain playfair cipher encrypt the plain text “SECRET MESSAGE” by using keyword “PROBLEM”.

REFERENCE FOR FURTHER READING

- Cryptography and Network Security, Atul Kahate, Tata McGraw-Hill, 2013.
- Cryptography and Network Security: Principles and Practice 5th Edition, William Stallings, Pearson,2010
- Syngress.The.Basics.of.Hacking.and.Penetration.Testing.Aug.2011
- Cryptography and Network, Behrouz A Fourouzan, Debdeep Mukhopadhyay, 2nd Edition,TMH,2011

❄❄❄❄❄❄❄

3

BLOCK CIPHER AND THE DES

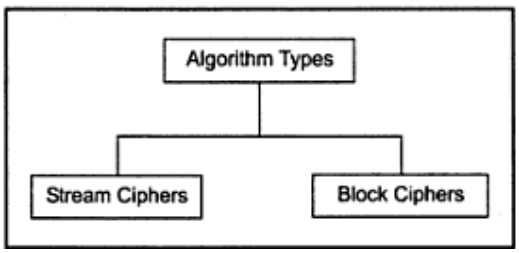
Unit Structure :

3.0 OBJECTIVES

- For cryptography various types of symmetric key and asymmetric key algorithms are used.
- These algorithms have two key aspects associated to them:
  - Algorithm Type: it defines what size of plaintext should be encrypted in each step of the algorithm.
  - Algorithm Mode: it defines the details of the cryptographic algorithm.

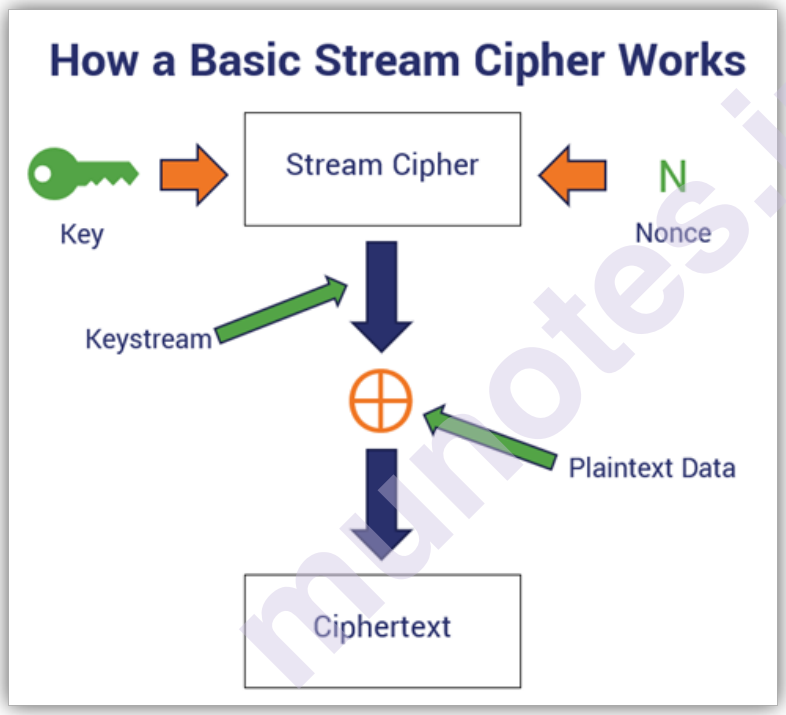
3.1 BLOCK CIPHER PRINCIPLES

- In general, if we want to encrypt any book and we convert pages topic wise or content wise then it is considered as block cipher whereas when we convert pages character by character by character it is considered as stream cipher.
- The conversion of plaintext to ciphertext is done in two basic ways based on which there are following two types of cryptographic algorithms:
  - Stream Ciphers
  - Block Ciphers



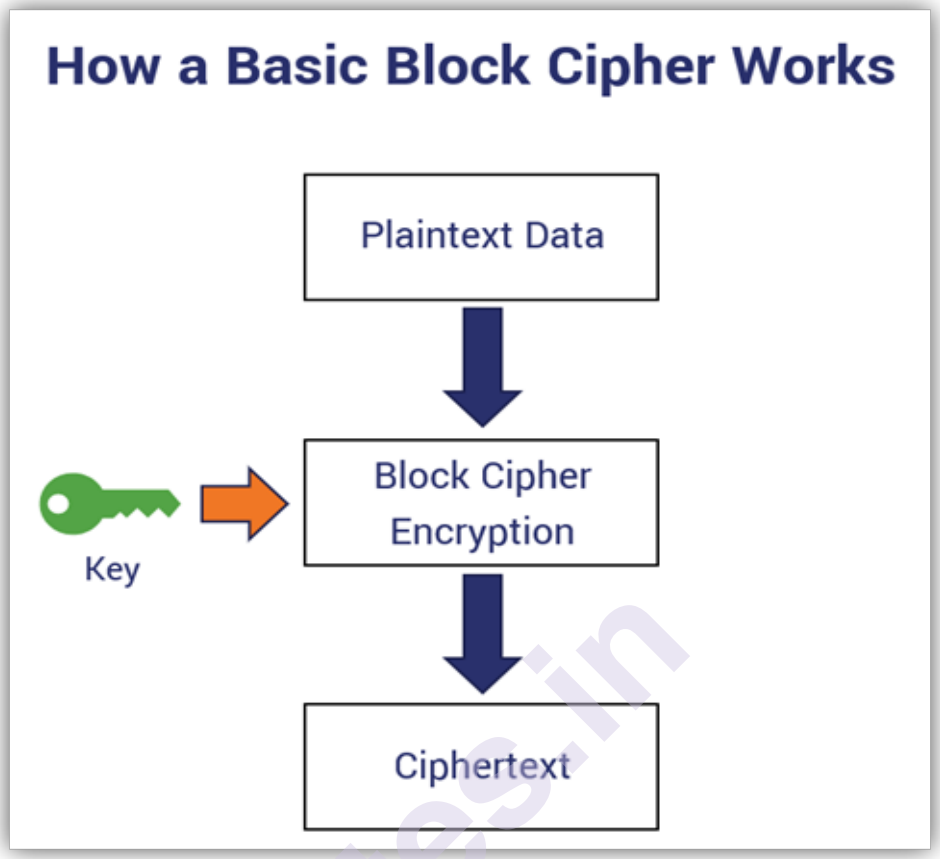
Stream Cipher

- Conversion of one bit or one byte at a time
- Example :- Vigenere cipher and Vernam cipher
- Conversion procedure of stream cipher should must implement bit stream generator.
- The main reason behind this is sender and receiver can produce cryptographic bit stream.
- Generator of bit stream is key management algorithm.
- It should produce a cryptographical strong bit stream.
- Now, participants of communication need to share generating key so that each one can produce keystream.



Block Cipher

- Plaintext is divided in to blocks and used to create cipher text of equal length.
- Example :- Network based symmetric cryptographic applications
- Blocks of data of predefined size is encrypted.
- To connect these blocks sequentially, blocks are then chained together.



Difference between Stream cipher and block cipher

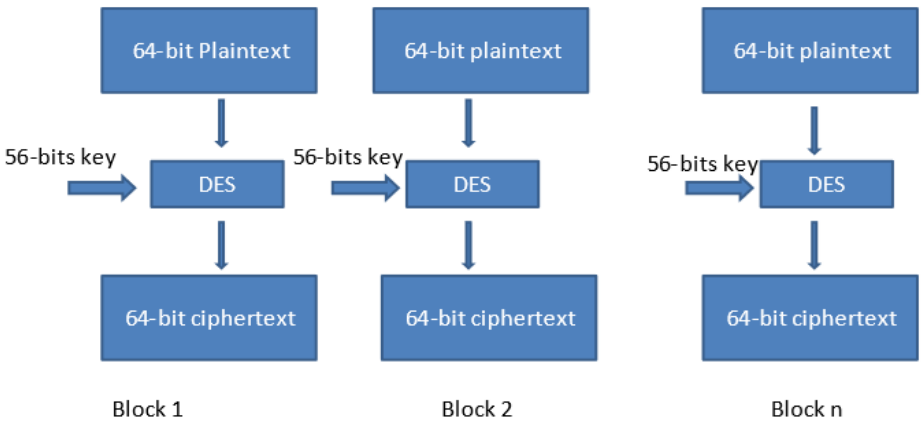
Block Cipher	**Stream Cipher**
Conversion of plain text into cipher text by taking block at a time	Conversion of plain text into cipher text by taking one byte at a time
Block size is 64 bits or more than that	Stream cipher uses 8 bits
No complexities	More complex
Uses confusion and diffusion	Uses only confusion
Reverse process is difficult	Reverse process is easy
Algorithm modes are :- ECB (Electronic Code Book), CBC (Cipher Block Chaining)	Algorithm modes are :- CFB (Cipher Feedback), OFB (Output Feedback)
Works on transposition techniques	Works on substitution techniques
Slow process	Fast process
Eg Feistel cipher	Eg. Vernam cipher

3.2 DES (DATA ENCRYPTION STANDARD)

- DES is also called as Data Encryption Algorithm.
- It is a cryptographic algorithm.
- This algorithm is used to provide security over three decades.
- DES is basically a landmark in cryptographic algorithm.
- DES is generally used in ECB (Electronic Code Book), CBC (Cipher Block Chaining), CFB (Cipher Feedback) mode.
- It was invented in 1972 in United States.
- This algorithm is used to give protection to systems and communication
- It was previously identified as Lucifer.

Working of DES

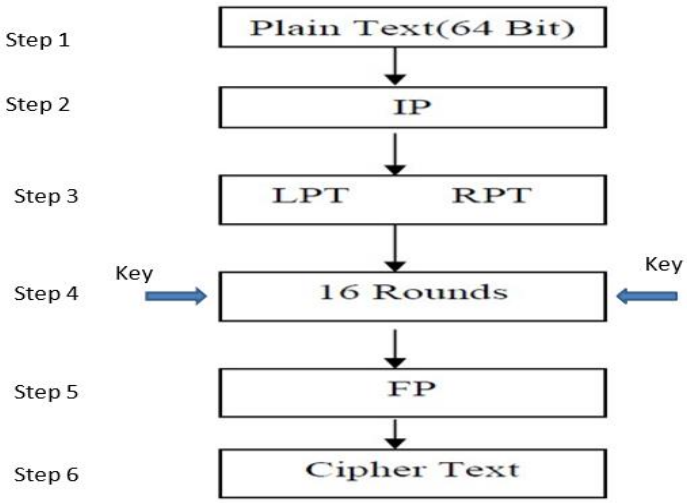
- DES is a block cipher.
- It encrypts input in block size of 64 bits each.
- That means, 64 bits of plain text is consider as input for DES and which produces 64 bits of output which is cipher text.
- The same algorithm and key are used for encryption and decryption with minor differences.
- Key size of DES is 56 bits.
- Actually, initial key size is 64 bits.
- Before starting DES process, every 8^{th} bit of the key is discarded to produce a 56 bits key.



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

Bit position 8,16,24,32,40,48,56 and 64 are discarded.

- DES algorithm is based on following two fundamental concepts of cryptography :-
  - Substitution (also called as confusion)
  - Transposition (also called as diffusion)
- There are 16 steps in DES.
- Each step is called round.
- Each round performs the steps of substitution and transposition.
- These steps are:
  - During the first step, 64 bit of plain text block is assigned to an initial permutation [IP] function.
  - The initial permutation is implemented on plain text.
  - Above process produces two parts of the permuted block:
- Left Plain Text (LPT)
- Right Plain Text (RPT)
  - Now each of LPT and RPT goes through 16 rounds of encryption process.
  - LPT and RPT are combined.
  - A final permutation [FP] is implemented on the combined block.
  - The output of this process produces 64 bits of cipher text.



- **Initial Permutation (IP):**
  - It occurs only once before the first round.
  - Following IP table shows, how the transposition in IP should proceed.

Bit position in the plain text box	To be overwritten with the content of this position
1	58
2	50
3	42
4	34
……..	……
64	7

Table after transposition is as follows

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

- After IP is completed, the resultant 64-bit permuted text block is separated into two half blocks.
- Each block consists of 32 bits.
- Left block is called LPT and right block is called RPT.
- 16 rounds are performed on these two blocks each.
- Each round consists of following steps:
  - Key transformation
  - Expansion permutation
  - S-box substitution
  - P-box permutation
  - XOR and Swap

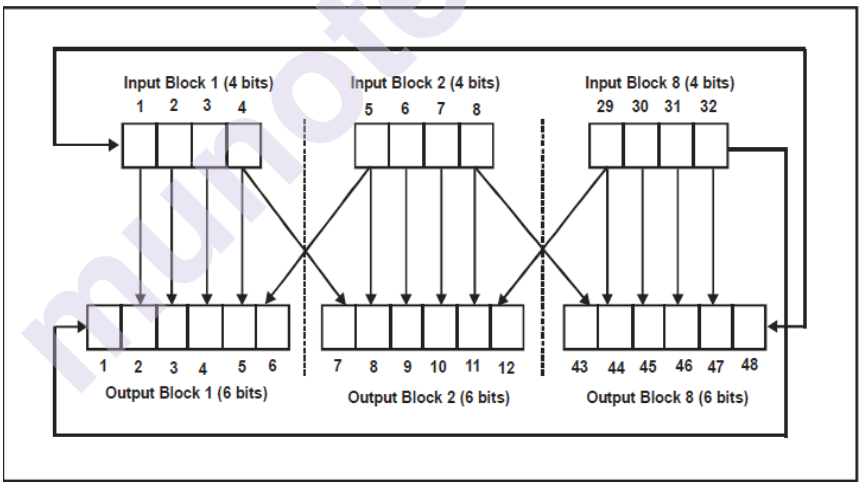
Step 1

- **Key Transformation:**
  - 56-bit key is available for each round.

- During each round, from this 56-bit key, a 48-bit subkey is created using a process called key transformation.
- 56-bit key is divided into two parts of 28-bits
- These parts are shifted towards left by one or two positions.
- For round number 1,2,9 or 16, shifting is done by one position and for rest of rounds shifting is done by two positions.
- After shifting random 48 bits are selected.
- This process is also known as compression permutation.

Step 2

- **Expansion permutation:**
  - 32-bit RPT is expanded to 48 bits as 48 bits key length is generated in step 1.
- 32-bit RPT is divided into 8 blocks of 4 bits each.
- Next each 4-bit block is expanded to 6-bit block, by adding 2 extra bits, that is the first bit of the block 1 is the last bit of the block 8 and the last bit of the block 8 is the first bit of the 7th block
- And thus 48-bit RPT is obtained.

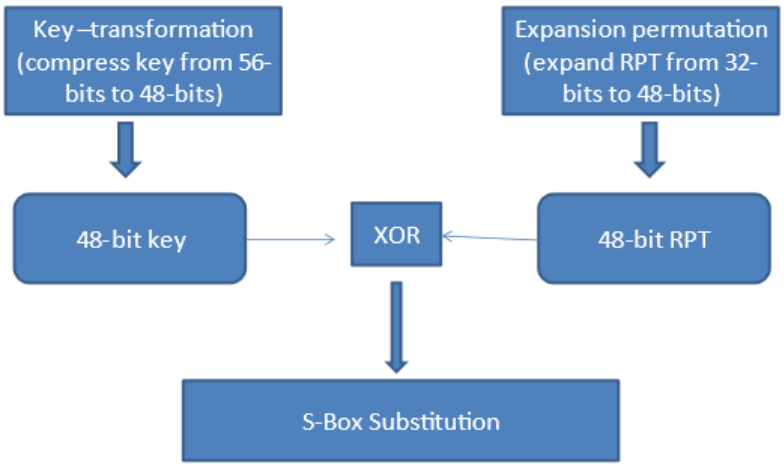


- Above process results into expansion as well as permutation of input bits while creating output bits.
- The expansion permutation is shown in the following table:

32	1	2	3	4	5	4	5	6	7	8	9
8	9	10	11	12	13	12	13	14	15	16	17
16	17	18	19	20	21	20	21	22	23	24	25
24	25	26	27	28	29	28	29	30	31	32	1

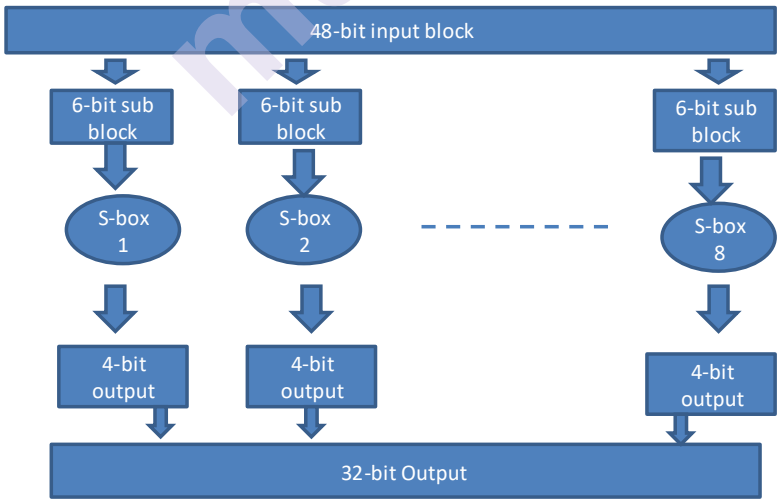
- 48-bit key is XORed with 48-bit RPT

– Result is considered for next step, which is S-box substitution.



Step 3

- **S-box Substitution:**
  - This step accepts 48 bits output from XOR operation of previous step and converts it into 32 bits using substitution technique.
  - Substitution process is performed by eight substitution boxes (**S-boxes**).
  - Each box takes 6 bits of inputs and produces 4 bits of output.
  - The 48-bit block text is then divided into 8 blocks of 6 bits each.
  - Each block is given to S-box.



  - For conversion, S-box tables are used.
  - There are 8 S-box tables for 8 blocks.
  - Each table has 4 (0-3) rows and 16 (0 -15) columns.

S_1

14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_2

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3

10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4

7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5

2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6

12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7

4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8

13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

- Output of each S-box is combined to form a 32-bit block
- This block is assigned to last stage of round, P-box Permutation.

Step 4

- **P-box Permutation:**
  - Output of S-Box which is 32 bits are permuted using a p-box.
  - This process involves simple permutation

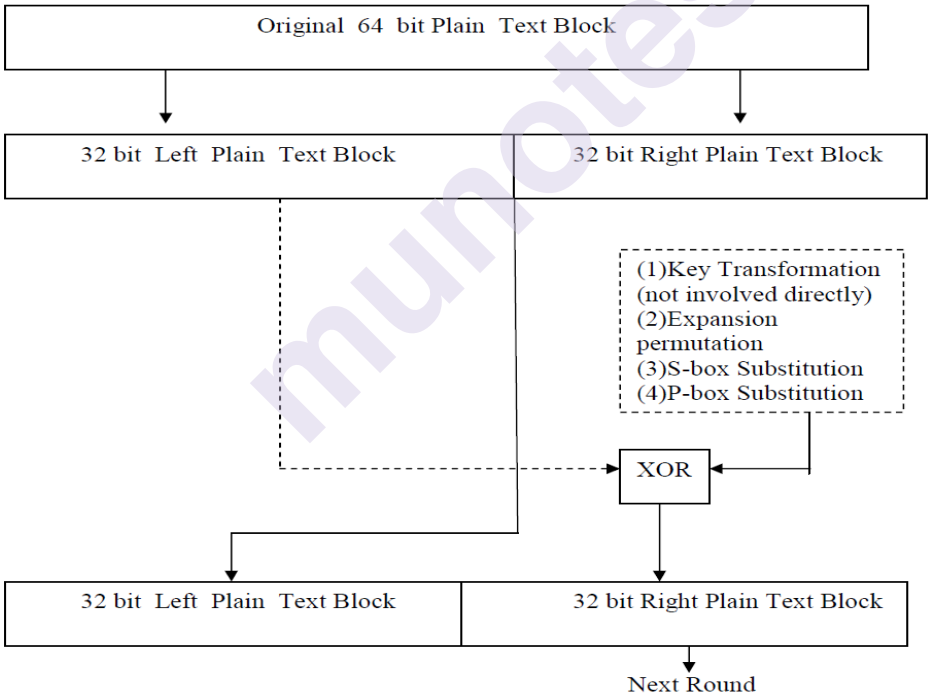
- Replacement of each bit with another bit as indicated in p-Box table without any expansion or compression.
- This is called as P-box Permutation.
- The P-Box is shown below.

16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

- For example, value 16 in first block specifies that bit at position 16 moves to bit at position 1 in the output.

Step 5

- **XOR and Swap:**
  - LPT, which is of 32 bits, which we have not processed at all.
  - LPT is XORed with output of RPT.
  - Result of this XOR operation becomes the new RPT.
  - The old value of RPT becomes new LPT in process of swapping.



- **Final Permutation:**
  - Finally, at the end of all 16 rounds, Final Permutation is performed only once.
  - A simple transposition is based on Final Permutation Table.
  - Output of the Final permutation is 64-bit encrypted block.
  - The below is table for final permutation values.

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

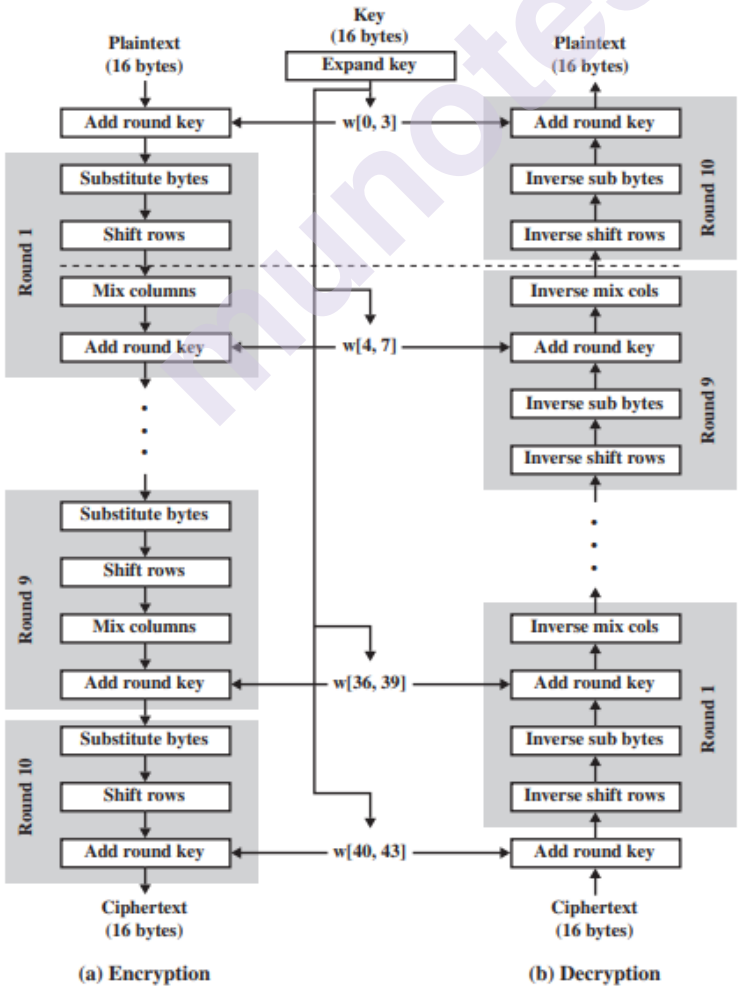
3.3 STRENGTH OF DES

- Strength of DES is based on following two factors :-
  - **Use of 56 bit key**
    - 56 bits key used in encryption
    - There are 256 such keys are possible
    - For these many number of keys brute force attack is impractical
  - **Nature of algorithm**
    - By exploiting characteristics of DES, cryptanalyst can perform cryptanalysis.

3.4 AES

- The more popular and widely accepted and adopted symmetric encryption algorithm nowadays is Advanced Encryption Standard (AES).
- At least six time faster than triple DES.
- Plain text block size is 128 bits or 16 bytes
- Key length can be 128, 192, 256 bits
- AES processes entire block of data as a single matrix in each round using substitution and transposition.
- The key which is provided as input is expanded into an array of forty four 32 bit words.
- Four distinct words are considered as a key for each round.
- This algorithm can be implemented on software and hardware both, so it seems to be the strongest security protocol.
- Its higher key length size makes it more tough against hacking.
- Its common application are e-commerce, wireless communication, financial transactions, secure data storage etc.
- Four different stages are used for AES; three for substitution and one for permutation.
- **Byte Substitution (SubBytes)**
  - 16 input bytes are replaced by looking into a S-box table. The result is in a matrix of four rows and four columns.

- **Shiftrows**
  - First row is not shifted.
  - Second row is shifted one (byte) position to the left.
  - Third row is shifted two positions to the left.
  - Fourth row is shifted three positions to the left.
  - The resultant new matrix consists of same 16 bytes but shifted with respect to each other.
- **MixColumns**
  - Each column of four bytes is transformed using a special mathematical function.
  - This function takes input as four bytes of one column and results four completely new bytes. The output is another new matrix consisting of 16 different bytes.
- **Addroundkey**
  - The 16 bytes of matrix are considered as 128 bits and are XORed with 128 bits of key.



3.5 MULTIPLE ENCRYPTION AND TRIPLE DES

Potential weakness of DES was brute force attack. So there was a need to design an algorithm which is completely new or make alterations in existing algorithm. Hence, Multiple encryption and triple DES arose.

Multiple encryption

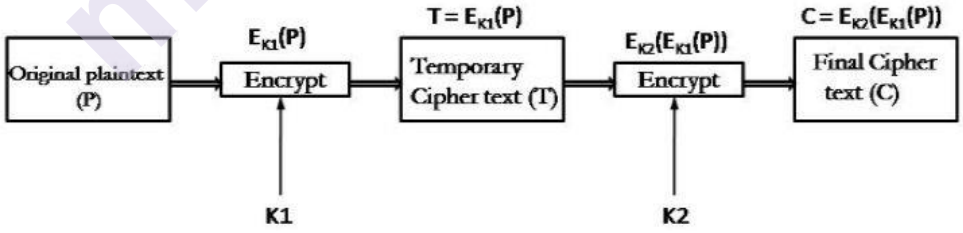
- An encryption algorithm is implemented many times.
- Plain text is converted to cipher text using any algorithm.
- Cipher text generate in above step is considered as input for the second step.
- Repeating above processes multiple times.

Multiple DES

- DES was prone to attacks due to advancement in computer hardware.
- Since DES was a very capable algorithm it would be better to reuse DES rather than writing a new cryptographic algorithm.
- Because of the above problems, variations of DES were introduced and are known as multiple DES.

Double DES

- Double DES makes use of two keys, let us assume K1 and K2.
- Firstly, it performs DES process on original plain text using key K1 and generate cipher text.
- DES process is repeated on cipher text generated in previous step with key K2.
- The final result is encryption of encrypted text with original plain text encrypted twice with two different keys shown in figure below.



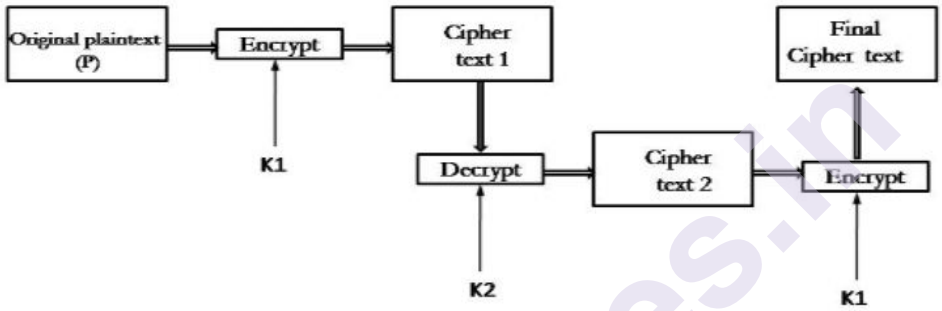
- Weakness of double DES is Meet-in-the-middle attack.
- This attack involves encryption from one side and decryption from the other side. Finally, matching results in middle somewhere.

Triple DES

- To enhance security of DES to an advance level triple DES was proposed.
- This uses three stages on DES for encryption and decryption.

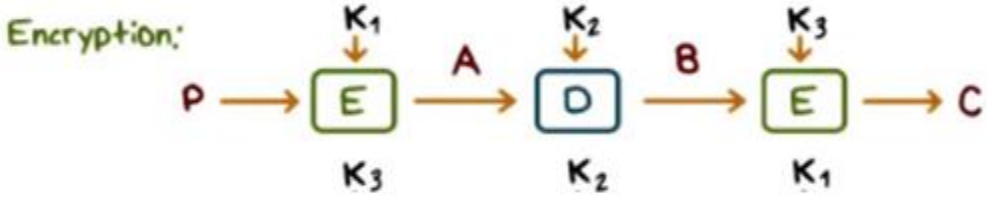
i. Triple DES with Two Keys-

- Triple DES with two keys uses two keys, let us assume K1 and K2.
- **Key K1** used in **first and third stage.**
- **Key K2** used in **second** stage.
- First **plain text** is **encrypted** using DES with key **K1** and generate cipher text. This cipher text is then **decrypted** with key **K2** and finally **output of second step** is **encrypted** again with key **K1**.
- Thus having EK1(DK2(EK1(P)))EK1(DK2(EK1(P))) shown below.
- This process is also called as ECE (Encrypt Decrypt Encrypt) mode.
- This process proves protection from man in the middle attack.



ii. Triple DES with Three Keys-

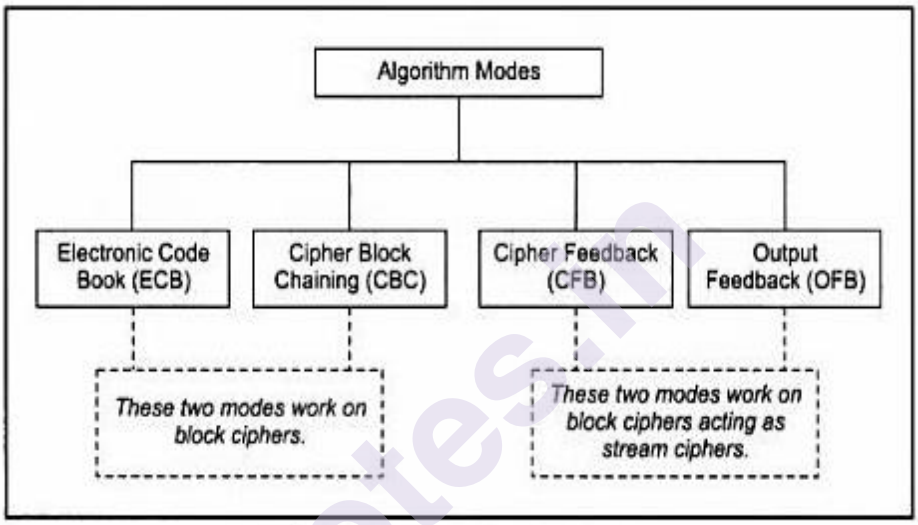
- Triple DES with three keys uses two keys, let us assume K1 K2 and K3.
- First plain text is encrypted using DES with key K1 and generate cipher text.
- This cipher text is then decrypted with key K2.
- Output of second step is encrypted again with key K3.
- Final Cipher text is the output of previous step.



3.6 BLOCK CIPHER MODES OF OPERATIONS

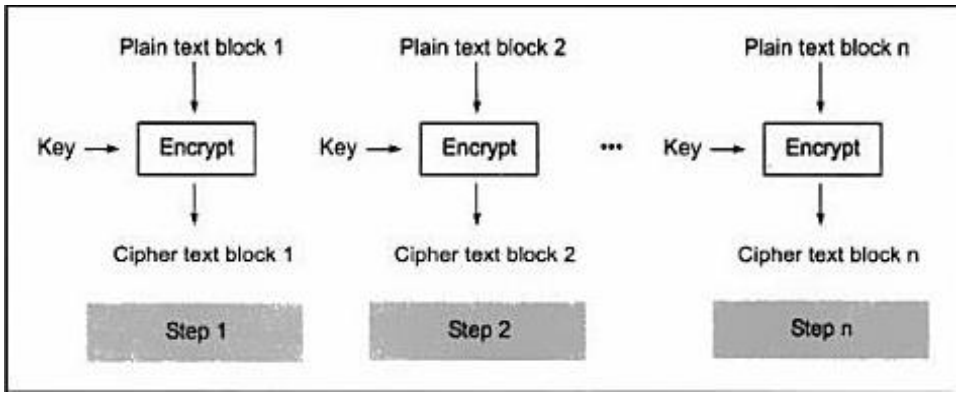
- A basic algorithm is designed to be effective but then also various cipher modes which is also known as algorithm modes are implemented to make the algorithms more capable and efficient in secreting the patterns.

- An algorithm mode is the permutation of series of the basic algorithm steps on block cipher and some feedback received from the previous step.
- There are 4 algorithm modes:
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)



- **Electronic Code Book (ECB)**
  - The simplest mode of operation used for data block.
  - The plaintext is divided into 64-bit block each.
  - Same key is used to encrypt each block.
  - Codebook means there is a unique cipher text for every n bit block of plain text for assigned key.
  - If the message is longer than b bits, group the message into b bit block, and use padding for the last block if needed.
  - Advantages
- Suitable for small amount of data
- Transmission of data over network like phone lines, then such transmission will affect into blocks.
- Faster and easy implementation
  - Disadvantages
- If same b bit block is repeating then it will result into the same cipher text.
- For lengthy message, ECB is not secure.

- No additional security measures are used so might be considered as weak

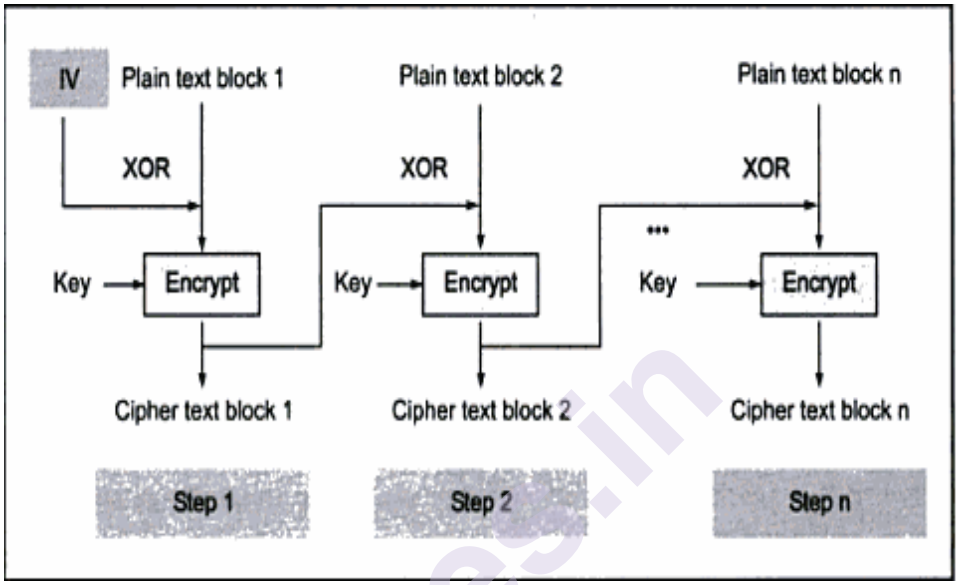


- **Cipher Block Chaining (CBC)**
  - To overcome the security issues with respect to ECB, this technique uses feedback mechanism.
  - Feedback method ensures if plain text block is repeating then also results into different cipher text block.
  - In this mode, before the encryption process starts, each plain text block is XORed with the previous cipher text block.
  - Each block is utilized to modify the encryption of the next block.
  - That means each block is rely on the corresponding current input plaintext block and also on the previous plaintext block.
  - The first step receives 2 inputs :-
- First block of plaintext
- Random block of text called Initialization vector (IV)
  - With the help of key, above two inputs are converted into cipher text block
  - Initialization vectors is a randomly produced set of characters used to make each message unique.
  - The second step makes use of second plaintext block XORed with the cipher text block generated in step 1 and uses the key for encryption to produce cipher text block 2.
  - The third step takes the third plaintext block XORed with the cipher text block of step 2 and uses the key for encryption to produce cipher text block 3.
  - This process lasts till all the blocks of plaintext are encrypted.
  - Advantages
    - Suitable for large amount of data
    - Maintains confidentiality

- Supports authentication
- Security enhancement as compared with ECB due to use of XOR and IV

o Disadvantages

- Not better choice for interactivity as used might need to wait for arrival and processing of 64 bits block



- **Cipher Feedback Mode (CFB)**

o This cipher mode is implemented for stream ciphers.

o Stream cipher removes the need of padding a message and also operates in real time.

o If stream is being transmitted then each character conversion can be done and use immediately for data transmission further.

o In this mode data is encrypted in units which are smaller like 8 bits rather than using a fixed size block of 64 bits.

o Used to encrypt any number of bits.

o In this mode, like CBC, 64-bit Initialization vector(IV) is used.

o Require use of Shift register is also.

o The following four steps are used for this cipher mode

o Assume that there are i-bits of plaintext taken at a time

- **Step 1**: Initially, shift register is filled with 64-bit initialization vector (IV), and encryption algorithm is executed once to create 64 bits IV ciphertext.
- **Step 2**: The left-most i-bits of encrypted IV is then XOR'ed with i-bits of plaintext which generates first portion of ciphertext (say C) and then C is transmitted to receiver.

