DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Regulation 2021

III Year – V Semester

CB3491- CRYPTOGRAPHY AND CYBER SECURITY

INTRODUCTION

Security trends - Legal, Ethical and Professional Aspects of Security, Need for Security at Multiple levels, Security Policies - Model of network security – Security attacks, services and mechanisms – OSI security architecture – Classical encryption techniques: substitution techniques, transposition techniques, steganography- Foundations of modern cryptography: perfect security – information theory – product cryptosystem – cryptanalysis.

1.1

Introduction

- Human being from ages had two inherent needs − (a) to communicate and share information and (b) to communicate selectively. These two needs gave rise to the art of coding the messages in such a way that only the intended people could have access to the information. Unauthorized people could not extract any information, even if the scrambled messages fell in their hand.
- The art and science of concealing the messages to introduce secrecy in information security is recognized as cryptography.
- The word 'cryptography' was coined by combining two Greek words, 'Krypto' meaning hidden and 'graphene' meaning writing.

1.1.1 History of Cryptography

- The art of cryptography is considered to be born along with the art of writing. As civilizations evolved, human beings got organized in tribes, groups, and kingdoms. This led to the emergence of ideas such as power, battles, supremacy, and politics. These ideas further fuelled the natural need of people to communicate secretly with selective recipient which in turn ensured the continuous evolution of cryptography as well.
- The roots of cryptography are found in Roman and Egyptian civilizations.
- Hieroglyph − The Oldest Cryptographic Technique
- The first known evidence of cryptography can be traced to the use of 'hieroglyph'. Some 4000 years ago, the Egyptians used to communicate by messages written in hieroglyph. This code was the secret known only to the scribes who used to transmit messages on behalf of the kings. One such hieroglyph is shown below.

- Later, the scholars moved on to using simple mono-alphabetic substitution ciphers during 500 to 600 BC. This involved replacing alphabets of message with other alphabets with some secret rule. This **rule** became a **key** to retrieve the message back from the garbled message.

1.2 Security Trends

- Computer data often travels from one computer to another, leaving the safety of its protected physical surroundings. Once the data is out of hand, people with bad intention could modify or forge your data, either for amusement or for their own benefit.
- Cryptography can reformat and transform our data, making it safer on its trip between computers. The technology is based on the essentials of secret codes, augmented by modern mathematics that protects our data in powerful ways.
    - **Computer Security** - Generic name for the collection of tools designed to protect data and to thwart hackers.
    - **Network Security** - Measures to protect data during their transmission.
    - **Internet Security** - Measures to protect data during their transmission over a collection of interconnected networks

1.2.1 Basic Concepts of Cryptography

- The Cryptography is the art or science encompassing the principles and methods of transforming an intelligible message into one that is unintelligible and then retransforming that message back to its original form.
    - **Plaintext:** The original intelligible message.
    - **Ciphertext:** The transformed message or coded message.
    - **Cipher:** An algorithm for transforming an intelligible message into one that is unintelligible by transposition and/or substitution methods.
    - **Key:** Some critical information used by the cipher, known only to the sender& receiver.

- **Encryption (encode):** The process of converting plaintext to cipher text using a cipher and a key.
- **Decryption (Decode):** The process of converting cipher text back into plaintext using a cipher and a key.
- **Cryptanalysis:** The study of principles and methods of transforming an unintelligible message back into an intelligible message without knowledge of the key. Also called code breaking.
- **Cryptology:** The area of cryptography and cryptanalysis together are called cryptology.
- **Code**: An algorithm for transforming an intelligible message into an unintelligible one using a code-book.
- **Cryptography:** Cryptographic systems are generally classified along 3 independent dimensions:

➢ **Type of operations used for transforming plain text to cipher text:**

- All the encryption algorithms are based on two general principles: substitution, in which each element in the plaintext is mapped into another element, and transposition, in which elements in the plaintext are rearranged.

➢ **The number of keys used**

- If the sender and receiver use same key then it is said to be symmetric key (or) single key (or) conventional encryption.
- If the sender and receiver use different keys then it is said to be public key encryption.

➢ **The way in which the plain text is processed**

- A block cipher processes the input and block of elements at a time, producing output block for each input block.

- A stream cipher processes the input elements continuously, producing output element one at a time, as it goes along.

- **Cryptanalysis:** The process of attempting to discover X or K or both is known as cryptanalysis. The strategy used by the cryptanalysis depends on the nature of the encryption scheme and the information available to the cryptanalyst.

- There are various types of cryptanalytic attacks based on the amount of information known to the cryptanalyst. They are:

  - **Cipher text only** – A copy of cipher text alone is known to the cryptanalyst.

  - **Known plaintext** – The cryptanalyst has a copy of the cipher text and the corresponding plaintext.

  - **Chosen plaintext** – The cryptanalysts gain temporary access to the encryption machine. They cannot open it to find the key, however; they can encrypt a large number of suitably chosen plaintexts and try to use the resulting cipher texts to deduce the key.

  - **Chosen cipher text** – The cryptanalyst obtains temporary access to the decryption machine, uses it to decrypt several strings of symbols, and tries to use the results to deduce the key.

1.3 Legal, Ethical and Professional Aspects of Security

- Today millions of people perform online transactions every day. There many ways to attack computer and networks to take advantage of what has made shopping, banking, transformation of messages, investments and leisure pursuits a simple matter of dragging and clicking for many people.

- Thus, the laws and ethics are important aspects in data and network security. The legal system has adapted quite well to computer technology by reusing some old forms of legal protection (copyrights and patents) and creating laws where no adequate one existed (malicious access).

- Still the courts are not a perfect form of protection for computer, for two reasons, first court tends to be reactive instead of proactive. That is, we have to wait for regression to occur and then adjudicative it, rather than try to prevent it in first place. Second fixing a problem through the courts can be time consuming and more expensive.

- The latter characteristic prevents all but the wealthy from addressing most wealthy. On other hand, 1ethics has not had to change , because ethic is more situational and personal than the law, for example the privacy of personal information becoming important part of computer network security and although technically this issue is just an aspect of confidentiality, practically it has a long history in both law and ethics.

- Law and security are related in several ways. First international, national, state, city laws affect privacy, secrecy. These statutes often apply to the rights of individuals to keep personal matters private. Second law regulates the use of development, and ownership of data and programs. Patents, copyrights, and trade secrets are legal devices to protect the right of developers and owners of the information and data.

Cryptography and Law

- Cyber-Crime: - Criminal activities or attacks in which computer and computer networks are tool, target, or place of criminal activity. Cybercrime categorize based on computer roles such as target, storage device and communication tool.

- Computers as targets: To get the information from the computer system or control the computer system without the authorization or payment or alter the interfaces or data in the particular system with use of server.

- Computers as storage devices: Computers can be used to further unlawful activity by using a computer or a computer device as a passive storage medium. For example, the computer can be used to store stolen password lists, credit card details and proprietary corporate information.

- Computers as communications tools: Many of the crimes falling within this category are simply traditional crimes that are committed online. Examples

include the illegal sale of prescription drugs, controlled substances, alcohol, and guns; fraud; gambling; and child pornography. Other than these crimes there are more specific crimes in computer networks. There are:

- Illegal access: The access to the whole or any part of a computer system without right.

  Illegal interception: The interception without right, made by technical means, of non-public transmissions of computer data to, from or within a computer system, including electromagnetic emissions from a computer system carrying such computer data.

- Data interference: The damaging, deletion, deterioration, alteration or suppression of computer data without right.

- System interference: The serious hindering without right of the functioning of a computer system by inputting, transmitting, damaging, deleting, deteriorating, altering or suppressing computer data.

- Computer-related forgery: The input, alteration, deletion, or suppression of computer data, resulting in inauthentic data with the intent that it be considered or acted upon for legal purposes as if it were authentic, regardless whether or not the data is directly readable and intelligible.

- Crime related to child pornography: Producing child pornography or distribution through a computer system and making available or distributing or transmitting child pornography through a computer system.

➢ The relative lack of success in bringing cyber-criminals to justice has led to an increase in their numbers, boldness, and the global scale of their operations. It is difficult to profile cybercriminals in the way that is often done with other types of repeat offenders.

➢ The success of cybercriminals and the relative lack of success of law enforcement, influence the behaviour of cybercrime victims. As with law enforcement, many

organizations that may be the target of attack have not invested sufficiently in technical, physical, and human-factor resources to prevent attacks.

- The law is used regulate people for their own good and for the greater good of society. Cryptography also regulated activity.

- Some Example laws that are forced on cryptography.

    - Control use of cryptography: Closely related to restrictions on content are restrictions on the use of cryptography imposed on users in certain countries. For examples, 2 In China, state council order 273 requires foreign organizations or individuals to apply permission to use encryption in China. Pakistan requires that all encryption hardware and software be inspected and approved by the Pakistan telecommunication authority.

    - Cryptography and Free speech: The Cryptography involve not just products, it involves ideas too, although governments effectively control the flow of products across borders, controlling the floe ideas either head or on the internet, is also impossible.

    - Cryptography and Escrow: Although laws enable governments to read encrypted communications. In 1996, US government offered to relax the export restriction for so called escrowed encryption, in which the government would able to obtain the encryption key for any encrypted communication. The victory in use of law enforcement depends much more on technical skills of the people. Management needs to understand the criminal investigation process, the inputs that investigators need, and the ways in which the victim can contribute positively to the investigation. Intellectual properties.

    - There are three main types of intellectual property for which legal protection is available.

      1) Copy rights: Copyright law protects the tangible or fixed expression of an idea, not the idea itself. Copy right properties exists when proposed work is original and creator has put original idea in concrete form and the copyright

owner has these exclusive rights, protected against infringement such as reproduction right, modification right, distribution right

2) Patents: A patent for an invention is the grant of a property right to the inventor. There are 3 types in patents:-

- Utility (any new and useful process, machine, article of manufacture, or composition of matter).
- Design (new, original, and ornamental design for an article of manufacture)
- Plant (discovers and asexually reproduces any distinct and new variety ofplant).

3) Trade-Marks: A trademark is a word, name, symbol or expression which used to identify the products or services in trade uniquely from others. Trade mark rights used to prevent others from using a confusingly similar mark, but not to prevent others from making the same goods or from selling the same goods or services under a clearly different mark.

- Intellectual Property Relevant to Network and Computer Security
  A number of forms of intellectual property are relevant in the context of network and computer security.

- Software programs: software programs are protected by using copyright, perhaps patent.

- Digital content: audio / video / media / web protected by copy right
  Algorithms: algorithms may be able to protect by patenting

- Privacy Law and Regulation: An issue with considerable overlap with computer security is that of privacy. Concerns about the extent to which personal privacy has been and may be compromised have led to a variety of legal and technical approaches to reinforcing privacy rights. A number of international organizations and national

governments have introduced laws and regulations intended to protect individual privacy.

- European Union Data Protection Directive was adopted in 1998 to ensure member states protect fundamental privacy rights when processing personal info and prevent member states from restricting the free flow of personal info within EU organized around principles of notice, consent, consistency, access, security, onward transfer and enforcement. US Privacy Law have Privacy Act of 1974 which permits individuals to determine records kept, forbid records being used for other purposes, obtain access to records, ensures agencies properly collect, maintain, and use personal info and creates a private right of action for individuals.
  Cryptography and Ethics.

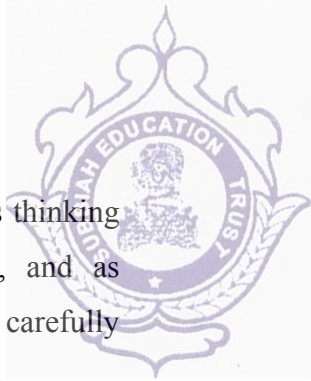
- There are many potential misuses and abuses of information and electronic communication that create privacy and security problems. Ethics refers to a system of moral principles that relates to the benefits and harms of particular actions. An ethic an objectively defined standard of right and wrong. Ethical standards are often idealistic principles because they focus on one objective. Even though religious group and professional organization promote certain standards of ethical behaviour, ultimately each person is responsible for deciding what do in a specific situation.

Ethical issues related to computer and info systems:

- Computers have become the primary repository of both personal information and negotiable assets, such as bank records, securities records, and other financial information.

    - Repositories and processors of information: Unauthorized use of otherwise unused computer services or of information stored in computers raises questions of appropriateness or fairness.

    - Producers of new forms and types of assets: For example, computer programs are entirely new types of assets, possibly not subject to the same concepts of ownership as other assets.



- Symbols of intimidation and deception: The images of computers as thinking machines, absolute truth producers, infallible, subject to blame, and as anthropomorphic replacements of humans who err should be carefully considered.

1.4 Need for Security at Multiple levels

- Multilevel security or multiple levels of security (MLS) is the application of a computer system to process information with incompatible classifications (i.e., at different security levels), permit access by users with different security clearances and needs-to-know, and prevent users from obtaining access to information for which they lack authorization.
- There are two contexts for the use of multilevel security.
  - One is to refer to a system that is adequate to protect itself from subversion and has robust mechanisms to separate information domains, that is, trustworthy.
  - Another context is to refer to an application of a computer that will require the computer to be strong enough to protect itself from subversion and possess adequate mechanisms to separate information domains, that is, a system we must trust. This distinction is important because systems that need to be trusted are not necessarily trustworthy.

Security Policies

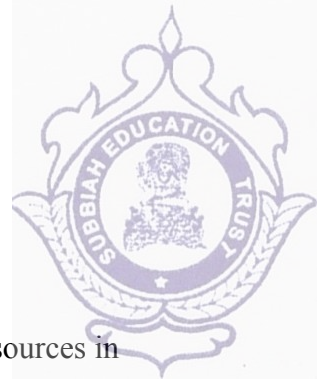
- The Cryptography Policy sets out when and how encryption should be used. It includes protection of sensitive information and communications, key management, and procedures to ensure encrypted information can be recovered by the organisation if necessary.

Role of the Security Policy in Setting up Protocols

Following are some pointers which help in setting u protocols for the security policy of an organization.

- Who should have access to the system?
- How it should be configured?
- How to communicate with third parties or systems?

Policies are divided in two categories:



- User policies
- IT policies.

➢ User policies generally define the limit of the users towards the computer resources in a workplace. For example, what are they allowed to install in their computer, if they can use removable storages?

Whereas, IT policies are designed for IT department, to secure the procedures and functions of IT fields.

- **General Policies** − This is the policy which defines the rights of the staff and access level to the systems. Generally, it is included even in the communication protocol as a preventive measure in case there are any disasters.
- **Server Policies** − This defines who should have access to the specific server and with what rights. Which software's should be installed, level of access to internet, how they should be updated?
- **Firewall Access and Configuration Policies** − It defines who should have access to the firewall and what type of access, like monitoring, rules change. Which ports and services should be allowed and if it should be inbound or outbound?
- **Backup Policies** − It defines who is the responsible person for backup, what should be the backup, where it should be backed up, how long it should be kept and the frequency of the backup.
- **VPN Policies** − These policies generally go with the firewall policy; it defines those users who should have a VPN access and with what rights. For site-to-site connections with partners, it defines the access level of the partner to your network, type of encryption to be set.

Structure of a Security Policy

When you compile a security policy you should have in mind a basic structure in order to make something practical. Some of the main points which have to be taken into consideration are:

- Description of the Policy and what is the usage for?
- Where this policy should be applied?

- Functions and responsibilities of the employees that are affected by this policy.
- Procedures that are involved in this policy.
- Consequences if the policy is not compatible with company standards.

Types of Policies

In this section we will see the most important types of policies.

- **Permissive Policy** − It is a medium restriction policy where we as an administrator block just some well-known ports of malware regarding internet access and just some exploits are taken in consideration.
- **Prudent Policy** − This is a high restriction policy where everything is blocked regarding the internet access, just a small list of websites is allowed, and now extra services are allowed in computers to be installed and logs are maintained for every user.
- **Acceptance User Policy** − This policy regulates the behavior of the users towards a system or network or even a webpage, so it is explicitly said what a user can do and cannot in a system. Like are they allowed to share access codes, can they share resources, etc.
- **User Account Policy** − This policy defines what a user should do in order to have or maintain another user in a specific system. For example, accessing an e-commerce webpage. To create this policy, you should answer some questions such as −
    - Should the password be complex or not?
    - What age should the users have?
    - Maximum allowed tries or fails to log in?
    - When the user should be deleted, activated, blocked?
- **Information Protection Policy** − This policy is to regulate access to information, hot to process information, how to store and how it should be transferred.
- **Remote Access Policy** − This policy is mainly for big companies where the user and their branches are outside their headquarters. It tells what should the users access, when they can work and on which software like SSH, VPN, RDP.

- **Firewall Management Policy** − This policy has explicitly to do with its management, which ports should be blocked, what updates should be taken, how to make changes in the firewall, how long should be the logs be kept.
- **Special Access Policy** − This policy is intended to keep people under control and monitor the special privileges in their systems and the purpose as to why they have it. These employees can be team leaders, managers, senior managers, system administrators, and such high designation based people.
- **Network Policy** − This policy is to restrict the access of anyone towards the network resource and make clear who all will access the network. It will also ensure whether that person should be authenticated or not. This policy also includes other aspects like, who will authorize the new devices that will be connected with network. The documentation of network changes. Web filters and the levels of access. Who should have wireless connection and the type of authentication, validity of connection session?
- **Email Usage Policy** − This is one of the most important policies that should be done because many users use the work email for personal purposes as well. As a result information can leak outside. Some of the key points of this policy are the employees should know the importance of this system that they have the privilege to use. They should not open any attachments that look suspicious. Private and confidential data should not be sent via any encrypted email.
- **Software Security Policy** − This policy has to do with the software's installed in the user computer and what they should have. Some of the key points of this policy are Software of the company should not be given to third parties. Only the white list of software's should be allowed, no other software's should be installed in the computer. Warez and pirated software's should not be allowed.

1.5 Model of network security

- When we send our data from source to destination, we have to use some transfer method like the internet or any other communication channel.
- The two parties, who are the principals in this transaction, must cooperate for each other to the exchange the message. When the transfer of data happened from one source to another source some logical information channel is established between

them by defining a route through the internet from source to destination and by the cooperative use of communication protocols (e.g., TCP/IP) by the two principals.

- It is necessary to protect the information from various types of attackers, who may launch a threat to confidentiality, authenticity, DoS and so on. All the technique providing some security components:
    - A security-related transformation on the information to be sent. That means, the encryption of the message, which scrambles the message so that it is unreadable by the attacker.
    - Some of the secret information shared by the two parties. So, it is hoped, unknown to the attacker.
    - A trusted third party may be needed to achieve secure transmission. For example, a third party may be responsible for distributing the secret information to the two principals while keeping it from any attacker.

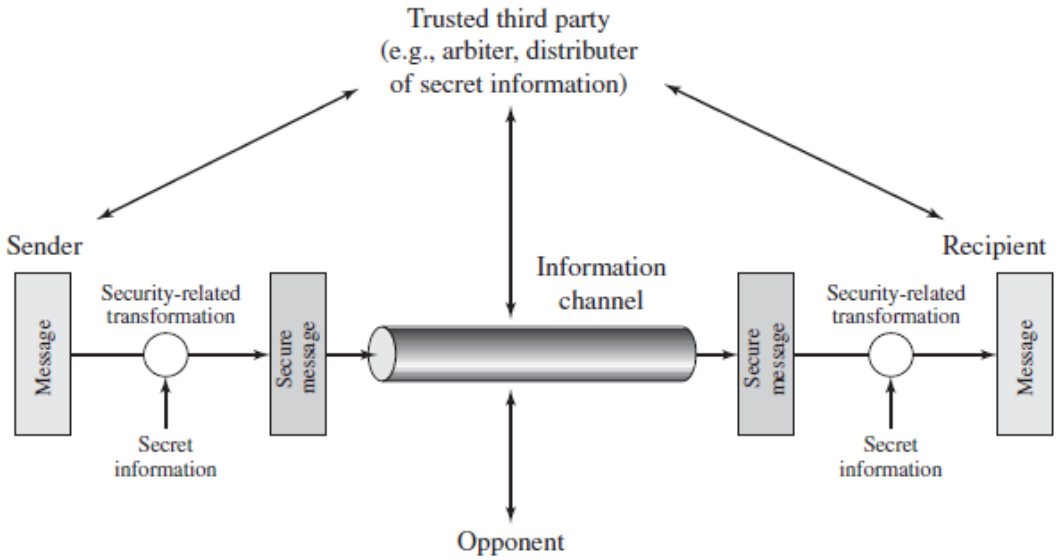


Figure 1.1 Network Security Model

This model (Figure 1.1) shows that there are four basic tasks in designing a particular security service:

1. Design an algorithm for performing the security-related transformation.
2. Generate the secret information to be used with the algorithm.
3. Develop methods for the distribution and sharing of secret information.

4. Specify a protocol to be used by the two principals that make use of the security algorithm and the secret information to achieve a particular security service.

- A network access security model is illustrated by Figure 1.2, which reflects a concern for protecting an information system from unwanted access. The hackers, who attempt to penetrate systems that can be accessed over a network. The hacker can be someone who, with no malign intent, simply gets satisfaction from breaking and entering a computer system. The intruder can be a disgruntled employee who wishes to do damage or a criminal who seeks to exploit computer assets for financial gain (e.g., obtaining credit card numbers or performing illegal money transfers).

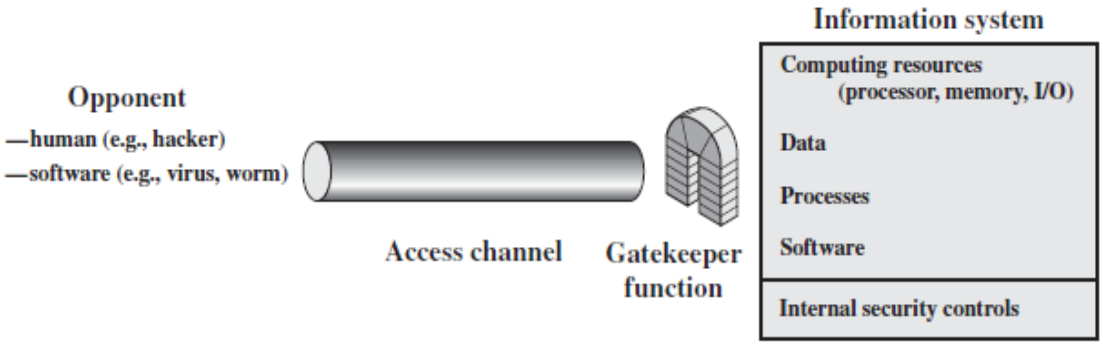


Figure 1.2 Network Access Security Model

- Another type of unwanted access can affect the application programs. The Viruses and Worms are two types of software attacks. There are two kinds of threat:
  - **Information access threats:** Intercept or modify data on behalf of users who should not have access to that data.
  - **Service threats:** Exploit service flaws in computers to inhibit use by legitimate users.

1.6 OSI Security Architecture

- The OSI Security Architecture is a framework that provides a systematic way of defining the requirements for security and characterizing the approaches to satisfying those requirements. X. 800 recommends this architecture for OSI.
- The OSI security architecture mainly focuses on:

- Security attacks- Any action that compromises the security of information
- Security services- The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service
- Security mechanisms- A process that is designed to detect, prevent, or recover from a security attack.

Security Attacks

An assault on system security that derives from an intelligent threat; that is, an intelligent act that is a deliberate attempt (especially in the sense of a method or technique) to evade security services and violate the security policy of a system. The security attacks are broadly classified into two types:

1. Passive Attack
2. Active Attack

Passive Attack

- A **passive attack** is a network **attack** in which a system is monitored and sometimes scanned for open ports and vulnerabilities. It attempts to learn or make use of information from the system but does not affect system resources. The attacker aims to obtain information that is in transit. The attacker does not perform any modification of data. There are two types of passive attacks.
    1. Release of Message Contents
    2. Traffic Analysis

Release of Message Contents

- For a release of message content (figure 1.3), a telephonic conversation, an E-mail message or a transferred file may contain confidential data.
- A passive attack monitors the contents of the transmitted data. When the messages are exchanged neither the sender nor the receiver is aware that a third party may capture the messages. We have to prevent an opponent from learning the contents of these transmissions.

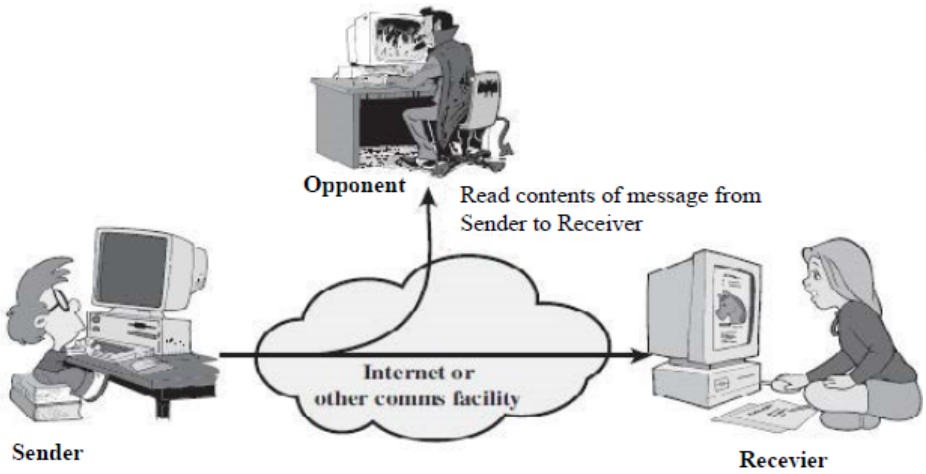


Figure 1.3 Release of Message Contents

Traffic Analysis

- Traffic analysis is the process of intercepting and examining network traffic in order to deduce information from patterns in communication. It can be performed even when the traffic is encrypted and cannot be decrypted by the party performing the analysis. Figure 1.4 shows the traffic analysis attack.

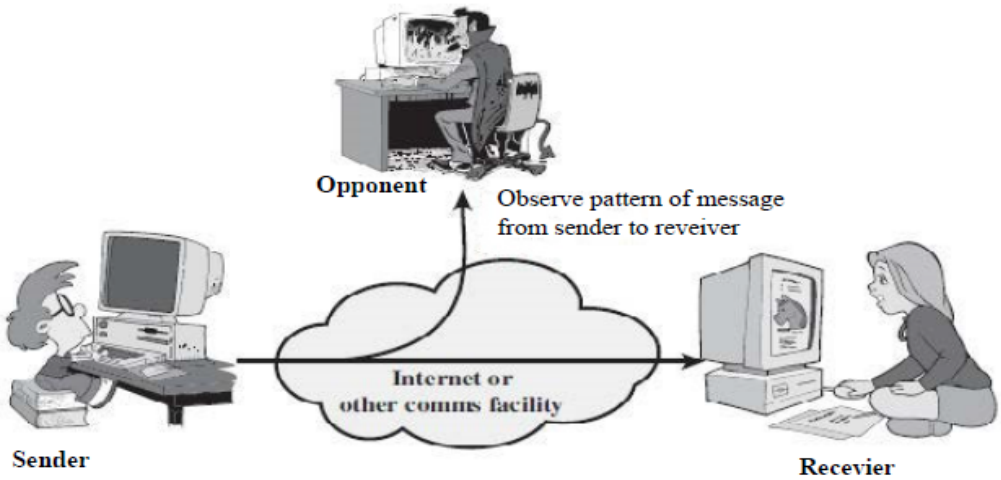


Figure 1.4 Traffic Analysis

- Passive attacks are very difficult to detect, because they do not involve any alteration of the data. But, using strong encryption algorithm we can prevent this attack.

Active Attack

- Active attacks involve some modification of the data stream or the creation of a false data and inject into the network.

- It can be subdivided into four categories:
  - Masquerade
  - Replay
  - Modification of messages
  - Denial of Service (DoS)

Masquerade

- A masquerade is a type of attack where the attacker pretends to be an authorized user of a system in order to gain access to it or to gain greater privileges than they are authorized for. Figure 1.5 shows this attack.

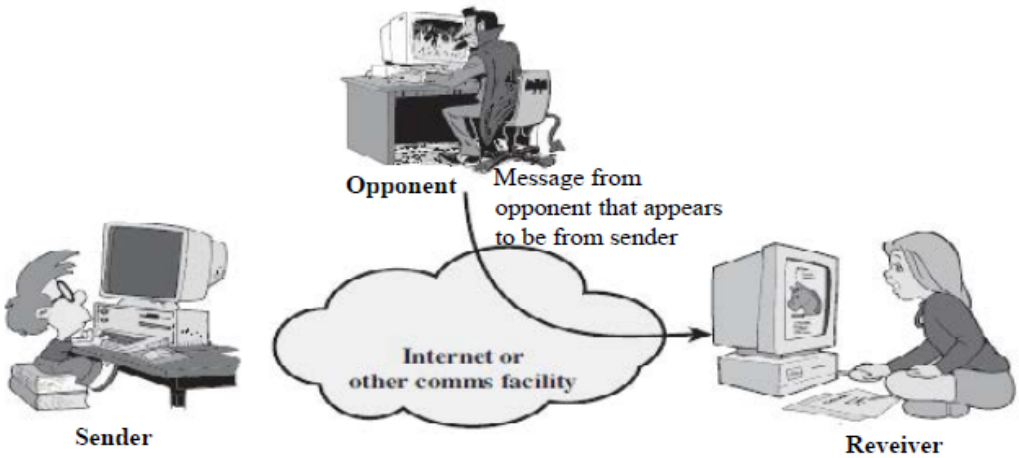


Figure 1.5 Masquerade

Replay

- A replay attack is a form of network attack in which a valid data transmission is maliciously or fraudulently repeated or delayed.
- This is carried out either by the originator or by an adversary who intercepts the data and re-transmits it, possibly as part of a masquerade attack by IP packet substitution. Figure 1.6 shows replay attack.

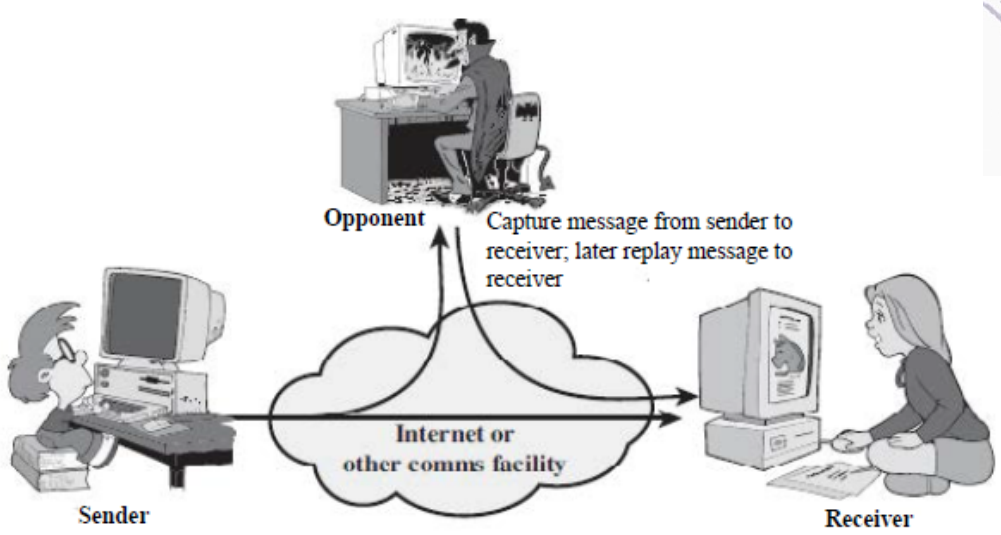


Figure 1.6 Replay

Modification of messages

- It simply means that some portion of an authorized message is altered, or that messages are delayed or reordered, to produce an unauthorized effect.
- For example, a message meaning "Allow Roy to read confidential file accounts" is changed to "Allow Darwin to read confidential file accounts". Figure 1.7 shows this attack.

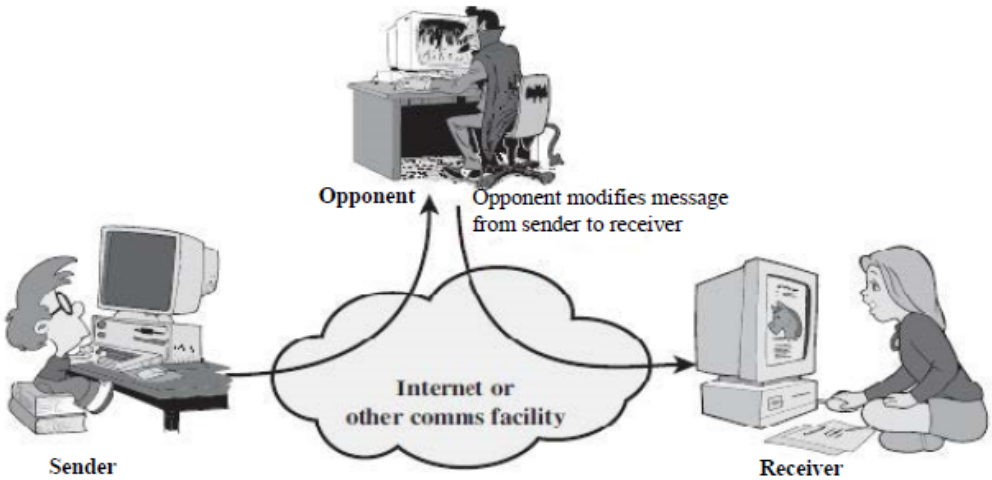


Figure 1.7 Moddification of Message

Denial of Service

- A Denial-of-Service attack (DoS attack) is an attack where an attacker attempts to disrupt the services provided by a host, by not allowing its intended users to access the host from the Internet.

- If the attack succeeds, the targeted computer will become unresponsive and nobody will be able to connect with it.

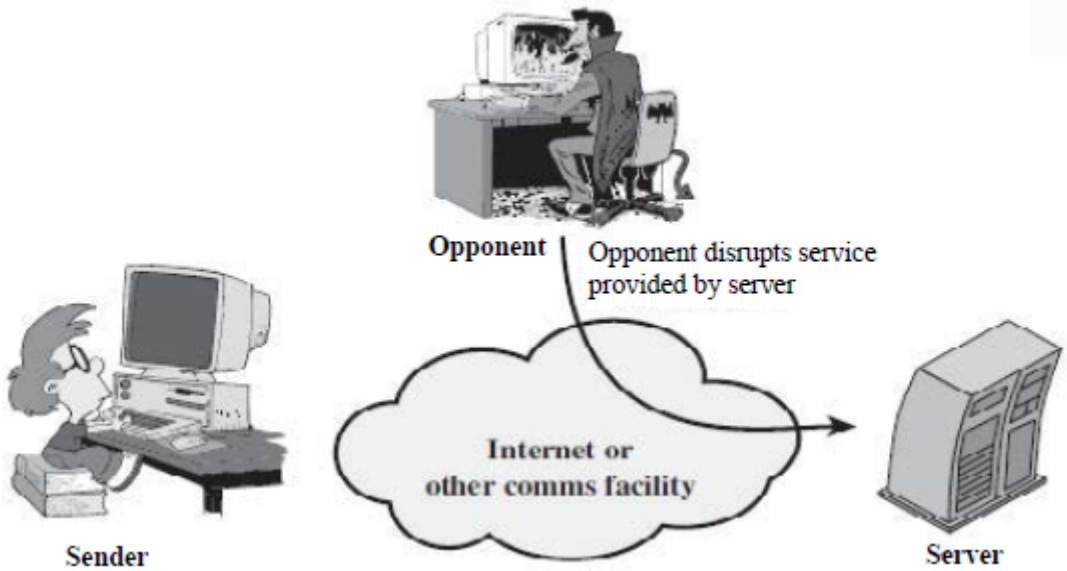


Figure 1.8 Denial of Service (DoS)

Security Services

- Security service is a service, provided by a layer of communicating open systems, which ensures adequate security of the systems or of data transfers as defined by ITU-T X. 800 Recommendation.
- X.800 divides security services into five different categories:
  - Authentication
  - Access control
  - Data confidentiality
  - Data integrity
  - Nonrepudiation
  - Availability Service

Authentication

- **Authentication** is the process of recognizing a user's identity. It is the mechanism of associating an incoming request with a set of identifying credentials. The Identification phase provides a user identity to the **security** system. This identity is provided in the form of a user ID.
- Two specific authentication services are defined in X.800:
  - **Peer entity authentication:** Provides for the corroboration of the identity of a peer entity in an association. Two entities are considered peers if they

implement to same protocol in different systems; e.g., two TCP modules in two communicating systems. It attempts to provide confidence that an entity is not performing either a masquerade or an unauthorized replay of a previous connection.

- **Data origin authentication:** Provides for the corroboration of the source of a data unit. It does not provide protection against the duplication or modification of data units. This type of service supports applications like electronic mail, where there are no prior interactions between the communicating entities.

Access control

- The goal of access control is to minimize the risk of unauthorized access to physical and logical systems.
- Access control is a fundamental component of security compliance programs that ensures security technology and access control policies are in place to protect confidential information, such as customer data.

Data confidentiality

- Confidentiality refers to protecting information from being accessed by unauthorized parties. In other words, only the people who are authorized to do so can gain access to sensitive data. Such a failure of confidentiality, commonly known as a breach, typically cannot be remedied.
- Confidentiality is classified into
  - **Connection Confidentiality**
    - The protection of all user data on a connection.
  - **Connectionless Confidentiality**
    - The protection of all user data in a single data block
  - **Selective-Field Confidentiality**
    - The confidentiality of selected fields within the user data on a connection or in a single data block.
  - **Traffic Flow Confidentiality**

- The protection of the information that might be derived from observation of traffic flows.

Data integrity

- Ensures that only authorized parties are able to modify computer system assets and transmitted information. Modification includes writing, changing status, deleting, creating and delaying or replaying of transmitted messages.
- Data Integrity can be classified into
  - **Connection Integrity with Recovery**
    - Provides for the integrity of all user data on a connection and detects any modification, insertion, deletion, or replay of any data within an entire data sequence, with recovery attempted.
  - **Connection Integrity without Recovery**
    - It provides only detection without recovery.
  - **Selective-Field Connection Integrity**
    - Provides for the integrity of selected fields within the user data of a data block transferred over a connection and takes the form of determination of whether the selected fields have been modified, inserted, deleted, or replayed.
  - **Connectionless Integrity**
    - Provides for the integrity of a single connectionless data block and may take the form of detection of data modification. Additionally, a limited form of replay detection may be provided.
  - **Selective-Field Connectionless Integrity**
    - Provides for the integrity of selected fields within a single connectionless data block; takes the form of determination of whether the selected fields have been modified.

Nonrepudiation

- Nonrepudiation Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication.
- Nonrepudiation can be related to
    - **Nonrepudiation, Origin**
        - Proof that the message was sent by the specified party.
    - **Nonrepudiation, Destination**
        - Proof that the message was received by the specified party.

Example: Imagine a user of online banking who has made a transaction, but later denied that. How the bank can protect itself in a such situation?

Availability Service

- An availability service is one that protects a system to ensure its availability. This service addresses the security concerns raised by denial-of-service attacks. It depends on proper management and control of system resources and thus depends on access control service and other security services.

Security Mechanisms

- Security mechanisms are technical tools and techniques that are used to implement security services.
- A mechanism might operate by itself, or with others, to provide a particular service. Examples of common security mechanisms are as follows: Cryptography, Message digests and digital signatures.

Difference between Active attack and Passive attack

Passive Attack	Active Attack
Attempts to learn or make use of information from the system but does not affect system resources.	Attempts to alter system resources or affect their operation.
Eavesdropping on, or monitoring of, transmissions.	Involve some modification of the data stream or the creation of a false stream.
Goal of attacker is to obtain information that is being transmitted	Goal of attacker is to damage any system.
Two types: 1. Release of message contents 2. Traffic analysis	Four categories: 1. Replay 2. Masquerade 3. Modification of messages 4. Denial of service

Difference between Threat and Attack

Threat

A potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm. That is, a threat is a possible danger that might exploit a vulnerability.

Attack

An assault on system security that derives from an intelligent threat; that is, an intelligent act that is a deliberate attempt (especially in the sense of a method or technique) to evade security services and violate the security policy of a system.

1.7 Classical Encryption Techniques

- Classical cryptography is based on the mathematics and it relies on the computational difficulty of factorizing large number. The security of classical cryptography is based on the high complexity of the mathematical problem for the instance factorization of large number.

Symmetric Cipher Model

- Single key is used for both encryption and decryption. A symmetric encryption scheme has five ingredients

- **Plaintext:** This is the original intelligible message or data that is fed into the algorithm as input.
- **Encryption algorithm:** The encryption algorithm performs various substitutions and transformations on the plaintext.
- **Secret key:** The secret key is also input to the encryption algorithm. The key is a value independent of the plaintext and of the algorithm. The algorithm will produce a different output depending on the specific key.
- **Ciphertext:** This is the scrambled message produced as output. It depends on the plaintext and the secret key.
- **Decryption algorithm:** This is essentially the encryption algorithm run in reverse. It takes the ciphertext and the secret key and produces the original plaintext

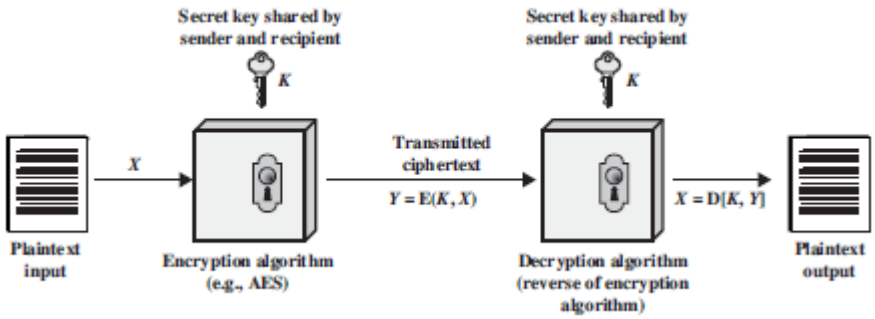


Figure 1.8 Symmetruc Cipher Model

Transmission Over Secure Channel

- Two requirements for secure use of symmetric encryption:
  - A strong encryption algorithm
  - A secret key known only to sender / receiver
- The message X and the encryption key K as input, the encryption algorithm forms the ciphertext $Y = [Y_1, Y_2, , Y_N]$.

$$Y = \mathrm{E}(K, X)$$

- Here, Y that is produced by using encryption algorithm E as a function of the plaintext X, with the specific function determined by the value of the key K.
- The receiver, in possession of the key, is able to reverse the transformation:

$$X = \mathrm{D}(K, Y)$$

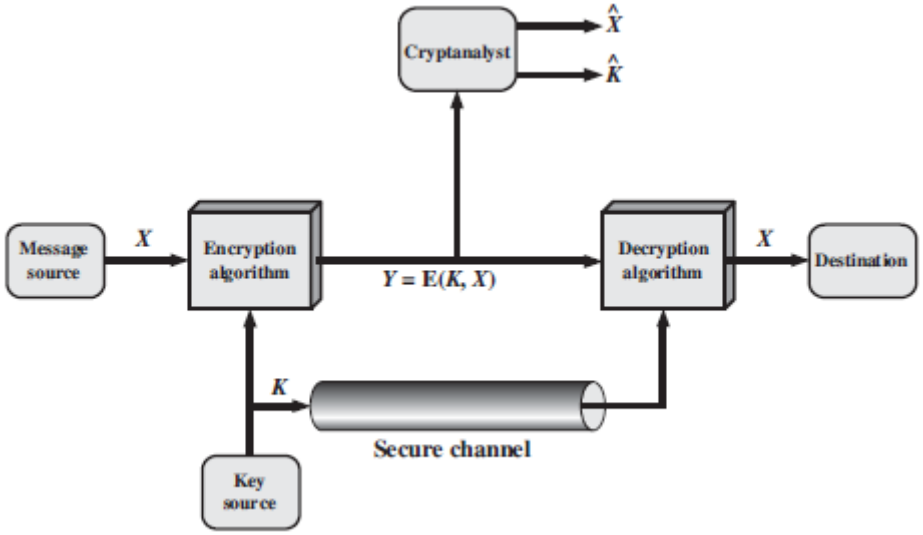


Figure 1.8 Transmission Over Secure Channel

- An opponent, observing Y but not having access to K or X, may attempt to recover X or Y or both X and Y. It is assumed that the opponent knows the encryption (E) and decryption (D) algorithms.

Cryptanalysis and Brute-Force Attack

Cryptanalysis

- **Cryptanalysis** is the investigation of systems, ciphertext, and ciphers in order to reveal the hidden meaning or details of the system itself. The goal of this type of study is to discover the hidden aspects even if the key or main algorithm is unable to be deciphered.

Brute-Force Attack

- The attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. On average, half of all possible keys must be tried to achieve success.
- There are two basic building blocks of all encryption techniques:
  - Substitution
  - Transposition

Substitution Techniques

- A substitution technique is one in which the letters of plaintext are replaced by other letters or by numbers or symbols. If the plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext bit patterns with cipher text bit patterns.

Caesar cipher (or) shift cipher

- The earliest known use of a substitution cipher and the simplest was by Julius Caesar. The Caesar cipher involves replacing each letter of the alphabet with the letter standing 3 places further down the alphabet. The plaintext will be written in lowercase, ciphertext will be written in uppercase. Let as assign a numerical equivalent to each letter.

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

Where a= 0, z = 25

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
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

Example

Plaintext: Pay more money

Ciphertext: SDB PRUH PRQHB

The general Caesar algorithm is,

$$c = E(k, p) = (p + k) \bmod 26$$
$$p = D(k, c) = (c - k) \bmod 26$$

Example

Let k = 3

C = E (3, P)

C = (P+3) mod 26

Encryption

Plaintext = cat

Let K = 3, C= 2

C = 2+ 3

C = 5

C = F

Next letter, a= 0

C = 0 +3

C = D

Next, t = 19

So, C = w

Ciphertext = FDW

Now, Decryption is just reverse process of Encryption

Drawbacks

- Bruteforce cryptanalysis can be easily performed by trying all the 25 possible keys.
- The language of the plaintext was english.

Monoalphabetic Ciphers

- Rather than just shifting the alphabet
- Could shuffle (jumble) the letters arbitrarily
- Each plaintext letter maps to a different random ciphertext letter
- Hence, key is 26 letters long

```
Plain:   abcdefghijklmnopqrstuvwxyz
Cipher:  DKVQFIBJWPESCXHTMYAUOLRGZN

Plaintext:   ifwewishtoreplaceletters
Ciphertext:  WIRFRWAJUHYFTSDVFSFUUFYA
```

- Now have a total of 26! = 4 x 1026 keys
- with so many keys, might think is secure

Drawback

- It is easy to break because they reflect the frequency data of the original alphabet.

Playfair Cipher

- The best-known multiple letter encryption cipher is the Playfair, which treats diagrams in the plaintext as single units and translates these units into cipher text diagrams. The Playfair algorithm is based on the use of 5x5 matrix of letters constructed using a keyword. The technique encrypts pairs of letters instead of single letters.

Example

Key: Monarchy

Plaintext: instruments

The Playfair Cipher Encryption Algorithm:

The Algorithm consists of 2 steps:

1. **Generate the key Square(5×5):**
   - The key square is a 5×5 grid of alphabets that acts as the key for encrypting the plaintext. Each of the 25 alphabets must be unique and one letter of the alphabet (usually J) is omitted from the table (as the table can hold only 25 alphabets). If the plaintext contains J, then it is replaced by I.
   - The initial alphabets in the key square are the unique alphabets of the key in the order in which they appear followed by the remaining letters of the alphabet in order.

   The key is "**monarchy**"

   Thus the initial entires are

   'm', 'o', 'n', 'a', 'r', 'c', 'h', 'y'

followed by remaining characters of **a-z(except 'j')** in that order.

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

2. Algorithm to encrypt the plain text: The plaintext is split into pairs of two letters (digraphs). If there is an odd number of letters, a Z is added to the last letter.

PlainText: "instruments"

After Split: 'in' 'st' 'ru' 'me' 'nt' 'sz'

Rules for Encryption:

- **If both the letters are in the same column**: Take the letter below each one (going back to the top if at the bottom).

  For example:

  Diagraph: "me"

  Encrypted Text: cl

  Encryption:

m -> c

e -> l

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

If both the letters are in the same row: Take the letter to the right of each one (going back to the leftmost if at the rightmost position).

For example:

Diagraph: "st"

Encrypted Text: tl

Encryption:

s -> t

t -> l

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

If neither of the above rules is true: Form a rectangle with the two letters and take the letters on the horizontal opposite corner of the rectangle.

For example:

Diagraph: "nt"

Encrypted Text: rq

Encryption:

n -> r

t -> q

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

For example:

Plain Text: "instrumentsz"

Encrypted Text: gatlmzclrqtx

Encryption:

i -> g

n -> a

s -> t

t -> l

r -> m

u -> z

m -> c

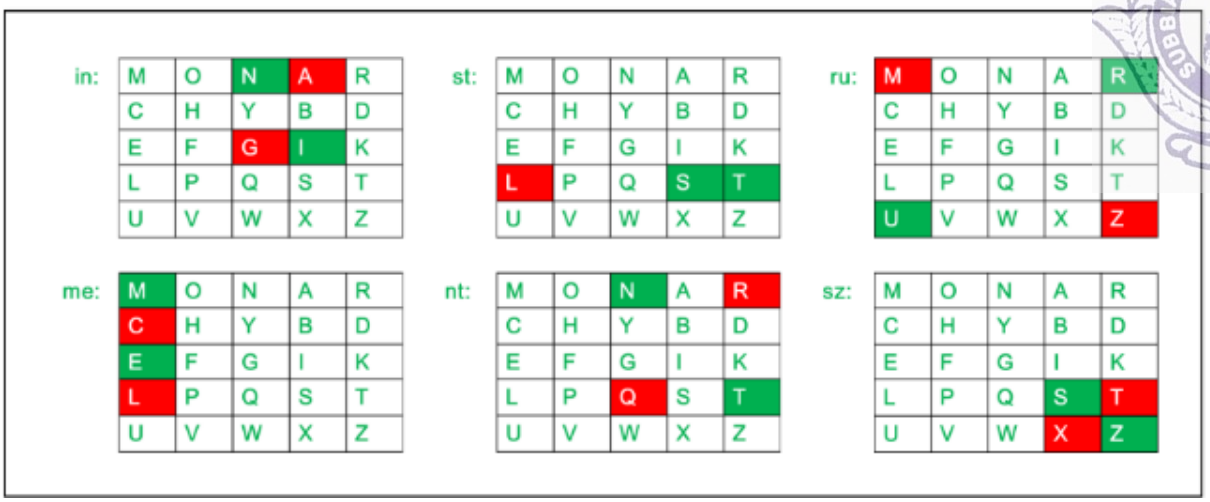
e -> l

n -> r

t -> q

s -> t

z -> x



Decryption

Plain Text: "gatlmzclrqtx"

Decrypted Text: instrumentsz

Decryption:

(red)-> (green)

ga -> in

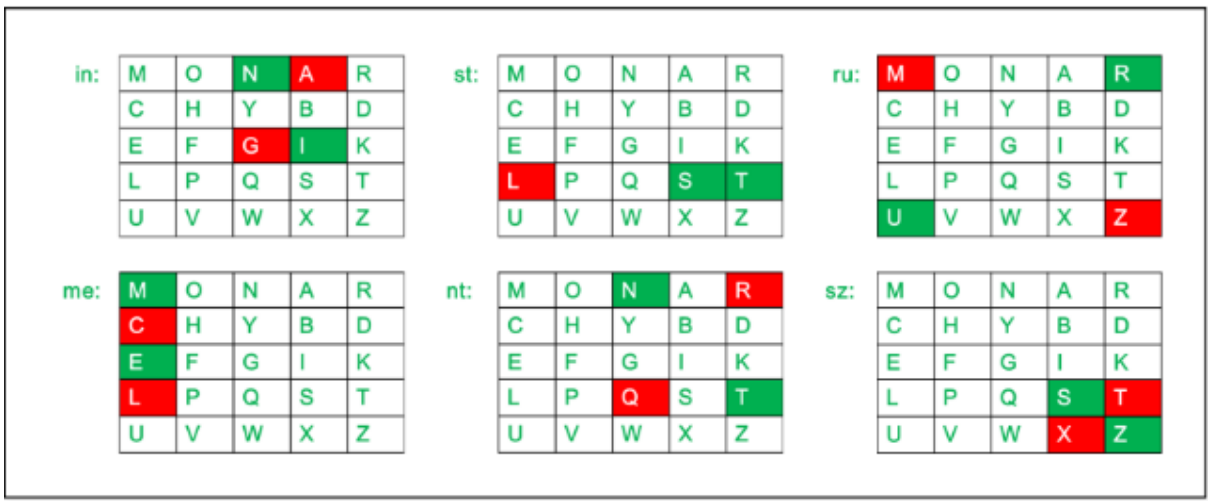
tl -> st

mz -> ru

cl -> me

rq -> nt

tx -> sz



Advantages

- Play fair cipher is a great advance over simple mono alphabetic ciphers.
- Since there are 26 letters, 26 X 26 = 676 diagrams are possible, so identificaion of individual diagram is more difficult.

- Frequency analysis is much more difficult.

