COURSE MATERIAL

CRYPTOGRAPHY AND NETWORK SECURITY

for

IV YEAR / VII SEMESTER

ACADEMIC YEAR 2020-21

BY

DR.K. ANITHA

ASSISTANT PROFESSOR

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

SRI CHANDRASEKARENDRA VISWA MAHA VIDYALAYA
(SCSVMV)
(Deemed to be university u/s 3 of the UGC Act 1956)
(Accredited ‘A’ Grade by NAAC)
ENATHUR, KANCHIPURAM – 631 561.

CRYPTOGRAPHY AND NETWORK SECURITY

L	T	P	C
3	1	0	3

Pre- requisite: Basic knowledge of Digital Communication

OBJECTIVES:

- ❖ To understand OSI security architecture and classical encryption techniques.
- ❖ To acquire fundamental knowledge on the concepts of finite fields and number theory.
- ❖ To understand various block cipher and stream cipher models.
- ❖ To describe the principles of public key cryptosystems, hash functions and digital signature.

UNIT I INTRODUCTION & NUMBER THEORY

Services, Mechanisms and attacks-the OSI security architecture-Network security model-Classical Encryption techniques (Symmetric cipher model, substitution techniques, transposition techniques, steganography).FINITE FIELDS AND NUMBER THEORY: Groups, Rings, Fields-Modular arithmetic- Euclid"s algorithm-Finite fields- Polynomial Arithmetic –Prime numbers-Fermat"s and Euler"s theorem- Testing for primality -The Chinese remainder theorem- Discrete logarithms.

UNIT II BLOCK CIPHERS & PUBLIC KEY CRYPTOGRAPHY

Data Encryption Standard-Block cipher principles-block cipher modes of operation-Advanced Encryption Standard (AES)-Triple DES-Blowfish-RC5 algorithm. **Public key cryptography:** Principles of public key cryptosystems-The RSA algorithm-Key management - Diffie Hellman Key exchange- Elliptic curve arithmetic-Elliptic curve cryptography.

UNIT III HASH FUNCTIONS AND DIGITAL SIGNATURES

Authentication requirement – Authentication function – MAC – Hash function – Security of hash function and MAC – MD5 - SHA - HMAC – CMAC - Digital signature and authentication protocols – DSS – EI Gamal – Schnorr.

UNIT IV SECURITY PRACTICE & SYSTEM SECURITY

Authentication applications – Kerberos – X.509 Authentication services - Internet Firewalls for Trusted System: Roles of Firewalls – Firewall related terminology- Types of Firewalls - Firewall designs - SET for E-Commerce Transactions. Intruder – Intrusion detection system – Virus and related threats – Countermeasures – Firewalls design principles – Trusted systems – Practical implementation of cryptography and security.

UNIT V E-MAIL, IP & WEB SECURITY

E-mail Security: Security Services for E-mail-attacks possible through E-mail - establishing keys privacy-authentication of the source-Message Integrity-Non-repudiation-Pretty Good Privacy- S/MIME.

IPSecurity: Overview of IPsec - IP and IPv6-Authentication Header-Encapsulation Security Payload (ESP)- Internet Key Exchange (Phases of IKE, ISAKMP/IKE Encoding). **Web Security:** SSL/TLS Basic Protocol- computing the keys- client authentication-PKI as deployed by SSL Attacks fixed in v3- Exportability-Encoding-Secure Electronic Transaction (SET).

OUTCOMES:

At the end of the course, the student should be able to:

- ❖ Design firewalls and intrusion detection system
- ❖ Design security services for E-mail

TEXT BOOKS:

1. William Stallings, Cryptography and Network Security, 6th Edition, Pearson Education, March 2013. (UNIT I, II, III, IV). ACC.NO: B101366
2. Charlie Kaufman, RadiaPerlman and Mike Speciner,"Network Security", Prentice Hall of India, 2002. (UNIT V).

REFERENCES:

1. Behrouz A. Ferouzan, "Cryptography & Network Security", Tata Mc Graw Hill, 2007.ACC.NO: B129837
2. Man Young Rhee, "Internet Security: Cryptographic Principles", "Algorithms and Protocols", Wiley Publications, 2003.
3. Charles Pfleeger, "Security in Computing", 4th Edition, Prentice Hall of India, 2006.ACC.NO: B128188
4. Ulysses Black, "Internet Security Protocols", Pearson Education Asia, 2000. ACC.NO: B64002
5. Charlie Kaufman and Radia Perlman, Mike Speciner, "Network Security, Second Edition, Private Communication in Public World", PHI 2002.
6. Bruce Schneier and Neils Ferguson, "Practical Cryptography", First Edition, Wiley Dreamtech India Pvt Ltd, 2003.
7. Douglas R Simson "Cryptography – Theory and practice", First Edition, CRC Press, 1995

UNIT I - INTRODUCTION & NUMBER THEORY

INTRODUCTION:

Computer security, **cybersecurity** or **information technology security** (**IT security**) is the protection of computer systems and networks from the theft of or damage to their hardware, software, or electronic data, as well as from the disruption or misdirection of the services they provide.

Computer and network security is essentially a battle of wits between a culprit who tries to find holes and the designer or administrator who tries to close them.

Computer security is a series of protocols that a company or an individual follows to ensure information maintains its "ICA" – integrity, confidentiality and availability.

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.

CRYPTOSYSTEM

A cryptosystem is an implementation of cryptographic techniques and their accompanying infrastructure to provide information security services. A cryptosystem is also referred to as a cipher system.

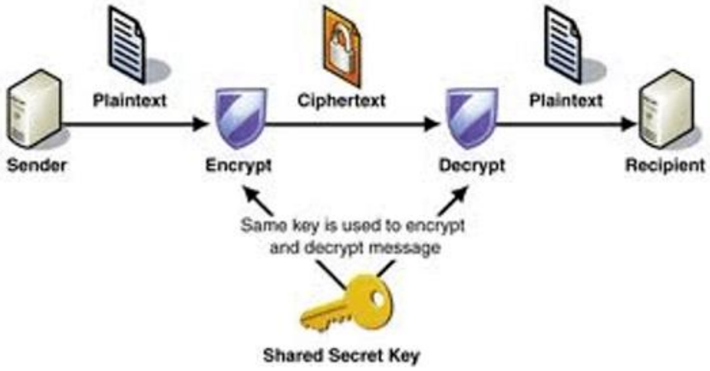


Figure.1. Cryptosystem

Cryptosystem shown in fig.1, is th study of secure communications techniques that allow only the sender and intended recipient of a message to view its contents. When transmitting electronic data, the most common use of **cryptography** is to encrypt and decrypt email and other plain-text messages. It reformats 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.