- **Step 3:** Contents of shift register including IV are shifted left by i positions and empty rightmost i places of shift register are then filled with C.
- **Step 4:** Steps 1 to 3 are repeated until all the plaintext units are encrypted.

- Advantages
  - Size of plain text and cipher text is same.
  - More secure
- Disadvantages
  - Due to complexities and it is more time-consuming process

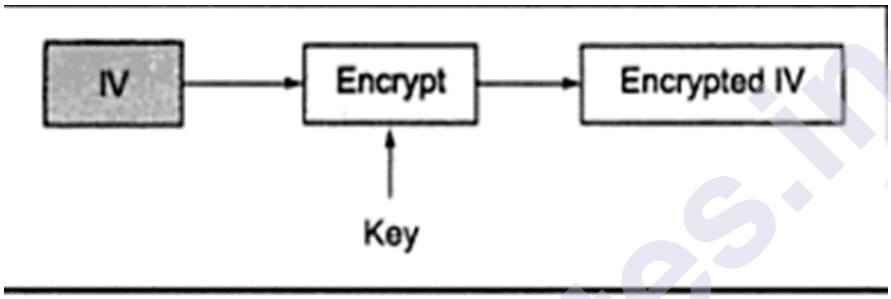


Fig. 3.10 *CFB – Step 1*

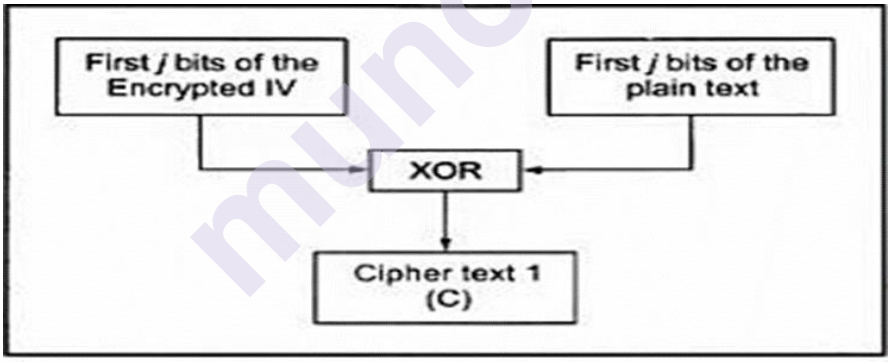


Fig. 3.11 *CFB – Step 2*

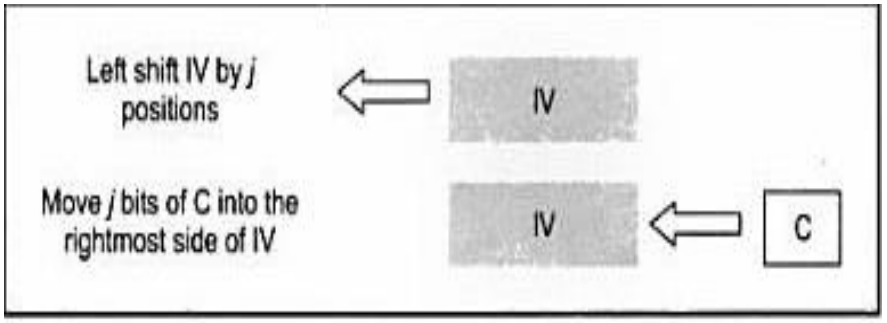


Fig. 3.12 *CFB – Step 3*

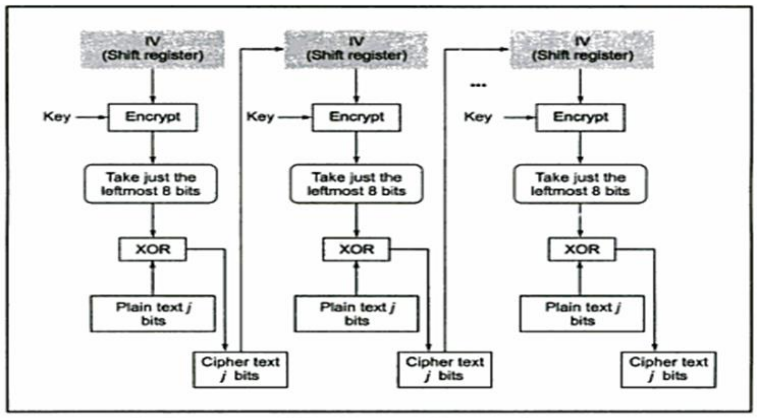


Fig. 3.13 *CFB – The overall encryption process*

- **Output Feedback Mode (OFB)**
  - Similar to CFB.
  - Use of an initialization vector (IV).
  - Encryption of varying block sizes process includes output of the IV is fed into the next step of encryption in place of the ciphertext.
  - The XOR (exclusive OR) value of each plaintext block is produced independently of both the plaintext and ciphertext.
  - Changing of IV in same plaintext block produces different ciphertext.
  - Advantages
    - Used when there is no tolerance for error propagation
    - No chaining dependencies
    - Bit errors in transmission do not propagate
  - Disadvantages
    - Vulnerable
    - Less secure

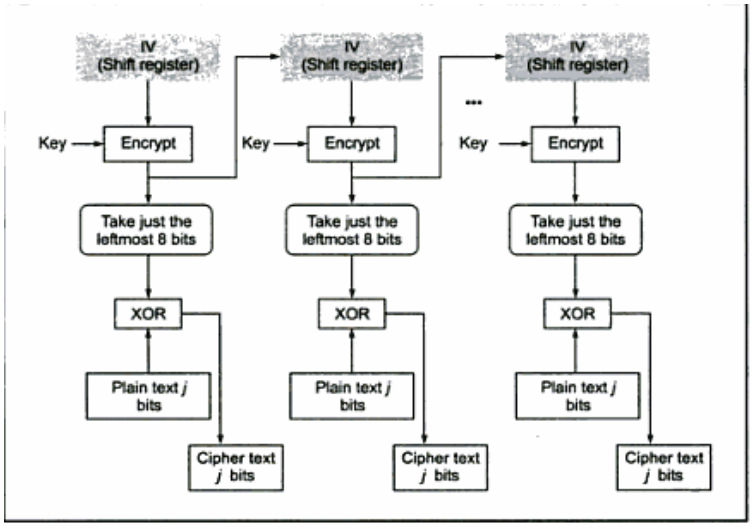


Fig. 3.14 *OFB – The overall encryption process*

3.7 STREAM CIPHERS

- Encrypts one byte of plain text at a time.
- A key is given as an input to pseudorandom bit generator which produces a stream of random numbers.
- Result of generator is called keystream which is then combined with plain text stream one byte at a time using XOR operation.

```
  11001100  plaintext
⊕ 01101100  key stream
  10100000  ciphertext
```

- Similar to OTP (one time pad) technique.
- Pseudorandom generator should produce a stream which repeats after longer period. The longer the period of repeat or longer the length of keyword, more time will be required for cryptanalysis.
- If keystream contains more random numbers, more randomized cipher text it will produce and as a result cryptanalysis will be difficult.
- To secure against Brute Force Attack, Key should be sufficient long enough minimum 128 bits is required.
- Faster as compared to block cipher.
- Less code is required as compared with block cipher.

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

Fig. 3.2 *Functioning of XOR logic*

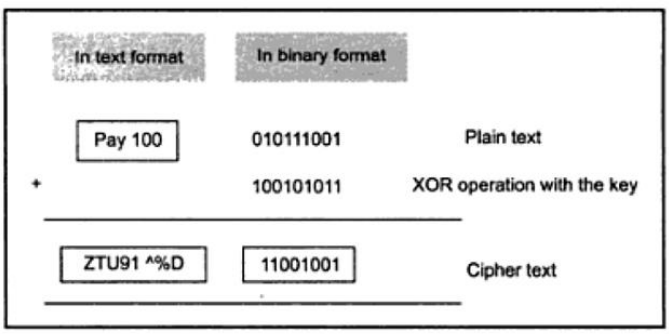


Fig. 3.3 *Stream ciphers*

3.8 QUESTIONS

1. Explain different types of algorithm modes in details.
2. Explain with neat diagram working of DES algorithm.
3. Explain the process involved in AES.
4. Explain multiple DES.

REFERENCE FOR FURTHER READING

- Cryptography and Network Security, Atul Kahate, Tata McGraw-Hill, 2013.
- Cryptography and Network Security: Principles and Practice 5th Edition, William Stallings, Pearson,2010
- Syngress.The.Basics.of.Hacking.and.Penetration.Testing.Aug.2011
- Cryptography and Network, Behrouz A Fourouzan, Debdeep Mukhopadhyay, 2nd Edition,TMH,2011

❄❄❄❄❄❄

4

PUBLIC KEY CRYPTOGRAPHY AND RSA

Unit Structure :

4.1 Principles of Public – key cryptosystems

4.2 The RSA Algorithm

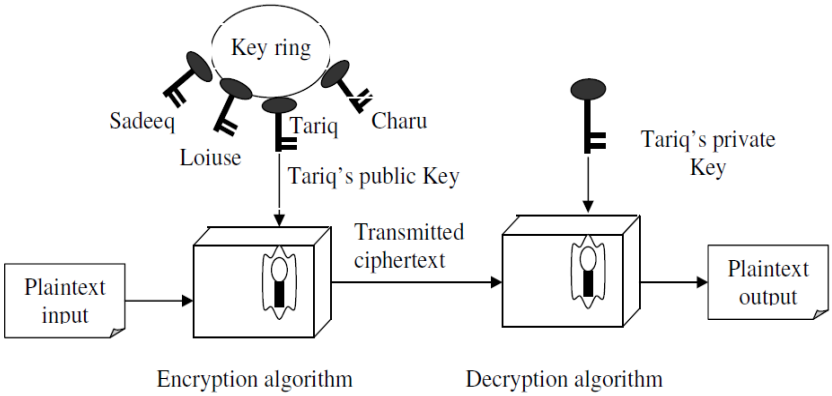
4.3 Questions and References

4.1 PRINCIPLES OF PUBLIC – KEY CRYPTOSYSTEMS

- Revolution of Public Key cryptography changed the history and pattern of cryptography.
- It is based on mathematical algorithms or formulas rather than using traditional methods of transposition and substitution.
- Also known as Asymmetric key cryptography.
- Use of two separate keys proved confidentiality and authentication
- Security of this cryptography depends on key length and computations involved in breaking a cipher.
- Drawback of symmetric Key cryptography is key distribution and mutual understanding for keys.

Public Key Cryptosystem

- Makes use of two different keys.
- One key is used for encryption and only the other corresponding key should be used for decryption.
- No other key can decrypt the message – not even the original (i.e. the first) key used for encryption.
- Every communicating party requires just a key pair for communicating with any number of other communicating parties.
- It should be impracticable to find decryption key provided with the knowledge of algorithm and encryption key.



Components of Public Key Cryptosystem

- **Plain Text**
  - A normal readable message which is considered as input for the process of encryption
- **Encryption Algorithm**
  - Sequence of steps which are used to convert plain text into cipher text
- **Public key and private key**
  - Key pair which is selected for asymmetric key cryptography.
  - One key is used in encryption process and other will be used for decryption process.
- **Cipher Text**
  - The resultant text which is produced after implementing encryption algorithm.
- **Decryption Algorithm**
  - Sequence of steps which are used to convert cipher text back into plain text with the help of matching key

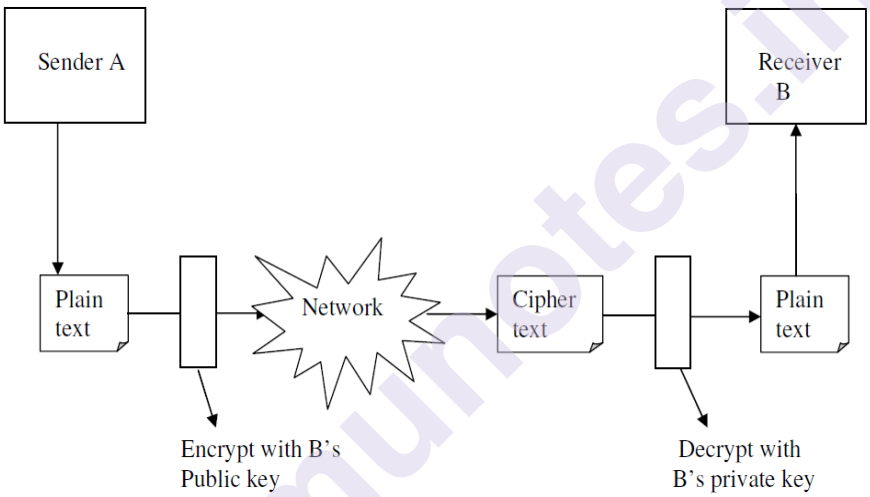
Steps involved in Asymmetric key cryptography

- Suppose user A wants to send a message to user B.
- Then A and B should each have a private key and a public key.
  - A should keep her private key secret.
  - B should keep her private key secret.
  - A should inform B about her public key.
  - B should inform A about her public key.

Key Details	**A** Should Know	**B** Should Know
A's Private key	Yes	No
A's Public Key	Yes	Yes
B's Private Key	No	Yes
B's Public Key	Yes	Yes

  - When A wishes to send a message to B, A encrypts the message using B's public key.
  - This is possible because A knows B's public key.
  - A sends the message encrypted with B's public key to B.

- When B receives the message, B decrypts A's message with B's private key.
- Only B knows her private key and the message could be decrypted only with B's private key.
- No one else or no other recipient would be able to make the sense of the message content even if one is able to intercept the message.
- All participants involved in communication gets access to public keys and private keys are created by each participant locally.
- Incoming communication is considered to be secure as long as private key is kept secure and protected.
- A system can change private key and corresponding public keys can be generated and distributed.



Difference between Symmetric key cryptography and Asymmetric key cryptography

Characteristic	Symmetric	Asymmetric
Key used for en/decryption	same for both	two keys used.One for Encryption and one for decryption.
Speed of en/decryption	Very Fast	Slower
Size of Resulting encrypted text	usually same or less than original	More than original
Key agreement/Exchange	Big Problem	No Problem
No. of keys required	Equal about the square of number of participants	Same as number of participants
Use	Mainly for encryption and Decryption, cannot be used For digital signatures.	Can be used for encryption and Decryption as well as Digital Signature

Applications

- Encryption / Decryption – Encryption of the message by the sender by receiver's public key
- Digital signature – Sender uses his private key to sign message. It proves integrity and confidentiality of the message.
- Key exchange – Session keys are exchanged to provide privacy.

Requirements

- It should be feasible for receiver to produce a pair of public key and private key.
- It should be feasible for sender by knowing public key and message to be encrypted to produce cipher text.
- It should be feasible for receiver B to decrypt cipher text using private key to recover plain text.
- It should be infeasible for others to find or know the private key by knowing public key.
- It should be infeasible for other to find plain text by knowing public key and cipher text.

4.2 THE RSA ALGORITHM

- Invented by Rivest, Shamir & Adleman of MIT in 1977.
- Widely used as public key cryptography.
- Use of two keys :- Public and private key
- One key which is known to be as public key is sent to other who are involved in communication and other key is kept secret.
- Based on block cipher.
- Plain text is encrypted in blocks.
- Public and private key are based on random prime numbers.
- Assume random prime numbers are P and Q.
- N = P * Q
- Sender and receiver should know the value of N.
- Public key which is known as encryption key E is a number which should not be a factor of (P – 1) (Q – 1).
- For encryption, Calculating Cipher Text (CT) with input of Plain Text (PT),

- $CT = PT^E \bmod N$
- For decryption, D is decryption key.
- $PT = CT^D \bmod N$
- It should be possible to find N, E, D for all plain text.
- It should be easy to calculate $PT^E \bmod N$ and $CT^D \bmod N$.
- It should be infeasible to calculate D, given values of E and N.

Example

- Select prime numbers.
- P = 7, Q = 17
- N = P * Q = 7 * 17 = 119
- (P – 1)(Q – 1) = 6 * 16 = 96
- 2, 3 and 4 are factors of 96.
- So, let us consider 5 as E.
- Assume, PT = 4.
- $CT = PT^E \bmod N = 4^5 \bmod 119 = 1024 \bmod 119 = 72$
- CT = 72

Security of RSA

Possible attacks on RSA are :-

- Brute Force
  - Trying all possible combinations to find keys
- Mathematical attack
  - Factoring products of two primes
  - If value of P and Q are known using N then its easy to find private key.
- Timing attack
  - Running time of decryption algorithm
- Chose cipher text attack
  - Exploits properties of RSA
- Short message attack
  - Attacker knows few blocks of plain text and tries to decrypt cipher text.
  - Padding of plain text before encryption to avoid such attacks

- Cycling attack
  - Plain text is converted into cipher text continuously by counting iterations till the occurrence of plain text.
- Unconcealed message attack
  - In few cases it is found that cipher text is same as plain text which means plain text is not hidden.

4.3 QUESTIONS

1. State and explain with example steps involved in RSA algorithm.
2. Using RSA algorithm, find the value of cipher text C, if the plaintext M = 5, p = 3, q = 11, d = 7.
3. What are the components of simple symmetric cipher model?
4. Differentiate between symmetric and asymmetric key cryptography.

REFERENCE FOR FURTHER READING

- Cryptography and Network Security, Atul Kahate, Tata McGraw-Hill, 2013.
- Cryptography and Network Security: Principles and Practice 5th Edition, William Stallings, Pearson,2010
- Syngress.The.Basics.of.Hacking.and.Penetration.Testing.Aug.2011
- Cryptography and Network, Behrouz A Fourouzan, Debdeep Mukhopadhyay, 2nd Edition,TMH,2011

❄❄❄❄❄❄❄

5

KEY MANAGEMENT

Unit Structure :

5.0 OBJECTIVE

Strong data encryption requires encryption key management. **Being able to own and manage your encryption keys is crucial to meet compliance standards and to satisfy regulatory sovereignty requirements.** Key management additionally ensures regulatory compliance and secures data from risks posed by privileged users. An effective key management solution also ensures that keys and their policies can be stored in an appliance that remains in full control of security teams, and not the storage administrators.

With the increasing dependence on cryptography to protect digital assets and communications, the ever-present vulnerabilities in modern computing systems, and the growing sophistication of cyber attacks, it has never been more important, nor more challenging, to keep your cryptographic keys safe and secure. A single compromised key could lead to a massive data breach with the consequential reputational damage, punitive regulatory fines and loss of investor and customer confidence.

5.1 INTRODUCTION

Cryptography lies at the heart of the modern business - protecting electronic communications and financial transactions, maintaining the privacy of sensitive data and enabling secure authentication and authorization. New regulations like GDPR and PSD2, the commercial pressure for digital transformation, the adoption of cloud technology and the latest trends in IoT and blockchain/DLT all help drive the need to embed cryptography into virtually every application – from toasters to core banking systems!

The good news is that modern cryptographic algorithms, when implemented correctly, are highly-resistant to attack – their only weak point is their keys. However, if a key is compromised, then it's game over! This makes such cryptographic keys one of your company's most precious assets, and they should be treated as such. The value of any key is equivalent to the value of all the data and/or assets it is used to protect.

5.2 TYPES OF KEYS

There are three primary types of keys that need to be kept safe and secure:

1. Symmetric keys – typically used to encrypt bulk data with symmetric algorithms like 3DES or AES; anyone with the secret key can decrypt the data

2. Private keys – the secret half of public/private key pairs used in public-key cryptography with asymmetric algorithms like RSA or ECDSA; anyone with the private key can impersonate the owner of the private key to decrypt private data, gain unauthorized access to

systems or generate a fraudulent digital signature that appears authentic

3. Hash keys – used to safeguard the integrity and authenticity of data and transactions with algorithms like HMAC-SHA256; anyone with the secret key can impersonate the originator of the data/transactions and thus modify the original data/transactions or create entirely false data/transactions that any recipient will believe is authentic

5.3 PUBLIC-KEY CRYPTOSYSTEMS

Public-key cryptography, or asymmetric cryptography, is an encryption scheme that uses two mathematically related, but not identical, keys - a public key and a private key. Unlike symmetric key algorithms that rely on one key to both encrypt and decrypt, each key performs a unique function. The public key is used to encrypt and the private key is used to decrypt.

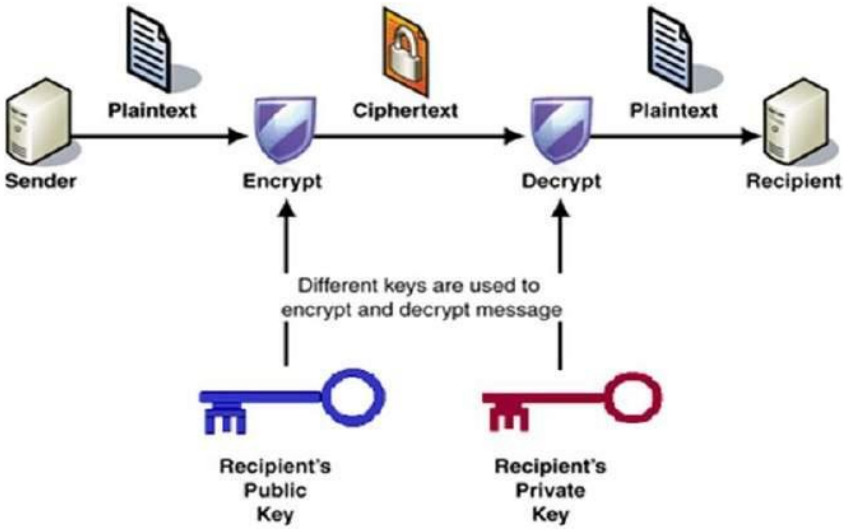
It is computationally infeasible to compute the private key based on the public key. Because of this, public keys can be freely shared, allowing users an easy and convenient method for encrypting content and verifying digital signatures, and private keys can be kept secret, ensuring only the owners of the private keys can decrypt content and create digital signatures.

Since public keys need to be shared but are too big to be easily remembered, they are stored on digital certificates for secure transport and sharing. Since private keys are not shared, they are simply stored in the software or operating system you use, or on hardware (e.g., USB token, hardware security module) containing drivers that allow it to be used with your software or operating system.

Symmetric cryptography was well suited for organizations such as governments, military, and big financial corporations were involved in the classified communication.

With the spread of more unsecure computer networks in last few decades, a genuine need was felt to use cryptography at larger scale. The symmetric key was found to be non-practical due to challenges it faced for key management. This gave rise to the public key cryptosystems.

The process of encryption and decryption is depicted in the following illustration −



The most important properties of public key encryption scheme are −

- Different keys are used for encryption and decryption. This is a property which set this scheme different than symmetric encryption scheme.
- Each receiver possesses a unique decryption key, generally referred to as his private key.
- Receiver needs to publish an encryption key, referred to as his public key.
- Some assurance of the authenticity of a public key is needed in this scheme to avoid spoofing by adversary as the receiver. Generally, this type of cryptosystem involves trusted third party which certifies that a particular public key belongs to a specific person or entity only.
- Encryption algorithm is complex enough to prohibit attacker from deducing the plaintext from the ciphertext and the encryption (public) key.
- Though private and public keys are related mathematically, it is not be feasible to calculate the private key from the public key. In fact, intelligent part of any public-key cryptosystem is in designing a relationship between two keys.

5.4 TYPES PUBLIC-KEY CRYPTOSYSTEMS

5.4.1 RSA algorithm (Rivest-Shamir-Adleman)

The RSA algorithm is the basis of a cryptosystem -- a suite of cryptographic algorithms that are used for specific security services or purposes -- which enables public key encryption and is widely used to secure sensitive data, particularly when it is being sent over an insecure network such as the internet.

RSA was first publicly described in 1977 by Ron Rivest, Adi Shamir and Leonard Adleman of the Massachusetts Institute of Technology, though the 1973 creation of a public key algorithm by British mathematician Clifford Cocks was kept classified by the U.K.'s GCHQ until 1997.

In RSA cryptography, both the public and the private keys can encrypt a message; the opposite key from the one used to encrypt a message is used to decrypt it. This attribute is one reason why RSA has become the most widely used asymmetric algorithm: It provides a method to assure the confidentiality, integrity, authenticity, and non-repudiation of electronic communications and data storage.

Many protocols like secure shell, OpenPGP, S/MIME, and SSL/TLS rely on RSA for encryption and digital signature functions. It is also used in software programs -- browsers are an obvious example, as they need to establish a secure connection over an insecure network, like the internet, or validate a digital signature. RSA signature verification is one of the most commonly performed operations in network-connected systems.

Algorithm

The RSA algorithm holds the following features −

- RSA algorithm is a popular exponentiation in a finite field over integers including prime numbers.
- The integers used by this method are sufficiently large making it difficult to solve.
- There are two sets of keys in this algorithm: private key and public key.

You will have to go through the following steps to work on RSA algorithm −

Step 1: Generate the RSA modulus

The initial procedure begins with selection of two prime numbers namely p and q, and then calculating their product N, as shown −

$N=p*q$

Here, let N be the specified large number.

Step 2: Derived Number (e)

Consider number e as a derived number which should be greater than 1 and less than (p-1) and (q-1). The primary condition will be that there should be no common factor of (p-1) and (q-1) except 1

Step 3: Public key

The specified pair of numbers n and e forms the RSA public key and it is made public.

Step 4: Private Key

Private Key d is calculated from the numbers p, q and e. The mathematical relationship between the numbers is as follows −

ed = 1 mod (p-1) (q-1)

The above formula is the basic formula for Extended Euclidean Algorithm, which takes p and q as the input parameters.

Encryption Formula

Consider a sender who sends the plain text message to someone whose public key is (n,e). To encrypt the plain text message in the given scenario, use the following syntax −

C = Pe mod n

Decryption Formula

The decryption process is very straightforward and includes analytics for calculation in a systematic approach. Considering receiver C has the private key d, the result modulus will be calculated as −

Plaintext = Cd mod n

Generating RSA keys

The following steps are involved in generating RSA keys −

- Create two large prime numbers namely p and q. The product of these numbers will be called n, where n= p*q
- Generate a random number which is relatively prime with (p-1) and (q-1). Let the number be called as e.
- Calculate the modular inverse of e. The calculated inverse will be called as d.

5.4.1.1 Algorithms for generating RSA keys

We need two primary algorithms for generating RSA keys using Python − Cryptomath module and Rabin Miller module.

1. Cryptomath Module

   The source code of cryptomath module which follows all the basic implementation of RSA algorithm is as follows −

```
def gcd(a, b):
  while a != 0:
    a, b = b % a, a
  return b
def findModInverse(a, m):
```

```
  if gcd(a, m) != 1:
    return None
  u1, u2, u3 = 1, 0, a
  v1, v2, v3 = 0, 1, m
  while v3 != 0:
    q = u3 // v3
      v1, v2, v3, u1, u2, u3 = (u1 - q * v1), (u2 - q * v2), (u3 - q *
v3), v1, v2, v3
  return u1 % m
```

2. RabinMiller Module

The source code of RabinMiller module which follows all the basic implementation of RSA algorithm is as follows −

```
import random
def rabinMiller(num):
  s = num - 1
  t = 0
  while s % 2 == 0:
    s = s // 2
    t += 1
  for trials in range(5):
    a = random.randrange(2, num - 1)
    v = pow(a, s, num)
    if v != 1:
      i = 0
      while v != (num - 1):
        if i == t - 1:
          return False
        else:
          i = i + 1
          v = (v ** 2) % num
    return True
def isPrime(num):
  if (num 7< 2):
    return False
```

```
  lowPrimes = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47,
53, 59, 61,
  67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137,
139, 149, 151,
  157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227,
229, 233, 239, 241,
  251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313,317,
331, 337, 347, 349,
  353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431,
433, 439, 443, 449,
  457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541,
547, 557, 563, 569,
  571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643,
647, 653, 659, 661,
  673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743, 751, 757,
761, 769, 773, 787,
  797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877,
881, 883, 887, 907,
  911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991, 997]
  if num in lowPrimes:
    return True
  for prime in lowPrimes:
    if (num % prime == 0):
      return False
  return rabinMiller(num)
def generateLargePrime(keysize = 1024):
  while True:
    num = random.randrange(2**(keysize-1), 2**(keysize))
    if isPrime(num):
      return num
```

The complete code for generating RSA keys is as follows −

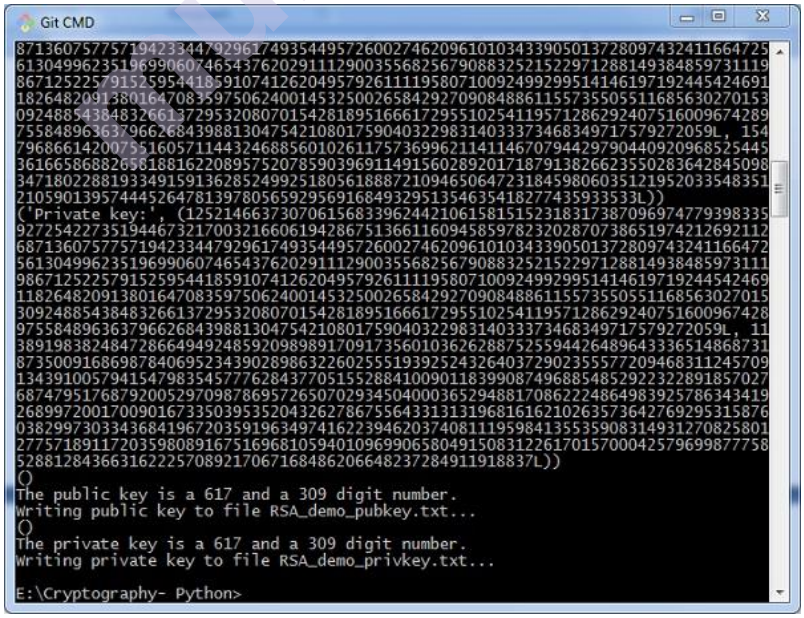
```
import random, sys, os, rabinMiller, cryptomath


def main():
  makeKeyFiles('RSA_demo', 1024)
```

```
def generateKey(keySize):
  # Step 1: Create two prime numbers, p and q. Calculate n = p * q.
  print('Generating p prime...')
  p = rabinMiller.generateLargePrime(keySize)
  print('Generating q prime...')
  q = rabinMiller.generateLargePrime(keySize)
  n = p * q
  # Step 2: Create a number e that is relatively prime to (p-1)*(q-1).
  print('Generating e that is relatively prime to (p-1)*(q-1)...')
  while True:
    e = random.randrange(2 ** (keySize - 1), 2 ** (keySize))
    if cryptomath.gcd(e, (p - 1) * (q - 1)) == 1:
      break
   # Step 3: Calculate d, the mod inverse of e.
  print('Calculating d that is mod inverse of e...')
  d = cryptomath.findModInverse(e, (p - 1) * (q - 1))
  publicKey = (n, e)
  privateKey = (n, d)
  print('Public key:', publicKey)
  print('Private key:', privateKey)
  return (publicKey, privateKey)
def makeKeyFiles(name, keySize):
  # Creates two files 'x_pubkey.txt' and 'x_privkey.txt'
    (where x is the value in name) with the the n,e and d,e integers
written in them,
  # delimited by a comma.
  if os.path.exists('%s_pubkey.txt' % (name)) or
os.path.exists('%s_privkey.txt' % (name)):
    sys.exit('WARNING: The file %s_pubkey.txt or %s_privkey.txt
already exists! Use a different name or delete these files and re-run
this program.' % (name, name))
  publicKey, privateKey = generateKey(keySize)
  print()
  print('The public key is a %s and a %s digit number.' %
(len(str(publicKey[0])), len(str(publicKey[1]))))
```

```
  print('Writing public key to file %s_pubkey.txt...' % (name))
  fo = open('%s_pubkey.txt' % (name), 'w')
fo.write('%s,%s,%s' % (keySize, publicKey[0], publicKey[1]))
fo.close()
  print()
  print('The private key is a %s and a %s digit number.' % (len(str(publicKey[0])), len(str(publicKey[1]))))
  print('Writing private key to file %s_privkey.txt...' % (name))
    fo = open('%s_privkey.txt' % (name), 'w')
  fo.write('%s,%s,%s' % (keySize, privateKey[0], privateKey[1]))
  fo.close()
# If makeRsaKeys.py is run (instead of imported as a module) call
# the main() function.
if __name__ == '__main__':
  main()
```

Output

The public key and private keys are generated and saved in the respective files as shown in the following output.



5.4.1.2 Example

An example of generating RSA Key pair is given below. (For ease of understanding, the primes p & q taken here are small values. Practically, these values are very high).

- Let two primes be p = 7 and q = 13. Thus, modulus n = pq = 7 x 13 = 91.
- Select e = 5, which is a valid choice since there is no number that is common factor of 5 and (p − 1)(q − 1) = 6 × 12 = 72, except for 1.
- The pair of numbers (n, e) = (91, 5) forms the public key and can be made available to anyone whom we wish to be able to send us encrypted messages.
- Input p = 7, q = 13, and e = 5 to the Extended Euclidean Algorithm. The output will be d = 29.
- Check that the d calculated is correct by computing −

  $de = 29 \times 5 = 145 = 1 \bmod 72$
- Hence, public key is (91, 5) and private keys is (91, 29).

5.4.1.3 RSA Encryption

- Suppose the sender wish to send some text message to someone whose public key is (n, e).
- The sender then represents the plaintext as a series of numbers less than n.
- To encrypt the first plaintext P, which is a number modulo n. The encryption process is simple mathematical step as −

  $C = P^e \bmod n$
- In other words, the ciphertext C is equal to the plaintext P multiplied by itself e times and then reduced modulo n. This means that C is also a number less than n.
- Returning to our Key Generation example with plaintext P = 10, we get ciphertext C −

  $C = 10^5 \bmod 91$

5.4.1.4 RSA Decryption

- The decryption process for RSA is also very straightforward. Suppose that the receiver of public-key pair (n, e) has received a ciphertext C.
- Receiver raises C to the power of his private key d. The result modulo n will be the plaintext P.

  $Plaintext = C^d \bmod n$
- Returning again to our numerical example, the ciphertext C = 82 would get decrypted to number 10 using private key 29 −

  $Plaintext = 82^{29} \bmod 91 = 10$

5.4.1.5 RSA Analysis

The security of RSA depends on the strengths of two separate functions. The RSA cryptosystem is most popular public-key cryptosystem strength of which is based on the practical difficulty of factoring the very large numbers.

- Encryption Function − It is considered as a one-way function of converting plaintext into ciphertext and it can be reversed only with the knowledge of private key d.
- Key Generation − The difficulty of determining a private key from an RSA public key is equivalent to factoring the modulus n. An attacker thus cannot use knowledge of an RSA public key to determine an RSA private key unless he can factor n. It is also a one way function, going from p & q values to modulus n is easy but reverse is not possible.

If either of these two functions are proved non one-way, then RSA will be broken. In fact, if a technique for factoring efficiently is developed then RSA will no longer be safe.

The strength of RSA encryption drastically goes down against attacks if the number p and q are not large primes and/ or chosen public key e is a small number.

5.4.1.6 RSA security

RSA security relies on the computational difficulty of factoring large integers. As computing power increases and more efficient factoring algorithms are discovered, the ability to factor larger and larger numbers also increases.

Encryption strength is directly tied to key size, and doubling key length can deliver an exponential increase in strength, although it does impair performance. RSA keys are typically 1024- or 2048-bits long, but experts believe that 1024-bit keys are no longer fully secure against all attacks. This is why the government and some industries are moving to a minimum key length of 2048-bits.

5.4.2 ElGamal Cryptosystem

The *ElGamal encryption system* is a public key encryption algorithm by Taher Elgamal in 1985 that is based on the Diffie-Hellman key exchange. We give an introduction to the ElGamal Encryption System and an example in the video

We assume that the message m that Alice encrypts and sends to Bob is an integer. We describe the three components of ElGamal encryption, namely key generation, encryption, and decryption.

Bob: Key Generation

To generate his private key and his public key Bob does the following.

1. Bob chooses a prime p and and a generator $g \in \mathbb{Z}_p^{\otimes}$.
2. Bob chooses a random .b ∈ N.
3. Bob computes $B = g^{b\otimes}$ in $(\mathbb{Z}_p^{\otimes}, \otimes)$.
4. Bob publishes his public key ,p,g, B in the key directory.

Alice: Encryption

To encrypt a message $m \in \mathbb{Z}_p^{\otimes}$ Alice does the following.

1. Alice gets Bob's public key ,p, ,g, B from the key directory.
2. Alice chooses a random .a∈N.
3. Alice computes the shared secret . $s = B^{a\otimes}$.
4. Alice computes . $A = g^{a\otimes}$.
5. Alice encrypts m by computing .X=m ⊗ s.
6. Alice sends (A,X) to Bob.