Hill Cipher

- It is developed by the mathematician Lester Hill in 1929. Hill cipher is a polygraphic substitution cipher based on linear algebra.Each letter is represented by a number modulo 26. Often the simple scheme A = 0, B = 1, …, Z = 25 is used, but this is not an essential feature of the cipher.
- To encrypt a message, each block of n letters (considered as an n-component vector) is multiplied by an invertible n × n matrix, against modulus 26. To decrypt the message, each block is multiplied by the inverse of the matrix used for encryption.
- The matrix used for encryption is the cipher key, and it should be chosen randomly from the set of invertible n × n matrices (modulo 26).
- The hill cipher can be expressed as

$$C = KP \bmod 26$$

Example

Input : Plaintext: ACT

Key: GYBNQKURP

Output : Ciphertext: POH

Encryption

We have to encrypt the message ‘ACT’ (n=3).The key is ‘GYBNQKURP’ which can be written as the nxn matrix:

$$\begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix}$$

The message ‘ACT’ is written as vector:

$$\begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix}$$

The enchipered vector is given as:

$$\begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix} \begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix} = \begin{bmatrix} 67 \\ 222 \\ 319 \end{bmatrix} \equiv \begin{bmatrix} 15 \\ 14 \\ 7 \end{bmatrix} (\text{mod } 26)$$

The Ciphertext is POH

Decryption

- To decrypt the message, we turn the ciphertext back into a vector, then simply multiply by the inverse matrix of the key matrix (IFKVIVVMI in letters).The inverse of the matrix used in the previous example is:

$$\begin{bmatrix} 6 & 24 & 1 \\ 13 & 16 & 10 \\ 20 & 17 & 15 \end{bmatrix}^{-1} \equiv \begin{bmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{bmatrix} (\text{mod } 26)$$

For the previous Ciphertext ‘POH’:

$$\begin{bmatrix} 8 & 5 & 10 \\ 21 & 8 & 21 \\ 21 & 12 & 8 \end{bmatrix} \begin{bmatrix} 15 \\ 14 \\ 7 \end{bmatrix} \equiv \begin{bmatrix} 260 \\ 574 \\ 539 \end{bmatrix} \equiv \begin{bmatrix} 0 \\ 2 \\ 19 \end{bmatrix} (\text{mod } 26)$$

The plaintext is ACT

One Time Pad Cipher

- It is an unbreakable cryptosystem. It represents the message as a sequence of 0s and 1s. this can be accomplished by writing all numbers in binary, for example, or by

using ASCII. The key is a random sequence of 0"s and 1"s of same length as the message.

- Once a key is used, it is discarded and never used again. The system can be expressed as follows:

$$C_i = P_i \oplus K_i$$

C_i - i^{th} binary digit of cipher text

P_i - i^{th} binary digit of plaintext

K_i - i^{th} binary digit of key

$\oplus$ – exclusive OR opearaiton

- Thus the cipher text is generated by performing the bitwise XOR of the plaintext and the key. Decryption uses the same key. Because of the properties of XOR, decryption simply involves the same bitwise operation:

$$P_i = C_i \oplus K_i$$

Example

Encryption

Plaintext is 00101001 and the key is 10101100, we obtain the ciphertext is,

Plaintext	00101001
Key	10101100
Ciphertext	10000101

Decryption

Ciphertext	10000101
Key	10101100
Plaintext	00101001

Advantages

- Encryption method is completely unbreakable.

Disadvantages

- It requires a very long key which is expensive to produce and expensive to transmit.
- Once a key is used it is dangerous to reuse it for second message.

Vigenere Cipher

- Vigenere Cipher is a method of encrypting alphabetic text. It uses a simple form of polyalphabetic substitution. A polyalphabetic cipher is any cipher based on

substitution, using multiple substitution alphabets .The encryption of the original text is done using the Vigenère square or Vigenère table.

- The table consists of the alphabets written out 26 times in different rows, each alphabet shifted cyclically to the left compared to the previous alphabet, corresponding to the 26 possible Caesar Ciphers.
- At different points in the encryption process, the cipher uses a different alphabet from one of the rows.
- The alphabet used at each point depends on a repeating keyword.

Example:

Input: Plaintext: GEEKSFORGEEKS

Keyword: AYUSH

Output: Cipher text: GCYCZFMLYLEIM

For generating key, the given keyword is repeated in a circular manner until it matches the length of the plain text.

The keyword "AYUSH" generates the key "AYUSHAYUSHAYU"

The plain text is then encrypted using the process explained below.

Encryption

- The first letter of the plaintext, G is paired with A, the first letter of the key. So use row G and column A of the Vigenère square, namely G. Similarly, for the second letter of the plaintext, the second letter of the key is used, the letter at row E and column Y is C. The rest of the plaintext is enciphered in a similar fashion.

Table to encrypt Geeks

	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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	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

Decryption

- Decryption is performed by going to the row in the table corresponding to the key, finding the position of the ciphertext letter in this row, and then using the column's label as the plaintext.
- For example, in row A (from AYUSH), the ciphertext G appears in column G, which is the first plaintext letter. Next we go to row Y (from AYUSH), locate the ciphertext C which is found in column E, thus E is the second plaintext letter.
- A more **easy implementation** could be to visualize Vigenère algebraically by converting [A-Z] into numbers [0–25].

Encryption

The plaintext(P) and key(K) are added modulo 26.

$E_i = (P_i + K_i) \bmod 26$

Decryption

$D_i = (E_i - K_i + 26) \bmod 26$

Transposition Techniques

- A very different kind of mapping is achieved by performing some sort of permutation on the plaintext letters. This technique is referred to as a transposition cipher. **transposition technique** rearranges the characters to form a ciphertext

Rail fence

- It is simplest of such cipher, in which the plaintext is written down as sequence of diagonals and then read off as a sequence of rows.
- The rail fence cipher offers essentially no communication security, and it will be shown that it can be easily broken even by hand.

Example

The key for the railfence cipher is just the number of rails. To encrypt a piece of text, e.g. defend the east wall of the castle

We write it out in a special way on a number of rails (the key here is 3)

```
d . . . n . . . e . . . t . . . l . . . h . . . s . . .
. e . e . d . h . e . s . w . l . o . t . e . a . t . e
. . f . . . t . . . a . . . a . . . f . . . c . . . l .
```

The ciphertext is read off along the rows:

dnetlhseedheswloteateftaafcl

With a key of 4

```
d . . . . . t . . . . . t . . . . . f . . . . . s . . .
. e . . . d . h . . . s . w . . . o . t . . . a . t . .
. . f . n . . . e . a . . . a . l . . . h . c . . . l .
. . . e . . . . . e . . . . . l . . . . . e . . . . . e
```

The ciphertext is again read off along the rows:

dttfsedhswotatfneaalhcleelee

Row Transposition Ciphers

- In a **transposition cipher**, the order of the alphabets is re-arranged to obtain the **cipher**-text. The message is written out in **rows** of a fixed length, and then read out again column by column, and the columns are chosen in some scrambled order.

Example

Encryption

Input : Geeks for Geeks

Key = HACK

Output : e kefGsGsrekoe_

Decryption

Input : e kefGsGsrekoe_

Key = HACK

Output : Geeks for Geeks

Encryption

In a transposition cipher, the order of the alphabets is re-arranged to obtain the cipher-text.

1. The message is written out in rows of a fixed length, and then read out again column by column, and the columns are chosen in some scrambled order.
2. Width of the rows and the permutation of the columns are usually defined by a keyword.
3. For example, the word HACK is of length 4 (so the rows are of length 4), and the permutation is defined by the alphabetical order of the letters in the keyword. In this case, the order would be "3 1 2 4".
4. Any spare spaces are filled with nulls or left blank or placed by a character
5. Finally, the message is read off in columns, in the order specified by the keyword.

Encryption

Given text = Geeks for Geeks

Keyword = HACK **Length of Keyword** = 4 (no of rows) **Order of Alphabets in HACK** = 3124

H	A	C	K
3	1	2	4
G	e	e	k
s	_	f	o
r	_	G	e
e	k	s	_

Print Characters of column 1,2,3,4

Encrypted Text = e kefGsGsrekoe_

Steganography

- Steganography is data hidden within data. Steganography is an encryption technique that can be used along with cryptography as an extra-secure method in which to

protect data. At any rate, steganography protects from pirating copyrighted materials as well as aiding in unauthorized viewing.

How is it different from cryptography?

- Cryptography and steganography are both methods used to hide or protect secret data. However, they differ in the respect that cryptography makes the data unreadable, or hides the meaning of the data, while steganography hides the existence of the data.
- In layman's terms, cryptography is similar to writing a letter in a secret language: people can read it, but won't understand what it means. However, the existence of a (probably secret) message would be obvious to anyone who sees the letter, and if someone either knows or figures out your secret language, then your message can easily be read.
- If you were to use steganography in the same situation, you would hide the letter inside a pair of socks that you would be gifting the intended recipient of the letter. To those who don't know about the message, it would look like there was nothing more to your gift than the socks. But the intended recipient knows what to look for, and finds the message hidden in them.
- Similarly, if two users exchanged media files over the internet, it would be more difficult to determine whether these files contain hidden messages, than if they were communicating using cryptography.

Image Steganography

- As the name suggests, Image Steganography refers to the process of hiding data within an image file. The image selected for this purpose is called the cover-image and the image obtained after steganography is called the stego-image.

Working Principle

- An image is represented as an N*M (in case of greyscale images) or N*M*3 (in case of colour images) matrix in memory, with each entry representing the intensity value of a pixel.
- In image steganography, a message is embedded into an image by altering the values of some pixels, which are chosen by an encryption algorithm. The recipient of the image must be aware of the same algorithm in order to known which pixels he or she must select to extract the message.

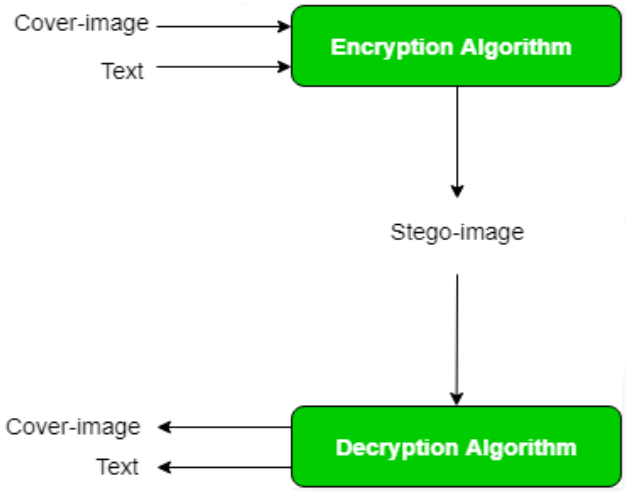


Figure 1.17 Steganography

- Detection of the message within the cover-image is done by the process of steganalysis.
- This can be done through comparison with the cover image, histogram plotting, or by noise detection. While efforts are being invested in developing new algorithms with a greater degree of immunity against such attacks, efforts are also being devoted towards improving existing algorithms for steganalysis, to detect exchange of secret information between terrorists or criminal elements.

1.8 Foundations of modern cryptography

- Modern encryption is the key to advanced computer and communication security. This stream of cryptography is completely based on the ideas of mathematics such as number theory and computational complexity theory as well as concepts of probability.

Characteristics of Modern Cryptography

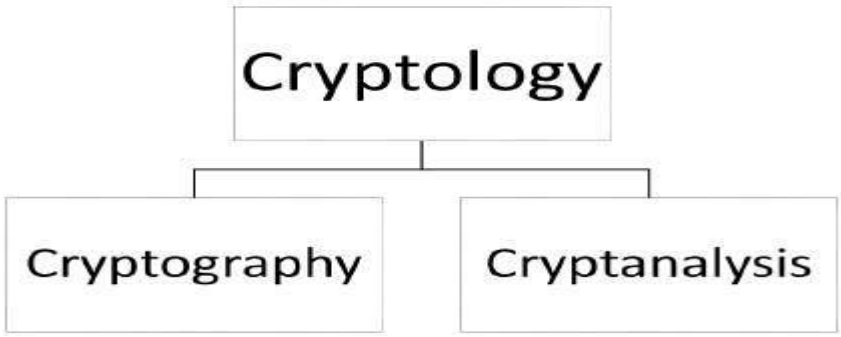
There are four major characteristics that separate modern cryptography from the classical approach.

Traditional Encryption	Modern Encryption
For making ciphertext, manipulation is done in the characters of the plain text.	For making ciphertext, operations are performed on binary bit sequence.
The whole of the ecosystem is required to communicate confidentially.	Here, only the parties who want to execute secure communication possess the secret key.
These are weaker as compared to modern encryption.	The encryption algorithm formed by this encryption technique is stronger as compared to traditional encryption algorithms.
It believes in the concept of security through obscurity.	Its security depends on the publicly known mathematical algorithm.

Context of Cryptography

Cryptology, the study of cryptosystems, can be subdivided into two branches −

- Cryptography
- Cryptanalysis



Cryptography

- Cryptography is the art and science of making a cryptosystem that is capable of providing information security.
- Cryptography deals with the actual securing of digital data. It refers to the design of mechanisms based on mathematical algorithms that provide fundamental information security services.

Cryptanalysis

- The art and science of breaking the cipher text is known as cryptanalysis.

- Cryptanalysis is the sister branch of cryptography and they both co-exist. The cryptographic process results in the cipher text for transmission or storage. It involves the study of cryptographic mechanism with the intention to break them. Cryptanalysis is also used during the design of the new cryptographic techniques to test their security strengths.

Note − Cryptography concerns with the design of cryptosystems, while cryptanalysis studies the breaking of cryptosystems.

Types of Modern Cryptography

- Different algorithms have come up with powerful encryption mechanisms incorporated in them. It gave rise to two new ways of encryption mechanism for data security. These are:
  - Symmetric key encryption
  - Asymmetric key encryption

Key

- It can be a number, word, phrase, or any code that will be used for encrypting as well as decrypting any ciphertext information to plain text and vice versa.
- Symmetric and asymmetric key cryptography is based on the number of keys and the way these keys work. Let us know about both of them in details:

Symmetric key encryption

- Symmetric key encryption technique uses a straight forward method of encryption. Hence, this is the simpler among these two practices. In the case of symmetric key encryption, the encryption is done through only one secret key, which is known as "Symmetric Key", and this key remains to both the parties.
- The same key is implemented for both encodings as well as decoding the information. So, the key is used first by the sender prior to sending the message, and on the receiver side, that key is used to decipher the encoded message.
- One of the good old examples of this encryption technique is Caesar's Cipher. Modern examples and algorithms that use the concept of symmetric key encryption are RC4, QUAD, AES, DES, Blowfish, 3DES, etc.

Asymmetric Key Encryption

- Asymmetric Encryption is another encryption method that uses two keys, which is a new and sophisticated encryption technique. This is because it integrates two cryptographic keys for implementing data security. These keys are termed as Public Key and Private Key.
- The "public key", as the name implies, is accessible to all who want to send an encrypted message. The other is the "private key" that is kept secure by the owner of that public key or the one who is encrypting.
- Encryption of information is done through public key first, with the help of a particular algorithm. Then the private key, which the receiver possesses, will use to decrypt that encrypted information. The same algorithm will be used in both encodings as well as decoding.
- Examples of asymmetric key encryption algorithms are Diffie-Hellman and RSA algorithm.

Security Services of Cryptography

- Confidentiality of information.
- Data Integrity.
- Authentication.
    - Message authentication.
    - Entity authentication.
- Non-repudiation.

Cryptography Primitives

- Cryptography primitives are nothing but the tools and techniques in Cryptography that can be selectively used to provide a set of desired security services −

- Encryption
- Hash functions
- Message Authentication codes (MAC)
- Digital Signatures

The following table shows the primitives that can achieve a particular security service on their own.

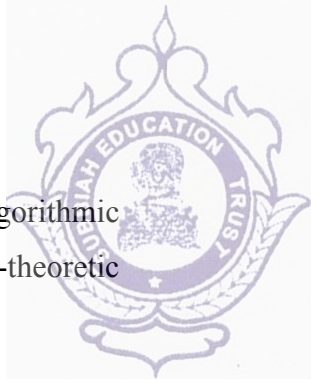
Primitives ⇨ Service ⇩	Encryption	Hash Function	MAC	Digital Signature
Confidentiality	Yes	No	No	No
Integrity	No	Sometimes	Yes	Yes
Authentication	No	No	Yes	Yes
Non Reputation	No	No	Sometimes	Yes

1.8.1 Perfect Security

- Perfect Secrecy (or information-theoretic secure) means that the ciphertext conveys no information about the content of the plaintext. ... However, part of being provably secure is that you need as much key material as you have plaintext to encrypt.

1.8.2 Information Theory

- Information theory studies the quantification, storage, and communication of information.
- It was originally proposed by Claude Shannon in 1948 to find fundamental limits on signal processing and communication operations such as data compression.
- Its impact has been crucial to the success of the Voyager missions to deep space, the invention of the compact disc, the feasibility of mobile phones, the development of the Internet, the study of linguistics and of human perception, the understanding of black holes, and numerous other fields.
- The field is at the intersection of mathematics, statistics, computer science, physics, neurobiology, information engineering, and electrical engineering.
- The theory has also found applications in other areas, including statistical inference, natural language processing, cryptography, neurobiology, human vision, the evolution and function of molecular codes (bioinformatics), model selection in statistics, thermal physics, quantum computing, linguistics, plagiarism detection, pattern recognition, and anomaly detection.



- Important sub-fields of information theory include source coding, algorithmic complexity theory, algorithmic information theory, information-theoretic security, Grey system theory and measures of information.
- Applications of fundamental topics of information theory include lossless data compression (e.g. ZIP files), lossy data compression (e.g. MP3s and JPEGs), and channel coding (e.g. for DSL).
- Information theory is used in information retrieval, intelligence gathering, gambling, and even in musical composition.
- A key measure in information theory is entropy. Entropy quantifies the amount of uncertainty involved in the value of a random variable or the outcome of a random process. For example, identifying the outcome of a fair coin flip (with two equally likely outcomes) provides less information (lower entropy) than specifying the outcome from a roll of a die (with six equally likely outcomes). Some other important measures in information theory are mutual information, channel capacity, error exponents, and relative entropy.

Product Cryptosystems

- A product cipher combines two or more transformations in a manner intending that the resulting cipher is more secure than the individual components to make it resistant to cryptanalysis.
- The product cipher combines a sequence of simple transformations such as substitution (S-box), permutation (P-box), and modular arithmetic.
- For transformation involving reasonable number of n message symbols, both of the foregoing cipher systems (the S-box and P-box) are by themselves wanting.
- The combination could yield a cipher system more powerful than either one alone. This approach of alternatively applying substitution and permutation transformation has been used by IBM in the Lucifer cipher system, and has become the standard for national data encryption standards such as the Data Encryption Standard and the Advanced Encryption Standard.
- A product cipher that uses only substitutions and permutations is called a SP-network. Feistel ciphers are an important class of product ciphers.

1.10 Cryptanalysis

- Cryptanalysis is the art of trying to decrypt the encrypted messages without the use of the key that was used to encrypt the messages. Cryptanalysis uses mathematical analysis & algorithms to decipher the ciphers.
- The success of cryptanalysis attacks depends

  - Amount of time available
  - Computing power available
  - Storage capacity available

The following is a list of the commonly used Cryptanalysis attacks;

- **Brute force attack**– this type of attack uses algorithms that try to guess all the possible logical combinations of the plaintext which are then ciphered and compared against the original cipher.
- **Dictionary attack**– this type of attack uses a wordlist in order to find a match of either the plaintext or key. It is mostly used when trying to crack encrypted passwords.
- **Rainbow table attack**– this type of attack compares the cipher text against pre-computed hashes to find matches.

Other Attacks using Cryptanalysis

- Known-Plaintext Analysis (KPA): Attacker decrypt ciphertexts with known partial plaintext.
- Chosen-Plaintext Analysis (CPA): Attacker uses ciphertext that matches arbitrarily selected plaintext via the same algorithm technique.
- Ciphertext-Only Analysis (COA): Attacker uses known ciphertext collections.
- Man-in-the-Middle (MITM) Attack: Attack occurs when two parties use message or key sharing for communication via a channel that appears secure but is actually compromised. Attacker employs this attack for the interception of messages that pass through the communications channel. Hash functions prevent MITM attacks.
- Adaptive Chosen-Plaintext Attack (ACPA): Similar to a CPA, this attack uses chosen plaintext and ciphertext based on data learned from past encryptions.

2

SYMMETRIC KEY CRYPTOGRAPHY

MATHEMATICS OF SYMMETRIC KEY CRYPTOGRAPHY: Algebraic structures - Modular arithmetic-Euclid‟s algorithm- Congruence and matrices - Groups, Rings, Fields- Finite fields- SYMMETRIC KEY CIPHERS: SDES – Block cipher Principles of DES – Strength of DES – Differential and linear cryptanalysis - Block cipher design principles – Block cipher mode of operation – Evaluation criteria for AES – Advanced Encryption Standard - RC4 – Key distribution.

2.1 Mathematics of Symmetric Key Cryptography

- Cryptology is the mathematics, such as number theory, and the application of formulas and algorithms, that underpin cryptography and cryptanalysis.
- Since the cryptanalysis concepts are highly specialized and complex, we concentrate here only on some of the key mathematical concepts behind cryptography.
- In order for data to be secured for storage or transmission, it must be transformed in such a manner that it would be difficult for an unauthorized individual to be able to discover its true meaning.
- To do this, certain mathematical equations are used, which are very difficult to solve unless certain strict criteria are met. The level of difficulty of solving a given equation is known as its intractability. These types of equations form the basis of cryptography.
- Symmetric ciphers use symmetric algorithms to encrypt and decrypt data. These ciphers are used in symmetric key cryptography.
- A symmetric algorithm uses the same key to encrypt data as it does to decrypt data. 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.

2.2 Algebraic structures

- Cryptography requires sets of integers and specific operations that are defined for those sets. The combination of the set and the operations that are applied to the elements of the set is called an algebraic structure. Figure 2.1 shows the common algebraic structures.

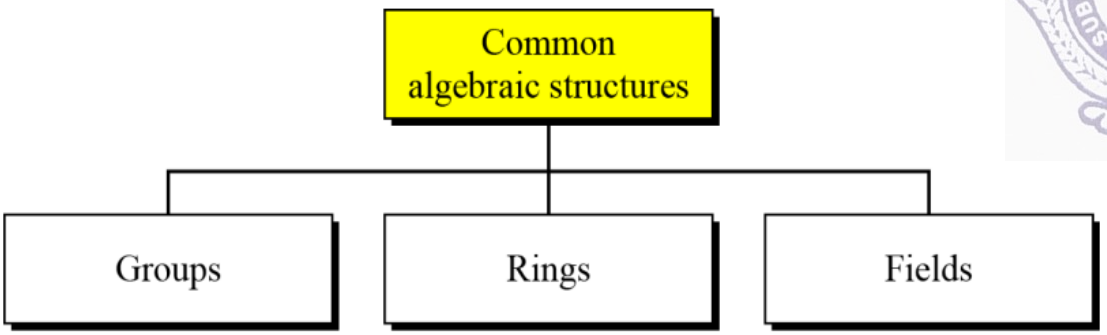


Figure 2.1 Common Algebraic Structures

2.2.1 Groups

- A group is an algebraic structure conssting of a set of elements together with an operation that combines any two elements to form a third element.
- A **group** G, sometimes denoted by $\{G, \cdot\}$ is a set of elements with a binary operation, denoted by $\cdot$,that associates to each ordered pair (a, b) of elements in G an element $(a \cdot b)$ in G, such that the following axioms are obeyed:
  - **Closure:** If a and b belong to G, then $a \cdot b$ is also in G.
  - **Associative:** $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ for all a, b, c in G.
  - **Identity element:** There is an element e in G such that $a \cdot e = e \cdot a = a$ for all a in G.
  - **Inverse element:** For each a in G there is an element a' in G such that $a \cdot a' = a' \cdot a = e$.
- If a group has a finite number of elements, it is referred to as a **finite group**, and the **order** of the group is equal to the number of elements in the group. Otherwise, the group is an **infinite group**.
- A group is said to be **abelian** if it satisfies the following additional condition:
  - **Commutative:** $a \cdot b = b \cdot a$ for all a, b in G.

Cyclic Subgroups

- If a subgroup of a group can be generated using the power of an element, the subgroup is called the cyclic subgroup.

$$a^n \longrightarrow a \bullet a \bullet \ldots \bullet a \quad (n \text{ times})$$

Example 1:

Four cyclic subgroups can be made from the group G = $<Z_6, +>$. There are $H_1 = <\{0\}, +>$, $H_2 = <\{0, 2, 4\}, +>$, $H_3 = <\{0, 3\}, +>$ and $H_4 = G$.

$0^0 \bmod 6 = 0$

$1^0 \bmod 6 = 0$
$1^1 \bmod 6 = 1$
$1^2 \bmod 6 = (1 + 1) \bmod 6 = 2$
$1^3 \bmod 6 = (1 + 1 + 1) \bmod 6 = 3$
$1^4 \bmod 6 = (1 + 1 + 1 + 1) \bmod 6 = 4$
$1^5 \bmod 6 = (1 + 1 + 1 + 1 + 1) \bmod 6 = 5$

$2^0 \bmod 6 = 0$
$2^1 \bmod 6 = 2$
$2^2 \bmod 6 = (2 + 2) \bmod 6 = 4$

$3^0 \bmod 6 = 0$
$3^1 \bmod 6 = 3$

$4^0 \bmod 6 = 0$
$4^1 \bmod 6 = 4$
$4^2 \bmod 6 = (4 + 4) \bmod 6 = 2$

$5^0 \bmod 6 = 0$
$5^1 \bmod 6 = 5$
$5^2 \bmod 6 = 4$
$5^3 \bmod 6 = 3$
$5^4 \bmod 6 = 2$
$5^5 \bmod 6 = 1$

Example 2:

Three cyclic subgroups can be made from the group G = $<Z_{10}^*, x>$. G has only four elements: 1, 3, 7 and 9. The Cyclic sub groups are $H_1 = <\{1\}, x>$, $H_2 = <\{1, 9\}, x>$, $H_3 = G$.

$1^0 \bmod 10 = 1$

$3^0 \bmod 10 = 1$
$3^1 \bmod 10 = 3$
$3^2 \bmod 10 = 9$
$3^3 \bmod 10 = 7$

$7^0 \bmod 10 = 1$
$7^1 \bmod 10 = 7$
$7^2 \bmod 10 = 9$
$7^3 \bmod 10 = 3$

$9^0 \bmod 10 = 1$
$9^1 \bmod 10 = 9$

2.2.2 Rings

- A ring R, sometimes denoted by {R, +, x}, is a set of elements with two binary operations, called addition and multiplication, such that for all a, b, c in R the following axioms are obeyed:
  - **Closure under multiplication:** If a and b belong to R, then ab is also in R.
  - **Associativity of multiplication:** a(bc) = (ab)c for all a, b, c in R.
  - **Distributive laws:**
    - a (b + c) = ab + ac for all a, b, c in R.

- (a + b) c = ac + bc for all a, b, c in R.

- A ring is said to be **commutative** if it satisfies the following additional condition:
  - **Commutativity of multiplication:** $ab = ba$ for all a, b in R.
- An **integral domain**, which is a commutative ring that obeys the following axioms:
  - **Multiplicative identity:** There is an element 1 in R such that $a1 = 1a = a$ for all a in R.
  - **No zero divisors:** If a, b in R and $ab = 0$, then either $a = 0$ or $b = 0$.

2.2.3 Fields

- A field F, sometimes denoted by {F, +, x}, is a set of elements with two binary operations, called addition and multiplication, such that for all a, b, c in F the following axioms are obeyed:
- F is an integral domain; that is, F satisfies axioms of Groups and Rings.
  - Multiplicative inverse: For each a in F, except 0, there is an element a^{-1} in F such that $aa^{-1} = (a^{-1})\ a = 1$.
- A field is a set in which we can do addition, subtraction, multiplication, and division without leaving the set. Division is defined with the following rule:
  - $a/b = a(b^{-1})$

2.3 Finite Fields

- Finite Fields, also known as Galois Fields, are cornerstones for understanding any cryptography.
- A field can be defined as a set of numbers that we can add, subtract, multiply and divide together and only ever end up with a result that exists in our set of numbers.
- Galois showed that for a field to be finite, the number of elements should be p^n, where p is a prime and n is a positive integer.
- A Galois field, $GF(p^n)$, is a finite field with p^n elements.

2.3.1 Finite Fields of Order p

- For a given prime, p, the finite field of order p, GF(p) is defined as the set Zp of integers {0, 1,..., p- 1}, together with the arithmetic operations modulo p.
- A very common field in this category is GF(2) with the set {0, 1} and two operations, addition and multiplication, as shown in Figure 2.2

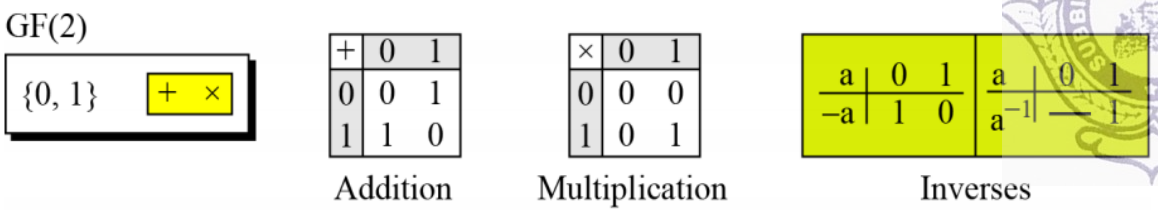


Figure 2.2 GF(2) field

- We can define GF(5) on the set Z5 (5 is a prime) with addition and multiplication operators as shown in Figure 2.3.

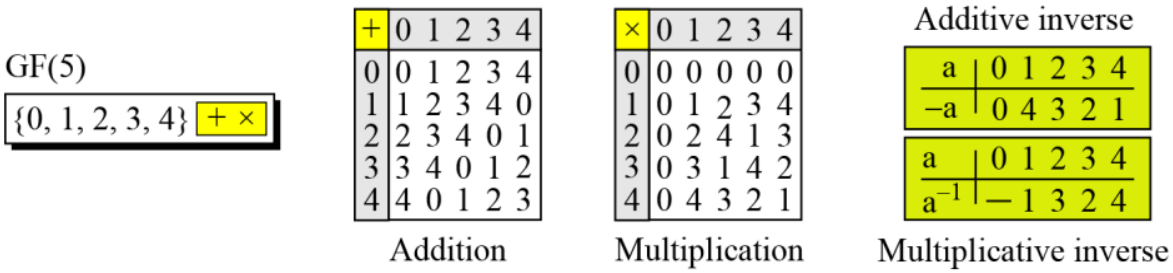


Figure 2.3 GF(5) field

2.4 Modular Arithmetic

- Modular arithmetic is a system of arithmetic for integers, where values reset to zero and begin to increase again, after reaching a certain predefined value, called the modulus (modulo). Modular arithmetic is widely used in computer science and cryptography.

2.4.1 The Modulus

- If *a* is an integer and *n* is a positive integer, we define *a* mod *n* to be the remainder when *a* is divided by *n*. The integer *n* is called the modulus.

Example

- 11 mod 7 = 4
- -5 mod 3 = 1

By using counter clockwise method

Let n = 3 and p = -5 so the values are taken 0, 1, 2. If these values are put in clockwise the numbers are 2 1 0 1 2 because p is 5. The starting from the counter clockwise direction the values are moved to n = 3, so it stop at 1. So, the answer is 1.

2.4.2 Congruence

- Congruences are an important and useful tool for the study of divisibility.
- If a and b are integers and n > 0, we write

$$a \equiv b \bmod n$$

to mean n| (b − a). We read this as "a is congruent to b modulo (or mod) n.

Properties of congruences

- $a \equiv a \pmod n$
- if $a \equiv b \pmod n$ then $b \equiv a \pmod n$
- if $a \equiv b \pmod n$ and $b \equiv c \pmod n$, then $a \equiv c \pmod n$

Example 1: $29 \equiv 8 \bmod 7$, and $60 \equiv 0 \bmod 15$.

- The notation is used because the properties of congruence "≡" are very similar to the properties of equality "=".

Example 2: $38 \equiv 14 \pmod{12}$

- Because $38 - 14 = 24$, which is a multiple of 12, or, equivalently, because both 38 and 14 have the same remainder 2 when divided by 12.
- The same rule holds for negative values:
  - $-8 \equiv 7 \pmod 5$
  - $2 \equiv -3 \pmod 5$
  - $-3 \equiv -8 \pmod 5$

Example 3: Find $17^{341} \bmod 5$.

We have $17 \equiv 2 \bmod 5$

Squaring, we have

$$17^2 \equiv 4 \equiv -1 \bmod 5$$

Squaring again, we find

$$17^4 \equiv 1 \bmod 5$$

Now, 1 to any power is 1, so we raise this last congruence to the 85th power. Why 85? Just wait a moment to find out. We then find

$$17^{340} \equiv 1 \bmod 5$$

Finally, multiply by the first congruence to obtain

$$17^{341} \equiv 2 \bmod 5$$

So, the required remainder is 2.

The strategy is to find some power of 17 to be 1 mod 5. Here, the power 4 worked. The we divided 4 into 341 to get a quotient 85, and this is the power we used on the congruence $17^4 \equiv 1 \bmod 5$. Note also the little trick of replacing 4 by $-1 \bmod 5$. This gives an easier number to square.

2.5 Modular Arithmetic Operations

Properties of Modular Arithmetic

1. [(a mod n) + (b mod n)] mod n = (a + b) mod n

2. [(a mod n) (b mod n)] mod n = (a b) mod n

3. [(a mod n) x (b mod n)] mod n = (a x b) mod n

2.5.1 Modular Addition

- Add two numbers
- Divide the sum and find modular

Example

Given a = 35, b = 10 and n = 12

(a + b) mod n

= (35 + 10) mod 12

= 45 mod 12

= 9

2.5.2 Modular Subtraction

- Subtract two numbers
- Find the mod value

Example 1:

a = 25, b = 8, and n = 12

(a - b) mod n

= (25 - 10) mod 12

= 17 mod 12

= 5

Example 2:

a = 11, b = 50, and n = 15

(a - b) mod n

= (11 - 50) mod 15

= -39 mod 15

= 6

2.5.3 Modular Multiplication

- Multiple two numbers
- Find the mod value

Example

a = 5, b = 8, and n = 12

= 40 mod 12

= 4

2.5.4 Modular Division

Example

Compute 5/7 mod 12

x = 5/7 mod 12

7x = 5 mod 12

Here, x takes the values from 0 to 11

If we put x = 11, we get

(7 x 11) mod 12 = 5 mod 12

= 77 mod 12

= 5

2.6 Euclid's algorithm

- The Euclid's algorithm (or Euclidean Algorithm) is a method for efficiently finding the greatest common divisor (GCD) of two numbers. The GCD of two integers X and Y is the largest number that divides both of X and Y (without leaving a remainder).
- For every non-negative integer, a and any positive integer b

gcd (a, b) = gcd (b, a mod b)

Example 1:

gcd (55, 22) = gcd (22, 55 mod 22)

= gcd (22, 11)

= gcd (11, 22 mod 11)

= gcd (11, 0)

gcd (55, 22) is 11

Example 2:

gcd (30, 50) = gcd (50, 30 mod 50)

= gcd (50, 30)

= gcd (30, 50 mod 30)

= gcd (30, 20)

= gcd (20, 30 mod 20)

= gcd (20, 10)

= gcd (10, 20 mod 10)

= gcd (10, 0)

gcd (30, 50) is 10

Another Method

To find gcd(1970, 1066)		
1970	= 1 x 1066 + 904	gcd(1066, 904)
1066	= 1 x 904 + 162	gcd(904, 162)
904	= 5 x 162 + 94	gcd(162, 94)
162	= 1 x 94 + 68	gcd(94, 68)
94	= 1 x 68 + 26	gcd(68, 26)
68	= 2 x 26 + 16	gcd(26, 16)
26	= 1 x 16 + 10	gcd(16, 10)
16	= 1 x 10 + 6	gcd(10, 6)
10	= 1 x 6 + 4	gcd(6, 4)
6	= 1 x 4 + 2	gcd(4, 2)
4	= 2 x 2 + 0	gcd(2, 0)
Therefore, gcd(1970, 1066) = 2		

Examples:

Find the GCD

- GCD (12, 8)
- GCD (200, 1000)
- GCD (7, 122)

2.7 Symmetric Key Ciphers

- Symmetric ciphers use the same cryptographic keys for both encryption of plaintext and decryption of ciphertext. They are faster than asymmetric ciphers and allow encrypting large sets of data. However, they require sophisticated mechanisms to securely distribute the secret keys to both parties

Definition A symmetric cipher defined over (K, M, C), where:

- K - a set of all possible keys,
- M - a set of all possible messages,
- C - a set of all possible ciphertexts

is a pair of efficient algorithms (E, D), where:

- $E: K \times M \rightarrow C$
- $D: K \times C \rightarrow M$

such that for every m belonging to M, k belonging to K there is an equality:

- $D(k, E(k, m)) = m$ (the consistency rule)

➡ *Function E is often randomized*

➡ *Function D is always deterministic*

Types of keys are used in symmetric key cryptography

- Symmetric encryption (figure 2.4) uses a single key that needs to be shared among the people who need to receive the message while asymmetrical encryption uses a pair of public key and a private key to encrypt and decrypt messages when communicating.

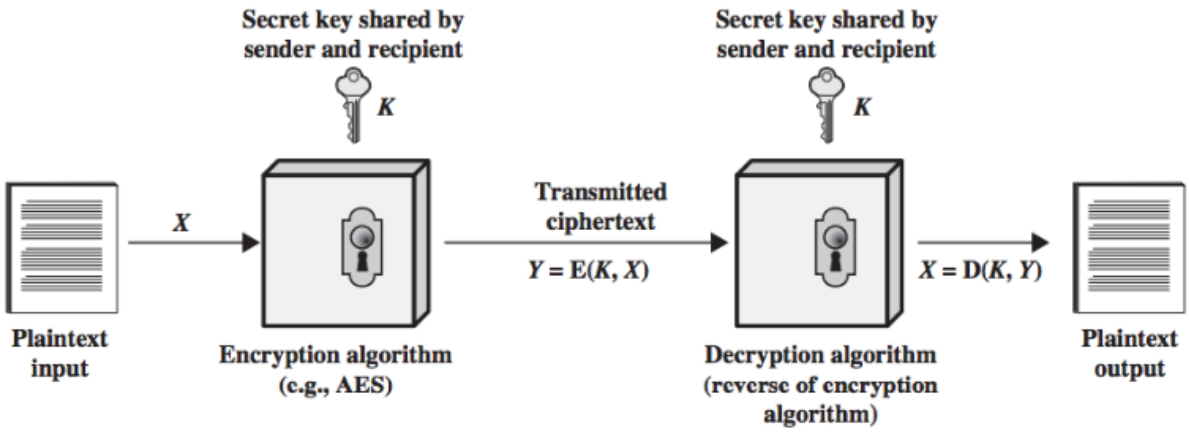


Figure 2.4 Simplified Model of Symmetric Encryption

2.8 Simplified Data Encryption Standard (S-DES)

- The overall structure of the simplified DES shown in Figure 2.5. The S-DES encryption algorithm takes an 8-bit block of plaintext (example: 10111101) and a 10-bit key as input and produces an 8-bit block of ciphertext as output.
- The S-DES decryption algorithm takes an 8-bit block of ciphertext and the same 10-bit key used to produce that ciphertext as input and produces the original 8-bit block of plaintext.

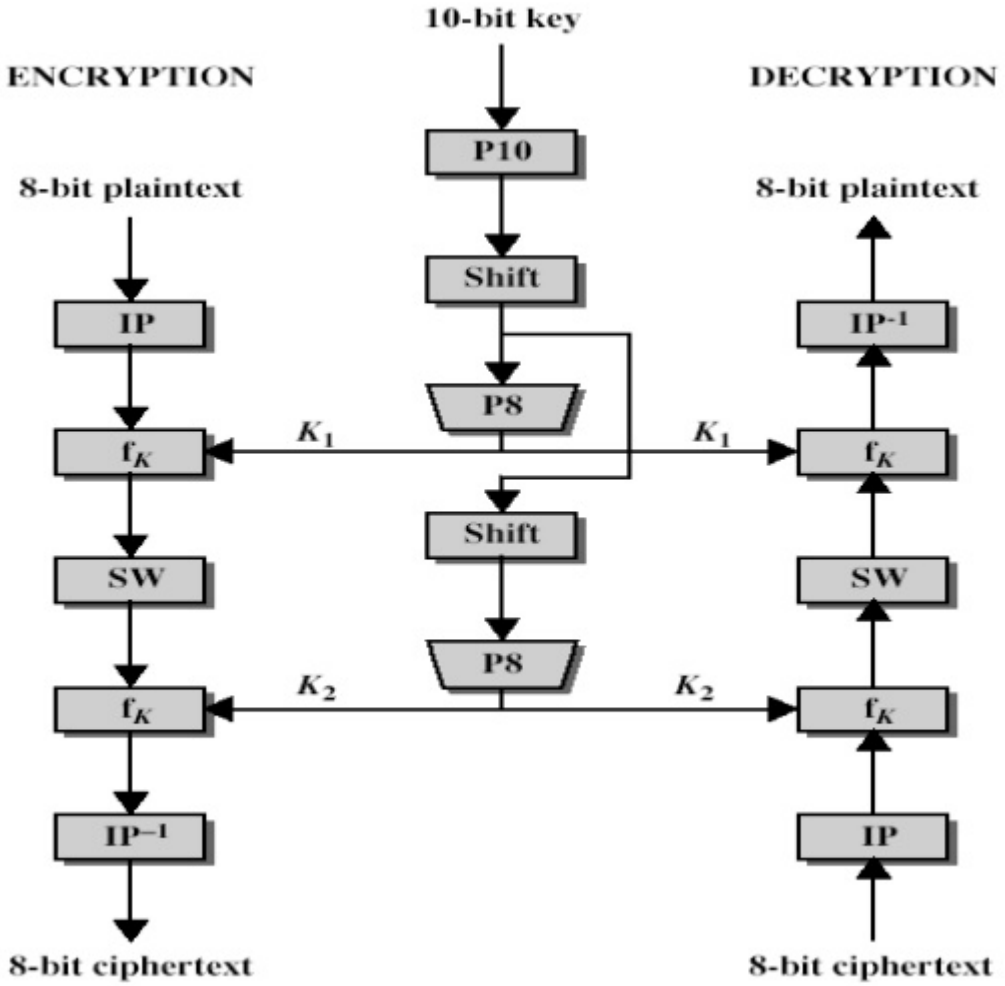


Figure 2.5 Overview of S-DES Algorithm

The encryption algorithm involves five functions:

- An initial permutation (IP)
- A complex function labeled f_k, which involves both permutation and substitution operations and depends on a key input.

- A simple permutation function that switches (SW) the two halves of the data.
- The function f_k again.
- A permutation function that is the inverse of the initial permutation

- The function f_k takes as input not only the data passing through the encryption algorithm, but also an 8-bit key. Here a 10-bit key is used from which two 8-bit subkeys are generated.
- The key is first subjected to a permutation (P10). Then a shift operation is performed. The output of the shift operation then passes through a permutation function that produces an 8-bit output (P8) for the first subkey (K1).
- The output of the shift operation also feeds into another shift and another instance of P8 to produce the second subkey (K2).
- The encryption algorithm can be expressed as a composition composition1 of functions:

IP^{-1} o f_{K2} o SW o f_{k1} o IP, which can also be written as

Ciphertext = IP^{-1} (f_{K2} (SW (f_{k1} (IP (plaintext)))))

Where

  - K1 = P8 (Shift (P10 (Key)))
  - K2 = P8 (Shift (shift (P10 (Key))))

- Decryption can be shown as Plaintext = IP^{-1} (f_{K1} (SW (f_{k2} (IP (ciphertext)))))

2.8.1 S-DES Key Generation

- S-DES depends on the use of a 10-bit key shared between sender and receiver. From this key, two 8-bit subkeys are produced for use in particular stages of the encryption and decryption algorithm.(Figure 2.6)

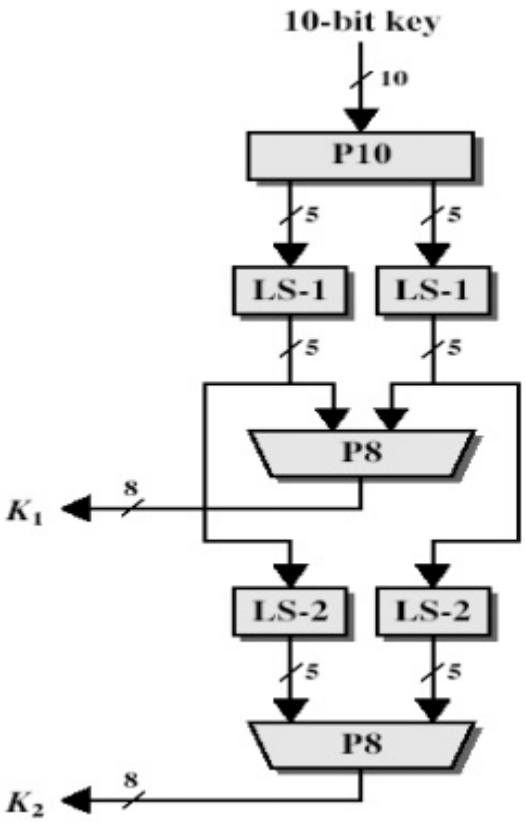


Figure 2.6 S-DES Key Generation

- First, permute the key in the following fashion. Let the 10-bit key be designated as (k1, K2, k3, k4, k5, k6, k7, k8, k9, k10). Then the permutation P10 is defined as:

  P10 (k1, K2, k3, k4, k5, k6, k7, k8, k9, k10) = (k3, k5, K2, k7, k4, k10 10, k1, k9, k8, k6).

- P10 can be concisely defined by the display:

P10									
3	5	2	7	4	10	1	9	8	6

- This table is read from left to right; each position in the table gives the identity of the input bit that produces the output bit in that position. So, the first output bit is bit 3 of the input; the second output bit is bit 5 of the input, and so on.

Example

- The 10 bit key is (1010000010), now find the permutation from P10 for this key so it becomes (10000 01100).

- Next, perform a circular left shift (LS-1), or rotation, separately on the first five bits and the second five bits. In our example, the result is (00001 11000).
- Next, apply P8, which picks out and permutes 8 of the 10 bits according to the following rule:

P8							
6	3	7	4	8	5	10	9

- So, The result is subkey 1 (K1). In our example, this yield (10100100).
- Then go back to the pair of 5-bit strings produced by the two LS-1 functions and performs a circular left shift of 2 bit positions on each string. In our example, the value (00001 11000) becomes (00100 00011).
- Finally, P8 is applied again to produce K2. In our example, the result is (01000011).

2.8.2 S-DES Encryption

- Encryption involves the sequential application of five functions (Figure 2.7).

1. **Initial Permutations**

- The input to the algorithm is an 8-bit block of plaintext, which we first permute using the IP function

IP							
2	6	3	1	4	8	5	7

The plaintext is 10111101

Permutated output is 01111110

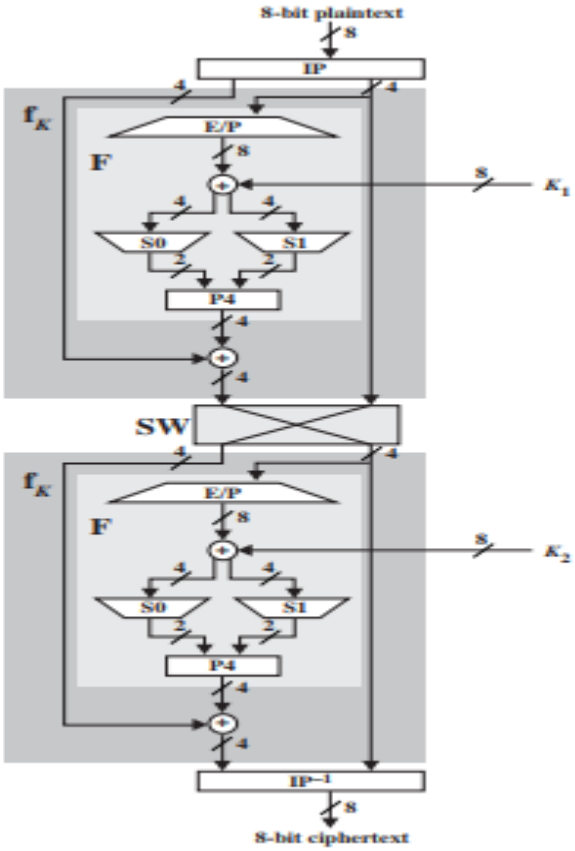


Figure 2.7 S-DES Encryption

2. **The Function f_k**

- The most complex component of S-DES is the function f_k, which consists of a combination of permutation and substitution functions. The functions can be expressed as follows. Let L and R be the leftmost 4 bits and rightmost 4 bits of the 8-bit input to f K, and let F be a mapping (not necessarily one to one) from 4-bit strings to 4-bit strings. Then we let

$$F_k (L, R) = (L \oplus F (R, SK), R)$$

Where SK is a sub key and ⊕ is the bit-by- bit exclusive OR function

- Now, describe the mapping F. The input is a 4-bit number (n1 n2 n3 n4). The first operation is an expansion/permutation operation:

E/P							
4	1	2	3	2	3	4	1

- Now, find the E/P from IP

  IP = 01111110, it becomes

  E/P = 01111101

- Now, XOR with K_1

  => 01111101 ⊕ 10100100 = 11011001

- The first 4 bits (first row of the preceding matrix) are fed into the S-box S0 to produce a 2- bit output, and the remaining 4 bits (second row) are fed into S1 to produce another 2-bit output.

- These two boxes are defined as follows:

$$S0 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 1 & 0 & 3 & 2 \\ 3 & 2 & 1 & 0 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 3 & 2 \end{bmatrix} \end{array} \qquad S1 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \begin{bmatrix} 0 & 1 & 2 & 3 \\ 2 & 0 & 1 & 3 \\ 3 & 0 & 1 & 0 \\ 2 & 1 & 0 & 3 \end{bmatrix} \end{array}$$

- The S-boxes operate as follows. The first and fourth input bits are treated as a 2-bit number that specify a row of the S-box, and the second and third input bits specify a column of the S-box. Each s box gets 4-bit input and produce 2 bits as output. It follows 00- 0, 01-1, 10-2, 11-3 scheme.

Here, take first 4 bits,

S_0 => 1101

11 - > 3

10 -> 2

=> 3 =>11

Second 4 bits

S1 => 1001

11 -> 3

00 -> 0

= > 2 => 10

So, we get 1110

- Now, find P_4

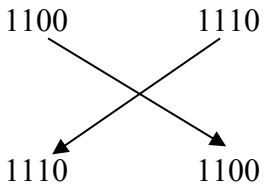
P4			
2	4	3	1

After P_4, the value is 1011

Now, XOR operation 1011⊕ 0111 => 1100

3. **The Switch function**

- The switch function (sw) interchanges the left and right 4 bits.



4. Second function f_k

- First, do E/P function and XOR with K_2, the value is 01101001⊕01000011, the answer is 00101010
- Now, find S_0 and S_1

S_0 => 00 - > 0, 01 -> 1 } => 0 = 00

S_1 = > 10 -> 2, 01 -> 1 } = > 0 => 00

Value is 0000

- Now, find P_4 and XOR operation

After P_4 => 0000 ⊕ 1110 = 1110, then concatenate last 4 bits after interchange in sw.

- Now value is 11101100

5. Find IP^{-1}

IP –1							
4	1	3	5	7	2	8	6

So, value is 01110101

The Ciphertext is 01110101

2.8.3 S-DES Decryption

- Decryption involves the sequential application of five functions.

1. **Find IP**
    - After IP, value is 11101100
2. **Function f_k**
    - After step 2, the answer is 11101100
3. **Swift**
    - The answer is 11001110
4. **Second f_k**
    - The answer is 01111110
5. **Find IP-1**
    - **101111101 -> Plaintext**

2.9 Block Cipher Principles

- All symmetric block encryption algorithms in current use are based on a structure referred to as Fiestel block cipher.