Network Security - measures to protect data during their transmission

Internet Security - measures to protect data during their transmission over a collection of interconnected networks

Computer Security: The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability, and confidentiality of information system resources (includes hardware, software, firmware, information/data, and telecommunications).

This definition introduces three key objectives that are at the heart of computer security:

Confidentiality (C): Preserving authorized restrictions on information access and disclosure, including means for protecting personal privacy and proprietary information. A loss of confidentiality is the unauthorized disclosure of information.

Integrity(I): Guarding against improper information modification or destruction, including ensuring information nonrepudiation and authenticity. A loss of integrity is the unauthorized modification or destruction of information.

Availability (A): Ensuring timely and reliable access to and use of information. A loss of availability is the disruption of access to or use of information or an information system.

These three concepts form what is often referred to as the **CIA triad**. The three concepts embody the fundamental security objectives for both data and for information and computing services.

Confidentiality: This term covers two related concepts:

Data1 **confidentiality:** Assures that private or confidential information is not made available or disclosed to unauthorized individuals.

Privacy: Assures that individual’s control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed.

Integrity: This term covers two related concepts:

Data integrity: Assures that information and programs are changed only in a specified and authorized manner.

System integrity: Assures that a system performs its intended function in an unimpaired manner, free from deliberate or inadvertent unauthorized manipulation of the system.

Availability: Assures that systems work promptly and service is not denied to authorized users. Although the use of the CIA triad to define security objectives is well established,
some in the security field feel that additional concepts are needed to present a complete picture. Two of the most commonly mentioned are as follows:

Authenticity: The property of being genuine and being able to be verified and trusted; confidence in the validity of a transmission, a message, or message originator. This means verifying that users are who they say they are and that each input arriving at the system came from a trusted source.

Accountability: The security goal that generates the requirement for actions of an entity to be traced uniquely to that entity. This supports nonrepudiation, deterrence, fault isolation, intrusion detection and prevention, and after-action recovery and legal action.

THE OSI SECURITY ARCHITECTURE:

To assess effectively the security needs of an organization and to evaluate and choose various security products and policies, the manager responsible for security needs, some systematic way of defining the requirements for security and characterizing the approaches to satisfying those requirements. The OSI security architecture was developed in the context of the OSI protocol architecture by ITU-T.

ITU-T: The International Telecommunication Union (ITU) Telecommunication Standardization Sector (ITU-T) is a United Nations sponsored agency that develops standards, called Recommendations, relating to telecommunications and to open systems interconnection (OSI).

Recommendation X.800, Security Architecture for OSI, defines a systematic approach. The OSI security architecture is useful to managers as a way of organizing the task of providing security.

The OSI security architecture focuses on security attacks, mechanisms, and services. These can be defined
briefly as

- **Security attack**
- **Security mechanism**.
- **Security service**

threat and attack are commonly used to mean more or less the same thing. The definitions taken from RFC 4949, Internet Security Glossary.

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 attack 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.

SECURITY ATTACK:

Any action that compromises the security of information owned by an organization. There are four general categories of attack which are listed below.

Interruption

An asset of the system is destroyed or becomes unavailable or unusable. This is an attack on availability.

e.g., destruction of piece of hardware, cutting of a communication line or disabling of file management system.

Sender
Receive

Figure.2a

Interception

An unauthorized party gains access to an asset. This is an attack on confidentiality. Unauthorized party could be a person, a program or a computer. e.g., wiretapping to capture data in the network, illicit copying of files.

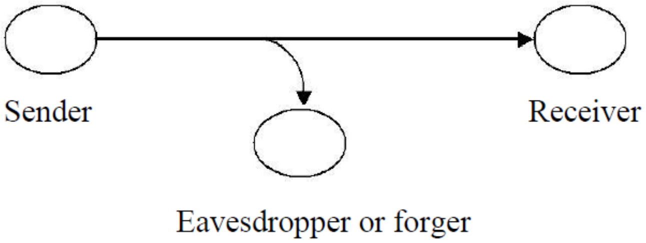


Figure. 2b

Modification

An unauthorized party not only gains access to but tampers with an asset. This is an attack on integrity.

e.g., changing values in data file, altering a program, modifying the contents of messages being transmitted in a network.

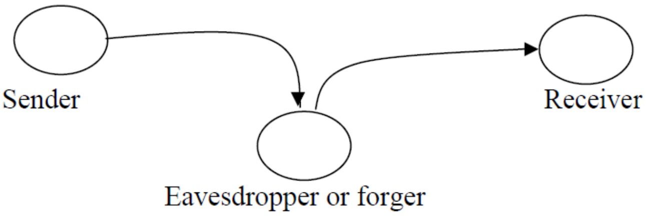


Figure. 2c

Fabrication

An unauthorized party inserts counterfeit objects into the system. This is an attack on authenticity.

e.g., insertion of spurious message in a network or addition of records to a file.

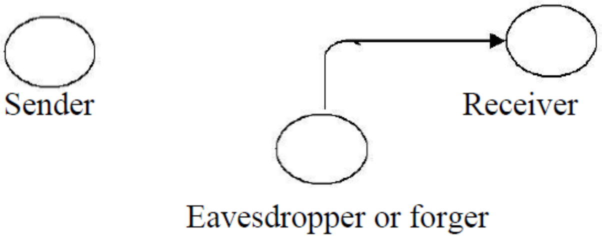


Figure.2d

The attack is majorly classified into two types:

- Active attack
- Passive Attack

PASSIVE ATTACK:

Passive attacks (Fig.3) are in the nature of eavesdropping on, or monitoring of, transmissions. The goal of the opponent is to obtain information that is being transmitted.