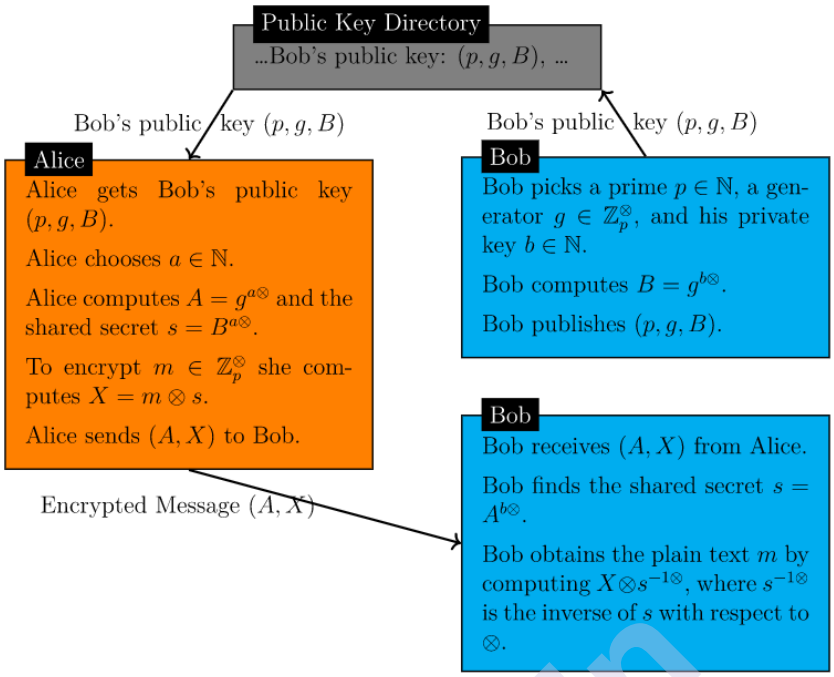
Bob: Decryption

The information available to Bob to decrypt a message are his private key b and his public key consisting of the prime ,p, the generator ,g, and .B=g^b. To decrypt a message (A,X) Bob does the following.

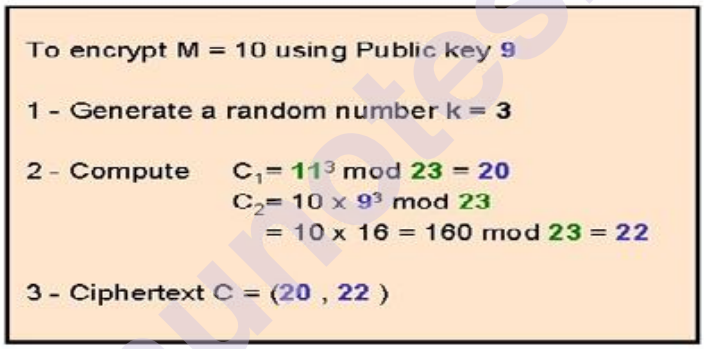
1. Bob receives (A,X) from Alice.
2. Bob computes the shared secret . $s = A^{b\otimes}$.
3. Bob computes the inverse $s^{-1\otimes}$ of s in $(\mathbb{Z}_p^{\otimes}, \otimes)$.
4. Bob decrypts the message by computing $M = X \otimes s^{-1\otimes}$.

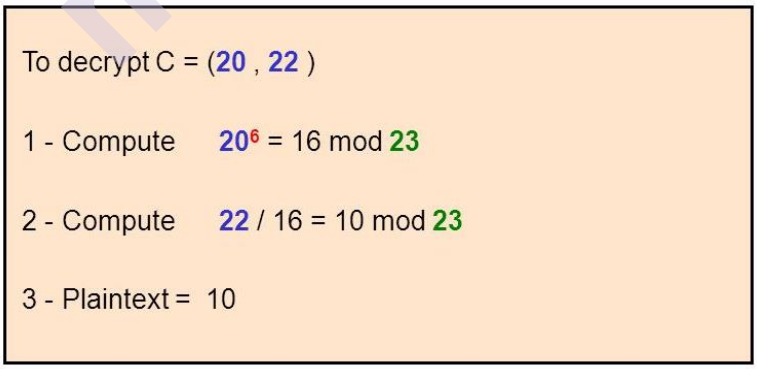
We now show that the message M received by Bob is equal to Alice's plain text message .m. We have

$$M = X \otimes s^{-1\otimes} = (m \otimes s) \otimes s^{-1\otimes} = m \otimes (s \otimes s^{-1\otimes}) = m \otimes 1 = m.$$



We work through a small example.





5.4.2.1 ElGamal Analysis

In ElGamal system, each user has a private key x. and has three components of public key − prime modulus p, generator g, and public $Y = g^x \bmod p$. The strength of the ElGamal is based on the difficulty of discrete logarithm problem.

The secure key size is generally > 1024 bits. Today even 2048 bits long key are used. On the processing speed front, Elgamal is quite slow, it is used mainly for key authentication protocols. Due to higher processing efficiency, Elliptic Curve variants of ElGamal are becoming increasingly popular.

5.4.3 Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography (ECC) is a term used to describe a suite of cryptographic tools and protocols whose security is based on special versions of the discrete logarithm problem. It does not use numbers modulo p.

ECC is based on sets of numbers that are associated with mathematical objects called elliptic curves. There are rules for adding and computing multiples of these numbers, just as there are for numbers modulo p.

ECC includes a variants of many cryptographic schemes that were initially designed for modular numbers such as ElGamal encryption and Digital Signature Algorithm.

It is believed that the discrete logarithm problem is much harder when applied to points on an elliptic curve. This prompts switching from numbers modulo p to points on an elliptic curve. Also an equivalent security level can be obtained with shorter keys if we use elliptic curve-based variants.

The shorter keys result in two benefits −

- Ease of key management
- Efficient computation

These benefits make elliptic-curve-based variants of encryption scheme highly attractive for application where computing resources are constrained.

5.4.3.1 RSA and ElGamal Schemes – A Comparison

Let us briefly compare the RSA and ElGamal schemes on the various aspects.

RSA	ElGamal
It is more efficient for encryption.	It is more efficient for decryption.
It is less efficient for decryption.	It is more efficient for decryption.
For a particular security level, lengthy keys are required in RSA.	For the same level of security, very short keys are required.
It is widely accepted and used.	It is new and not very popular in market.

5.5 KEY MANAGEMENT

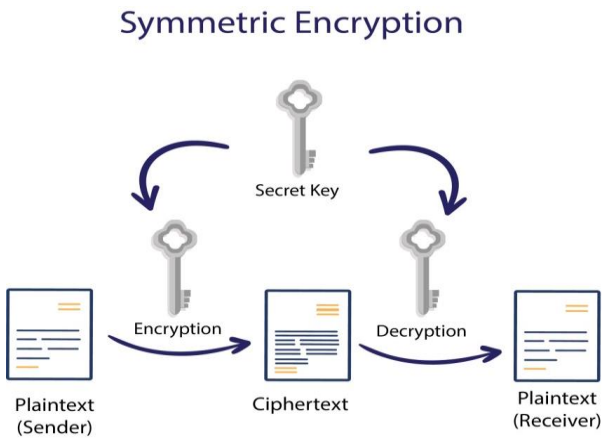
Cryptographic keys are a vital part of any security system. They do everything from data encryption and decryption to user authentication. The compromise of any cryptographic key could lead to the collapse of an organization's entire security infrastructure, allowing the attacker to decrypt sensitive data, authenticate themselves as privileged users, or give themselves access to other sources of classified information. Luckily, proper management of keys and their related components can ensure the safety of confidential information. Key Management is the process of putting certain standards in place to ensure the security of cryptographic keys in an organization. Key Management deal with the creation, exchange, storage, deletion, and refreshing of keys. They also deal with the members access of the keys.

5.5.1 Why is Key Management Important

Key management forms the basis of all data security. Data is encrypted and decrypted via the use of encryption keys, which means the loss or compromise of any encryption key would invalidate the data security measures put into place. Keys also ensure the safe transmission of data across an Internet connection. With authentication methods, like code signing, attackers could pretend to be a trusted service like Microsoft, while giving victim's computers malware, if they steal a poorly protected key. Keys provide compliance with certain standards and regulations to ensure companies are using best practices when protecting cryptographic keys. Well protected keys are only accessible by users who need them.

5.5.2 Types of Keys

There are two types of cryptographic keys, symmetric and asymmetric keys. Symmetric keys deal with data-at-rest, which is data stored in a static location, such as a database. Symmetric key encryption uses the same key for both encryption and decryption. Using data in a database as an example, while the data is stored in the database, it is encrypted with the symmetric key. Once an authorized user attempts to access the data, the information is decrypted with the same symmetric key and made accessible to the user. The other type of cryptographic key is an asymmetric key.



Encryption using asymmetric keys is a little more complicated than symmetric key encryption. Instead of using the same key for both encryption and decryption, two separate keys called a public and private key, are used for the encryption and decryption of data. These keys are created as a pair, so that they relate to each other. The public key of a pair of asymmetric keys is mainly used to encrypt data. This key can be shared with anyone since it encrypts, not decrypts, data. The private key is used for the decryption of data encrypted by its public key counterpart, so it must stay secure.

Asymmetric keys focus on encrypting data-in-motion. Data-in-motion is data sent across a network connection, whether it be a public or private connection. When transporting sensitive data, most encryption processes use both symmetric and asymmetric keys to encrypt data.

- The data is first encrypted-at-rest by a symmetric encryption key.
- The symmetric key is now encrypted by the public key of the person who the data is being sent to. That encrypted symmetric key and the ciphertext are sent to the recipient of the data.
- Once the ciphertext and key reach the recipient, the symmetric key is decrypted by that user's private key, and the ciphertext is decrypted.

5.5.3 How Key Management Works

Key management follows a lifecycle of operations which are needed to ensure the key is created, stored, used, and rotated securely. Most cryptographic keys follow a lifecycle which involves key

- Generation
- Distribution
- Use
- Storage
- Rotation
- Backup/Recovery
- Revocation
- Destruction

The generation of a key is the first step in ensuring that key is secure. If the key in question is generated with a weak encryption algorithm, then any attacker could easily discover the value of the encryption key. Also, if the key is generated in an insecure location, the key could be compromised as soon as it is created, resulting in a key that cannot be

safely used for encryption. Key generators, AES encryption algorithms, or random number generators tend to be used for secure key generation.

The next step of the key lifecycle is ensuring the safe distribution of the keys. Keys should be distributed to the required user via a secure TLS or SSL connection, to maintain the security of the keys being distributed. If an insecure connection is used to distribute the cryptographic keys, then the security of any data encrypted by these keys is in question, as an attacker could execute a man-in-the-middle attack and steal the keys.

After distribution of the key, it is used for cryptographic operations. As previously noted, the key should only be used by authorized users, to make certain the key is not misused, copied, etc. When the key is used to encrypt data, it must then be stored for later decryption. The most secure method is via a Hardware Security Module (HSM) or CloudHSM. If an HSM is not used, then the keys can either be securely stored on the client's side, or, if the keys are used on the Cloud, then the Cloud Service Provider's Key Management Service can be used.

Once a key's cryptoperiod, or time period the key is usable, passes, the key must be rotated. When the key of an encrypted set of data expires, the key is retired and replaced with a new key. First the data is decrypted by the old key or key pair and then encrypted by the new key or key pair. Rotation is necessary because the longer a key is in rotation, the more chance there is for someone to steal or find out the key. Rotation of keys can happen before the cryptoperiod expires in cases where the key is suspected to be compromised.

Two other ways of dealing with a compromised key are revoking or destroying the key in question. Revoking a key means the key can no longer be used to encrypt or decrypt data, even if its cryptoperiod is still valid. Destroying a key, whether that is due to compromise or due to it no longer being used, deletes the key permanently from any key manager database or other storage method. This makes it impossible to recreate the key, unless a backup image is used. NIST standards require that deactivated keys be kept in an archive, to allow for reconstruction of the keys if data encrypted in the past must now be decrypted by that key or key pair.

5.6 DIFFIE-HELLMAN KEY EXCHANGE

The Diffie-Hellman key exchange was **one of the most important developments in public-key cryptography** and it is still frequently implemented in a range of today's different security protocols.

It allows two parties who have not previously met to securely establish a key which they can use to secure their communications. In this article, we'll explain what it's used for, how it works on a step-by-step basis, its different variations, as well as the security considerations that need to be noted in order to implement it safely.

The Diffie-Hellman key exchange was the **first widely used method of safely developing and exchanging keys over an insecure channel**.

It may not seem so exciting or groundbreaking in the above terms, so let's give an example that explains why the Diffie-Hellman key exchange was such an important milestone in the world of cryptography, and why it is still so frequently used today.

Diffie Hellman Key Exchange Algorithm-

Let-

- Private key of the sender = X_s
- Public key of the sender = Y_s
- Private key of the receiver = X_r
- Public key of the receiver = Y_r

Using Diffie Hellman Algorithm, the key is exchanged in the following steps-

Step-01:

One of the parties choose two numbers 'a' and 'n' and exchange with the other party.

- 'a' is the primitive root of prime number 'n'.
- After this exchange, both the parties know the value of 'a' and 'n'.

Step-02:

- Both the parties already know their own private key.
- Both the parties calculate the value of their public key and exchange with each other.
- Sender calculate its public key as-
- $Y_s = a^{X_s} \bmod n$
- Receiver calculate its public key as-
- $Y_r = a^{X_r} \bmod n$

Step-03:

- Both the parties receive public key of each other.
- Now, both the parties calculate the value of secret key.
- Sender calculates secret key as-
- Secret key = $(Y_r)^{X_s} \bmod n$

- Receiver calculates secret key as-
- Secret key = $(Y_s)^{X_r} \bmod n$

Finally, both the parties obtain the same value of secret key.

5.6.1 PRACTICE PROBLEMS BASED ON DIFFIE HELLMAN KEY EXCHANGE-

Problem-01:

Suppose that two parties A and B wish to set up a common secret key (D-H key) between themselves using the Diffie Hellman key exchange technique. They agree on 7 as the modulus and 3 as the primitive root. Party A chooses 2 and party B chooses 5 as their respective secrets. Their D-H key is-

3

4

5

6

Solution-

Given-

- n = 7
- a = 3
- Private key of A = 2
- Private key of B = 5

Step-01:

Both the parties calculate the value of their public key and exchange with each other.

Public key of A

$= 3^{\text{private key of A}} \bmod 7$

$= 3^2 \bmod 7$

$= 2$

Public key of B

$= 3^{\text{private key of B}} \bmod 7$

$= 3^5 \bmod 7$

$= 5$

Step-02:

Both the parties calculate the value of secret key at their respective side.

Secret key obtained by A

$= 5^{\text{private key of A}} \bmod 7$

$= 5^2 \bmod 7$

$= 4$

Secret key obtained by B

$= 2^{\text{private key of B}} \bmod 7$

$= 2^5 \bmod 7$

$= 4$

Finally, both the parties obtain the same value of secret key.

The value of common secret key = 4.

Thus, Option (B) is correct.

Problem-02:

In a Diffie-Hellman Key Exchange, Alice and Bob have chosen prime value q = 17 and primitive root = 5. If Alice's secret key is 4 and Bob's secret key is 6, what is the secret key they exchanged?

16

17

18

19

Solution-

Given-

- n = 17
- a = 5
- Private key of Alice = 4
- Private key of Bob = 6

Step-01:

Both Alice and Bob calculate the value of their public key and exchange with each other.

Public key of Alice

$= 5^{\text{private key of Alice}} \bmod 17$

$= 5^4 \bmod 17$

$= 13$

Public key of Bob

$= 5^{\text{private key of Bob}} \bmod 17$

$= 5^6 \bmod 17$

$= 2$

Step-02:

Both the parties calculate the value of secret key at their respective side.

Secret key obtained by Alice

$= 2^{\text{private key of Alice}} \bmod 7$

$= 2^4 \bmod 17$

$= 16$

Secret key obtained by Bob

$= 13^{\text{private key of Bob}} \bmod 7$

$= 13^6 \bmod 17$

$= 16$

Finally, both the parties obtain the same value of secret key.

The value of common secret key = 16.

5.6.2 Establishing a shared key between multiple parties

The Diffie-Hellman key exchange can also be used to set up a shared key with a greater number of participants. It works in the same manner, except further rounds of the calculations are needed for each party to add in their secret number and end up with the same shared secret.

Just like in the two-party version of the Diffie-Hellman key exchange, some parts of the information are sent across insecure channels, but not enough for an attacker to be able to compute the shared secret.

5.6.3 Why is the Diffie-Hellman key exchange secure?

On a mathematical level, the Diffie-Hellman key exchange relies on one-way functions as the basis for its security. These are calculations which are simple to do one way, but much more difficult to calculate in reverse.

More specifically, it relies on the Diffie-Hellman problem, which assumes that under the right parameters, it is infeasible to calculate *gab* from the

separate values of *g*, *ga* and *gb*. There is currently no publicly known way to easily find *gab* from the other values, which is why the Diffie-Hellman key exchange is considered secure, despite the fact that attackers can intercept the values *p*, *g*, *A*, and *B*.

5.6.4 Authentication & the Diffie-Hellman key exchange

In the real world, the Diffie-Hellman key exchange is rarely used by itself. The main reason behind this is that **it provides no authentication, which leaves users vulnerable to man-in-the-middle attacks**.

These attacks can take place when the Diffie-Hellman key exchange is implemented by itself, because **it has no means of verifying whether the other party in a connection is really who they say they are**. Without any form of authentication, **users may actually be connecting with attackers** when they think they are communicating with a trusted party.

For this reason, the Diffie-Hellman key exchange is generally implemented alongside some means of authentication. This often involves using digital certificates and a public-key algorithm, such as RSA, to verify the identity of each party.

5.6.5 Variations of the Diffie-Hellman key exchange

The Diffie-Hellman key exchange can be implemented in a number of different ways, and it has also provided the basis for several other algorithms. Some of these implementations provide authorization, while others have various cryptographic features such as perfect forward secrecy.

5.7 SUMMARY

- Cryptography lies at the heart of the modern business - protecting electronic communications and financial transactions, maintaining the privacy of sensitive data and enabling secure authentication and authorization.
- There are three primary types of keys - Symmetric keys, Private keys, Hash keys
- **Types Public-Key Cryptosystems- RSA algorithm (Rivest-Shamir-Adleman), ElGamal Cryptosystem, Elliptic Curve Cryptography (ECC)**
- Cryptographic keys are a vital part of any security system. They do everything from data encryption and decryption to user authentication.
- Key management forms the basis of all data security. Data is encrypted and decrypted via the use of encryption keys, which means the loss or compromise of any encryption key would invalidate the data security measures put into place.

- The Diffie-Hellman key exchange was **one of the most important developments in public-key cryptography** and it is still frequently implemented in a range of today's different security protocols.

5.8 QUESTIONS

1. Explain Public Key Cryptography.
2. Explain importance of Public Key.
3. Differentiate between Public Key Cryptography and Private Key Cryptography.
4. Write a short not on Key Management.
5. Explain Types Public-Key Cryptosystems.
6. How RSA algorithm works?
7. How ElGamal algorithm works?
8. Compare RSA and ElGamal.
9. Why is Key Management Important?

5.9 REFERENCE FOR FURTHER READING

- https://dis-blog.thalesgroup.com/security/2018/09/26/the-importance-of-key-management-when-implementing-a-secure-information-gateway/
- https://www.cryptomathic.com/news-events/blog/cryptographic-key-management-the-risks-and-mitigations
- https://www.globalsign.com/en/ssl-information-center/what-is-public-key-cryptography
- https://searchsecurity.techtarget.com/definition/RSA
- https://www.tutorialspoint.com/cryptography/public_key_encryption.htm
- https://mathstats.uncg.edu/sites/pauli/112/HTML/secelgamal.html
- https://www.encryptionconsulting.com/education-center/what-is-key-management/
- https://www.comparitech.com/blog/information-security/diffie-hellman-key-exchange/

❄❄❄❄❄❄❄

6

MESSAGE AUTHENTICATION AND HASH FUNCTIONS

Unit Structure :

6.0 OBJECTIVE

In this chapter we will discuss about Message Authentication, importance of Message Authentication. Types of Authentication and finally Hash Function.

In today's world everywhere there are net of information and now only data transfer is not important rather it must be **"Data transfer with security!!!".** Hence data must be AUTHENTICATED before it is being used by user. Here we will discuss all techniques.

6.1 INTRODUCTION

In most people's minds, privacy is the goal most strongly associated to cryptography. But message authentication is arguably even more important. Indeed you may or may not care if some particular message you send out stays private, but you almost certainly do want to be sure of the originator of each message that you act on. Message authentication is what buys you that guarantee.

Message authentication allows one party—the Sender—to send a message to another party—the Receiver—in such a way that if the message is modified en route, then the Receiver will almost certainly detect this. Message authentication is also called "data-origin authentication," since it authenticates the point-of-origin for each message. Message authentication is said to protect the "integrity" of messages, ensuring that each that is received and deemed acceptable is arriving in the same condition that it was sent out—with no bits inserted, missing, or modified.

6.2 AUTHENTICATION REQUIREMENTS

In the context of communications across a network, the following attacks can be identified:

1. **Disclosure**: Release of message contents to any person or process not possessing the appropriate cryptographic key.

2. **Traffic analysis**: Discovery of the pattern of traffic between parties. In a connection oriented application, the frequency and duration of connections could be determined. In either a connection-oriented or connectionless environment, the number and length of messages between parties could be determined.

3. **Masquerade**: Insertion of messages into the network from a fraudulent source. This includes the creation of messages by an opponent that are purported to come from an authorized entity. Also included are fraudulent acknowledgments of message receipt or non-receipt by someone other than the message recipient.

4. **Content Modification**: Changes to the contents of a message, including insertion, deletion, transposition, or modification.

5. **Sequence modification**: Any modification to a sequence of messages between parties, including insertion, deletion, and reordering.

6. **Timing modification**: Delay or replay of messages. In a connection-orientated application, an entire session or sequence of messages could be a replay of some previous valid session, or individual messages in the sequence could be delayed or replayed.

7. **Repudiation**: Denial of receipt of message by destination or denial of transmission of message by source.

Message authentication is a procedure to verify that received messages come from the alleged source and have not been altered. Message authentication may also verify sequencing and timeliness. A digital signature is an authentication technique that also includes measures to counter repudiation by either source or destination. Any message authentication or digital signature mechanism can be viewed as having fundamentally two levels. At the lower level, there must be some sort of function that produces an authenticator: a value to be used to authenticate

a message. This lowerlevel function is then used as primitive in a higher-level authentication protocol that enables a receiver to verify the authenticity of a message.

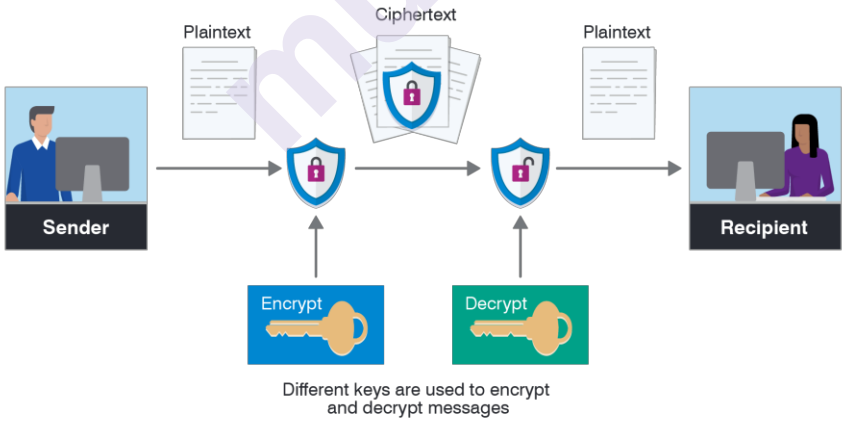
6.3 AUTHENTICATION FUNCTIONS

There are three types of functions that may be used to produce an authenticator:

1. Message Encryption
2. Message authentication code (MAC)
3. A Hash function

6.3.1 Message Encryption

Encryption is a means of securing digital data using one or more mathematical techniques, along with a password or "key" used to decrypt the information. The encryption process translates information using an algorithm that makes the original information unreadable. The process, for instance, can convert an original text, known as plaintext, into an alternative form known as ciphertext. When an authorized user needs to read the data, they may decrypt the data using a binary key. This will convert ciphertext back to plaintext so that the authorized user can access the original information. Encryption is an important way for individuals and companies to protect sensitive information from hacking. For example, websites that transmit credit card and bank account numbers should always encrypt this information to prevent identity theft and fraud. The mathematical study and application of encryption is known as cryptography.



There are two types of encryptions schemes as listed below:

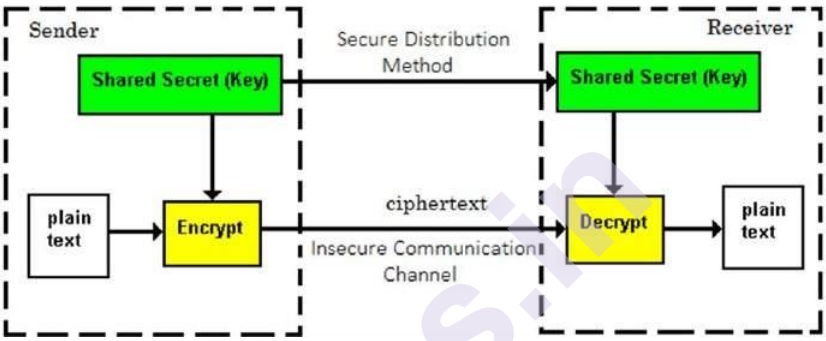
- Symmetric Key encryption
- Public Key encryption

1. Symmetric Key encryption

The encryption process where **same keys are used for encrypting and decrypting** the information is known as Symmetric Key Encryption.

The study of symmetric cryptosystems is referred to as **symmetric cryptography**. Symmetric cryptosystems are also sometimes referred to as **secret key cryptosystems**.

A few well-known examples of symmetric key encryption methods are − Digital Encryption Standard (DES), Triple-DES (3DES), IDEA, and BLOWFISH.



Prior to 1970, all cryptosystems employed symmetric key encryption. Even today, its relevance is very high and it is being used extensively in many cryptosystems. It is very unlikely that this encryption will fade away, as it has certain advantages over asymmetric key encryption.

The salient features of cryptosystem based on symmetric key encryption are −

- Persons using symmetric key encryption must share a common key prior to exchange of information.
- Keys are recommended to be changed regularly to prevent any attack on the system.
- A robust mechanism needs to exist to exchange the key between the communicating parties. As keys are required to be changed regularly, this mechanism becomes expensive and cumbersome.
- In a group of **n** people, to enable two-party communication between any two persons, the number of keys required for group is **n × (n – 1)/2**.
- Length of Key (number of bits) in this encryption is smaller and hence, process of encryption-decryption is faster than asymmetric key encryption.
- Processing power of computer system required to run symmetric algorithm is less.

Challenge of Symmetric Key Cryptosystem

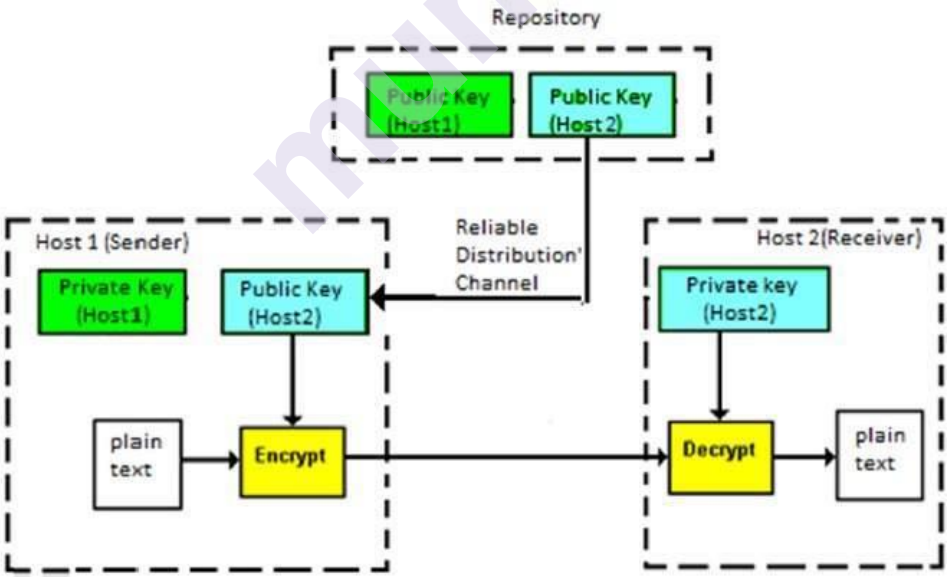
There are two restrictive challenges of employing symmetric key cryptography.

- **Key establishment** − Before any communication, both the sender and the receiver need to agree on a secret symmetric key. It requires a secure key establishment mechanism in place.
- **Trust Issue** − Since the sender and the receiver use the same symmetric key, there is an implicit requirement that the sender and the receiver 'trust' each other. For example, it may happen that the receiver has lost the key to an attacker and the sender is not informed.

These two challenges are highly restraining for modern day communication. Today, people need to exchange information with non-familiar and non-trusted parties. For example, a communication between online seller and customer. These limitations of symmetric key encryption gave rise to asymmetric key encryption schemes.

2. Public Key encryption / Asymmetric Key Encryption

The encryption process where **different keys are used for encrypting and decrypting the information** is known as Asymmetric Key Encryption. Though the keys are different, they are mathematically related and hence, retrieving the plaintext by decrypting ciphertext is feasible. The process is depicted in the following illustration −



Asymmetric Key Encryption was invented in the 20th century to come over the necessity of pre-shared secret key between communicating persons. The salient features of this encryption scheme are as follows −

- Every user in this system needs to have a pair of dissimilar keys, **private key** and **public key**. These keys are mathematically related − when one key is used for encryption, the other can decrypt the ciphertext back to the original plaintext.
- It requires to put the public key in public repository and the private key as a well-guarded secret. Hence, this scheme of encryption is also called **Public Key Encryption**.
- Though public and private keys of the user are related, it is computationally not feasible to find one from another. This is a strength of this scheme.
- When *Host1* needs to send data to *Host2,* he obtains the public key of *Host2* from repository, encrypts the data, and transmits.
- *Host2* uses his private key to extract the plaintext.
- Length of Keys (number of bits) in this encryption is large and hence, the process of encryption-decryption is slower than symmetric key encryption.
- Processing power of computer system required to run asymmetric algorithm is higher.

  Symmetric cryptosystems are a natural concept. In contrast, public-key cryptosystems are quite difficult to comprehend.

  You may think, *how can the encryption key and the decryption key are 'related', and yet it is impossible to determine the decryption key from the encryption key?* The answer lies in the mathematical concepts. It is possible to design a cryptosystem whose keys have this property. The concept of public-key cryptography is relatively new. There are fewer public-key algorithms known than symmetric algorithms.

Challenge of Public Key Cryptosystem

Public-key cryptosystems have one significant challenge − the user needs to trust that the public key that he is using in communications with a person really is the public key of that person and has not been spoofed by a malicious third party.

This is usually accomplished through a Public Key Infrastructure (PKI) consisting a trusted third party. The third party securely manages and attests to the authenticity of public keys. When the third party is requested to provide the public key for any communicating person X, they are trusted to provide the correct public key.

The third party satisfies itself about user identity by the process of attestation, notarization, or some other process − that X is the one and only, or globally unique, X. The most common method of making the

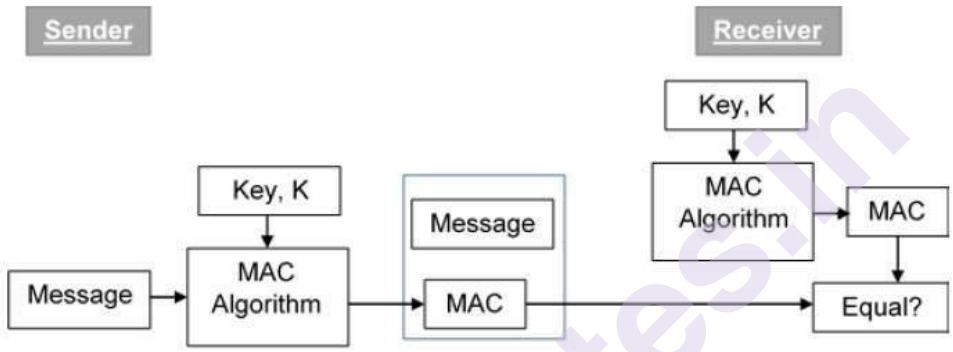
verified public keys available is to embed them in a certificate which is digitally signed by the trusted third party.

6.3.2 Message authentication code (MAC)

MAC algorithm is a symmetric key cryptographic technique to provide message authentication. For establishing MAC process, the sender and receiver share a symmetric key K.

Essentially, a MAC is an encrypted checksum generated on the underlying message that is sent along with a message to ensure message authentication.

The process of using MAC for authentication is depicted in the following illustration −



Let us now try to understand the entire process in detail −

- The sender uses some publicly known MAC algorithm, inputs the message and the secret key K and produces a MAC value.
- Similar to hash, MAC function also compresses an arbitrary long input into a fixed length output. The major difference between hash and MAC is that MAC uses secret key during the compression.
- The sender forwards the message along with the MAC. Here, we assume that the message is sent in the clear, as we are concerned of providing message origin authentication, not confidentiality. If confidentiality is required then the message needs encryption.
- On receipt of the message and the MAC, the receiver feeds the received message and the shared secret key K into the MAC algorithm and re-computes the MAC value.
- The receiver now checks equality of freshly computed MAC with the MAC received from the sender. If they match, then the receiver accepts the message and assures himself that the message has been sent by the intended sender.
- If the computed MAC does not match the MAC sent by the sender, the receiver cannot determine whether it is the message that has been

altered or it is the origin that has been falsified. As a bottom-line, a receiver safely assumes that the message is not the genuine.

Limitations of MAC

There are two major limitations of MAC, both due to its symmetric nature of operation −

- **Establishment of Shared Secret.**
    - It can provide message authentication among pre-decided legitimate users who have shared key.
    - This requires establishment of shared secret prior to use of MAC.
- **Inability to Provide Non-Repudiation**
    - Non-repudiation is the assurance that a message originator cannot deny any previously sent messages and commitments or actions.
    - MAC technique does not provide a non-repudiation service. If the sender and receiver get involved in a dispute over message origination, MACs cannot provide a proof that a message was indeed sent by the sender.
    - Though no third party can compute the MAC, still sender could deny having sent the message and claim that the receiver forged it, as it is impossible to determine which of the two parties computed the MAC.

6.3.3 A Hash function

A hash function is a serious mathematical process that holds a critical role in public key cryptography. Why? Because it's what helps you to:

- Securely store passwords in a database
- Ensure data integrity (in a lot of different applications) by indicating when data has been altered,
- Make secure authentication possible, andOrganize content and files in a way that increases efficiency.

A hash function is a unique identifier for any given piece of content. It's also a process that takes plaintext data of any size and converts it into a unique ciphertext of a specific length.

A basic illustration of how the hashing process works. A simple illustration of what a hash function does by taking a plaintext data input and using a mathematical algorithm to generate an unreadable output.

Properties of a Strong Hash Algorithm

So, what makes for a strong hashing algorithm? There are a few key traits that all good ones share:

- **Determinism** — A hash algorithm should be **deterministic**, meaning that it always gives you an output of identical size regardless of the size of the input you started with. This means that if you're hashing a single sentence, the resulting output should be the same size as one you'd get when hashing an entire book.

- **Pre-Image Resistance** — The idea here is that a strong hash algorithm is one that's preimage resistance, meaning that it's infeasible to reverse a hash value to recover the original input plaintext message. Hence, the concept of hashes being irreversible, one-way functions.

- **Collision Resistance** — A collision occurs when two objects collide. Well, this concept carries over in cryptography with hash values. If two unique samples of input data result in identical outputs, it's known as a collision. This is bad news and means that the algorithm you're using to hash the data is broken and, therefore, insecure. Basically, the concern here is that someone could create a malicious file with an artificial hash value that matches a genuine (safe) file and pass it off as the real thing because the signature would match. So, a good and trustworthy hashing algorithm is one that is resistant to these collisions.

- **Avalanche Effect** — What this means is that any change made to an input, no matter how small, will result in a massive change in the output. Essentially, a small change (such as adding a comma) snowballs into something much larger, hence the term "avalanche effect."

- **Hash Speed** — Hash algorithms should operate at a reasonable speed. In many situations, hashing algorithms should compute hash values quickly; this is considered an ideal property of a cryptographic hash function. However, this property is a little more subjective. You see, faster isn't always better because the speed should depend on how the hashing algorithm is going to be used. Sometimes, you want a faster hashing algorithm, and other times it's better to use a slower one that takes more time to run through. The former is better for website connections and the latter is better for password hashing.

What Does a Hash Function Do?