Stream Ciphers and Block Ciphers

- **A stream cipher** is one that encrypts a digital data stream one bit or one byte at a time. E.g, vigenere cipher. Figure (2.8a)
- **A block cipher** is one in which a block of plaintext is treated as a whole and used to produce a cipher text block of equal length. Typically, a block size of 64 or 128 bits is used. Figure (2.8b)

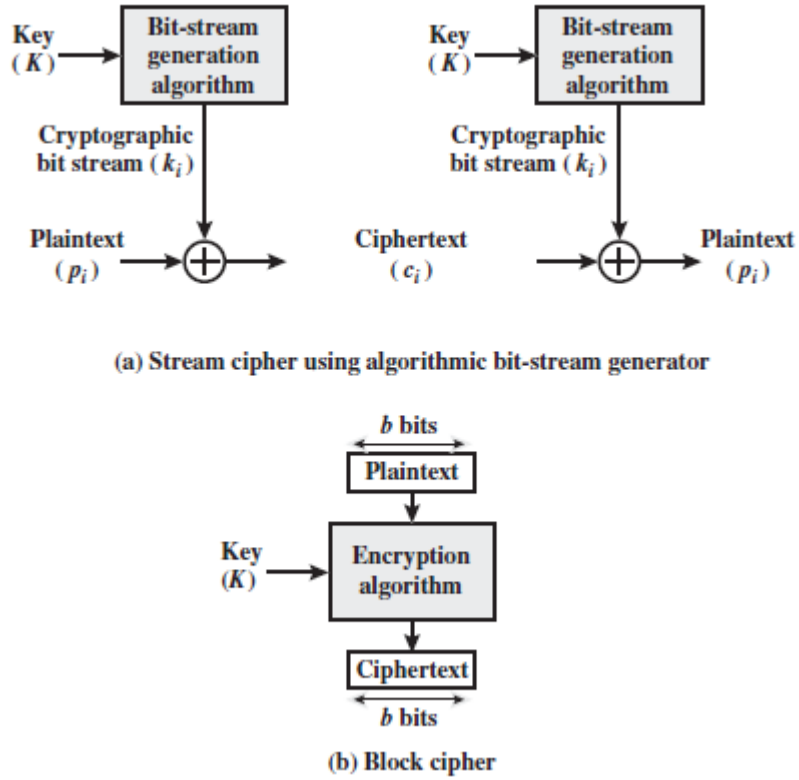


Figure 2.8 Stream Cipher and Block Cipher

- Many block ciphers have a Feistel structure. Such a structure consists of a number of identical rounds of processing.
- In each round, a substitution is performed on one half of the data being processed, followed by a permutation that interchanges the two halves.
- The original key is expanded so that a different key is used for each round.
- The Data Encryption Standard (DES) has been the most widely used encryption algorithm. It exhibits the classic Feistel structure.
- The DES uses a 64-bit block and a 56-bit key. Two important methods of cryptanalysis are differential cryptanalysis and linear cryptanalysis. DES has been shown to be highly resistant to these two types of attack.
- A block cipher operates on a plaintext block of n bits to produce a ciphertext block of n bits. There are possible different plaintext blocks and, for the encryption to be reversible (i.e., for decryption to be possible), each must produce a unique ciphertext block. Such a transformation is called reversible, or nonsingular
- In particular, Feistel proposed the use of a cipher that alternates substitutions and permutations, where these terms are defined as follows:

- **Substitution:** Each plaintext element or group of elements is uniquely replaced by a corresponding ciphertext element or group of elements.
- **Permutation:** A sequence of plaintext elements is replaced by a permutation of that sequence. That is, no elements are added or deleted or replaced in the sequence, rather the order in which the elements appear in the sequence is changed.

➢ Two methods for frustrating statistical cryptanalysis are:

- **Diffusion** – Each plaintext digit affects many ciphertext digits, or each ciphertext digit is affected by many plaintext digits.
- **Confusion** – Make the statistical relationship between a plaintext and the corresponding ciphertext as complex as possible in order to thread attempts to deduce the key.

2.9.1 Feistel cipher structure

➢ The left-hand side of figure 2.9 depicts the structure proposed by Feistel.

➢ The input to the encryption algorithm is a plaintext block of length 2w bits and a key K. the plaintext block is divided into two halves L_0 and R_0.

➢ The two halves of the data pass through n rounds of processing and then combine to produce the ciphertext block. Each round i has inputs L_{i-1} and R_{i-1}, derived from the previous round, as well as the subkey K_i, derived from the overall key K.

➢ In general, the subkeys K_i are different from K and from each other. All rounds have the same structure.

➢ A substitution is performed on the left half of the data (as similar to S-DES). This is done by applying a round function F to the right half of the data and then taking the XOR of the output of that function and the left half of the data.

➢ The round function has the same general structure for each round but is parameterized by the round subkey k_i. Following this substitution, a permutation is performed that consists of the interchange of the two halves of the data.

➢ This structure is a particular form of the substitution-permutation network.

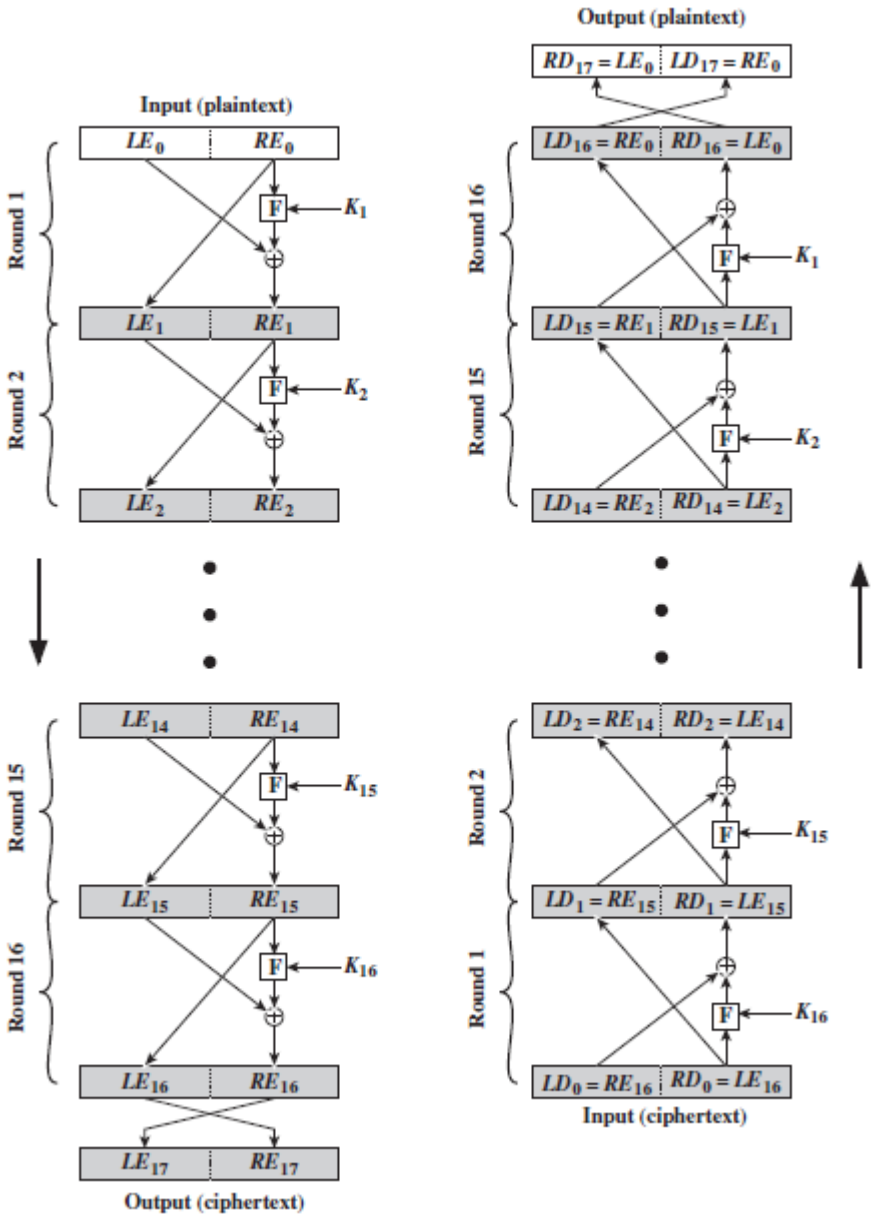


Figure 2.9 Feistel Encryption and Decryption (16 rounds)

The features of Feistel network are:

- **Block size** - Increasing size improves security, but slows cipher
- **Key size** - Increasing size improves security, makes exhaustive key searching harder, but may slow cipher
- **Number of rounds** - Increasing number improves security, but slows cipher
- **Subkey generation** - Greater complexity can make analysis harder, but slows cipher

- **Round function** - Greater complexity can make analysis harder, but slows cipher

- The process of decryption is essentially the same as the encryption process.
- The rule is as follows: use the cipher text as input to the algorithm, but use the subkey k_i in reverse order. i.e., k_n in the first round, k_{n-1} in second round and so on.
- For clarity, we use the notation LE_i and RE_i for data traveling through the decryption algorithm and LD_i and RD_i.
- The above diagram indicates that, at each round, the intermediate value of the decryption process is same (equal) to the corresponding value of the encryption process with two halves of the value swapped.

$$\text{i.e., } RE_i \,||\, LE_i \text{ (or) equivalently } RD_{16-i} \,||\, LD_{16-i}$$

- After the last iteration of the encryption process, the two halves of the output are swapped, so that the cipher text is $RE_{16} \,||\, LE_{16}$.
- The output of that round is the cipher text. Now take the cipher text and use it as input to the same algorithm.
- The input to the first round is $RE_{16} \,||\, LE_{16}$, which is equal to the 32-bit swap of the output of the sixteenth round of the encryption process.
- Now we will see how the output of the first round of the decryption process is equal to a 32-bit swap of the input to the sixteenth round of the encryption process.
- First consider the encryption process,

$$LE_{16} = RE_{15}$$

$$RE_{16} = LE_{15} \oplus F(RE_{15}, K_{16})$$ On the decryption side,

$$LD_1 = RD_0 = LE_{16} = RE_{15}$$

$$RD_1 = LD_0 \oplus F(RD_0, K_{16})$$

$$= RE_{16} \oplus F(RE_{15}, K_{16})$$

$$= [LE_{15} \oplus F(RE_{15}, K_{16})] \oplus F(RE_{15}, K_{16})$$

$$= LE_{15}$$

Therefore, $LD_1 = RE_{15}$ $RD_1 =$

$$LE_{15}$$

In general, for the i^{th} iteration of the encryption algorithm,

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

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

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

- Finally, the output of the last round of the decryption process is $RE_0 \| LE_0$. A 32-bit swap recovers the original plaintext.

2.10 Data Encryption Standard (DES)

- The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).
- DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm. These 8 bits can be used as parity bits or simply set arbitrarily
- The general Structure of DES is depicted in the following illustration −Figure 2.10

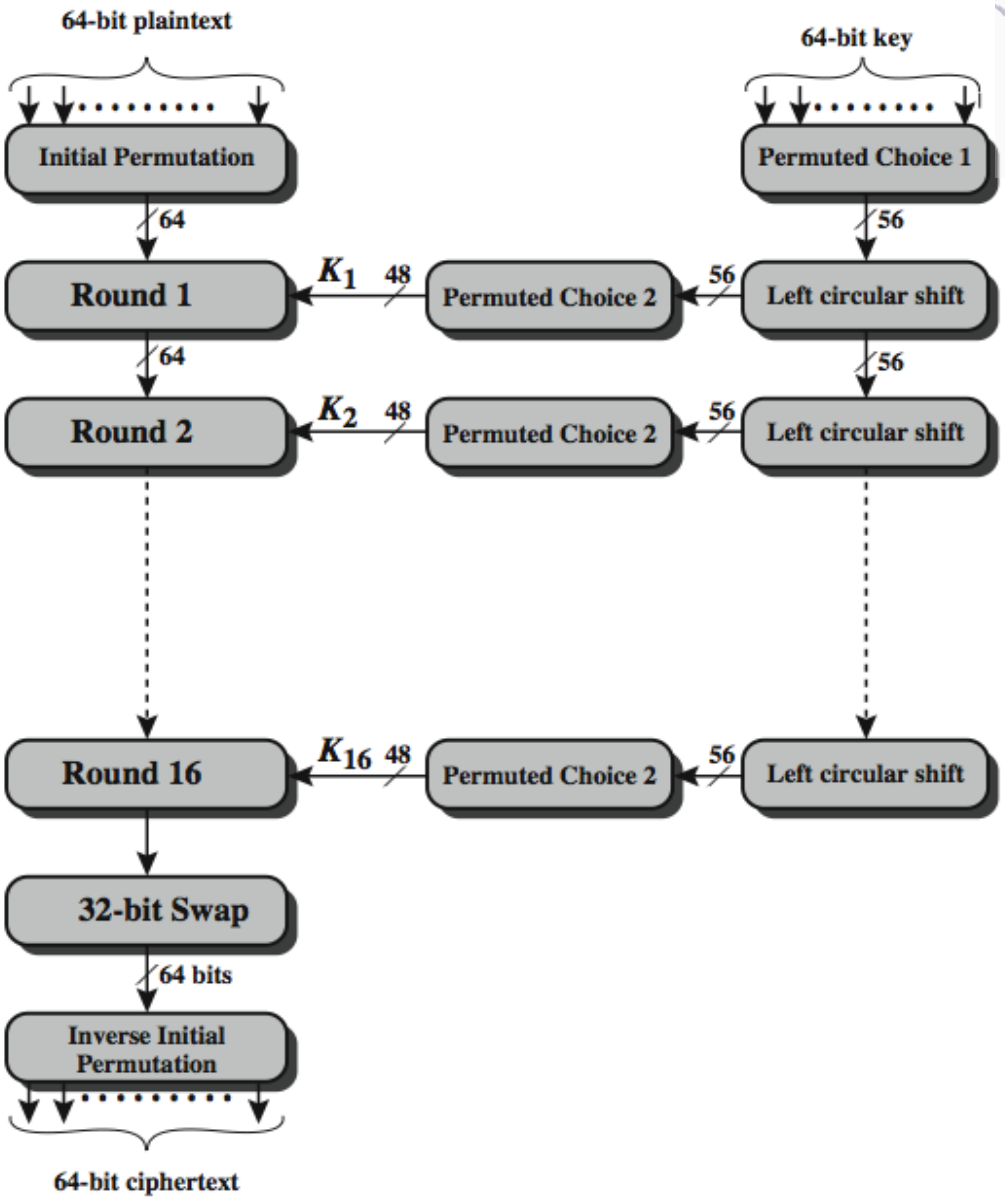


Figure 2.10 General Structure of DES Encryption Algorithm

2.10.1 DES Encryption

- The processing of the plaintext proceeds in three phases. First, the 64-bit plaintext passes through an initial permutation (IP) that rearranges the bits to produce the permuted input.
- Second phase consisting of sixteen rounds of the same function, which involves both permutation and substitution functions. The output of the last (sixteenth) round consists of 64 bits that are a function of the input plaintext and the key. The left and right halves of the output are swapped to produce the preoutput.

- Finally, the preoutput is passed through a permutation [IP-1] that is the inverse of the initial permutation function, to produce the 64-bit ciphertext.
- Figure 2.10 shows the way in which the 56-bit key is used. Initially, the key is passed through a permutation function. Then, for each of the sixteen rounds, a subkey (K_i) is produced by the combination of a left circular shift and a permutation. The permutation function is the same for each round, but a different subkey is produced because of the repeated shifts of the key bits.

Initial Permutation

- The initial permutation and its inverse are defined by tables, as shown in Tables 2.1 a and 2.1b, respectively.
- The input to a table consists of 64 bits numbered from 1 to 64. The 64 entries in the permutation table contain a permutation of the numbers from 1 to 64. Each entry in the permutation table indicates the position of a numbered input bit in the output, which also consists of 64 bits.
- Consider the following 64-bit input M:

M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8
M_9	M_{10}	M_{11}	M_{12}	M_{13}	M_{14}	M_{15}	M_{16}
M_{17}	M_{18}	M_{19}	M_{20}	M_{21}	M_{22}	M_{23}	M_{24}
M_{25}	M_{26}	M_{27}	M_{28}	M_{29}	M_{30}	M_{31}	M_{32}
M_{33}	M_{34}	M_{35}	M_{36}	M_{37}	M_{38}	M_{39}	M_{40}
M_{41}	M_{42}	M_{43}	M_{44}	M_{45}	M_{46}	M_{47}	M_{48}
M_{49}	M_{50}	M_{51}	M_{52}	M_{53}	M_{54}	M_{55}	M_{56}
M_{57}	M_{58}	M_{59}	M_{60}	M_{61}	M_{62}	M_{63}	M_{64}

Where M_i is a binary digit. Then the permutation $X = (\mathrm{IP}(M)$ is as follows:

M_{58}	M_{50}	M_{42}	M_{34}	M_{26}	M_{18}	M_{10}	M_2
M_{60}	M_{52}	M_{44}	M_{36}	M_{28}	M_{20}	M_{12}	M_4
M_{62}	M_{54}	M_{46}	M_{38}	M_{30}	M_{22}	M_{14}	M_6
M_{64}	M_{56}	M_{48}	M_{40}	M_{32}	M_{24}	M_{16}	M_8
M_{57}	M_{49}	M_{41}	M_{33}	M_{25}	M_{17}	M_9	M_1
M_{59}	M_{51}	M_{43}	M_{35}	M_{27}	M_{19}	M_{11}	M_3
M_{61}	M_{53}	M_{45}	M_{37}	M_{29}	M_{21}	M_{13}	M_5
M_{63}	M_{55}	M_{47}	M_{39}	M_{31}	M_{23}	M_{15}	M_7

If we then take the inverse permutation, $Y = \mathrm{IP}\text{-}1(X) = \mathrm{IP}\text{-}1(\mathrm{IP}(M))$ it can be seen that the original ordering of the bits is restored.

Table 2.1 Permutation Tables for DES

(a) Initial Permutation (IP)

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

(b) Inverse Initial Permutation (IP^{-1})

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

(c) Expansion Permutation (E)

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

(d) Permutation Function (P)

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

2.10.2 Details of Single Round

- Figure 2.11 shows the internal structure of a single round. The left and right halves of each 64-bit intermediate value are treated as separate 32-bit quantities, labeled L (left) and R (right).
- The overall processing at each round can be summarized in the following formulas:

$$Li = Ri\text{-}1$$

$$Ri = Li\text{-}1 \oplus F(Ri\text{-}1, Ki)$$

- The round key *Ki* is 48 bits. The input R is 32 bits. This input R is first expanded to 48 bits by using a table that defines a permutation plus an expansion that involves duplication of 16 of the R bits (Table 2.1c).The resulting 48 bits are XORed with K_i This 48-bit result passes through a substitution function that produces a 32-bit output, which is permuted as defined by Table 2.1d.

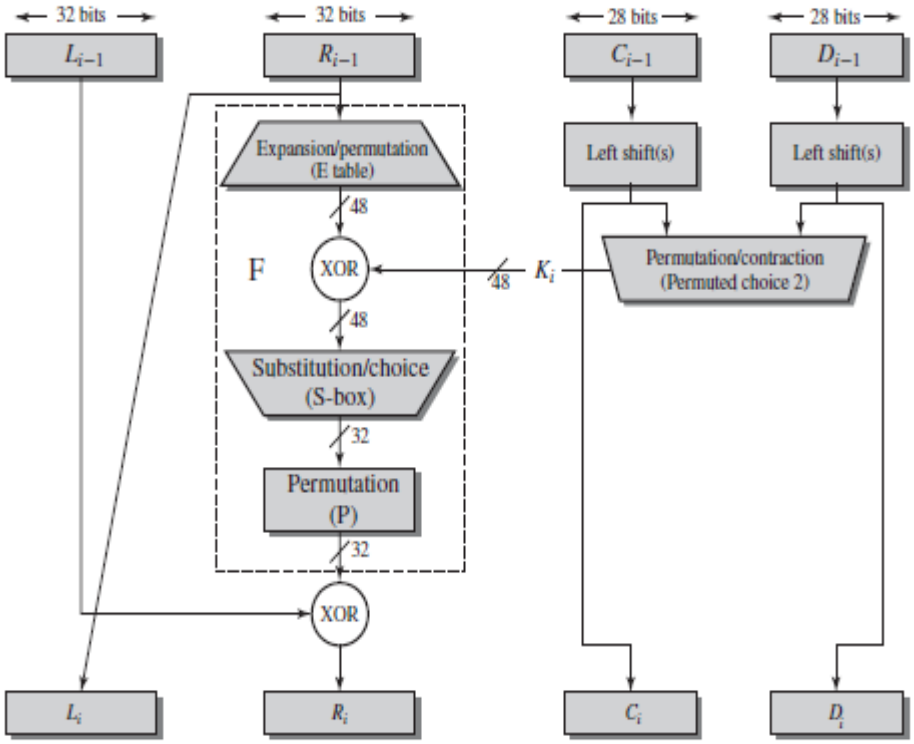


Figure 2. 11 Single Round of DES Algorithm

2.10.3 Key Generation

- A 64-bit key is used as input to the algorithm. The bits of the key are numbered from 1 through 64; every eighth bit is ignored, as indicated by the lack of shading in Table 2.2a.
- The key is first subjected to a permutation governed by a table labeled Permuted Choice One (Table 2.2b).
- The resulting 56-bit key is then treated as two 28-bit quantities, labelled C_0 and D_0.
- At each round, and are separately subjected to a circular left shift or (rotation) of 1 or 2 bits, as governed by Table 2.2d.
- These shifted values serve as input to the next round.
- They also serve as input to the part labeled Permuted Choice Two (Table 2.2c), which produces a 48-bit output that serves as input to the function $F(R_{i-1}, K_i)$

Table 2.2 DES Key Calculation

(a) Input 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	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

(b) Permuted Choice One (PC-1)

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

(c) Permuted Choice Two (PC-2)

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

(d) Schedule of Left Shifts

Round Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits Rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

2.10.4 S Boxes

- The substitution consists of a set of eight S-boxes (Figure 2.12), each of which accepts 6 bits as input and produces 4 bits as output.
- The 32-bit output from the eight S-boxes is then permuted, so that on the next round, the output from each S-box immediately affects as many others as possible.

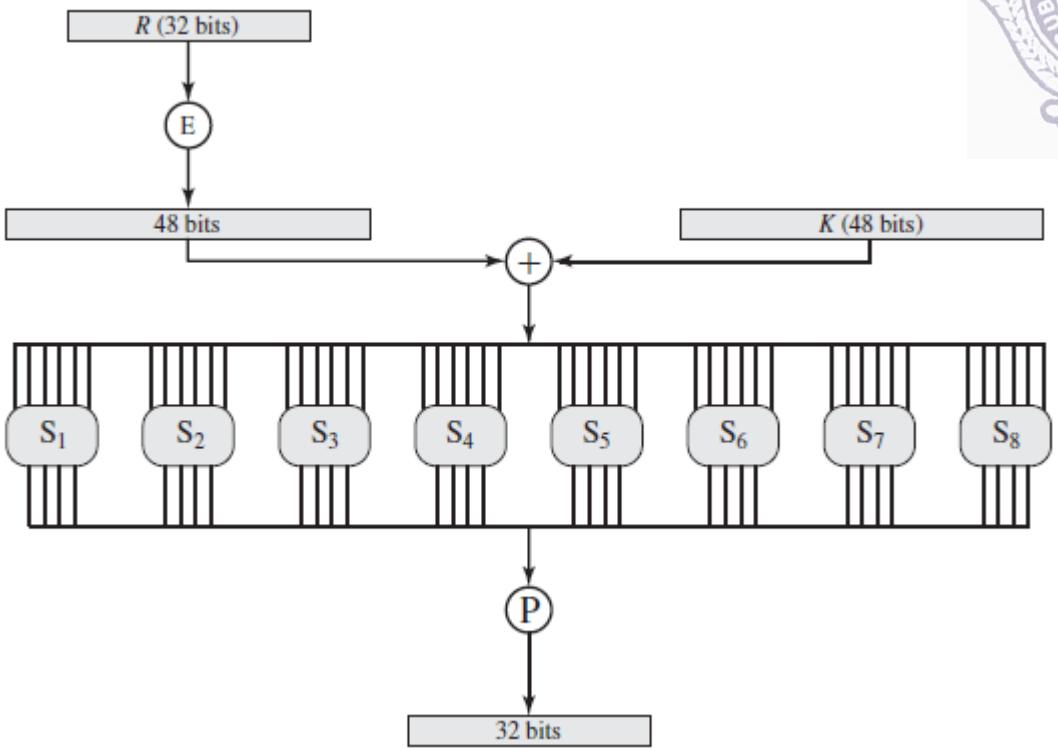


Figure 2.12 Calculation of F (R. K)

2.10.5 Avalanche Effect

- A desirable property of any encryption algorithm is that a small change in either the plaintext or the key should produce a significant change in the ciphertext.
- In particular, a change in one bit of the plaintext or one bit of the key should produce a change in many bits of the ciphertext. Figure 2.13 shows the avalanche effect.

(a) Change in Plaintext		(b) Change in Key	
Round	Number of bits that differ	Round	Number of bits that differ
0	1	0	0
1	6	1	2
2	21	2	14
3	35	3	28
4	39	4	32
5	34	5	30
6	32	6	32
7	31	7	35
8	29	8	34
9	42	9	40
10	44	10	38
11	32	11	31
12	30	12	33
13	30	13	28
14	26	14	26
15	29	15	34
16	34	16	35

Figure 2.13 Avalanche Effect in DES

2.11 The Strength of DES

- The Use of 56-Bit Keys With a key length of 56 bits, there are 2^{56} possible keys, which is approximately 7.2×10^{16} keys.
- Thus, on the face of it, a brute-force attack appears impractical. Assuming that, on average, half the key space has to be searched, a single machine performing one DES encryption per microsecond would take more than a thousand years to break the cipher.
- DES finally and definitively proved insecure in July 1998, when the Electronic Frontier Foundation (EFF) announced that it had broken a DES encryption using a special-purpose "DES cracker" machine that was built for less than $250,000. The attack took less than three days.
- The EFF has published a detailed description of the machine, enabling others to build their own cracker and hardware prices will continue to drop as speeds increase, making DES virtually worthless.
- There are a number of alternatives to DES, the most important of which are AES and triple DES.

2.12 Differential and Linear Cryptanalysis

- The DES algorithm is vulnerability against brute-force attack because of its relatively short (56 bits) key length.
- The increasing popularity of block ciphers with longer key lengths, including triple DES, brute-force attacks have become increasingly impractical. Thus, there has been increased emphasis on cryptanalytic attacks on DES and other symmetric block ciphers.

2.12.1 Differential Cryptanalysis

- Differential cryptanalysis is a general form of cryptanalysis applicable primarily to block ciphers, but also to stream ciphers and cryptographic hash functions. In the broadest sense, it is the study of how differences in information input can affect the resultant difference at the output.
- Differential cryptanalysis is the first published attack that is capable of breaking DES in less than 2^{55}encryptions. The scheme can successfully cryptanalyze DES with an effort on the order of 2^{47}encryptions, requiring chosen plaintexts. Although 2^{47} is

certainly significantly less than 2^{55}, the need for the adversary to find 2^{47} chosen plaintexts makes this attack of only theoretical interest.

2.12.2 Linear Cryptanalysis

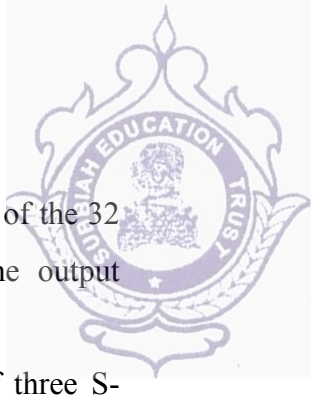
- This attack is based on finding linear approximations to describe the transformations performed in DES.
- This method can find a DES key given 2^{43} known plaintexts, as compared 2^{47} to chosen plaintexts for differential cryptanalysis. Although this is a minor improvement, because it may be easier to acquire known plaintext rather than chosen plaintext, it still leaves linear cryptanalysis infeasible as an attack on DES.

2.13 Block Cipher Design Principles

- There are three critical aspects of block cipher design:
  - ***The number of rounds***
  - ***Design of the function F***
  - ***Key scheduling***

DES Design Criteria

- The criteria used in the design of DES, focused on the design of the S-boxes and on the P function that takes the output of the S-boxes. The criteria for the S-boxes are as follows.
  - No output bit of any S-box should be too close a linear function of the input bits. Specifically, if we select any output bit and any subset of the six input bits, the fraction of inputs for which this output bit equals the XOR of these input bits should not be close to 0 or 1, but rather should be near 1/2.
  - Each row of an S-box (determined by a fixed value of the leftmost and rightmost input bits) should include all 16 possible output bit combinations.
  - If two inputs to an S-box differ in exactly one bit, the outputs must differ in at least two bits.
  - If two inputs to an S-box differ in the two middle bits exactly, the outputs must differ in at least two bits.
  - If two inputs to an S-box differ in their first two bits and are identical in their last two bits, the two outputs must not be the same.



- For any nonzero 6-bit difference between inputs, no more than eight of the 32 pairs of inputs exhibiting that difference may result in the same output difference.
- This is a criterion similar to the previous one, but for the case of three S-boxes.

➢ The S-boxes are the only nonlinear part of DES. If the S-boxes were linear (i.e., each output bit is a linear combination of the input bits), the entire algorithm would be linear and easily broken.

2.13.1 Number of Rounds

➢ The greater the number of rounds, the more difficult it is to perform cryptanalysis, even for a relatively weak F.

➢ In general, the criterion should be that the number of rounds is chosen so that known cryptanalytic efforts require greater effort than a simple brute-force key search attack.

➢ This criterion was certainly used in the design of DES. It observes that for 16-round DES, a differential cryptanalysis attack is slightly less efficient than brute force.

➢ The differential cryptanalysis attack requires $2^{55.1}$ operations, whereas brute force requires 2^{55}. If DES had 15 or fewer rounds, differential cryptanalysis would require less effort than a brute-force key search.

2.13.2 Design of Function F

➢ The heart of a Feistel block cipher is the function F. The function DES relies on the use of S-boxes.

Design Criteria for F

➢ The function F provides the element of confusion in a Feistel cipher. Thus, it must be difficult to "unscramble" the substitution performed by F. One obvious criterion is that F be nonlinear. If so, it will be very difficult any type of cryptanalysis. Several other criteria should be considered in designing F.

➢ The algorithm to have good avalanche properties. That means, a change in one bit of the input should produce a change in many bits of the output. A more stringent version of this is the **Strict Avalanche Criterion (SAC)** which states that any output bit of an S-box should change with probability 1/2 when any single input bit i is inverted for all i, j.

- Another criterion is the **Bit Independence Criterion (BIC)**, which states that output bits j and k should change independently when any single input bit i is inverted for all i, j and k.

S -Box Design

- One of the most intense areas of research in the field of symmetric block ciphers is that of S-box design.
- One obvious characteristic of the S-box is its size. An n x m S-box has n input bits and m output bits. DES has 6 x 4 S-boxes.
- The encryption algorithm Blowfish, has 8 x 32 S-boxes. Larger S-boxes, by and large, are more resistant to differential and linear cryptanalysis. The S-box design suggests the following approaches:
    - **Random:** Use some pseudorandom number generation or some table of random digits to generate the entries in the S-boxes. This may lead to boxes with undesirable characteristics for small sizes (e.g.,6 x 4) but should be acceptable for large S-boxes (e.g.,8 x 32).
    - **Random with testing:** Choose S-box entries randomly, then test the results against various criteria.
    - **Human-made:** This is a more or less manual approach with only simple mathematics to support it. It is apparently the technique used in the DES design. This approach is difficult to carry through for large S-boxes.
    - **Math-made:** Generate S-boxes according to mathematical principles. By using mathematical construction, S-boxes can be constructed that offer proven security against linear and differential cryptanalysis, together with good diffusion.

2.13.3 Key Scheduling

- A final area of block cipher design is the key schedule algorithm. With any Feistel block cipher, the key is used to generate one subkey for each round. In general, select subkeys to maximize the difficulty of deducing individual subkeys and the difficulty of working back to the main key.

2.14 Block Cipher Mode of Operation

- Encryption algorithms are divided into two categories based on input type, as block cipher and stream cipher.
- Block cipher is an encryption algorithm which takes fixed size of input say *b* bits and produces a ciphertext of *b* bits again.
- If input is larger than *b* bits it can be divided further. For different applications and uses, there are several modes of operations for a block cipher.
- The five standard Modes of Operation:
  - ***Electronic Code Book (ECB)***
  - ***Cipher Block Chaining (CBC)***
  - ***Cipher Feedback (CFB)***
  - ***Output Feedback (OFB)***
  - ***Counter (CTR)***

Electronic Code Book (ECB)

- Electronic code book is the easiest block cipher mode of functioning. It is easier because of direct encryption of each block of input plaintext and output is in form of blocks of encrypted ciphertext (Figure 2.14).
- Generally, if a message is larger than *b* bits in size, it can be broken down into bunch of blocks and the procedure is repeated. In this approach, the plaintext is handled one block at a time and each block of plaintext is encrypted using the same key.
- The term codebook is used because, for a given key, there is a unique ciphertext for every b-bit block of plaintext.

Advantages

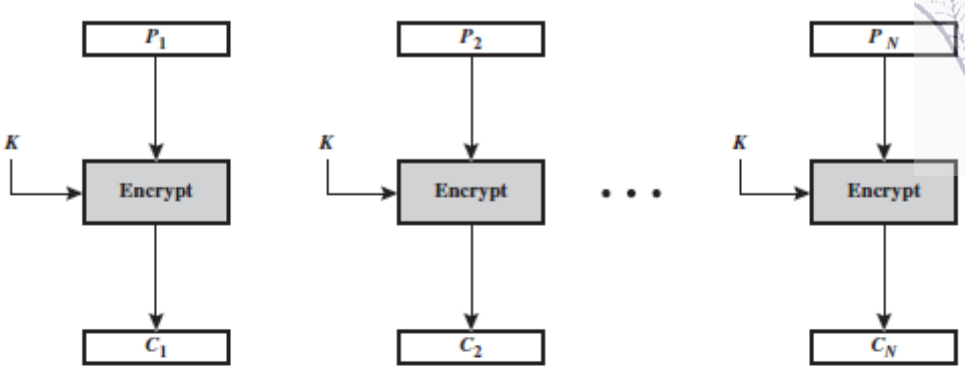
- Parallel encryption of blocks of bits is possible, thus it is a faster way of encryption.
- Simple way of block cipher.

Disadvantages

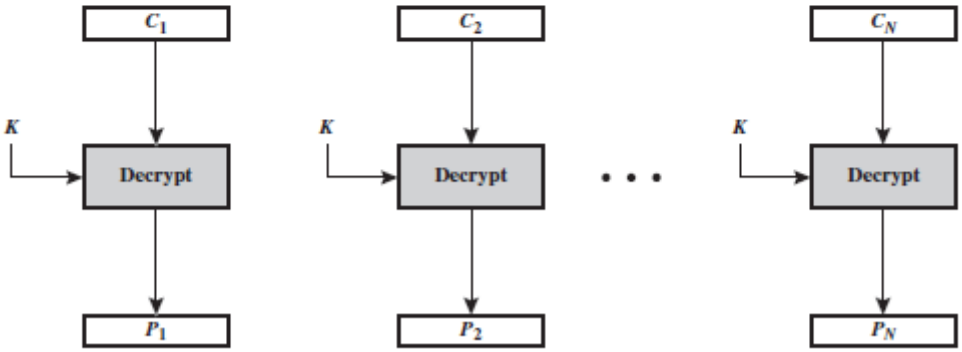
- Prone to cryptanalysis since there is a direct relationship between plaintext and ciphertext.

$$C_j = E(K, P_j) \qquad j = 1\ldots, N$$

$$P_j = D(K, C_j) \qquad j = 1\ldots, N$$



(a) Encryption



(b) Decryption

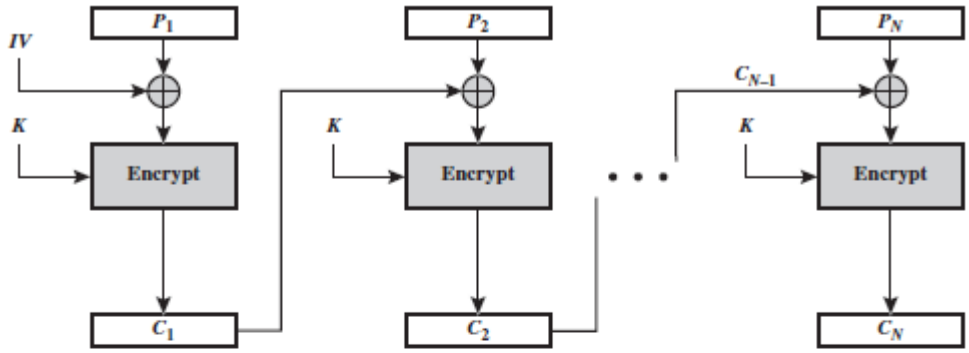
Figure 2.14 Electronic Code Book

Cipher Block Chaining (CBC)

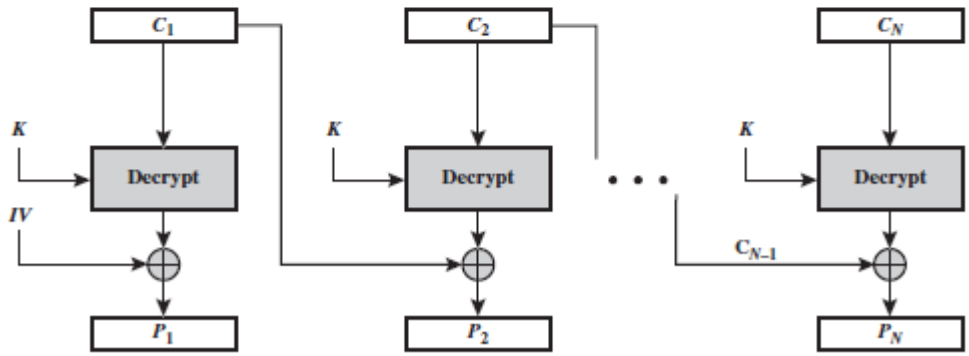
- This approach overcome the security deficiencies of ECB. In this scheme (Figure 2.15), the input to the encryption algorithm is the XOR of the current plaintext block and the preceding ciphertext block; the same key is used for each block.
- In effect, we have chained together the processing of the sequence of plaintext blocks.
- The CBC mode requires that the last block be padded to a full bits if it is a partial block.
- To produce the first block of ciphertext, an Initialization Vector (IV) is XORed with the first block of plaintext.
- On decryption, the IV is XORed with the output of the decryption algorithm to recover the first block of plaintext.
- The IV is a data block that is that same size as the cipher block. The IV must be known to both the sender and receiver but be unpredictable by a third party. In particular, for any given plaintext, it must not be possible to predict the IV that will be

associated to the plaintext in advance of the generation of the IV. For maximum security, the IV should be protected against unauthorized changes.

CBC	$C_1 = \mathrm{E}(K, [P_1 \oplus \mathrm{IV}])$ $C_j = \mathrm{E}(K, [P_j \oplus C_{j-1}])\ \ j = 2, \ldots, N$	$P_1 = \mathrm{D}(K, C_1) \oplus \mathrm{IV}$ $P_j = \mathrm{D}(K, C_j) \oplus C_{j-1}\ \ j = 2, \ldots, N$



(a) Encryption



(b) Decryption

Figure 2.15 Cipher Block Chaining

Advantages

- CBC works well for input greater than *b* bits.
- CBC is a good authentication mechanism.
- Better resistive nature towards cryptanalysis than ECB.

Disadvantages

- Parallel encryption is not possible since every encryption requires previous cipher.

Cipher Feedback (CFB)

- In this approach (figure 2.16), the input to the encryption function is a b-bit shift register that is initially set to some initialization vector (IV).
- The leftmost (most significant) s bits of the output of the encryption function are XORed with the first segment of plaintext P_1 to produce the first unit of ciphertext C_1, which is then transmitted.
- In addition, the contents of the shift register are shifted left by s bits, and C_1 is placed in the rightmost (least significant) s bits of the shift register. This process continues until all plaintext units have been encrypted.
- For decryption, the same scheme is used, except that the received ciphertext unit is XORed with the output of the encryption function to produce the plaintext unit. Let MSBs (X) be defined as the most significant s bits of X. Then,

$$C1 = P1 \oplus \text{MSB}s\ [\text{E}\ (K, \text{IV})]$$

$$P1 = C1 \oplus \text{MSB}s\ [\text{E}\ (K, \text{IV})]$$

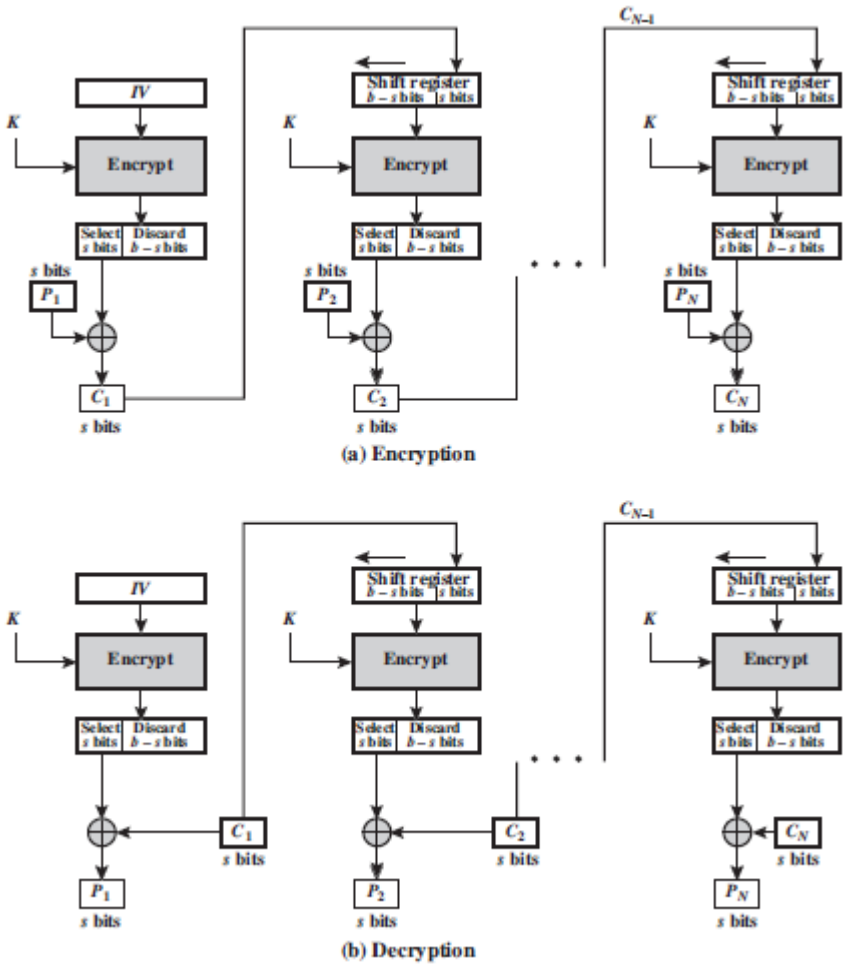


Figure 2.16 Cipher Feedback

Advantages

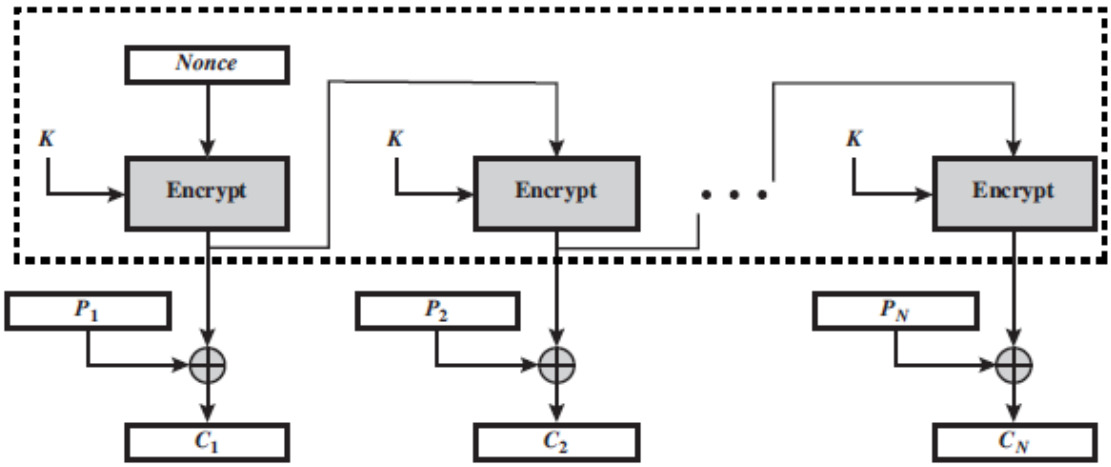
- Since, there is some data loss due to use of shift register, thus it is difficult for applying cryptanalysis.

Output Feedback (OFB)

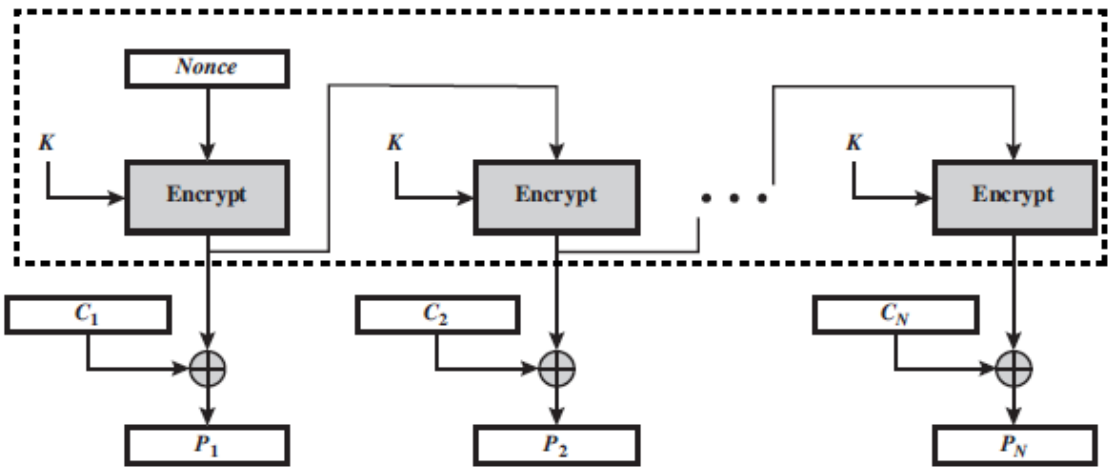
- The output feedback (OFB) mode is similar in structure to that of CFB (Figure 2.17), it is the output of the encryption function that is fed back to the shift register in OFB, whereas in CFB, the ciphertext unit is fed back to the shift register.
- The other difference is that the OFB mode operates on full blocks of plaintext and ciphertext, not on an s bit subset. Encryption and Decryption can be expressed as

$$Cj = Pj \oplus E\ (K, [Cj\text{-}i\_\ Pj\text{-}1])$$

$$Pj = Cj \oplus E\ (K, [Cj\text{-}1\ \_\ Pj\text{-}1])$$



(a) Encryption



(b) Decryption

Figure 2.17 Output Feedback

Advantages

- Bit errors in transmission do not propagate

Disadvantages

- It is more vulnerable to a message stream modification attack than is CFB.

Counter (CTR)

- A counter equal to the plaintext block size is used. The only requirement stated is that the counter value must be different for each plaintext block that is encrypted (Figure 2.18).
- Typically, the counter is initialized to some value and then incremented by 1 for each subsequent block (modulo, where is the block size).
- For encryption, the counter is encrypted and then XORed with the plaintext block to produce the ciphertext block; there is no chaining.
- For decryption, the same sequence of counter values is used, with each encrypted counter XORed with a ciphertext block to recover the corresponding plaintext block. Thus, the initial counter value must be made available for decryption.
- Given a sequence of counters T_1, T_2, …, T_N, we can define CTR mode as follows.

CTR	$C_j = P_j \oplus \mathrm{E}(K, T_j) \quad j = 1, \ldots, N-1$ $C_N^* = P_N^* \oplus \mathrm{MSB}_u[\mathrm{E}(K, T_N)]$	$P_j = C_j \oplus \mathrm{E}(K, T_j) \quad j = 1, \ldots, N-1$ $P_N^* = C_N^* \oplus \mathrm{MSB}_u[\mathrm{E}(K, T_N)]$

- For the last plaintext block, which may be a partial block of bits, the most significant bits of the last output block are used for the XOR operation; the remaining bits are discarded.

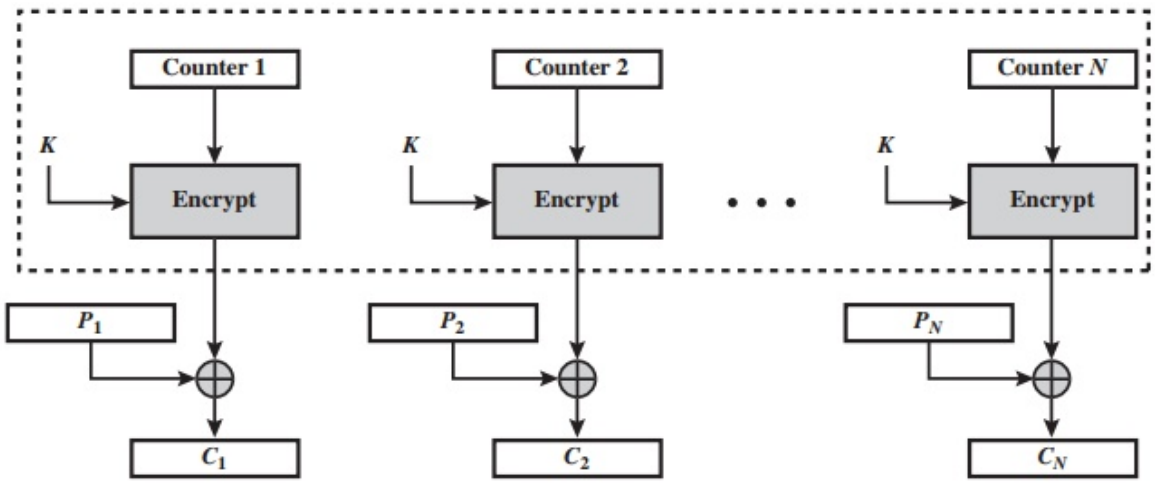


Figure 2.18 Counter

Advantages

- Hardware efficiency
- Software efficiency
- Preprocessing
- Random access
- Provable security
- Simplicity

2.15 Evaluation criteria for AES

- AES has been subjected to more scrutiny than any other encryption algorithm over a longer period of time, and no effective cryptanalytic attack based on the algorithm rather than brute force has been found.
- The principal 3DES is that the algorithm is relatively sluggish in software. The original DES was designed in 1970, the hardware implementation does not produce efficient software code. 3DES which has three times as many rounds as DES, is correspondingly slower. The secondary drawback is that both DES and 3DES use a 64 bits block size.
- Because of these drawbacks 3DES is not a reasonable for long time use. As a replacement, proposed a new symmetric encryption algorithm which is called Advanced Encryption Standard, which should have security strength equal to or better than 3DES and significantly improved efficiency. The three categories of criteria were:
    - Security: This refers to the effort required to cryptanalyze an algorithm. The emphasis in the evaluation was on the practicality of the attack. Because the minimum key size for AES is 128 bits, brute-force attacks with current and projected technology were considered impractical. Therefore, the emphasis, with respect to this point, is cryptanalysis other than a brute-force attack.
    - Cost: NIST intends AES to be practical in a wide range of applications. Accordingly, AES must have high computational efficiency, so as to be usable in high-speed applications, such as broadband links.
    - Algorithm and implementation characteristics: This category include a variety of considerations, including flexibility; suitability for a variety of hardware

and software implementations; and simplicity, which will make an analysis of security more straightforward.

2.16 Advanced Encryption Standard (AES)

- The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES). It is found at least six time faster than triple DES.
- The AES algorithm developed by two Belgian cryptographers, Vincent Rijmen and Joan Daemen.
- The cipher takes a plaintext block size of 128 bits, or 16 bytes. The key length can be 16, 24, or 32 bytes (128, 192, or 256 bits). The algorithm is referred to as AES-128, AES-192, or AES-256, depending on the key length.