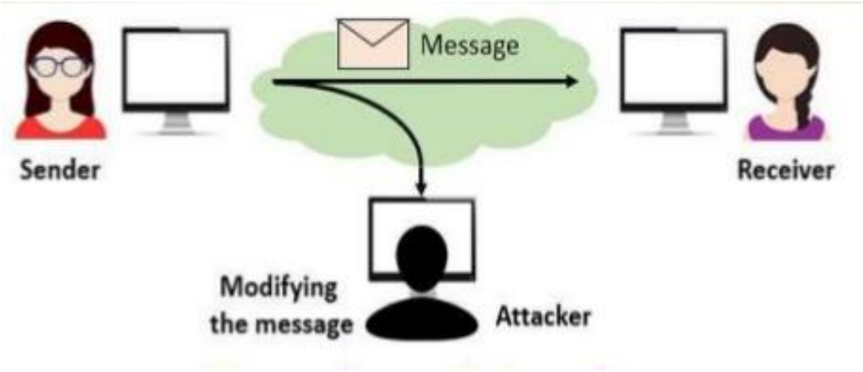


Figure.3

Passive attacks are of two types:

Release of message contents: A telephone conversation, an e-mail message and a transferred file may contain sensitive or confidential information. We would like to prevent the opponent from learning the contents of these transmissions.

Traffic analysis: If we had encryption protection in place, an opponent might still be able to observe the pattern of the message. The opponent could determine the location and identity of communication hosts and could observe the frequency and length of messages being exchanged. This information might be useful in guessing the nature of communication that was taking place.

Passive attacks are very difficult to detect because they do not involve any alteration of data. However, it is feasible to prevent the success of these attacks.

ACTIVE ATTACKS:

These attacks involve some modification of the data stream or the creation of a false stream.

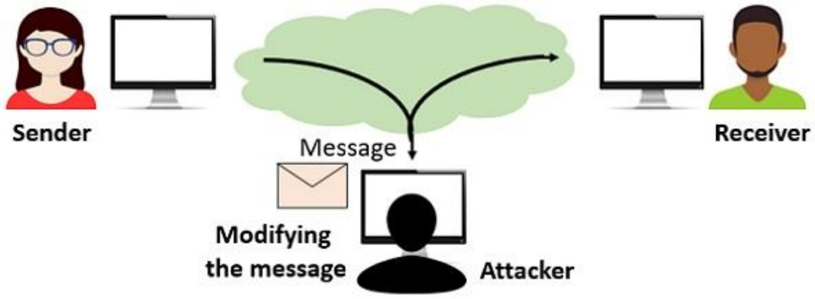


Figure.4

These attacks can be classified in to four categories:

Masquerade – One entity pretends to be a different entity.

Replay – involves passive capture of a data unit and its subsequent transmission to produce an unauthorized effect.

Modification of messages – Some portion of message is altered or the messages are delayed or recorded, to produce an unauthorized effect.

Denial of service – Prevents or inhibits the normal use or management of communication facilities. Another form of service denial is the disruption of an entire network, either by disabling the network or overloading it with messages so as to degrade performance. It is quite difficult to prevent active attacks absolutely, because to do so would require physical protection of all communication facilities and paths at all times. Instead, the goal is to detect them and to recover from any disruption or delays caused by them.

Security mechanism: A process (or a device incorporating such a process) that is designed to detect, prevent, or recover from a security attack.

Cryptanalysis: Cryptanalysis is the study of methods for obtaining the meaning of encrypted information, without access to the secret information that is typically required to do so. Typically, this involves knowing how the system works and finding a secret key. Cryptanalysis is also referred to as codebreaking or cracking the code.

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.

SECURITY SERVICE: A processing or communication service that enhances the security of the data processing systems and the information transfers of an organization. The services are intended to counter security attacks, and they make use of one or more security mechanisms to provide the service.

X.800 defines a security service as a service that is provided by a protocol layer of communicating open systems and that ensures adequate security of the systems or of data transfers. X.800 divides these services into five categories

Confidentiality: Ensures that the information in a computer system and transmitted information are accessible only for reading by authorized parties.

Eg., printing, displaying and other forms of disclosure.

Authentication: Ensures that the origin of a message or electronic document is correctly identified, with an assurance that the identity is not false.

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.

Non repudiation: Requires that neither the sender nor the receiver of a message be able to deny the transmission.

Access control: Requires that access to information resources may be controlled by or the target system.

Availability: Requires that computer system assets be available to authorized parties when needed.

AUTHENTICATION:

The authentication service is concerned with assuring that a communication is Authentic, the function of the authentication service is to assure the recipient that the message is from the source that it claims to be from. In the case of an ongoing interaction, such as the connection of a terminal to a host, two aspects are involved. Two specific authentication services are defined in X.800:

Peer Entity Authentication

Used in association with a logical connection to provide confidence in the identity of the entities connected.

Data Origin Authentication

In a connectionless transfer, provides assurance that the source of received data is as claimed.

ACCESS CONTROL

The prevention of unauthorized use of a resource (i.e., this service controls who can have access to a resource, under what conditions access can occur, and what those accessing the resource is allowed to do).

DATA CONFIDENTIALITY

The protection of data from unauthorized disclosure. Confidentiality is the protection of transmitted data from passive attacks. With respect to the content of a data transmission, several levels of protection can be identified.

Connection Confidentiality: The protection of all user data on a connection.

Connectionless Confidentiality: The protection of all user data in a single data block

AUTHENTICATION

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

The assurance that data received are exactly as sent by an authorized entity (i.e., contain no modification, insertion, deletion, or replay).

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: As above, but 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

Provides protection against denial by one of the entities involved in a communication of having participated in all or part of the communication.

Nonrepudiation, Origin: Proof that the message was sent by the specified party.

Nonrepudiation, Destination: Proof that the message was received by the specified party.

SECURITY MECHANISMS

One of the most specific security mechanisms in use is cryptographic techniques. Encryption or encryption-like transformations of information are the most common means of providing security.

SPECIFIC SECURITY MECHANISMS

May be incorporated into the appropriate protocol layer in order to provide some of the OSI security services.

Encipherment: The use of mathematical algorithms to transform data into a form that is not readily intelligible. The transformation and subsequent recovery of the data depend on an algorithm and zero or more encryption keys.

Digital Signature: Data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery (e.g., by the recipient).

Access Control: A variety of mechanisms that enforce access rights to resources

Data Integrity: A variety of mechanisms used to assure the integrity of a data unit or stream of data units

Authentication Exchange: A mechanism intended to ensure the identity of an entity by means of information exchange.

Traffic Padding: The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts.