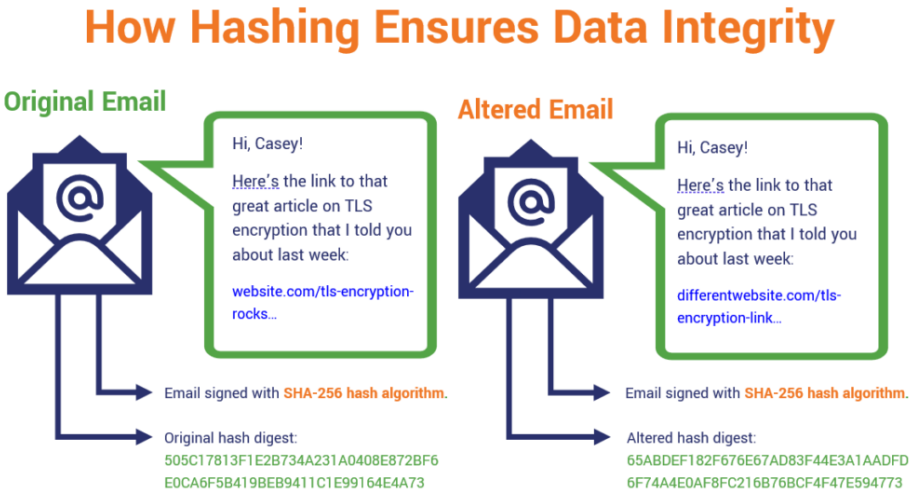
One purpose of a hash function in cryptography is to take a plaintext input and generate a hashed value output of a specific size in a way that can't be reversed. But they do more than that from a 10,000-foot perspective. You see, hash functions tend to wear a few hats in the world of cryptography. In a nutshell, strong hash functions:

- Ensure data integrity,
- Secure against unauthorized modifications,
- Protect stored passwords, and
- Operate at different speeds to suit different purposes.

1. Ensure Data Integrity

Hash functions are a way to ensure data integrity in public key cryptography. What I mean by that is that hash functions serve as a check-sum, or a way for someone to identify whether data has been tampered with after it's been signed. It also serves as a means of identity verification.

For example, let's say you've logged on to public Wi-Fi to send me an email. (Don't do that, by the way. It's very insecure.) So, you write out the message, sign it using your digital certificate, and send it on its way across the internet. This is what you might call prime man-in-the-middle attack territory — meaning that someone could easily intercept your message (again, because public wireless networks are notoriously insecure) and modify it to suit their evil purposes.



The example above is of a digitally signed email that's been manipulated in transit via a MitM attack. The hash digest changes completely when any of the email content gets modified after being digitally signed, signaling that it can't be trusted.

2. Secure Against Unauthorized Modifications

One of the best aspects of a cryptographic hash function is that it helps you to ensure data integrity. But if you apply a hash to data, does it mean that the message can't be altered? No. But what it does is inform the message recipient that the message has been changed. That's because even the smallest of changes to a message will result in the creation of an entirely new hash value.

Think of hashing kind of like you would a smoke alarm. While a smoke alarm doesn't stop a fire from starting, it does let you know that there's danger before it's too late.

3. Enable You to Verify and Securely Store Passwords

Nowadays, many websites allow you to store your passwords so you don't have to remember them every time you want to log in. But storing plaintext passwords like that in a public-facing server would be dangerous because it leaves that information vulnerable to cybercriminals. So, what websites typically do is hash passwords to generate hash values, which is what they store instead.

But password hashes on their own isn't enough to protect you against certain types of attacks, including brute force attacks. This is why you first need to add a salt. A **salt** is a unique, random number that's applied to plaintext passwords before they're hashed. This provides an additional layer of security and can protect passwords from password cracking methods like rainbow table attacks.

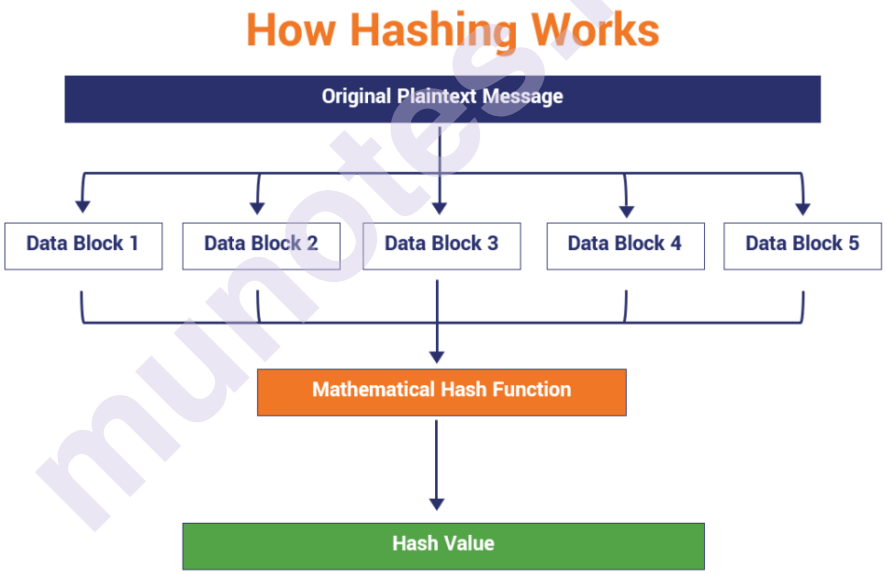
4. Operate at Different Speeds, Suiting Different Purposes

It's also important to note that hash functions aren't one-size-fits-all tools. As we mentioned earlier, different hash functions serve different purposes depending on their design and hash speeds. They work at different operational speeds — some are faster while others are much slower. These speeds can aid or impede the security of a hashing algorithm depending on how you're using it. So, some fall under the umbrella of secure hashing algorithms while others do not.

How Does Hashing Work?

When you hash a message, you take a string of data of any size as your input, run it through a mathematical algorithm that results in the generation of an output of a fixed length.

In some methods of hashing, that original data input is broken up into smaller blocks of equal size. If there isn't enough data in any of the blocks for it to be the same size, then padding (1s and 0s) can be used to fill it out. Then those individual blocks of data are run through a hashing algorithm and result in an output of a hash value. The process looks something like this:



6.4 SECURITY OF HASH FUNCTIONS AND MACS

We can group attacks on the basis of hash functions and MACs: brute-force attacks and cryptanalysis.

Brute-Force Attacks-The nature of brute-force attacks differs somewhat for hash functions and MACs.

Hash Functions:-The strength of a hash function against brute-force attacks depends on the length of the hash code produced by the algorithm. There are three desirable properties:

- One-way: For any given code h, it is computationally infeasible to find x such that H(x) = h.

- Weak collision resistance: For any given block x, it is computationally infeasible to find y ≠x with H(y) = H(x).
- Strong collision resistance: It is computationally infeasible to find any pair (x, y) such that H(x) = H(y).

For a hash code of length n, the level of effort required, as we have seen is proportional to the following:

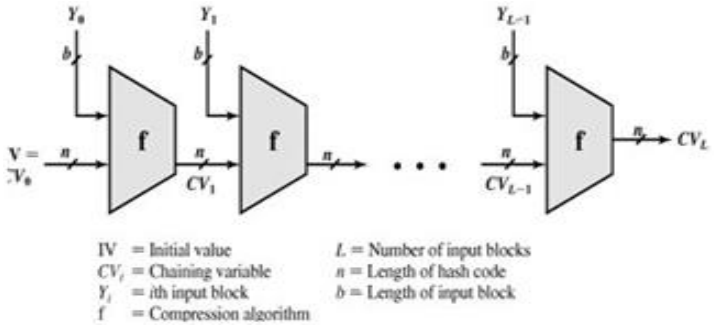
One way	2^n
Weak collision resistance	2^n
Strong collision resistance	$2^{n/2}$

If strong collision resistance is required, then the value $2^{n/2}$ determines the strength of the hash code against brute-force attacks. Thus a 128-bit code may be viewed as inadequate. The next step up, if a hash code is treated as a sequence of 32 bits, is a 160-bit hash length. With a hash length of 160 bits, the same search machine would require over four thousand years to find a collision. However, even 160 bits is now considered weak.

Message Authentication Codes:-A brute-force attack on a MAC is a more difficult undertaking because it requires known message-MAC pairs. To attack a hash code, we can proceed in the following way. Given a fixed message x with n-bit hash code h = H(x), a brute-force method of finding a collision is to pick a random bit string y and check if H(y) = H(x). The attacker can do this repeatedly off line. Whether an off-line attack can be used on a MAC algorithm depends on the relative size of the key and the MAC.

To proceed, we need to state the desired security property of a MAC algorithm, which can be expressed as follows: Computation resistance: Given one or more text-MAC pairs [x_i, C(K, x_i)], it is computationally infeasible to compute any text-MAC pair [x, C(K, x)] for any new input x ≠ x_i.In other words, the attacker would like to come up with the valid MAC code for a given message x. There are two lines of attack possible: Attack the key space and attack the MAC value.

General Structure of Secure Hash Code



IV = Initial value
CV_i = Chaining variable
Y_i = ith input block
f = Compression algorithm
L = Number of input blocks
n = Length of hash code
b = Length of input block

Cryptanalysis-The way to measure the resistance of a hash or MAC algorithm to cryptanalysis is to compare its strength to the effort required for a brute-force attack. That is, an ideal hash or MAC algorithm will require a cryptanalytic effort greater than or equal to the brute-force effort.

Hash Functions:-In past few years there has been considerable effort, and some successes, in developing cryptanalytic attacks on hash functions. To understand these, we need to look at the overall structure of a typical secure hash function, This structure, referred to as an iterated hash function and is the structure of most hash functions in use today, including SHA and Whirlpool. The hash function takes an input message and partitions it into L fixed-sized blocks of b bits each. If necessary, the final block is padded to b bits. The final block also includes the value of the total length of the input to the hash function. The inclusion of the length makes the job of the opponent more difficult.

6.5 SECURE HASH ALGORITHM

The Secure Hash Algorithm (SHA) was developed by the National Institute of Standards and Technology (NIST) and published as a federal information processing standard (FIPS 180) in 1993; a revised version was issued as FIPS 180-1 in 1995 and is generally referred to as SHA-1. The actual standards document is entitled Secure Hash Standard. SHA is based on the MD4 algorithm which is a message digest algorithm that was developed by Ron Rivest at MIT (the "R" in the RSA (Rivest-ShamirAdelman) public key encryption algorithm). MD4 was later replaced with the popular MD5 algorithm also by Ron Rivest however advances in cryptanalysis and computing power have led to their decline in popularity. Both MD4 and MD5 produce a 128 bit message digest whereas SHA-1 produces a 160 bit as will be seen.

In 2002, NIST produced a revised version of the standard, FIPS180-2, that defined three new versions of SHA, with hash value lengths of 256, 384, and 512 bits, known as SHA-256, SHA-384, and SHA-512 respectively. These new versions have the same underlying structure and use the same types of modular arithmetic and logical binary operations as SHA-1. In 2005, NIST announced the intention to phase out approval of SHA-1 and move to a reliance on the other SHA versions by 2010. Shortly thereafter, a research team described an attack in which two separate messages could be found that deliver the same SHA-1 hash using 2 69 operations, far fewer that the 2 80 operations previously thought needed to find a collision with an SHA-1 hash. This result should hasten the transition to the other versions of SHA however we will still concentrate on SHA-1 as the underlying structures of the others is the same. SHA-1 takes as input a message with a maximum length of less than 2 64 bits and produces as output a 160 bit message digest.

The input is processed in 512-bit blocks. Figure 10.4 depicts the overall processing of a message to produce a digest. Although this diagram has MD5 as the hash function the structure is exactly the same for SHA-1 with the exception that the message length is limited in size (its isn't for MD5) and the hash value (and intermediate values CVi) are 160 bits and not 128 as shown (which is the case for MD5). The processing consists of the following 5 steps:

1. Append padding bits: The message is padded so that its length is congruent to 448 modulo 512 (length ≡ 448 (mod 512)). That is, the length of the padded message is 64 bits less than an integer multiple of 512 bits. Padding is always added, even if the message is already of the desired length. Thus, the number of padding bits is in the range of 1 to 512 bits. The padding consists of a single 1-bit followed by the necessary number of 0-bits.

2. Append length: A block of 64 bits is appended to the message. This block is treated as an unsigned 64-bit integer (most significant byte first) and contains the length of the original message (before padding).

3. Initialize MD buffer: A 160 bit buffer is used to hold intermediate values and final results of the Hash function represented as 5, 32 bit registers (A, B, C, D, E) initialized as follows: A = 67452301 B = EF CDAB89 C = 98BADCF E D = 10325476 E = C3D2E1F0

4. Process message in 512 bit (16 word) blocks: The heart of the algorithm is a module which consists of four "rounds" of processing of 20 steps each (see figure). Each round has similar structure but uses a different primitive logical function (f1, f2, f3 and f4). Each round takes as input the current 512-bit block being processed (Yq) and the 160-bit buffer value ABCDE and updates the contents of the buffer. Each round also makes use of an additive constant Kt where $0 \le t \le 79$ indicates one of the 80 steps across four rounds. In fact, only four distinct constants are used (one for $0 \le t \le 19$, $20 \le t \le 39$, $40 \le t \le 59$ and $60 \le t \le 79$). The output of the fourth round is added (modulo 2 32) to the input to the first round (CVq) to produce CVq+1.

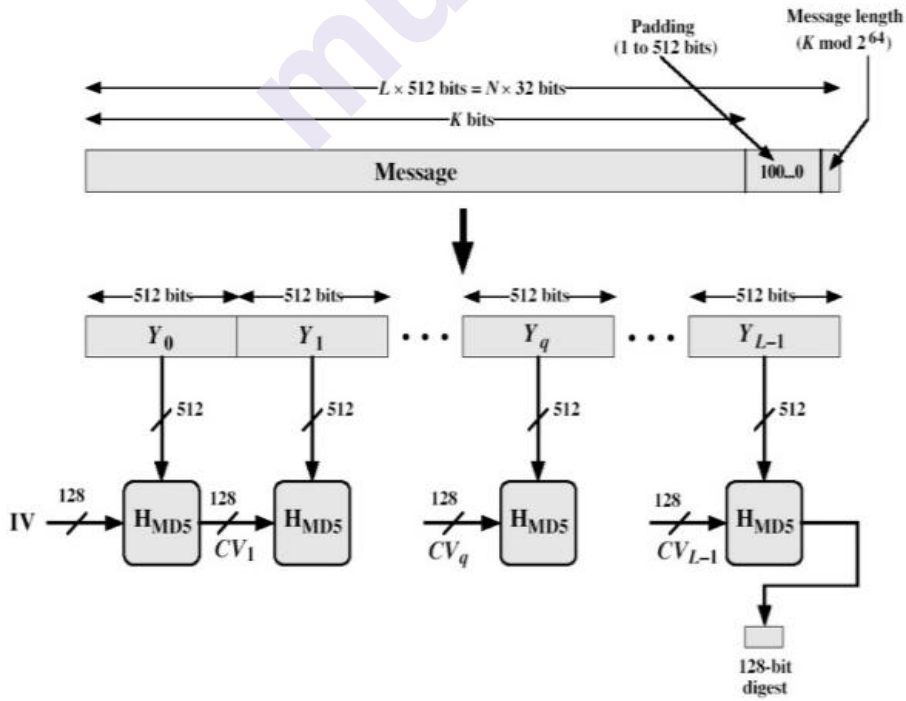
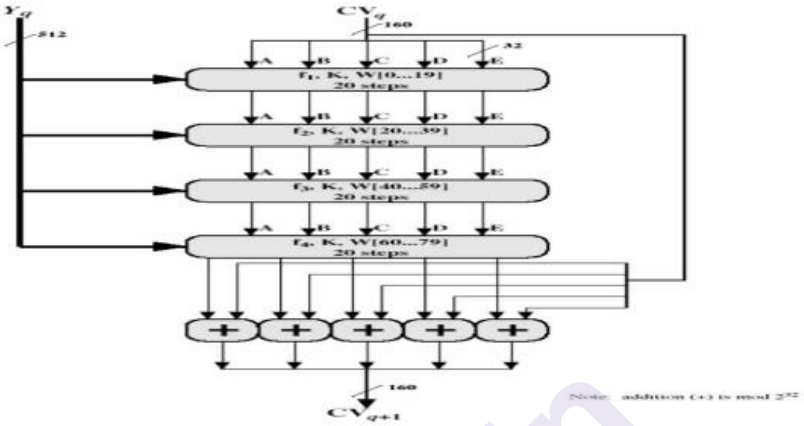


Fig. Message Digest Generation Using MD5 (equally applicable to SHA-1 with 160 bits instead of 128 etc.).

5. Output after all L 512 bit blocks have been processed the output from the Lth stage is the 160 bit digest.



We can summarise the behavior SHA-1 as follows:

$CV_0 = IV$

$CV_{q+1} = SUM32(CV_q, ABCDE_q)$

$MD = CV_L$

where

IV = initial value of the ABCDE buffer, defined in step 3.

$ABCDE_q$ = the output of the last round of processing of the qth message block.

L = the number of blocks in the message (including padding and length fields).

SUM_{32} = Addition modulo 2^{32} performed separately on each word of the pair of inputs. MD = final message digest value.

6.6 HMAC

HMAC algorithm stands for Hashed or Hash based Message Authentication Code. It is a result of work done on developing a MAC derived from cryptographic hash functions. HMAC is a great resistant towards cryptanalysis attacks as it uses the Hashing concept twice. HMAC consists of twin benefits of Hashing and MAC, and thus is more secure than any other authentication codes. RFC 2104 has issued HMAC, and HMAC has been made compulsory to implement in IP security. The FIPS 198 NIST standard has also issued HMAC.

Objectives –

As the Hash Function, HMAC is also aimed to be one way, i.e, easy to generate output from input but complex the other way round.

It aims at being less effected by collisions than the hash functions.

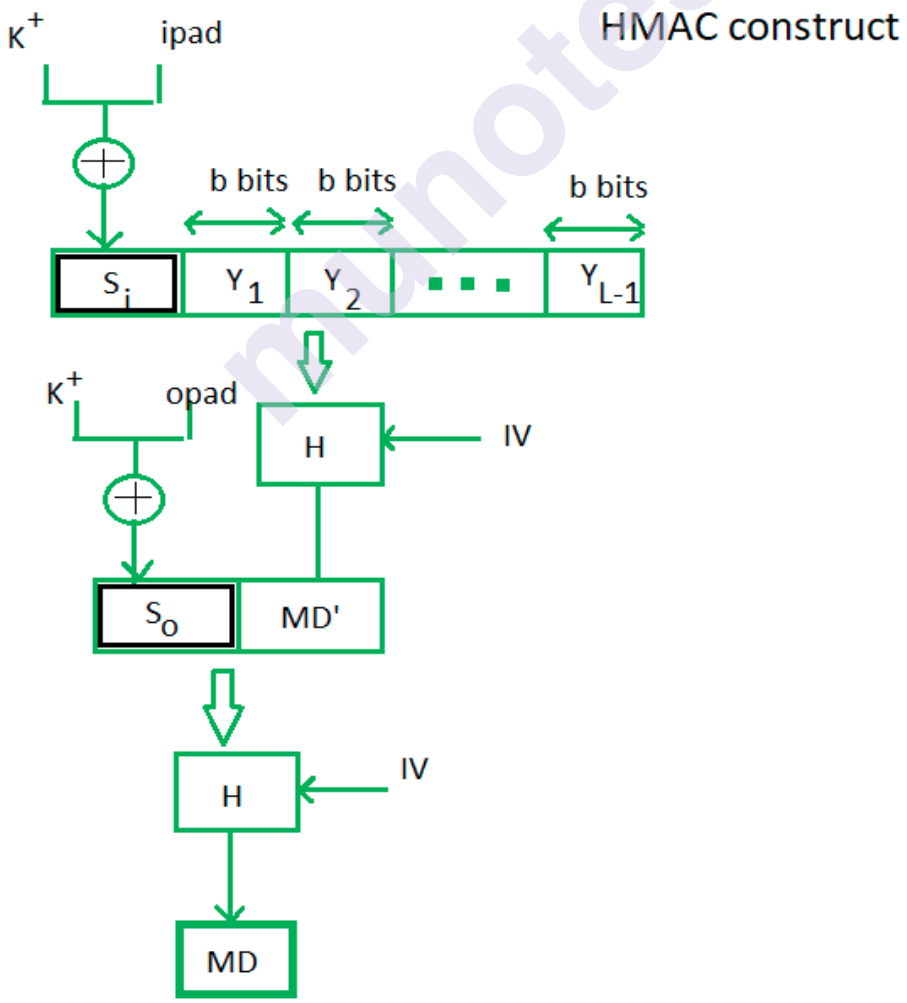
HMAC reuses the algorithms like MD5 and SHA-1 and checks to replace the embedded hash functions with more secure hash functions, in case found.

HMAC tries to handle the Keys in more simple manner.

HMAC algorithm –

The working of HMAC starts with taking a message M containing blocks of length b bits. An input signature is padded to the left of the message and the whole is given as input to a hash function which gives us a temporary message digest MD'. MD' again is appended to an output signature and the whole is applied a hash function again, the result is our final message digest MD.

Here is a simple structure of HMAC:



Here, H stands for Hashing function,

M is original message

Si and So are input and output signatures respectively,

Yi is the ith block in original message M, where i ranges from [1, L)

L = the count of blocks in M

K is the secret key used for hashing

IV is an initial vector (some constant)

The generation of input signature and output signature Si and So respectively.

$S_i = K^+ \oplus ipad$ where K^+ is nothing but K padded with zeros on the left so that the result is b bits in length

$S_o = K^+ \oplus opad$ where ipad and opad are 00110110 and 01011100 respectively taken b/8 times repeatedly.

$MD' = H(S_i || M)$

$MD = H(S_o || MD')$ or $MD = H(S_o || H(S_i || M))$

6.7 SUMMARY

- Message authentication allows one party—the Sender—to send a message to another party—the Receiver—in such a way that if the message is modified en route, then the Receiver will almost certainly detect this.
- Message authentication is also called "data-origin authentication," since it authenticates the point-of-origin for each message.
- In the context of communications across a network, the following attacks can be identified:
    - Disclosure, Traffic analysis, Masquerade, Content Modification, Sequence modification, Timing modification, Repudiation
- There are three types of functions that may be used to produce an authenticator:
    i. Message Encryption
    ii. Message authentication code (MAC)
    iii. A Hash function

- Encryption is a means of securing digital data using one or more mathematical techniques, along with a password or "key" used to decrypt the information.
- The encryption process where **same keys are used for encrypting and decrypting** the information is known as Symmetric Key Encryption.
- The encryption process where **different keys are used for encrypting and decrypting the information** is known as Asymmetric Key Encryption.
- MAC algorithm is a symmetric key cryptographic technique to provide message authentication. For establishing MAC process, the sender and receiver share a symmetric key K.
- A hash function is a serious mathematical process that holds a critical role in public key cryptography.
- HMAC algorithm stands for Hashed or Hash based Message Authentication Code. It is a result of work done on developing a MAC derived from cryptographic hash functions. HMAC is a great resistant towards cryptanalysis attacks as it uses the Hashing concept twice.

6.8 QUESTIONS

1. In the context of communications across a network, what are the attacks can be identified?
2. What are the types of Authentication Function?
3. Write a short note on Message Encryption.
4. Explain Message authentication code (MAC).
5. Write a short note on A Hash function.
6. Write a short note on Symmetric Key encryption
7. Write a short note on Asymmetric Key encryption
8. Differentiate between Symmetric Key encryption and Asymmetric Key encryption
9. Write Properties of a Strong Hash Algorithm.
10. How Does Hashing Work?
11. Write a short note on HMAC.

6.9 REFERENCE FOR FURTHER READING

- https://www.cc.gatech.edu/~aboldyre/teaching/Fall05cs6260/m-mac.pdf
- http://www.facweb.iitkgp.ac.in/~sourav/AuthenticationRequirements.pdf
- https://www.slideshare.net/RajasekarVr/message-authentication-75333084-
- https://www.thesslstore.com/blog/what-is-a-hash-function-in-cryptography-a-beginners-guide/
- http://www.faadooengineers.com/online-

❄❄❄❄❄❄❄

7

DIGITAL SIGNATURES AND AUTHENTICATION AND AUTHENTICATION APPLICATIONS

Unit Structure :

7.0 OBJECTIVE

In this chapter we mainly discuss about Authentication Process and Protocol which are responsible for implementing Authentication. Authentication is important because it enables organizations to keep their networks secure by permitting only authenticated users (or processes) to access its protected resources, which may include computer systems, networks, databases, websites and other network-based applications or services.

7.1 INTRODUCTION

Producing a secure authentication process that keeps users happy is easier said than done, but it's necessary in order to keep them safe online.

Controlling access is the basis of all security. The right people should be allowed in, and the wrong people kept out. This is done by confirming – or authenticating – the identity of the person seeking access, and then checking that the person is authorized to enter.

Authentication is normally achieved by the presentation of a User ID (usually the user's email address) to identify the person, and a secret password known only to that person to confirm the identity.

But there are huge problems with this process. Fundamentally, it does not authenticate the person; if a criminal acquires and uses the person's User ID and password, the criminal is automatically authorized to gain access. So, strictly speaking, a password does not authenticate the user, it simply authorizes a device regardless of who is using it.

7.2 DIGITAL SIGNATURES

The Digital Signature is a technique which is used to validate the authenticity and integrity of the message. We know that there are four aspects of security: privacy, authentication, integrity, and non-repudiation. We have already discussed the first aspect of security and other three aspects can be achieved by using a digital signature.

The basic idea behind the Digital Signature is to sign a document. When we send a document electronically, we can also sign it. We can sign a document in two ways: to sign a whole document and to sign a digest.

Message authentication protects two parties who exchange messages from any third party. However, it does not protect the two parties against each other. Several forms of dispute between the two are possible. For example, suppose that John sends an authenticated message to Mary using one of the schemes described earlier. Consider the following disputes that could arise:

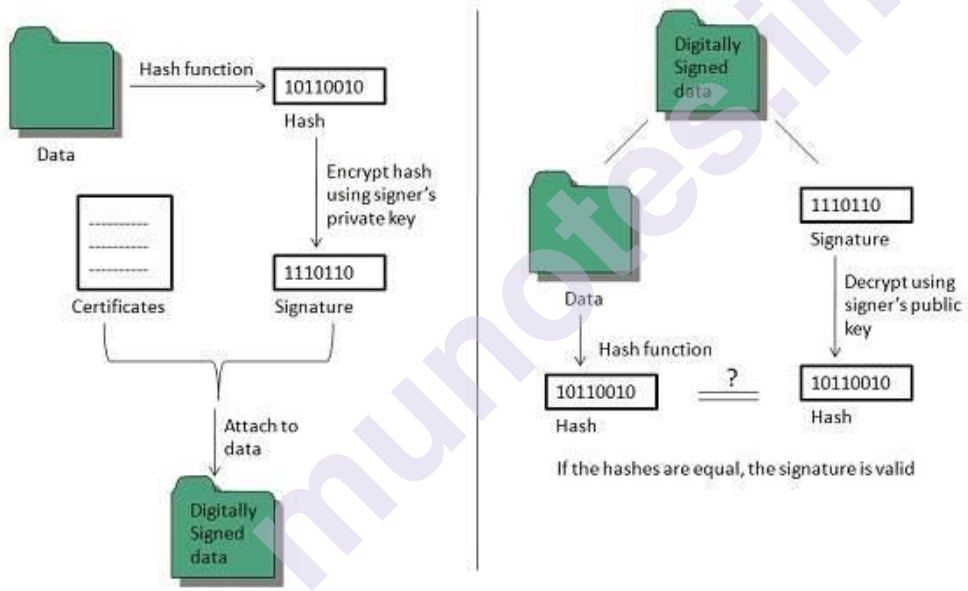
- Mary may forge a different message and claim that it came from John. Mary would simply have to create a message and append an authentication code using the key that John and Mary share.
- John can deny sending the message. Because it is possible for Mary to forge a message, there is no way to prove that John did in fact send the message.

  Both scenarios are of legitimate concern. In situations where there is not complete trust between sender and receiver, something more than authentication is needed. The most attractive solution to this problem is the digital signature. The digital signature is analogous to the handwritten signature. It must have the following properties:

- It must verify the author and the date and time of the signature.
- It must authenticate the contents at the time of the signature.
- It must be verifiable by third parties, to resolve disputes. Thus, the digital signature function includes the authentication function.

On the basis of these properties, we can formulate the following requirements for a digital signature:

- The signature must be a bit pattern that depends on the message being signed.
- The signature must use some information unique to the sender, to prevent both forgery and denial.
- It must be relatively easy to produce the digital signature.
- It must be relatively easy to recognise and verify the digital signature.
- It must be computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message.
- It must be practical to retain a copy of the digital signature in storage.



Digital Signature is used to achieve the following three aspects:

- **Integrity:** The Digital Signature preserves the integrity of a message because, if any malicious attack intercepts a message and partially or totally changes it, then the decrypted message would be impossible.
- **Authentication:** We can use the following reasoning to show how the message is authenticated. If an intruder (user X) sends a message pretending that it is coming from someone else (user A), user X uses her own private key to encrypt the message. The message is decrypted by using the public key of user A. Therefore this makes the message unreadable. Encryption with X's private key and decryption with A's public key results in garbage value.

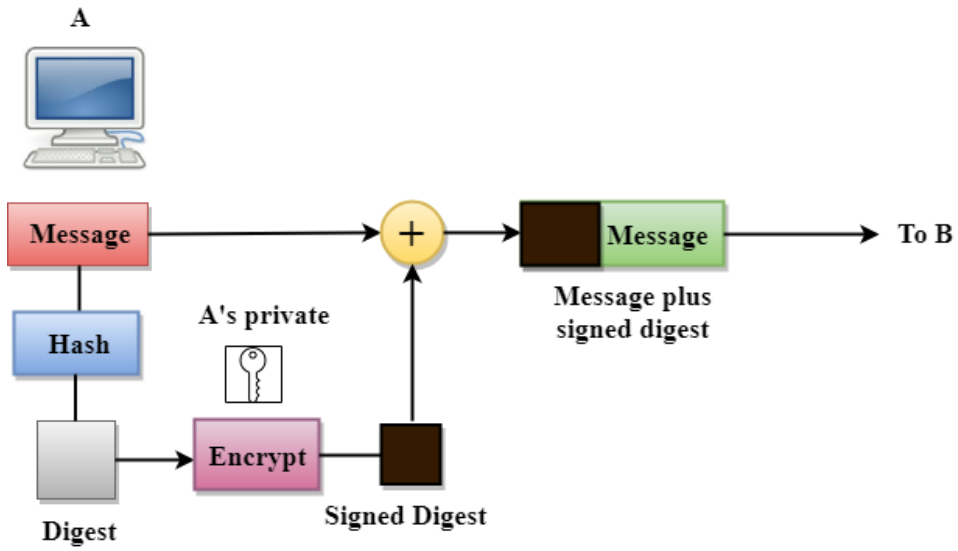
- **Non-Repudiation:** Digital Signature also provides non-repudiation. If the sender denies sending the message, then her private key corresponding to her public key is tested on the plaintext. If the decrypted message is the same as the original message, then we know that the sender has sent the message.

  Note: Digital Signature does not provide privacy. If there is a need for privacy, then another layer of encryption/decryption is applied.

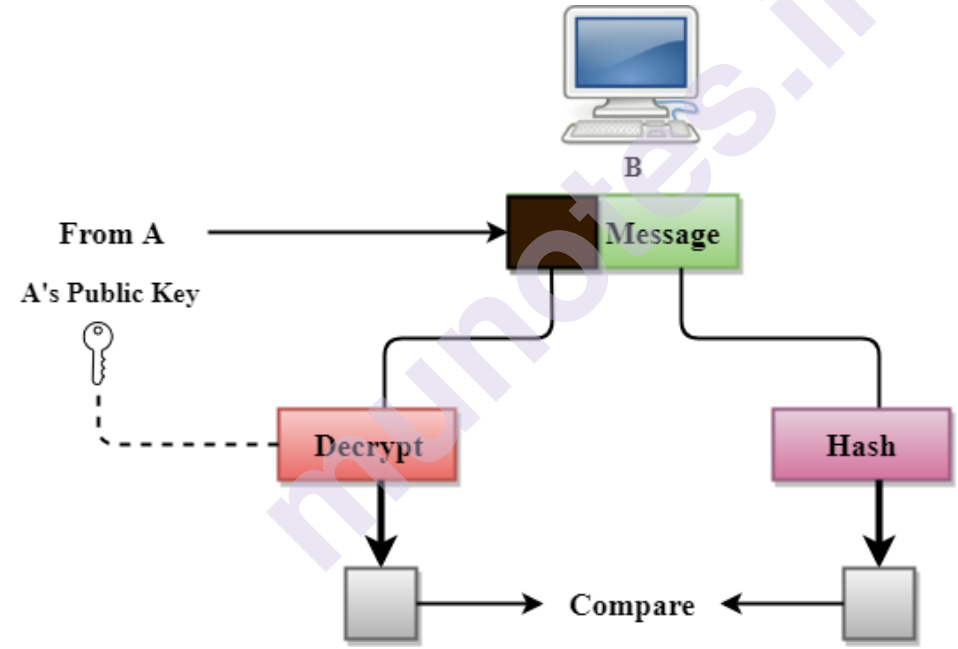
Signing the Digest

- Public key encryption is efficient if the message is short. If the message is long, a public key encryption is inefficient to use. The solution to this problem is to let the sender sign a digest of the document instead of the whole document.
- The sender creates a miniature version (digest) of the document and then signs it, the receiver checks the signature of the miniature version.
- The hash function is used to create a digest of the message. The hash function creates a fixed-size digest from the variable-length message.
- The two most common hash functions used: MD5 (Message Digest 5) and SHA-1 (Secure Hash Algorithm 1). The first one produces 120-bit digest while the second one produces a 160-bit digest.
- A hash function must have two properties to ensure the success:
- First, the digest must be one way, i.e., the digest can only be created from the message but not vice versa.
- Second, hashing is a one-to-one function, i.e., two messages should not create the same digest.
- Following are the steps taken to ensure security:
- The miniature version (digest) of the message is created by using a hash function.
- The digest is encrypted by using the sender's private key.
- After the digest is encrypted, then the encrypted digest is attached to the original message and sent to the receiver.
- The receiver receives the original message and encrypted digest and separates the two. The receiver implements the hash function on the original message to create the second digest, and it also decrypts the received digest by using the public key of the sender. If both the digests are same, then all the aspects of security are preserved.

At the Sender site



At the Receiver site



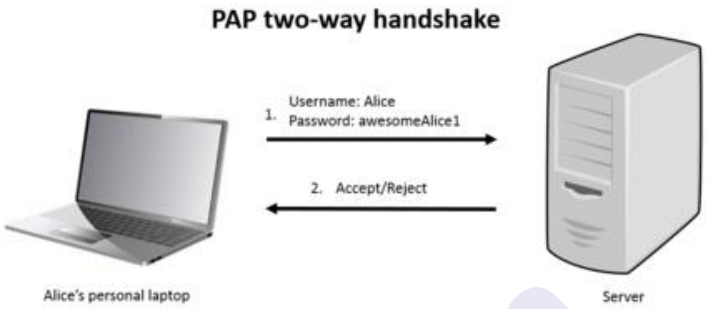
7.3 AUTHENTICATION PROTOCOLS

7.3.1 Authentication protocols developed for PPP Point-to-Point Protocol

1. PAP - Password Authentication Protocol

Password Authentication Protocol is one of the oldest authentication protocols. Authentication is initialized by the client sending a packet with credentials (username and password) at the beginning of the connection, with the client repeating the authentication request until acknowledgement is received. It is highly insecure because credentials are sent "in the clear" and repeatedly, making it

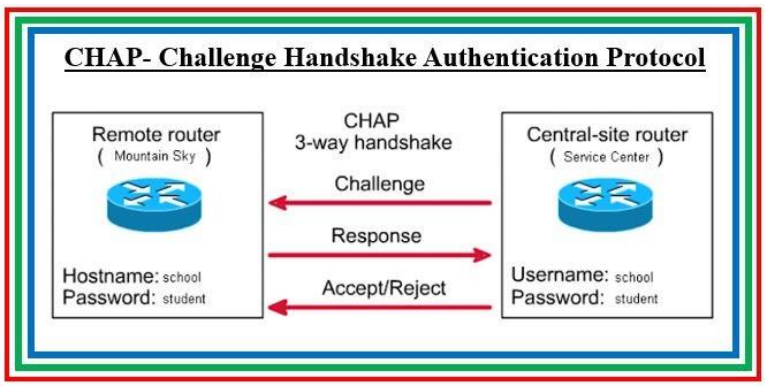
vulnerable even to the most simple attacks like eavesdropping and man-in-the-middle based attacks. Although widely supported, it is specified that if an implementation offers a stronger authentication method, that method must be offered before PAP. Mixed authentication (e.g. the same client alternately using both PAP and CHAP) is also not expected, as the CHAP authentication would be compromised by PAP sending the password in plain-text.