Overall Structure of AES

- The AES is an iterative rather than Feistel cipher. It is based on 'substitution–permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).
- Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix.
- Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.
- The overall structure of AES (figure 2.19) focus particularly on the four steps used in each round of AES:
    - Byte Substitution
    - Shift Rows
    - Mix Columns
    - Add Round Key

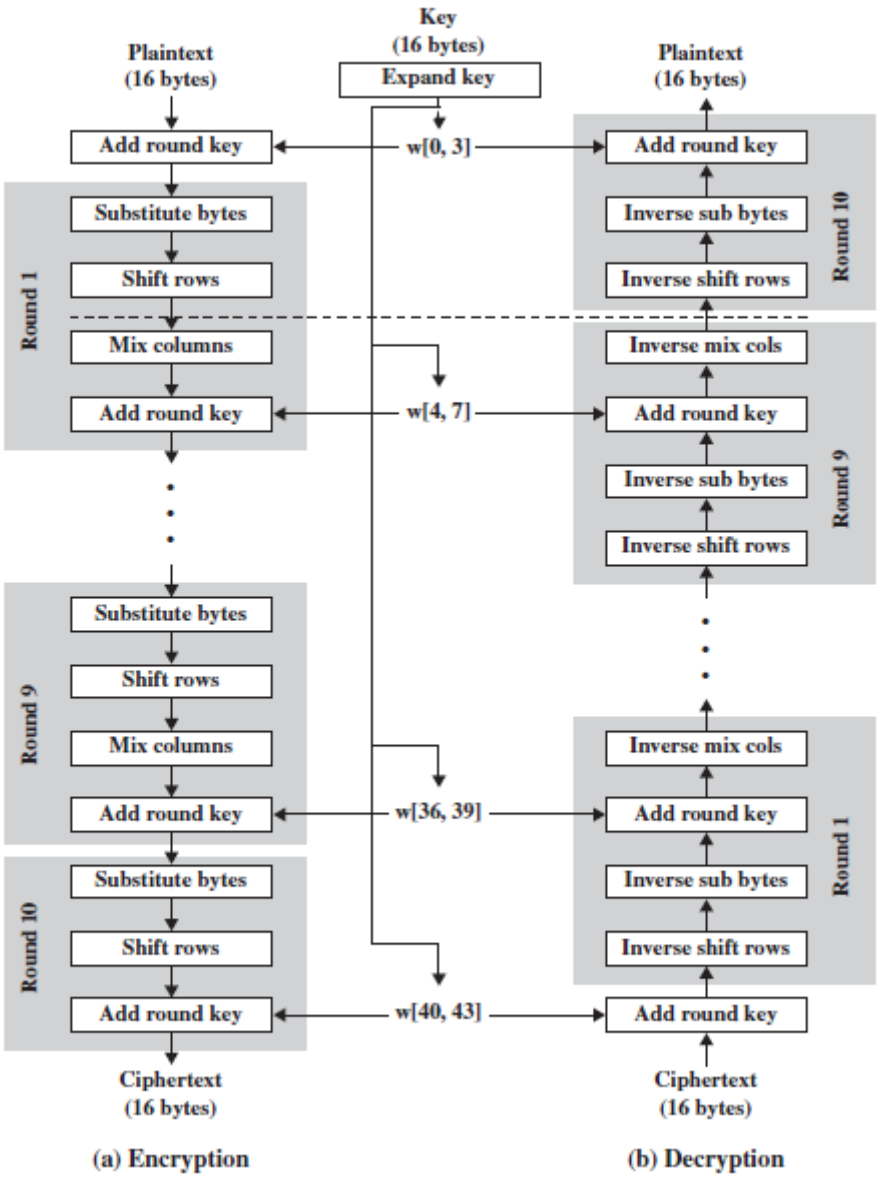


Figure 2.19 Overal Structure of AES

2.16.1 Encryption Process

Byte Substitution (SubBytes)

- Uses an S-box to perform a byte-by-byte substitution of the block. The forward substitute byte transformation, called SubBytes, is a simple lookup the S-box table and replace the value (Figure 2.20). AES defines a 16 X 16 matrix of byte values, called an S-box, that contains a permutation of all possible 256 8-bit values.
- Each individual byte of State is mapped into a new byte in the following way: The leftmost 4 bits of the byte are used as a row value and the rightmost 4 bits are used as

a column value. These row and column values serve as indexes into the S-box to select a unique 8-bit output value.

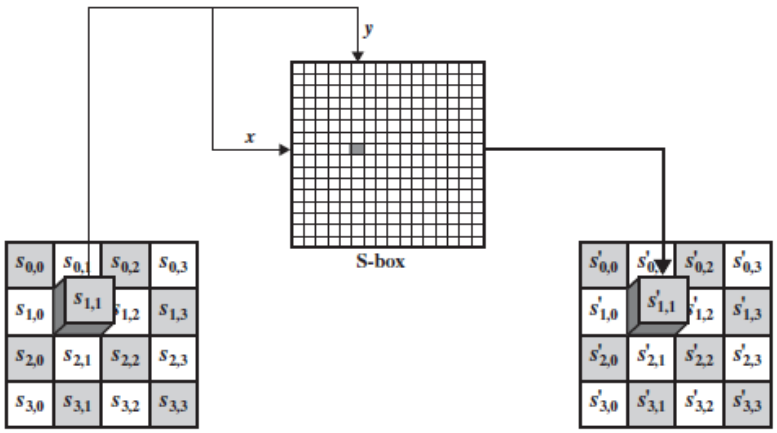


Figure 2.20 Substitute Byte Transformation

- For example (Table 2.3), the hexadecimal value {86} references row 8, column 6 of the S-box, which contains the value {44}. Accordingly, the value {44} is mapped into the value {86} from Inverse S-box at decryption stage.

x \ y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

(a) S-box

x \ y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

(b) Inverse S-box

Table 2.3

Example of SubBytes transformation

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

→

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

Shift Rows Transformation

- In Shift Rows transformation (figure 2.21), the first row of State is not altered. For the second row, a 1-byte circular left shift is performed. For the third row, a 2-byte circular left shift is performed. For the fourth row, a 3-byte circular left shift is performed.

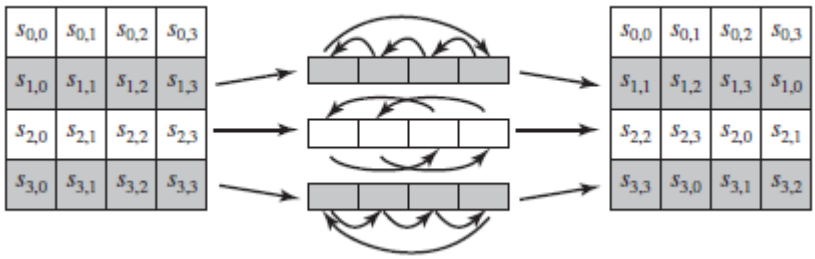


Figure 2.21 Shift Row Transformation

Example of Shift Rows Tranformation

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

→

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

MixColumns Transformation

- It operates on each column individually. Each byte of a column is mapped into a new value that is a function of all four bytes in that column. The transformation can be defined by the following matrix multiplication on State (Figure 2.22)

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

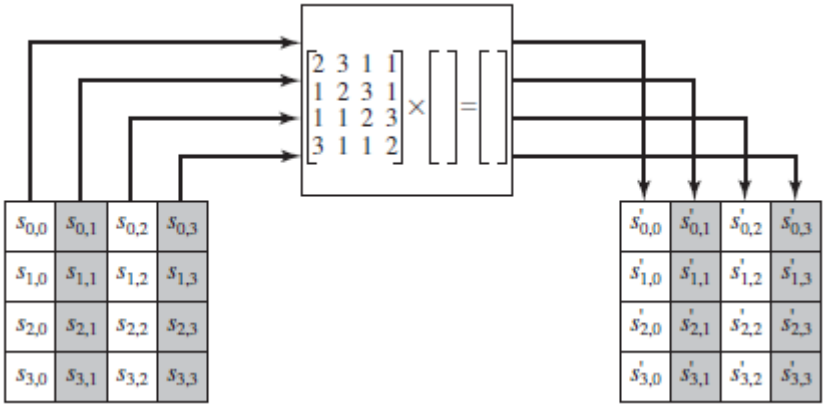


Figure 2.22 MixColumns Transformation

Example of MixColumns Transsormation

87	F2	4D	97	→	47	40	A3	4C
6E	4C	90	EC		37	D4	70	9F
46	E7	4A	C3		94	E4	3A	42
A6	8C	D8	95		ED	A5	A6	BC

AddRoundKey Transormation

- It is a simple bitwise XOR of the current block with a portion of the expanded key. The 128 bits of **State** are bitwise XORed with the 128 bits of the round key. As shown in Figure 2.23, the operation is viewed as a columnwise operation between the 4 bytes of a **State** column and one word of the round key; it can also be viewed as a byte-level operation.

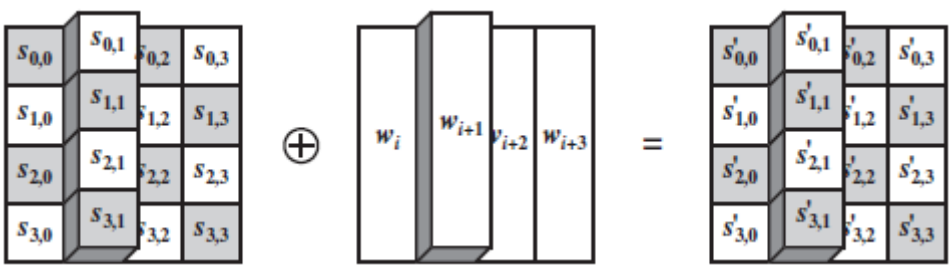


Figure 2.23 AddRoundKey Transformation

Example of AddRoundKey

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$\oplus$

AC	19	28	57
77	FA	D1	5C
66	DC	29	00
F3	21	41	6A

=

EB	59	8B	1B
40	2E	A1	C3
F2	38	13	42
1E	84	E7	D6

2.16.2 AES Key Expansion Algorithm

- This algorithm takes as input a four-word (16-byte) key and produces a linear array of 44 words (176 bytes).
- This is sufficient to provide a four-word round key for the initial AddRoundKey stage and each of the 10 rounds of the cipher.
- The key is copied into the first four words of the expanded key. The remainder of the expanded key is filled in four words at a time. Each added word w[i] depends on the immediately preceding word w[i-1], and the word four positions back, w[i-4].
- In three out of four cases, a simple XOR is used. For a word whose position in the **w** array is a multiple of 4, a more complex function is used. Figure 2.24 illustrates the generation of the expanded key, using the symbol g to represent that complex function.

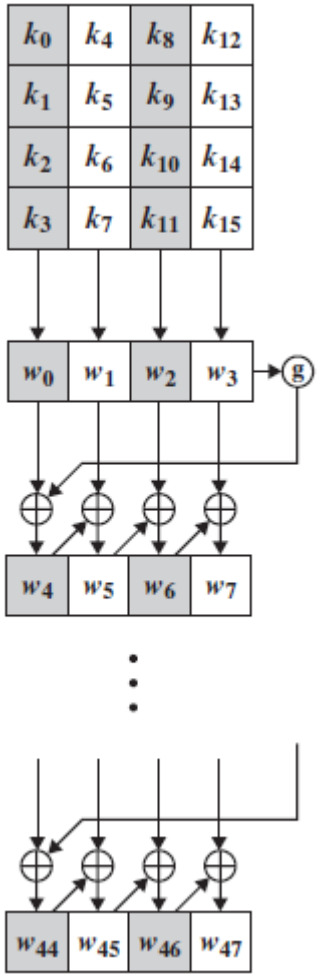


Figure 2.24 AES Key Expansion

2.17 RC4

- RC4 is an encryption algorithm created in 1987 by Ronald Rivest of RSA Security. It is a stream cipher (figure 2.25), which means that each digit or character is encrypted one at a time. A cipher is a message that has been encoded.
- A key input is pseudorandom bit generator that produces a stream 8-bit number that is unpredictable without knowledge of input key.
- The output of the generator is called key-stream, is combined one byte at a time with the plaintext stream cipher using X-OR operation.

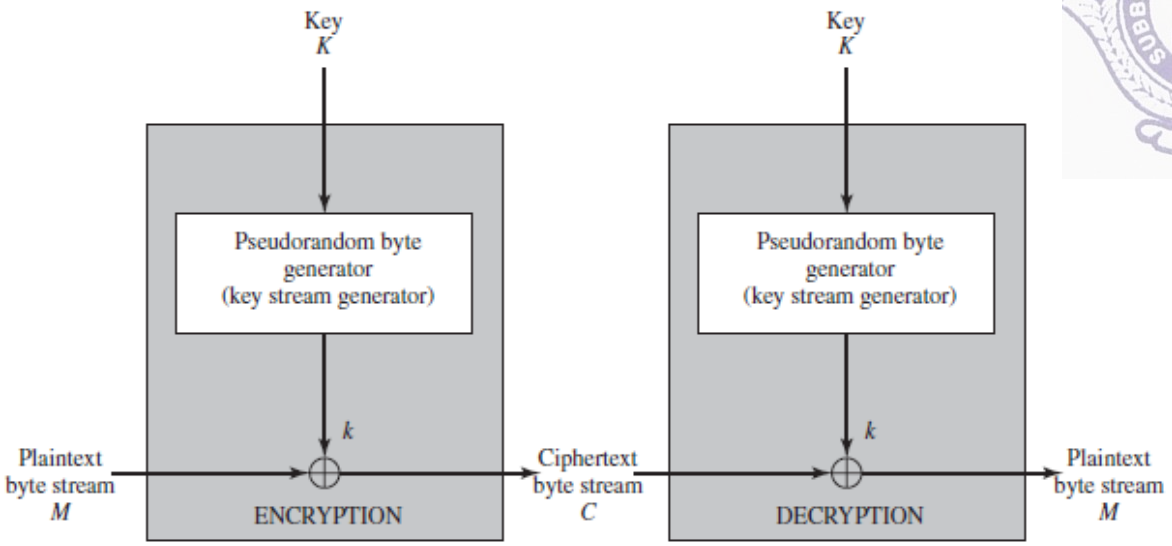


Figure 2.25 Stream Cipher Diagram

Example

RC4 Encryption		RC4 Decryption	
10011000	Plaintext	11001000	Ciphertext
⊕ 01010000	Key Stream	⊕ 01010000	Key Stream
11001000	Ciphertext	10011000	Plaintext

2.17.1 Key Generation Algorithm

- A variable-length key from 1 to 256 byte is used to initialize a 256-byte state vector S, with elements S[0] to S[255]. For encryption and decryption, a byte k is generated from S by selecting one of the 255 entries in a systematic fashion, then the entries in S are permuted again(Figure 2.26).

Initialization of S

- The entries of S are set equal to the values from 0 to 255 in ascending order, a temporary vector T, is created. If the length of the key k is 256 bytes, then k is assigned to T. Otherwise, for a key with length(k-len) bytes, the first k-len elements of T as copied from K and then K is repeated as many times as necessary to fill T.

```
// Initialization
for
i = 0 to 255 do S[i] = i;
T[i] = K[i mod k - len];
```

- Next, use T to produce the initial permutation of S. Starting with S[0] to S[255], and for each S[i] algorithm swap it with another byte in S according to a scheme dictated by T[i], but S will still contain values from 0 to 255:

```
// Initial Permutation of S
j = 0;
for
i = 0 to 255 do
 {
 j = (j + S[i] + T[i]) mod 256;
 Swap(S[i], S[j]);
 }
```

Pseudo random generation algorithm (Stream Generation)

- Once the vector S is initialized, the input key will not be used. In this step, for each S[i] algorithm swap it with another byte in S according to a scheme dictated by the current configuration of S. After reaching S[255] the process continues, starting from S[0] again

```
//Stream Generation
i, j = 0;
while (true)
    i = (i + 1)mod 256;
j = (j + S[i])mod 256;
Swap(S[i], S[j]);
t = (S[i] + S[j])mod 256;
k = S[t];
```

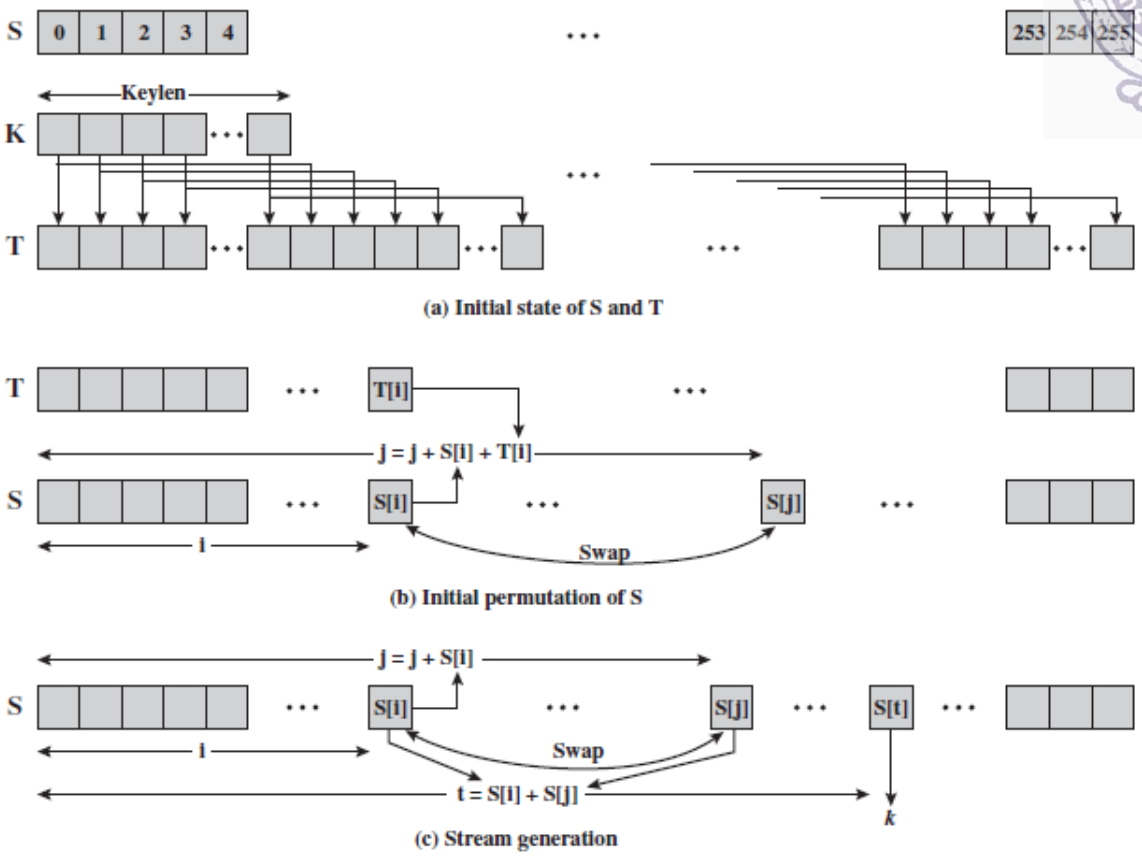


Figure 2.26 RC4 Algorithm

Encrypt using XOR

- To encrypt, XOR the value k with the next byte of plaintext.

Decrypt using XOR

- To decrypt, XOR the value k with the next byte of ciphertext.

Advantage

- It is faster and more suitable for streaming application

2.18 Key Distribution

Symmetric Key Distribution Using Symmetric Encryption

- In Symmetric key encryption, the two parties to an exchange must share the same key, and that key must be protected from access by others. Therefore, the term that refers to the means of delivering a key to two parties who wish to exchange data, without allowing others to see the key.
- For two parties A and B, key distribution can be achieved in a number of ways, as follows:

1. A can select a key and physically deliver it to B.
2. A third party can select the key and physically deliver it to A and B.
3. If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.
4. If A and B each has an encrypted connection to a third-party C, C can deliver a key on the encrypted links to A and B.

- Physical delivery (1 & 2) is simplest - but only applicable when there is personal contact between recipient and key issuer. This is fine for link encryption where devices & keys occur in pairs, but does not scale as number of parties who wish to communicate grows. 3 is mostly based on 1 or 2 occurring first.
- A third party, whom all parties trust, can be used as a trusted intermediary to mediate the establishment of secure communications between them (4). Must trust intermediary not to abuse the knowledge of all session keys. As number of parties grow, some variant of 4 is only practical solution to the huge growth in number of keys potentially needed.

Key Distribution Centre

- The use of a key distribution center is based on the use of a hierarchy of keys. At a minimum, two levels of keys are used.
- Communication between end systems is encrypted using a temporary key, often referred to as a Session key.
- Typically, the session key is used for the duration of a logical connection and then discarded
- Master key is shared by the key distribution center and an end system or user and used to encrypt the session key.

Key Distribution Scenario

- Let us assume that user A wishes to establish a logical connection with B and requires a one-time session key to protect the data transmitted over the connection. A has a master key, K_a, known only to itself and the KDC; similarly, B shares the master key K_b with the KDC(Figure 2.27). The following steps occur:

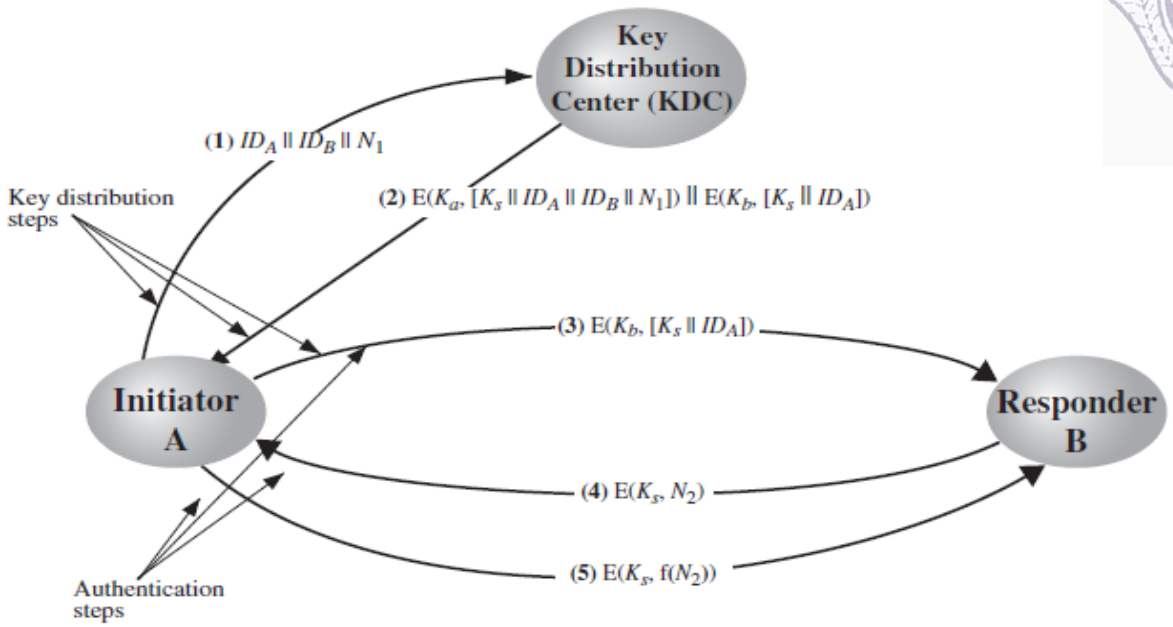


Figure 2.27 Key Distribution Scenario

1. An issue a request to the KDC for a session key to protect a logical connection to B. The message includes the identity of A and B and a unique identifier, N_1, for this transaction, which we refer to as a nonce. The nonce may be a timestamp, a counter, or a random number; the minimum requirement is that it differs with each request. Also, to prevent masquerade, it should be difficult for an opponent to guess the nonce. Thus, a random number is a good choice for a nonce.

2. The KDC responds with a message encrypted using Ka Thus, A is the only one who can successfully read the message, and A knows that it originated at the KDC. The message includes two items intended for A:

   - The one-time session key, Ks, to be used for the session
   - The original request message, including the nonce, to enable A to match this response with the appropriate request

   Thus, A can verify that its original request was not altered before reception by the KDC and, because of the nonce, that this is not a replay of some previous request. In addition, the message includes two items intended for B:

   - The one-time session key, Ks to be used for the session
   - An identifier of A (e.g., its network address), IDA

These last two items are encrypted with Kb (the master key that the KDC shares with B). They are to be sent to B to establish the connection and prove A's identity.

3. A store the session key for use in the upcoming session and forwards to B the information that originated at the KDC for B, namely, E (K_b, [K_s || ID_A]). Because this information is encrypted with K_b, it is protected from eavesdropping. B now knows the session key (K_s), knows that the other party is A (from ID_A), and knows that the information originated at the KDC (because it is encrypted using K_b). At this point, a session key has been securely delivered to A and B, and they may begin their protected exchange. However, two additional steps are desirable:
4. Using the newly minted session key for encryption, B sends a nonce, N_2, to A.
5. Also using K_s, A responds with f(N_2), where f is a function that performs some transformation on N_2 (e.g., adding one).

Session Key Lifetime

- The distribution of session keys delays the start of any exchange and places a burden on network capacity. A security manager must try to balance these competing considerations in determining the lifetime of a particular session key.
- For connection-oriented protocols, one obvious choice is to use the same session key for the length of time that the connection is open, using a new session key for each new session.
- If a logical connection has a very long lifetime, then it would be prudent to change the session key periodically, perhaps every time the PDU (protocol data unit) sequence number cycles.
- For a connectionless protocol, such as a transaction-oriented protocol, there is no explicit connection initiation or termination.
- Thus, it is not obvious how often one needs to change the session key. The most secure approach is to use a new session key for each exchange.
- A better strategy is to use a given session key for a certain fixed period only or for a certain number of transactions.

3

PUBLIC KEY CRYPTOGRAPHY

MATHEMATICS OF ASYMMETRIC KEY CRYPTOGRAPHY: Primes – Primality Testing – Factorization – Euler‘s totient function, Fermat‘s and Euler‘s Theorem - Chinese Remainder Theorem – Exponentiation and logarithm - ASYMMETRIC KEY CIPHERS: RSA cryptosystem – Key distribution – Key management – Diffie Hellman key exchange - ElGamal cryptosystem – Elliptic curve arithmetic-Elliptic curve cryptography.

3.1 Mathematics of Asymmetric Key Cryptography

3.1.1 Number Theory

- Number Theory plays an important role in encryption algorithm. It is a vast and fascinating field of mathematics, sometimes called "higher arithmetic," consisting of the study of the properties of whole numbers.
- Primes and Prime Factorization are especially important in number theory, as are a number of functions including the Totient function.
- Cryptography is the practice of hiding information, converting some secret information to not readable texts.
- Cryptography is the study of methods to send and receive the secret messages. In general, we have a sender who is trying to send a message to receiver. There is also an adversary, who wants to steal the message. We are successful if sender is able to communicate a message to the receiver without adversary learning what the message was.
- Applications of number theory in cryptography are very important in constructions of public key cryptosystems.
- The most popular public key cryptosystems are based on the problem of factorization of large integers and discrete logarithm problem in finite groups, in particular in the multiplicative group of finite fields and the group of points on elliptic curve over finite field.

3.2 Important concepts in Number Theory

3.2.1 Prime Numbers

- A positive integer p is said to be a prime if it has only two factors namely 1 and p itself.

  For Example: 2, 3, 5, 7, 11, 13, 17 are prime numbers. 4,6,8,9,10 are not.
- Prime numbers are central to number theory
- List of prime number less than 200 is:

  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

3.2.2 **Relatively Prime Numbers**

- Two integers are relatively prime (or coprime) if there is no integer greater than one that divides them both (that is, their greatest common divisor is one). For example, 12 and 13 are relatively prime, but 12 and 14 are not.

3.2.3 Divisors

- A positive integer a is said to divide an integer b if there exist an integer c such that b = a.c and written as a | b.

  For Example, 2 |10 as 10 = 2.5 but 3 do not divide 10 as there does not exist any integer c such that 10 = 3. C

3.2.4 Greatest Common Divisor

- Let a and b be two positive integers then an integer d is called greatest common divisor of a and b if d | a and d | b i.e. d is common divisor of a and b. And if any integer c is such that c | a and c | b then c | d, i.e. any other common divisor of a and b will divide d it is denoted by d = (a, b)
- Conversely can determine the greatest common divisor by comparing their prime factorizations and using least powers

  For Example, $300=2^1x3^1x5^2$ $18=2^1x3^2$ hence GCD (18, 300) = $2^1x3^1x5^0=6$

Using Euclid's algorithm

- A much more efficient method is the Euclidean algorithm, which uses the division algorithm in combination with the observation that the gcd of two numbers also divides their difference.

  gcd(a,0) =a

  gcd (a, b) =gcd (b, a mod b)

- For example,

Program 1: Write C++ program to find GCD of two numbers using Euclidean algorithm

```
#include <iostream>
using namespace std;
// Recursive function to return gcd of a and b
int gcd(int a, int b)
{
   if (b == 0)
      return a;
   return gcd(b, a % b);

}

// Driver program to test above function
int main ()
{
```

```
    int a = 98, b = 56;
    cout<<"GCD of "<<a<<" and "<<b<<" is "<<gcd(a, b);
    return 0;
}
```

Output:

GCD of 98 and 56 is 14

3.3 Primality Testing

- A **primality test** is an algorithm for determining whether an input **number** is prime. Among other fields of mathematics, it is used for cryptography. Unlike integer factorization, **primality tests** do not generally give prime factors, only stating whether the input **number** is prime or not.

The Fermat Test

- By Fermat's Theorem, if n is prime, then for any a we have

$a^{n-1} = 1 \pmod{n}$. This suggests the Fermat test for a prime: pick a random **a** $\in \{1,...,n-1\}$ **a** $\in \{1,...,n-1\}$ and see if $a^{n-1}=1 \pmod{n}$. If not, then **n** must be composite.

However, we may still get equality even when **n** is not prime.

For example, take **n** = 561 = 3 × 11 × 17. By the Chinese Remainder Theorem

$$Z_{561}=Z_3\times Z_{11}\times Z_{17}$$

thus, each $a \in Z^*_{561}$ corresponds to some

$$(x, y, z) \in Z^*_3\times Z^*_{11}\times Z^*_{17}$$

By Fermat's Theorem, $x^2 = 1$, $y^{10} = 1$ and $z^{16} = 1$. Since 2, 10, and 16 all divide 560, this means $(x, y, z)^{560} = (1, 1, 1)$ in other words, $a^{560} = 1$ for any $a \in Z^*_{561}$.

Thus, no matter what **"a"** we pick, 561 always passes the Fermat test despite being composite so long as aa is coprime to **n**. Such numbers are called Carmichael numbers, and it turns out there are infinitely many of them.

If a is not coprime to n then the Fermat test fails, but in this case, we may as well forgo tests and recover a factor of n simply by computing gcd (a, n).

3.4 **Factorization**

- Prime Factorization (or integer factorization) is a commonly used mathematical problem often used to secure public-key encryption systems. A common practice is to use very large semi-primes (that is, the result of the multiplication of two prime numbers) as the number securing the encryption.

3.4.1Fundamental Theorem of Arithmetic

- Any positive integer greater than one can be written uniquely in the following prime factorization form where P_1, P_2…., Pk are primes and e_1, e_2…e_k are positive integers.

$$n = p_1^{e1} \times p_2^{e2} \times \cdots \times p_k^{ek}$$

Applications of factorization

- **Greatest Common Divisor**
  - The GCD of two numbers, gcd (a, b). This value can also be found if we know the factorization of a and b.

$$a = p_1^{a_1} \times p_2^{a_2} \times \cdots \times p_k^{a_k} \qquad b = p_1^{b_1} \times p_2^{b_2} \times \cdots \times p_k^{b_k}$$

$$\gcd(a, b) = p_1^{\min(a_1, b_1)} \times p_2^{\min(a_2, b_2)} \times \cdots \times p_k^{\min(a_k, b_k)}$$

- **Least Common Multiplier**
  - The least common multiplication, lcm (a, b), is the smallest integer that is multiple of both a and b. using factorization, can also find lcm (a, b).

$$a = p_1^{a1} \times p_2^{a2} \times \cdots \times p_k^{ak} \qquad b = p_1^{b1} \times p_2^{b2} \times \cdots \times p_k^{bk}$$

$$\text{lcm}(a, b) = p_1^{\max(a_1, b_1)} \times p_2^{\max(a_2, b_2)} \times \cdots \times p_k^{\max(a_k, b_k)}$$

  - It can be proved that gcd (a, b) and lcm (a, b) are related to each other as shown below.

$$\text{lcm}(a, b) \times \gcd(a, b) = a \times b$$

3.4.2 Factorization Methods

- Although there are several algorithms that can factor a number, none are capable of factoring a very large number in a reasonable amount of time.

Trial Division Method

- It is the simplest and least efficient algorithm.
- The essential idea behind trial division tests to see if an integer *n*, the integer to be factored, can be divided by each number in turn that is less than *n*. For example, for the integer n = 12, the only numbers that divide it are 1, 2, 3, 4, 6, 12. Selecting only the largest powers of primes in this list gives that $12 = 3 \times 4 = 3 \times 2^2$.

- The algorithm 3.4.2 shows the pseudocode for this method. The algorithm has two loops, outer and inner. The outer loop finds unique factors and the inner loop finds duplicates of a factor.
- For example, $24=2^3*3$. The outer loop finds the factors 2 and 3. The inner loop finds that 2 is a multiple factor.

Algorithm 3.4.2 Pseudocode for trial-division factorization

```
Trial_Division_Factorization (n)          // n is the number to be factored
{
    a ← 2
    while (a ≤ √n )
    {
        while (n mod a = 0)
        {
            output a                      // Factors are output one by one
            n = n / a
        }
        a ← a + 1
    }
    if (n > 1) output n                   // n has no more factors
}
```

- **Example 1:** Use the trial division algorithm to find the factors of 1233.

Solution

We run a program based on the algorithm and get the following result.

$1233=3^2 * 137$

3.4.3 Fermat Method

- The Fermat's Factorization method is based on the representation of an odd integer as the difference of two squares. For an integer n, we want a and b such as:

$n = a^2 - b^2 = (a + b) (a - b)$

where (a + b) and (a - b) are the factors of the number n

- **Example**:

Input: n = 6557

Output: [79,83]

Explanation:

For the above value, the first try for a is ceil value of square root of 6557, which is 81.

Then,

$b^2 = 81^2 - 6557 = 4$, as it is a perfect square.

So, b = 2

So, the factors of 6557 are:

(a - b) = 81 - 2 = 79 &

(a + b) = 81 + 2 = 83.

3.4.4 Pollard p – 1 Method

- This method finds a prime factor p of a number based on the condition that p-1 has no factor larger than a predefined value B, which is called bound.

$$p = \gcd(2^{B!} - 1, n)$$

- Algorithm 3.4.4 shows the pseudocode **for pollard p - 1** factorization method.

Algorithm 3.4.4 Pseudocode for Pollard p – 1 factorization

```
Pollard_ (p − 1) _Factorization (n, B)          // n is the number to be factored
{
   a ← 2
   e ← 2
   while (e ≤ B)
   {
      a ← a^e mod n
      e ← e + 1
   }
   p ← gcd (a − 1, n)
   if 1 < p < n   return p
   return failure
}
```

Example

Use the Pollard $p - 1$ method to find a factor of 57247159 with the bound B = 8.

Solution

We run a program based on the algorithm and find that $p = 421$. As a matter of fact $57247159 = 421 \times 135979$. Note that 421 is a prime and $p - 1$ has no factor greater than 8

$$421 - 1 = 2^2 \times 3 \times 5 \times 7$$

3.4.5 Pollard's rho algorithm

- Pollard's rho algorithm is an algorithm for integer factorization. It was invented by John Pollard in 1975. It uses only a small amount of space, and its expected running

time is proportional to the square root of the size of the smallest prime factor of the composite number being factorized.

- Given a positive integer n, and that it is composite, find a divisor of it.
- **Example:**

  Input: n = 12;

  Output: 2 [OR 3 OR 4]

  Input: n = 187;

  Output: 11 [OR 17]

Concepts used in Pollard's Rho Algorithm

1. Two numbers x and y are said to be congruent modulo n (x = y modulo n) if
   - their absolute difference is an integer multiple of n, OR,
   - each of them leaves the same remainder when divided by n.
2. The Greatest Common Divisor is the largest number which divides evenly into each of the original numbers.
3. Birthday Paradox: The probability of two persons having same birthday is unexpectedly high even for small set of people.
4. Floyd's cycle-finding algorithm: If tortoise and hare start at same point and move in a cycle such that speed of hare is twice the speed of tortoise, then they must meet at some point.

3.5 Fermat's and Euler's Theorem

- Two theorems that play important roles in public-key cryptography are Fermat's theorem and Euler's theorem.

Fermat's Theorem (Fermat's Little Theorem)

- If p is prime and a is a positive integer not divisible by p, then

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

  where p is prime and gcd (a, p) = 1

Example 1:

If $a = 2$ and $p = 7$, then $2^6 = 64$, and $64 - 1 = 63 = 7 \times 9$ is thus a multiple of 7.

Example 2:

If a =7 and p = 19 solve using Fermat's theorem

$a^{18} \equiv 1 \bmod P$

$a^2 = 7^2 = 49 \equiv 11 \bmod 19$

$7^4 = 121 \bmod 19 = 7$

$7^8 = 7^4 * 7^4$

$= 7 * 7 = 49 \bmod 19 = 11$

$7^{16} = 7^8 * 7^8$

$= 11 * 11 = 121 \bmod 19 = 7$

$7^{18} = 7^{16} * 7^2$

$= 7 * 11$

$= 77 \bmod 19$

$=1$

Euler 's Totient Function

- Euler's totient function is a multiplicative function, meaning that if two numbers m and n are relatively prime, then φ(mn) = φ(m)φ(n).
- This function gives the order of the multiplicative group of integers modulo n (the group of units of the ring /n).
- It is also used for defining the RSA encryption system.
- The Euler's totient function, or phi (φ) function is a very important number theoretic function having a deep relationship to prime numbers and the so-called order of integers.
- The totient φ(*n*) of a positive integer *n* greater than 1 is defined to be the number of positive integers less than *n* that are coprime to *n*. φ(*1*) is defined to be 1.
- The following table shows the function values for the first several natural numbers:

Table 3.5 Some values of Euler's Totient Function φ(*n*)

n	φ(*n*)	numbers coprime to n
1	1	1
2	1	1
3	2	1, 2
4	2	1,3
5	4	1,2,3,4
6	2	1,5
7	6	1,2,3,4,5,6
8	4	1,3,5,7
9	6	1,2,4,5,7,8
10	4	1,3,7,9
11	10	1,2,3,4,5,6,7,8,9,10
12	4	1,5,7,11
13	12	1,2,3,4,5,6,7,8,9,10,11,12
14	6	1,3,5,9,11,13
15	8	1,2,4,7,8,11,13,14

- Table 3.5 lists the first 15 values of φ(*n*). the value *φ(1)* is without meaning but is defined to have the value 1.
- For the prime number p,

$$\varphi(p) = p-1$$

- Now, two prime numbers p and q with p ≠ q. Then we can show that, for n=pq.

$$\varphi(n) = \varphi(pq) = \varphi(p) * \varphi(q) = (p-1) * (q-1)$$

- For example, $\varphi(10) = \varphi(5) * \varphi(2) = (5-1) * (2-1) = 4 * 1 = 4$

$$\varphi(15) = \varphi(5) * \varphi(3) = (5-1) * (3-1) = 4 * 2 = 8$$

Euler's Theorem

- Euler's theorem states that, "if p and q are relatively prime, then", where φ is Euler's totient function for integers. That is, is the number of non-negative numbers that are less than q and relatively prime to q.

$$a^{(n)} \equiv 1 \pmod{n}$$

  - for any a, n where gcd (a, n) = 1

Example 1:

a=3, n= 10 prove Euler's theorem

Solution

$\varphi(n) = \varphi(10) => \varphi(p * q)$

$= \varphi(2) * \varphi(5)$ // 2 and 5 are relative prime numbers for 10

$= (2-1) * (5-1) = 4$

$= 3^4 \bmod 10$

$= 81 \bmod 10 = 1$

Example 2:

a=2; *n*=11 prove Euler's theorem

Solution

$\phi(11) = 10;$

hence $2^{10} = 1024 = 1 \bmod 11$

3.6 Chinese Remainder Theorem (CRT)

- The Chinese Remainder Theorem is an ancient, which is developed in 3rd centuries by Chinese mathematician Sun-tzu.
- The CRT is a result about congruences in number theory and its generalization in abstract algebra.
- It enables one to solve simultaneous equation with respect to different moduli in considerable generality.

Theorem

- Chinese Remainder Theorem: If $m_1, m_2, .., m_k$ are pairwise relatively prime positive integers, and if $a_1, a_2, .., a_k$ are any integers, then the simultaneous congruences

$$x \equiv a_1 \pmod{m_1},$$
$$x \equiv a_2 \pmod{m_2},$$
$$\vdots$$
$$x \equiv a_k \pmod{m_k}$$

have a solution, and the solution is unique modulo m, where $m = m_1, m_2 \cdots m_k$. That is a unique solution x with $0 \leq x \leq m$.

Algorithm

- Let $m = m_1, m_2, ..., m_k$
- Let $M_k = m / m_k$ for all K = 1, 2, 3, … k
- For all K = 1, 2, 3, … k find integers 1 / K such $M_k, Y_k \equiv (1 \bmod m_k)$

  Since gcd $(M_k, m_k) = 1$
- Euclid's extended algorithm can be used to find y_k
- The integer $x \equiv (a_1 M_1 y_1 + a_2 M_2 y_2 + ... + a_k M_k Y_k) \pmod{M}$ is a unique solution.

Example 1:

$x \equiv 1 \pmod 4$

$x \equiv 2 \pmod 5$

$x \equiv 4 \pmod 7$

solve the value for x using Chinese Remainder Theoren.

Solution

$m_1 = 4$ $a_1 = 1$

$m_2 = 5$ $a_2 = 2$

$m_3 = 7$ $a_3 = 4$

Step 1:

$m = m_1 * m_2 * m_3$

$= 4 * 5 * 7$

m = 140

Step 2:

$M_1 = m/ m_1 \Rightarrow 140/4 = 35$

$M_2 = m/m_2 \Rightarrow 140/5 = 28$

$M_3 = m/m_3 \Rightarrow 140/7 = 20$

Step 3:

$M_k Y_k \equiv 1 \pmod{mk}$

Put k=1

$M_1 y_1 \equiv 1 \pmod{m_1}$

$35y_1 \equiv 1 \pmod 4$

Put k = 2

$M_2 y_2 \equiv 1 \pmod{m_2}$

$28y_2 \equiv 1 \pmod 5$

Put k = 3

$M_3 y_3 \equiv 1 \pmod{m_3}$

$20y_3 \equiv 1 \pmod 7$

$35y_1 \equiv 1 \pmod 4$

⇨ **$28y_2 \equiv 1 \pmod 5$**

$20y_3 \equiv 1 \pmod 7$

To find y_1

$35y_1 \equiv 1 \pmod 4$

gcd (M_k, m_k)

gcd (35, 4)

gcd (4, 35 mod 4)

gcd (4, 3)

gcd (3, 4 mod 3)

gcd (3, 1) when n = 1

y1 = 3 gcd (m, n) = n

Similarly,

Find y_2 and y_3

Here, **$y_2 = 2$**

$y_3 = 6$

Step 4:

$x = (a_1 M_1 y_1 + a_2 M_2 y_2 + a_3 M_3 Y_3) \pmod m$

$= ((1 * 35 * 3) + (2 * 28 * 2) + (4 * 20 * 6)) \bmod 140$

$= (105 + 112 + 480) \bmod 140$

$= 697 \bmod 140$

x = 137

Example 2:

$x \equiv 3 \pmod 4$

$x \equiv 2 \pmod 3$

$x \equiv 4 \pmod 5$

solve the value for x using Chinese Remainder Theoren.

Solution

$m_1 = 4$ $a_1 = 3$

$m_2 = 3$ $a_2 = 2$

$m_3 = 5$ $a_3 = 4$

Step 1:

$m = m_1 * m_2 * m_3$

$= 4 * 3 * 5$

m = 60

Step 2:

$M_1 = m/ m_1 \Rightarrow 60/4 = 15$

$M_2 = m/m_2 \Rightarrow 60/3 = 20$

$M_3 = m/m_3 \Rightarrow 60/5 = 12$

Step 3:

$M_k Y_k \equiv 1 \pmod{mk}$

Put k=1

$M_1y_1 \equiv 1(\text{mod } m_1)$

$15y_1 \equiv 1 \text{ (mod 4)}$

Put k = 2

$M_2y_2 \equiv 1(\text{mod } m_2)$

$20y_2 \equiv 1 \text{ (mod 3)}$

Put k = 3

$M_3y_3 \equiv 1(\text{mod } m_3)$

$12y_3 \equiv 1 \text{ (mod 5)}$

$15y_1 \equiv 1$ (mod 4)

⇨ **$20y_2 \equiv 1$ (mod 3)**

$12y_3 \equiv 1$ (mod 5)

Find **$y_1 = 3$**

$y_2 = 2$

$y_3 = 3$

Step 4:

$x = (a_1 M_1 y_1 + a_2 M_2 y_2 + a_3 M_3 Y_3) \text{ (mod m)}$

$= ((3 * 15 * 3) + (2 * 20 * 2) + (4 * 12 * 3)) \text{ mod } 60$

$= (135 + 80 + 144) \text{ mod } 60$

$= 359 \text{ mod } 60$

x = 59

3.7 Exponentiation and Logarithm

3.7.1 Modular Exponentiation

- Modular exponentiation is a type of exponentiation performed over a modulus.
- It is useful in computer science, especially in the field of cryptography. The operation of modular exponentiation calculates the remainder when an integer b (the base) raised to the eth power (the exponent), b^e, is divided by a positive integer m (the modulus).
- In symbols, given base b, exponent e, and modulus m, the modular exponentiation c is: $c = b^e \text{ mod } m$.

- From the definition of c, it follows that $0 \leq c < m$. For example, given $b = 5$, $e = 3$ and $m = 13$, the solution $c = 8$ is the remainder of dividing $5^3 = 125$ by 13.
- Modular exponentiation can be performed with a negative exponent e by finding the modular multiplicative inverse d of b modulo m using the extended Euclidean algorithm.
- That is: $c = b^e \bmod m = d^{-e} \bmod m$, where $e < 0$ and $b \cdot d \equiv 1 \pmod{m}$.
- Modular exponentiation similar to the one described above is considered easy to compute, even when the integers involved are enormous.
- On the other hand, computing the modular discrete logarithm – that is, the task of finding the exponent e when given b, c, and m – is believed to be difficult.
- This one-way function behaviour makes modular exponentiation a candidate for use in cryptographic algorithms.

3.7.2 Discreate Logarithms

- In the mathematics of the real numbers, the logarithm $\log_b a$ is a number x such that $b^x = a$, for given numbers a and b. Analogously, in any group G, powers b^k can be defined for all integers k, and the discrete logarithm $\log_b a$ is an integer k such that $b^k = a$.
- In number theory, the more commonly used term is index: we can write $x = \operatorname{ind}_r a \pmod m$ (read the index of a to the base r modulo m) for $r^x \equiv a \pmod m$ if r is a primitive root of m and gcd(a, m) = 1.
- Discrete logarithms are quickly computable in a few special cases. However, no efficient method is known for computing them in general.
- Several important algorithms in public-key cryptography base their security on the assumption that the discrete logarithm problem over carefully chosen groups has no efficient solution.
- Let G be any group. Denote its group operation by multiplication and its identity element by 1. Let b be any element of G. For any positive integer k, the expression b^k denotes the product of b with itself k times:

$$b^k = \underbrace{b \cdot b \cdots b}_{k \text{ factors}}.$$

- Similarly, let b^{-k} denote the product of b^{-1} with itself k times. For k = 0, the kth power is the identity: $b^0 = 1$.
- Let a also be an element of G. An integer k that solves the equation $b^k = a$ is termed a discrete logarithm (or simply logarithm, in this context) of a to the base b. One writes $k = \log_b a$.

3.8 Asymmetric Key Ciphers

- Asymmetric cryptography, also known as public key cryptography, uses public and private keys to encrypt and decrypt data.
- The keys are simply large numbers that have been paired together but are not identical (asymmetric). One key in the pair can be shared with everyone; it is called the public key.

3.8.1 Public Key Cryptography

- 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 −

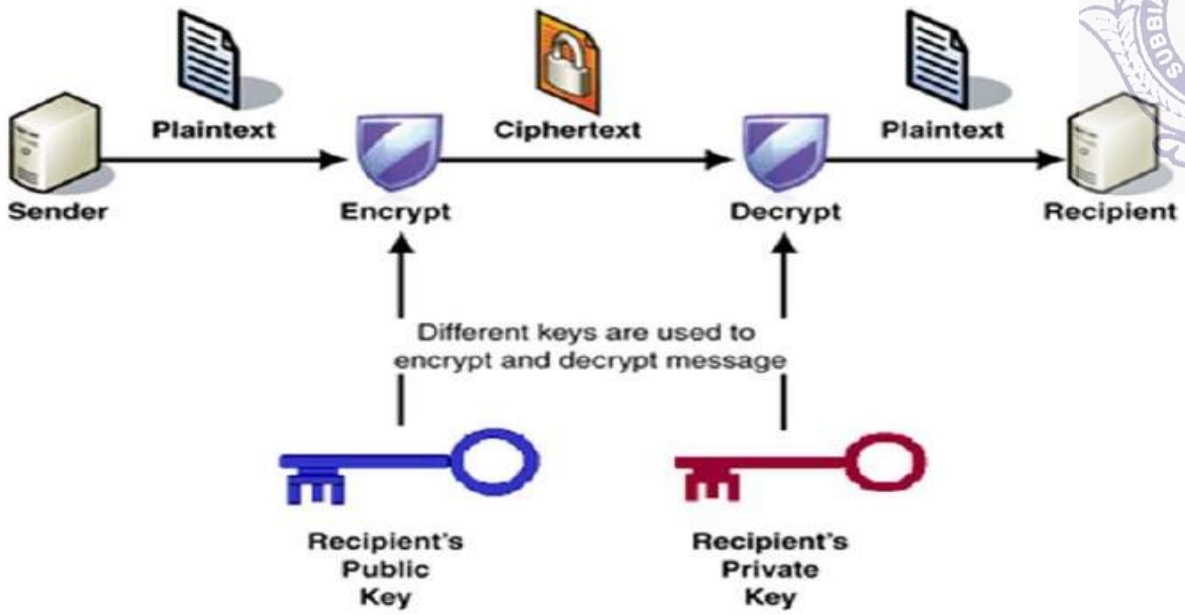


Figure 3.1 Public Key Cryptosystem

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.

3.9 RSA Cryptosystem

- This cryptosystem is one the initial system. It remains most employed cryptosystem even today. The system was invented by three scholars **Ron Rivest, Adi Shamir,** and **Len Adleman** and hence, it is termed as RSA cryptosystem.
- We will see two aspects of the RSA cryptosystem, firstly generation of key pair and secondly encryption-decryption algorithms.

Generation of RSA Key Pair

- Each person or a party who desires to participate in communication using encryption needs to generate a pair of keys, namely public key and private key. The process followed in the generation of keys is described below −
  - **Generate the RSA modulus (n)**
    - Select two large primes, p and q.
    - Calculate n=p*q. For strong unbreakable encryption, let n be a large number, typically a minimum of 512 bits.
    - note ø(n) = (p-1) (q-1)
  - **Find Derived Number (e)**
    - Number **e** must be greater than 1 and less than (p − 1) (q − 1).
    - There must be no common factor for e and (p − 1) (q − 1) except for 1. In other words, two numbers e and (p – 1) (q – 1) are co-prime.
  - **Form the public key**
    - The pair of numbers (n, e) form the RSA public key and is made public.
    - Interestingly, though n is part of the public key, difficulty in factorizing a large prime number ensures that attacker cannot find in finite time the two primes (p & q) used to obtain n. This is strength of RSA.
  - **Generate the private key**
    - Private Key d is calculated from p, q, and e. For given n and e, there is unique number d.

- Number d is the inverse of e modulo (p - 1) (q – 1). This means that d is the number less than (p - 1) (q - 1) such that when multiplied by e, it is equal to 1 modulo (p - 1) (q - 1).
- Calculate d, $d \equiv e^{-1} (\text{mod } ø(n))$
- This relationship is written mathematically as follows –
  - $ed = 1 \text{ mod } (p - 1) (q - 1)$

- **Public key PU = {e, n}**
- **Private key PR = {d, n}**

Key Generation	
Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$de \bmod \phi(n) = 1$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

Encryption	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod n$

Decryption	
Ciphertext:	C
Plaintext:	$M = C^d \pmod n$

Figure 3.2 The RSA Algorithm

Encryption and Decryption

- To encrypt a message M the sender:
  - Computes: **$C=M^e \bmod N$**, where $0 \leq M < N$
- To decrypt the ciphertext C the receiver
  - Computes: **$M=C^d \bmod N$**

Example

- Select primes: p=17 & q=11
- Compute $n = p \times q = 17 \times 11 = 187$
- Compute $ø(n) = (p–1)(q-1) = 16 \times 10 = 160$
- Select e: gcd (e,160) = 1; choose e=7

- Determine d: de = 1 mod 160 and d < 160 Value is d = 23 since 23×7=161= 10×16+1
- Publish public key PU = {7, 187}
- Keep secret private key PR = {23, 187}

Encryption

- Given message (Plaintext) M = 88

  $88^7 \bmod 187 = [(88^4 \bmod 187) \times 88^2 \bmod 187) \times 88^1 \bmod 187)] \bmod 187$

  $88^1 \bmod 187 = 88$

  $88^2 \bmod 187 = 7744 \bmod 187 = 77$

  $88^4 \bmod 187 = 59{,}969{,}536 \bmod 187 = 132$

  $88^7 \bmod 187 = (88 \times 77 \times 132) \bmod 187$

  $= 8{,}94432 \bmod 187$

  $= 11$

  So, Ciphertext C = 11

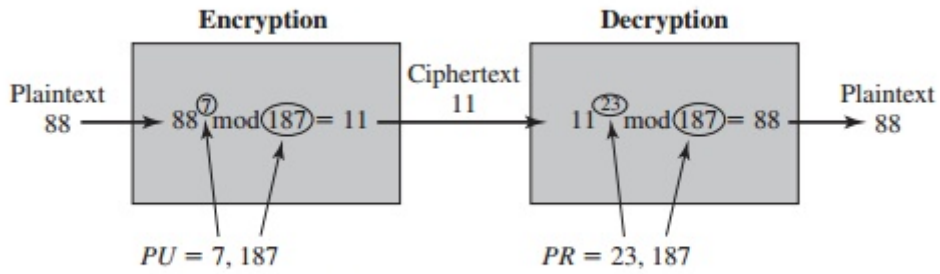


Figure 3.3 Example of RSA Algorithm

Decryption

- $M = 11^{23} \bmod 187$

  $11^{23} \bmod 187 = [(11^1 \bmod 187) \times (11^2 \bmod 187) \times (11^4 \bmod 187) \times (11^8 \bmod 187) \times (11^8 \bmod 187)] \bmod 187$

  $11^1 \bmod 187 = 11$

  $11^2 \bmod 187 = 121$

  $11^4 \bmod 187 = 14641 \bmod 187 = 55$

  $11^8 \bmod 187 = 2{,}14{,}358{,}881 \bmod 187 = 33$

  $11^8 \bmod 187 = 2{,}14{,}358{,}881 \bmod 187 = 33$

11^{23} mod 187 = (11 x 121 x 55 x 33 x 33) mod 187

= 79, 720, 245 mod 187

= 88

So, Plaintext M =88

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.