Routing Control: Enables selection of particular physically secure routes for certain data and allows routing changes, especially when a breach of security is suspected.

Notarization: The use of a trusted third party to assure certain properties of a data exchange.

PERVASIVE SECURITY MECHANISMS

Mechanisms that are not specific to any particular OSI security service or protocol layer.

Trusted Functionality: That which is perceived to be correct with respect to some criteria (e.g., as established by a security policy).

Security Label: The marking bound to a resource (which may be a data unit) that names or designates the security attributes of that resource.

Event Detection: Detection of security-relevant events.

Security Audit Trail: Data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activities.

Security Recovery: Deals with requests from mechanisms, such as event handling and management functions, and takes recovery actions.

NETWORK SECURITY MODEL:

A model for a network security is shown in the below figure. 5

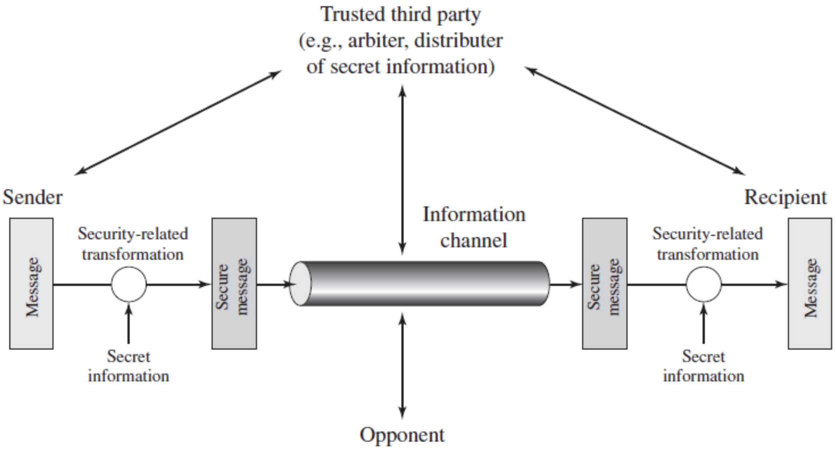


Figure.5 Network Security Model

A message is to be transferred from one party to another across some sort of Internet service. The two parties, who are the *principals* in this transaction, must cooperate for the exchange to take place. A logical information channel is established 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.

Security aspects come into play when it is necessary or desirable to protect the information transmission from an opponent who may present a threat to confidentiality, authenticity, and so on. All the techniques for providing security have two components:

- A security-related transformation on the information to be sent. Examples include the encryption of the message, which scrambles the message so that it is unreadable by the opponent, and the addition of a code based on the contents of the message, which can be used to verify the identity of the sender.
- Some secret information shared by the two principals and, it is hoped, unknown to the opponent. An example is an encryption key used in conjunction with the transformation to scramble the message before transmission and unscramble it on reception.

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 opponent. Or a third party may be needed to arbitrate disputes between the two principals concerning the authenticity of a message transmission.

This general model shows that there are four basic tasks in designing a particular security service:

1. Design an algorithm for performing the security-related transformation. The algorithm should be such that an opponent cannot defeat its purpose.
2. Generate the secret information to be used with the algorithm.
3. Develop methods for the distribution and sharing of the secret information.
4. Specify a protocol to be used by the two principals that makes use of the security algorithm and the secret information to achieve a particular security service.

However, there are other security-related situations of interest that do not neatly fit this model but are considered. A general model of these other situations is illustrated in Figure.6 which reflects a concern for protecting an information system from unwanted access.

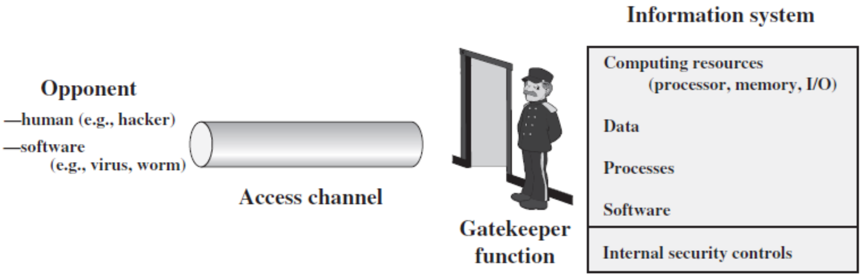


Figure.6 Network Access Security Model

Another type of unwanted access is the placement in a computer system of logic that exploits vulnerabilities in the system and that can affect application programs as well as utility programs, such as editors and compilers. Programs can present two kinds of threats:

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.

Classical Encryption Techniques: A SYMMETRIC CIPHER MODEL:

Symmetric encryption, also referred to as conventional encryption or single-key encryption, was the only type of encryption in use prior to the development of public key encryption in the 1970s.

Some basic terminologies used:

- **ciphertext** - the coded message
- **cipher** - algorithm for transforming plaintext to ciphertext
- **key** - info used in cipher known only to sender/receiver
- **encipher (encrypt)** - converting plaintext to ciphertext
- **decipher (decrypt)** - recovering ciphertext from plaintext
- **cryptography** - study of encryption principles/methods
- **cryptanalysis (codebreaking)** - the study of principles/ methods of deciphering ciphertext *without* knowing key
- **cryptology** - the field of both cryptography and cryptanalysis

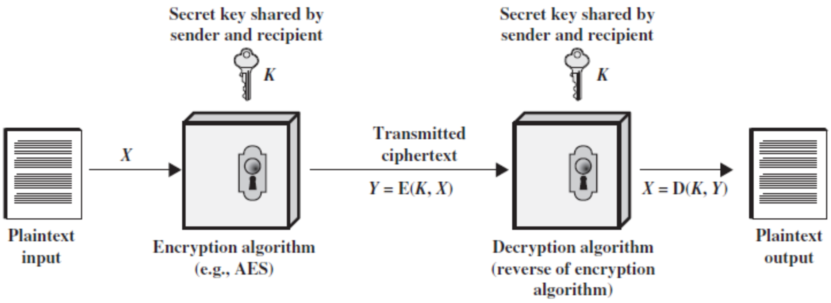

Secret key shared by sender and recipient
K
Secret key shared by sender and recipient
K
X
Transmitted ciphertext
Y = E(K, X)
X = D(K, Y)
Plaintext input
Encryption algorithm (e.g., AES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Fig.7 Simplified Model of Symmetric Encryption

A symmetric encryption scheme has five ingredients

A symmetric encryption scheme has five ingredients (Fig.7). Here the original message, referred to as plaintext, is converted into apparently random nonsense, referred to as cipher text. The encryption process consists of an algorithm and a key.