2. CHAP - Challenge-handshake authentication protocol

The Challenge-Handshake Authentication Protocol (CHAP) is used to periodically verify the identity of the peer using a 3-way handshake. This is done upon initial link establishment, and MAY be repeated anytime after the link has been established.

1. After the Link Establishment phase is complete, the Authenticator sends a "challenge" message to the peer.
2. The peer responds with a value calculated using a "one-way hash" function.
3. The authenticator checks the response against its own calculation of the expected hash value. If the values match, the authentication is acknowledged; otherwise the connection SHOULD be terminated.
4. At random intervals, the authenticator sends a new challenge to the peer, and repeats steps 1 to 3.



3. EAP - Extensible Authentication Protocol

Extensible Authentication Protocol (EAP) is an authentication framework, not a specific authentication mechanism, frequently used in wireless networks and point-to-point connections. It provides some common functions and negotiation of authentication methods called EAP methods.

The EAP protocol can support multiple authentication mechanisms without having to pre-negotiate a particular one. There are currently about 40 different methods defined.

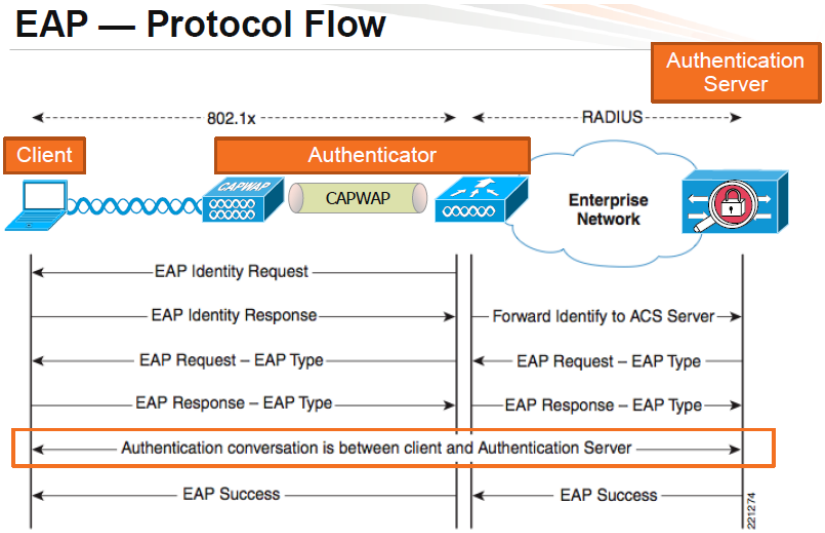
EAP authentication is initiated by the server (authenticator), whereas many other authentication protocols are initiated by the client (peer). The EAP authentication exchange proceeds as follows:

1) The authenticator (the server) sends a Request to authenticate the peer (the client).

2) The peer sends a Response packet in reply to a valid Request.

3) The authenticator sends an additional Request packet, and the peer replies with a Response. The sequence of Requests and Responses continues as long as needed. EAP is a 'lock step' protocol, so that other than the initial Request, a new Request cannot be sent prior to receiving a valid Response.

4) The conversation continues until the authenticator cannot authenticate the peer (unacceptable Responses to one or more Requests), in which case the authenticator implementation MUST transmit an EAP Failure (Code 4). Alternatively, the authentication conversation can continue until the authenticator determines that successful authentication has occurred, in which case the authenticator MUST transmit an EAP Success (Code 3).



7.4 DIGITAL SIGNATURE STANDARD

The National Institute of Standards and Technology (NIST) has published Federal Information Processing Standard FIPS 186, known as the Digital Signature Standard (DSS). The DSS was originally proposed in 1991 and revised in 1993 in response to public feedback concerning the security of the scheme. There was a further minor revision in 1996. In 2000, an expanded version of the standard was issued as FIPS 186-2, subsequently updated to FIPS 186-3 in 2009. This latest version also incorporates digital signature algorithms based on RSA and on elliptic curve cryptography. In this section, we discuss the original DSS algorithm.

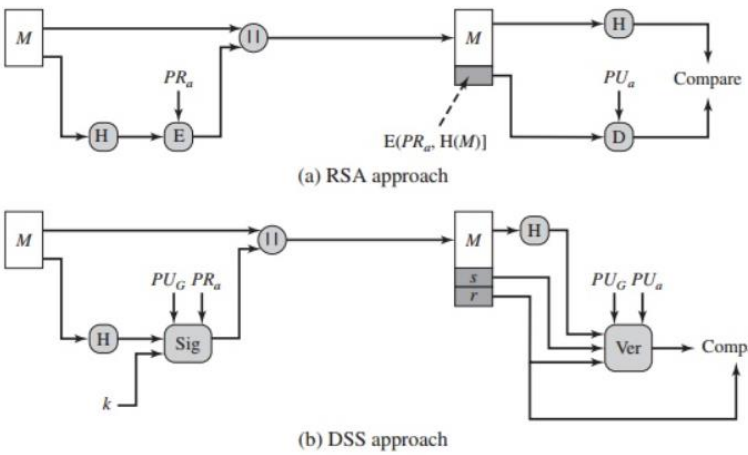
THE DSS APPROACH

The DSS uses an algorithm that is designed to provide only the digital signature function. Unlike RSA, it cannot be used for encryption or key exchange. Nevertheless, it is a public-key technique.

Figure contrasts the DSS approach for generating digital signatures to that used with RSA. In the RSA approach, the message to be signed is input to a hash function that produces a secure hash code of fixed length. This hash code is then encrypted using the sender's private key to form the signature. Both the mes- sage and the signature are then transmitted. The recipient takes the message and produces a hash code. The recipient also decrypts the signature using the sender's public key. If the calculated hash code matches the decrypted signature, the signa- ture is accepted as valid. Because only the sender knows the private key, only the sender could have produced a valid signature.

The DSS approach also makes use of a hash function. The hash code is pro- vided as input to a signature function along with a random number k generated for this particular signature. The signature function also depends on the sender's private key (PRa) and a set of parameters known to a group of communicating principals.

We can consider this set to constitute a global public key (PUG).1 The result is a signature consisting of two components, labeled s and r.



At the receiving end, the hash code of the incoming message is generated. This plus the signature is input to a verification function. The verification function also depends on the global public key as well as the sender's public key (PUa), which is paired with the sender's private key. The output of the verification function is a value that is equal to the signature component r if the signature is valid. The signa- ture function is such that only the sender, with knowledge of the private key, could have produced the valid signature.

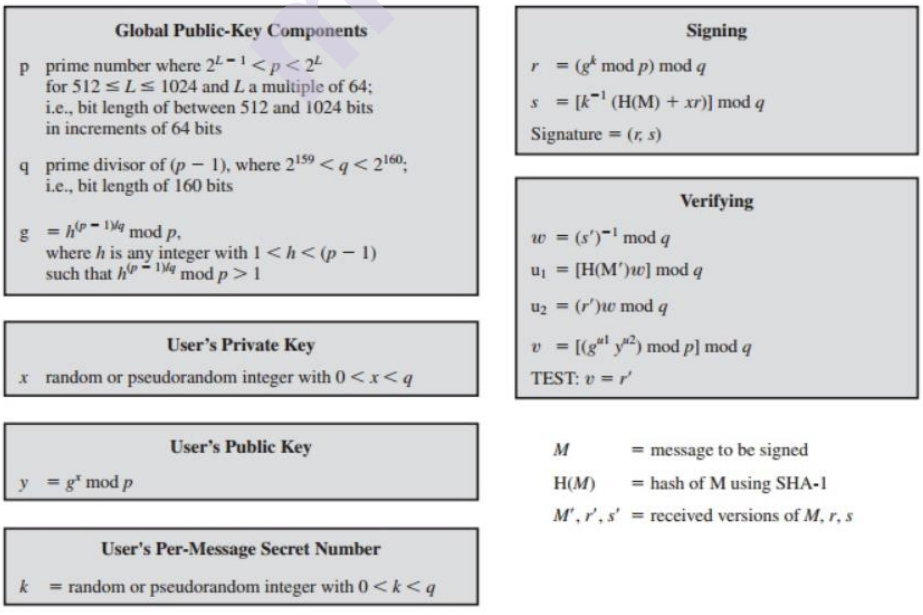
We turn now to the details of the algorithm.

The Digital Signature Algorithm

The DSA is based on the difficulty of computing discrete logarithms and is based on schemes originally presented by ElGamal [ELGA85] and Schnorr [SCHN91].

Figure summarizes the algorithm. There are three parameters that are public and can be common to a group of users. A 160-bit prime number q is chosen. Next, a prime number p is selected with a length between 512 and 1024 bits such that q divides (p - 1). Finally, g is chosen to be of the form h(p - 1)/q mod p, where h is an integer between 1 and (p - 1) with the restriction that g must be greater than 1.2 Thus, the global public-key components of DSA have the same for as in the Schnorr signature scheme.

With these numbers in hand, each user selects a private key and generates a public key. The private key x must be a number from 1 to (q - 1) and should be cho- sen randomly or pseudorandomly. The public key is calculated from the private key as y = gx mod p. The calculation of y given x is relatively straightforward. However, given the public key y, it is believed to be computationally infeasible to determine x, which is the discrete logarithm of y to the base g, modp

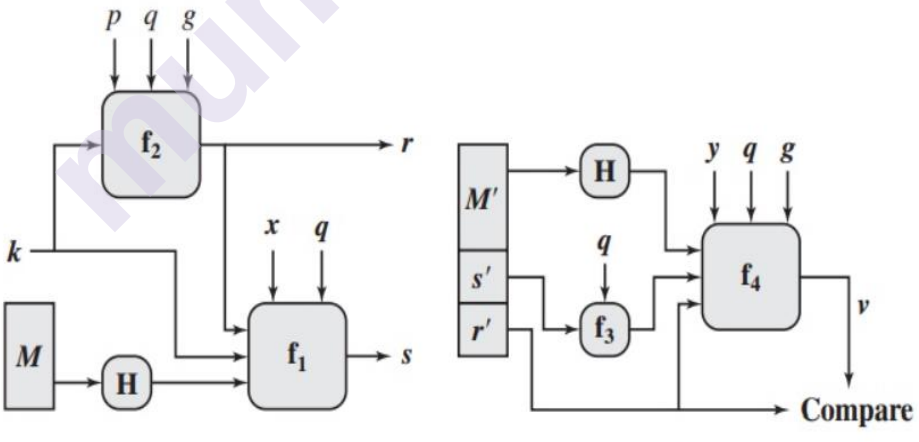


To create a signature, a user calculates two quantities, r and s, that are func- tions of the public key components (p, q, g), the user's private key (x), the hash code of the message H(M), and an additional integer k that should be generated randomly or pseudorandomly and be unique for each signing. At the receiving end, verification is performed using the formulas shown in Figure.The receiver generates a quantity v that is a function of the public key com- ponents, the sender's public key, and the hash code of the incoming message. If this quantity matches the r component of the signature, then the signature is validated.

Figure depicts the functions of signing and verifying.

The structure of the algorithm, as revealed in Figure, is quite interesting. Note that the test at the end is on the value r, which does not depend on the message at all. Instead, r is a function of k and the three global public-key components. The multiplicative inverse of k (mod q) is passed to a function that also has as inputs the message hash code and the user's private key. The structure of this function is such that the receiver can recover r using the incoming message and signature, the public key of the user, and the global public key. It is certainly not obvious from Figure that such a scheme would work. A proof is provided in Appendix K. Given the difficulty of taking discrete logarithms, it is infeasible for an oppo-nent to recover k from r or to recover x from s.

Another point worth noting is that the only computationally demanding task in signature generation is the exponential calculation gk mod p. Because this value does not depend on the message to be signed, it can be computed ahead of time.



$s = f_1(H(M), k, x, r, q) = (k^{-1} (H(M) + xr)) \bmod q$

$r = f_2(k, p, q, g) = (g^k \bmod p) \bmod q$

(a) Signing

$w = f_3(s', q) = (s')^{-1} \bmod q$

$v = f_4(y, q, g, H(M'), w, r')$

$= ((g^{(H(M')w) \bmod q} y^{r'w \bmod q}) \bmod p) \bmod q$

(b) Verifying

Indeed, a user could precalculate a number of values of *r* to be used to sign documents as needed. The only other somewhat demanding task is the determina- tion of a multiplicative inverse, k^{-1}. Again, a number of these values can be precalculated.

7.5 AUTHENTICATION APPLICATIONS

7.5.1 Kerberos

Kerberos authentication is currently the default authorization technology used by Microsoft Windows, and implementations of Kerberos exist in Apple OS, FreeBSD, UNIX, and Linux.

Microsoft introduced their version of Kerberos in Windows2000. It has also become a standard for websites and Single-Sign-On implementations across platforms. The Kerberos Consortium maintains Kerberos as an open-source project.

Kerberos is a vast improvement on previous authorization technologies. The strong cryptography and third-party ticket authorization make it much more difficult for cybercriminals to infiltrate your network. It is not totally without flaws, and in order to defend against those flaws, you need to first understand them.

Kerberos has made the internet and its denizens more secure, and enables users to do more work on the Internet and in the office without compromising safety.

Kerberos Protocol Flow Overview

Let's take a more detailed look at what Kerberos authentication is and how it works by breaking it down into its core components.

Here are the principal entities involved in the typical Kerberos workflow:

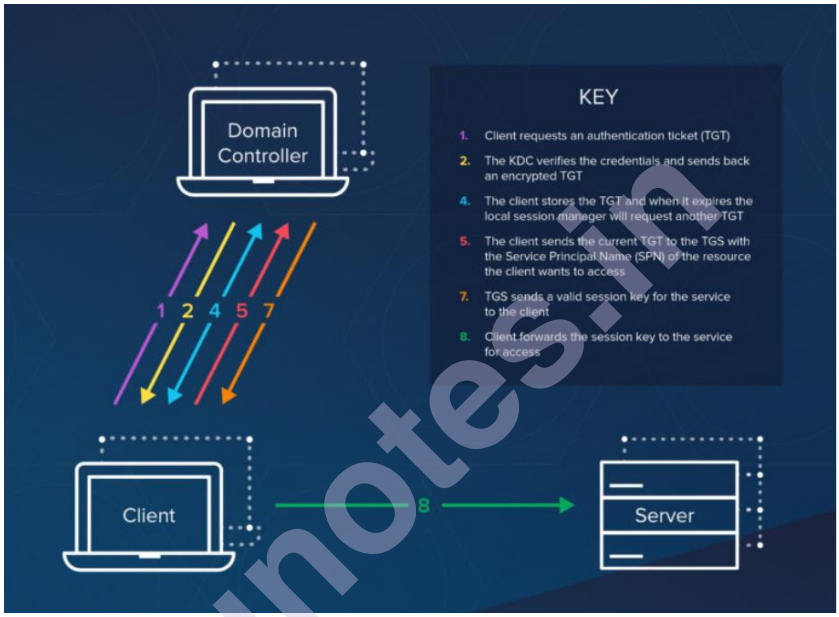
- **Client.** The client acts on behalf of the user and initiates communication for a service request
- **Server.** The server hosts the service the user wants to access
- **Authentication Server (AS).** The AS performs the desired client authentication. If the authentication happens successfully, the AS issues the client a ticket called TGT (Ticket Granting Ticket). This ticket assures the other servers that the client is authenticated
- **Key Distribution Center (KDC).** In a Kerberos environment, the authentication server logically separated into three parts: A database (db), the Authentication Server (AS), and the Ticket Granting Server (TGS). These three parts, in turn, exist in a single server called the Key Distribution Center
- **Ticket Granting Server (TGS).** The TGS is an application server that issues service tickets as a service

Now let's break down the protocol flow.

First, there are three crucial secret keys involved in the Kerberos flow. There are unique secret keys for the client/user, the TGS, and the server shared with the AS.

- **Client/user.** Hash derived from the user's password
- **TGS secret key.** Hash of the password employed in determining the TGS
- **Server secret key.** Hash of the password used to determine the server providing the service.

The protocol flow consists of the following steps:



Here are the most basic steps taken to authenticate in a Kerberized environment.

1. Client requests an authentication ticket (TGT) from the Key Distribution Center (KDC)
2. The KDC verifies the credentials and sends back an encrypted TGT and session key
3. The TGT is encrypted using the Ticket Granting Service (TGS) secret key
4. The client stores the TGT and when it expires the local session manager will request another TGT (this process is transparent to the user)

   If the Client is requesting access to a service or other resource on the network, this is the process:

5. The client sends the current TGT to the TGS with the Service Principal Name (SPN) of the resource the client wants to access

6. The KDC verifies the TGT of the user and that the user has access to the service

7. TGS sends a valid session key for the service to the client

8. Client forwards the session key to the service to prove the user has access, and the service grants access.

7.5.2 X.509 Authentication

X.509 is a standard format for public key certificates, digital documents that securely associate cryptographic key pairs with identities such as websites, individuals, or organizations.

First introduced in 1988 alongside the X.500 standards for electronic directory services, X.509 has been adapted for internet use by the IETF's Public-Key Infrastructure (X.509) (PKIX) working group. RFC 5280 profiles the X.509 v3 certificate, the X.509 v2 certificate revocation list (CRL), and describes an algorithm for X.509 certificate path validation.

Common applications of X.509 certificates include:

- SSL/TLS and HTTPS for authenticated and encrypted web browsing
- Signed and encrypted email via the S/MIME protocol
- Code signing
- Document signing
- Client authentication
- Government-issued electronic ID

Key Pairs and Signatures

No matter its intended application(s), each X.509 certificate includes a **public key**, **digital signature**, and information about both the identity associated with the certificate and its issuing **certificate authority (CA)**:

- The **public key** is part of a **key pair** that also includes a **private key**. The private key is kept secure, and the public key is included in the certificate. This public/private key pair:
- Allows the owner of the private key to digitally sign documents; these signatures can be verified by anyone with the corresponding public key.
- Allows third parties to send messages encrypted with the public key that only the owner of the private key can decrypt.
- A **digital signature** is an encoded hash (fixed-length digest) of a document that has been encrypted with a private key. When an

X.509 certificate is signed by a **publicly trusted CA**, such as SSL.com, the certificate can be used by a third party to verify the identity of the entity presenting it.

Note: Not all applications of X.509 certificates require public trust. For example, a company can issue its own privately trusted certificates for internal use. For more information, please read our article on Private vs. Public PKI.

- Each X.509 certificate includes fields specifying the **subject**, **issuing CA**, and other required information such as the certificate's **version** and **validity period**. In addition, v3 certificates contain a set of **extensions** that define properties such as acceptable key usages and additional identities to bind a key pair to.

Certificate Fields and Extensions

To review the contents of a typical X.509 certificate in the wild, we will examine www.ssl.com's SSL/TLS certificate, as shown in Google Chrome. (You can check all of this in your own browser for any HTTPS website by clicking the lock on the left side of the address bar.)

- The first group of details includes information about the **Subject**, including the name and address of the company and the **Common Name** (or Fully Qualified Domain Name) of the website that the certificate is intended to protect. (**Note:** the **Serial Number** shown in this subject field is a Nevada business identification number, not the serial number of the certificate itself.)

- Scrolling down, we encounter information about the **Issuer**. Not coincidentally, in this case, the **Organization** is "SSL Corp" for both subject and issuer, but the issuer's **Common Name** is the name of the issuing CA certificate rather than a URL.

Issuer Name	
Country or Region	US
State/Province	Texas
Locality	Houston
Organization	SSL Corp
Common Name	SSL.com EV SSL Intermediate CA RSA R3

- Below the Issuer, we see the certificate's **Serial Number** (a positive integer uniquely identifying the certificate), **X.509 Version** (3), the **Signature Algorithm**, and dates specifying the certificate's **Validity Period**.

Serial Number	72 14 11 D3 D7 E0 FD 02 AA B0 4E 90 09 D4 DB 31
Version	3
Signature Algorithm	SHA-256 with RSA Encryption (1.2.840.113549.1.1.11)
Parameters	None
Not Valid Before	Thursday, April 18, 2019 at 5:15:06 PM Central Daylight Time
Not Valid After	Saturday, April 17, 2021 at 5:15:06 PM Central Daylight Time

- Next, we arrive at the **Public Key**, **Signature**, and associated information

Public Key Info	
Algorithm	RSA Encryption (1.2.840.113549.1.1.1)
Parameters	None
Public Key	256 bytes : AD 0F EF C1 97 5A 9B D8 1E B0 44 8D C6 C9 A0 28 C3 0E 68 1B 94 91 2E 77 EC AC AE BE 6C 78 04 5B A4 78 04 CE FB 07 4B 5D 34 F3 57 E5 0F FB 6B A4 2A A5 53 D3 D5 7F 3A 3C 54 4C EB 73 7B 5E A1 0A D9 7E 5F A9 5A C0 71 71 43 9D 6F BD 4C CC CC 43 8C CF 77 4B 9D 1A 75 CB 1F BD F7 3B D3 66 C6 CE 7C B0 5A FC D4 14 24 3A 2A C5 A8 61 6D 04 4D A6 36 2D B0 FC C4 B0 BF FC 41 27 71 E4 C3 90 AD 37 07 67 BE 5A 1A 81 9D AB 8A 71 92 A3 85 1D 99 E7 20 19 CF C4 FD AD 9F 6E 98 9F 5B CE 17 A1 FE 7B 4A 4F C9 F2 AD 21 C8 F7 1B 5D 10 79 59 85 DF 7E B8 A8 FE 3A D7 2F E2 02 DF D8 67 67 F4 63 9F FA B3 E7 47 63 48 3A C1 98 73 3D 9A 8D 8D DA AC C8 DF 50 32 BC A1 21 A6 10 56 AE E6 C6 10 2A 4E 54 41 5D 38 C1 37 77 78 1E 43 F8 70 2A 4B 4D EA B7 F9 51 CC 1C 17 4F 2A 1B 67 1C 2E E0 E0 2D 7C 59
Exponent	65537
Key Size	2,048 bits
Key Usage	Encrypt, Verify, Wrap, Derive
Signature	512 bytes : 36 07 E7 3B B7 45 97 CA 4D 6C B0 2A 3F 3F 38 43 12 3D 1C 4C 8E F6 87 18 5C 66 54 C5 E2 5B 4B ED ED DC 4C 23 EC 93 21 A1 19 28 DD 78 6D A6 0D E7 F4 F5 64 2E 1B 49 22 B4 EE FE E7 D3 0B 34 85 6A 12 14 09 33 4F 4E 52 FD 6B B0 04 9A EF 62 3C E3 78 6C 08 7A 87 25 63 61 28 B2 2C 22 10 5E 51 0F 03 7B 53 41 48 74 47 7D 3C 06 C3 E6 56 4D 96 9C 09 62 B2 76 00 9F 1A 3C C8 08 67 05 A1 C1 55 48 C2 37 EA 32 69 6A 12 E2 53 26 DB AC AB 79 94 88 8B 5B 5A 72 76 04 76 0D 53 CC 3D A9 38 95 E6 C1 BE E0 A4 C8 7E F6 AC 7E FF 34 ED 3B 5D 38 46 67 1C C5 79 D4 A8 81 8E 9C D0 CA F7 75 64 4F DC F8 4A 38 7C 88 18 DC D1 9B 50 F1 DB E8 61 D4 7D AE D8 9E 6E 86 E9 73 4A D4 2A F1 C7 CA 69 19 89 56 B5 FC BE 8D 90 F4 5A 21 89 A4 9A B7 3B F5 BA 24 34 A0 FD 5E 59 80 7A 45 93 3B 56 89 62 E3 4E E3 7E EB 13 2B 28 24 B9 86 EC DA 93 49 A1 0F 14 EF 54 93 BE 1E F4 55 CF 17 20 C5 01 C5 84 62 D5 64 38 1D 1C 59 08 D1 31 F8 AE 05 A4 1B BA 0A 67 51 9E A8 15 F2 E8 CF 8E 9E D8 88 52 21 89 CC 4F 98 13 0A 41 40 71 69 79 B0 A5 6A BE 77 AB 5E A1 D4 89 66 6C 02 C2 D1 43 0D A2 CA D7 7A 71 01 8B F7 98 21 74 89 E8 8B 27 38 28 CD 3E EA A7 78 AD 2A 3A 63 DB 3A D0 05 6B 4F C9 20 4E 01 38 DF 05 75 49 F7 9F 2E DC 19 31 A9 96 D7 2F 2D 4E 84 7C FA 7E F6 67 5A A1 E7 5C A1 72 3B 22 DC A5 FA F2 E7 DC D6 A8 6D A0 4D FD 78 C5 5C DC 34 D9 86 76 5B 1C 0D BB B1 E5 DB 64 2A 55 7F 20 4D 5D 4D 44 01 1D 79 A3 2D EC F5 6B CD BE 7B 52 67 1D FF 05 42 FB 42 7A A1 BC 4C 23 DF AF 16 B9 76 C9 69 86 02 34 F2 A9 CB B8 15 39 BA A5 F1 E6 72 7C 1D 5E 0C 48 D7 99 1F 50 98 2B 75 2D 67 58 79 A1 1A 05 5A

- In addition to the fields above, X.509 v3 certificates include a group of **Extensions** that offer additional flexibility in certificate use. For example, the **Subject Alternative Name** extension allows the certificate to be bound to multiple identities. (For this reason, multiple-domain certificates are sometimes referred to as **SAN certificates**). In the example below, we can see that the certificate actually covers eleven different SSL.com subdomains:

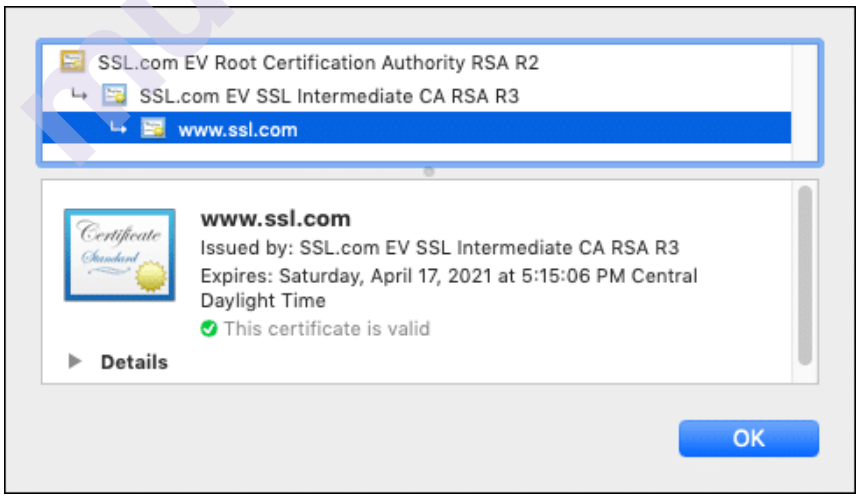
Extension	Subject Alternative Name (2.5.29.17)
Critical	NO
DNS Name	www.ssl.com
DNS Name	answers.ssl.com
DNS Name	faq.ssl.com
DNS Name	info.ssl.com
DNS Name	links.ssl.com
DNS Name	reseller.ssl.com
DNS Name	secure.ssl.com
DNS Name	ssl.com
DNS Name	support.ssl.com
DNS Name	sws.ssl.com
DNS Name	tools.ssl.com

- The **Fingerprints** shown below the certificate information in Chrome are not part of the certificate itself, but are independently calculated hashes that can be used to uniquely identify a certificate.

Fingerprints	
SHA-256	79 E0 E2 8E ED C9 A9 52 D3 6B 41 3B A9 F9 09 DD 60 70 E5 A7 C9 05 B1 67 A8 6C C6 5E 57 C0 F7 A7
SHA-1	CB A9 CA 35 60 64 6A D3 47 23 E3 AD DA C6 2B 1D D1 A4 0A 52

Certificate Chains

For both administrative and security-related reasons, X.509 certificates are typically combined into **chains** for validation. As shown in the screenshot from Google Chrome below, the SSL/TLS certificate for www.ssl.com is signed by one of SSL.com's intermediate certificates, `SSL.com EV SSL Intermediate CA RSA R3`. In turn, the intermediate certificate is signed by SSL.com's EV RSA root:



For publicly trusted websites, the web server will provide its own **end-entity** certificate, plus any intermediates required for validation. The root CA certificate with its public key will be included in the end user's operating system and/or browser application, resulting in a complete **chain of trust**.

Revocation

X.509 certificates that must be invalidated before their **Not Valid After** date may be **revoked**. As mentioned above, RFC 5280 profiles certificate revocation lists (CRLs), time-stamped lists of revoked certificates that can be queried by browsers and other client software.

7.5.3 Public-Key Infrastructure

The most distinct feature of Public Key Infrastructure (PKI) is that it uses a pair of keys to achieve the underlying security service. The key pair comprises of private key and public key.

Since the public keys are in open domain, they are likely to be abused. It is, thus, necessary to establish and maintain some kind of trusted infrastructure to manage these keys.

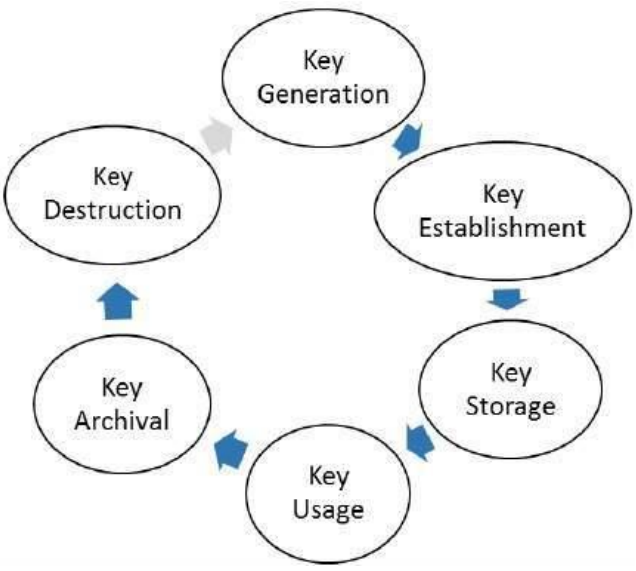
Key Management

It goes without saying that the security of any cryptosystem depends upon how securely its keys are managed. Without secure procedures for the handling of cryptographic keys, the benefits of the use of strong cryptographic schemes are potentially lost.

It is observed that cryptographic schemes are rarely compromised through weaknesses in their design. However, they are often compromised through poor key management.

There are some important aspects of key management which are as follows −

- Cryptographic keys are nothing but special pieces of data. Key management refers to the secure administration of cryptographic keys.
- Key management deals with entire key lifecycle as depicted in the following illustration −



- There are two specific requirements of key management for public key cryptography.
    - **Secrecy of private keys.** Throughout the key lifecycle, secret keys must remain secret from all parties except those who are owner and are authorized to use them.
    - **Assurance of public keys.** In public key cryptography, the public keys are in open domain and seen as public pieces of data. By default there are no assurances of whether a public key is correct, with whom it can be associated, or what it can be used for. Thus key management of public keys needs to focus much more explicitly on assurance of purpose of public keys.

The most crucial requirement of 'assurance of public key' can be achieved through the public-key infrastructure (PKI), a key management systems for supporting public-key cryptography.

Public Key Infrastructure (PKI)

PKI provides assurance of public key. It provides the identification of public keys and their distribution. An anatomy of PKI comprises of the following components.

- Public Key Certificate, commonly referred to as 'digital certificate'.
- Private Key tokens.
- Certification Authority.
- Registration Authority.
- Certificate Management System.

Digital Certificate

For analogy, a certificate can be considered as the ID card issued to the person. People use ID cards such as a driver's license, passport to prove their identity. A digital certificate does the same basic thing in the electronic world, but with one difference.

Digital Certificates are not only issued to people but they can be issued to computers, software packages or anything else that need to prove the identity in the electronic world.

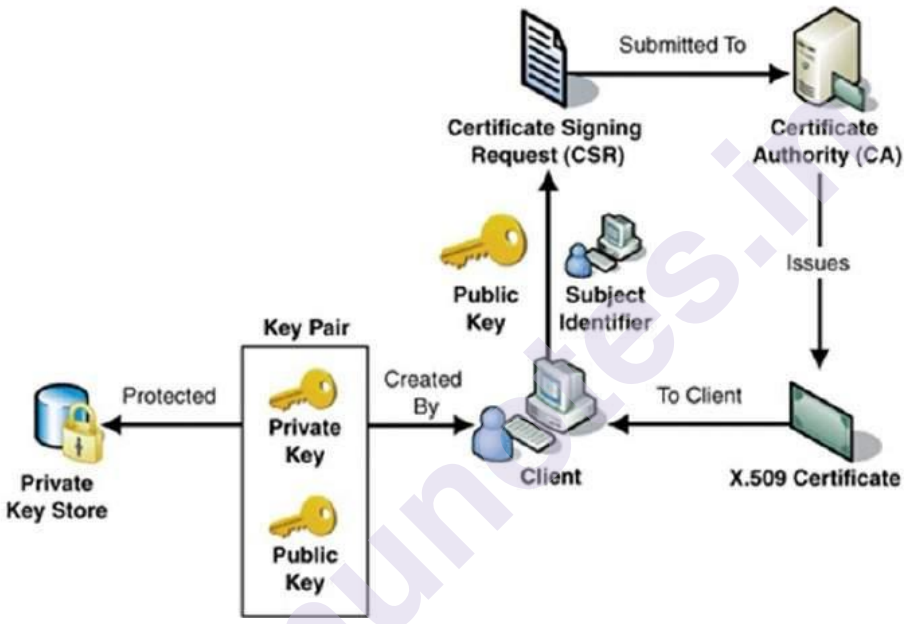
- Digital certificates are based on the ITU standard X.509 which defines a standard certificate format for public key certificates and certification validation. Hence digital certificates are sometimes also referred to as X.509 certificates.

  Public key pertaining to the user client is stored in digital certificates by The Certification Authority (CA) along with other relevant

information such as client information, expiration date, usage, issuer etc.

- CA digitally signs this entire information and includes digital signature in the certificate.
- Anyone who needs the assurance about the public key and associated information of client, he carries out the signature validation process using CA's public key. Successful validation assures that the public key given in the certificate belongs to the person whose details are given in the certificate.

  The process of obtaining Digital Certificate by a person/entity is depicted in the following illustration.



As shown in the illustration, the CA accepts the application from a client to certify his public key. The CA, after duly verifying identity of client, issues a digital certificate to that client.

Certifying Authority (CA)

As discussed above, the CA issues certificate to a client and assist other users to verify the certificate. The CA takes responsibility for identifying correctly the identity of the client asking for a certificate to be issued, and ensures that the information contained within the certificate is correct and digitally signs it.

Key Functions of CA

The key functions of a CA are as follows −

- **Generating key pairs** − The CA may generate a key pair independently or jointly with the client.

- **Issuing digital certificates** − The CA could be thought of as the PKI equivalent of a passport agency − the CA issues a certificate after client provides the credentials to confirm his identity. The CA then signs the certificate to prevent modification of the details contained in the certificate.

- **Publishing Certificates** − The CA need to publish certificates so that users can find them. There are two ways of achieving this. One is to publish certificates in the equivalent of an electronic telephone directory. The other is to send your certificate out to those people you think might need it by one means or another.

- **Verifying Certificates** − The CA makes its public key available in environment to assist verification of his signature on clients' digital certificate.

- **Revocation of Certificates** − At times, CA revokes the certificate issued due to some reason such as compromise of private key by user or loss of trust in the client. After revocation, CA maintains the list of all revoked certificate that is available to the environment.

Classes of Certificates

There are four typical classes of certificate −

- **Class 1** − These certificates can be easily acquired by supplying an email address.

- **Class 2** − These certificates require additional personal information to be supplied.

- **Class 3** − These certificates can only be purchased after checks have been made about the requestor's identity.

- **Class 4** − They may be used by governments and financial organizations needing very high levels of trust.

Registration Authority (RA)

CA may use a third-party Registration Authority (RA) to perform the necessary checks on the person or company requesting the certificate to confirm their identity. The RA may appear to the client as a CA, but they do not actually sign the certificate that is issued.

Certificate Management System (CMS)

It is the management system through which certificates are published, temporarily or permanently suspended, renewed, or revoked. Certificate management systems do not normally delete certificates because it may be necessary to prove their status at a point in time, perhaps for legal reasons. A CA along with associated RA runs certificate management systems to be able to track their responsibilities and liabilities.

Private Key Tokens

While the public key of a client is stored on the certificate, the associated secret private key can be stored on the key owner's computer. This method is generally not adopted. If an attacker gains access to the computer, he can easily gain access to private key. For this reason, a private key is stored on secure removable storage token access to which is protected through a password.

Different vendors often use different and sometimes proprietary storage formats for storing keys. For example, Entrust uses the proprietary .epf format, while Verisign, GlobalSign, and Baltimore use the standard .p12 format.