3.10 Key Management

- Key management refers to management of cryptographic keys in a cryptosystem. Figure 3.4 illustrates the lifecycle of key management.

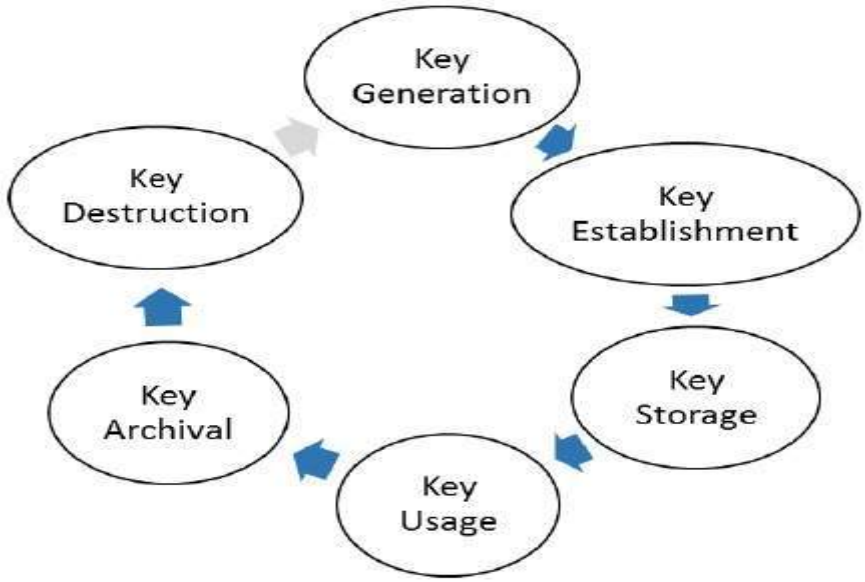


Fig. 3.4 Lifecycle of Key Management

- This includes dealing with the generation, exchange, storage, use, crypto-shredding (destruction) and replacement of keys. Successful key management is critical to the security of a cryptosystem.
- In cryptography it is a very tedious task to distribute the public and private key between sender and receiver.
- If key is known to the third party (forger/eavesdropper) then the whole security mechanism becomes worthless. So, there comes the need to secure the exchange of keys.
- There are 2 aspects for Key Management:
  - Distribution of public keys.
  - Use of public-key encryption to distribute secret.

3.10.1 **Distribution of Public Keys**

- Several techniques have been proposed for the distribution of public keys which can mostly be grouped into the categories shown.
  - Public announcement
  - Publicly available directory
  - Public-key authority
  - Public-key certificates

Public Announcement

- The point of public-key encryption is that the public key is public, hence any participant can send his or her public key to any other participant, or broadcast the key to the community at large. eg. append PGP keys to email messages or post to news groups or email list.
- Figure 3.5 illustrates the public key distribution

Figure 3.5 Uncontrolled Public Key Distribution

- Its major weakness is forgery, anyone could pretend to be user A and send a public key to another participant or broadcast such a public key. Until the forgery is discovered they can masquerade as the claimed user.

Publicly Available Directory

- The user obtains greater security by registering keys with a public directory.
- The directory must be trusted with properties:
  - The authority maintains a directory with a {name, public key} entry for each participant.
  - Each participant registers a public key with the directory authority.
  - A participant may replace the existing key with a new one at any time because the corresponding private key has been compromised in some way.
  - Participants could also access the directory electronically. For this purpose, secure, authenticated communication from the authority to the participant is mandatory.
  - Figure 3.6 illustrates the public key publication

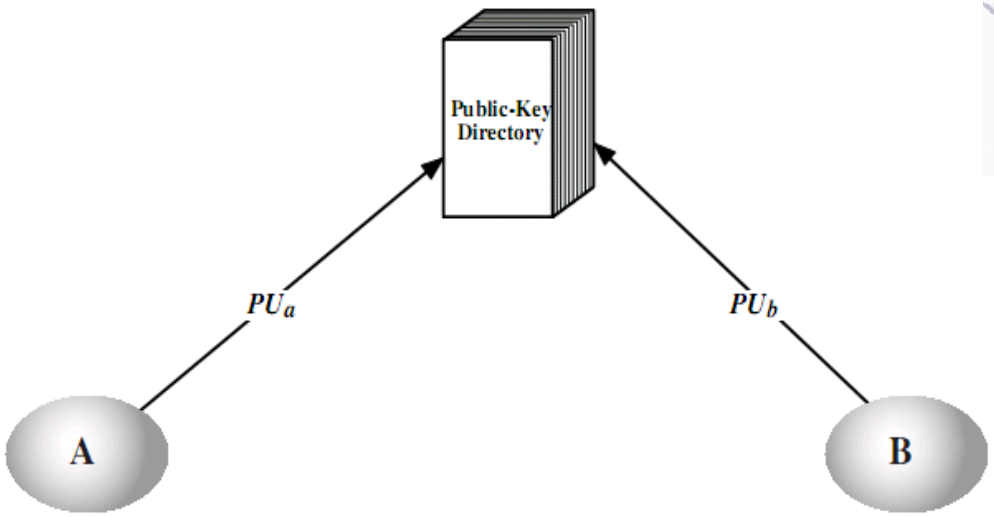


Figure 3.6 Public Key Publication

- This scheme is clearly more secure than individual public announcements but still has vulnerabilities.
- If an adversary succeeds in obtaining or computing the private key of the directory authority, the adversary could authoritatively pass out counterfeit public keys and subsequently impersonate any participant and eavesdrop on messages sent to any participant.
- Another way to achieve the same end is for the adversary to tamper with the records kept by the authority.

Public-Key Authority

- Stronger security for public-key distribution can be achieved by providing tighter control over the distribution of public keys from the directory.
- It requires users to know the public key for the directory, and that they interact with directory in real-time to obtain any desired public key securely.
- Totally seven messages are required.
- Figure 3.7 illustrates the public key distribution Scenario

1. A sends a timestamped message to the public-key authority containing a request for the current public key of B.
2. The authority responds with a message that is encrypted using the authority's private key, PR_{auth} Thus, A is able to decrypt the message using the authority's public key. Therefore, A is assured that the message originated with the authority. The message includes the following:
    - B's public key, PU_b which A can use to encrypt messages destined for B.

- The original request, to enable A to match this response with the corresponding earlier request and to verify that the original request was not altered before reception by the authority.
- The original timestamp, so A can determine that this is not an old message from the authority containing a key other than B's current public key.

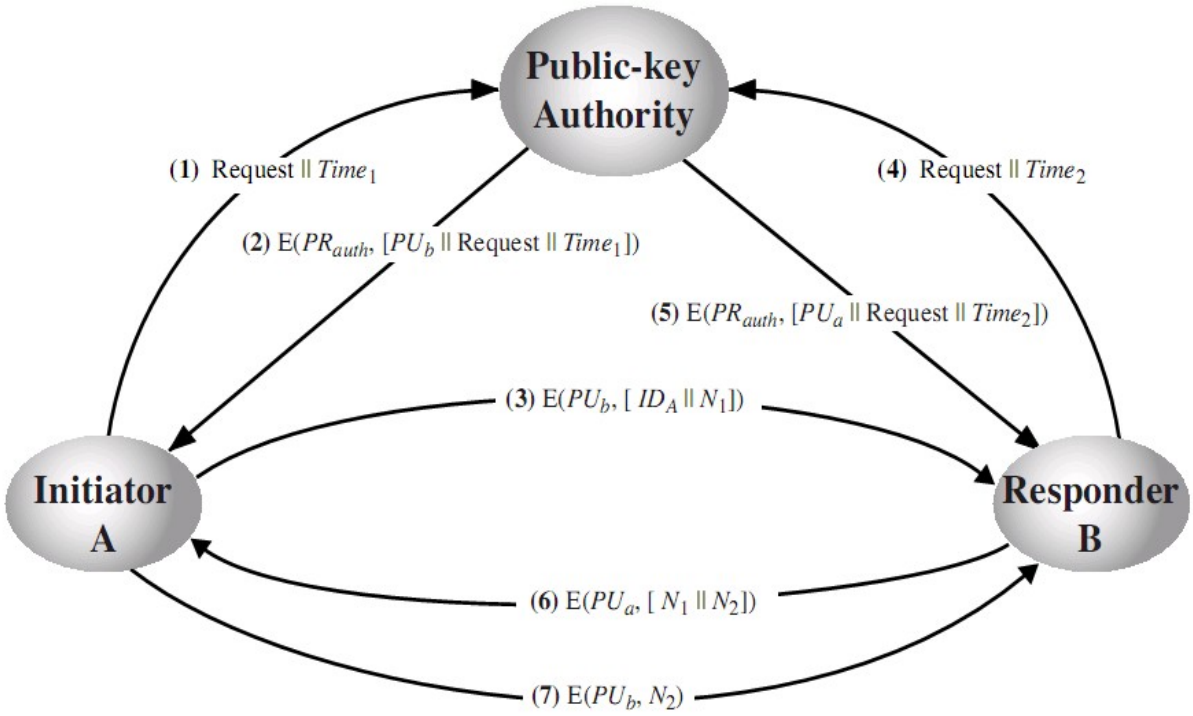


Figure 3.7 Public key distribution Scenario

3. A stores B's public key and also uses it to encrypt a message to B containing an identifier of A (ID_A) and a nonce (N_1), which is used to identify this transaction uniquely.
4. B retrieves A's public key from the authority in the same manner as A retrieved B's public key.
5. At this point, public keys have been securely delivered to A and B, and they may begin their protected exchange. However, two additional steps are desirable:
6. B sends a message to A encrypted with PU_a and containing A's nonce (N_1) as well as a new nonce generated by B (N_2) Because only B could have decrypted message (3), the presence of N_1 in message (6) assures A that the correspondent is B.
7. A returns N_2, encrypted using B's public key, to assure B that its correspondent is A.

Public-Key Certificates

- A user must appeal to the authority for a public key for every other user that it wishes to contact and it is vulnerable to tampering too.
- Public key certificates can be used to exchange keys without contacting a public-key authority.
- Figure 3.8 illustrates the public key Certificate exchanges
- A certificate binds an **identity** to **public key**, with all contents **signed** by a trusted Public- Key or Certificate Authority (CA).
- This can be verified by anyone who knows the public-key authorities public-key.
- A participant can also convey its key information to another by transmitting its certificate.
- Other participants can verify that the certificate was created by the authority. We can place the following requirements on this scheme:

1. Any participant can read a certificate to determine the name and public key of the certificate's owner.
2. Any participant can verify that the certificate originated from the certificate authority and is not counterfeit.
3. Only the certificate authority can create and update certificates.
4. Any participant can verify the currency of the certificate.

- One scheme has become universally accepted for formatting public-key certificates.
- The X.509 standard. X.509 certificates are used in most network security applications, including IP security, secure sockets layer (SSL), secure electronic transactions (SET), and S/MIME.

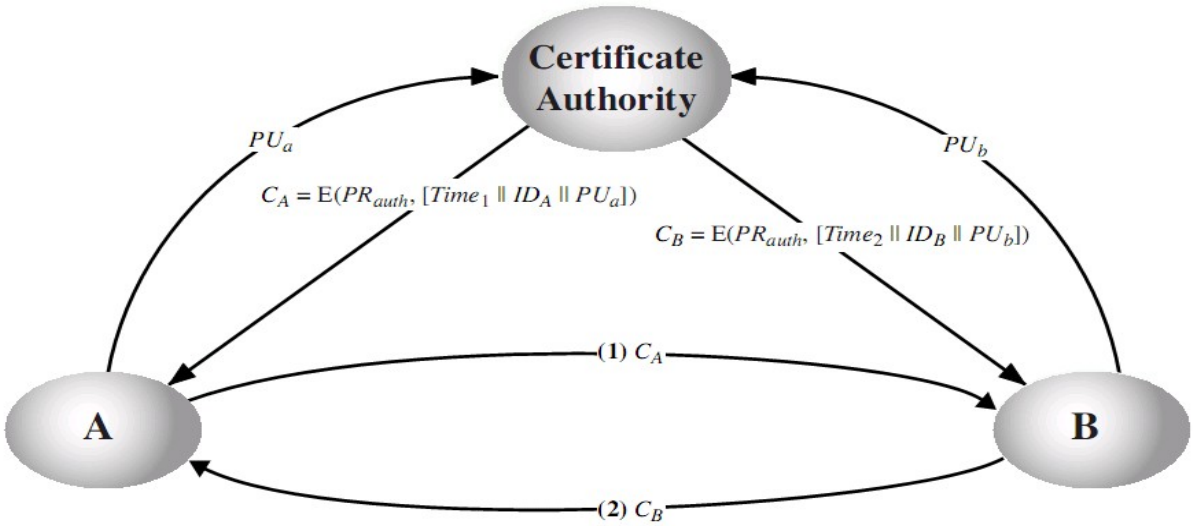


Figure 3.8 Public Key Certificates

3.10.2 Symmetric Key Distribution using Public Key Cryptography

- Once public keys have been distributed or have become accessible, secure communication that thwarts eavesdropping, tampering, or both, is possible.
- The Public-key encryption provides for the distribution of secret keys to be used for conventional encryption.

Simple Secret Key Distribution

- A generates a public/private key pair {PUa, PRa} and transmits a message to B consisting of PUa and an identifier of A, IDA.
- B generates a secret key, Ks, and transmits it to A, encrypted with A's public key.
- A computes D(PRa, E(PUa, Ks)) to recover the secret key. Because only A can decrypt the message, only A and B will know the identity of Ks.
- A discards PUa and PRa and B discards PUa.

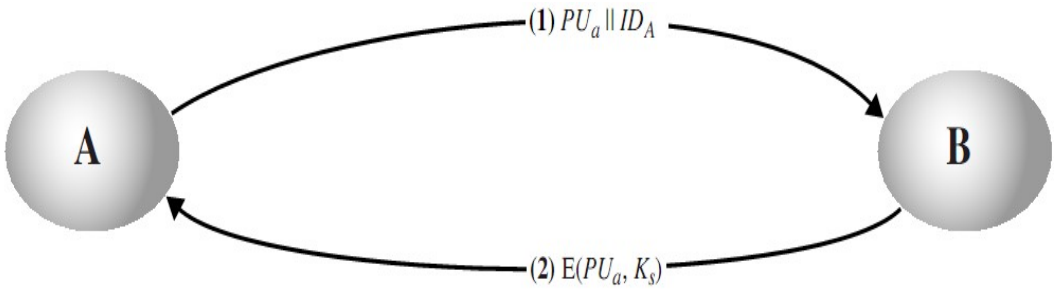


Figure 3.9 Simple Secret Key Distribution

- Figure 3.9 illustrates the simple secret key distribution
- Here third party can intercept messages and then either relay the intercepted message or substitute another message Such an attack is known as a **man-in-the-middle attack.** Figure 3.10 shows the mam-in-the-middle attack.

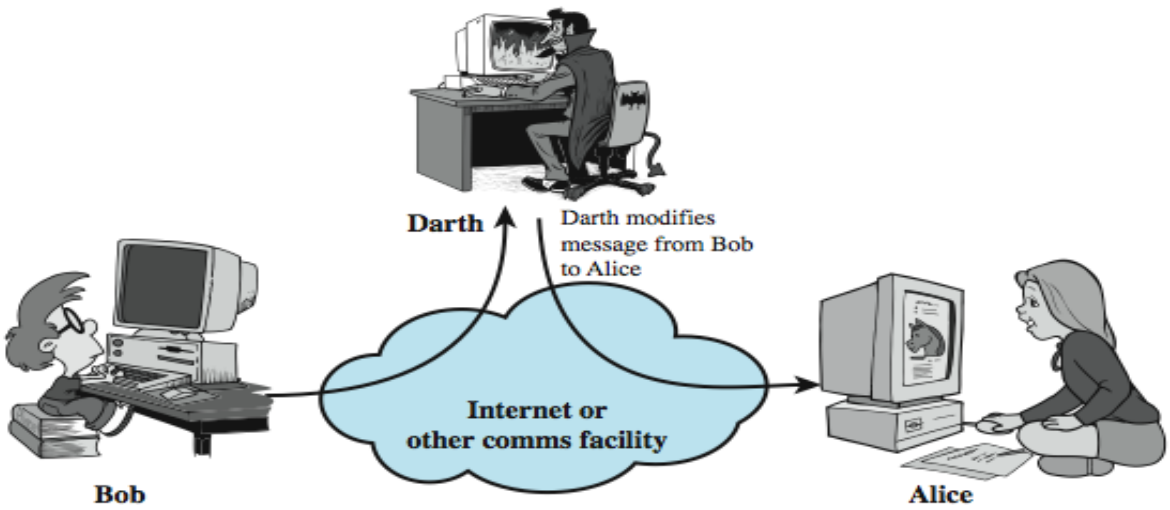


Figure 3.10 Man-in-the-Middle Attack

Secret Key Distribution with Confidentiality and Authentication

- A uses B's public key to encrypt a message to B containing an identifier of A (ID_A) and a nonce (N_1), which is used to identify this transaction uniquely.
- B sends a message to A encrypted with PU_a and containing A's nonce (N_1) as well as a new nonce generated by B (N_2) Because only B could have decrypted message (1), the presence of N_1 in message (2) assures A that the correspondent is B.
- Figure 3.11 illustrates the secret key distribution

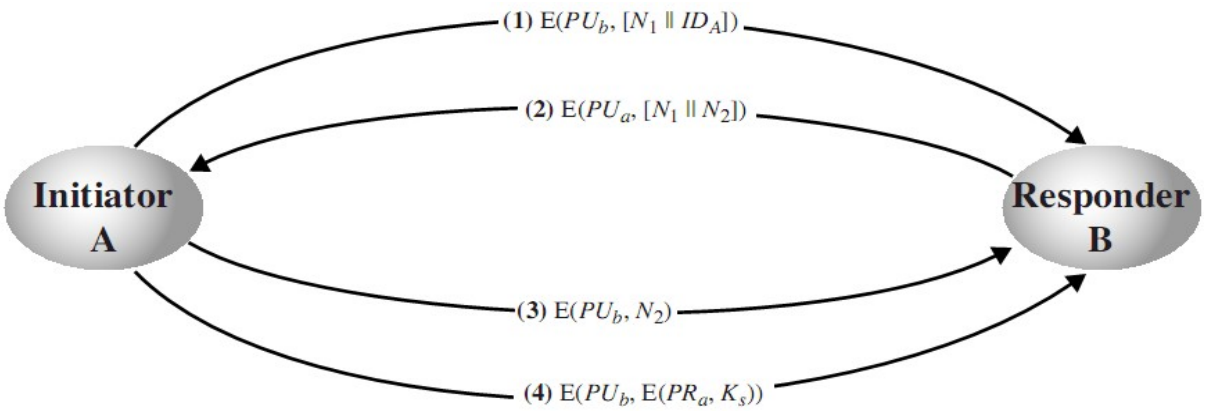


Figure 3.11 Secret Key Distribution

- A returns N_2 encrypted using B's public key, to assure B that its correspondent is A.
- A selects a secret key K_s and sends $M = E(PU_b, E(PR_a, K_s))$ to B. Encryption of this message with B's public key ensures that only B can read it; encryption with A's private key ensures that only A could have sent it.
- B computes $D(PU_a, D(PR_b, M))$ to recover the secret key.

A Hybrid Scheme

- Another way to use public-key encryption to distribute secret keys is a hybrid approach.
- This scheme retains the use of a Key Distribution Center (KDC) that shares a secret master key with each user and distributes secret session keys encrypted with the master key.
- A public key scheme is used to distribute the master keys.

- The addition of a public-key layer provides a secure, efficient means of distributing master keys.

3.11 Diffie-Hellman Key Exchange Algorithm

- The Diffie–Hellman key exchange or Key Agreement is a method of securely exchanging cryptographic keys over a public channel.
- It was developed by Whitfield Diffie and Martin Hellman in 1976.
- This protocol allows two users to exchange a secret key over an untrusted network without any prior secrets. Security of transmission is critical for many network and Internet applications.
- A number of commercial products employ this exchange technique.
- The purpose of the algorithm is to enable two users to securely exchange a key that can be used for subsequent encryption of messages. So, two persons can talk in untrusted network.
- The D-H, Based on the difficulty of computing discrete logarithms of large numbers.
- Suppose A and B wish to exchange a secret key, the following steps are needed.
  - There are two publicly known numbers: one is prime number q and an integer α that is primitive root of q.
  - Suppose the user A and B wish to exchange a key.
  - User A selects a random integer $X_A < q$ and
    - computes $Y_A = \alpha^{xA} \bmod q$.
  - Similarly, user B selects a random integer $X_B < q$ and
    - computes $Y_B = \alpha^{xB} \bmod q$.
- Then user A computes the key as $K= (Y_B)^{xA} \bmod q$
- User B computes $K= (Y_A)^{xB} \bmod q$
- Then two calculations produce identical results.

Global Public Elements	
q	prime number
α	$\alpha < q$ and α a primitive root of q

User A Key Generation	
Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{XA} \bmod q$

User B Key Generation	
Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{XB} \bmod q$

Calculation of Secret Key by User A

$K = (Y_B)^{XA} \bmod q$

Calculation of Secret Key by User B

$K = (Y_A)^{XB} \bmod q$

Figure 3.12 The D-H Algorithm

Example 1:

- Choose global public elements

  q=23, α = 9

- User A select value **X_A is 4**
- Calculate public Y_A

$Y_A = \alpha^{xA} \bmod q$

$= 9^4 \bmod 23$

$= 6561 \bmod 23$

$Y_A = 6$

- User B select value **X_B is 3**
- Calculate public Y_B

$Y_B = \alpha^{xB} \bmod q$

$= 93 \bmod 23$

$= 729 \bmod 23$

$Y_B = 16$

- **Now, exchange their public keys**
- Figure 3.13 shows the exchange of keys

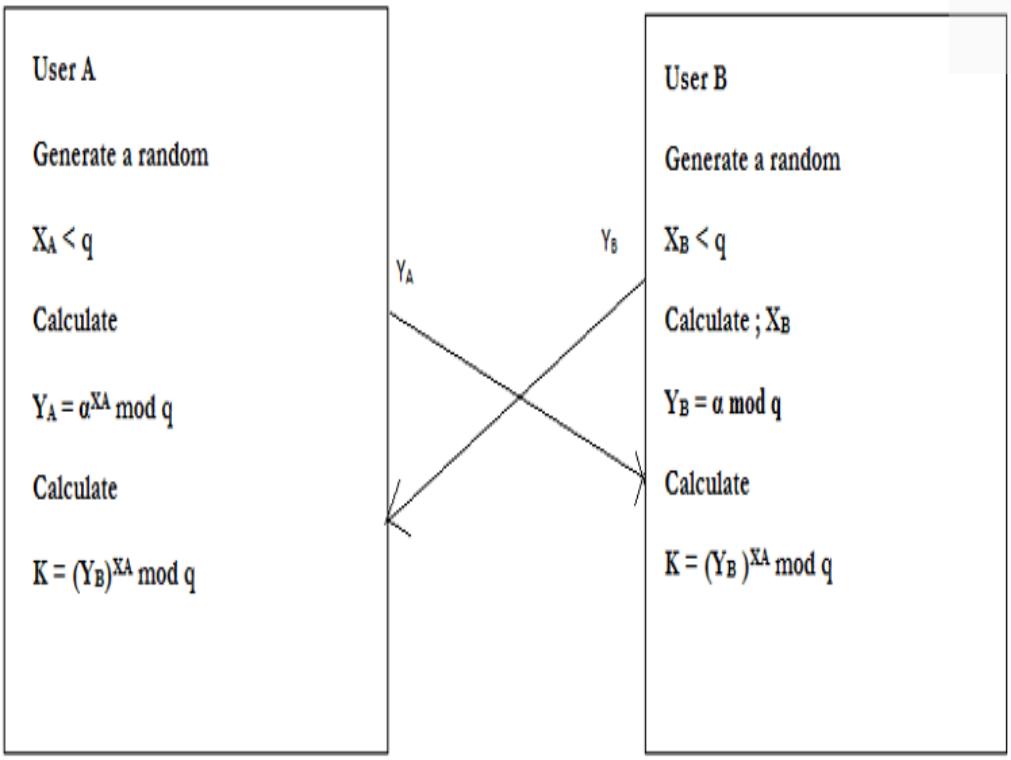


Figure 3.13 Diffie- Hellmam Key Exchange

After exchange their public keys, each can compute the common key.

- A compute $K = (Y_B)^{xA} \bmod q$
  $= 16^4 \bmod 23$
  $= 65536 \bmod 23$
  K = 9
- B compute $K = (Y_A)^{xB} \bmod q$
  $= 6^3 \bmod 23$
  $= 216 \bmod 23$
  K = 9

Now A and B can talk securely

Example 2:

- users Alice & Bob who wish to swap keys:
- agree on prime q=353 and a=3
- select random secret keys:
  - A chooses X_A=97, B chooses X_B=233
- compute respective public keys:

- $Y_A=3^{97} \bmod 353 = 40$ (Alice)
- $Y_B=3^{233} \bmod 353 = 248$ (Bob)

➢ compute shared session key as:

- $K_{AB}= Y_B^{xA} \bmod 353 = 248^{97} = 160$ (Alice)
- $K_{AB}= Y_A^{xB} \bmod 353 = 40^{233} = 160$ (Bob)

Advantages

- ➢ The sender and receiver don't need any prior knowledge of each other.
- ➢ Once the keys are exchanged, the communication of data can be done through an insecure channel.
- ➢ The sharing of the secret key is safe.

Disadvantages

- ➢ The algorithm cannot be sued for any asymmetric key exchange.
- ➢ Similarly, it cannot be used for signing digital signatures.
- ➢ Since it doesn't authenticate any party in the transmission, the Diffie Hellman key exchange is susceptible to a man-in-the-middle attack.

Man-in-the-Middle Attack

Figure 3.14 Man-in-the-Middle Attack

1. Darth prepares by creating two private / public keys
2. Alice transmits her public key to Bob
3. Darth intercepts this and transmits his first public key to Bob. Darth also calculates a shared key with Alice
4. Bob receives the public key and calculates the shared key (with Darth instead of Alice)
5. Bob transmits his public key to Alice
6. Darth intercepts this and transmits his second public key to Alice. Darth calculates a shared key with Bob
7. Alice receives the key and calculates the shared key (with Darth instead of Bob)

Now, Darth can then intercept, decrypt, re-encrypt, forward all messages between Alice & Bob

Applications

- Diffie-Hellman is currently used in many protocols, namely:
    - Secure Sockets Layer (SSL)/Transport Layer Security (TLS)
    - Secure Shell (SSH)
    - Internet Protocol Security (IPSec)
    - Public Key Infrastructure (PKI)

3.12 ElGamal cryptosystem

- In cryptography, ElGamal encryption is an public-key cryptosystem. It uses asymmetric key encryption for communicating between two parties and encrypting the message, which is based on the Diffie–Hellman key exchange algorithm.
- It is developed by Elgamal in 1984.
- The Elgamal algorithm consists of three components.
    - Key Generation
    - Encryption Algorithm
    - Decryption Algorithm

Key Generation

- Like D-H algorithm, generate of global elements are a prime number q and α
- User A generate private/public key pair as follows:
    - Generate a random integer X_A, $1 < X_A < q-1$
    - Compute their public key $Y_A = \alpha^{X_A} \bmod q$
    - A's private key is X_A, and public key $\{q, \alpha, Y_A\}$

Encryption

- User B that has access to A's public key can encrypt a message as follows
    - Represent message M in range $0 \leq M \leq q-1$
    - Longer messages are sent as a sequence of blocks, with each block being an integer less than q.
    - Choose random integer k with $1 \leq k \leq q-1$
    - Compute one-time key $K = (YA)^k \bmod q$
    - Encrypt M as a pair of integers (C_1, C_2) where

- **$C_1 = \alpha^k \bmod q$; $C_2 = KM \bmod q$**

Decryption

- User A recovers the plaintext.
  - Recover the key by computing K as $K = (C_1)^{X_A} \bmod q$
  - computing M as $M = (C_2\ K^{-1}) \bmod q$

Example

- ***Alice generates a public/private key pair; Bob encrypts using Alice's public key and Alice decrypts using her private key***
- Global elements q = 19, α = 10

Alice Generates a key pair as follows:

- *Alice chooses X_A=5*
- *Computes* $Y_A = \alpha^{X_A} \bmod q$ => $Y_A = 10^5 \bmod 19$

  $= 10000 \bmod 19$

  $\boldsymbol{Y_A = 3}$
- ***Alice private key is 5; public key*** $\{q, \alpha, Y_A\} = \{19, 10, 3\}$

Suppose Bob wants to send the message with the value M = 17, then

Encryption

- ***Bob choose K = 6***
  - $k = (YA)^k \bmod q => 3^6 \bmod 19$

    $= 729 \bmod 19$

    $\boldsymbol{k = 7}$
- ***Calculate*** $\mathbf{C_1}$
  - $C_1 = \alpha^k \bmod q => 10^6 \bmod 19$

    $= 1000000 \bmod 19$

    $\boldsymbol{C_1 = 11}$
- ***Calculate*** $\boldsymbol{C_2}$
  - $C_2 = KM \bmod q => 7 * 17 \bmod 19$

    $= 119 \bmod 19$

    $\mathbf{C_2 = 5}$
- ***Bob sends the ciphertext (11, 5)***

Decryption

- *Alice Calculates recover* $K = (C_1)^{X_A} \bmod q = 11^5 \bmod 19 = 7$

- *Compute inverse $K^{-1} = 7^{-1} = 11$*
- *Finally, $M = (C_2 K^{-1}) \bmod q = 5 * 11 \bmod 19$*

$= 55 \bmod 19$

M= 17

3.13 Elliptic Curve Arithmetic

- Most of the products and standards that use public-key cryptography for encryption and digital signatures use RSA.
- The key length for secure RSA use has increased over recent years, and this has put a heavier processing load on applications using RSA. This burden has ramifications, especially for electronic commerce sites that conduct large numbers of secure transactions.
- The principal attraction of ECC, compared to RSA, is that it appears to offer equal security for a far smaller key size, thereby reducing processing overhead.

3.13.1 Why the name Elliptic Curve?

- The mathematical properties of certain cubic equations that are today known as elliptic curves were seen to be generalizations of those of conics.
- However, the advent of calculus helped highlight marked differences between conics and elliptic curves. While conic sections can be parameterized by rational functions, elliptic curves cannot be parameterized by rational functions.
- The simplest functions that can parameterize elliptic curves are elliptic functions encountered in calculus as the inverses of so-called elliptic integrals.
- The Elliptic integrals are called so, as a typical example is the integral for the arc length of an ellipse. Thus, the name elliptic curve.

3.13.2 Abelian Groups

- An abelian **group** G, sometimes denoted by {G, •}, is a set of elements with a binary operation, denoted by •, that associates to each ordered pair (a, b) of elements in G an element (a • b) in G, such that the following axioms are obeyed:
  - **Closure:** If a and b belong to G, then a • b is also in G.
  - **Associative:** a • (b • c) = (a • b) • c for all a, b, c in G.
  - **Identity element:** There is an element e in G such that a • e = e • a = a for all a in G.

- **Inverse element:** For each a in G there is an element a' in G such that a • a' = a' • a = e.
- **Commutative:** a • b = b • a for all a, b in G.

- A number of public-key ciphers are based on the use of an abelian group. For example, Diffie-Hellman key exchange involves multiplying pairs of nonzero integers modulo a prime number q.
- The Keys are generated by exponentiation over the group, with exponentiation defined as repeated multiplication. For example, a^k mod q

$$q = \underbrace{(a \times a \times \ldots \times a)}_{k \text{ times}} \text{ mod q.}$$

- To attack Diffie-Hellman, the attacker must determine k given a and a^k;
- For elliptic curve cryptography, an operation over elliptic curves, called addition, is used. Multiplication is defined by repeated addition. For example,

$$a \times k = \underbrace{(a + a + \ldots + a)}_{k \text{ times}}$$

where the addition is performed over an elliptic curve. The Cryptanalysis involves determining k given a and (a x k).

3.13.3 Elliptic Curves over Real Numbers

- Elliptic curves are not ellipses. They are so named because they are described by cubic equations, similar to those used for calculating the circumference of an ellipse.
- In general, cubic equations for elliptic curves take the form

$$y^2 + axy + by = x^3 + cx^2 + dx + e$$

where a, b, c, d, and e are real numbers and x and y take on values in the real numbers. For our purpose, it is sufficient to limit ourselves to equations of the form

$$y^2 = x^3 + ax + b$$

- Such equations are said to be cubic, or of degree 3, because the highest exponent they contain is a 3. Also included in the definition of an elliptic curve is a single element denoted O and called the *point at infinity* or the *zero point*, which we discuss subsequently. To plot such a curve, we need to compute

$$y = \sqrt{x^3 + ax + b}$$

- For given values of *a* and *b*, the plot consists of positive and negative values of *y* for each value of *x*. Thus, each curve is symmetric about $y = 0$. Figures 3.15 shows two examples of elliptic curves.

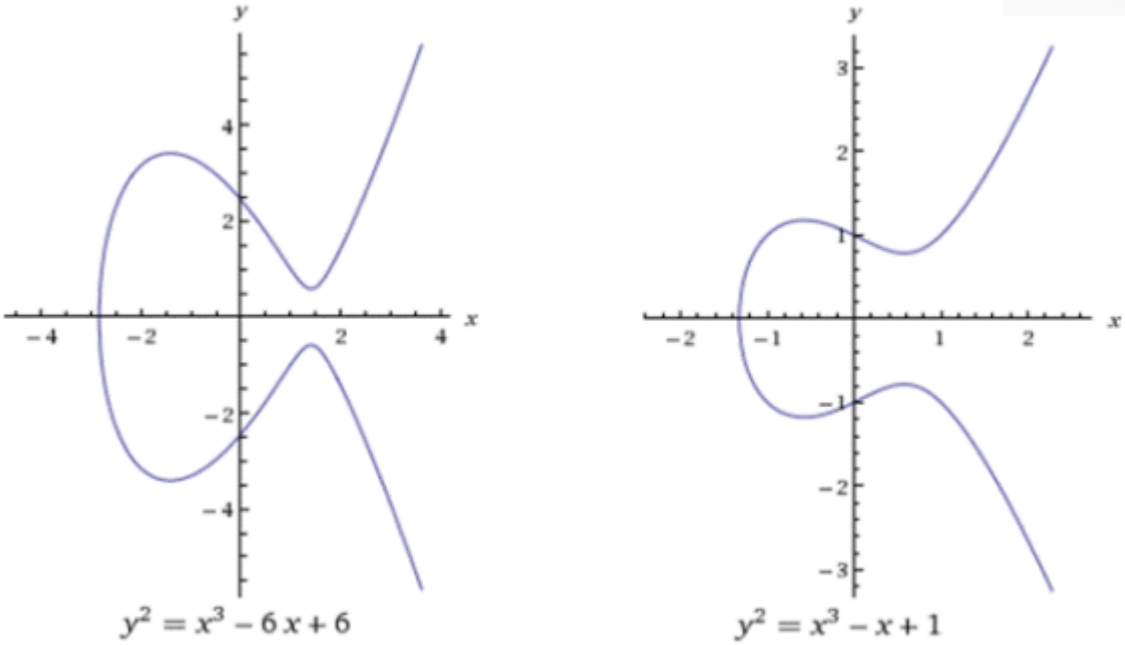


Figure 3.15 Examples of Elliptic Curve

3.14 Elliptic curve cryptography (ECC)

- Diffie-Hellman and RSA cryptographic methods are based on the creation of keys by using very large prime numbers. Hence, key creation requires a lot computational power.

- **Elliptic curve cryptography (ECC) is a public key encryption technique based on an elliptic curve theory that can be used to create faster, smaller, and more efficient cryptographic keys.**

- **ECC generates keys through the properties of the elliptic curve equation instead of the traditional method of generation as the product of very large prime numbers.**

- The technology can be used in conjunction with most public key encryption methods, such as RSA and Diffie-Hellman.

- The ECC can achieve the same level of security with a 164-bit key that other systems require a 1,024-bit key. Because ECC helps to establish equivalent security with lower

computing power and battery resource usage, it is becoming widely used for mobile applications. The use of elliptic curves in cryptography was suggested independently by Neal Koblitz and Victor S. Miller in 1985 and elliptic curve cryptography algorithms entered wide use around 2004.

3.14.1 How Does ECC work?

- An elliptic curve is the set of points that satisfy a specific mathematical equation. The equation for an elliptic curve looks like this $y^2=x^3+ax+b$ and is being represented graphically like the image below.

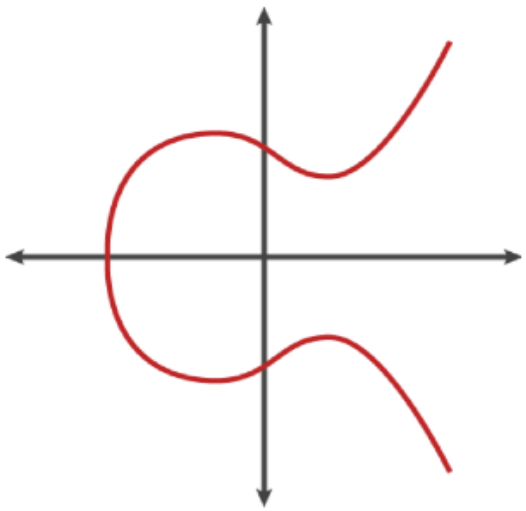


Figure 3.16 Elliptic Curve

- Multiplying a point on the curve by a number will produce another point on the curve, but it is very difficult to find what number was used, even if you know the original point and the result.

- The Equations based on elliptic curves have a characteristic that is very valuable for cryptography purposes: they are relatively easy to perform, and extremely difficult to reverse.

- The addition operation in ECC is the counterpart of modular multiplication in RSA, and multiple addition is the counterpart of modular exponentiation. To form a cryptographic system using elliptic curves, we need to find a "hard problem" corresponding to factoring the product of two primes or taking the discrete logarithm.

- Consider the equation Q = kP where Q, P Ep (a, b) and k < p. It is relatively easy to calculate Q given k and P, but it is relatively hard to determine k given Q and P. This is called the discrete logarithm problem for elliptic curves.

- Consider the group E_{23} (9, 17). This is the group defined by the equation $y^2 \bmod 23 = (x^3 + 9x + 17) \bmod 23$. What is the discrete logarithm k of Q = (4, 5) to the base P = (16.5)? The brute-force method is to compute multiples of P until Q is found. Thus,

  P = (16, 5); 2P = (20, 20); 3P = (14, 14); 4P = (19, 20); 5P = (13, 10); 6P = (7, 3); 7P = (8, 7); 8P (12, 17); 9P = (4, 5).

  Because 9P = (4, 5) = Q, the discrete logarithm Q = (4, 5) to the base P = (16, 5) is k = 9. In a real application, k would be so large as to make the brute-force approach infeasible.

UNIT IV MESSAGE AUTHENTICATION AND INTEGRITY

UNIT IV MESSAGE AUTHENTICATION AND INTEGRITY Authentication requirement – Authentication function – MAC – Hash function – Security of hash function and MAC – SHA – Digital signature and authentication protocols – DSS- Entity Authentication: Biometrics, Passwords, Challenge Response protocols- Authentication applications - Kerberos, X.509

4. AUTHENTICATION REQUIREMENT

Communication across the network, the following attacks can be identified.

Disclosure – release of message contents to any person or process not possessing the appropriate cryptographic key.

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.

Masquerade – insertion of messages into the network from fraudulent source. This can be creation of message by the attacker using the authorized port.

Content modification – changes to the contents of a message, including insertion, deletion, transposition, and modification.

Sequence modification – any modification to a sequence of messages between parties, including insertion, deletion, and reordering.

Timing modification – delay or replay of messages.

- In a connection oriented application, an entire session or sequence of messages could be replay of some previous valid session, or individual messages in the sequence could be delayed or replayed.
- In a connectionless application, an individual message could be delayed or replayed.

Source repudiation – denial of transmission of message by source.

Destination repudiation – denial of receipt of message by destination.

AUTHENTICATION FUNCTION

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.

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

The types of function that may be used to produce an authenticator are grouped into three classes.

Message Encryption – the ciphertext of the entire message serves as its authenticator.

Message Authentication Code (MAC) – a public function of the message and a secret key that produces a fixed length value that serves as the authenticator.

Hash Function – a public function that maps a message of any length into a fixed-length hash value, which serves as the authenticator.

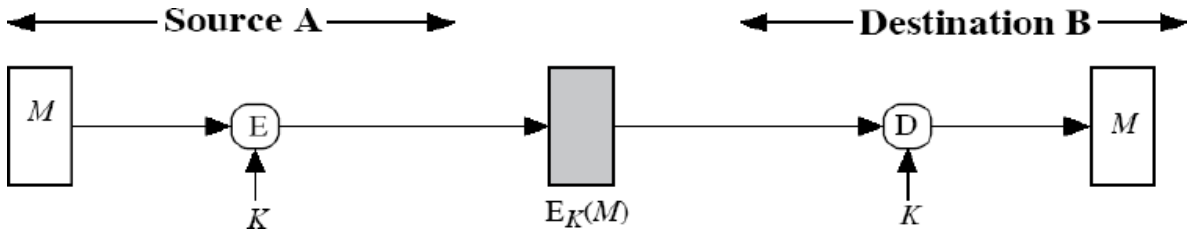
Message Encryption:

Message encryption Message encryption by itself can provide a measure of authentication.

The analysis differs from symmetric and public key encryption schemes.

(a) If symmetric encryption(fig.a) is used then:

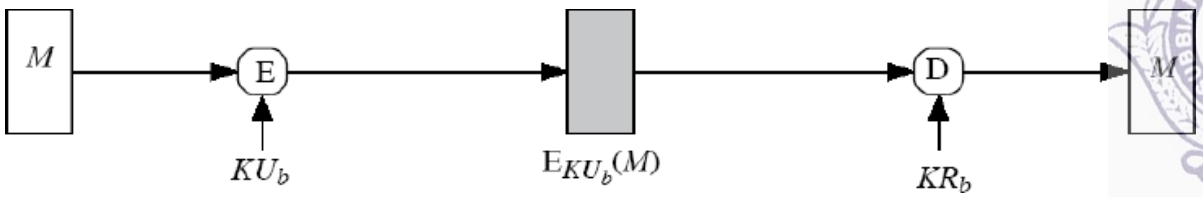
- A message m, transmitted from source A to destination B is encrypted using a secret key shared by A and B.
- Since only sender and receiver knows key used
- Receiver knows sender must have created it. Hence authentication is provided.
- Know content cannot have been altered. Hence confidentiality is also provided.
- If message has suitable structure, redundancy or a checksum to detect any changes
- Therefore Symmetric Encryption provides authentication and confidentiality.



(a).Symmetric key encryption confidentiality, authentication and signature

(b) If public-key encryption(Fig b) is used:

This method is the use of public key cryptography which provides confidentiality only. The sender A makes use of the public key of the receiver to encrypt the message. Here there is no authentication because any user can use B‟s public key to send a message and claim that only A has sent it.



(b) Public key encryption confidentiality

In this method(Fig c) to have only authentication, the message is encrypted with the sender‟s A‟s private key. The receiver B uses the sender‟s A‟s public key to decrypt the message. Now A cannot deny that it has not transmitted since it only knows its private key. This is called as authentication or Digital Signature. Hence the problem is the,

- Receiver cannot determine whether the packet decrypted contains some useful message or random bits.
- The problem is that anyone can decrypt the message when they know the public key of sender A.

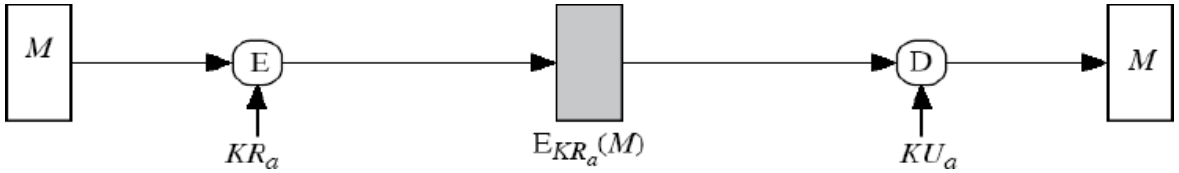


Figure (c) Public key encryption authentication and signature

This method (Fig d) provides authentication, confidentiality and digital signature. But the problem with this method is the complex public key cryptography algorithm should be applied twice during encryption and twice during decryption.

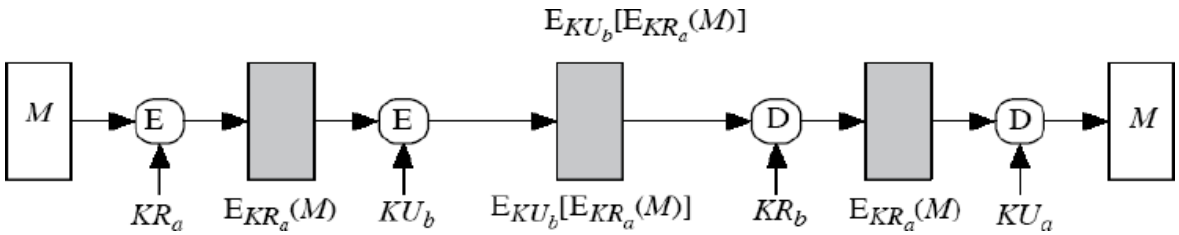


Figure (d) Public key encryption confidentiality, authentication and signature

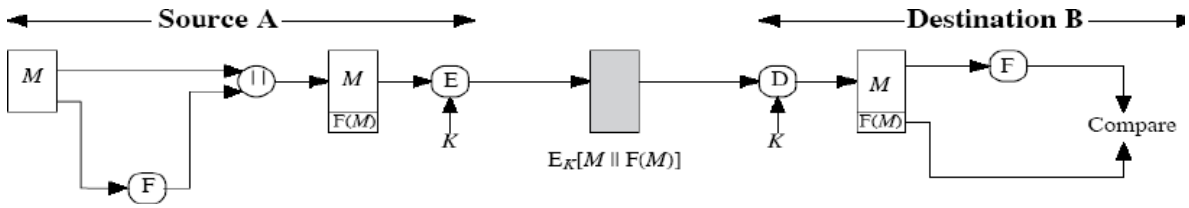
Suppose the message can be any arbitrary bit pattern, in that case, there is no way to determine automatically, at the destination whether an incoming message is the ciphertext of a legitimate message.

One solution to this problem is to force the plaintext to have some structure that is easily recognized but that cannot be replicated without recourse to the encryption function.

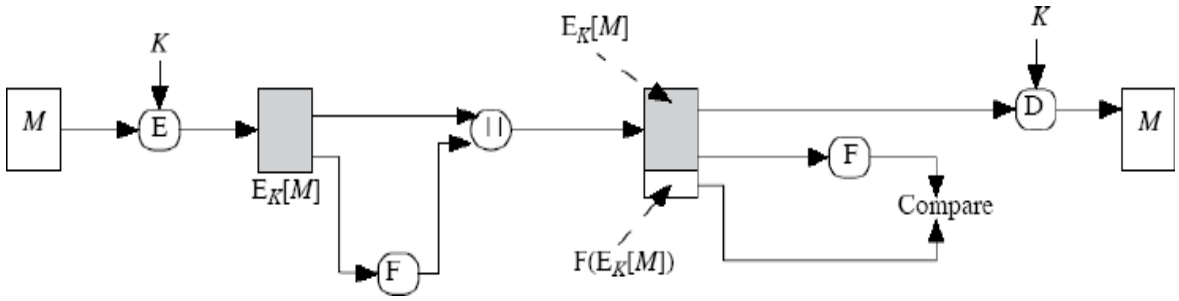
Append an error detecting code, also known as Frame Check Sequence (FCS) or checksum to each message before encryption „A‟ prepares a plaintext message M and then provides this as input to a function F that produces an FCS. The FCS is appended to M and the entire block is then encrypted.

At the destination, B decrypts the incoming block and treats the result as a message with an appended FCS. B applies the same function F to attempt to reproduce the FCS.

If the calculated FCS is equal to the incoming FCS, then the message is considered authentic. *In the internal error control, the function F is applied to the plaintext, whereas in external error control, F is applied to the ciphertext (encrypted message fig e and d).*



(e) Internal error control



(f) External error control

4.2.MAC

An alternative authentication technique involves the use of secret key to generate a small fixed size block of data, known as cryptographic checksum or MAC that is appended to the message.

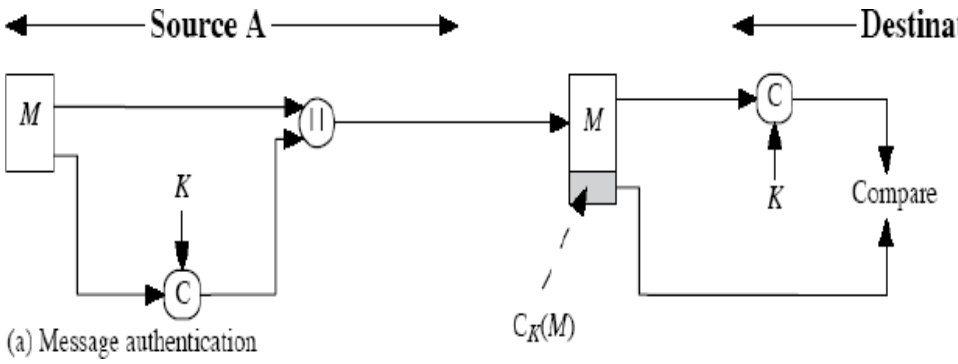
This technique assumes that two communication parties say A and B, share a common secret key „k". When A has to send a message to B, it calculates the MAC as a function of the message and the key.

$$MAC = C_K (M)$$

Where M – input message C – MAC function K – Shared secret key

The message plus MAC are transmitted to the intended recipient. The recipient performs the same calculation on the received message, using the shared secret key, to generate a new MAC.

The received MAC is compared to the calculated MAC. If it is equal, then the message is considered authentic(Fig g and h). A MAC function is similar to encryption. One difference is that MAC algorithm need not be reversible, as it must for decryption. In general, the MAC function is a many-to-one function.



(g) Message Authentication

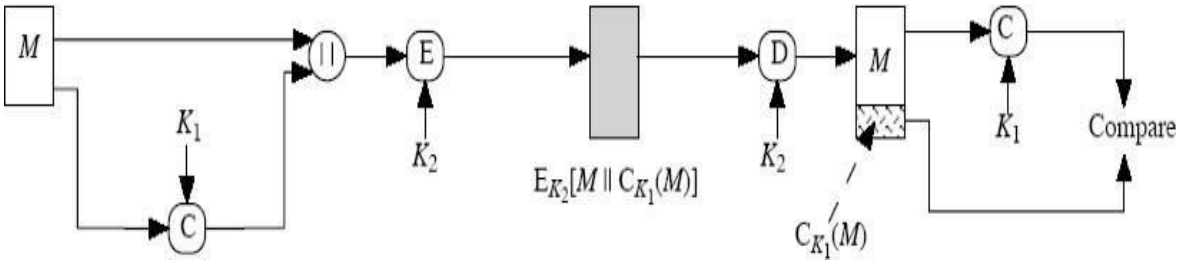


Figure (h) Message authentication and confidentiality, authentication tied to plain text

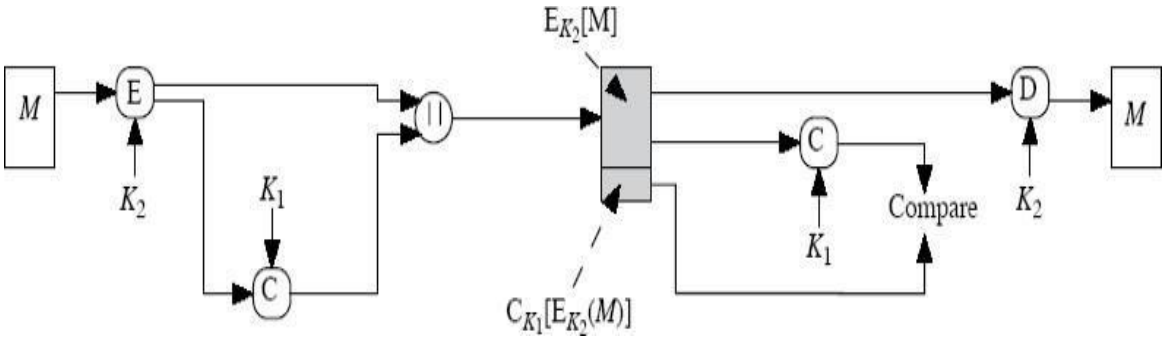


Figure (i) Message authentication and confidentiality, authentication tied to ciphertext

Requirements for MAC:

When an entire message is encrypted for confidentiality, using either symmetric or asymmetric encryption, the security of the scheme generally depends on the bit length of the key.