The key is a value independent of the plaintext. Changing the key changes, the output of the algorithm. Once the cipher text is produced, it may be transmitted. Upon reception, the cipher text can be transformed back to the original plaintext by using a decryption algorithm and the same key that was used for encryption.

The security depends on several factors. First, the encryption algorithm must be powerful enough that it is impractical to decrypt a message on the basis of cipher text alone. Beyond that, the security depends on the secrecy of the key, not the secrecy of the algorithm.

Two requirements for secure use of symmetric encryption:

- A strong encryption algorithm
- A secret key known only to sender / receiver
- $Y = \mathrm{E}K(X)$
- $X = \mathrm{D}K(Y)$

assume encryption algorithm is known implies a secure channel to distribute key

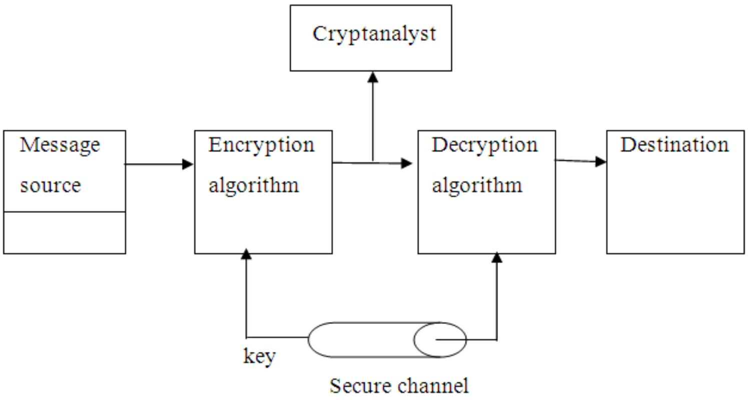


Fig.8. conventional cryptosystem

A source produces a message in plaintext, X = [X1, X2, …, XM] where M are the number of letters in the message. A key of the form K = [K1, K2, …, KJ] is generated. If the key is generated at the source, then it must be provided to the destination by means of some secure channel. With the message X and the encryption key K as input, the encryption algorithm forms

the cipher text Y = [Y1, Y2, …, YN]. This can be expressed as Y = EK(X)

The intended receiver, in possession of the key, is able to invert the transformation: X = DK(Y) An opponent, observing Y but not having access to K or X, may attempt to recover X or K or both. It is assumed that the opponent knows the encryption and decryption algorithms. If the opponent is interested in only this particular message, then the focus of effort is to recover X by generating a plaintext estimate. Often if the opponent is interested in being able to read future messages as well, in which case an attempt is made to recover K by generating an estimate.

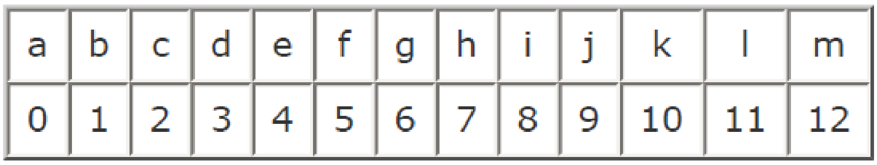
Substitution Encryption Techniques:

Substitution encryption technique is one type of classic encryption technique, 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 ciphertext bit patterns.

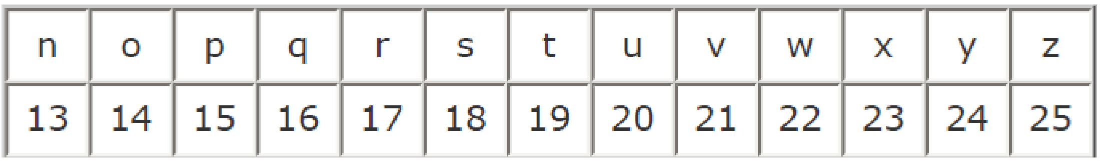
- **(i)Caesar cipher (or) shift cipher**
- The earliest known use of a substitution cipher and the simplest was by Julius Caesar.
- The Caesar Cipher is a type of **shift cipher**. Shift Ciphers work by using the modulo operator to encrypt and decrypt messages. The Shift Cipher has a **key K**, which is an **integer from 0 to 25**. We will only share this key with people that we want to see our message
- The Caesar cipher involves replacing each letter of the alphabet with the letter standing 3 places further down the alphabet.
- e.g., Plain text: pay more mone Cipher text: SDB PRUH PRQHB
- Note that the alphabet is wrapped around, so that letter following „z‟ is „a‟.
- Note that the alphabet is wrapped around, so that the letter following Z is A.
- We can define the transformation by listing all possibilities, as follows:

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

- Let us assign a numerical equivalent to each letter:



a	b	c	d	e	f	g	h	i	j	k	l	m
0	1	2	3	4	5	6	7	8	9	10	11	12



n	o	p	q	r	s	t	u	v	w	x	y	z
13	14	15	16	17	18	19	20	21	22	23	24	25

For Encrypt each plaintext letter p, substitute the cipher text letter c such that

C = E(p) = (p+3) mod 26,

a shift may be any amount, so that general Caesar algorithm is

C = E (p) = (p+k) mod 26

where k takes on a value in the range 1 to 25.

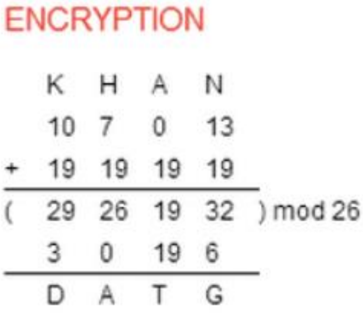
The decryption algorithm is simply P = D(C) = (C-k) mod 26 (or) to Encrypt a message M. Convert the letter into the number that matches its order in the alphabet starting from 0, and call this number X, (A=0, B=1, C=2, ..., Y=24, Z=25).

Calcúlate: Y = (X + K) mod 26

Convert the number Y into a letter that matches its order in the alphabet starting from 0.