Hierarchy of CA

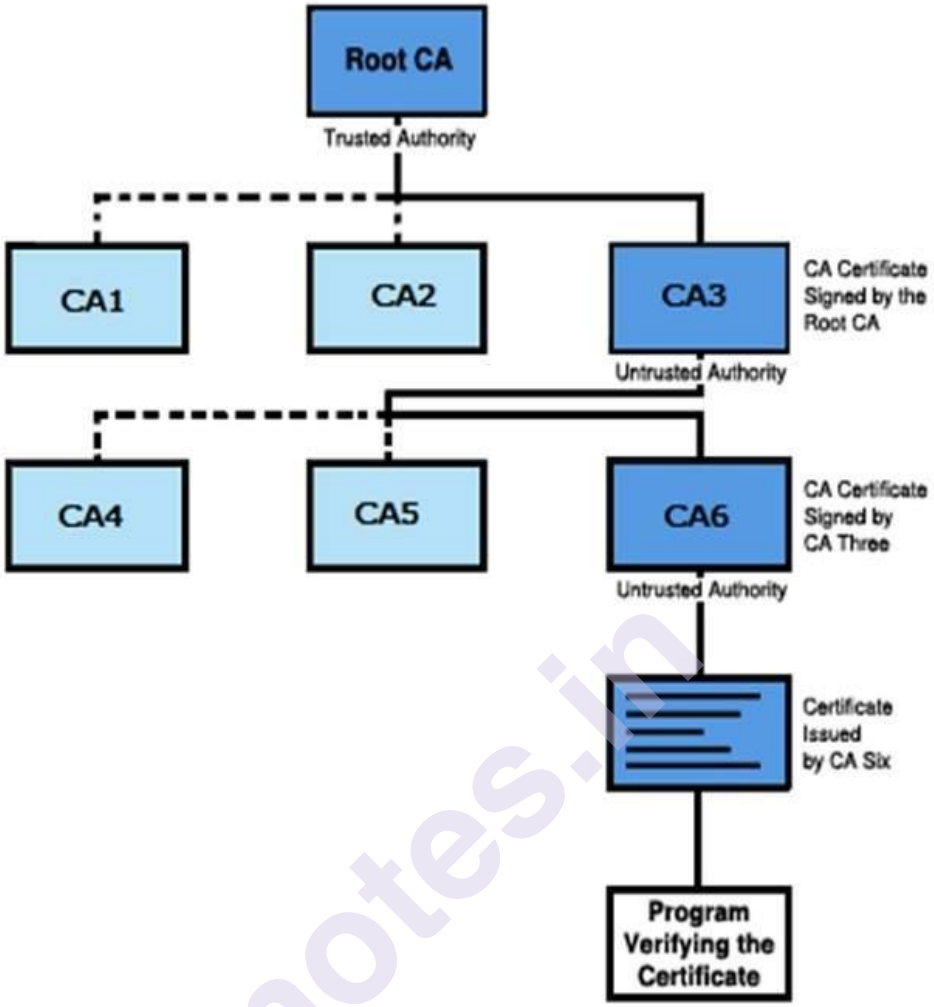
With vast networks and requirements of global communications, it is practically not feasible to have only one trusted CA from whom all users obtain their certificates. Secondly, availability of only one CA may lead to difficulties if CA is compromised.

In such case, the hierarchical certification model is of interest since it allows public key certificates to be used in environments where two communicating parties do not have trust relationships with the same CA.

- The root CA is at the top of the CA hierarchy and the root CA's certificate is a self-signed certificate.
- The CAs, which are directly subordinate to the root CA (For example, CA1 and CA2) have CA certificates that are signed by the root CA.
- The CAs under the subordinate CAs in the hierarchy (For example, CA5 and CA6) have their CA certificates signed by the higher-level subordinate CAs.

Certificate authority (CA) hierarchies are reflected in certificate chains. A certificate chain traces a path of certificates from a branch in the hierarchy to the root of the hierarchy.

The following illustration shows a CA hierarchy with a certificate chain leading from an entity certificate through two subordinate CA certificates (CA6 and CA3) to the CA certificate for the root CA.



Verifying a certificate chain is the process of ensuring that a specific certificate chain is valid, correctly signed, and trustworthy. The following procedure verifies a certificate chain, beginning with the certificate that is presented for authentication −

- A client whose authenticity is being verified supplies his certificate, generally along with the chain of certificates up to Root CA.
- Verifier takes the certificate and validates by using public key of issuer. The issuer's public key is found in the issuer's certificate which is in the chain next to client's certificate.
- Now if the higher CA who has signed the issuer's certificate, is trusted by the verifier, verification is successful and stops here.
- Else, the issuer's certificate is verified in a similar manner as done for client in above steps. This process continues till either trusted CA is found in between or else it continues till Root CA.

7.6 SUMMARY

- The Digital Signature is a technique which is used to validate the authenticity and integrity of the message.
- Message authentication protects two parties who exchange messages from any third party. However, it does not protect the two parties against each other. Several forms of dispute between the two are possible.
- Authentication Protocols are PAP - Password Authentication Protocol, CHAP - Challenge-handshake authentication protocol, EAP - Extensible Authentication Protocol.
- The DSS uses an algorithm that is designed to provide only the digital signature function. Unlike RSA, it cannot be used for encryption or key exchange. Nevertheless, it is a public-key technique
- Kerberos has made the internet and its denizens more secure, and enables users to do more work on the Internet and in the office without compromising safety.
- X.509 is a standard format for public key certificates, digital documents that securely associate cryptographic key pairs with identities such as websites, individuals, or organizations.
- The most distinct feature of Public Key Infrastructure (PKI) is that it uses a pair of keys to achieve the underlying security service. The key pair comprises of private key and public key.
- PKI provides assurance of public key. It provides the identification of public keys and their distribution
- For analogy, a certificate can be considered as the ID card issued to the person. People use ID cards such as a driver's license, passport to prove their identity. A digital certificate does the same basic thing in the electronic world, but with one difference.
- Digital Certificates are not only issued to people but they can be issued to computers, software packages or anything else that need to prove the identity in the electronic world.

7.7 QUESTIONS

1. Write a short note on Digital Signature.
2. What are the aspects to achieve Digital Signature?
3. What are the Authentication Protocols?
4. Write a short note on PAP - Password Authentication Protocol.

5. How does CHAP - Challenge-handshake authentication protocol works?
6. Write a short note on EAP - Extensible Authentication Protocol.
7. Elaborate Digital Signature Standard
8. Explain The Digital Signature Algorithm.
9. Write a short note on Kerberos
10. How X.509 Authentication works?
11. What is Public Key Infrastructure (PKI)?
12. Write a short note on Digital Certificate.

7.8 REFERENCE FOR FURTHER READING

- http://www.facweb.iitkgp.ac.in/~sourav/AuthenticationRequirements.pdf
- https://doubleoctopus.com/security-wiki/protocol/extensible-authentication-protocol/
- https://www.brainkart.com/article/Digital-Signature-Standard_8465/
- https://www.varonis.com/blog/kerberos-authentication-explained/
- https://www.simplilearn.com/what-is-kerberos-article
- https://www.ssl.com/faqs/what-is-an-x-509-certificate/
- https://blog.avast.com/the-importance-of-authentication-avast

❄❄❄❄❄❄❄

8

ELECTRONIC MAIL SECURITY AND IP SECURITY

Unit Structure :

8.1 OBJECTIVES

This chapter would make you understand the following concepts

- Why is Email Security needed?
- How does Pretty Good Privacy ensure Email Security
- S/MIME provides Security
- How to secure transmission of packets at network layer using IP protocol
- How to ensure confidentiality and authentication

8.2 INTRODUCTION

Electronic mail is commonly known as E-mail or Email is a technique of exchanging information(mail) between people using any electronic devices. In 1971, Ray Tomlinson invented networked email. He developed the first system which made it possible to send mail between people/organizations using different devices across the Internet, using the @ symbol which links the username with the destination device.

By the mid-1970s, it gained its popularity as email.

Nowadays, Emails are being used by everyone, maybe for professional or personal use which may even carry confidential information.

Hence, it becomes very essential to provide security to emails as well.

Email security ensures us that there is control over the content of any email account.

An email service provider enables efficient email security to protect any email accounts and its data from attackers or hackers and deliver the mail to a valid recipient.

8.3 PRETTY GOOD PRIVACY (PGP)

Prior to the Pretty Good Privacy technique was introduced, the email provider, Internet Provider, hackers, or even the government could view and read your messages.

But later, PGP was developed in the 1990s by Phil Zimmermann, allowing email and any other types of messages to be exchanged securely.

In today's time, PGP has already been standardized into OpenPGP, which enables anyone to write Pretty Good Privacy (PGP) software which is interoperable and compatible with other implementations.

8.3.1 What is PGP?

PGP is a cryptographic technique that allows people to communicate securely online. PGP is an open source and free of cost software package for providing email security.

When you send any mail using PGP technique, the mail content is converted into ciphertext which is in an unreadable format before it passes over the Internet. Its only the recipient who has the key can convert the unreadable text back into the readable format.

PGP also helps in authenticating the identity of the sender and verifies that the message was not modified during the transmission of the message.

OpenPGP.js is one of the world's most widely used OpenPGP libraries and it has been closely audited by the security experts.

It was designed to provide all the four aspects of security, i.e., confidentiality, authentication, integrity, and non-repudiation while sending an email.

It uses a combination of public key encryption and secret key encryption to provide confidentiality.

It uses the technique of digital signature (i.e., combining public key encryption and hashing) to provide authentication, integrity, and non-repudiation.

Hence, we can easily say that the digital signature technique uses one secret key, one hash function, and two private-public key pairs.

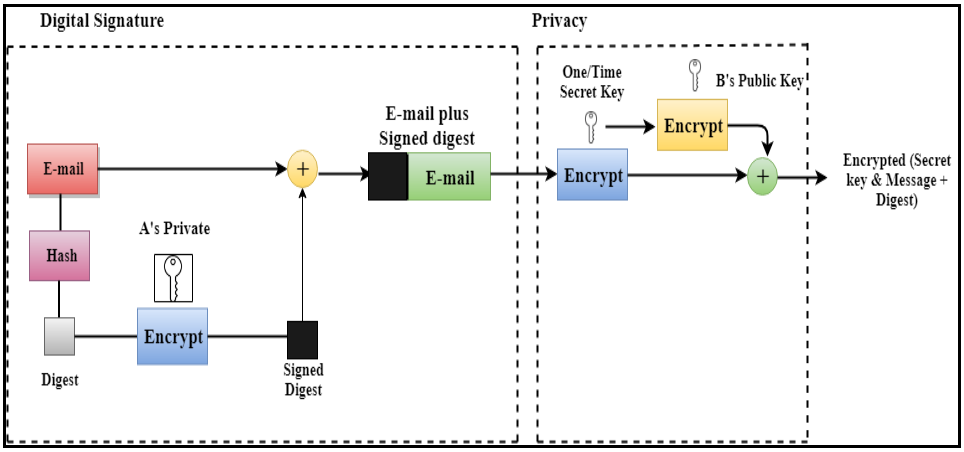
It also provides compression by using EMAIL compatibility using the radix-64 encoding scheme and the ZIP algorithm.

8.3.2 How does PGP work?

Initially, PGP was very difficult to be used as it required additional software applications with an email provider or a client. We also had to manually create encryption keys and exchange them with our contacts. Using ProtonMail which is a library, PGP is built and runs automatically and privately to the users. When you compose any mail to another ProtonMail user and click the send button, the message encryption and signature techniques are applied automatically without any human interference. You don't have to do anything at all, nor do we need any specialized technical knowledge or technical person. ProtonMail has made PGP encryption easy, accessible, and convenient to everyone.

The steps taken by PGP to create secure e-mail are as follows:

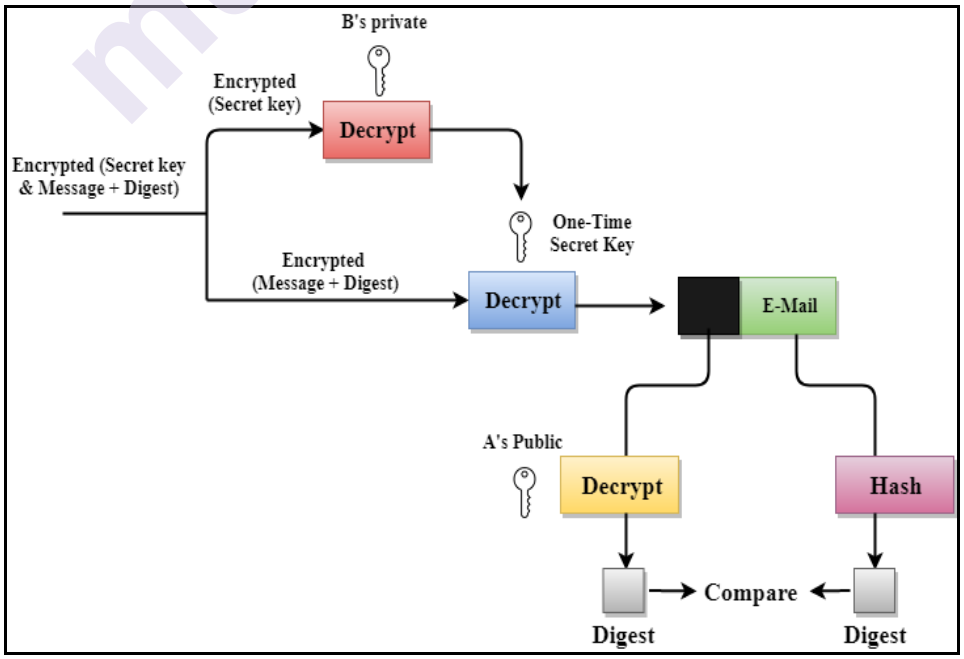
i) PGP implementation at the Sender's side



- A hashing function is used to hash an email message to create a digest.
- Using the sender's private key, the digest is encrypted to form a signed digest, and then this signed digest is added to the original message.
- The signed digest and the original message are together encrypted by using a one-time secret key which is created by the sender.
- Using the receiver's public key, the secret key is encrypted for security reasons.
- The encrypted combination of digest and the message along with encrypted secret key is sent together.

The following steps shows how PGP generates the original message using a combination of three keys and hashing:

ii) PGP implementation at the Receiver's side



- At the receiver's end, the recipient receives the combination of message, digest and the encrypted secret key and message digest.
- By using the receiver's private key, the encrypted secret key is decrypted to get the one-time secret key.
- The secret key is however then used to decrypt the combination of the message and digest.
- By using the sender's public key, the digest is decrypted
- To create a digest, the original message is hashed by using a hashing function.
- Now, both the digests are compared. If both of the digests are equal then it means that all the aspects of security are maintained.

8.3.3 Disadvantages of PGP Encryption

- **Complicated:** Different versions of PGP makes the understanding difficult for the administration.
- **Compatibility issues:** Both the recipient and the sender must have compatible versions of PGP. For example, if a sender encrypts an email message by using PGP with one encryption technique, and the receiver has a different version of PGP then the data cannot be read.
- **Complexity:** Usually, other security schemes use symmetric encryption which uses one key or asymmetric encryption which uses two different keys. But PGP uses a hybrid approach which implements symmetric encryption with two keys. PGP is more complex, and hence, it is less familiar than any other symmetric or asymmetric methods.
- **No Recovery:** Computer administrators face a common problem of losing their passwords. In such a scenario, an administrator must use a special program to restore the passwords. For example, a technician has access to a PC which can be used to restore a password. However, PGP does not provide any such facility for password recovery as the encryption methods used are very strong.

8.4 S/MIME

Isn`t it enough that our Email Server Is Already Encrypted?

Keeping in mind the security concerns, encrypting your email servers with Digital Certificates is a good initiative towards security. This has also prevented outsiders from getting access to your email and mail servers and prevented from viewing confidential data. Still, your emails will be protected to and from the encrypted server but the hackers can still invade into your emailing system and view your messages or have access to them while they are passing through other servers.

8.4.1 What is S/MIME?

S/MIME, or Secure/Multipurpose Internet Mail Extensions, is a type of "end-to-end" encryption technology that allows the sender to encrypt the emails messages.

S/MIME implements asymmetric cryptography to protect our emails from unwanted access by unauthorized persons.

It allows us to digitally sign our emails to verify the sender as an authorised sender of the email message, making it effective against attacks.

8.4.2 How does S/MIME work?

S/MIME uses a pair of mathematically associated keys (private key and a public key) to operate.

It is operationally impossible to find out the private key based on the public key.

Emails are basically encrypted using the recipient's public key.

The email can only be decrypted with the recipient's private key, which is only known to the recipient.

You can be assured that only the intended recipient is able to access the confidential data of your emails unless the private key is compromised.

An S/MIME certificate must be installed on both the email clients, the recipient and the sender.

S/MIME also attaches a digital signature to an email message which verifies that the sender is authorized to send emails.

People usually think, is S/MIME Certificate needed even If we Have an Encrypted Email Server?

There's a difference in the way email server certificates work and S/MIME certificates work.

Rather than the email messages, Email server certificates basically encrypt the email communication channel; meaning it encrypts emails during transmission.

While S/MIME certificates encrypt the email messages.

Hence, hackers can get inside the server and have access to your emails even if you have a digital certificate installed on your email server,

Here's where S/MIME certificates come into picture.

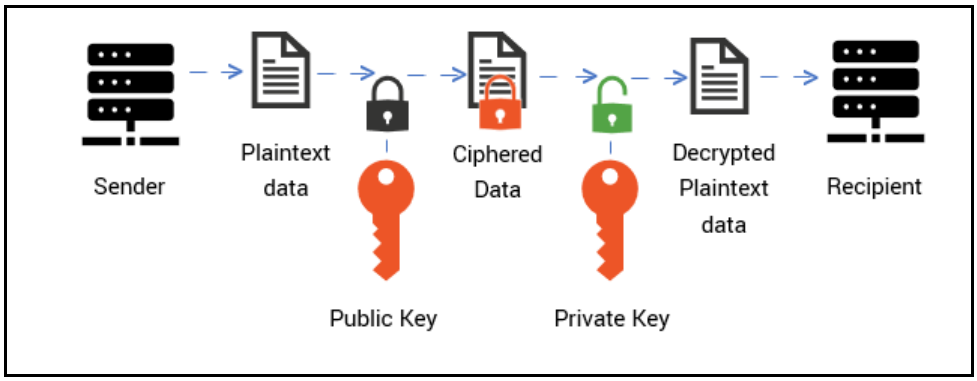


Figure: Public Key encryption (Asymmetric)

Verification of a message is necessary and to do so, an email which is signed with an S/MIME certificate generates a hash which in turn helps to create the digital signature in encrypting the email. The recipient then confirms the digital signature, and the email gets decrypted by the authorised recipient..

Hence, when a person sends an email message, it is encrypted with the public key. This encrypted email is decrypted by the private key which is related to the public key.

In this way, it protects email from unauthorized access and also authenticates the identity of the sender/receiver

8.4.3 Benefits of S/MIME Certificates

The benefits of S/MIME Certificates are as follows:

- Non repudiation: At any point, the sender cannot deny the email and its contents were sent by him/her as a digital signature as a proof that the email message has been received from the signer's email client.
- Protection from email corruption: No attacker can modify or insert any malicious software such as viruses, computer worms, trojan horses, spyware, rootkit, etc. while the email is in transmission.
- Protection against email spoofing: Digital signature prevents the email recipients from email spoofing. Nobody can forge the digital signature of the organization's official staff members.
- Warns the recipients: If anybody tampers the email or digital signature, it immediately sends alerts to the recipients about the security risk. So that the recipients can take preventive measures to protect themselves from becoming victims of any cyberattack.
- Protects business's reputation: If the customers/clients receives an email containing viruses or malicious software from the organization's official email ID then, it can cause the loss of

business's reputation. With the implementation of S/MIME certificates, no email can be tampered to an extent.

- Prevents data leakage: An S/MIME certificate prevents data from data leakage and eavesdropping. The organization's confidential communications with external and internal stakeholders remain secured.

8.5 DIFFERENCE BETWEEN PGP AND S/MIME:

S.NO	PGP	S/MIME
1.	It is basically designed for processing the plain readable text email messages	It is designed to protect email messages as well as multimedia files.
2.	PGP is less expensive.	S/MIME is comparatively more expensive than PGP.
3.	PGP is good for office as well as personal use.	S/MIME is best for industrial use.
4.	PGP is comparatively less efficient	S/MIME is more efficient than PGP.
5.	It is dependent on user key exchange.	S/MIME relies on a valid certificate for key exchange.
6.	PGP is less convenient than S/MIME.	S/MIME is more convenient because of the secure transformation of all the applications.
7.	PGP possibly contains 4096 public keys.	S/MIME contains only 1024 public keys.
8.	PGP is the standard for strong encryption.	S/MIME is also a standard for strong encryption but has few drawbacks.
9.	PGP can also be used in VPNs.	S/MIME cannot be used in VPNs, It is only used in email services.
10.	PGP implements Diffie hellman digital signature.	S/MIME implements Elgamal digital signature.

8.6 IPSECURITY - OVERVIEW

The **IP security (IPSec)** is a secure, standard network protocol suite which helps in communication between two points across the IP network. It provides data confidentiality, integrity, and authentication. It also supports the encryption, decryption and authentication of packets.

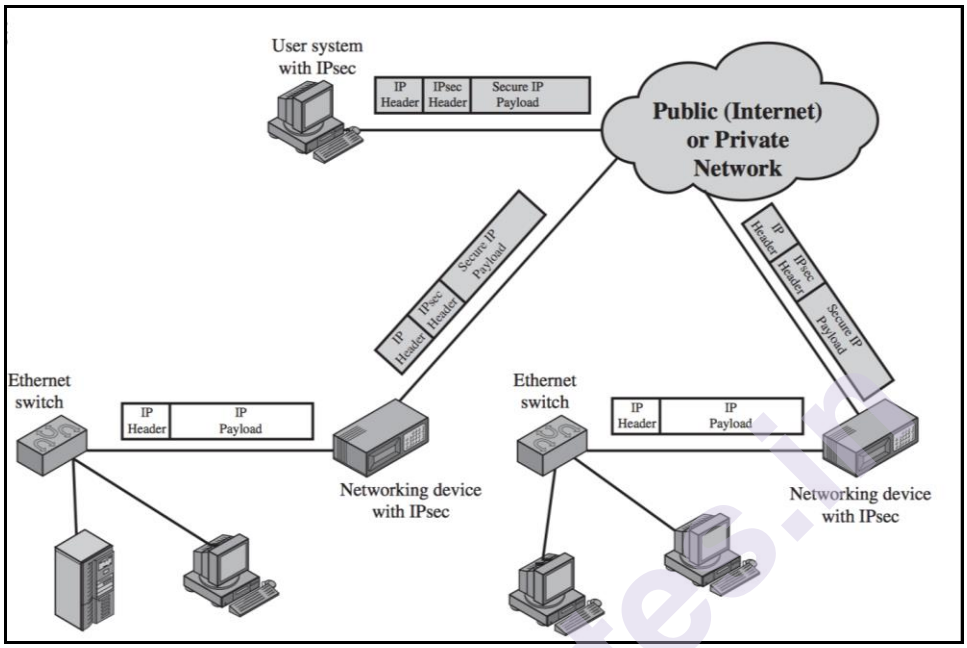


Figure: IP Security Scenario

8.6.1 Benefits of IP Security –

IPsec can be used achieving the following:

- It provides encryption of application layer data.
- It provides security for switches/routers sending data across the internet.
- It also provides authentication without encryption, such as to authenticate that the data originated from a known or a valid sender.
- It protects network data by setting circuits using IPsec tunneling through which data can be encrypted and sent between the two endpoints, as it's done in Virtual Private Network(VPN) connection.

8.6.2 Components of IP Security –

The components of IP Security is as follows:

1. **Authentication Header (AH) –** It provides data authentication, integrity, and anti replay but it does not provide encryption. The anti replay technique protects against unauthorized transmission of data packets. It does not protect the confidentiality of data.

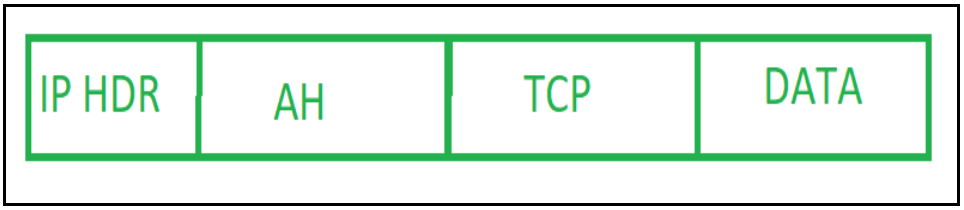


Figure: Authentication Header (AH)

2. **Encapsulating Security Payload (ESP) –** It provides encryption, authentication, data integrity, and anti replay. It also provides authentication and encryption for packet payload.

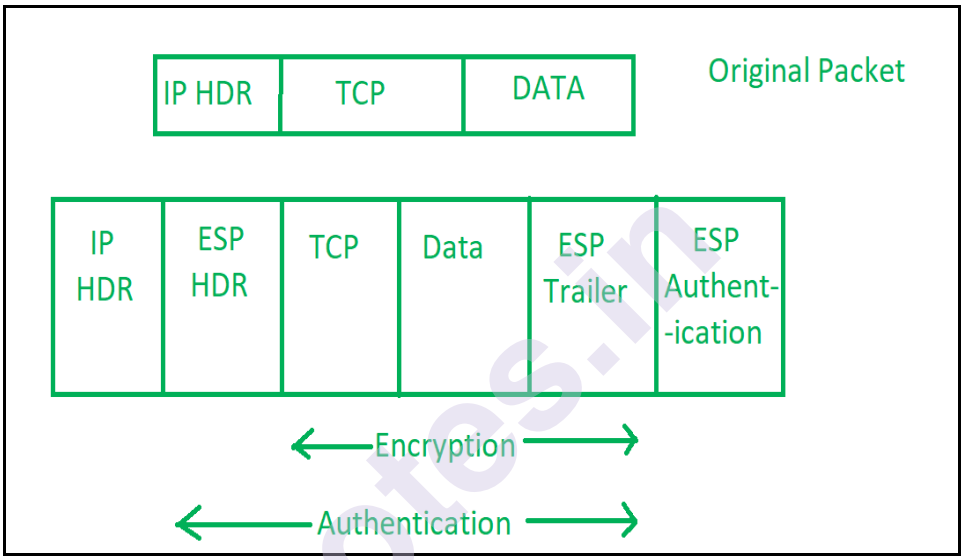


Figure: Encapsulating Security Payload (ESP)

3. **Internet Key Exchange (IKE) –** It's a network layer security protocol designed to dynamically exchange encryption keys and support Security Association (SA) between 2 endpoints. The ISAKMP (Internet Security Association and Key Management Protocol) not only supports key exchange but also authentication of packets.

   Internet Key Exchange (IKE) provides a framework for implementing algorithms such as MD5, SHA and also protects message content.

8.6.3 Working of IP Security –

1. The host (sender) checks if the packet has to be transmitted using IPsec or not depending on the contents in the packet. These packet traffic sets the security policy for themselves by applying appropriate encryption. The incoming packets are also examined by the host whether they are properly encrypted or not.

2. The IKE(Internet Key Exchange) Phase 1 begins in which the 2 participating hosts (using IPsec) authenticate themselves to each other to initiate a secure channel.

3. The channel which was created is now being used to securely deal with the method that the IP Security provides confidentiality of data.

4. The IKE(Internet Key Exchange) Phase 2 is managed over the secure channel in which the two hosts will agree upon the type of cryptography algorithms to be used and also the secret key to be used by those algorithms.

5. The data is then exchanged across the created IPsec encrypted tunnel. These packets are further decrypted and encrypted by the hosts using IPsec SAs.

6. Once the communication between the hosts is done or when the session times out then the IPsec tunnel is terminated.

8.7 IPSEC (IP SECURITY) ARCHITECTURE

IPSec uses two protocols to secure the data or traffic. These protocols are AH (Authentication Header) and ESP (Encapsulation Security Payload).

IPSec Architecture also includes algorithms, protocols, DOI(Domain of Interpretation), and Key Management. All the above stated components are essential to provide the three main services:

- Confidentiality
- Integrity
- Authentication

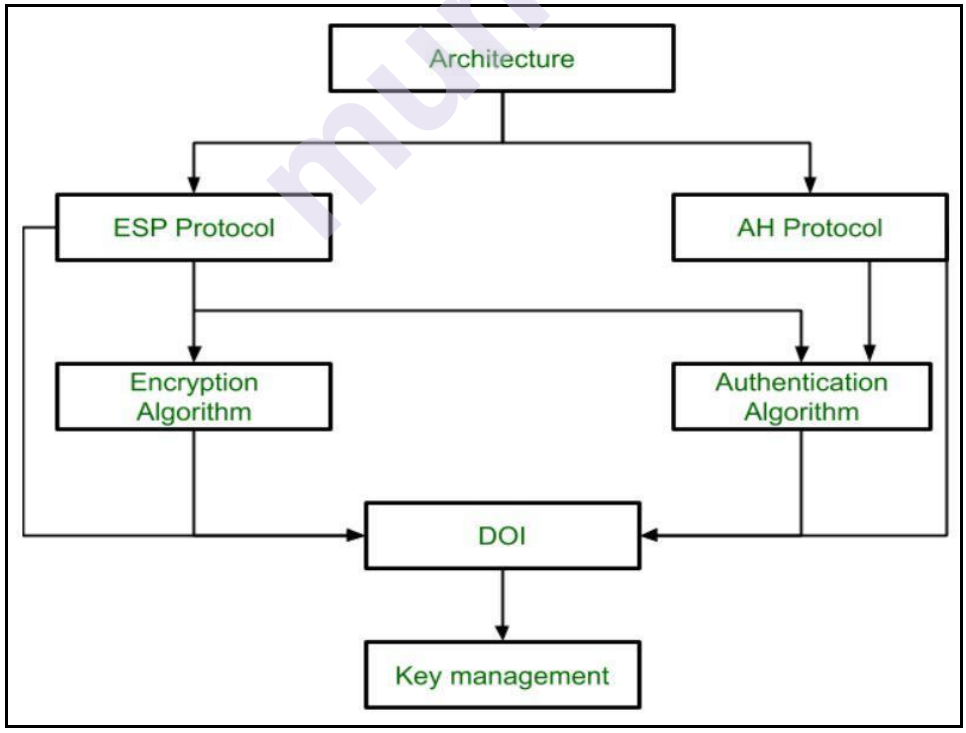


Figure: IP Security Architecture

1. **Architecture:**

   IP Security Architecture includes the concepts, protocols, definitions, algorithms and security requirements of IP Security technology.

2. **ESP Protocol:**

   ESP(Encapsulation Security Payload) provides confidentiality or authentication or both the services.

3. **Encryption algorithm:**

   Encryption algorithm includes documents that specify the encryption algorithms used for Encapsulation Security Payload(ESP).

4. **AH Protocol:**

   AH (Authentication Header) Protocol provides both Integrity and Authentication service.

5. **Authentication Algorithm:**

   The Authentication Algorithm includes documents that specify the authentication algorithm which is used for both AH and also for the authentication of ESP.

6. **DOI (Domain of Interpretation):**

   DOI is an identifier (unique) which supports both AH and ESP protocols. It contains values required for documentation related to each.

7. **Key Management:**

   Key Management contains a document that defines how the keys must be exchanged between sender and receiver.

8.8 AUTHENTICATION HEADER (AH)

Authentication Header (AH) authenticates the origin of data and also verifies whether the data has been modified (integrity) during the transmission between source and destination.

Authentication Header is implemented only in one way: Authentication along with Integrity.

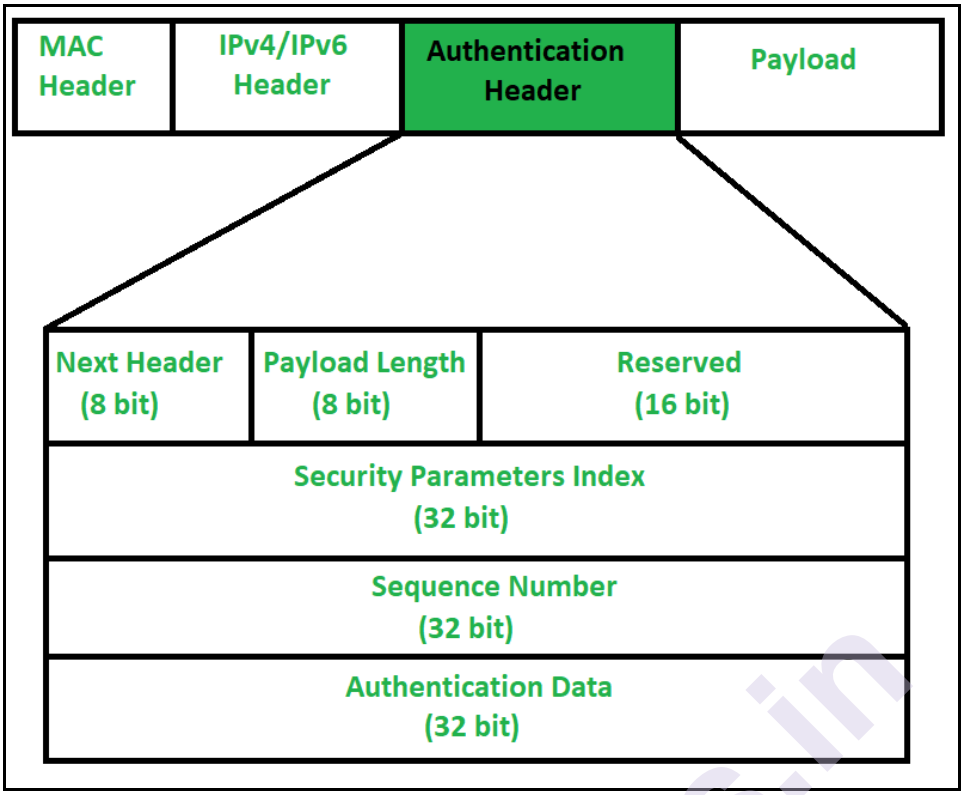


Figure: Authentication Header Packet Format

Next Header – The 8-bit field that defines the type of header which is attached after the Authentication Header. For example, 4 indicates IPv4, 6 indicates TCP.

Payload Length – Payload length is basically the length of the Authentication header and we will use a scaling factor of 4. Whatever may be the size of the header, it has to be divided by 4 and then subtract by 2. We will subtract by 2 because we are not counting the first 8 bytes of the Authentication header.

Say that the payload length is given as X. Then (X+2)*4 will be the original Authentication header length.

Reserved – This 16-bit field is set to "zero" by the sender as this field is reserved for future use.

Security Parameter Index (SPI) – It is a 32-bit field which helps to identify all packets which belong to the present connection. If we are sending data from Source A to Destination B. Both A and B will know the algorithm and key they will use. For Authentication, key and hashing functions will be needed which only the source and destination will know. Secret key between A and B is usually exchanged by using the Diffie Hellman algorithm. So the Hashing algorithm and secret key for Security parameter index of connection has to be finalized.

Sequence Number – This unsigned 32-bit field consists of a counter value that increases by one for each packet being sent. Every packet will

be assigned a sequence number. It will start from 0 till 232 – 1. sequence numbers are used to prevent replay attack.

In Replay attack, the same message is transmitted twice or more without the receiver knowing whether both messages are sent from the same source or not.

Authentication Data Authentication data is a variable length field that contains Integrity Check Value (ICV) for packets during transmission through the internet. Using a secret key and a hashing function, the sender will generate a message digest which will be sent to the receiver. On the other hand, Receive will use the same hashing function and secret key. If both message digest is the same then the receiver will accept data. If the message digest is not the same, then the receiver will be discarded concluding that the message has been modified during transmission.

8.9 ENCAPSULATION SECURITY PAYLOAD (ESP)

Encapsulating Security Payload (ESP) is a set of protocols that enables encrypting and authenticating the packets of data to and fro during transmission using a Virtual Private Network (VPN) to function securely.

Encapsulation Security Payload is implemented in any one of the following ways:

- ESP with only encryption keeping Authentication as optional.
- ESP with Authentication.