Barring some weakness in the algorithm, the opponent must resort to a brute-force attack using all possible keys. On average, such an attack will require 2(k-1) attempts for a k-bit key.

If confidentiality is not employed, the opponent has access to plaintext messages and their associated MACs. Suppose k > n; that is, suppose that the key size is greater than the

MAC size. Then, given a known M_1 and MAC_1, with $MAC_1 = C_K(M_1)$, the cryptanalyst can perform $MAC_i = CK_i(M1)$ for all possible key values Ki.

At least one key is guaranteed to produce a match of $MAC_i = MAC_1$.

Note that a total of 2^k MACs will be produced, but there are only $2^n < 2^k$ different MAC values. Thus, a number of keys will produce the correct MAC and the opponent has no way of knowing the correct key. On average, a total of $2^k/2^n = 2^{(k-n)}$ keys will produce a match. Thus, the opponent must iterate the attack:

Round 1

Given: M_1, $MAC_1 = C_K(M_1)$
Compute $MAC_i = C_{Ki}(M_1)$ for all 2^k keys
Number of matches $\approx 2^{(k-n)}$

Round 2

Given: M_2, $MAC_2 = C_K(M_2)$
Compute $MAC_i = C_{Ki}(M_2)$ for the $2^{(k-n)}$ keys resulting from Round 1
Number of matches $\approx 2^{(k-2xn)}$ and so on

Consider the following MAC algorithm. Let $M = (X_1||X_2||...||X_m)$ be a message that is treated as a concatenation of 64-bit blocks X_i. Then define

$$\Delta(M) = X_1 + X_2 \ldots X_m$$
$$C_k(M) = E_k(\Delta(M))$$

Thus, the key length is 56 bits and the MAC length is 64 bits. If an opponent observes {M||C(K, M)}, a brute-force attempt to determine K will require at least 2^{56} encryptions.

But the opponent can attack the system by replacing X1 through X_{m-1} with any desired values Y_1 through Y_{m-1} and replacing X_m with Y_m where Y_m is calculated as follows:

$$Y_m = Y_1 + Y_2 \ldots\ldots Y_{m1} + \Delta(M)$$

The opponent can now concatenate the new message, which consists of Y_1 through Y_m, with the original MAC to form a message that will be accepted as authentic by the receiver. With this tactic, any message of length 64 $X_{(m-1)}$ bits can be fraudulently inserted.

Then the MAC function should satisfy the following requirements: The MAC function should have the following properties:

- If an opponent observes M and $C_K(M)$, it should be computationally infeasible for the opponent to construct a message M‟ such that $C_K(M') = C_K(M)$
- $C_K(M)$ should be uniformly distributed in the sense that for randomly chosen messages, M and M‟, the probability that $C_K(M) = C_K(M')$ is 2^{-n} where n is the number of bits in the MAC.
- Let M‟ be equal to some known transformation on M. i.e., M‟ = f(M).

MAC based on DES

One of the most widely used MACs, referred to as Data Authentication Algorithm (DAA) is based on DES.

The algorithm(Fig 2) can be defined as using cipher block chaining (CBC) mode of operation of DES with an initialization vector of zero. The data to be authenticated are grouped into contiguous 64-bit blocks: D_1, D_2 ... D_n. if necessary, the final block is padded on the right with zeros to form a full 64-bit block. Using the DES encryption algorithm and a secret key, a data authentication code (DAC) is calculated as follows:

$O_1 = E_K(D_1)$
$O_2 = E_K(D_2 + O_1)$
$O_3 = E_K(D_3 + O_2)$

$O_N = E_K(D_N + O_{N-1})$

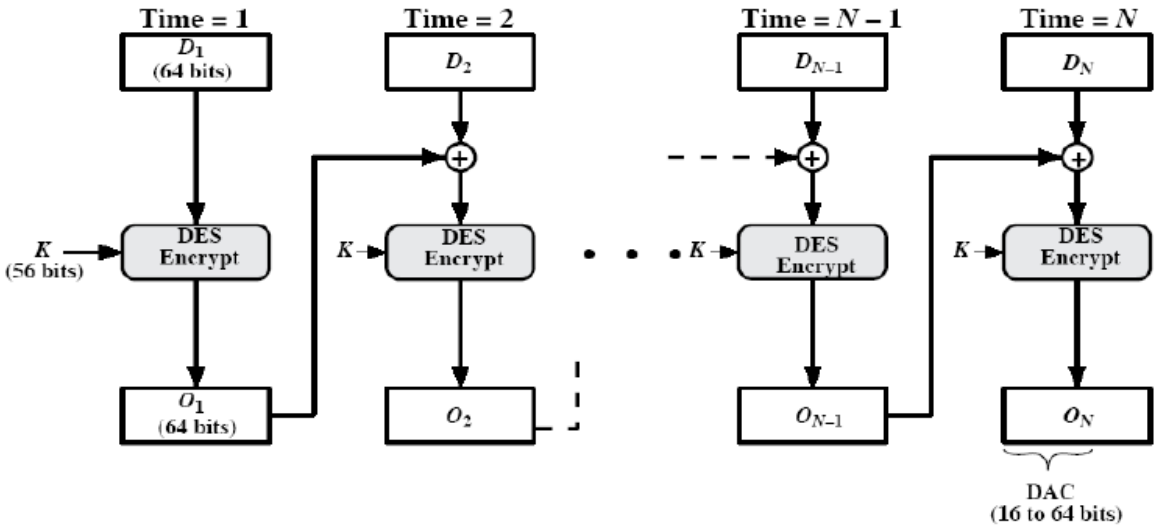


Figure.2 Data Authentication Algorithm

4.3.HASH FUNCTION

A variation on the message authentication code is the one way hash function. As with MAC, a hash function accepts a variable size message M as input and produces a fixed-size output, referred to as hash code H(M).

Unlike a MAC, a hash code does not use a key but is a function only of the input message. The hash code is also referred to as a message digest or hash value. There are varieties of ways in which a hash code can be used to provide message authentication, as follows:

In Fig (a) The message plus the hash code is encrypted using symmetric encryption. This is identical to that of internal error control strategy. Because encryption is applied to the entire message plus the hash code, confidentiality is also provided.

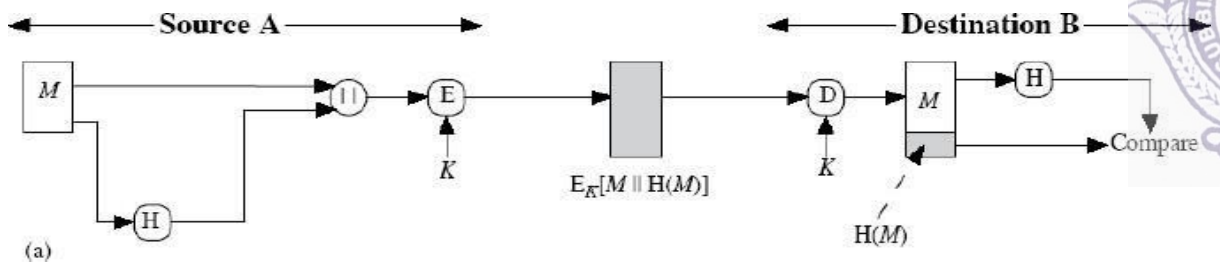
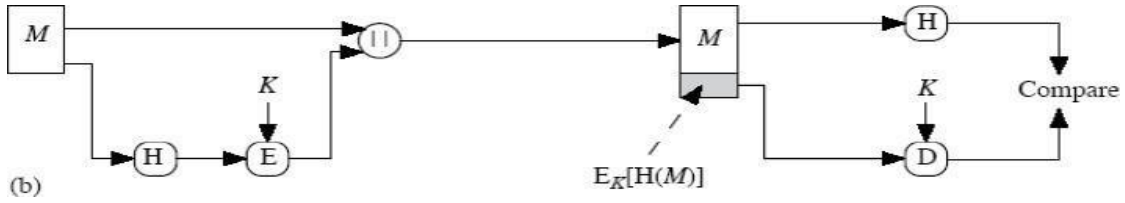


Figure (a) Hash Function

In Fig (b) Only the hash code is encrypted, using symmetric encryption. This reduces the processing burden for those applications that do not require confidentiality.



In Fig (c) Only the hash code is encrypted, using the public key encryption and using the sender"s private key. It provides authentication plus the digital signature.

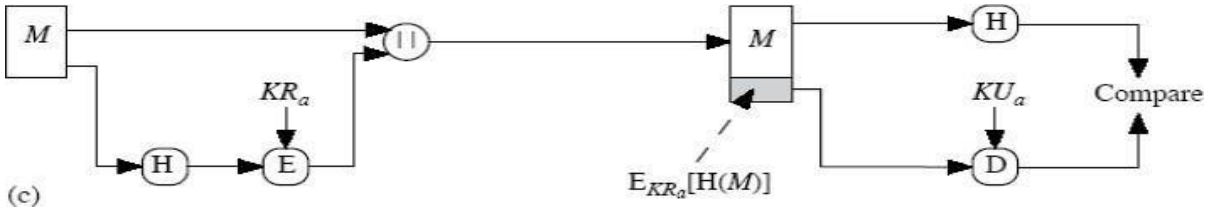
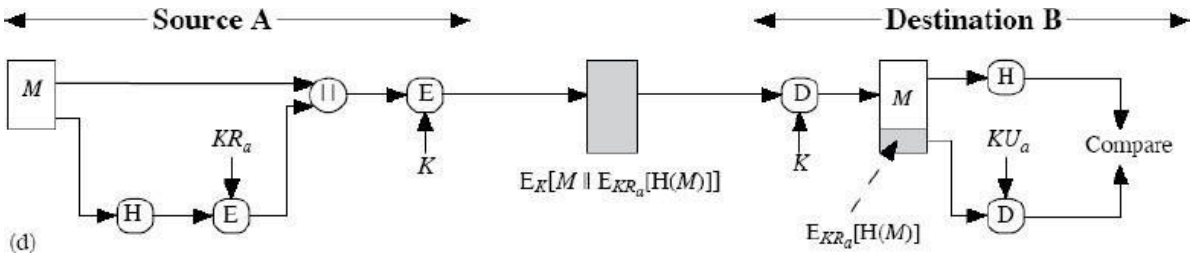
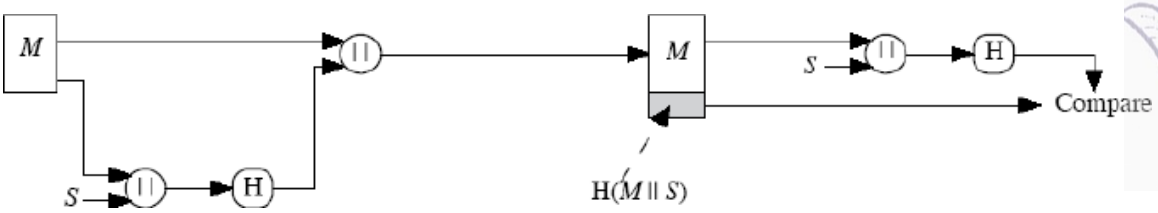


Figure (b & c) Basic use of Hash Function

In Fig (d) If confidentiality as well as digital signature is desired, then the message plus the public key encrypted hash code can be encrypted using a symmetric secret key.



In Fig (e) This technique uses a hash function, but no encryption for message authentication. This technique assumes that the two communicating parties share a common secret value „S". The source computes the hash value over the concatenation of M and S and appends the resulting hash value to M.



In Fig(f) Confidentiality can be added to the previous approach by encrypting the entire message plus the hash code.

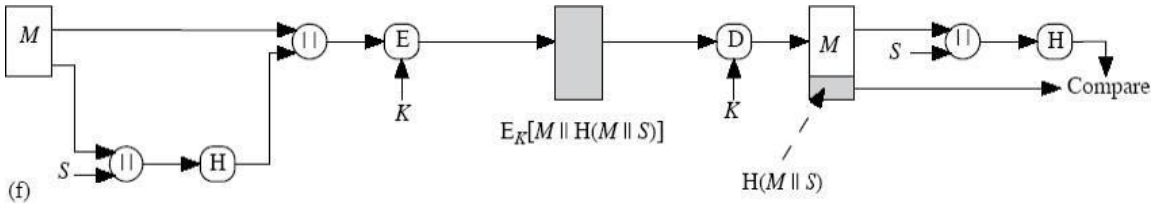


Figure (d,e & f) Basic use of Hash Function

A hash value h is generated by a function H of the form

$$h = H(M)$$

where M is a variable-length message and H(M) is the fixed-length hash value.

The hash value is appended to the message at the source at a time when the message is assumed or known to be correct. The receiver authenticates that message by recomputing the hash value.

Requirements for a Hash Function

1. H can be applied to a block of data of any size.

2. H produces a fixed-length output.

3. H(x) is relatively easy to compute for any given x, making both hardware and software implementations practical.

4. For any given value h, it is computationally infeasible to find x such that H(x) = h. This is sometimes referred to in the literature as the one-way property.

5. For any given block x, it is computationally infeasible to find y x such that H(y) = H(x). This is sometimes referred to as weak collision resistance.

6. It is computationally infeasible to find any pair (x, y) such that H(x) = H(y). This is sometimes referred to as strong collision resistance.
The first three properties are requirements for the practical application of a hash function to message authentication.

The fourth property, the one-way property, states that it is easy to generate a code given a message but virtually impossible to generate a message given a code.

The fifth property guarantees that an alternative message hashing to the same value as a given message cannot be found. This prevents forgery when an encrypted hash code is used.

The sixth property refers to how resistant the hash function is to a type of attack known as the birthday attack.

Simple Hash Functions

All hash functions operate using the following general principles. The input (message, file, etc.) is viewed as a sequence of n-bit blocks. The input is processed one block at a time in an iterative fashion to produce an n-bit hash function. One of the simplest hash functions is the bit-by-bit exclusive-OR (XOR) of every block.

This can be expressed as follows: $C_i = b_{i1} + b_{i1} \ldots + b_{im}$

where

$C_i = i^{th}$ bit of the hash code, $1 \leq i \leq n$
m = number of n-bit blocks in the input
b_{ij}= ith bit in jth block

Procedure:

1. Initially set the n-bit hash value to zero.

2. Process each successive n-bit block of data as follows:

a. Rotate the current hash value to the left by one bit.
b. XOR the block into the hash value.

Birthday Attacks

Suppose that a 64-bit hash code is used. One might think that this is quite secure. For example, if an encrypted hash code C is transmitted with the corresponding unencrypted message M, then an opponent would need to find an M' such that H(M') = H(M) to substitute another message and fool the receiver.

On average, the opponent would have to try about 2^{63} messages to find one that matches the hash code of the intercepted message.

However, a different sort of attack is possible, based on the birthday paradox. The source, A, is prepared to "sign" a message by appending the appropriate m-bit hash code and encrypting that hash code with A's private key.

1. The opponent generates $2^{m/2}$ variations on the message, all of which convey essentially the same meaning. (Fraudulent message)

2. The two sets of messages are compared to find a pair of messages that produces the same hash code. The probability of success, by the birthday paradox, is greater than 0.5. If no match is found, additional valid and fraudulent messages are generated until a match is made.

3. The opponent offers the valid variation to A for signature. This signature can then be attached to the fraudulent variation for transmission to the intended recipient. Because the two variations have the same hash code, they will produce the same signature; the opponent is assured of success even though the encryption key is not known.

Thus, if a 64-bit hash code is used, the level of effort required is only on the order of 2^{32}

MEET-IN-THE-MIDDLE ATTACK.

Divide a message M into fixed-size blocks $M_1, M_2, ..., M_N$ and use a symmetric encryption system such as DES to compute the hash code G as follows:

$$H_o = \text{initial value}$$
$$H_i = E_{Mi} [H_{i-1}]$$
$$G = H_N$$

This is similar to the CBC technique, but in this case there is no secret key. As with any hash code, this scheme is subject to the birthday attack, and if the encryption algorithm is DES and only a 64-bit hash code is produced, then the system is vulnerable.

Furthermore, another version of the birthday attack can be used even if the opponent has access to only one message and its valid signature and cannot obtain multiple signings.

Here is the scenario; we assume that the opponent intercepts a message with a signature in the form of an encrypted hash code and that the unencrypted hash code is m bits long:

1. Calculate the unencrypted hash code G.

2. Construct any desired message in the form $Q_1, Q_2, ..., Q_{N2}$.

3. Compute for $H_i = EQ_i [H_{i-1}]$ for $1 \leq i \leq (N-2)$.

4. Generate $2^{m/2}$ random blocks; for each block X, compute $E_X[H_{N-2}.]$ Generate an additional $2^{m/2}$ random blocks; for each block Y, compute $D_Y[G]$, where D is the decryption function corresponding to E.

5. Based on the birthday paradox, with high probability there will be an X and Y such that $E_X [H_{N-2}] = D_Y[G]$.

6. Form the message $Q_1, Q_2, ..., Q_{N-2}$, X, Y. This message has the hash code G and therefore can be used with the intercepted encrypted signature.

SECURITY OF HASH FUNCTION AND MAC

Just as with symmetric and public-key encryption, we can group attacks on hash functions and MACs into two categories: 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 solely on the length of the hash code produced by the algorithm.

Requirements of Hash Function:

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	$2^{n/2}$

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.

To proceed, we need to state the desired security property of a MAC algorithm, which can be expressed as follows:

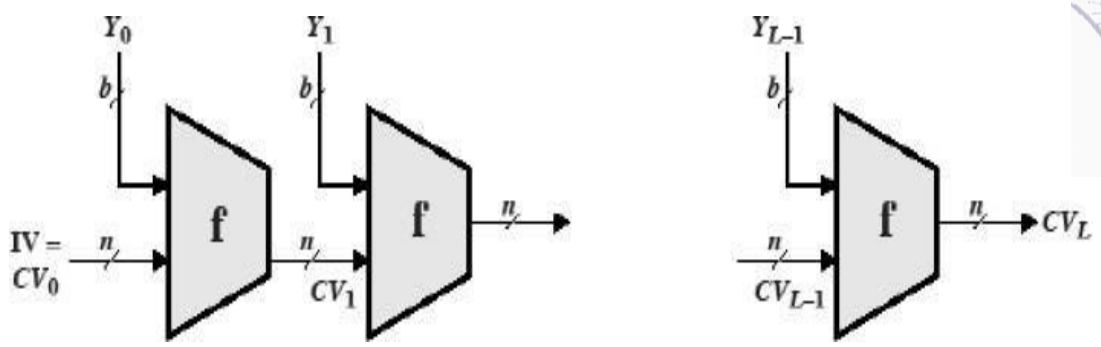
Cryptanalysis

As with encryption algorithms, cryptanalytic attacks on hash functions and MAC algorithms seek to exploit some property of the algorithm to perform some attack other than an exhaustive search.

Hash Functions

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(Fig 3.4). The inclusion of the length makes the job of the opponent more difficult.

Either the opponent must find two messages of equal length that hash to the same value or two messages of differing lengths that, together with their length values, hash to the same value.



IV=Initial Value
Y_i = ith input block
n=Length of Hash code

CV=Changing Variable
L=number of input blocks
b=Length of input block

General structure of secure hash code

The hash algorithm involves repeated use of a compression function,f, that takes two inputs (an n-bit input from the previous step, called the chaining variable, and a b-bit block) and produces an n-bit output.

At the start of hashing, the chaining variable has an initial value that is specified as part of the algorithm. The final value of the chaining variable is the hash value. Often b > n; hence the term compression.

The hash function can be summarized as follows:

CV_0 = IV = initial n-bit value
$CV_I = f(CV_{i-1}, Yi_{-1})$ $1 \leq i \leq L$
$H(M) = CV_L$
Where the input to the hash function is a message M consisting of the blocks Y_o, Y_1,..., Y_{L-1}. The structure can be used to produce a secure hash function to operate on a message of any length.

Message Authentication Codes :

There is much more variety in the structure of MACs than in hash functions, so it is difficult to generalize about the cryptanalysis of MACs.

SHA

The algorithm takes as input a message with a maximum length of less than bits and produces as output a 512-bit message digest. The input is processed in 1024-bit blocks. Figure **3.1** depicts the overall processing of a message to produce a digest.

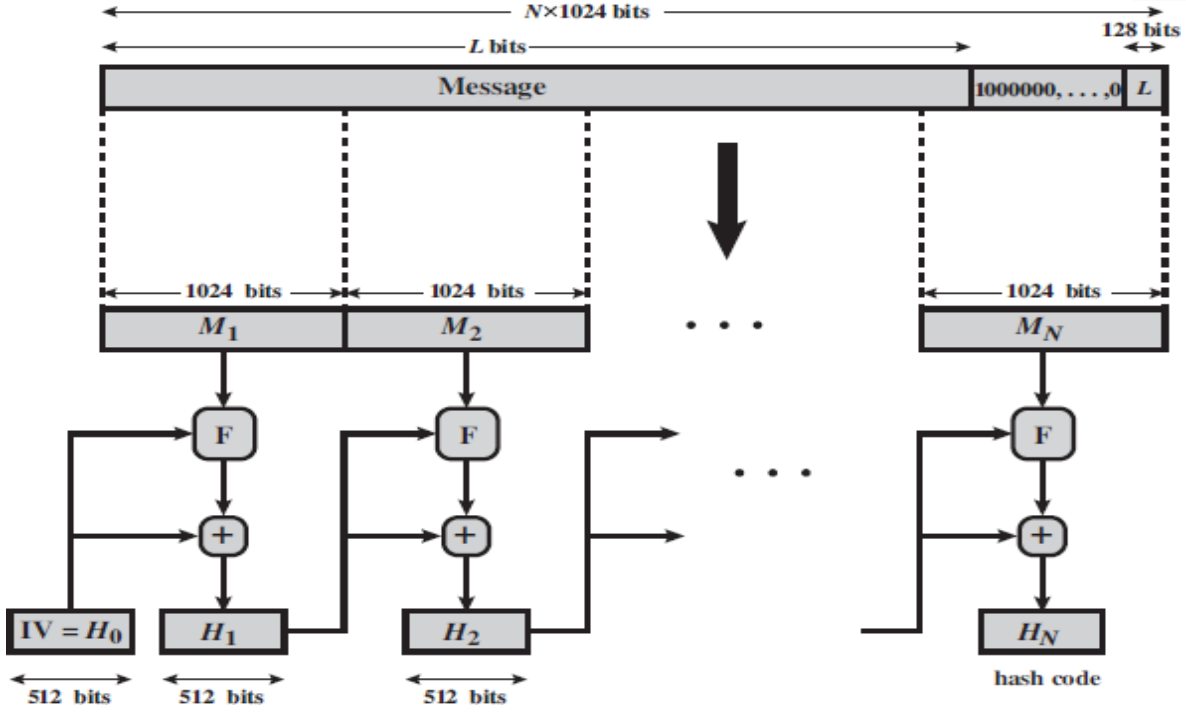


Fig .Message Digest Generation Using SHA-512

The processing consists of the following steps.

Step 1: Append padding bits.

The message is padded so that its length is congruent to 896 modulo 1024. 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 1024. The padding consists of a single 1 bit followed by the necessary number of 0 bits.

Step 2: Append length.

A block of 128 bits is appended to the message. This block is treated as an unsigned 128-bit integer (most significant byte first) and contains the length of the original message (before the padding).

The outcome of the first two steps yields a message that is an integer multiple of 1024 bits in length. In Figure 3,8 , the expanded message is represented as the sequence of 1024-bit blocks M_1, M_2, ... M_N , so that the total length of the expanded message is NX1024 bits.

Step 3: Initialize hash buffer.

A 512-bit buffer is used to hold intermediate and final results of the hash function.The buffer can be represented as eight 64-bit registers (a, b, c, d, e, f, g, h).These registers are initialized to the following 64-bit integers (hexadecimal values):

a = 6A09E667F3BCC908 e = 510E527FADE682D1
b = BB67AE8584CAA73B f = 9B05688C2B3E6C1F
c = 3C6EF372FE94F82B g = 1F83D9ABFB41BD6B
d = A54FF53A5F1D36F1 h = 5BE0CD19137E2179

These values are stored in big-endian format, which is the most significant byte of a word in the low-address (leftmost) byte position. These words were obtained by taking the first sixty-four bits of the fractional parts of the square roots of the first eight prime numbers.

Step 4: Process message in 1024-bit (128-word) blocks.

The heart of the algorithm (Fig 3.9) is a module that consists of 80 rounds; Each round takes as input the 512-bit buffer value, abcdefgh, and updates the contents of the buffer. At input to the first round, the buffer has the value of the intermediate hash value, H_{i-1}

Each round makes use of a 64-bit value W_t, derived from the current 1024-bit block (M_i) being processed .These values are derived using a message schedule described subsequently.

Each round also makes use of an additive constant k_t, where 0<=t<=79 indicates one of the 80 rounds.

The output of the eightieth round is added to the input to the first round (H_i-1) to produce H_i.The addition is done independently for each of the eight words in the buffer with each of the corresponding words in Hi-1, using addition modulo 264 .

Step 5: Output.

After all N 1024-bit blocks have been processed, the output from the Nth stage is the 512-bit message digest.

The behavior of SHA-1 is summarized as follows:

$$H_0 = IV$$

$$H_i = SUM_{64}(H_{i-1}, ABCDEFGH_i)$$

$$MD = H_N$$

Where

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

$ABCDE_q$ = the output of the last round of processing of the *i*th 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

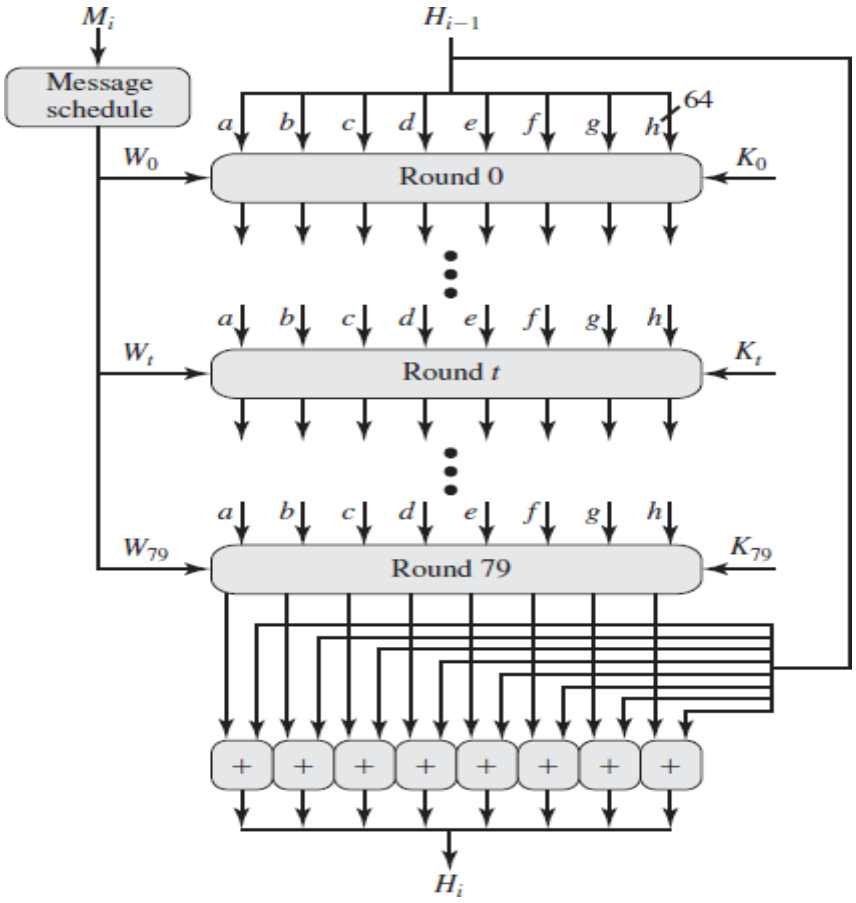


Figure. SHA-512 Processing of a Single 1024-Bit Block

SHA-512 Round Function

Let us look in more detail at the logic in each of the 80 steps of the processing of one 512-bit block. Each round is defined by the following set of equations:

$T_1 = h + Ch(e,f,g) + (\Sigma_1^{512} e) + Wt + Kt)$

$T_2 = (\Sigma_0^{512} a) + Maj(a,b,c)$

$h = g$ $g = f$ $f = e$ $e = d + T_1$ $d = c$
$c = b$ $b = a$ $a = T_1 + T_2$

Where

T = Step number; $0 \le t \le 79$

$Ch(e,f,g) = (a \text{ AND } f) \oplus (\text{NOT } e \text{ AND } g)$

The conditional function: If e then f else g (Fig 3.10)

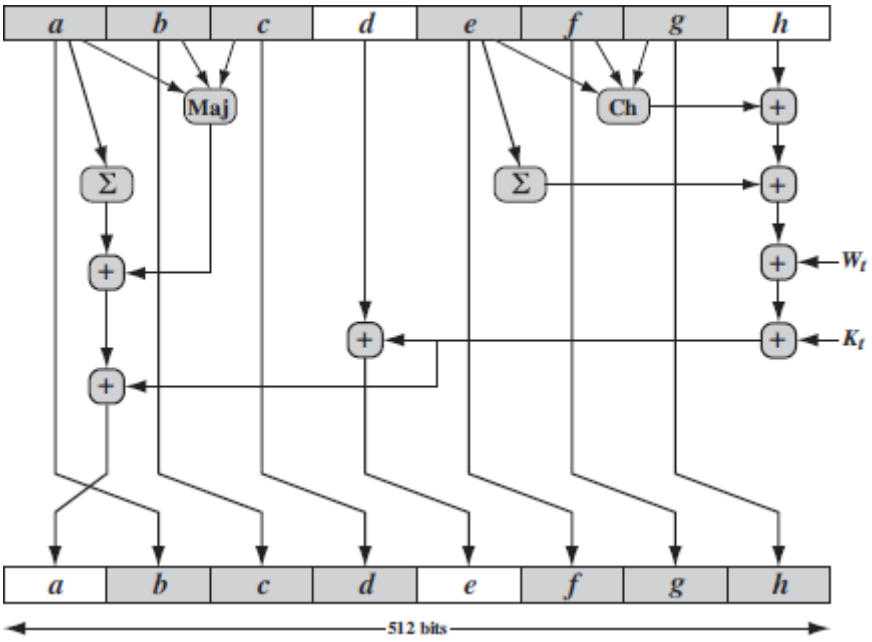


Fig .Elementary SHA Operation(single step)

The functions can be summarized as follows:

Steps	Function Name	Function Value
0≤t≤9	f1 = f(t,B,C,D)	(B Λ C) V (B! Λ D)
20≤t≤39	f2 = f(t,B,C,D)	B ⊕C ⊕D
40≤t≤59	f3 = f(t,B,C,D)	(B Λ C) V (B Λ D) V (C Λ D)
60≤t≤79	f4 = f(t,B,C,D)	B ⊕C ⊕D

The logical operators AND,OR,NOT,XOR, are represented by the symbols Λ V ! ⊕

Only three different functions are used.

For, 0≤ t ≤ 19 the function is the conditional function.

For 20 ≤ t ≤ 39 and 60 ≤ t ≤ 79 the function produces a parity bit.

For A0 ≤ t ≤ 59 the function is true if two or three of the argument are true.

The following diagram illustrates how the 32bit word values wt are derived from the 512 bit message.

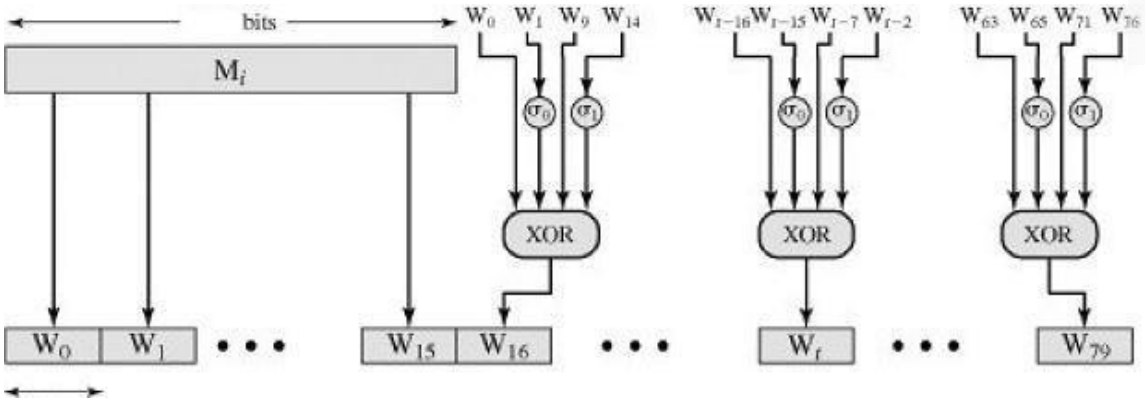


Figure.Creation of 80-word Input Sequence for SHA-512 Processing of Single Block

The first 16 values of wt are taken directly from the 16 words of the current block.the remaining values are defined as follows.

$$w_t = S^{\alpha}(w_{t-16} + w_{t-14} + w_{t-8} + w_{t-3})$$

Thus in the first 16 steps of processing the values of wt is equal to the corresponding word in the message block. For the remaining 64 steps the value of wt consists of the circular left shift by one bit of the XOR of four of the processing values of w_t.

Both MD5 and RIPEMD-160 uses one of the 16 words of a message block directly as input to each step function only the order of the word is permuted from round to round.

SHA-1 expands the 16 block words to 80 words for use in the compression function.

Comparison of SHA-1 and MD5

Because both are derived from MD4,SHA-1 and MD5 are similar to one another.

1. Security against brute – force attacks:

The most important difference is that the SHA-1 digest is 32bits longer than the MD5 digest.

Using a brute force technique the difficulty of producing any message having a given message digest is on the order of 2^{128} operations for MD5 and 2^{160} for SHA-1.

Using brute force technique the difficulty of producing two messages having the same message digest is on the order of 2^{64} operations for MD5 and 2^{80} for SHA-1.Thus SHA-1 is considerably stronger against brute force attacks.

2. Security against cryptanalysis:

MD5 is vulnerable to cryptanalytic attacks.

SHA-1 is not vulnerable to such attacks.

3. Speed:

Both algorithms rely on addition module 2^{32}, so both do well on 32 bit architecture

SHA-1 involves more steps (80) and must process a 160 bit buffer compared to MD5‟s 128 bit buffer.

Thus SHA-1 should execute more slowly than MD5 on the same hardware.

4. Simplicity and compactness:

Both algorithms are simple to describe and simple to implement and do not require large programs or substitution tables.

5. Little endian versus big endian architecture:

MD5 uses a little endian scheme and SHA-1 uses a big endian scheme.

HMAC

HMAC Design Objectives:

- To use hash functions that perform well in software and for which code is freely and widely available.
- To allow for easy replacement of the embedded hash function in case faster or more secure hash functions are found or required.
- To preserve the original performance of the hash function without incurring a significant degradation.
- To use and handle keys in a simple way.
- To have a well understood cryptographic analysis of the strength of the authentication mechanism based on reasonable assumptions about the embedded hash function.

The first two objectives are important to the acceptability of HMAC.

HMAC treats the hash function as a “black box.” This has two benefits.

First, an existing implementation of a hash function can be used as a module in implementing HMAC. In this way, the bulk of the HMAC code is prepackaged and ready to use without modification.

Second, if it is ever desired to replace a given hash function in an HMAC implementation, remove the existing hash function module and drop in the new module.

HMAC Algorithm:

Definition of terms used in algorithm(Fig 3.12).

H = embedded hash function (e.g., MD5, SHA-1, RIPEMD-160)

IV = initial value input to hash function

M = message input to HMAC

Y_i =*i* th block of *M*, $0 \le i \le (L - 1)$

L =number of blocks in *M*

b = number of bits in a block

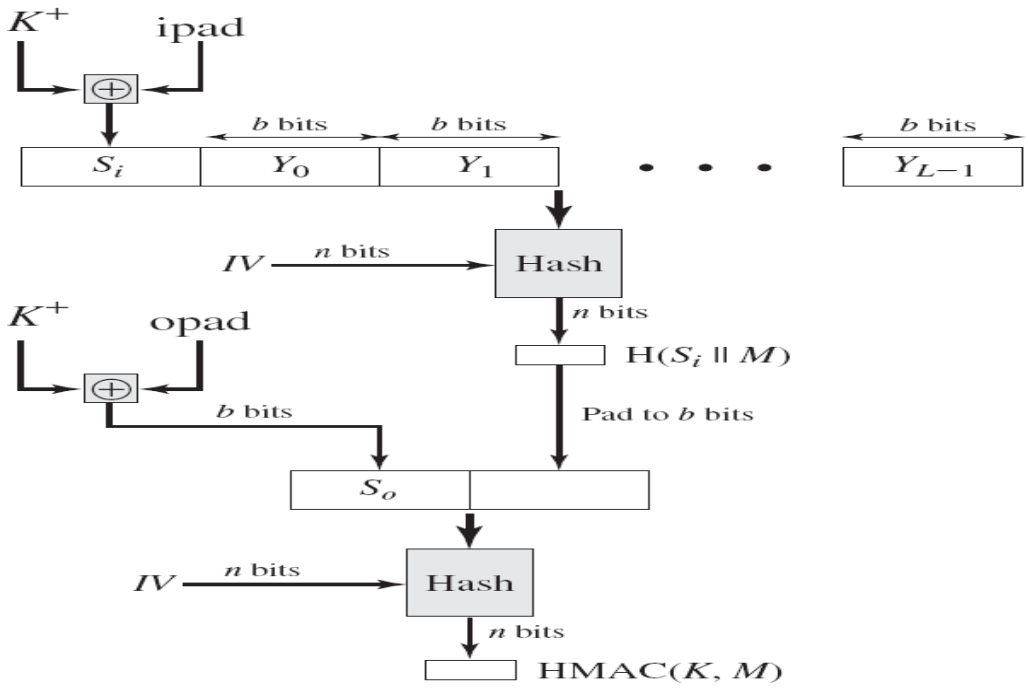


Figure .HMAC Structure

n = length of hash code produced by embedded hash function

K = secret key; recommended length is *n*; if key length is greater than *b*, the key is input to the hash function to produce an *n*-bit key

K^+ = *K* padded with zeros on the left so that the result is *b* bits in length

ipad = 00110110 (36 in hexadecimal) repeated *b*/8 times

opad = 01011100 (5C in hexadecimal) repeated *b*/8 times

Then HMAC can be expressed as

$$\text{HMAC}(K, M) = \text{H}[(K^+ \oplus \text{opad}) \,||\, \text{H}[(K^+ \oplus \text{ipad}) \,||\, M]]$$

We can describe the algorithm as follows:

1. Append zeros to the left end of to create a -bit string (e.g., if is of length 160 bits and , then will be appended with 44 zeroes).

2. XOR (bitwise exclusive-OR) with ipad to produce the b -bit block S_i .

3. Append M to S_i.

4. Apply H to the stream generated in step 3.

5. XOR K^+ with opad to produce the b-bit block S_0.

6. Append the hash result from step 4 to S_0.

7. Apply H to the stream generated in step 6 and output the result.

A more efficient implementation is possible. Two quantities are precomputed:

$$f(IV, (K^+ \oplus \text{ipad}))$$

$$f(IV, (K^+ \oplus \text{opad}))$$

In effect, the precomputed quantities substitute for the initial value (IV) in the hash function. With this implementation (Fig 3.13), only one additional instance of the compression function is added to the processing normally produced by the hash function.

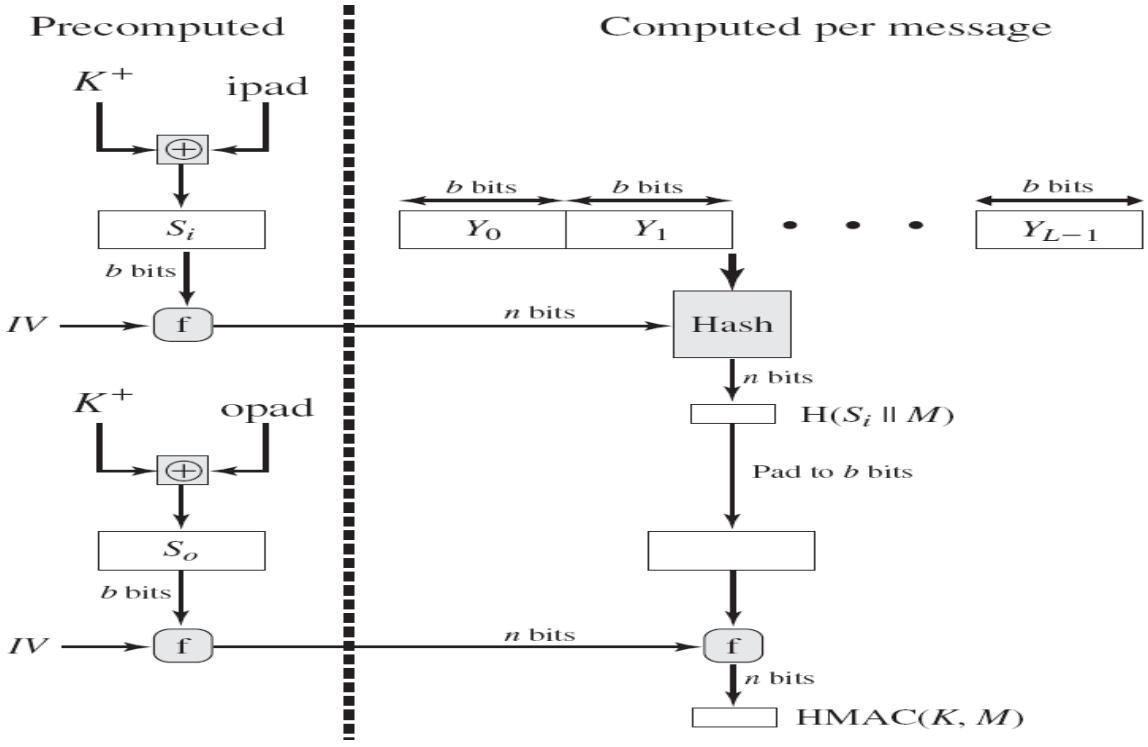
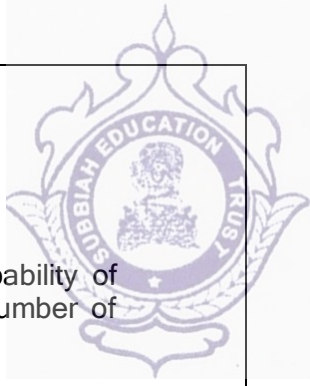


Figure .Efficient Implementation of HMAC



Security of HMAC

The security of a MAC function is generally expressed in terms of the probability of successful forgery with a given amount of time spent by the forger and a given number of message-tag pairs created with the same key.

In essence, it is proved in that for a given level of effort (time, message-tag pairs) on messages generated by a legitimate user and seen by the attacker, the probability of successful attack on HMAC is equivalent to one of the following attacks on the embedded hash function.

1. The attacker is able to compute an output of the compression function even with an that is random, secret, and unknown to the attacker.

2. The attacker finds collisions in the hash function even when theIV is random and secret.

In the first attack, we can view the compression function as equivalent to the hash function applied to a message consisting of a single b bit block. For this attack, the IV of the hash function is replaced by a secret, random value of bits. An attack on this hash function requires either a brute-force attack on the key, which is a level of effort on the order of 2^n, or a birthday attack.

In the second attack, the attacker is looking for two messages M & M‟and that produce the same hash: H(*M*) = H(*M*‟) .

CMAC

Only messages of one fixed length of *mn* bits are processed, where *n* is the cipher block size and *m* is a fixed positive integer. a simple example, notice that given the CBC MAC of a one-block message *X*, say *T* = MAC(*K*, *X*), the adversary immediately knows the CBC MAC for the two block message $X \| (X \oplus T)$since this is once again *T*.

Black and Rogaway [BLAC00] demonstrated that this limitation could be overcome using three keys: one key *K* of length *k* to be used at each step of the cipher block chaining and two keys of length *b*, where *b* is the cipher block length.

The **Cipher-based Message Authentication Code** (CMAC) mode of operation for use with AES and triple DES. It is specified in NIST Special Publication 800-38B.

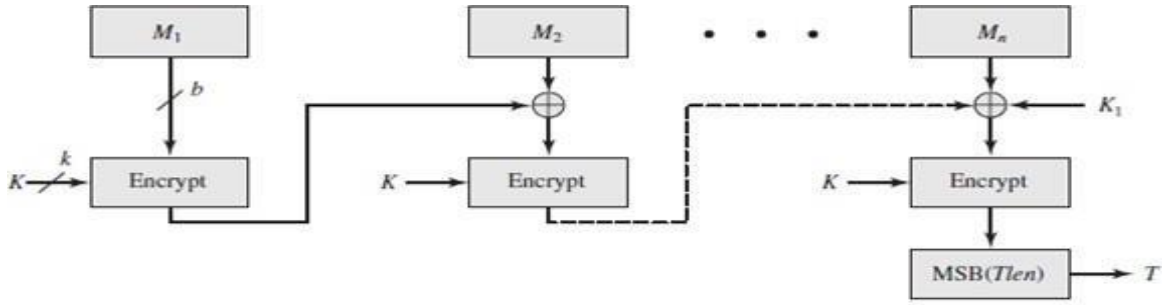
First, let us define the operation of CMAC when the message is an integer multiple *n* of the cipher block length *b*. For AES, *b* = 128, and for triple DES, *b* = 64. The message is divided into *n* blocks (*M*1, *M*2,c, *Mn*). The algorithm makes use of a *k*-bit encryption key *K* and a *b*-bit constant, *K*1. For AES, the key size *k* is 128, 192, or 256 bits; for triple DES, the key size is 112 or 168 bits. CMAC is calculated as follows

$$
\begin{aligned}
C_1 &= \mathrm{E}(K, M_1) \\
C_2 &= \mathrm{E}(K, [M_2 \oplus C_1]) \\
C_3 &= \mathrm{E}(K, [M_3 \oplus C_2]) \\
&\cdot \\
&\cdot \\
&\cdot \\
C_n &= \mathrm{E}(K, [M_n \oplus C_{n-1} \oplus K_1]) \\
T &= \mathrm{MSB}_{Tlen}(C_n)
\end{aligned}
$$

where

T = message authentication code, also referred to as the tag
Tlen = bit length of T
MSB*s*(*X*) = the *s* leftmost bits of the bit string *X*

The CMAC operation(Fig 3.14) then proceeds as before, except that a different *b*-bit key *K*2 is used instead of *K*1.



(a) Message length is integer multiple of block size

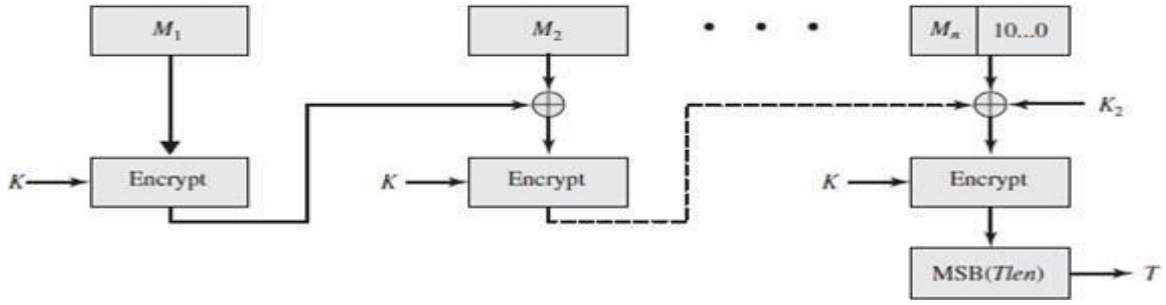


Fig .Cipher-based Message Authentication Code

The two *b*-bit keys are derived from the *k*-bit encryption key as follows.

$$L = \mathrm{E}(K, 0^b)$$
$$K_1 = L \cdot x$$
$$K_2 = L \cdot x^2 = (L \cdot x) \cdot x$$

where multiplication (#) is done in the finite field GF(2*b*) and *x* and *x*2 are first and second-order polynomials that are elements of GF(2*b*). Thus, the binary representation of *x* consists of *b* - 2 zeros followed by 10; the binary representation of *x*2 consists of *b* - 3 zeros followed by 100.

DIGITAL SIGNATURE AND AUTHENTICATION PROTOCOLS

Digital Signature Requirements

Message authentication protects two parties who exchange messages from any third party. However, it does not protect the two parties against each other.

Disputes created by message authentication are:

- Creation of fraud message.

- Deny the sending of message

For example, suppose that John sends an authenticated message to Mary, the following disputes that could arise:

1 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.

2. 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.

Properties of digital signature :

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

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 recognize 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.

Direct Digital Signature

The term **direct digital signature** refers to a digital signature scheme that involves only the communicating parties (source, destination). It is assumed that the destination knows the public key of the source.

Confidentiality can be provided by encrypting the entire message plus signature with a shared secret key (symmetric encryption). Note that it is important to perform the signature function first and then an outer confidentiality function. In case of dispute, some third party must view the message and its signature.

If the signature is calculated on an encrypted message, then the third party also needs access to the decryption key to read the original message. However, if the signature is the inner operation, then the recipient can store the plaintext message and its signature for later use in dispute resolution.

The validity of the scheme just described depends on the security of the sender‟s private key.

Weakness of Direct Digital Signature:

- If a sender later wishes to deny sending a particular message, the sender can claim that the private key was lost or stolen and that someone else forged his or her signature.
- Another threat is that some private key might actually be stolen from X at time T. The opponent can then send a message signed with X"s signature and stamped with a time before or equal to T.

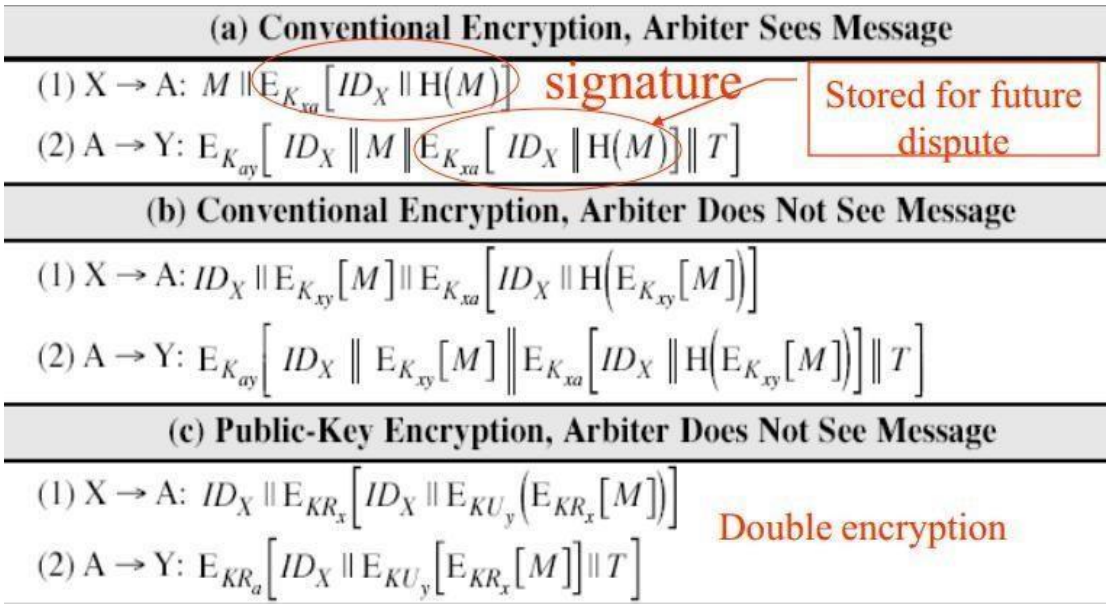
Arbitrated Digital Signatures

The problem associated with the Direct digital signature can be overcome by using arbitrated schemes.

In the arbitrated scheme, the entire signed message from the sender goes to the arbiter A. The arbiter subjects the message and signature to a number of tests to check the origin and control. The date and time is attached to the message. This indicates that the digital signature has been verified and is satisfied. The message is then transmitted to the receiver.

Requirement of the arbiter:

- As the arbiter plays a sensitive and crucial role, it should be a trusted third party.



Notation:X = Sender Y = Recipient A = Arbiter M=Message T=Timestamp

Scheme 1: Conventional encryption, Arbiter sees the message:

The sender X and arbiter A share the master key K_{ax} the receiver y and the arbiter A share the master key K_{ay}

When X wants to send a message M to Y, construct a message computes the hash value H(M). This hash is encrypted using symmetric encryption with the key Kax which acts as signature. The message along with the signature is transmitted to A.

At A, it decrypts the signature and checks the hash value to validate the message. A transmit the message to Y, encrypted with K_{ay}. Y decrypt to extract the message and signature.

Disadvantage:

Eaves dropper can read the message as there is no confidentiality.