Example:

By using the Shift Cipher with **key K=19** for our message. We encrypt the message **"KHAN",** as follows



ENCRYPTION

```
     K    H    A    N
     10   7    0    13
+    19   19   19   19
(    29   26   19   32  ) mod 26
     3    0    19   6
     D    A    T    G
```

- So, after applying the Shift Cipher with key K=19 our message text "**KHAN**" gave us **cipher text** "**DATG**".
- For every letter in the cipher text **C, convert** the letter into the number that matches its order in the alphabet starting from 0, and call this number Y.
- If it is known that a given ciphertext is a Caesar cipher, then a brute-force cryptanalysis is easily performed: Simply try all the 25 possible keys.

Monoalphabetic Ciphers:

With only 25 possible keys, the Caesar cipher is far from secure. A dramatic increase in the key space can be achieved by allowing an arbitrary substitution. Before proceeding, the term *permutation can be defined*.

A permutation of a finite set of elements *S* is an ordered sequence of all the elements of *S*, with each element appearing exactly once.

For example, if *S* = {a, b, c}, there are six permutations of *S*:

abc, acb, bac, bca, cab, cba

In general, there are *n*! permutations of a set of *n* elements, because the first element can be chosen in one of *n* ways, the second in *n* - 1 ways, the third in *n* – 2 ways, and so on.

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

If, instead, the "cipher" line can be any permutation of the 26 alphabetic characters, then there are 26! or greater than 4 * 1026 possible keys.

This is 10 orders of magnitude greater than the key space for DES and would seem to eliminate brute-force techniques for cryptanalysis. Such an approach is referred to as a mono alphabetic substitution cipher, because a single cipher alphabet (mapping from plain alphabet to cipher alphabet) is used per message.

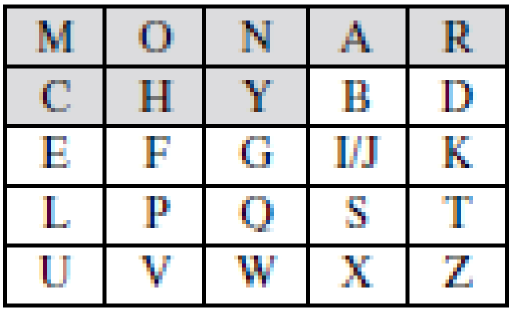
Monoalphabetic ciphers are easy to break because they reflect the frequency data of the original alphabet.

A countermeasure is to provide multiple substitutes known as homophones, for a single letter. For example, the letter e could be assigned a number of different cipher symbols, such as 16, 74, 35, and 21, with each homophone assigned to a letter in rotation or randomly.

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

he Playfair algorithm is based on the use of a 5 * 5 matrix of letters constructed using a keyword. Here is an example, solved by Lord Peter Wimsey in Dorothy Sayers‟s Have His Carcase



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

In this case, the keyword is *monarchy*. The matrix is constructed by filling in the letters of the keyword (minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetic order. The letters I and J count as one letter.

Plaintext is encrypted two letters at a time, according to the following rules:

Repeating plaintext letters that are in the same pair are separated with a filler letter, such as x, so that balloon would be treated as ba lx lo on.

Two plaintext letters that fall in the same row of the matrix are each replaced by the letter to the right, with the first element of the row circularly following the last. For example, ar is encrypted as RM.

Two plaintext letters that fall in the same column are each replaced by the letter beneath, with the top element of the column circularly following the last. For example, mu is encrypted as CM.

Otherwise, each plaintext letter in a pair is replaced by the letter that lies in its own row and the column occupied by the other plaintext letter. Thus, hs becomes BP and ea becomes IM

The Playfair cipher is a great advance over simple monoalphabetic ciphers. For one thing, whereas there are only 26 letters, there are 26 * 26 = 676 digrams, so that identification of individual digrams is more difficult. Furthermore, the relative frequencies of individual letters exhibit a much greater range than that of digrams, making frequency analysis much more difficult.

For these reasons, the Playfair cipher was for a long time considered unbreakable. It was used as the standard field system by the British Army in World War I and still enjoyed considerable use by the U.S. Army and other Allied forces during World War II.

Hill Cipher:

Another interesting multiletter cipher is the Hill cipher, developed by the mathematician Lester Hill in 1929.

The Hill Algorithm

This encryption algorithm takes *m* successive plaintext letters and substitutes for them *m* ciphertext letters. The substitution is determined by *m* linear equations in which each character is assigned a numerical value (a = 0, b = 1,….., z = 25). For *m* = 3, the system can be described as

$c_1 = (k_{11}p_1 + k_{21}p_2 + k_{31}p_3) \bmod 26$ $c_2 = (k_{12}p_1 + k_{22}p_2 + k_{32}p_3) \bmod 26$ $c_3 = (k_{13}p_1 + k_{23}p_2 + k_{33}p_3) \bmod 26$

This can be expressed in terms of row vectors and matrices:

$$\boxed{c_1c_2c_3 \;=\; p_1p_2p_3 \begin{array}{ccc} k_{11} & k_{12} & k_{13} \\ & k_{22} & k_{23}\, mod\ 26 \\ k_{21} & & \\ k_{31} & k_{32} & k_{33} \end{array}}$$

or

C = **PK** mod 26

Where **C** and **P** are row vectors of length 3 representing the plaintext and ciphertext, and **K** is a 3 x 3 matrix representing the encryption key. Operations are performed mod 26.

Polyalphabetic ciphers

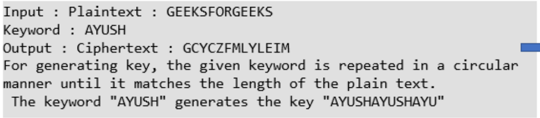
A **polyalphabetic cipher** is any **cipher** based on substitution, using multiple substitution alphabets. The Vigenère **cipher** is probably the best-known example of a **polyalphabetic cipher**.

Difference between monoalphabetic cipher and polyalphabetic cipher:

A **monoalphabetic cipher** is a substitution **cipher** in which the **cipher** alphabet is fixed through the encryption process. ... A **polyalphabetic cipher** is a substitution **cipher** in which the **cipher** alphabet changes during the encryption process.