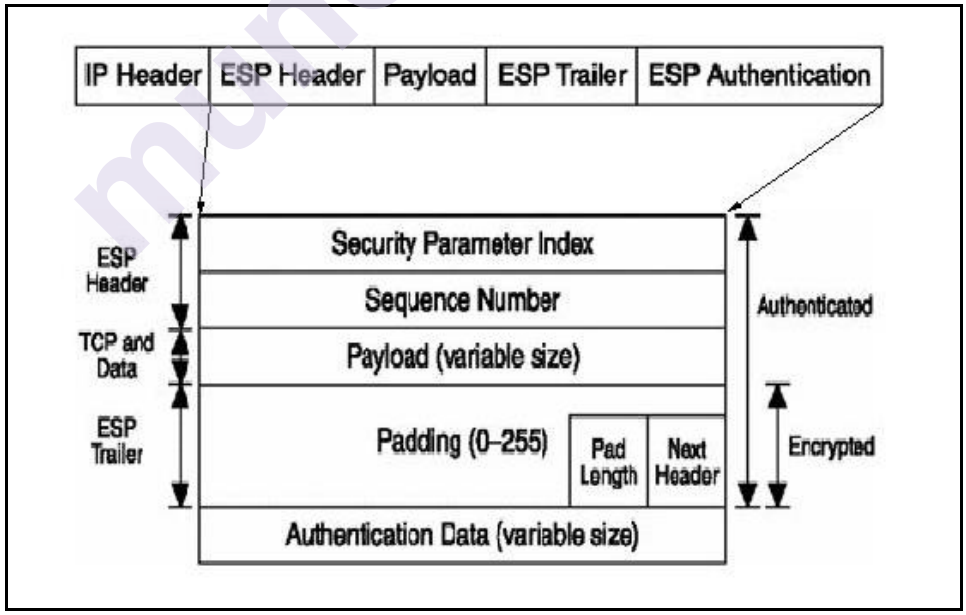


Figure: ESP Packet Format

- **Security Parameter Index(SPI):** This parameter is defined in the Security Association. It gives a unique number to the connection that is established between a Client and a Server.
- **Sequence Number:** A Unique Sequence number is assigned to every packet so that at the receiver end the packets are arranged sequentially to maintain the order and read the message meaningful..
- **Payload Data:** Payload data is the actual message that is to be encrypted using encryption algorithm to achieve confidentiality during transmission of packet.
- **Padding:** These are the extra bits added to the original message in order to maintain confidentiality.
- **Next Header:** Next header defines the type of header which is attached after the Encapsulating Security Payload Header
- **Authentication Data** This field is optional in ESP packet format which is used to verify the identity of the sender.

8.10 COMBINING SECURITY ASSOCIATIONS

8.10.1 Implementing SA

The Security Association (SA) agrees upon a shared security protocol between 2 network endpoints systems to communicate securely.

In the Security Association (SA), both the parties (the sender and the receiver) need to communicate before the actual data exchange. Security association informs what security parameter index, secret key and hashing algorithm are to be used.

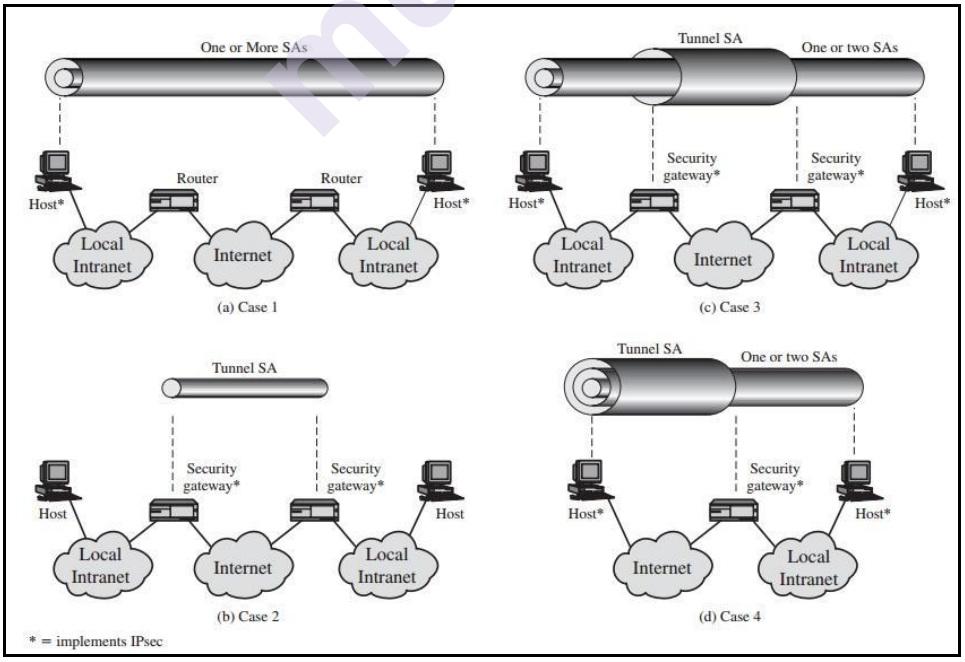


Figure: Various scenarios in implementing Security Association (SA)

The IPsec Architecture document contains four scenarios of combinations of Security Associations that are supposed to be supported by security gateways (e.g. router, firewall) and IPsec hosts (e.g. server or a workstation). These are illustrated in Figure above.

The lower diagram of the above figure indicates the physical connectivity between the elements whereas the upper diagram indicates the logical connectivity via one or more SAs. Each SA can either be ESP or AH.

Case 1. All security is contributed between endpoint systems that implement IPsec. For any two endpoint systems to communicate through an SA, they need to share the secret keys before actual transmission. Among the possible combinations are

A. Authentication Header (AH) in transport mode

B. Encapsulating Security Payload (ESP) in transport mode

C. ESP followed by AH in transport mode (an ESP Security Association inside an AH Security Association)

D. Any one of A, B, or C inside an ESP or AH in tunnel mode

Case 2. Security is provided between gateways (routers, firewalls, etc.) only and IPSec is not implemented to any host. This case demonstrates a simple virtual private network (VPN) support. The security architecture document states that only a single tunnel SA is required in this case. The tunnel can support ESP, ESP with authentication or AH. Since the IPsec services are applied to the entire inner packet, nested tunnels are not needed..

Case 3. This case is built on case 2 by also adding end-to-end security. The gateway-to-gateway tunnelling provides either confidentiality, authentication, or both for all traffic moving between the endpoints. If the gateway-to-gateway tunnelling is ESP, it also provides traffic confidentiality. Individual hosts can apply additional IPsec services whenever required for any given applications or users by way of end to end SAs.

Case 4. It supports a remote host who uses the Internet to reach an organization's firewall and gains access to some private network entities (eg. workstation or server) working behind the firewall. Only tunnel mode is needed to be implemented between the remote host and the firewall. We can use one or two SAs between the local host and the remote host.

8.10.2 Modes of Operation in IPSecurity

The encapsulation mode which is used defines the way in which the original IP data packet is modified. There are basically two encapsulation modes used by ESP and AH:

1. Transport mode

2. Tunnel mode.

1. Transport Mode

The Transport mode encapsulation keeps the original IP header.

Hence, when the transport mode is used, the IP header retains the original source and destination of the packet.

Transport Mode is mostly used in a host-to-host scenario, where the security endpoints and the data endpoints are the same.

Once the datagram is encapsulated in transport mode, the datagram is transported in the same way as the original packet.

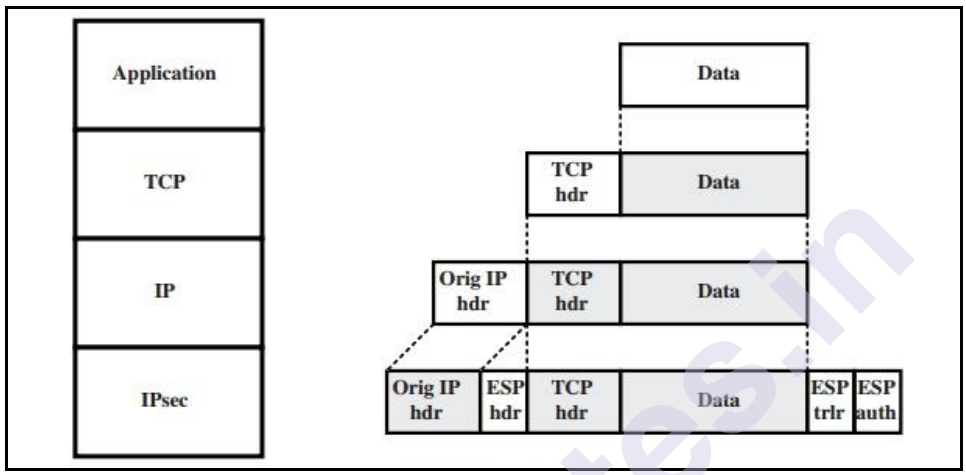


Figure: Transport Mode encapsulation of IP Packet

Transport Mode encapsulation can be implemented using AH or ESP

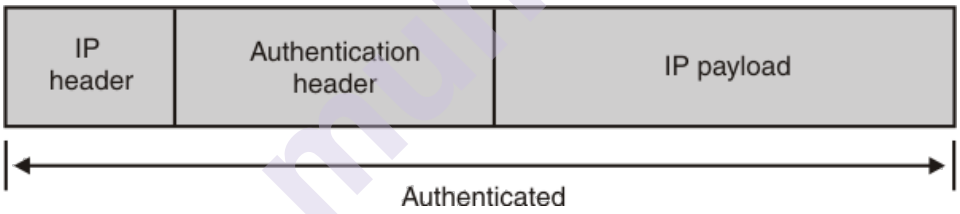


Figure: IP packet encapsulation using AH in transport mode

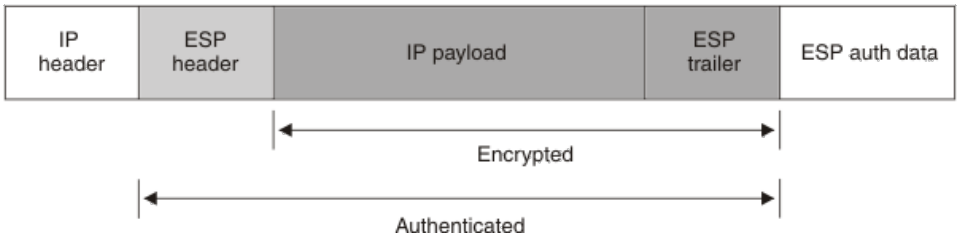


Figure: IP packet encapsulation using ESP in transport mode

2. Tunnel Mode:

Tunnel mode encapsulates the original IP header and payload and creates a new IP header containing the source address and the destination address.

When tunnel mode is being used, the outer IP header (new IP header) specifies the source and destination of the endpoints.

Tunnel Mode encapsulation can be implemented using AH or ESP

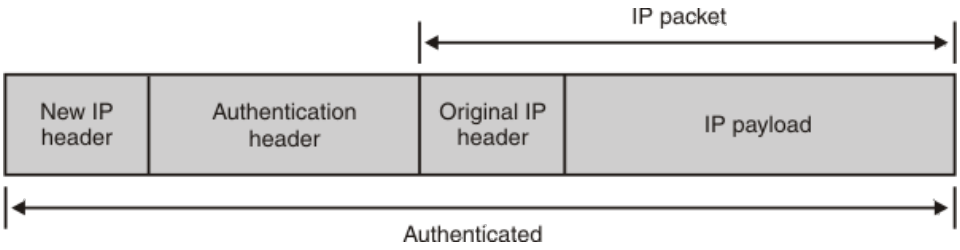


Figure: IP packet encapsulation using AH in tunnel mode

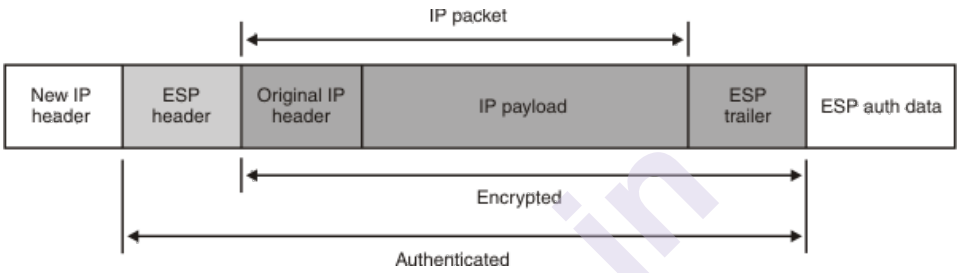


Figure: IP packet encapsulation using ESP in tunnel mode

The choice of using either transport mode or tunnel mode depends on the design of the network and majorly on the logical connections between the end to end systems. Possibly, tunnel mode is needed if any of the IKE (Internet Key Exchange) end is a security gateway that is applying IPSecurity on behalf of another host.

A datagram that is encapsulated using the tunnel mode is transmitted, through the security gateways, assuming that the secure IPSec packet will not transmit through the same network path as that of the original datagram.

8.11 KEY MANAGEMENT

Security associations (SAs) support key generation for authentication of the peer connections and for confidentiality of information thus providing internet security. This is known as key management. The key management of IPSecurity includes determining and distributing secret keys.

IPSec supports two types of key management:

1. Manual
2. Automated

1. **Manual Key Management:**

   A system administrator manually configures the system with its own keys and keys of other communicating systems.

2. **Automated Key Management:**

   This is a more practical approach which helps in creating keys for security associations (SAs) and the keys used to be distributed for secure configuration and communication.

The default automated key management protocol of IPSec include:

- **Oakley Key Determination Protocol -**

It implements the popular key exchange algorithm ie Diffie Hellman algorithm but with added security functionality.

Oakley can be used for major 3 authentication methodologies which are as follows:

- Digital Signature
- Symmetric Key encryption
- Public Key encryption
- **Internet Security Association and Key Management Protocol (ISAKMP)**

It basically provides a framework for key management and contains many message types which can be used to implement key exchange algorithm

It is used for creating, modifying, negotiating, and deleting SAs and its parameters. It also specifies various techniques for key generation and methodologies for authenticating end to end systems.

8.12 UNIT END QUESTIONS

1. Why is Email security needed?
2. How can Pretty Good Privacy ensure Email security?
3. State and explain the disadvantages of PGP
4. How can S/MIME promote Email security?
5. State and explain the benefits of S/MIME Certificate
6. Differentiate between PGP and S/MIME
7. What is IPSecurity? What is the need of IPSecurity?
8. Explain Authentication Header and its packet format?
9. Explain ESP and its packet format?
10. State and explain the components of IPSecurity architecture
11. How does the Security association provide IPSecurity?
12. Explain the two modes of operations implemented by IPSecurity

❄❄❄❄❄❄❄

9

WEB SECURITY AND INTRUSION

Unit Structure :

9.1 OBJECTIVE

This chapter would make you understand the following concepts

- Threats in the network
- How to establish a secure connection between devices?
- Who are intruders?
- Intrusion techniques used by the intruders/hacker
- Techniques used for detecting any unauthorised access to private data

9.2 WEB SECURITY CONSIDERATIONS

9.2.1 Introduction:

Web security is a client/server programme that runs via the Internet and intranets using TCP. Web security, sometimes referred to as "cyber security," is safeguarding data by avoiding, detecting, and reacting to assaults.

The World Wide Web is highly apparent. Many security vulnerabilities are hidden in complex software. Web servers are simple to set up and maintain.

9.2.2 Web Security Threats -

Passive and active assaults are two methods to categorize these risks.

- Passive assault: Passive Eavesdropping on network communication between the browser and the server, as well as getting access to material on a Web site that is meant to be limited, are examples of passive attacks.

- Active assault: Impersonating another user, changing communications in transit between client and server, and manipulating content on a Web site are all examples of active assaults.

Another approach to categories Web security concerns is by the threat's location: the Web server, the Web browser, and the network traffic between the browser and the server. Computer system security includes issues such as server and browser security

9.2.3 Web Traffic Security Approaches -

There are several ways to establish Web security.

The different techniques discussed are comparable in terms of the services they provide and, to some degree, the mechanisms they employ, but they differ in terms of their scope of application and position within the TCP/IP protocol stack.

- IP security is one technique to provide Web security. IPsec has the benefit of being transparent to end users and applications while also providing a general-purpose solution. IPsec also contains a filtering feature, allowing only chosen traffic to be subjected to IPsec processing.

- SSL (Secure Sockets Layer) and TLS (Transport Layer Security) are two Internet standards (TLS). There are two options for implementation at this level. SSL can also be included as part of a specific bundle.

Although Web browsers are simple to use, Web servers are simple to set up and administer, and Web content is becoming more complicated, the underlying software is extremely complex.

Web-based services are frequently used by casual and unskilled users. Such users are often unaware of the security risks they face and lack the tools and knowledge necessary to take effective countermeasures.

Some of the countermeasures to be adopted are as follows:

- **Updated Software -**

  Keeping your software up to date is essential. It is crucial in maintaining the security of your website.

- **SQL Injection -**

  It's a hacker's effort to alter your database. It's simple to include malicious code in your query that may be used to modify your database, such as changing tables, retrieving information, or deleting data.

- **Cross Site Scripting (XSS) -**

  It gives attackers the ability to insert client-side script into web pages. As a result, it is important to ensure that when building a form, you examine the data being submitted and encode or strip away any HTML.

- **Error Messages -**

  You must be cautious in how much information you provide in error messages. If a user is unable to log in, for example, the error message should not indicate which field is incorrect: username or password.

- **Validation of Data -**

  Validation should be done both on the server and on the client.

- **Passwords -**

  It's a good idea to impose password restrictions such as a minimum of eight characters, including upper- and lower-case letters, numbers, and special characters. In the long run, it will aid in the protection of user information.

- **Upload files -**

  The user's file might contain a script that, when run on the server, launches your website.

- **SSL -**

  When sending personal information between a website and a web server or database, it is best to utilize the SSL protocol.

9.3 SECURE SOCKET LAYER AND TRANSPORT LAYER SECURITY

9.3.1 SSL - Introduction

SSL stands for Secure Socket layer. SSL is a technique for establishing a secure connection between two devices linked over the Internet or a private network. SSL is a part of transport layer security.

It was developed by Netscape in 1995 for the purpose of sending data from browser and server.

You can check that the website uses SSL/TLS by checking the URL contains HTTPS rather than HTTP.

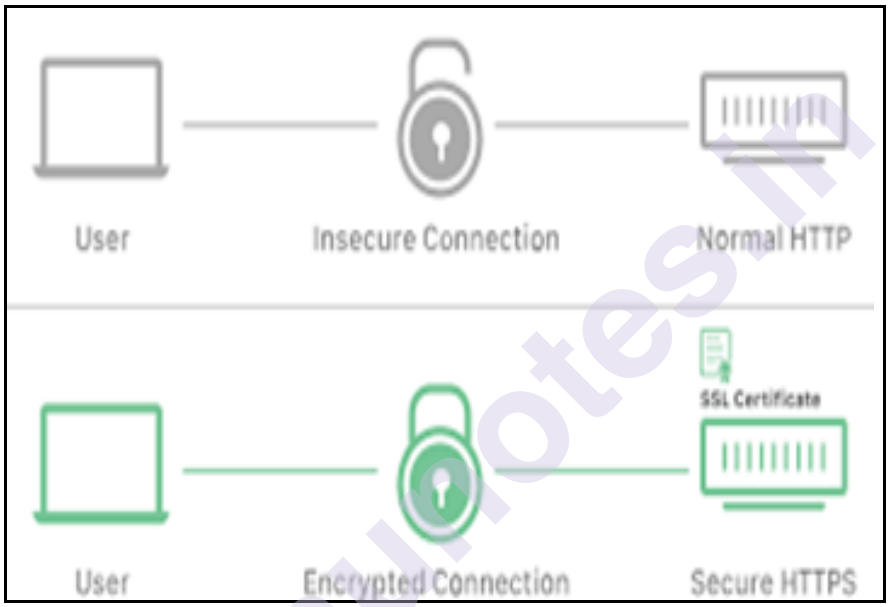


Figure: HTTP vs HTTPS

- SSL encrypts the data before it is sent to the internet i.e., anyone who tries to steal the data will see a ciphertext that is impossible to decode.
- SSL authenticates the devices and then communicates between them by which data integrity is achieved. So, there will be no interference between the user and server.
- SSL is implemented by website by SSL certificate.

9.3.2 SSL Protocol Stack

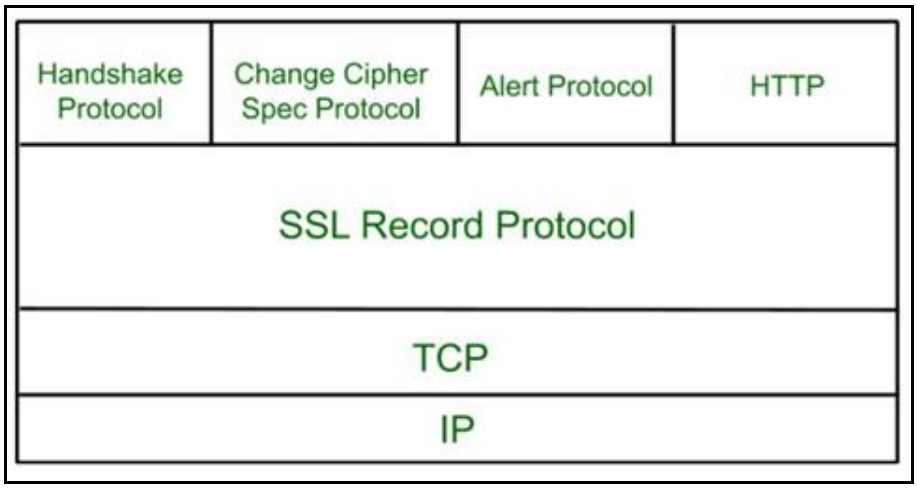


Figure: SSL Protocol Stack

In the SSL protocol, the application data is split into fragments. The fragment is encrypted and then appended by SSL header.

The SSL Protocol Stack three main protocols implemented that are stated as follows:

1. **Handshake protocol:**

   This allows client and server to connect securely before sending the data.

2. **Change cipher protocol:**

   This protocol will not be triggered unless the handshake protocol is complete.

3. **Alert Protocol:**

   Sends alert messages to the receiving party

   After the handshake protocol is completed, the change cipher protocol is executed.

   Basically, it changes the pending state to the current state by sending 1 byte of message.

   Then the Alert protocol is continued and so on.

9.3.3 TLS - Introduction

Transport Layer Securities (TLS) are designed to provide safety on the transport layer. TLS is derived from SSL. Transport Layer Security (TLS) is an internet Engineering task pressure (IETF) general protocol that gives authentication, privacy and information integrity among communicating computer applications.

There are 3 primary components to what the TLS protocol accomplishes:

Encryption, Authentication, and Integrity.

- **Encryption**

  hides the information being transferred from third events.

- **Authentication**

  guarantees that the events replacing information are who they declare to be.

- **Integrity**

  verifies that the statistics have no longer been cast or tampered with.

9.3.4 How does transport Layer security work?

TLS implements customer server handshake mechanism to create an encrypted, authenticated and secure connection

Here are the following steps:

- Communicating devices exchange encryption abilities.
- An authentication method happens by using a digital certificate to assist show the server is the entity it claims to be.
- A consultation key change happens. For the duration of this procedure, customers and servers need to agree on a key to set up the truth that the secure session is certainly between the customer and server -- and no longer something inside the middle trying to hijack the information.

TLS makes use of a public key trade method to set up a shared secret among the communicating devices.

The two handshake methods used by TLS are:

1. Rivest-Shamir-Adleman (RSA) handshake
2. Diffie-Hellman handshake.

Each method results in the same goal of setting up a shared secret between communicating devices so the communication cannot be hijacked. Once the keys are exchanged, statistics transmissions between devices on the encrypted session can start.

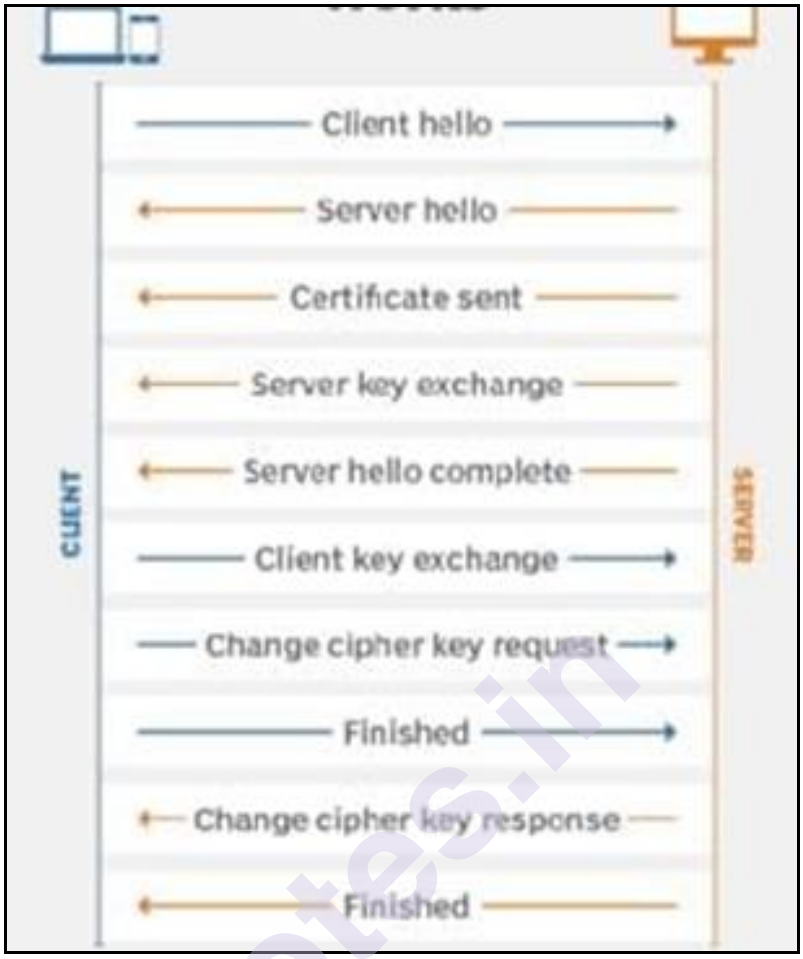


Figure: Working of TLS

TLS is composed of two layers namely

1) TLS record Protocol

2) TLS Handshake Protocol.

TLS Handshake Protocol:

The TLS Handshake Protocol lets the server and purchaser to authenticate each other and to negotiate an encryption algorithm and cryptographic keys earlier than records are exchanged. The main responsibility of the TLS Handshake Protocol is to certify the authentication of the source and key exchange which is required to establish secure connection. whilst establishing a connection, the Handshake Protocol manages the following:

- Cipher suite negotiation.
- Authentication of the server and optionally, the consumer.
- Consultation key data exchange.
- Cipher Suite Negotiation:

The consumer and server make touch and choose the cipher suite with a view to be used in the course of their message change. (Authentication & Encryption mixture)

- **Authentication:**

  In TLS, a server proves its identification to the patron. The patron may additionally want to prove its identification to the server. PKI, the use of public/personal key pairs, is the premise of this authentication. The correct method used for authentication depends on the cipher suite negotiated.

- **Key change:**

  The client and server change random numbers and a special variety referred to as the Pre-grasp secret. Those numbers are mixed with additional facts allowing patrons and servers to create their shared secret, called the grasp secret. The grasp keys used by consumer and server to generate the write MAC mystery, that is the session key used for hashing, and the write key, that's the consultation key used for encryption.

The TLS Handshake Protocol entails the subsequent steps:

- The customer sends a "hey customer" message to the server, along with the customer's random value and supported cipher suites.
- The server responds through sending a "Hey Server" message to the customer, in conjunction with the server's random value.
- For authentication purposes, the server sends it's certificate to the customer and may request a certificate from the customer as well.
- The server in response sends the "Hey Server finished" message to the customer.
- If the server has asked for a certificate from the customer, the customer sends it.
- The customer generates any random Premaster secret and also encrypts it with the general public key from the server's certificate to achieve confidentiality, thus sending the encrypted Premaster mystery to the server.

9.4 SET ELECTRONIC TRANSACTION

Earlier, before the implementation of Secure Electronic Transaction, Transaction via credit card was done. Here the customer used to fill the transaction details via form or phone call, these details were sent to the bank for verification. After a successful verification, the actual transaction may happen in a few hours or few days.

Service providers who used to accept credit card payments had to pay card charge, authorization fee, etc.

Also, there were security and confidentiality issues as merchants were able to view card details of customers and customers sometimes filled wrong card details.

On 1st February 1996, MasterCard and Visa in collaboration with some renowned companies like Microsoft, IBM, Netscape, etc made a single safeguarding technique for online transactions through open networks.

This transaction standard is known as Secure Electronic Transaction (SET).

SET was created to ensure security and integrity of online/electronic transactions. It also ensured that all e-commerce systems must meet 3 integral requirements of security namely, authentication, integrity and non-repudiation.

SET is not a transaction system or payment system. It is a security transaction protocol that is used to create a trust between service provider and service seeker. The trust was created by providing digital signatures like barcodes and one time passwords (OTP), authorization certificates, etc.

1996 was not the 1st time when SET protocol was used. It was introduced way back, but due to disadvantages like installing complex software for SET and strong encryption technology that made the system complex to process, it was not very popular.

Later on modifications were made in SET protocol by overcoming the disadvantages like making the verification process simple, providing authentication to merchants and customers, strong encryption techniques to maintain confidentiality of data, encrypting sensitive data for merchants as well, storing passwords in hashed forms, etc.

All these modifications made SET protocol easy to use, safe and authentic.

SET is successful by providing features mentioned below:

- Provide confidentiality – achieved by encryption of data
- Ensures integrity – achieved by using digital signatures
- Merchant authentication – achieved by use of merchant certificate and digital signatures
- Protection of legitimate parties – by using best security standards
- Interoperability between network and software – achieved by use of protocols and message formats

Requirements in SET:

Mutual authentication (As mentioned above it provides a mutual trust between customer and seller).

- Payment information (PI) and order information (OI) confidentiality.
- Resistive against message modification.
- Provides interoperability.

Entities in SET:

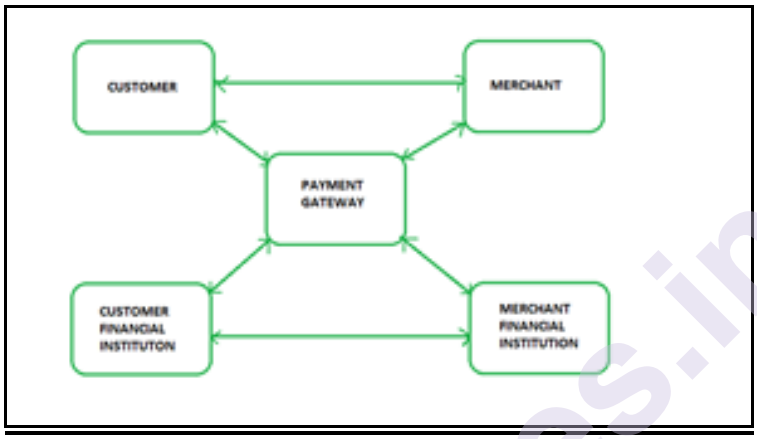


Figure: shows entities and relation between different entities of SET.

- cardholder – customer
- merchant – supplier
- certificate authority – authority that issue certificates like X.509V3
- issuer – institute that provider payment card
- acquirer – provides relation with merchant
- payment gateway – 3rd party

Functionalities of SET:

1. **Authentication:**
   - Merchant authentication: Earlier merchants also used to perform some illegal activities as they used to receive the credit card details of the customers. But after SET these thefts were prevented by encrypting the data on the merchant side and only showing it to financial institutes. SET also allows cardholders to check merchants' relationship with financial institutes.
   - Cardholders authentication: SET also checks if the credit card holder is authorized or not.

Both of these authentications are done using standard X.509V3 certificates.

2. Integrity:

modification of messages with the help of signatures in SET is not supported as a result instead digital signatures are used to protect any unauthorized access.

3. Confidentiality:

SET ensures that messages of merchants and customers are not made publically available. Hence, maintains confidentiality of both. Messages are encrypted using traditional DES.

How does SET protocol work?

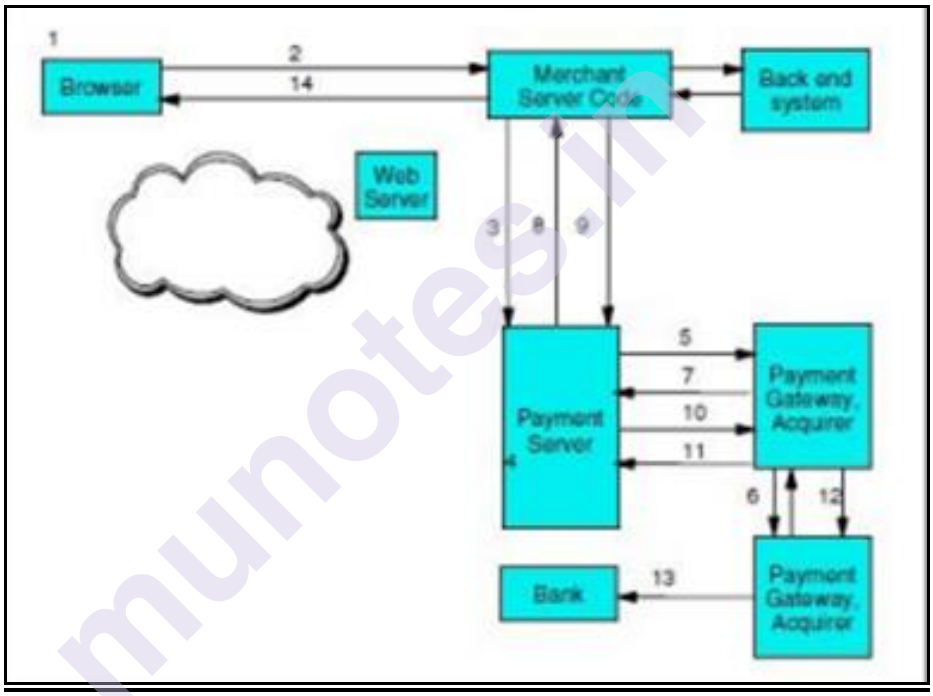


Figure: Working of the SET protocol

The process starts where the merchant customizes a form as a wakeup message for payment initiation. After that payment initiation is done, where a message is received by the customer's browser, the customer must fill in the credit card details there.

After filling the card details, the authentication process of the customer proceeds. Where a verification message is sent on card holders mobile number. Unless and until the customer will not enter the correct OTP the order will not be placed.

After successful verification of the customer and receiving credit card details payment gateway is applied. These credit card details are encrypted before sending for payment gateway through SSL (Secure Socket Layer).

Later these credit card details are checked if they are valid or not. If the card is valid it will check if the card has enough to make a purchase or not.

The merchant will now fulfill the order and after shipping the order he requests for payment. The payment gateway receives the request and transfers the fund from cardholders issuing institute to the merchant's account.

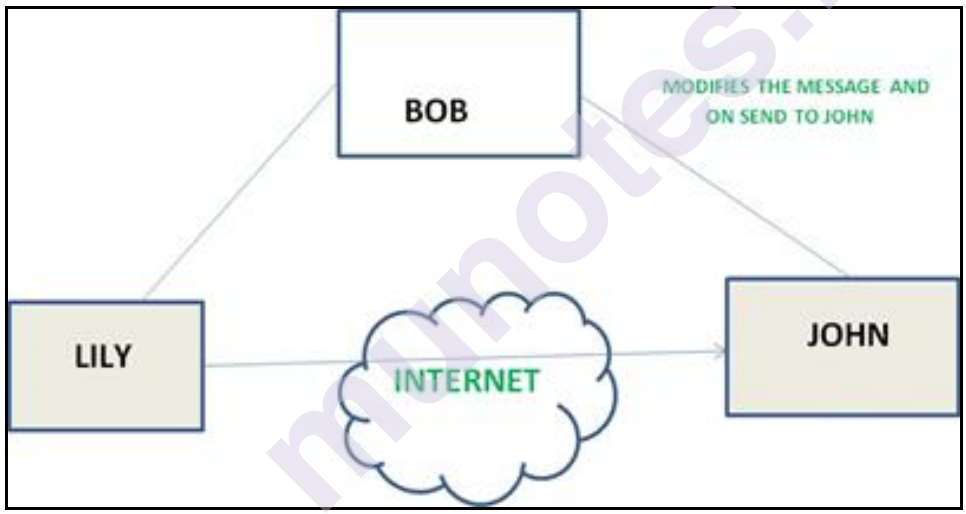
The merchant then sends the confirmation to the cardholder.

The process is completed by updating the amount in cardholders issuing bank after depositing payment to merchants bank.

9.5 INTRUDERS

9.5.1 Introduction:

An intruder is someone who enters an area without a permit to enter. The legal definition of intruder is, Intruders are the attackers who try to breach the protection of a network. They attack the network so as to gain unauthorized access to confidential data.



In the above example, Lily and John are communicating over a web connection. Bob being the intruder is trying to gain access to their communication and modifying the messages before they are sent to the receiver. Thus, we can see there's loss of confidentiality and integrity. We conclude that the intruder can result in loss of three main security goals (CIA triads) i.e., confidentiality, integrity and availability. Intrusions usually involve stealing of confidential information or a network's valuable resource, endangering the network and it's data.

To detect and respond to network intrusions, organizations need to know how intrusions work and study the response systems that are designed with various attack techniques for destruction. Considering the current scenario where moving on digital networks, it can be very difficult to detect unauthorised activities.

9.5.2 Types of Intruders

Based on the type of attack Intruders are classified into three types they are:

1. Masquerader

An unauthorized person penetrates a system to gain access to a legitimate account. it's an attack during which the attacker pretends to be some other person. For e.g. The attacker could send the updated sales figure to Bob and sign the email as Alice.