Scheme 2: Conventional encryption, Arbiter does not see the message:

- K_{ax} and K_{ay} are the master keys.
- K_{xy} is the key shared between the X and Y
- When x wants to transmit a message to Y, the packet goes to arbiter.
- The same procedure as that of I scheme is used X transmit an identifier, a copy of the message encrypted with K_{xy} and a signature to A.
- The signature is the hash of the message encrypted with K_{xa}
- A decrypt the signature, and checks the hash value to validate the message.
- A cannot read the message, A attaches to it the time stamps, encrypt with K_{xa} and transmit to Y.

Attack: The arbiter can join with an attacker and deny a message with sender‟s signature.

Scheme 2: Public key encryption, Arbiter does not see the message:

This method uses the public key cryptography which gives authentication and digital signature.

The doubly encrypted message is concatenated with ID_x and sent to arbiter.

- A can decrypt the outer encryption to ensure that the message has come from X.
- A then transmit the message with ID_x and time stamp.

Advantages:

- No information is shared among parties before communication, hence fraud is avoided.
- No incorrectly dated message can be sent.

Disadvantages:

The complex public key algorithm is to be twice for encryption and twice for decryption.

Authentication Protocols

Authentication Protocols used to convince parties of each other‟s identity and to exchange session keys.

Mutual Authentication

An important application area is that of mutual authentication protocols. Such protocols enable communicating parties to satisfy themselves mutually about each other‟s identity and to exchange session keys.

Key issues are

- **confidentiality** – to protect session keys and prevent masqueraded and compromised
- **timeliness** - to prevent replay attacks

Replay Attacks
Where a valid signed message is copied and later resent

- Simple replay
  The opponent simply copies the message and replays it later.

- Repetition that can be logged
  The opponent replay a time stamped message within a valid time window.

- Repetition that cannot be detected
  The attacker would have suppressed the original message from the receiver. Only the replay message alone arrives.

- Backward replay without modification
  This replay back to the sender itself.This is possible if the sender cannot easily recognize the difference between the message sent and the message received based on the content.

Countermeasures include
One approach to coping with replay attacks is to attach a sequence number to each message used in an authentication exchange. A new message is accepted only if its sequence number is in the proper order.

The difficulty with this approach is that it requires each party to keep track of the last sequence number for each claimant it has dealt with. Because of this overhead, sequence numbers are generally not used for authentication and key exchange. Instead, one of the following two general approaches is used:

- **Timestamps:** Party A accepts a message as fresh only if the message contains a **Timestamp** that is close enough to A"s knowledge of current time. This approach requires that clocks among the various participants be synchronized.
- **Challenge/response:** Party A, expecting a fresh message from B, first sends B a **nonce** (challenge) and requires that the subsequent message (response) received from B contain the correct nonce value.

Using Symmetric Encryption
- Use a two-level hierarchy of keys
- Usually with a trusted Key Distribution Center (KDC)
  - Each party shares own master key with KDC
  - KDC generates session keys used for connections between parties
  - Master keys used to distribute the session ke

Needham-Schroeder Protocol

- Original third-party key distribution protocol
- For session between A and B mediated by KDC
- Protocol overview is:

1. A→KDC: $ID_A || ID_B || N_1$
2. KDC→A: $EK_a[K_s || ID_B || N_1 || EK_b[K_s || ID_A]]$
3. A→B: $EK_b[K_s || ID_A]$
4. B→A: $EK_s[N_2]$
5. A→B: $EK_s[f(N_2)]$

Step 1: A to KDC ,transmit the id of source and destination and a nonce N_1 as a request.
Step 2: A securely acquires the session key in step 2 and a packet to B encrypted with EK_b is also received from KDC.
Step 3: A transmit to B the message it got from KDC.
Step 4: As a hand shake, B encrypts a new nonce N_2 and transmit to A with K_s.
Step 5: As a hand shake, A encrypt the function of N_2 with K_s
Step 4 and Step 5 as used as hand shake and prevent the reply attacks.

Attacks:

- Used to securely distribute a new session key for communications between A &B
- But is vulnerable to a replay attack if an old session key has been Compromised
  - Then message 3 can be resent convincing B that is communicating With A
- Modifications to address this require:
  - Timestamps
  - Using an extra nonce

DenningProtocol:
To overcome the above weakness by a modification to the Needham/Schroeder protocol that includes the addition of a timestamp to steps 2 and 3.

A→KDC: $ID_A || ID_B$

KDC → A: $E(K_a, [K_s || ID_B \; II \; T \; II \; E(K_b, [K_s \; III D_A \; II \; T])])$

A→B: $E(K_b, [K_s \; II \; ID_A \; IIT])$

B→A: $E(K_s, N1)$

A→ B: $E(K_s, f(N1))$

T is a timestamp that assures A and B that the session key has only just been generated. Thus, both A and B know that the key distribution is a fresh exchange.A and B can verify timeliness by checking that

$$| Clock - T | < \Delta t1 + \Delta t2$$

The **Denning protocol** seems to provide an increased degree of security compared to the Needham/Schroeder protocol. However, a new concern is raised: namely, that this new scheme requires reliance on clocks that are synchronized throughout the network.

suppress-replay attacks:

The problem occurs when a sender‟s clock is ahead of the intended recipient‟s clock. In this case, an opponent can intercept a message from the sender and replay it later when the timestamp in the message becomes current at the recipient‟s site. This replay could cause unexpected results.

Method to overcome:

One way to counter suppress-replay attacks is to enforce the requirement that parties regularly check their clocks against the KDC‟s clock. The other alternative, which avoids the need for clock synchronization, is to rely on handshaking protocols using nonce.

This latter alternative is not vulnerable to a suppress-replay attack, because the nonce the recipient will choose in the future are unpredictable to the sender.

An attempt is made to respond to the concerns about suppress replay attacks and at the same time fix the problems in the Needham/Schroeder protocol.

The protocol is

1. A: →B: $ID_A||N_a$
2. B: → KDC: $ID_B||N_b||E(K_b, [ID_A||N_a||T_b])$
3. KDC: → A: $E(K_a, [ID_B f\ fN_a f\ fKs\ f\ fT_b]) \„ E(Kb\ ,\ [ID_A\„\ Ks\ \„\ T_b]) \„ N_b$
4. A: →B: $Kb, [ID_A\„\ K_s\„\ T_b]) \„ E(K_s, N_b)$

1. A initiates the authentication exchange by generating a nonce,N_a , and sending that plus its identifier to B in plaintext. This nonce N_a will be returned to A in an encrypted message that includes the session key, assuring A of its timeliness.

2. B alerts the KDC that a session key is needed. Its message to the KDC includes its identifier and a nonce, . This nonce will be returned to B in an encrypted message that includes the session key, assuring B of its timeliness.B‟s message to the KDC also includes a block encrypted with the secret key shared by B and the KDC. This block is used to instruct the KDC to issue credentials to A; the block specifies the intended recipient of the credentials, a suggested expiration time for the credentials, and the nonce received from A.

3. The KDC passes on to A B‟s nonce and a block encrypted with the secret key that B shares with the KDC. The block serves as a “ticket” that can be used by A for subsequent authentications, as will be seen. The KDC also sends to A block encrypted with the secret key shared by A and the KDC. This block verifies that B has received A‟s initial message (ID_B) and that this is a timely message and not a replay (N_a), and it provides A with a session key (K_S) and the time limit on its use (T_b).

4. A transmits the ticket to B, together with the B‟s nonce, the latter encrypted with the session key. The ticket provides B with the secret key that is used to decrypt $E(K_S,N_b)$ to recover the

nonce. The fact that B‟s nonce is encrypted with the session key authenticates that the message came from A and is not a replay.

Using Public-Key Encryption

- Have a range of approaches based on the use of public-key encryption
- Need to ensure have correct public keys for other parties
- Using a central authentication server (AS)
- Various protocols exist using timestamps or nonces

Denning AS Protocol

- Denning presented the following:

1. A→AS: ID_A|| ID_B
2. AS→A: $E_{KRas}[ID_A||KU_a||T]$ || $EKRas[ID_B||KU_b||T]$
3. A→B: $E_{KRas}[ID_A||KU_a||T]$ || $EKRas[ID_B||KU_b||T]$ ||$EKU_b[E_{KRas}[Ks||T]]$

- Note session key is chosen by A, hence AS need not be trusted to protect it
- timestamps prevent replay but require synchronized clocks

Another approach proposed by woo and lam makes use of nonce.

1. A→KDC: ID_A|| ID_B
2. *KDC*→A: $E_{KRauth}[ID_B$ II $K_{Ub}]$
3. a→b: $EK_{Ub}[N_a$ II $ID_A]$
4. B→KDC: ID_B II ID_A II EK_{Uauth} $[N_a]$
5. KDC→B: E_{KRauth} [IDA II KUa] II $E_{KUb}[E_{KRauth}[N_a$ II K_s II $ID_B]]$
6. B→A: $E_{KUa}[E_{KRauth}[N_a$ II K_s II $ID_B]$ II $N_b]$
7. A→B: $E_{Ks}[N_b]$

Step 1: A informs the KDC of its intention to establish a secure connection with B.

Step 2: The KDC returns to A a copy of B‟s public key certificate.

Step 3: A informs B of its desire to communicate and sends a nonce N_a.

Step 4: B asks the KDC for A‟s public key certificate and request a session key.;B includes A‟s nonce so that the KDC can stamp the session key with that nonce.The nonce is protected using the KDC‟s public key.

Step 5: The KDC returns to B a copy of A‟s public key certificate, plus the information$[N_a,K_s,ID_B]$.

Step 6: The triple $[N_a,K_s,ID_B]$, still encrypted with the KDC‟s private key,is relayed to A,together with a nonce N_b generated by B.

All the foregoing are encrypted using A‟s public key. A retrieves the session key K_s and uses it to encrypt N_b and return it to B.

Step 7: Assures B of A‟s knowledge of the session key.

One-Way Authentication

- Required when sender & receiver are not in communications at same time (eg. Email)
- Have header in clear so can be delivered by email system
- May want contents of body protected & sender authenticated

Using Symmetric Encryption

- can refine use of KDC but can‟t have final exchange of nonce

  1. A→KDC: $ID_A || ID_B || N_1$
  2. KDC→A: $EK_a[K_s || ID_B || N_1 || EK_b[K_s || ID_A]]$
  3. A→B: $EK_b[K_s || ID_A] || EKs[M]$

- Does not protect against replays
  - could rely on timestamp in message, though email delays make this problematic

Public-Key Approaches

- If confidentiality is major concern, can use:

  $A \rightarrow B: EKU_b[K_s] || EK_s[M]$

In this case, the message is encrypted with a one-time secret key. A also encrypts this one-time key with B‟s public key. Only B will be able to use the corresponding private key to recover the one-time key and then use that key to decrypt the message. This scheme is more efficient than simply encrypting the entire message with B‟s public key.

- If authentication needed use a digital signature with a digital certificate:

  $A \rightarrow B: M „ II\ E_{KRa}(H(M))$

This method guarantees that A cannot later deny having sent the message. However, this technique is open to another kind of fraud. Bob composes a message to his boss Alice that contains an idea that will save the company money. He appends his digital signature and sends it into the e-mail system.

Eventually, the message will get delivered to Alice‟s mailbox. But suppose that Max has heard of Bob‟s idea and gains access to the mail queue before delivery. He finds Bob‟s message, strips off his signature, appends his, and requeues the message to be delivered to Alice. Max gets credit for Bob‟s idea.

To counter such a scheme, both the message and signature can be encrypted with the recipient‟s public key:

$$A \rightarrow B: E_{KUb}, [M || E_{KRa}, H(M)]$$

The latter two schemes require that B know A‟s public key and be convinced that it is timely. An effective way to provide this assurance is the digital certificate

$$A \rightarrow B: M \,||\, EKR_a [H(M)] \,||\, EKR_{as} \,II\, T|| \,ID_A \,||\, KU_a]$$

In addition to the message,A sends B the signature encrypted with A‟s private key and A‟s certificate encrypted with the private key of the authentication server. The recipient of the message first uses the certificate to obtain the sender‟s public key and verify that it is authentic and then uses the public key to verify the message itself.

DSS

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.

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 message 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 signature is accepted as valid. Because only the sender knows the private key, only the sender could have produced a valid signature.

DSS approach

The DSS approach also makes use of a hash function. The hash code is provided as input to a signature function along with a random number 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 (PU_G). The result is a signature consisting of two components, labeled *s* and *r(fig 3.15)*.

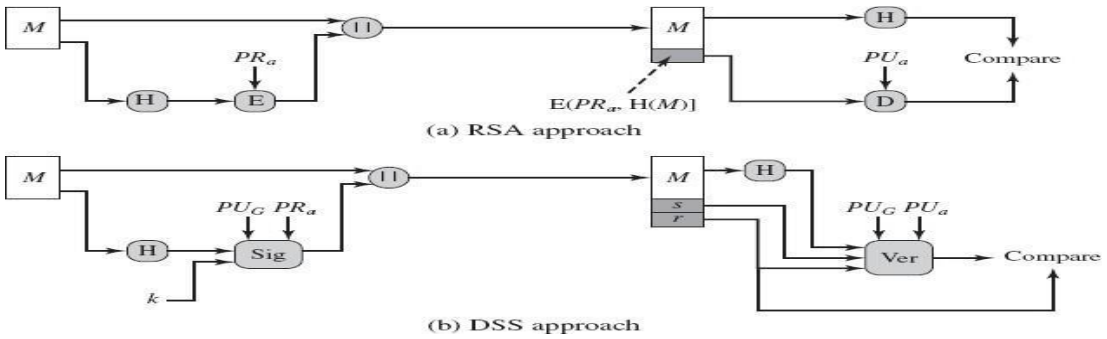


Figure 3.15 Two Approaches to Digital Signatures

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, 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 if the signature is valid. The signature function is such that only the sender, with knowledge of the private key, could have produced the valid signature.

The Digital Signature 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 $h^{(p-1)/q} \bmod p$, where *h* is an integer between 1 and (p-1).

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 chosen randomly. T

The public key is calculated from the private key as $y = g^x \bmod p$ .The calculation of given(Fig 3.16) 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*, mod p.

At the receiving end, verification is performed using the formulas. The receiver generates a quantity v that is a function of the public key components, the sender"s public key, and the hash code of the incoming message. If this quantity matches the component of the signature, then the signature is validated.

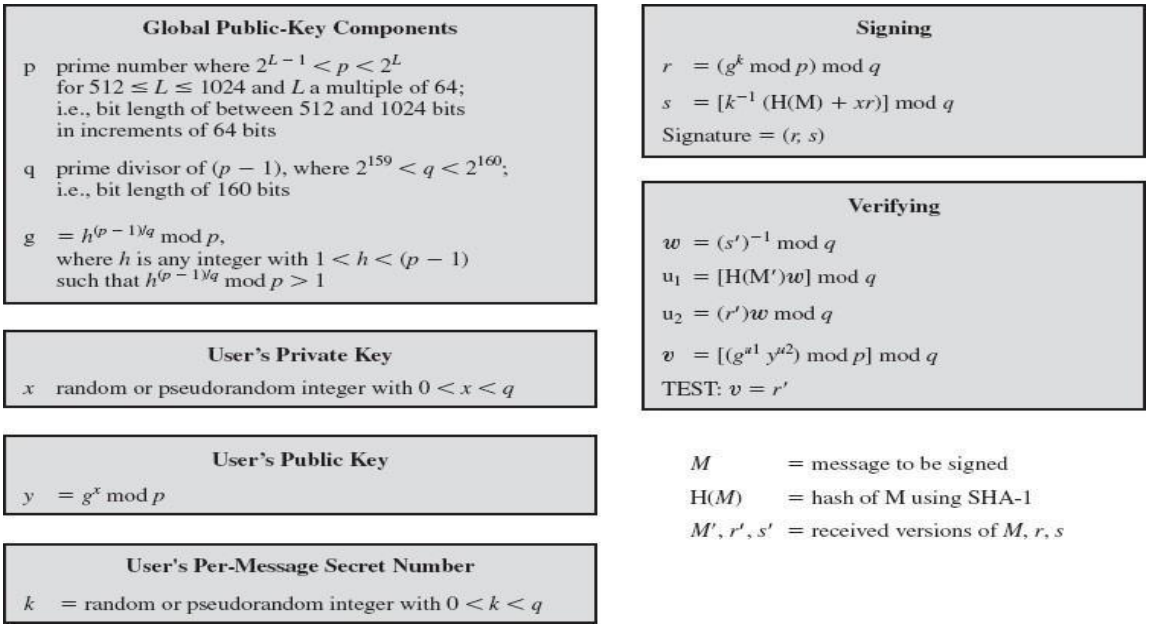


Figure.The Digital Signature Algorithm (DSA)

The value r 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 using the incoming message and signature, the public key of the user, and the global public key. Given the difficulty of taking discrete logarithms, it is infeasible for an opponent to recover k from r to recover x from s.

The only computationally demanding task in signature generation is the exponential calculation $g^k \bmod p$. Because this value does not depend on the message to be signed, it can be computed ahead of time.

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 chosen randomly. The public key is calculated from the private key as $y = g^x \bmod p$.

To create a signature, a user calculates two quantities, *r* and *s*, that are functions 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 and be unique for each signing.

At the receiving end, verification is performed using the formulas. The receiver generates a quantity *v* that is a function of the public key components, 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 (Fig).

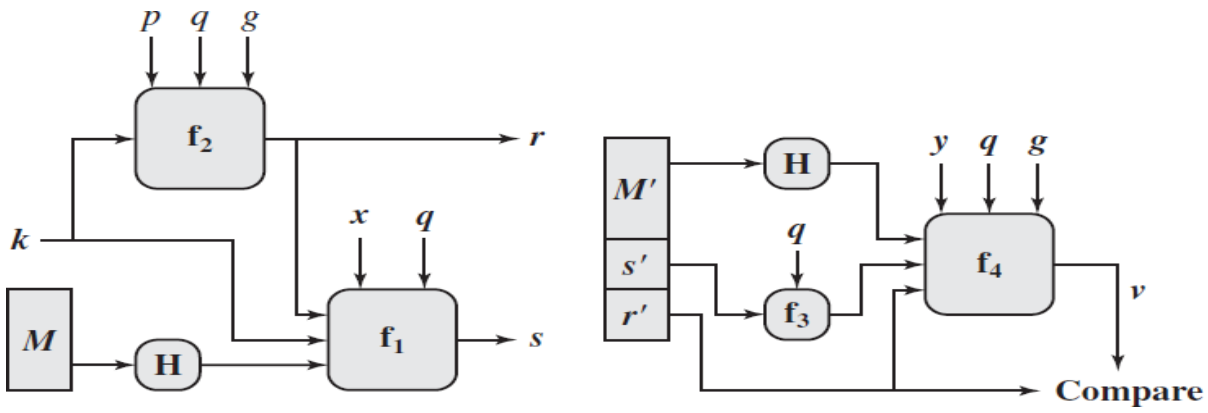


Figure DSS Signing and Verifying

AUTHENTICATION APPLICATIONS

One of the key aspects of cryptography and network security is authentication. It helps to establish trust by identifying a particular user or a system. There are many ways to authenticate a user. Traditionally, user ids and passwords have been used.

1.Authentication Requirements

During communication across networks, following attacks can be identified.

1. **Disclosure:** Releases 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.

3. **Masquerade:** Insertion of messages into the network fraudulent source.

4. **Content modification:** Changes to the content of the message, including insertion deletion, transposition and 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.

7. **Source repudiation:** Denial of transmission of message by source.

8. **Destination repudiation:** Denial of transmission of message by destination.

The security measures for the above mentioned attacks are as follows

For 1,2 - Message Confidentiality
3,4,5,6 - Message Authentication
7 - Digital Signatures
8 - Digital signature with protocol designed to counter the attack

2.Authentication Functions

Any message authentication or digital signature mechanism can be viewed as having fundamentally two levels.

1. **Lower level:** Some function that produces an authenticator: a value to be used to authenticate a message.

2. **Higher Level:** Lower layer functions are used to create a protocol that enables a receiver to verify the authenticity of message

The different types of functions that may be used to produce an authenticator are as follows:

1. **Message encryption:** The cipher text of the entire message serves as its authenticator.

2. **Message Authentication Code (MAC):** A public function of the message and a secret key that produces a fixed length value serves as the authenticator.

3. **Hash function:** A public function that maps a message of any length into a fixed length hash value, which serves as the authenticator.

KERBEROS

Kerberos provides a centralized authentication server whose function is to authenticate users to servers and servers to users. Kerberos relies exclusively on conventional encryption, making no use of public-key encryption.

Motivation

A distributed architecture consists of dedicated user workstations (clients) and distributed or centralized servers. In this environment, there are three approaches to security:

- Rely on each individual client workstation to assure the identity of its user or users and rely on each server to enforce a security policy based on user identification (ID).
- Require that client systems authenticate themselves to servers, but trust the client system concerning the identity of its user.
- Require the user to prove his or her identity for each service invoked. Also require that servers prove their identity to clients.

The following are the **requirements for Kerberos**:

- **Secure:** A network eavesdropper should not be able to obtain the necessary information to impersonate a user. More generally, Kerberos should be strong enough that a potential opponent does not find it to be the weak link.
- **Reliable:** For all services that rely on Kerberos for access control, lack of availability of the Kerberos service means lack of availability of the supported services. Hence, Kerberos should be highly reliable and should employ distributed server architecture, with one system able to back up another.
- **Transparent:** Ideally, the user should not be aware that authentication is taking place, beyond the requirement to enter a password.
- **Scalable:** The system should be capable of supporting large numbers of clients and servers. This suggests a modular, distributed architecture.

To support these requirements, the overall scheme of Kerberos is that of a trusted third-party authentication service that uses a protocol based on Needham and Schroeder.

It is trusted in the sense that clients and servers trust Kerberos to mediate their mutual authentication. Assuming the Kerberos protocol is well designed, and then the authentication service is secure if the Kerberos server itself is secure.

Two versions of Kerberos are in common use. **Version 4** and **Version 5**

Kerberos Version 4

Version 4 of Kerberos makes use of DES, in a rather elaborate protocol, to provide the authentication service

Simple Authentication Dialogue

In an unprotected network environment, any client can apply to any server for service. The obvious security risk is that of impersonation. To counter this threat, servers must be able to confirm the identities of clients who request service. But in an open environment, this places a substantial burden on each server.

An alternative is to use an authentication server (AS) that knows the passwords of all users and stores these in a centralized database. In addition, the AS shares a unique secret key with each server. The simple authentication dialogue is as follows:

1. C >> AS: $ID_c||P_c||ID_v$

2. AS >> C: Ticket

3. C >> V: $ID_c||$Ticket

Ticket= $EK_v(ID_c||ADc||ID_v)$

C : Client,
AS : Authentication Server,
V : Server, ID_c : ID of the client,
P_c : Password of the client,
AD_c : Address of client, ID_v : ID of the server,
K_v : secret key shared by AS and V,
|| : concatenation.

More Secure Authentication Dialogue

There are two major problems associated with the previous approach:

- Plaintext transmission of the password.
- Each time a user has to enter the password.

To solve these problems, we introduce a scheme for avoiding plaintext passwords, and a new server, known as ticket granting server (TGS). The hypothetical scenario is as follows:

Once per user logon session:-

1. C >> AS: $ID_c||ID_{tgs}$
2. AS >> C: $Ek_c(Ticket_{tgs})$

Once per type of service:

3. C >> TGS: $ID_c||ID_v||Ticket_{tgs}$
4. TGS >> C: $ticket_v$

Once per service session:

5. C >> V: $ID_c||$ $Ticket_v$

$Ticket_{tgs}= Ekt_{gs}(ID_c||AD_c||IDt_{gs}||TS_1||Lifetime_1)$

$Ticket_v= Ek_v(ID_c||AD_c||ID_v||TS2||Lifetime_2)$

C: Client, AS: Authentication Server, V: Server,
IDc : ID of the client, Pc:Password of the client, ADc: Address of client,
IDv : ID of the server, Kv: secret key shared by AS and V,
|| : concatenation, IDtgs: ID of the TGS server, TS1, TS2: time stamps, lifetime: lifetime of the ticket.

The new service, TGS, issues tickets to users who have been authenticated to AS. Thus, the user first requests a ticket-granting ticket ($Ticket_{tgs}$) from the AS. The client module in the user workstation saves this ticket.

Each time the user requires access to a new service, the client applies to the TGS, using the ticket to authenticate itself. The TGS then grants a ticket for the particular service. The client saves each service-granting ticket and uses it to authenticate its user to a server each time a particular service is requested.

Let us look at the details of this scheme:

1. The client requests a ticket-granting ticket on behalf of the user by sending its user's ID and password to the AS, together with the TGS ID, indicating a request to use the TGS service
2. The AS responds with a ticket that is encrypted with a key that is derived from the user's password.

When this response arrives at the client, the client prompts the user for his or her password, generates the key, and attempts to decrypt the incoming message.

If the correct password is supplied, the ticket is successfully recovered.

Because only the correct user should know the password, only the correct user can recover the ticket. Thus, we have used the password to obtain credentials from Kerberos without having to transmit the password in plaintext. Now that the client has a ticket-granting ticket, access to any server can be obtained with steps 3 and 4:

3. The client requests a service-granting ticket on behalf of the user. For this purpose, the client transmits a message to the TGS containing the user's ID, the ID of the desired service, and the ticket-granting ticket
4. The TGS decrypts the incoming ticket and verifies the success of the decryption by the presence of its ID. It checks to make sure that the lifetime has not expired. Then it compares the user ID and network address with the incoming information to authenticate the user. If the user is permitted access to the server V, the TGS issues a ticket to grant access to the requested service.

The service-granting ticket has the same structure as the ticket-granting ticket. Indeed, because the TGS is a server, we would expect that the same elements are needed to authenticate a client to the TGS and to authenticate a client to an application server.

Again, the ticket contains a timestamp and lifetime. If the user wants access to the same service at a later time, the client can simply use the previously acquired service-granting ticket and need not bother the user for a password.

Note that the ticket is encrypted with a secret key (K_v) known only to the TGS and the server, preventing alteration.

Finally, with a particular service-granting ticket, the client can gain access to the corresponding service with step 5:

5. The client requests access to a service on behalf of the user. For this purpose, the client transmits a message to the server containing the user's ID and the service-granting ticket. The server authenticates by using the contents of the ticket.

This new scenario satisfies the two requirements of only one password query per user session and protection of the user password.

Kerberos V4 Authentication Dialogue Message Exchange

Two additional problems remain in the more secure authentication dialogue:

- Lifetime associated with the ticket granting ticket. If the lifetime is very short, then the user will be repeatedly asked for a password. If the lifetime is long, then the opponent has the greater opportunity for replay.

- Requirement for the servers to authenticate themselves to users.

The actual Kerberos protocol version 4 is as follows
:

- A basic third-party authentication scheme
- Have an Authentication Server (AS)
    - Users initially negotiate with AS to identify self
    - AS provides a non-corruptible authentication credential (ticket granting ticket TGT)

- Have a Ticket Granting
    - Users subsequently request access to other services from TGS on basis of users TGT

(a) Authentication service exchange: to obtain ticket granting ticket
(1) $C \rightarrow AS : ID_C \text{ II } ID_{tgs} \text{ II } TS_1$ (2) $AS \rightarrow C : EKc [K_{c,tgs} \text{ II } ID_{tgs} \text{ II } TS_2 \text{ II } Lifetime_2 \text{ II } Ticket_{tgs}]$
(b) Ticket-Granting Service Exchange: to obtain service-granting ticket

(3) $C \rightarrow TGS: ID_v \parallel Ticket_{tgs} \parallel Authenticator_c$

(4) $TGS \rightarrow C: EK_{c,tgs}[K_{c,y} \parallel ID_v \parallel TS_4 \parallel Ticket_v]$

$Ticket_{tgs} = E_{K,tgs}[K_{c,tgs} \parallel ID_C \parallel AD_C \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2]$

$Ticket_v = E_{Kv}[K_{c,v} \parallel ID_C \parallel AD_C \parallel ID_v \parallel TS_4 \parallel Lifetime_4]$

$Authenticator_C = E_{Ktgs}[ID_C \parallel AD_C \parallel TS_3]$

(c) Client/Server Authentication Exchange: to obtain service

(5) $C \rightarrow V : Ticket_v \parallel Authenticator_c$

(6) $V \rightarrow C: E_{kc,v}[TS_5 + 1]$

$Ticket_v = EK_v[K_{c,v} \parallel ID_C \parallel AD_C \parallel Id_v \parallel TS_4 \parallel Lifetime_4]$

$Authenticator_c = EK_{tgs}[ID_C \parallel AD_C \parallel TS_3]$

Kerberos 4 Overview

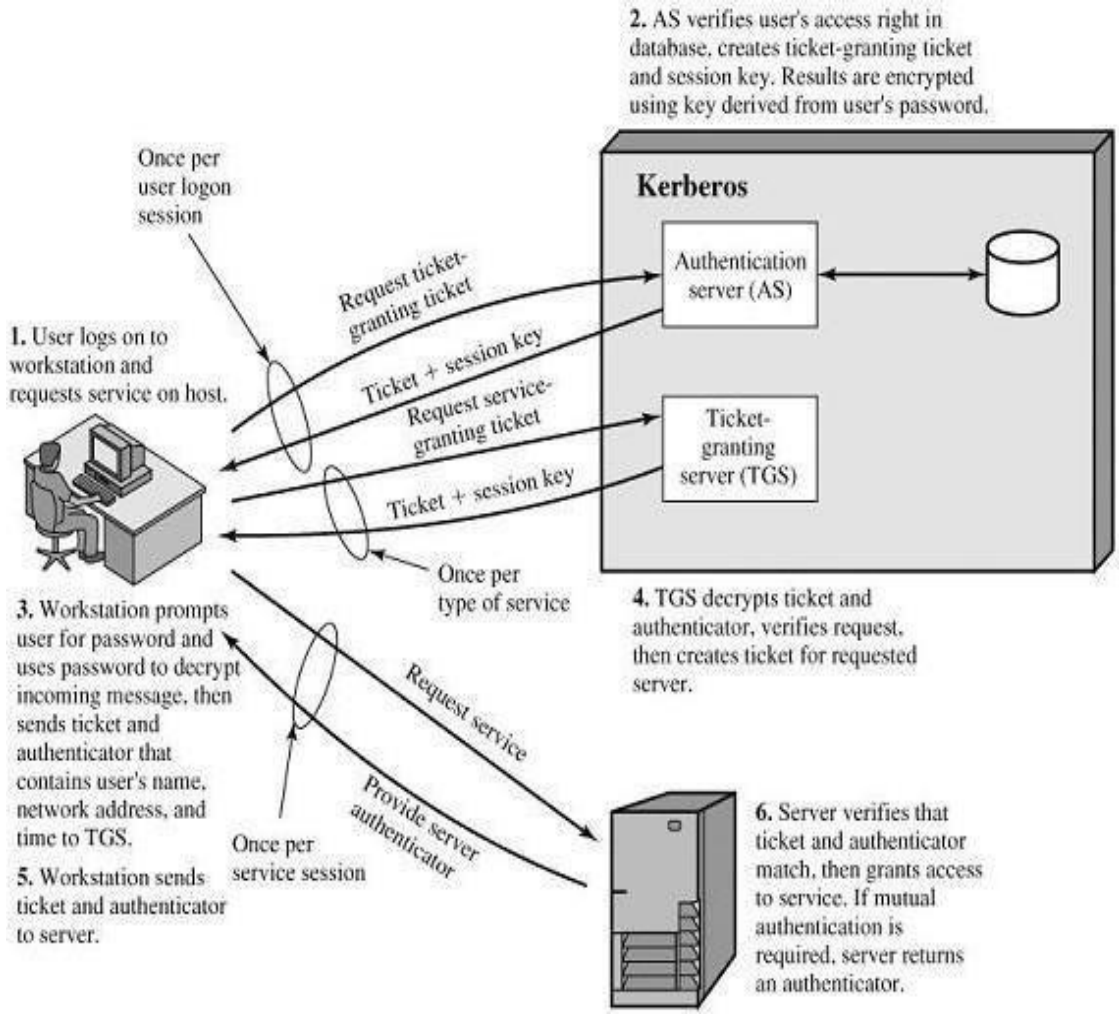


Fig 4.1 Overview of Kerberos 4

Kerberos Realms and Multiple Kerberi

A full-service Kerberos environment consisting of a Kerberos server, a number of clients, and a number of application servers requires the following:

4. The Kerberos server must have the user ID and hashed passwords of all participating users in its database. All users are registered with the Kerberos server.
5. The Kerberos server must share a secret key with each server. All servers are registered with the Kerberos server.

Such an environment is referred to as a **Kerberos realm**

The concept of *realm* can be explained as follows.

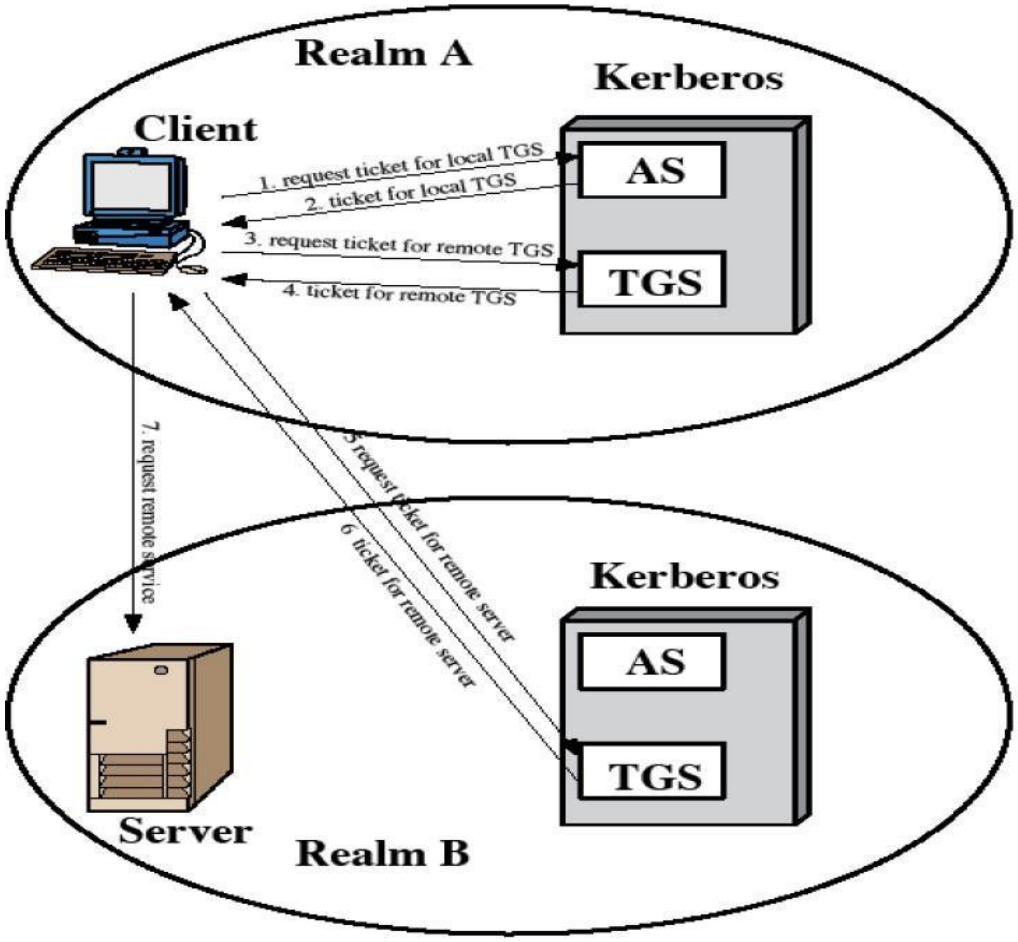


Fig .Request for service in another Realm

A Kerberos realm is a set of managed nodes that share the same Kerberos database. The Kerberos database resides on the Kerberos master computer system, which should be kept in a physically secure room. A read-only copy of the Kerberos database might also reside on other Kerberos computer systems.

However, all changes to the database must be made on the master computer system. Changing or accessing the contents of a Kerberos database requires the Kerberos master password. A related concept is that of a Kerberos principal, which is a service or user that is known to the Kerberos system.

Each Kerberos principal is identified by its principal name. Principal names consist of three parts: a service or user name, an instance name, and a realm name. Networks of clients and servers under different administrative organizations typically constitute different realms.

That is, it generally is not practical, or does not conform to administrative policy, to have users and servers in one administrative domain registered with a Kerberos server elsewhere.

However, users in one realm may need access to servers in other realms, and some servers may be willing to provide service to users from other realms, provided that those users are authenticated.

Kerberos provides a mechanism for supporting such inter realm authentication. For two realms to support inter realm authentication, a third requirement is added:

6. The Kerberos server in each interoperating realm shares a secret key with the server in the other realm. The two Kerberos servers are registered with each other.

The scheme requires that the Kerberos server in one realm trust the Kerberos server in the other realm to authenticate its users. Furthermore, the participating servers in the second realm must also be willing to trust the Kerberos server in the first realm.

The details of the exchanges illustrated in Fig 2are as follows:

$C \rightarrow AS$:ID_C II ID_{tgs} II TS_1

$AS \rightarrow C$:$EK_c[K_{c,tgs}$ ii ID_{tgs} II TS_2 II $Lifetime_2$ II $Ticket_{tgs}$

$C \rightarrow TGS$:ID_{tgsrem} II $Ticket_{tgs}$ II $Authenticator_c$

$TGS \rightarrow C$:$E\ K_{c,tgs}[K_{c,tgsrem}$ II ID_{tgsrem} II TS_4 II $Ticket_{tgsrem}$

$C \rightarrow TGS_{rem}$:ID_{vrem} II $Ticket_{tgsrem}$ II $Authenticator_c$

$TGS_{rem} \rightarrow C$:$EK_{c,tgsrem}$ [$K_{c,vrem}$ II ID_{vrem} II TS_6 II $Ticket_{vrem}$:

$C \rightarrow V_{rem}$:$Ticket_{vrem}$ II$Authenticator_c$

Differences between Versions 4 and 5

Version 5 is intended to address the limitations of version 4 in two areas: environmental shortcomings and technical deficiencies.

Environmental shortcomings:

7. **Encryption system dependence:**

Version 4 requires the use of DES. In version 5, ciphertext is tagged with an encryption type identifier so that any encryption technique may be used.

8. **Internet protocol dependence:**

Version 4 requires the use of Internet Protocol (IP) addresses. Version 5 network addresses are tagged with type and length, allowing any network address type to be used.

9. **Message byte ordering:**

In version 4, the sender of a message employs a byte ordering of its own choosing and tags the message to indicate least significant byte in lowest address In version 5, all message structures are defined using Abstract Syntax Notation One (ASN.1) and Basic Encoding Rules (BER), which provide an unambiguous byte ordering.

10. **Ticket lifetime:**

Lifetime values in version 4 are encoded in an 8-bit quantity in units of five minutes. In version 5, tickets include an explicit start time and end time, allowing tickets with arbitrary lifetimes.

11. **Authentication forwarding:**

Version 4 does not allow credentials issued to one client to be forwarded to some other host and used by some other client. Version 5 provides this capability.

Technical deficiencies in the version 4 protocol:

- Double encryption
- PCBC encryption
- Session keys
- Password attacks

The Version 5 Authentication Dialogue

(a) Authentication Service Exchange: to obtain ticket-granting ticket
(1) $C \rightarrow AS$: Options II ID_c II $Realm_c$ II Times II $Nonce_1$
(2) $AS \rightarrow C$: $Realm_c$ II ID_c II $Ticket_{tgs}$ II EK_c [$K_{c,tgs}$ II Times II $Nonce_1$ II $Realm_{tgs}$ II ID_{tgs}]
$Ticket_{tgs} = EK_{tgs}$ [Flags II $K_{c,tgs}$ II $Realm_c$ II ID_c II AD_c II Times]
(b) Ticket – Granting Service Exchange:to obtain service-granting ticket
(3) $C \rightarrow TGS$: Optionns II ID_v II Times II $Nonce_1$
(4) $TGS \rightarrow C$: $Realm_c$ II ID_c II $Ticket_v$ II $EK_{c,tgs}[K_{c,v}$ II Times II $Nonce_2$ II $Realm_v$ II $ID_v]$
$Ticket_{tgs} = EK_{tgs}$[Flags II $K_{c,tgs}$ II $Realm_c$ II ID_c II AD_c II Times]
$Ticket_v = Ek_v$[[Flags II $K_{c,v}$ II $Realm_c$ II ID_c II AD_c II Times]

$Authenticator_c = EK_{c,tgs}[ID_c \text{ II } Realm_c \text{ II } TS_1]$

(c) Client/Server AUTHENTICATION Exchange: to obtain service

(5) $C \rightarrow V$: Options II $Ticket_v$ II $Authenticator_c$
(6) $V \rightarrow C$: $EK_{c,v}$ [TS_2 II subkey II Seq #]
$Ticketv = EK_v[Flags \text{ II } K_{c,v} \text{ II } Realm_c \text{ II } ID_c \text{ II } AD_c \text{ II Times}]$
$Authenticator_c = E_{Kc,v}[ID_c \text{ II } Realm_c \text{ II } TS_2 \text{ II Subkey II Seq\#}]$

First, consider the authentication service exchange. Message (1) is a client request for a ticket-granting ticket. It includes the ID of the user and the TGS.

The following new elements are added:

- Realm: Indicates realm of user
- Options: Used to request that certain flags be set in the returned ticket
- Times: Used by the client to request the following time settings in the ticket:
  - from : the desired start time for the requested ticket
  - till : the requested expiration time for the requested ticket
  - r_{time} : requested renew-till time

Nonce: A random value to be repeated in message (2) to assure that the response is fresh and has not been replaced by an opponent .

Message (2) returns a ticket-granting ticket, identifying information for the client, and a block encrypted using the encryption key based on the user's password. This block includes the session key to be used between the client and the TGS, times specified in message (1), the nonce from message (1), and TGS identifying information.

The ticket itself includes the session key, identifying information for the client, the requested time values, and flags that reflect the status of this ticket and the requested options.

Let us now compare the ticket-granting service exchange for versions 4 and 5.

We see that message (3) for both versions include an authenticator, a ticket, and the name of the requested service.

In addition, version 5 includes requested times and options for the ticket and a nonce, all with functions similar to those of message (1). The authenticator itself is essentially the same as the one used in version 4.

The authenticator itself is essentially the same as the one used in version 4.

Message (4) has the same structure as message (2), returning a ticket plus information needed by the client, the latter encrypted with the session key now shared by the client and the TGS.

Finally, for the client/server authentication exchange, several new features appear in version 5. In message (5), the client may request as an option that mutual authentication is required. The authenticator includes several new fields as follows:

- **Subkey**: The client's choice for an encryption key to be used to protect this specific application session. If this field is omitted, the session key from the ticket (Kc,v) is used.

- **Sequence number**: An optional field that specifies the starting sequence number to be used by the server for messages sent to the client during this session. Messages may be sequence numbered to detect replays.

If mutual authentication is required, the server responds with message (6). This message includes the timestamp from the authenticator. Note that in version 4, the timestamp was incremented by one. This is not necessary in version 5 because the nature of the format of messages is such that it is not possible for an opponent to create message (6) without knowledge of the appropriate encryption keys.

X.509 AUTHENTICATION SERVICES

X.509 defines a framework for authentication services by the X.500 directory to its users. The directory consists of public-key certificates.

Each certificate contains the public key of a user and is signed with the private key of a trusted certification authority.

X.509 defines authentication protocols based on public key certificates. X.509 standard certificate format used in S/MIME, IP Security and SSL/TLS and SET.

Certificates

The certificates are created and stored in the directory by the trusted Certification Authority (CA). The directory server not having certification functions and not create public key. But the user obtains the certificate from some easily accessible location

The general format of the certificate as shown below Fig 4.3

The elements of the certificates are

1. **Version(V):** The default version is 1. The issuer and subject unique identifier are present in version 2. If one or more extensions are present in version 3.
2. **Serial Number (SN):** Unique integer value issued by CA
3. **Signature Algorithm Identifier (AI):** This algorithm is used to sign the certificate with some parameters
4. **Issuer Name (CA):** The name of the CA that created and signed this certificate
5. **Period of validity (T_A):** The first and last on which the certificate is valid
6. **Subject Name (A):** The name of the user to whom this certificate refers
7. **Subject's Public Key Information (AP):** The public key of the subject plus identifier of the algorithm for which this key is to be used, with associated parameters.

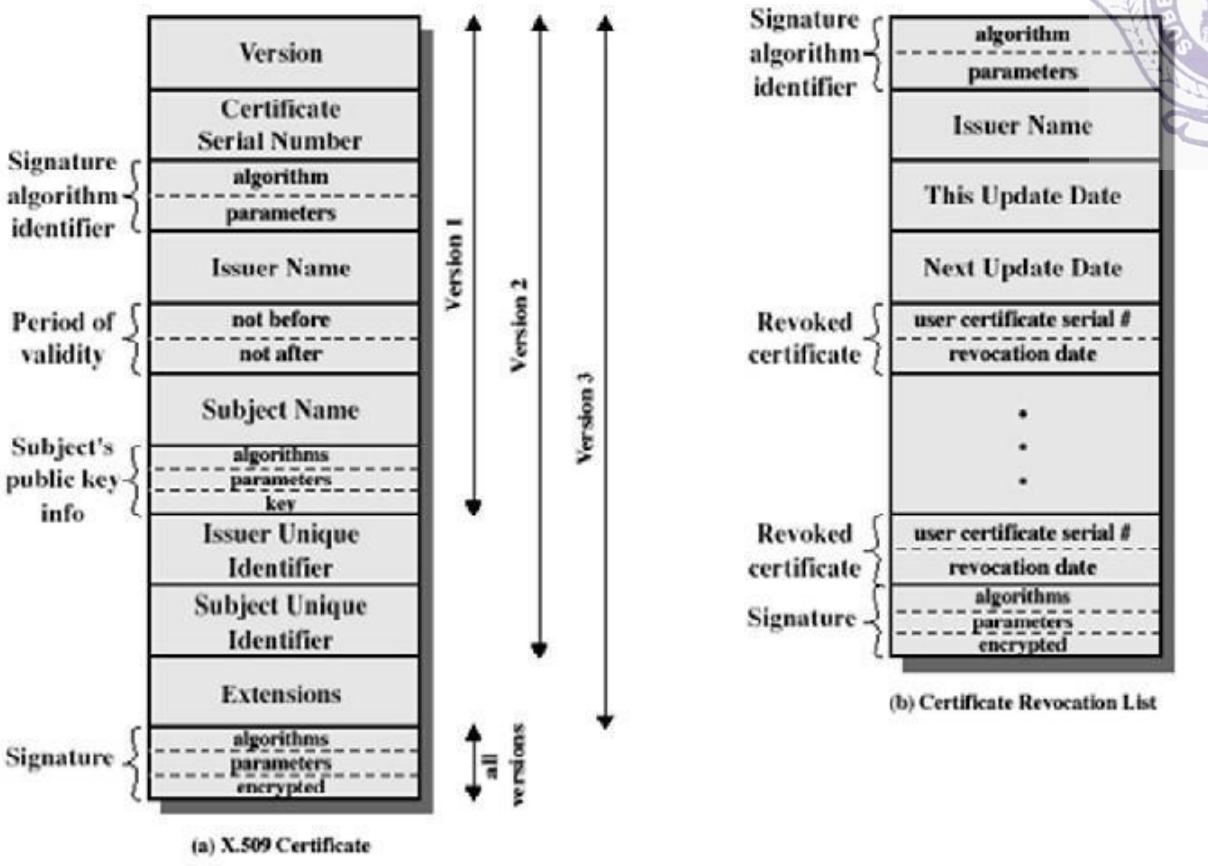


Fig X.509 AUTHENTICATION SERVICES

Issuer Unique Identifier: It is used to identify uniquely the issuing CA

8. **Subject Unique Identifier:** It is used to identify uniquely the subject
9. **Extensions:** A set of one or more extension fields
10. **Signature:** Covers all of the other fields of certificate; it contains hash code of other fields, encrypted with the CA‟s private key.

[**Note:** Unique identifier is used to avoid reuse of subject and issuer names over time]

Notation to define a certificate

CA<<A>> = CA {V, SN, AI, CA, TA, A, Ap}
where
Y<<X>> = The certificate of user X issued by certification authority Y.
Y {I} = The signing of I by Y. It consists of I with an encrypted hash code appended.

The CA signs the certificate with its private key. If the corresponding public key is known to a user, then that user can verify that a certificate signed by the CA is valid.

Generation and usage of certificate by a user

The user certificates generated by CA have the following characteristics:

11. Any user with access to the public key of the CA can verify the user public key that was certified.
12. No party other than the Certification Authority (CA) can modify the certificate without this being detected.
13. Certificates are unforgeable,

If all users belong to the same CA, the certificates can be placed in the directory for access by all users. If the number of users increased, single CA could not be satisfied all user requirements.

For example, User A has obtained the certificate from certificate authority x_1 and user B from x_2. Now the two CAs (x_1 and x2) securities exchange their own public keys in the form of certificates. That is x_1 must hold x_2"s certificate and x_2 holds x_1"s certificate

Now A want s to access B"s public key, it follows the following chain to obtain B"s public key.

x_1<< x_2>> x_2<<B>>

i.e., first A gets x_2"s certificate from x_1"s directory to obtain x_2"s public key. Then using x_2"s public key to obtain B"s certificate from x_2"s directory to obtain„s public key.

In the same method, B can obtain A"s public key with the reverse chain
x_2<< x_1>> x_1<<A>>

Hierarchy of CAs

To obtain public key certificate of user efficiently, more than one CAs can be arranged in a hierarchy, so that navigation in easy.

The connected circles indicate the hierarchical(Fig 4.4) relationship among the CAs; the associated boxes indicate certificates maintained in the directory for each CA entry. The directory entry for each CA includes two types of certificates:

- **Forward certificates**: Certificates of X generated by other CAs
- **Reverse certificates**: Certificates generated by X that are the certificates of other CAs

User A can acquire the following certificates from the directory to establish a certification path to B:

X<<W>> W <<V>> V <<Y>><<Z>> Z <<B>>

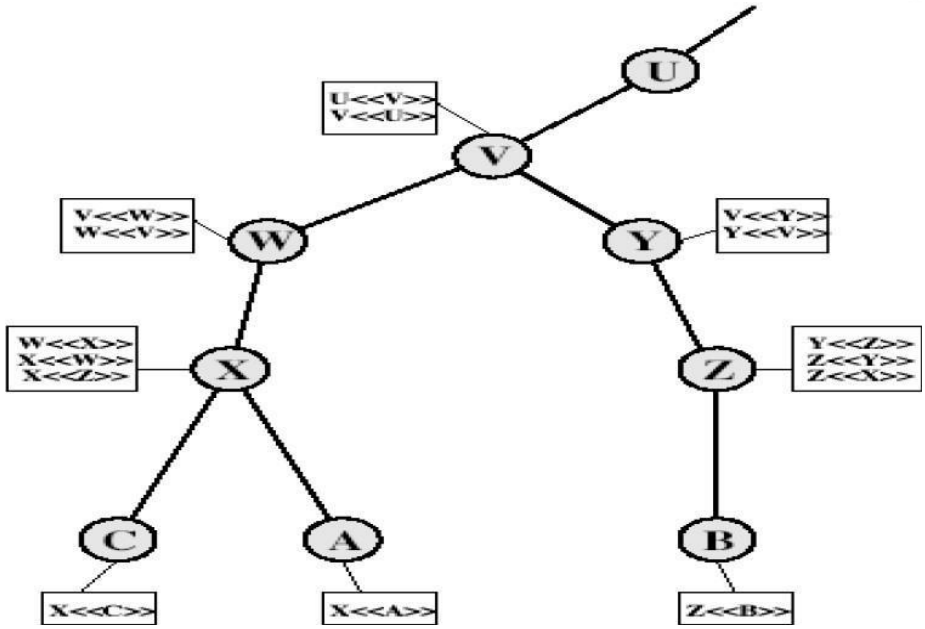


Fig4.4 : Hierarchy of X.509

Revocation of certificates

- Certificates have a period of validity
- May need to revoke before expiry, for the following reasons eg:
  - ✓ User's private key is compromised
  - ✓ User is no longer certified by this CA
  - ✓ CA's certificate is compromised
- CA"s maintain list of revoked certificates
  - ✓ The Certificate Revocation List (CRL)
- Users should check Certificates with CA"s CRL

Authentication Procedures

X.509 includes three alternative authentication procedures:

- One-Way Authentication
- Two-Way Authentication
- Three-Way Authentication

One-Way Authentication

One message (A→B) used to establish

- The identity of A and that message is from A
- Message was intended for B
- Integrity & originality of message

Message must include timestamp, nonce, B's identity and is signed by A

Two-Way Authentication

Two messages (A→B, B→A) which also establishes in addition:

- The identity of B and that reply is from B
- That reply is intended for A
- Integrity & originality of reply

Reply includes original nonce from A, also timestamp and nonce from B

Three-Way Authentication

Three messages (A→B, B→A, A→B) which enables above authentication without synchronized clocks (Fig 4.5)

- Has reply from A back to B containing signed copy of nonce from B
- Means that timestamps need not be checked or relied upon

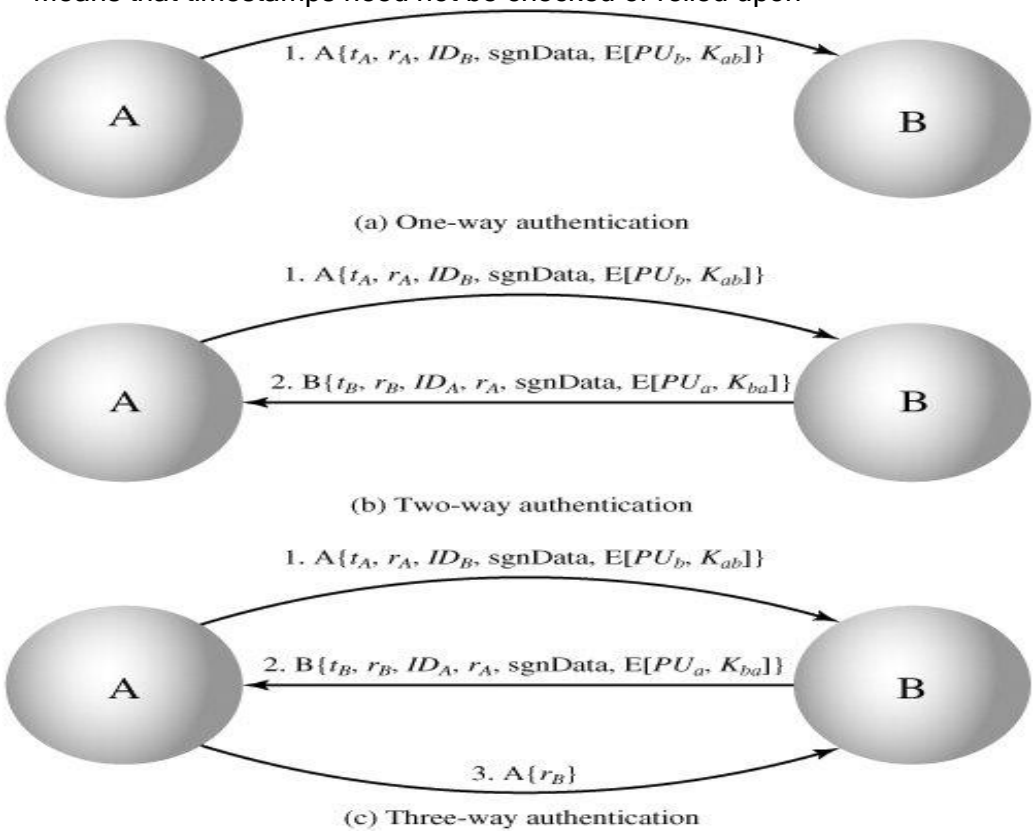


Fig: X509 Strong Authentication Procedure

X.509 Version 3

The following requirements not satisfied by version 2:

14. The Subject field is inadequate to convey the identity of a key owner to a public-key user.
15. The Subject field is also inadequate for many applications, which typically recognize entities by an Internet e-mail address, a URL, or some other Internet-related identification.
16. There is a need to indicate security policy information. There is a need to limit the damage that can result from a faulty or malicious CA by setting constraints on the applicability of a particular certificate.
17. It is important to be able to identify different keys used by the same owner at different times.