Vigenere cipher:

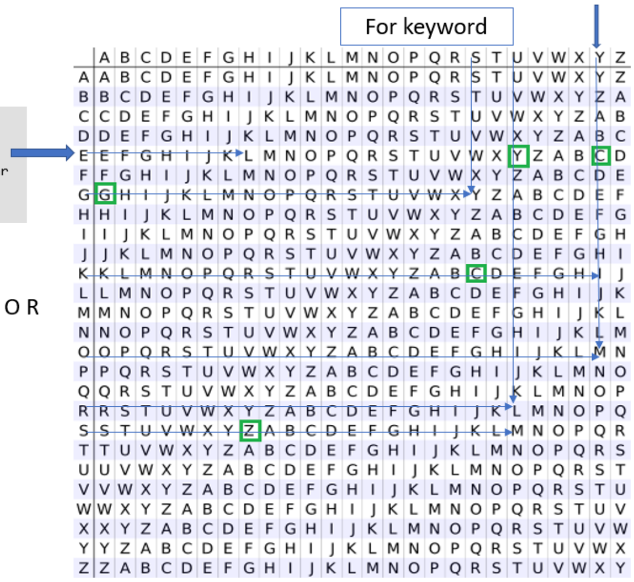
- **Vigenere Cipher is a method of encrypting alphabetic text. It uses a simple form of polyalphabetic substitution. A polyalphabetic cipher is any cipher based on substitution, using multiple substitution alphabets .The encryption of the original text is done using the *Vigenère square or Vigenère table*.**
- **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**



```
Input : Plaintext : GEEKSFORGEEKS
Keyword : AYUSH
Output : Ciphertext : 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"
```

Plaintext: GEEKSFOR
GEEKS
Repeated Keyword: AYUSHAYUSHAYU

Ciphertext: GCYCZFMLYLEIM


For keyword

	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

The Vigenère cipher can be expressed in the following manner. Assume a sequence of plaintext letters $P = p_0, p_1, p_2, \ldots\ldots, p_{n\text{-}1}$ and a key consisting of the sequence of letters $K = k_0, k_1, k_2, \ldots\ldots, k_{m\text{-}1}$, where typically $m < n$. The sequence of ciphertext letters $C = C_0, C_1, C_2, \ldots\ldots, C_{n\text{-}1}$ is calculated as follows:

$$C = C_0, C_1, C_2, \ldots\ldots, C_{n\text{-}1} = \mathrm{E}(K, P) = \mathrm{E}[(k_0, k_1, k_2, \ldots\ldots, k_{m\text{-}1}), (p_0, p_1, p_2 \ldots., p_{n\text{-}1})]$$

= $(p_0 + k_0)$ mod 26, $(p_1 + k_1)$ mod 26, ……, $(p_{m-1} + k_{m-1})$ mod 26, $(p_m + k_0)$ mod 26, $(p_{m+1} + k_1)$ mod 26, ……. , $(p_{2m-1} + k$

m-1) mod 26,…..

Thus, the first letter of the key is added to the first letter of the plaintext, mod 26, the second letters are added, and so on through the first *m* letters of the plaintext. For the next *m* letters of the plaintext, the key letters are repeated. This process continues until all of the plaintext sequence is encrypted. A general equation of the encryption process is

$C_i = (p_i + k_{i \bmod m}) \bmod 26$

A general equation for decryption is

$p_i = (C_i - k_{i \bmod m}) \bmod 26$

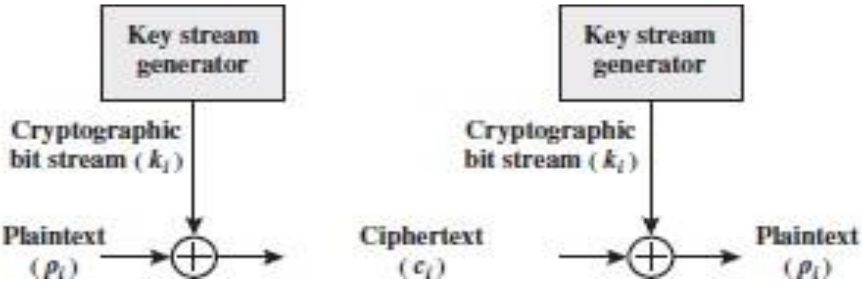
To encrypt a message, a key is needed that is as long as the message. Usually, the key is a repeating keyword. For example, if the keyword is *deceptive*, the message "we are discovered save yourself" is encrypted as

Key :*deceptivedeceptivedeceptive* plaintext : wearediscoveredsaveyourself

ciphertext : ZICVTWQNGRZGVTWAVZHCQYGLMGJ

The strength of this cipher is that there are multiple ciphertext letters for each plaintext letter, one for each unique letter of the keyword. Thus, the letter frequency information is obscured. However, not all knowledge of the plaintext structure is lost.

Vernam Cipher The ultimate defense against such a cryptanalysis is to choose a keyword that is as long as the plaintext and has no statistical relationship to it. Such a system was introduced by an AT&T engineer named Gilbert Vernam in 1918.



- The system can be expressed as:

$$ci = pi \oplus ki$$

where

p_i= *i*th binary digit of plaintext

k_i= *i*th binary digit of key

c_i= *i*th binary digit of ciphertext

$\oplus$ s= exclusive-or (XOR) operation

- Thus, the ciphertext is generated by performing the bitwise XOR of the plaintext and the key. Because of the properties of the XOR, decryption simply involves the same bitwise operation

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:

Ci = Pi Ki

Ci - ith binary digit of cipher text Pi - ith binary digit of plaintext

Ki - ith binary digit of key – 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:

Pi = Ci Ki

e.g., plaintext = 0 0 1 0 1 0 0 1

Key = 1 0 1 0 1 1 0 0

----

ciphertext = 1 0 0 0 0 1 0 1

Advantage:

□ Encryption method is completely unbreakable for a ciphertext only attack.

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 a second message; any knowledge on the first message would give knowledge of the second.