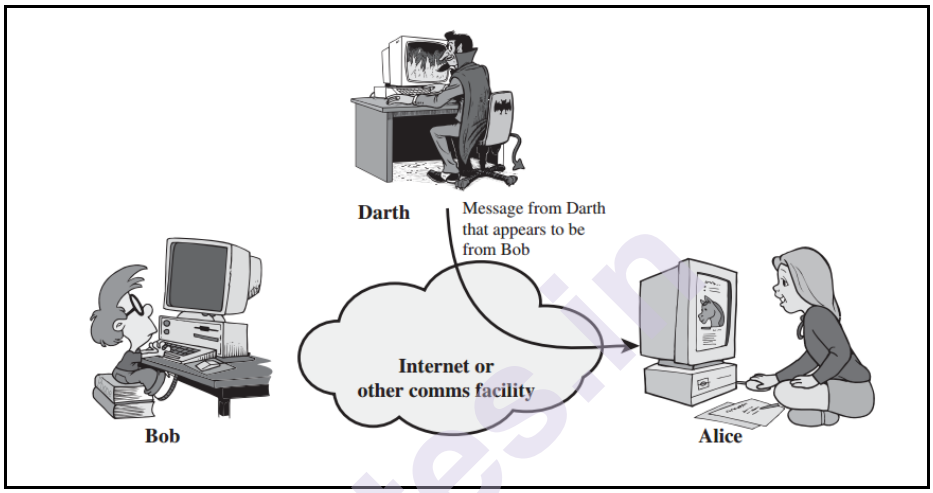


Figure: Masquerade attack

Again, the e-mail will result in a series of packets over the net that are received by Bob. Changing sender's email address is comparatively simple, although protection mechanisms exist like sender policy framework technically it's also easy to vary the source of IP message. Such a technique is known as IP spoofing.

A masquerader is typically an outsider and the above example is a masquerade attack.

2. Misfeasor

Misfeasor is someone who has access to data but misuses it for his/her own privileges. Misfeasors (basically insiders) are considered among the foremost difficult intruders to detect because of their knowledge and authorization within the organization. There are forms of insider threats,

- Malicious insider
- Careless insider
- A mole

Traditional security measures tend to target external threats and aren't always capable of finding an interior threat emanating from inside the organization. So, misfeasors may be a dangerous attack.

3. Clandestine user

Clandestine user is a person who takes administrative control of the system having confidential information and uses this control to escape audit and gain access controls.

Clandestine can be either an outsider or insider of an organisation.

An intruder attack ranges from benign to dangerous. At the benign end, there are many people who only explore the internet and see what is out there.

At the other end, people who try to read confidential data, perform unauthorized modifications to the data, or hamper the information. One of the most common attacks in intrusion is password guessing

Examples of intrusions:

- Defacing an online server
- Guessing and cracking passwords
- Copying a database holding Master card numbers
- Viewing confidential data, such as medical information or payroll records without authorization
- Using unauthorised application for capturing passwords and usernames
- Posing as an executive, calling the assistance desk, resetting the executive's email password, and learning the new password.
- Using an unattended, logged-in workstation without permission

Intruder's behavioral patterns are changing to evade detection and countermeasures. Intrusions violate the security policy of a system.

9.6 INTRUSION TECHNIQUES

- Any unauthorised action on the network is stated as a Network Intrusion. Network breaches include the stealing of valuable network resources and virtually risking network security and/or data security.
- Organizations and their respective cyber security teams need to have a complete hold of what network intrusion is and how intrusions techniques work so as to detect and take collective action against network intrusions.
- Some of the common intrusion techniques are as follows

1. Buffer overflow attacks:

i. Buffer is basically a memory storage space that stores data as it is transmitted from one location to another. A buffer overflows when the amount of data exceeds the actual storage capacity of the memory buffer. Due to which, the application that wants to write data to the buffer overwrites memory in the buffer.

ii. Attackers take advantage of buffer overflow errors by overwriting an application's memory. This alters the program's execution path, leading to damage to files or the disclosure of personal information. This is popularly known as Buffer Overflow Attack.

iii. To obtain access to IT systems, an attacker might insert extra code and send new instructions to the software.

iv. If boundary-checking logic is applied and executable code or malicious strings are recognized before they can be put to the buffer, this attack technique becomes much more difficult to execute.

2. Asymmetric routing:

i. When a packet travels one route to its destination and then returns to the source from a different route, this is known as Asymmetric Routing.

ii. If a network supports asymmetric routing, hackers will frequently use alternative routes to get access to the target system or a network.

iii. This attack approach is ineffective against networks that are not enabled for asymmetric routing.

iv. Because of their low cost, easily available equipment, and considerable damaging potential, asymmetric cyber-attacks are becoming increasingly widespread.

3. CGI Scripts:

i. The Common Gateway Interface (CGI) is a standard network protocol for providing Web interaction between servers and clients.

ii. A CGI software invokes other server applications to construct a user-specific response in accordance to the requirements received. When the CGI software has finished tasks, it sends the result to the web server, which then delivers a response to the client (user).

iii. The CGI program is intelligent enough to detect and provide user-specific information by verifying the authenticity of a

user. Mostly, Dropbox, Google Drive use CGI to provide the user-specific data.

iv. By confirming a user's authenticity, the CGI application is sophisticated enough to detect and display user-specific information. Dropbox most likely uses CGI to offer user-specific data.

v. The software's Intrusion Detection and Prevention System (IDPS) provides fully automated monitoring services to combat CGI attacks

4. Trojans:

i. The word "Trojan" comes from an old Greek legend about a deceiving Trojan horse that led to the destruction of the city of Troy. A Trojan virus is similar to a computer virus as it hides behind seemingly harmless applications or tries to manipulate you into downloading it.

ii. It is a computer program intended to gain access to your confidential data or seems to be a game and destroys data on a hard disk.

iii. Trojan horse seems to be functional software but it damages data or resources once it is executed or installed on the system. Backdoors are created by using Trojans on the systems.

iv. Once the Trojans programs are executed, then it can allow any hackers to steal your confidential data, and gain access to your system.

5. Worms:

i. Worms are pieces of code (software) that replicate themselves by exploiting security and network flaws.

ii. Spreading of worms does not need any human intervention. - The capacity of worms' replication is more. It can transmit thousands of copies of itself thus can result in destructive effects.

iii. Human involvement is not required in the spread of worms. Worms have a higher replication capacity. It has the ability to generate thousands of copies of itself, which can have a catastrophic effect.

iv. Worms propagate without the assistance of any external software or files. A worm is typically transmitted to a computer or network via a link or file sent over the email, chat, or other forms of digital communication.

v. Worms are capable of deleting and altering files. They can also infect a workstation or other device with far more malicious code.

6. **Traffic Flooding**:

   i. Traffic Flood is one of the types of DoS (Denial of Service) attack which aim at web servers.

   ii. It simply creates traffic loads that are too large for network intrusion detection systems to fully examine.

   iii. The attack takes advantage of a feature of the HTTP protocol by simultaneously initiating many connections to attend to a single demand.

   iv. Attackers can sometimes carry out undetected attacks and even induce an undetected "fail-open" state in the resultant congested and chaotic network environment.

9.7 INTRUSION DETECTION

9.7.1 What is an Intrusion Detection System?

In general intrusion detection means finding out an Intruder who causes an intrusion in the system. In the IT field there are Intrusion Detection Systems or Softwares that monitors a system for Malicious activities. These Malicious activities are detected and collected by a Central event management and security information system. There are some IDS's (Intrusion Detection Systems) which can prior detect the intrusion and take necessary actions against them, these types of systems are called Intrusion Prevention System (IPS).

9.7.2 Types of Intrusion Detection techniques:

There are wide variety of IDS which ranging from tiered monitoring system to antivirus but most common types are:

1. Network intrusion detection system (NIDS):

   This is a system which Analyzes incoming network traffic.

2. Host based intrusion detection system (HIDS):

   This is a system that Monitors important system files.

Some subset of IDS types are as follows:

1. **Signature–based:**

   These types of IDS detect threats by looking for specific patterns in a network or system. For example: There are a communication two system and network protocol has defined that the message length should be only of 8 bit while sending, if a malicious threat interferes in that message then that message length will increase, So, because of this IDS will come to know there is an intruder in the system and it will notify the system.

2. Anomaly – based:

This technology is also known as adaptive IDS, in this type of system it uses machine learning and AI model to find the behavior of that malware and detect it.

9.8 UNIT END QUESTIONS

1. What are the web security threats?
2. Explain SSL
3. Explain SSL Protocol Stack
4. How does TLS provide security?
5. How can we adopt secure transactions?
6. What are intruders?
7. State and explain the types of intruders
8. Explain the intrusion techniques
9. Explain the ways of detecting intrusions

❄❄❄❄❄❄❄

10

MALICIOUS SOFTWARE AND FIREWALL

Unit Structure :

10.1 OBJECTIVE

This chapter would make you understand the following concepts

- What are malwares and in detail about virus and its countermeasures
- What are firewalls
- Need of firewall to prevent from security attack
- Which types of firewall suites best in various situation

10.2 WHAT IS MALICIOUS SOFTWARE?

Malware means malicious software. This software helps hackers to gather sensitive information, interrupt users computers operations, and unauthorized access to computers.

It can be in the form of annoying script or code or software.

Viruses, spyware, Trojan Horse, Worms and other malicious programs are examples of malware.

Sometimes, malware can be genuine software which can be created by the company's official website.

10.2.1 Types of malicious software

The malicious software are mainly categorised into two types:

1. **Independent of host program:**
   - Worms
   - Zombies
2. **Need host program:**
   - Trapdoors
   - Logic Bombs
   - Trojan horses
   - Viruses

10.3 VIRUS AND ITS THREATS

10.3.1 What is a virus?

A small piece of software attached to programs is known as virus

It spreads from one machine to another by leaving its infection on route.

Some virus effects are mild which can be ignored while others can damage software, data, files or hardware.

Virus needs human intervention to get spread.

It cannot infect the system unless we execute or open the file or program containing it.

10.3.2 Types of viruses:

- **Record infectors**

  These most often connect to program documents but can infect any file with executable code, consisting of script files or software configuration documents. When the program, script, or configuration is carried out, the virus is executed nicely.

- **Machine or boot-report infectors**

  Machine or boot-document infectors do not always infect a file. They target, instead, positive areas of a difficult disk used solely for

device processes. These regions encompass the boot document, which is a phase of the disk committed to booting the operating system. On a diskette, these infectors attach themselves to the DOS boot zone; on a tough disk, they attach themselves to the master Boot file. Having infected a master Boot document, the virus spreads to the boot sectors of the inserted media.

- **Multipartite viruses**

  Multipartite viruses infect boot information as well as files. With its hybrid nature, a multipartite virus inherits the worst features of each of its parents and consequently is way more contagious and adverse than both.

- **Macro viruses**

  Macro viruses infect macro-enabled files, especially in the Microsoft office suite of programs-extra particularly, Microsoft Word and Excel. Whilst opened, an inflamed document executes a macro robotically, or the consumer does so by accident. The macro inflicts damage after which infects other files on the disk. A macro is hard and fast of executable instructions designed to run in place of a repetitive task. Although macros aren't particular to Microsoft merchandise, it is through Microsoft merchandise that many macro viruses unfold. Macro viruses are the most commonplace kind of viruses, but they do pretty little damage.

- **Stealth viruses**

  Stealth viruses use many strategies to thwart detection. One approach is to redirect the addresses inside an application that point to different programs or gadget facts and have them a factor to the virus file as a substitute. Whilst this system calls for that supplementary software or system data, it truly runs the virus code. This infects the record without absolutely injecting additional code, which could show up as a symptom of virus scanning software programs. Every other, not unusual stealth approach modifies a record but displays its size as it turned into before contamination. Therefore, it nullifies the ability to use the reporting period as an indicator of contamination.

- **Encrypted viruses**

  Encrypted viruses enjoy the advantages of other encrypted fabrics. Initially, encrypted viruses seem no longer as viruses, but as nondescript without sense. However, whilst an inflamed program is accomplished, a small piece of undeniable, unencrypted code decrypts the relaxation of the virus, which then proceeds to do its damage. When, and if, an encrypted virus is detected, it is very tough to research because it is not difficult to reverse engineer just like the unencrypted viruses. This makes it hard to decide the structure of the virus and the perfect scope of its payload.

Encryption is most useful while coupled with a polymorphic approach.

- **Polymorphic viruses**

  Polymorphic viruses try to avoid detection by altering their shape or the encryption strategies. Each time contamination happens, a polymorphic virus modifications its shape, puzzling virus (detection) scanning software. Due to the fact virus scanners use specific "signature" characteristics to pick out viruses, any virus that adjusts its shape affords a formidable new assignment.

10.3.3 Life Cycle of Virus:

There are four stages in a life cycle of virus:

1. Dormant Phase

2. Propagation Phase

3. Triggering Phase

4. Execution Phase

1. Dormant Phase:

- Basically, in this phase the virus is inactive
- The virus gets activated by executing the event
- Examples:
- Date/timestamp events, presence of another file or disk usage reaching some capacity.
- All viruses may not have this phase.

2. Propagation Phase:

- Virus replicates itself and attaches itself to some program.
- The replica may have slight changes as viruses change their forms to avoid detection.
- Every virus infected program contains a copy of the virus which enters itself into a propagation phase.

3. Triggering Phase:

- Virus performs the function it is intended for.
- Many system events cause the triggering phase such as counting how many times the virus replicates itself.

4. Execution Phase:

- Virus executes its function which can be harmless like just a message on the screen or damaging like deletion of information or programs.

10.3.4 Virus countermeasures

1. Antivirus:

Rather than detecting and removing viruses, it is always better to prevent it. It does not allow viruses to enter into the system and also blocks it to make any modifications.

Tries to locate the virus by the type of its infection.

Once detected, it finds out that specific virus which has infected the program.

Once identified, try to remove almost all virus traces from infected areas so that it cannot spread anymore.

A good antivirus should be installed and a\updated regularly Generation of antivirus

There are mainly four generations of antivirus as stated below:

A. 1st Generation

a. st generation antiviruses are also known as simple scanners.

b. Virus signature needed for identification of virus.

c. But these signature specific scanners can scan only known viruses

d. Length of programs and alteration is observed to check for virus attacks.

B. 2nd Generation

a. 2nd generation antivirus are also known as Heuristic scanners,

b. It does not depend on simple virus signatures.

c. Basic concept behind this is to search those blocks of codes which are associated with viruses.

d. Example, such programs can find out the encryption key used for a virus, then decrypt it and eliminate the virus and cleans code.

e. It checks for integrity means if a virus affects the program without altering checksum then integrity check traps that alteration.

C. 3rd Generation

a. 3rd generation antiviruses are also known as activity traps.

b. They reside in memory.

c. Rather than viruses' structure, it searches for viruses based on their action.

d. But maintaining a huge virus signatures database is not needed.

e. But it is needed to find out a small set of actions which identifies the attempted infection and then intervenes it.

D. 4th Generation

a. 4th generation antivirus is also known as full featured protection.

b. It consists of many antivirus techniques combined with other packages.

c. It includes components for activity and scanning traps.

d. It also controls access capability that means restricting virus ability to enter into a system and restricting virus capabilities to alter files in order to pass infection.

e. A wide range of defence strategy is implemented with this generation of antivirus extending the scope of defence to tackle more general purpose measures for computer security.

2. Generic Decryption (GD)

GD facilitates the antivirus program to speedily scan and easily detect most complicated polymorphic viruses.

3. Digital Immune System

- IBM developed the Digital Immune system for virus protection.
- The main aim behind this is growing threats of internet based virus spreads.
- When a new virus enters a network, the digital immune system automatically detects it, analyse it. capture it, protect against it, deletes it and also spreads information about it to other systems running IBM Digital Immune system so that it can easily be detected before its execution.

4. Behaviour Blocking Software

- Behaviour blocking software along with OS observes behaviour of a program in real time for harmful actions.
- Then this software blocks potentially harmful actions before it harms the system.

Observed behaviour includes:

- Attempts to view, modify or delete files.
- Attempt to perform operations which are unrecoverable or formatting of disk drives.
- Modifying the contents or logic of the executable files.
- Alterations in crucial system settings. -Interruption in network communication

5. Updating vaccine softwares like antivirus

Antivirus software should be installed and updated periodically to protect against viruses.

6. Scanning of email attachment files While downloading email attachment files we should

- Be very cautious with email attachments of unidentified sources.
- Not be misled by the appearance of attachment files.
- Be cautious of doubtful files attached to emails even though you received them from known resources.

7. Symptoms of Virus Infection

Must not be ignored Virus symptoms like disappearing files, appearance of unknown icons on screen or taskbar, appearance of unknown codes in programs, unknown emails, etc. are observed on the systems then do not ignore these symptoms and scan computers for viruses and take necessary actions.

8. Downloaded Files should be scanned
9. For Applications utilize Security Functions
10. Security Patches Should be applied
11. Back up the data on a regular basis.

10.4 DDOS

10.4.1 DOS Attack

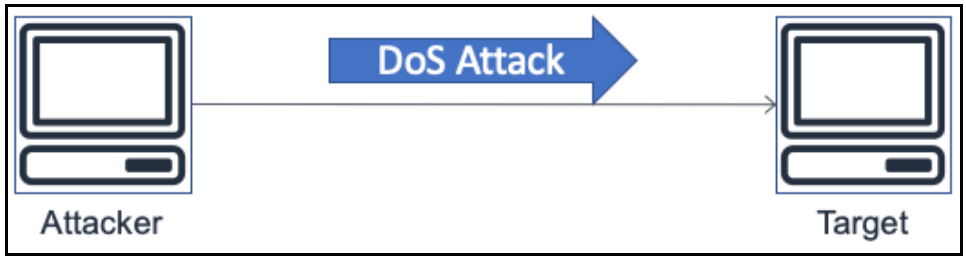


Figure: Denial of Service (DoS) attack

- It stands for Denial of Service.
- The cyber attack in which the attacker makes a machine or network resource unavailable to the intended users by temporarily or deliberately stopping services of a host machine connected to the internet.
- A single machine is used to launch an attack.
- Comparatively less complicated.
- There is no malware involvement.

10.4.2 DDOS

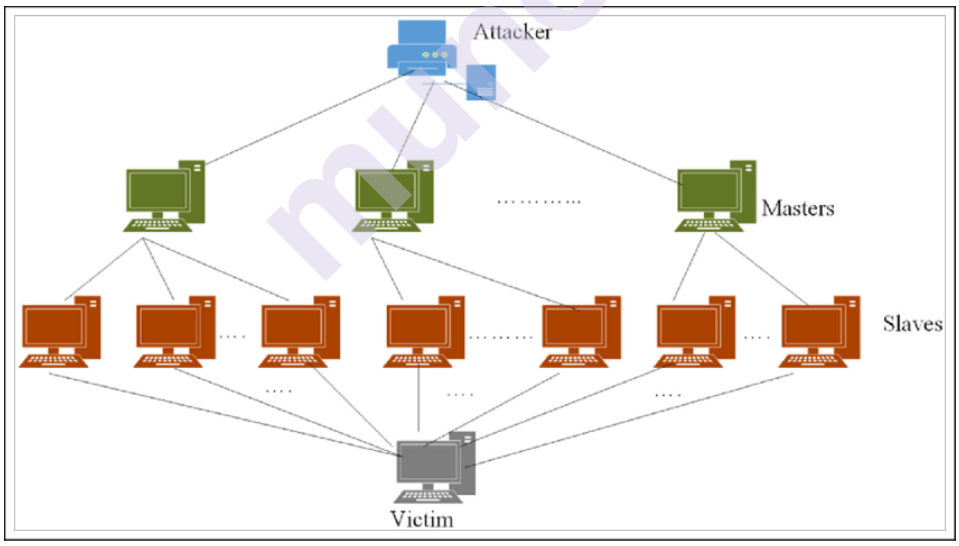


Figure: Distributed Denial of Service(DDoS) attack

- It stands for Distributed Denial of Service.
- A cyber attack in which the incoming traffic floods the victim machine that originates from various different sources.

- The attacker gains access to number of machines and use these machines to attack the victim
- More complicated and difficult to prevent.
- Uses malware to affect multiple machines.

10.5 FIREWALL DESIGN PRINCIPLE

10.5.1 Introduction

A firewall is basically a single point which keeps unauthorized users blocked from the protected network, thus prohibiting potential vulnerable services from leaving or entering the private network, and provides protection from various kinds of cyber attacks, IP spoofing and routing attacks.

A firewall monitors security related events. Audits and alarms are usually implemented on the firewall system.

A firewall is a protection technology that controls outgoing and incoming packets totally based on predefined security rules. A firewall is generally like a barrier between a reliable and an untrusted community, which includes the Internet, Intranet. The devices which provide firewalls for people's safety can be routers which are used for routing. A firewall analyses the packets entering and leaving and filters them so that suspicious and unsecured actions don't cause harm to the system.

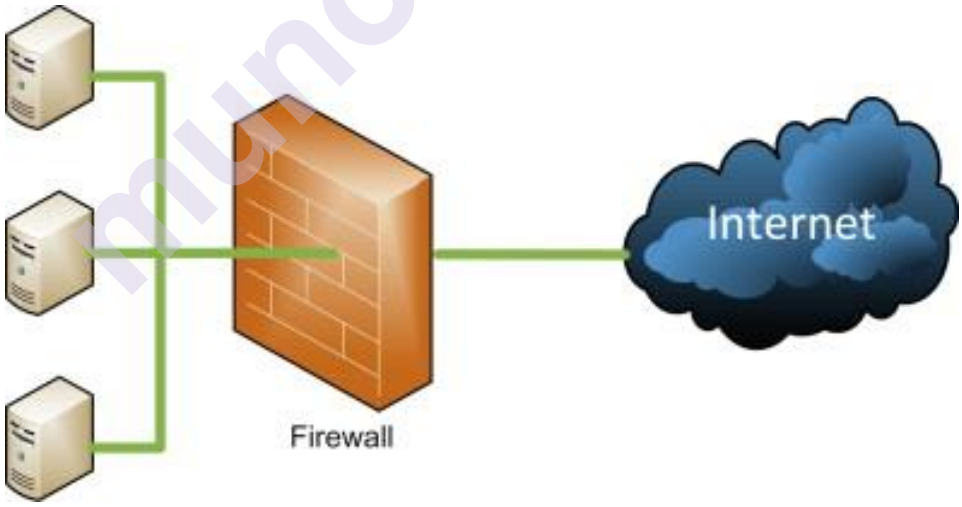


Figure. Firewall protects private network from public network

While internet access provides benefits to the institute/organization, it allows the outside world to reach and interact with private network assets. Thus creating a threat to the organization's assets.

The firewall is basically like a barrier that is inserted between the private network and internet(public network) to establish a monitored link and to build a security wall. The aim of this security wall is to protect the private network from the internet threats and to provide a single secure point

where audit and security can be applied. The firewall may be implemented using a single device or a set of devices that need to cooperate collectively to perform the firewall functionality.

10.5.2 Characteristics of Firewall:

- All traffic passing from outside to inside, and vice versa, has to pass through the firewall. This is possible by blocking all access to the private network except via the firewall.
- Various firewall configurations are possible and need to be done.
- As specified by the local security policy, only authorized traffic will be allowed to pass through the firewall.
- There are various types of firewalls that are used which implement different types of security policies.

10.6 TYPES OF FIREWALL

The various types of firewall are as follows:

1. **Packet filtering router**

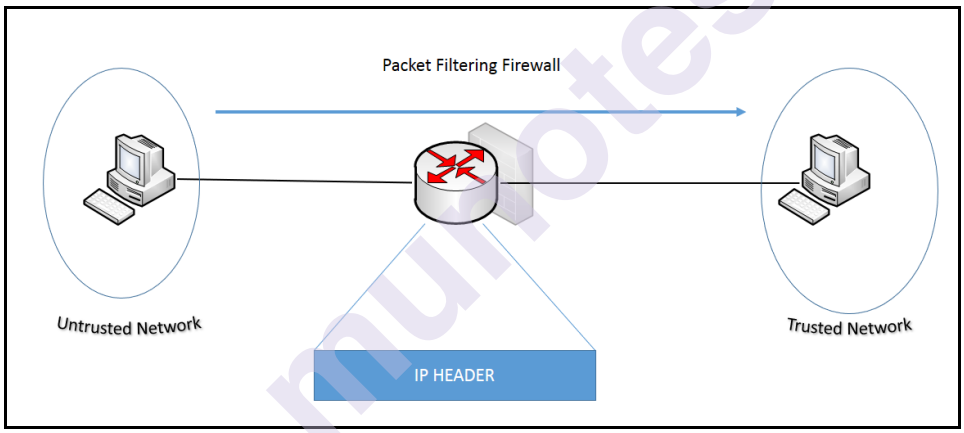


Figure: Packet Filtering Firewall

In packet filtering router technique, the router applies a set of predefined rules to every incoming IP packet and then based on the rules forwards or discards the packet. The router is configured to filter out packets going in and out the private network. Filtering rules defined are based on the information within a network packet:

- Source IP address – It's the IP address of the sender of the IP packet.
- Destination IP address – It's the IP address of the receiver.
- Source and destination transport level address – They are transport level port numbers.

- IP protocol field – It defines the transport protocol through which the packet is received.

The packet filter has a list of rules, based on the matches in the TCP or IP header fields, appropriate action will be taken regarding the packets. If there's any match found to one of the predefined rules, then that rule is invoked to decide whether to discard or forward the packet. If there's no match to any rule, then a predefined default action will be taken.

Two default policies are possible:

- discard: The one which is not permitted is prohibited.
- forward: The one which is not prohibited is permitted.

The discard policy as default is more conservative. Initially everything is blocked, and then services need to be added.

Advantages of packet filter router

- It is very Simple to configure and maintain
- It's very Transparent to users and quick in action

Disadvantages of packet filter firewalls

- Packet filter firewalls cannot prevent attacks from the application layer as they don't examine upper layer data.
- It does not provide advanced level user authentication schemes.
- They are vulnerable to attacks like IP spoofing.

2. Application level gateway

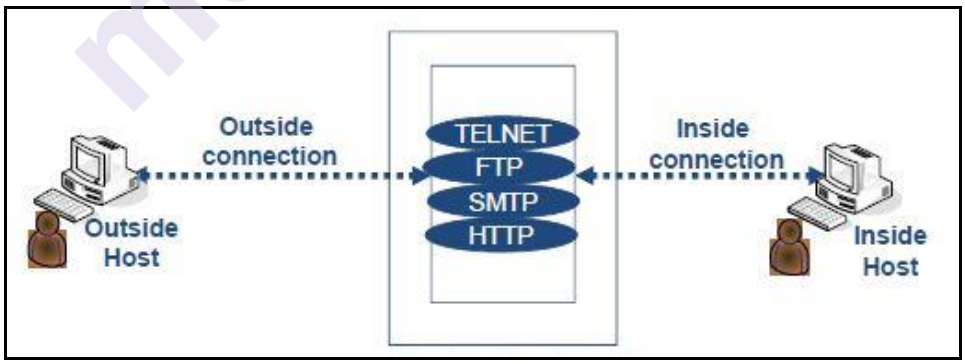


Figure: Application level Gateway Firewall

An Application level gateway is also known as a proxy server which acts as a relay of application level traffic. The user will contact the gateway using a TCP/IP application, such as FTP or Telnet or HTTP, and the gateway in turn will ask the user for the name(identity) of the remote host which it has to access. When the user responds back and provides it with the authentication information and a valid user ID, the gateway will

contact the application on the remote host and then will relay the TCP segments which contain the application data between the two endpoints.

Advantages of Application level gateway

- It is far more secure than packet filter firewall
- It is very easy to log and audit all the incoming traffic at the application level

Disadvantages of Application level gateway

- There's additional processing overhead for each connection.

3. Circuit level gateway

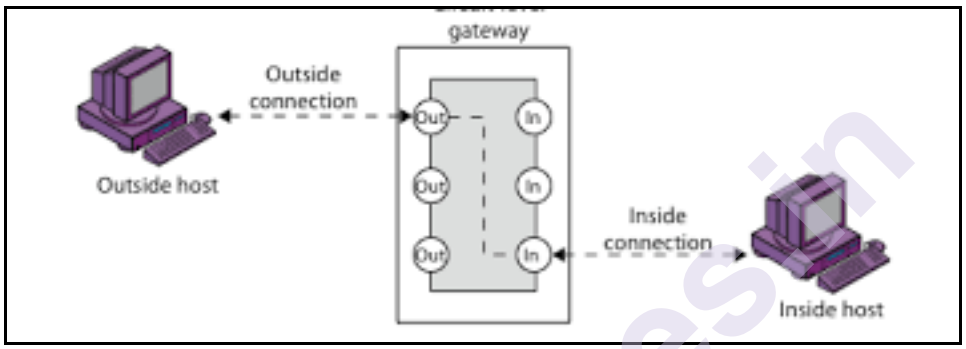


Figure: Circuit Level Gateway Firewall

Circuit level gateway can work as a standalone system or can be a specific function executed by an application level gateway for specific applications Unlike other firewalls, a Circuit level gateway does not allow an end-to-end TCP connection; rather, the gateway creates two TCP connections, one between it's own self and a TCP user on an inner host and another between its ownself and a TCP user on an outer host. As these two connections are established then the gateway will transmit the TCP segments from one connection to the other connection as shown in figure above without inspecting the contents inside the segments. The security function consists of determining which connections will be allowed and which is not to be allowed.

A Circuit level gateways is best applied where the system administrator trusts the internal users fully. The gateway is configured to provide proxy level or application level services on both inbound connections as well as circuit level functionality for outbound connections

Bastion host

It is basically a system identified by the firewall system administrator which serves as a platform for both circuit level gateway and an application level.

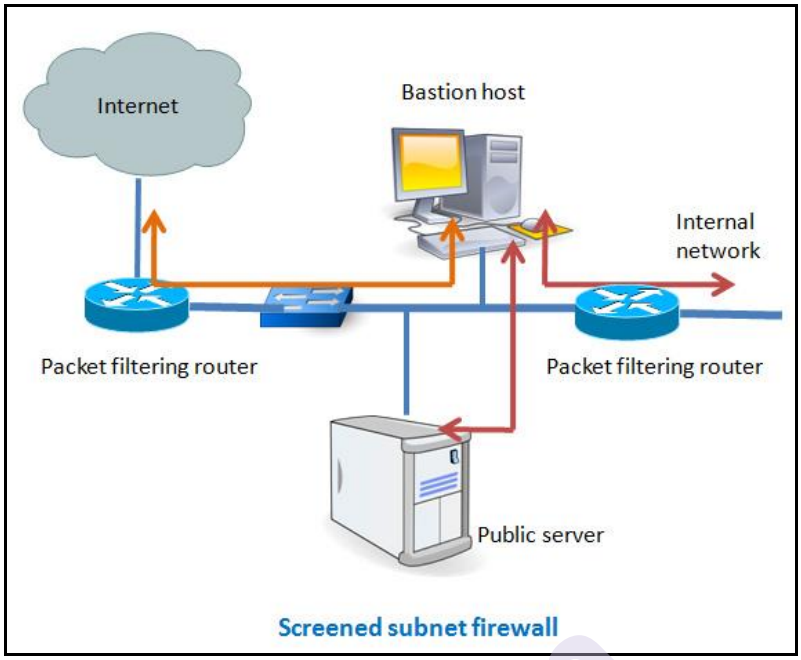


Figure:Bastion Host Firewall

Some characteristics of a Bastion host are listed below:

- The Bastion host hardware platform contributes to the execution of a secure operating system (OS) thus, making it a trusted system.
- For security reasons, only the services that are considered essential are installed on the Bastion host by the network administrator.
- It requires additional authentication to be applied before a user is allowed any access to the proxy services.
- Every proxy supports only a subset of the standard application's command set.
- Every proxy is configured to allow access only to specific host systems.
- Every proxy maintains audit information about all traffic logs, every connection established and the duration of every connection.
- On the Bastion host, each and every proxy works independently.
- A proxy only reads its initial configuration file.
- On the Bastion host, every proxy runs on a non privileged user in a secured and private directory.

4. Stateless Firewall Filter

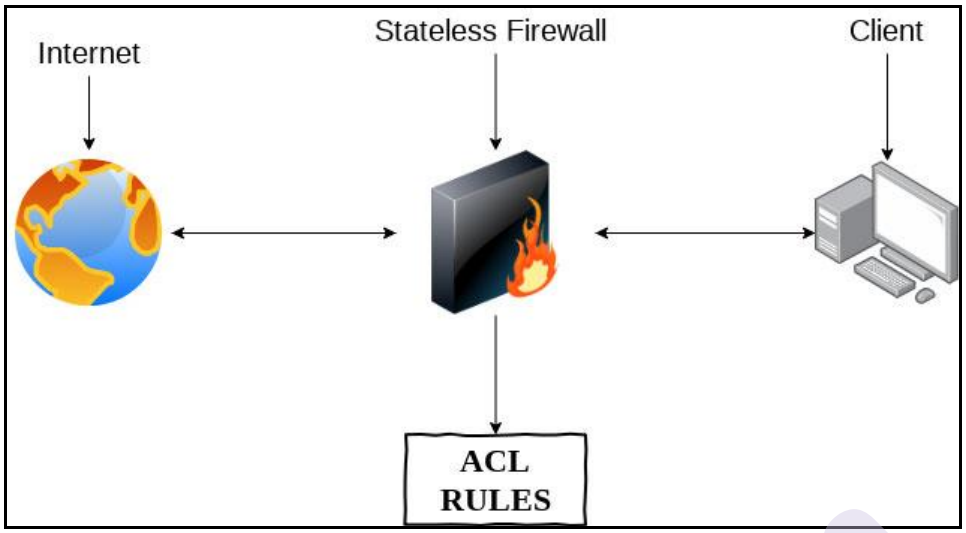


Figure: Stateless Firewall

This type of firewall monitors the network traffic. It does not have any information about the traffic patterns and restricts the pattern based on either the source or the destination. It is also known as the Access control list (ACL). This firewall basically does not inspect the traffic. It simply allows or denies packets into the network based on the set of filtering rules.

File Transfer Protocol (FTP) is a type of Stateless firewall which is implemented for sending and receiving files between two computer systems.

Stateless firewall advantages-

- It is comparatively less complex and much cheaper.
- It is very simple to implement and configure.
- It supports fast delivery Performance.
- It works efficiently under heavy traffic and pressure.

Stateless firewall disadvantages-

- It does not check the entire packet but only checks if the packet satisfies the redefined set of security rules.
- It does not inspect or monitor the traffic.
- Some additional configurations are required to increase the level of protection for security concerns.

5. Stateful Firewall

Unlike Stateless firewalls, Stateful firewalls monitor the entire state of network connections. It tracks the state of network connections when the data packets are being filtered.

As the Stateful firewalls keeps track of the state, it is aware of all the communication paths and has the capability to implement many IP security functions such as encryptions, authentication and so on.

This Firewall inspects the packets for a match with the rule in the firewall. If a match is not found, it is simply discarded but if the match is found then it is allowed into the network and can travel freely within the network.

Common example is the Transport Control Protocol(TCP). It simply saves the connection record by storing its port number, source and destination IP address, etc.

Stateful firewall advantages-

- This firewall is comparatively faster in detecting forgery or any unauthorized communication taking place.
- It has an ability for making future filtering suggestions based on the cumulative of present and past findings.
- For effective communication, many ports are not required to be kept open.
- It has a Powerful memory and extensive logging capabilities, and it is robust in attack prevention.

Stateful firewall disadvantages-

- The data transfer rate is quite slow.
- The firewall has to be updated periodically with the latest available technologies or else the system will be compromised by the hackers.
- As this firewall maintains tables of the access list, it requires a high processing and memory power.

10.7 UNIT END QUESTIONS

1. What is malicious software? How are they classified?
2. What are viruses?
3. State and explain types of viruses
4. Explain the lifecycle of a virus

5. What measures can be taken to protect the system from virus attack?
6. What is DDOS attack
7. Differentiate between DoS and DDoS attack
8. What is Firewall? What is the purpose of a firewall?
9. Explain the design principle of a firewall
10. Explain Packet Filter firewall in brief
11. Explain Application layer firewall in brief
12. Explain Circuit level firewall in brief
13. Explain Stateless firewall
14. Explain Stateful firewall
15. Distinguish between Stateless and Stateful Firewall

REFERENCE

1. Cryptography and Network, Behrouz A Fourouzan, Debdeep Mukhopadhyay, 2nd Edition,TMH,2011
2. https://www.researchgate.net/publication/320758708_E-COMMERCE_SECURITY_WITH_SECURE_ELECTRONIC_TRANSACTION_PROTOCOL_A_SURVEY_AND_IMPLEMENTATION
3. https://docs.microsoft.com/en-us/windows/win32/secauthn/tls-handshake-protocol
4. https://www.brainkart.com/article/Web-Security-Considerations_8479/
5. https://www.sciencedirect.com/topics/computer-science/authentication-header
6. https://www.techopedia.com/definition/1504/encapsulating-security-payload-esp
7. https://www.researchgate.net/publication/3728380_A_reference_model_for_firewall_technology

❄❄❄❄❄❄❄