The certificate extensions fall into three main categories: key and policy information, subject and issuer attributes, and certification path constraints.

Key and Policy Information

These extensions convey additional information about the subject and issuer keys, plus indicators of certificate policy.. For example, a policy might be applicable to the authentication of electronic data interchange (EDI) transactions for the trading of goods within a given price range.

This area includes the following:

Authority key identifier: Identifies the public key to be used to verify the signature on this certificate or CRL.
Subject key identifier: Identifies the public key being certified.
Key usage: Indicates a restriction imposed as to the purposes for which, and the policies under which, the certified public key may be used.
Private-key usage period: Indicates the period of use of the private key corresponding to the public key. For example, with digital signature keys, the usage period for the signing private key is typically shorter than that for the verifying public key.
Certificate policies: Certificates may be used in environments where multiple policies apply.
Policy mappings: Used only in certificates for CAs issued by other CAs.

Certificate Subject and Issuer Attributes

These extensions support alternative names, in alternative formats, for a certificate subject or certificate issuer and can convey additional information about the certificate subject, to increase a certificate user's confidence that the certificate subject is a particular person or entity. For example, information such as postal address, position within a corporation, or picture image may be required.

The extension fields in this area include the following:

- **Subject alternative name:** Contains one or more alternative names, using any of a variety of forms
- **Subject directory attributes:** Conveys any desired X.500 directory attribute values for the subject of this certificate.

Certification Path Constraints

These extensions allow constraint specifications to be included in certificates issued for CAs by other CAs. The extension fields in this area include the following:

- **Basic constraints:** Indicates if the subject may act as a CA. If so, a certification path length constraint may be specified.
- **Name constraints**: Indicates a name space within which all subject names in subsequent certificates in a certification path must be located.
- **Policy constraints**: Specifies constraints that may require explicit certificate policy identification or inhibit policy mapping for the remainder of the certification path.

UNIT-5
CYBERCRIME AND SECURITY

5.1 Cybercrime or a computer-oriented crime is a crime that includes a computer and a network. The computer may have been used in the execution of a crime or it may be the target. Cybercrime is the use of a computer as a weapon for committing crimes such as committing fraud, identity theft, or breaching privacy. Cybercrime, especially through the Internet, has grown in importance as the computer has become central to every field like commerce, entertainment, and government. Cybercrime may endanger a person or a nation's security and financial health. Cybercrime encloses a wide range of activities, but these can generally be divided into two categories:

1. Crimes that aim at computer networks or devices. These types of crimes involve different threats (like virus, bugs etc.) and denial-of-service (DoS) attacks.
2. Crimes that use computer networks to commit other criminal activities. These types of crimes include cyber stalking, financial fraud or identity theft.

Cybercrime and Information Security

The terms **Cyber Security** and **Information Security** are often used interchangeably. As they both are responsible for the security and protecting the computer system from threats and information breaches and often Cybersecurity and information security are so closely linked that they may seem synonymous and unfortunately, they are used synonymously.

Parameters	CYBER SECURITY	INFORMATION SECURITY
Basic Definition	It is the practice of protecting the data from outside the resource on the internet.	It is all about protecting information from unauthorized users, access, and data modification or removal in order to provide confidentiality, integrity, and availability.
Protect	It is about the ability to protect the use of cyberspace from cyber attacks.	It deals with the protection of data from any form of threat.
Scope	Cybersecurity to protect anything in the cyber realm.	Information security is for information irrespective of the realm.
Threat	Cybersecurity deals with the danger in cyberspace.	Information security deals with the protection of data from any form of threat.
Attacks	Cybersecurity strikes against Cyber crimes, cyber frauds, and law enforcement.	Information security strikes against unauthorized access, disclosure modification, and disruption.

Parameters	CYBER SECURITY	INFORMATION SECURITY
Professionals	Cyber security professionals deal with the prevention of active threats or Advanced Persistent threats (APT).	Information security professionals are the foundation of data security and security professionals associated with it are responsible for policies, processes, and organizational roles and responsibilities that assure confidentiality, integrity, and availability.
Deals with	It deals with threats that may or may not exist in the cyber realm such as protecting your social media account, personal information, etc.	It deals with information Assets and integrity, confidentiality, and availability.
Defence	Acts as first line of defence.	Comes into play when security is breached.
Threats	Primarily deals with digital threats, such as hacking, malware, and phishing	Addresses a wider range of threats, including physical theft, espionage, and human error
Goal	Protects against unauthorized access, use, disclosure, disruption, modification, or destruction of digital information	Protects the confidentiality, integrity, and availability of all types of information, regardless of the medium in which it is stored
Technologies	Relies on a variety of technologies, such as firewalls, antivirus software, and intrusion detection systems	Uses a range of technologies, including encryption, access controls, and data loss prevention tools
Skills required	Requires specialized knowledge of computer systems and networks, as well as	Requires knowledge of risk management, compliance, legal and regulatory issues, as well as technical knowledge

Parameters	CYBER SECURITY	INFORMATION SECURITY
	programming and software development skills	
Focus on data	Emphasizes protecting the data itself, regardless of where it is stored or how it is transmitted	Emphasizes the protection of information assets, which includes data but also other information such as intellectual property, trade secrets, and confidential customer information
Threat landscape	Deals with constantly evolving threats, such as new forms of malware and emerging cybercrime techniques	Deals with a wide range of threats, including physical security breaches, insider threats, and social engineering attacks

5.2 Classification of Cyber Crime:

1. **Cyber Terrorism**
   Cyber terrorism is the use of the computer and internet to perform violent acts that result in loss of life. This may include different type of activities either by software or hardware for threatening life of citizens.
   In general, Cyber terrorism can be defined as an act of terrorism committed through the use of cyberspace or computer resources.

2. **Cyber Extortion**
   Cyber extortion occurs when a website, e-mail server or computer system is subjected to or threatened with repeated denial of service or other attacks by malicious hackers. These hackers demand huge money in return for assurance to stop the attacks and to offer protection.

3. **Cyber Warfare**
   Cyber warfare is the use or targeting in a battle space or warfare context of computers, online control systems and networks. It involves both offensive and defensive operations concerning to the threat of cyber attacks, espionage and sabotage.

4. **Internet Fraud**
   Internet fraud is a type of fraud or deceit which makes use of the Internet and could include hiding of information or providing incorrect information for the purpose of deceiving victims for money or property. Internet fraud is not considered a single, distinctive crime but covers a range of illegal and illicit actions that are committed in cyberspace.

5. **Cyber Stalking**
   This is a kind of online harassment wherein the victim is subjected to a barrage of online messages and emails. In this case, these stalkers know their victims and instead

of offline stalking, they use the Internet to stalk. However, if they notice that cyber stalking is not having the desired effect, they begin offline stalking along with cyber stalking to make the victims' lives more miserable.

Challenges of Cyber Crime:

1. **People are unaware of their cyber rights**
   The Cybercrime usually happen with illiterate people around the world who are unaware about their cyber rights implemented by the government of that particular country.

2. **Anonymity**
   Those who Commit cybercrime are **anonymous** for us so we cannot do anything to that person.

3. **Less numbers of case registered**
   Every country in the world faces the challenge of cyber crime and the rate of cyber crime is increasing day by day because the people who even don't register a case of cyber crime and this is major challenge for us as well as for authorities as well.

4. **Mostly committed by well educated people**
   Committing a cyber crime is not a cup of tea for every individual. The person who commits cyber crime is a very **technical** person so he knows how to commit the crime and not get caught by the authorities.

5. **No harsh punishment-**
   In Cyber crime there is no harsh punishment in every cases. But there is harsh punishment in some cases like when somebody commits cyber terrorism in that case there is harsh punishment for that individual. But in other cases there is no harsh punishment so this factor also gives encouragement to that person who commits cyber crime.

Prevention of Cyber Crime:

Below are some points by means of which we can prevent cyber crime:

1. **Use strong password**
   Maintain different password and username combinations for each account and resist the temptation to write them down. Weak passwords can be easily cracked using certain attacking methods like Brute force attack, Rainbow table attack etc, So make them complex. That means combination of letters, numbers and special characters.

2. **Use trusted antivirus in devices**
   Always use trustworthy and highly advanced antivirus software in mobile and personal computers. This leads to the prevention of different virus attack on devices.

3. **Keep social media private**
   Always keep your social media accounts data privacy only to your friends. Also make sure only to make friends who are known to you.

4. **Keep your device software updated**

Whenever you get the updates of the system software update it at the same time because sometimes the previous version can be easily attacked.

5. **Use secure network**
   Public Wi-Fi are vulnerable. Avoid conducting financial or corporate transactions on these networks.

6. **Never open attachments in spam emails**
   A computer get infected by malware attacks and other forms of cybercrime is via email attachments in spam emails. Never open an attachment from a sender you do not know.
7. **Software should be updated –** Operating system should be updated regularly when it comes to internet security. This can become a potential threat when cybercriminals exploit flaws in the system.

5.2 Tools and Methods for Cybersecurity

The Internet is a booming technology now and every organization uses the internet for their day-to-day activities. Cyber security is a major concern in the present technology world. With the rapid growth in the internet there will be a lot of tools and techniques readily available for cyber attackers to attack any system or network.

Every institution in the internet is now vulnerable to attackers as they are using public cyberspace. Everyone is concerned to protect their network and system from the attacks. The three main measures that the institutions follow in cyberspace is

1. Preventive measures
2. Detective measures
3. Corrective measures

In cyber security, the organizations must know many techniques and technologies that help them to keep the attacker at bay without getting attacked. These techniques will help to maintain these three measures in cyber security

1. Encryption
2. Firewall
3. VPN
4. Intrusion Detection
5. Access control (authorization & authentication)
6. Antivirus

Encryption

We already discussed the method of encryption in our previous tutorials. Encryption is a cyber-security method that helps us to store and send sensitive information across the internet without losing privacy and security.

The process of changing the information to another format using a secret key is called encoding or encryption and the information that is changed into such format is called ciphertext.

The process of converting back to this cipher text using the secret key to the readable format is called decryption or decoding. And the information that is decrypted is called decipher text.

Encryption is of different types like those that are symmetric and asymmetric based on the key in which symmetric uses only one key for encoding and decoding, where asymmetric uses two keys one is public and other is secret for encoding and decoding the text.

Importance of Encryption

- Data transit is vulnerable
- Security threats are evolving
- Many applications expose data
- Hacking is highly profitable business

Firewall

Like that name, firewalls act as a wall of fire which helps to secure a private network from the vast ocean of networks in the public internet. Firewall can be defined as a network security system which can be hardware or software or combination of both that protects a private network from the unauthorized access and usage of private network resources from the outside public network.

When a Fire is installed all the data packets that are leaving or coming to a private network should pass the firewall and the firewall check each packet for any malicious activity.

Firewalls can be classified on different criteria, where we disucss only the important aspects.

1. Processing mode
2. Deployment area
3. Architectural Implementation

We can divide the firewalls based on processing mode such as

1. Packet filtering
2. Application gateways
3. Circuit gateways
4. Mac layer firewalls
5. Hybrid

Packet Filtering

Each data in the internet is travelled as packets from one network computer to another network. These packets have a header that contains the sender and receiver details and a data part which contains the data that to be transferred.

Firewall will act as a wall that checks each packet header for the sender and receiver information and validates the packet. Firewalls take the decision depending on the authenticity of a packet, whether to forward the packet or drop it there.

There will be a well set of rules inside the firewall to take the decision and it scans the network packets for any malicious activity in the packets. Most of the firewall will work on the rules combination such as

1. IP source address and destination
2. Direction
3. TCP and UDP port requests.

In detail packet filtering can be divided into different yes such as

1. Static filtering: In this type the rules in a firewall are decided by a administrator
2. Dynamic filtering: In this type the rules for the firewall are made by the firewall itself.
3. Stateful inspection: Helps to track the connections from internal and external systems.

Application Gateways

Gateways are the place where a packet enters or leaves the system or a network. It is a firewall proxy, which provides network security for a system, which needs high security. It provides a secure communication between the user and server. This application gateway helps us to protect the data at the application level, which means it can filter some specific

data from some applications like bittorrent, FTP, telnet, etc. This firewall will be the middleman between the requested user and the server which blocks malicious packets.
For example, when a user request a data from a server, the connection is firs establish between the user and proxy server and then the proxy make the connection with real server.

Circuit Gateways

Circuit level gateways are the firewall, which works in the transport layer. Transport layer handles the TCP and UDP connections which mean these circuit gateways can able to handle the packets in a TCP or UDP connection.
These circuit gateways act in between the transport and application layers called session layer, which can handle and monitor packets and handshaking in TCP or UDP connections. This gateway can also act as Virtual private networks.

MAC Layer Firewalls

MAC layer firewalls will work in the media access control layer. It can able to filter the MAC address of the users who make requests with the server and is able to block a user if any malicious activity is found.
This layer will have a list of entries that include the MAC address of some of the systems, which act as host, and that list is called Access Control List. This list act as an important role in deciding which packet has to be sent to the host system.

Hybrid Firewalls

A firewall is the combination of all the firewalls we mentioned above so it can have all the features of the firewalls we discussed above.
Now another classification of firewall is based on the place where that firewall is intended to use like,

1. Commercial Appliances
2. Small Office
3. Home software

We have to choose a firewall configuration for an organization depend on some factors, that includes

1. Objective of network
2. Ability to implement the firewall
3. Cost affordability

We can divide the firewall configuration into 4 types according to these principles

1. Packet filtering routers
2. Screened host firewalls
3. Dual homed host firewalls
4. Screened Subnet firewalls

VPN

We all heard about VPN and other proxy methods to surf the internet to access any of the website without losing our privacy. A VPN stands for Virtual Private Network; It is a cyber-security technique to transfer files and sensitive information across the internet using a safe and secure tunnel.

How VPN achieve a secure transmission?

A VPN makes a connection from a user need system to a network that is encrypted and safe. This connection can be used for transmitting data and sensitive information without any eavesdropping or illegal accessing. Using a VPN can hide our IP and our geographical information so we can access any website that is restricted geographically.
VPN is like a firewall but protecting the user data and information in the internet. End user must have a login to enter the VPN servers and there the secure tunnels begin. Once the user enters the VPN server hey can send the sensitive information through these tunnels securely with privacy.

Intrusion Detection System (IDS)

Intrusion Detection System is a security technique in cyber security, which monitors the system and the network of an organization. IDS help the system administrator to find the attack that originated from outside or inside the organization.

Firewall is a security measure for an organization that helps us to filter the outside traffic to check the malicious activity from outside the organization, the IDS helps the system admin to protect the firewall as it give alarm to the system admin if anyone tries to break the firewall.

Intrusion detection system has different types that are the following

1. NIDS
2. HIDS
3. Signature based
4. Anomaly based

NIDS

It monitors the network traffic to find all the anomalies or attacks that originates from inside or outside the organization.

HIDS

HIDS stands for Host Intrusion Detection System, which runs on almost all the computers and networks in the organization. HIDS helps to monitor all the internal traffic and its anomalies. It also detects the traffic anomalies which the NIDS security monitoring failed to catch.

Signature Based Intrusion Detection

This is another monitoring system which is able to detect patterns that are malicious in nature. It helps to detect all the anomalies that come from internal or external sources. This helps to detect all the known patterns such as malwares but will fail to detect new threats.

Anomaly Based Detection System

It is used to monitor the threats that are unknown to the current scenario as the malwares are increasing a lot. It senses the malicious activity and informs the administrator.

Access Control (Authentication & Authorization)

Access control is a method of restricting access to a place or a system for unauthorized persons. Access control is a process for reducing the security risk from unauthorized access to sensitive information.

According to this principle, the privileges for accessing a resource will be given only to the required entities for the essential time required for them to complete the task.

For example consider a employee needs to access a system which have sensitive information for complete a assigned task for him. The system admin has to give him only the essential privileges to that information he needed and revoke the access after he complete his task.

Access control is a combination of two components Authorization and Authentication. Authentication is a process, which is done by the user with his credentials. For example, the username and password of a Gmail account. Authorization is a process that is done by an admin who gives access to the users.

Access control is of two types, which are

1. **Physical Access Control:** This is the access control, which is in physical form like access to a building, inside a bank etc.
2. **Logical Access Control:** Logical Access is like accessing a Gmail account, login to a computer etc.

Antivirus

Antivirus is a software which helps to protect a system from all types of threats like virus, worms, Trojans, adwares, malwares and many more. We have different methods to protect a network but we need an antivirus in every system to make them protected and always make sure the virus database is updated.

Antivirus software has a virus database which has the patterns and features all known viruses and threats. Antivirus scan all the files in our system for checking these kinds of patterns or

features present and remove such programs into a vault and delete fro that vault. So We must use only an updated antivirus database for efficient results.

5.3 Password Cracking

Password cracking is one of the imperative phases of the hacking framework. Password cracking is a way to recuperate passwords from the information stored or sent by a PC or mainframe. The motivation behind password cracking is to assist a client with recuperating a failed authentication or recovering a password, as a preventive measure by framework chairmen to check for effectively weak passwords, or an assailant can utilize this cycle to acquire unapproved framework access.

Types of Password Attacks :

Password cracking is consistently violated regardless of the legal aspects to secure from unapproved framework access, for instance, recovering a password the customer had forgotten etc. This hack arrangement depends upon aggressors exercises, which are ordinarily one of the four types:

1. **Non-Electronic Attacks**
   This is most likely the hacker's first go-to to acquire the target system password. These sorts of password cracking hacks don't need any specialized ability or information about hacking or misuse of frameworks. Along these lines, this is a non-electronic hack. A few strategies used for actualizing these sorts of hacks are social engineering, dumpster diving, shoulder surfing, and so forth.
2. **Active Online Attacks**
   This is perhaps the most straightforward approach to acquire unapproved manager-level mainframe access. To crack the passwords, a hacker needs to have correspondence with the objective machines as it is obligatory for password access. A few techniques used for actualizing these sorts of hacks are word reference, brute-forcing, password speculating, hash infusion, phishing, LLMNR/NBT-NS Poisoning, utilizing Trojan/spyware/keyloggers, and so forth.
3. **Passive Online Attacks**
   An uninvolved hack is a deliberate attack that doesn't bring about a change to the framework in any capacity. In these sorts of hacks, the hacker doesn't have to deal with the framework. In light of everything, he/she idly screens or records the data ignoring the correspondence channel to and from the mainframe. The attacker then uses the critical data to break into the system. Techniques used to perform passive online hacks incorporate replay attacks, wire-sniffing, man-in-the-middle attack, and so on.
4. **Offline Attacks**
   Disconnected hacks allude to password attacks where an aggressor attempts to recuperate clear content passwords from a password hash dump. These sorts of hacks are habitually dreary yet can be viable, as password hashes can be changed due to their more modest keyspace and more restricted length. Aggressors utilize preprocessed hashes from rainbow tables to perform disconnected and conveyed network hacks.

Some of the best practices protecting against password cracking include :

1. Perform data security reviews to screen and track password assaults.
2. Try not to utilize a similar password during the password change.
3. Try not to share passwords.
4. Do whatever it takes not to use passwords that can be found in a word reference.
5. Make an effort not to use clear content shows and shows with weak encryption.
6. Set the password change technique to 30 days.
7. Try not to store passwords in an unstable area.
8. Try not to utilize any mainframe's or PC's default passwords.

9. Unpatched computers can reset passwords during cradle flood or Denial of Service assaults. Try to refresh the framework.
10. Empower account lockout with a specific number of endeavors, counter time, and lockout span. One of the best approaches to oversee passwords in associations is to set a computerized password reset.
11. Ensure that the computer or server's BIOS is scrambled with a password, particularly on devices that are unprotected from real perils, for instance, centralized servers and PCs.

5.4 Keyloggers

Keyloggers are many hackers and script kiddie's favorite tools. Keylogging is a method that was first imagined back in the year 1983. Around then, the utilization of this product was uncommon and just the top examination organizations and spies could get their hands on it, yet today, it is a typical element offered by most government operative applications like TheOneSpy. Individuals use it as an opportunity to guarantee the assurance of their families, organizations, and the ones they care about.

Keylogger is a software that records each and every keystroke you enter, including mouse clicks. Hardware keyloggers are also available which will be inserted between keyboard and CPU. It provides the following features:

1. It takes a minute to install this software/hardware in the victim's system, from the next second onwards attacker will get every activity going on in the victim computer.
2. Each and every activity happening in the victim's system with screenshots will be recorded. This activity will be saved in the victim's system or it can be mailed to the attacker email or can be uploaded to the FTP server. Wondered? Let's see how attackers do this along with protection techniques.
3. Keylogging highlight of spy applications is adept at recording each and every keystroke made by utilizing a console, regardless of whether it is an on-screen console.
4. It likewise takes a screen capture of the screen when the client is composing (Usually this screen capture is taken when a catch on the mouse is clicked).
5. It works watchfully, escaped the client's view, for example, the focused on the client could never discover that all his keystrokes are being recorded.
6. Keyloggers recorder can record writings, email, and any information you compose at whatever point using your support.
7. The log record made by the keyloggers would then have the option to be sent to a predefined gatherer.
8. Some keyloggers tasks will likewise record any email that tends to your use and Web website URLs you visit.

Some software keyloggers code can capture additional information without requiring any keyboard key presses as input. They include:

1. **Clipboard logging:** Anything duplicated to the clipboard is caught.
2. **Screen logging:** Randomly coordinated screen captures of your PC are logged.
3. **Control text capture:** The Windows API allows for programs to request the text value of some controls, it means a password can still be captured albeit it is behind a password mask.
4. **Activity tracking:** Recording of which programs, folders, and windows are opened and also the screenshots of every.
5. Recording of program queries, instant message conversations, FTP downloads alongside the other internet activities.

Types Of Keylogger

There are basically two types of Keyloggers:

1. **Hardware Keylogger:** This is a thumb-size device. It records all the keystrokes you enter from the keyboard then saves it in its memory. Later this data will be analyzed. The drawback of this device is, It can't record mouse clicks, can't take screenshots, and even can't email, more importantly, It requires physical access to the machine. Hardware Keylogger is advantageous because it's not hooked into any software nor can it's detected by any software.
2. **Software Keylogger:** Software Keylogger can be installed in the victim's system even if they use updated Antivirus. There are lots of software available in market which make a Keylogger undetectable by latest antivirus, we are going to study about them too in upcoming chapters. There are many keyloggers available in market with various features. Some examples of Software Keyloggers are:
    1. Revealer Keylogger
    2. Ardamax Keylogger
    3. WinSpy
    4. Invisible Keylogger
    5. Refog Keylogger
    6. Activity Keylogger
    7. Keystroke Keyloggers

NOTE: Before purchasing any Keylogger make sure it has the following features:

1. Undetectable by Antivirus.
2. Remote Installation.
3. Stealth mode.

Uses of a Keylogger

Keylogger offers following uses to the user:

1. **Parental control:** It's a great way to track the activity of your children through Keylogger without getting caught. Likewise, you can get informed rapidly in regard to a specific page they got to on the web, their area, and numerous other fundamental things. Numerous guardians decide to control the perusing history of their children, with some extraordinary keyloggers.
2. **Security:** If you need to be certain your staff is regarding the standards, at that point you can get a warning with respect to their exercises through a Keylogger. Along these lines, you get the chance to save them in a protected mode for the organization's advantage.
3. **Partner exercises following:** On the off chance that you sense things showing that your mate is undermining you; at that point, you can utilize a Keylogger for android phones that will empower you to realize what your companion is doing. Leave it alone over the web fueled projects (i.e., WhatsApp, Snapchat, looking in the versatile program, and so on.) or even on phone messages. So with everything taken into account, the Android Keylogger applications can assist you with following your life partner's movement.

Acquistion of Keylogger

A Keylogger is often installed on your computer in one of many ways. Anybody with access to your PC could introduce it; keyloggers could come as an area of a plague or from any application establishment, in spite of how misleadingly honest it's getting the chance to look. This is a part of the rationale why you ought to always make certain that you're downloading files from a trusted resource.

Most companies implant Keylogger software to send recorded data to a foreign location. This happens by using one of the following methods:

1. Uploading the info to an internet site, database, or FTP server.
2. Periodically emailing data to a predefined email address.
3. Wirelessly transmitting information through a joined equipment framework.
4. Software empowering far off login to your neighborhood machine.

Detection and removal of Keylogger

There is a way to detect a Keylogger, though none are a catchall, so if you've got a reason to suspect your computer features a Keylogger, we recommend trying a variety of these tactics:

1. Choose the best **Antivirus**, to detect a Keylogger on your system. There is some specific sort of AV dedicated for such scans.
2. Press **Ctrl+Alt+Delete** to check the task list on your computer. Examine the tasks running, and if you're unacquainted any of them, look them abreast of an inquiry engine.
3. Scan your hard disc for the foremost recent files stored. Look at the contents of any files that **often update**, as they could be logs.
4. Use your system configuration utility to look at which programs are loaded at computer start-up. Access this list by typing "**msconfig**" into the run box.

Pros Of Keylogger

1. Monitor Every Keystroke Made.
2. Protect Confidential Information.
3. Safety Concerns.

Cons Of Keylogger

1. Zero Privacy.
2. Release of Sensitive Information.
3. Gives Keylogging Service Providers Free Reign.

Keyloggers Examples:

1. Revealer Keylogger
2. Spyrix Free Keylogger
3. Elite Keylogger for Windows
4. Best Free Keylogger
5. KidLogger

5.5 Spyware

Spyware is a breach of cyber security as they usually get into the laptop/ computer system when a user unintentionally clicks on a random unknown link or opens an unknown attachment, which downloads the spyware alongside the attachment. It is a best practice to be cautious of the sites that are used for downloading content on the system. Spyware is a type of software that unethically without proper permissions or authorization steals a user's personal or business information and sends it to a third party. Spyware may get into a computer or laptop as a hidden component through free or shared wares.

Spywares perform the function of maliciously tracking a user's activity, having access to data, or even resulting in the crashing of the computer/ laptop system. Spyware in many cases runs as a background process and slows down the normal functioning of the computer system.

Spyware enters the laptop/computer system through the below-listed ways:

- **Phishing:** It is a form of a security breach where spyware enters the system when a suspicious link is clicked or an unknown dangerous attachment is downloaded.
- **Spoofing:** It goes alongside phishing and makes the unauthorized emails appear to come from legitimate users or business units.
- **Free Softwares or Shared Softwares:** It gets into the system when a user installs software that is free of cost but has additional spyware added to them.
- **Misleading software:** This is advertised as very beneficial for the system and would boost up the speed of the system but lead to stealing confidential information from the system.

Spyware Entering the Computer System

Spyware entering the system is very dangerous and therefore proper knowledge of them can save a lot of trusted information from being accessible to third-party. Spywares are classified on the basis of the function they perform. There are different types of Spyware, which can attack our system. These are listed as below:

- **Adware:** It is a type of Spyware that keeps track of the user's activity and gives advertisements based on the tracked activity of the user.
- **Tracking Cookies:** It is a type of Spyware that tracks a user's activity and supplies the same to third parties.
- **Trojans:** It is a type of Spyware that is the most dangerous. It aims to steal confidential user information such as bank details, passwords and transfers it to a third party to perform illegal transactions or frauds.
- **Keyloggers:** It is a type of Spyware that keeps a track of all the keystrokes that the user enters through the keyboard. It is dangerous as it contributes bro cyber fraud where sensitive passwords can be stolen by keeping an eye on the user who entered the information.
- **Stalkerware:** It is a type of Spyware that is installed on mobile phones to stalk the user. It tracks the movement of the user and sends the same to the third party.
- **System Monitor:** It is a type of Spyware that monitors and keep a track of the entire system including users activity, sensitive information, keystrokes, calls, and chats. It is extremely dangerous to user privacy.

Spyware Infects Devices

Spyware gets attached to websites and downloads without going much into the notice of the user. There are many software's that get downloaded without any warning alongside the needed software and are very dangerous for our computer system. Another way of spyware, entering our systems is when the user clicks unverified links or downloads malicious contents on the computer system.

When spyware enters the computer system it unethically accesses the information that it is not authorized to view. In most cases, it also supplies this information to third-party users leading to data leaks. Sensitive information such as passwords and bank information are at much risk if spyware enters the computer system. Data leak, stealing of sensitive information, tracking user's activity/ preferences, making the system slow down, and even crashing the computer system are the effects that can be caused when spyware enters the computer system without the user's consent.

Prevention of Spyware

- **Installing Antivirus/ Antispyware:** The best way to protect your system from spyware is to install a good quality Anti-spyware or Antivirus such as MalwareBytes, Adaware, AVG Antivirus, SpywareBlaster, etc. This will help in protecting the computer system in case spyware tries to attach to our system. Installing Antivirus/ Antispyware also protects the system from harmful threats by blocking sites that try to steal data or leak the data to third-party users.
- **Beware of Cookie Settings:** There are some websites that transfer confidential information alongside cookies. It is always advisable to keep a check on the cookie settings and set the settings to high security.
- **Beware of the Pop-ups on Websites:** Don't click on the pop-ups that appear on your website without reading them. Never accept their terms and conditions as it is highly dangerous. Always close the pop-up windows without clicking on 'ok'.
- **Never Install Free Software:** Always be very cautious when you install free software on your systems. Free software mostly has spyware attached to them and it can directly leak confidential user information.
- **Always read Terms & Conditions:** Always read Terms and Conditions before installing apps on your system. Never accept policies that breach privacy. Download only trusted and verified apps from Google PlayStore or Apple PlayStore for mobile phones to protect them from Spyware.

5.6 SQL Injection

The SQL Injection is a code penetration technique that might cause loss to our database. It is one of the most practiced web hacking techniques to place malicious code in SQL statements, via webpage input. SQL injection can be used to manipulate the application's web server by malicious users.

SQL injection generally occurs when we ask a user to input their username/userID. Instead of a name or ID, the user gives us an SQL statement that we will unknowingly run on our database. For Example - we create a SELECT statement by adding a variable "demoUserID" to select a string. The variable will be fetched from user input (getRequestString).

Types of SQL injection attacks

SQL injections can do more harm other than passing the login algorithms. Some of the SQL injection attacks include:

- Updating, deleting, and inserting the data: An attack can modify the cookies to poison a web application's database query.
- It is executing commands on the server that can download and install malicious programs such as Trojans.

- We are exporting valuable data such as credit card details, email, and passwords to the attacker's remote server.
- Getting user login details: It is the simplest form of SQL injection. Web application typically accepts user input through a form, and the front end passes the user input to the back end database for processing.

Example of SQL Injection

We have an application based on employee records. Any employee can view only their own records by entering a unique and private employee ID. We have a field like an Employee ID. And the employee enters the following in the input field:

236893238 or 1=1

It will translate to:

The SQL code above is valid and will return EMPLOYEE_ID row from the EMPLOYEE table. The 1=1 will return all records for which this holds true. All the employee data is compromised; now, the malicious user can also similarly delete the employee records.

Example:

Now the malicious user can use the '=' operator sensibly to retrieve private and secure user information. So instead of the query mentioned above, the following query, when exhausted, retrieve protected data, not intended to be shown to users.

SQL injection based on Batched SQL statements

Several databases support batched SQL statements. It is a group of two or more SQL statements separated by semicolons.

The SQL statement given below will return all rows from the Employee table, then delete the Employee_Add table.

Detection of SQL Injection attacks

Creating a SQL Injection attack is not difficult, but even the best and good-intentioned developers make mistakes. The detection of SQL Injection is, therefore, an essential component of creating the risk of an SQL injection attack. Web Application Firewall can detect and block basic SQL injection attacks, but we should depend on it as the sole preventive measure.

Intrusion Detection System (IDS) is both network-based and host-based. It can be tuned to detect SQL injection attacks. Network-based IDSec can monitor all connections to our database server, and flags suspicious activities. The host-based IDS can monitor web server logs and alert when something strange happens.

Impact of SQL Injection

The intruder can retrieve all the user-data present in the database, such as user details, credit card information, and social security numbers, and can also gain access to protected areas like the administrator portal. It is also possible to delete the user data from the tables. These days all the online shopping applications, bank transactions use back-end database servers. If the intruder can exploit SQL injection, the entire server is compromised.

How to prevent SQL Injection attack

- We should use user authentication to validate input from the user by pre-defining length, input type, and the input field.
- Restricting the access privileges of users and defining the amount of data any outsider can access from the database. Generally, the user cannot be granted permission to access everything in the database.
- We should not use system administrator accounts.

5.7 Network Access Control

Network Access Control is a security solution that uses a set of protocols to keep unauthorized users and devices out of a private network or give restricted access to the devices which are compliant with network security policies. It is also known as **Network Admission Control.** It handles network management and security that implements security policy, compliance, and management of access control to a network.
NAC works on wired and wireless networks by identifying different devices that are connected to the network. For setting up an NAC network security solution, administrators will determine the protocols that will decide how devices and users are authorized for the right level of authorization. Access rules are generally based on the criterion such as device used, the location accessed from, the access rights of various individuals, as well as the specific data and resources being accessed.

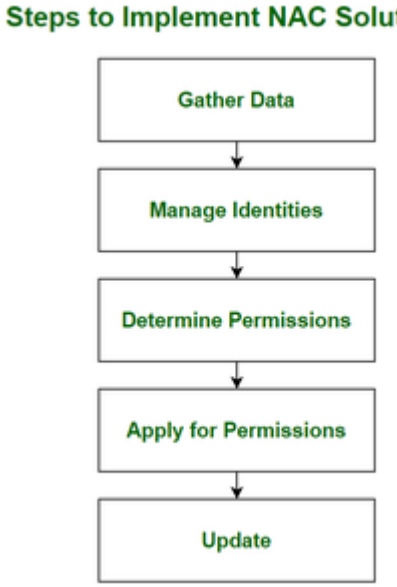
Components of Network Access Control Scheme:

1. **Restricted Access:** It restricts access to the network by user authentication and authorization control. For example, the user can't access a protected network resource without permission to access it.
2. **Network Boundary Protection:** It monitors and controls the connectivity of networks with external networks. It includes tools such as controlled interfaces, intrusion detection, and anti-virus tools. It is also called perimeter defense. For example, the firewall can be used to prevent unauthorized access to network resources from outside of the network.

Types of Network Access Control:

1. **Pre-admission:** It happens before access to the network is granted on initialization of request by user or device to access the network. It evaluates the access attempt and only allows the access if the user or device is compliant with organization security policies and authorized to access the network.
2. **Post-admission:** It happens within the network when the user or device attempts to access the different parts of the network. It restricts the lateral movement of the device within the network by asking for re-authentication for each request to access a different part of the network.

Steps to Implement NAC Solutions:



Implement NAC Solutions

1. **Gather Data:** Perform an exhaustive survey and collect information about every device, user, and server that has to interface with the network resources.
2. **Manage Identities:** Verify user identities within the organization by authentication and authorization.
3. **Determine Permissions:** Create permission policies stating different access levels for identified user groups.
4. **Apply for Permissions:** Apply permission policies on identified user groups and register each user in the NAC system to trace their access level and activity within the network.
5. **Update:** Monitor security operations and make adjustments to permission policies based on changing requirements of the organization with time.

Importance of Network Access Control:

There has been exponential growth in the number of mobile devices accessing private networks of organizations in the past few years. This has led to an increase in security risks

for the organization's resources and therefore, some tools are required that can provide the visibility, access control, and compliance capabilities to strengthen the network security infrastructure.

A NAC system can deny network access to non-compliant devices or give them only restricted access to computing resources, thus preventing insecure nodes from infecting the network. Also, NAC products can handle large enterprise networks that have a large range of different device types connected to the network.

Responsibilities:

1. It allows only compliant, authenticated devices to access network resources and infrastructure.
2. It controls and monitors the activity of connected devices on the network.
3. It restricts the availability of network resources of private organizations to devices that follow their security policy.
4. It regulates the access of network resources to the users.
5. It mitigates network threats by enforcing security policies that block, isolate, and repair non-compliant machines without administrator attention.

Common Use-Cases:

1. Organizations that allow employees to use their own devices or take corporate devices home use NAC to ensure network security.
2. Organizations use NAC to grant access to different network resources to people or devices that are outside of the organization and are subjected to different security controls.
3. NAC protects from threats caused due to use of IoT devices by categorizing IoT devices into groups that have limited permission and constantly monitoring their activities.

Benefits:

1. Users can be required to authenticate via multi-factor authentication, which is much more secure than identifying users based on IP addresses or username and password combinations.
2. It provides additional levels of protection around individual parts of the network.

Limitations:

1. It has low visibility in IoT devices and devices with no specific users associated with it.
2. It does not protect from threats present inside the network.
3. It may not work for organizations if it is not compatible with existing security controls.

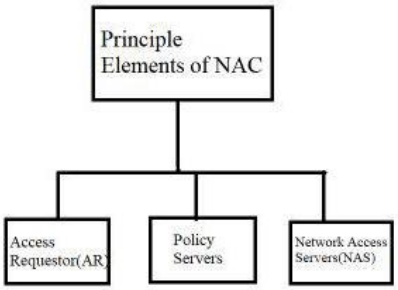
Principle Elements of NAC(Network Access Control):

There are mainly three principle elements of NAC which are:

1.Access Requestor(AR).

2.Policy Servers.

3.Network Access Servers(NAS).



Three Principle Elements of NAC(Network Access Control).

1.Access Requestor(AR): We may determine from the name that it is someone attempting to gain access by requesting it. This access can be granted to any entity, such as a device, person, or process.

This entity attempts to get access to network resources. It might be any device handled by the NAC system, such as servers, cameras, printers, and other IP-enabled devices.

ARs are also known as supplicants or clients at times. ARs ensures that no entity has illegal access to protected resources.

To get access, these ARs must follow to the organization's specific guidelines or policies.

2.Policy Server: The policy server analyzes what access should be provided to AR based on the AR's identity, permission level, attempted request, and an organization's established access policy.

The policy server frequently relies on backend services, such as antivirus, patch management, or a user directory, to function.

The policy server helps to determine the host's state. An organization creates different access policies to clearly authorize or reject such access. If the AR follows the organization's policy, the policy server gives access based on the requestor's permission; otherwise, the AR will not be permitted access based on its permission.

It should be noted that there are various commercial systems on the market now that provide such policy servers for both on-premises and cloud computing. Some of the most common examples include the Cisco Identity Services Engine(ISE), Forescout Platform, Aruba ClearPass Policy Manager, and FortiNAC.

These tools offer highly detailed ways to set organizational rules and control the organization's full IP infrastructure.

3.Network Access Server(NAS): Users connecting to an organization's internal network from distant locations utilize the NAS as an access control point. These often serve as VPNs and give users access to the company's internal network. These days, NAS functionality is frequently included in policy server systems.

Remote employees can connect to the company's internal network via NAS, which serves as an access point for them. This allows the company and its employees to create a secure connection and grant authorized access to the network.

Thus, these were the Three Principle Elements of NAC (Network Access control).

5.8 Cloud Security

Cloud computing refers to the on demand delivery of computing services such as applications, computing resources, storage, database, networking resources etc. through internet and on a pay as per use basis. At the present time the demand for cloud computing services are increasing with respect to that demand for cloud computing skills is also increasing. It provides three main types of service models i.e. SaaS (Software as a Service), PaaS (Platform as a Service) and IaaS (Infrastructure as a Service). With this as starting from small to large organizations have started using cloud services so depending upon their requirement they go for the different types of cloud like Public cloud, Private cloud, Hybrid cloud, Community cloud.

Security In Cloud Computing :

Cloud computing which is one of the most demanding technology of the current time, starting from small to large organizations have started using cloud computing services. Where there are different types of cloud deployment models are available and cloud services are provided as per requirement like that internally and externally security is maintained to keep the cloud system safe. Cloud computing security or cloud security is an important concern which refers to the act of protecting cloud environments, data, information and applications against unauthorized access, DDOS attacks, malwares, hackers and other similar attacks. Community Cloud : These allow to a limited set of organizations or employees to access a shared cloud computing service environment.

Planning of security in Cloud Computing :

As security is a major concern in cloud implementation, so an organization have to plan for security based on some factors like below represents the three main factors on which planning of cloud security depends.

- Resources that can be moved to the cloud and test its sensitivity risk are picked.
- The type of cloud is to be considered.
- The risk in the deployment of the cloud depends on the types of cloud and service models.

Types of Cloud Computing Security Controls :

There are 4 types of cloud computing security controls i.e.

1. **Deterrent Controls** : Deterrent controls are designed to block nefarious attacks on a cloud system. These come in handy when there are insider attackers.
2. **Preventive Controls** : Preventive controls make the system resilient to attacks by eliminating vulnerabilities in it.
3. **Detective Controls** : It identifies and reacts to security threats and control. Some examples of detective control software are Intrusion detection software and network security monitoring tools.
4. **Corrective Controls** : In the event of a security attack these controls are activated. They limit the damage caused by the attack.

Importance of cloud security :

For the organizations making their transition to cloud, cloud security is an essential factor while choosing a cloud provider. The attacks are getting stronger day by day and so the security needs to keep up with it. For this purpose it is essential to pick a cloud provider who offers the best security and is customized with the organization's infrastructure. Cloud security has a lot of benefits –

- **Centralized security :** Centralized security results in centralizing protection. As managing all the devices and endpoints is not an easy task cloud security helps in doing

so. This results in enhancing traffic analysis and web filtering which means less policy and software updates.

- **Reduced costs :** Investing in cloud computing and cloud security results in less expenditure in hardware and also less manpower in administration
- **Reduced Administration :** It makes it easier to administer the organization and does not have manual security configuration and constant security updates.
- **Reliability :** These are very reliable and the cloud can be accessed from anywhere with any device with proper authorization.

When we are thinking about cloud security it includes various types of security like access control for authorized access, network segmentation for maintaining isolated data, encryption for encoded data transfer, vulnerability check for patching vulnerable areas, security monitoring for keeping eye on various security attacks and disaster recovery for backup and recovery during data loss.

There are different types of security techniques which are implemented to make the cloud computing system more secure such as SSL (Secure Socket Layer) Encryption, Multi Tenancy based Access Control, Intrusion Detection System, firewalls, penetration testing, tokenization, VPN (Virtual Private Networks), and avoiding public internet connections and many more techniques.

But the thing is not so simple how we think, even implementation of number of security techniques there is always security issues are involved for the cloud system. As cloud system is managed and accessed over internet so a lot of challenges arises during maintaining a secure cloud. Some cloud security challenges are

- Control over cloud data
- Misconfiguration
- Ever changing workload
- Access Management
- Disaster recovery

5.9 Web Security

Web Security is very important nowadays. Websites are always prone to security threats/risks. Web Security deals with the security of data over the internet/network or web or while it is being transferred to the internet. For e.g. when you are transferring data between client and server and you have to protect that data that security of data is your web security.

Hacking a Website may result in the theft of Important Customer Data, it may be the credit card information or the login details of a customer or it can be the destruction of one's business and propagation of illegal content to the users while somebody hacks your website they can either steal the important information of the customers or they can even propagate the illegal content to your users through your website so, therefore, security considerations are needed in the context of web security.

Security Threats:

A Threat is nothing but a possible event that can damage and harm an information system. Security Threat is defined as a risk that which, can potentially harm Computer systems & organizations. Whenever an Individual or an Organization creates a website, they are vulnerable to security attacks.

Security attacks are mainly aimed at stealing altering or destroying a piece of personal and confidential information, stealing the hard drive space, and illegally accessing passwords. So whenever the website you created is vulnerable to security attacks then the attacks are going to steal your data alter your data destroy your personal information see your confidential information and also it accessing your password.

Top Web Security Threats :

Web security threats are constantly emerging and evolving, but many threats consistently appear at the top of the list of web security threats. These include:

- Cross-site scripting (XSS)
- SQL Injection
- Phishing
- Ransomware
- Code Injection
- Viruses and worms
- Spyware
- Denial of Service

Security Consideration:

- **Updated Software:** You need to always update your software. Hackers may be aware of vulnerabilities in certain software, which are sometimes caused by bugs and can be used to damage your computer system and steal personal data. Older versions of software can become a gateway for hackers to enter your network. Software makers soon become aware of these vulnerabilities and will fix vulnerable or exposed areas. That's why It is mandatory to keep your software updated, It plays an important role in keeping your personal data secure.
- **Beware of SQL Injection:** SQL Injection is an attempt to manipulate your data or your database by inserting a rough code into your query. For e.g. somebody can send a query to your website and this query can be a rough code while it gets executed it can be used to manipulate your database such as change tables, modify or delete data or it can retrieve important information also so, one should be aware of the SQL injection attack.
- **Cross-Site Scripting (XSS):** XSS allows the attackers to insert client-side script into web pages. E.g. Submission of forms. It is a term used to describe a class of attacks that allow an attacker to inject client-side scripts into other users' browsers through a website. As the injected code enters the browser from the site, the code is reliable and can do things like sending the user's site authorization cookie to the attacker.
- **Error Messages:** You need to be very careful about error messages which are generated to give the information to the users while users access the website and some error messages are generated due to one or another reason and you should be very careful while providing the information to the users. For e.g. login attempt – If the user fails to login the error message should not let the user know which field is incorrect: Username or Password.
- **Data Validation:** Data validation is the proper testing of any input supplied by the user or application. It prevents improperly created data from entering the information system. Validation of data should be performed on both server-side and client-side. If we perform data validation on both sides that will give us the authentication. Data validation should occur when data is received from an outside party, especially if the data is from untrusted sources.

- **Password:** Password provides the first line of defense against unauthorized access to your device and personal information. It is necessary to use a strong password. Hackers in many cases use sophisticated software that uses brute force to crack passwords. Passwords must be complex to protect against brute force. It is good to enforce password requirements such as a minimum of eight characters long must including uppercase letters, lowercase letters, special characters, and numerals.

5.10 Wireless security

Wireless Network provides various comfort to end users but actually they are very complex in their working. There are many protocols and technologies working behind to provide a stable connection to users. Data packets traveling through wire provide a sense of security to users as data traveling through wire probably not heard by eavesdroppers.

To secure the wireless connection, we should **focus on** the following areas –
- Identify endpoint of wireless network and end-users i.e., Authentication.
- Protecting wireless data packets from middleman i.e., Privacy.
- Keeping the wireless data packets intact i.e., Integrity.

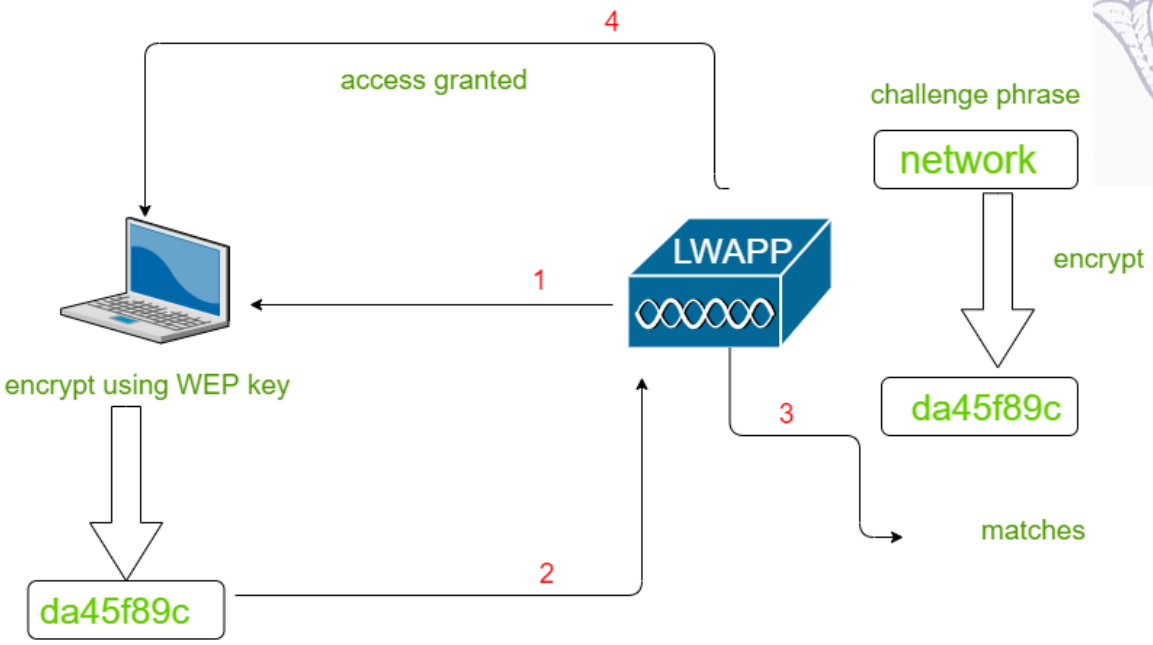
We know that wireless clients form an association with Access Points (AP) and transmit data back and forth over the air. As long as all wireless devices follow 802.11 standards, they all coexist. But all wireless devices are not friendly and trustworthy, some rogue devices may be a threat to wireless security. Rogue devices can steal our important data or can cause the unavailability of the network.

Wireless security is **ensured by** following methods-
- Authentication
- Privacy and Integrity

In this article, we talk about Authentication. There are broadly two types of Authentication process: Wired Equivalent Privacy (WEP), and Extensible Authentication Protocol (802.1x/EAP).
These are explained as following below.

1. Wired Equivalent Privacy (WEP) :

For wireless data transmitting over the air, open authentication provides no security. WEP uses the RC4 cipher algorithm for making every frame encrypted. The RC4 cipher also encrypts data at the sender side and decrypt data at the receiving site, using a string of bits as key called WEP key.
WEP key can be used as an authentication method or encryption tool. A client can associate with AP only if it has the correct WEP key. AP tests the knowledge of the WEP key by using a challenge phrase. The client encrypts the phrase with his own key and send back to AP. AP compares the received encrypted frame with his own encrypted phrase. If both matches, access to the association is granted.



Working of WEP Authentication

2. Extensible Authentication Protocol (802.1x/EAP) :

In WEP authentication, authentication of the wireless clients takes place locally at AP. But Scenario gets changed with 802.1x. A dedicated authentication server is added to the infrastructure. There is the participation of three devices –

1. **Supplicant** –
   Device requesting access.
2. **Authenticator** –
   Device that provides access to network usually a Wlan controller (WLC).
3. **Authentication Server** –
   Device that takes client credentials and deny or grant access.