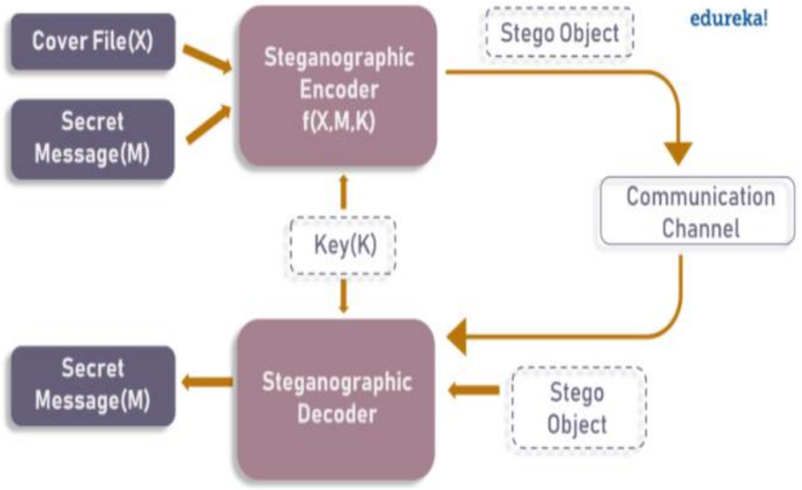
STEGANOGRAPHY:

➨ Steganography is the technique of hiding secret data within an ordinary, non-secret, file or message in order to avoid detection; the secret data is then extracted at its destination.

The use of steganography can be combined with encryption as an extra step for hiding or protecting data.

- It stems from two Greek words, which are *steganos,* means covered and *graphia,* means writing
- Examples,

1. Playing an audio track backwards to reveal a secret message
2. Playing a video at a faster frame rate (FPS) to reveal a hidden image
3. Embedding a message in the red, green, or blue channel of an RGB image
4. Hiding information within a file header or metadata
5. Embedding an image or message within a photo through the addition of digital noise



- As the image depicts, both cover file(X) and secret message(M) are fed into steganographic encoder as input.
- Steganographic Encoder function, f(X,M,K) embeds the secret message into a cover file.
- Resulting Stego Object looks very similar to your cover file, with no visible changes.
- This completes encoding. To retrieve the secret message, Stego Object is fed into Steganographic Decoder.

- Steganography Techniques
- Depending on the nature of the cover object (actual object in which secret data is embedded), steganography can be divided into five types:

1. Text Steganography
2. Image Steganography
3. Video Steganography
4. Audio Steganography

5. Network Steganography

➨ Text Steganography: Text Steganography is hiding information inside the text files. Various techniques used to hide the data in the text are:

- Format Based Method
- Random and Statistical Generation
- Linguistic Method

➨ Image Steganography: Hiding the data by taking the cover object as the image is known as image steganography. There are a lot of ways to hide information inside an image. Common approaches include:

- Least Significant Bit Insertion
- Masking and Filtering
- Redundant Pattern Encoding
- Encrypt and Scatter
- Coding and Cosine Transformation

➨ Audio Steganography: In audio steganography, the secret message is embedded into an audio signal which alters the binary sequence of the corresponding audio file. Different methods of audio steganography include:

- Least Significant Bit Encoding
- Parity Encoding
- Phase Coding
- Spread Spectrum

➨ Video Steganography: In Video Steganography you can hide kind of data into digital video format. Two main classes of Video Steganography include:

- embedding data in uncompressed raw video and compressing it later
- Embedding data directly into the compressed data stream
- Network Steganography (Protocol Steganography): It is the technique of embedding information within network control protocols used in data transmission such TCP, UDP, ICMP etc. For Example, you can hide information in the header of a TCP/IP packet in some fields that are either optional**.**

Example:

(i) the sequence of first letters of each word of the overall message spells out the real (hidden) message.

(ii) Subset of the words of the overall message is used to convey the hidden message.

Various other techniques have been used historically, some of them are:

- **Character marking** – selected letters of printed or typewritten text are overwritten in pencil. The marks are ordinarily not visible unless the paper is held to an angle to bright light.

Invisible ink – a number of substances can be used for writing but leave no visible trace until heat or some chemical is applied to the paper.

- **Pin punctures** – small pin punctures on selected letters are ordinarily not visible unless the paper is held in front of the light.
- **Typewritten correction ribbon** – used between the lines typed with a black ribbon, the results of typing with the correction tape are visible only under a strong light.

Drawbacks of steganography

- Requires a lot of overhead to hide a relatively few bits of information.
- Once the system is discovered, it becomes virtually worthless.

TRANSPOSITION TECHNIQUES:

All the techniques examined so far involve the substitution of a cipher text symbol for a plaintext symbol. 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.

Rail fence is simplest of such cipher, in which the plaintext is written down as a sequence of diagonals and then read off as a sequence of rows.

Plaintext = meet at the school house

To encipher this message with a rail fence of depth 2, we write the message as
follows:

m e a t e c o l o s
e t t h s H o h u e

The encrypted message is MEATECOLOSETTHSHOHUE

Row Transposition Ciphers-A more complex scheme is to write the message in a rectangle, row by row, and read the message off, column by column, but permute the order of the columns. The order of columns then becomes the key of the algorithm.

e.g., plaintext = meet at the school house

Key = 4 3 1 2 5 6 7

PT = m e e t a t t
h e s c h o o
l h o u s e

CT = ESOTCUEEHMHLAHSTOETO

A pure transposition cipher is easily recognized because it has the same letter frequencies as the original plaintext. The transposition cipher can be made significantly more secure by performing more than one stage of transposition. The result is more complex permutation that is not easily reconstructed.

FINITE FIELDS AND NUMBER THEORY:

- Finite fields have become increasingly important in cryptography.
- A number of cryptographic algorithms rely heavily on properties of finite fields, notably the Advanced Encryption Standard (AES) and elliptic curve cryptography.
- Other examples include the message authentication code CMAC and the authenticated encryption scheme GCM
    - Groups, Rings, Fields, Modular arithmetic, Euclid’s algorithm
    - Finite fields Euclid’s algorithm
    - Polynomial Arithmetic
    - Prime numbers-Fermat’s and Euler’s theorem
    - Testing for primality
    - The Chinese remainder theorem
    - Discrete logarithms
- Widely used in cryptography to perform large calculations
- Some basic concepts are
- **Prime Number**: a number that is divisible only by itself and 1 (e.g. 2, 3, 5, 7, 11)
- **Relative Prime Number**: 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, GCD (12,13) = 1→12 and 13 are **relatively prime**, but 12 and 14 are not.,
- Modular

Congruent Modulo

- **Modular :** When we divide two integers we will have an equation that looks like the following:
- A/B=Q remainder R
- A is the dividend

  B is the divisor

  Q is the quotient

  R is the remainder

- Sometimes, we are only interested in what the **remainder** is when we divide A by B. For these cases there is an operator called the modulo operator (abbreviated as mod).
- Using the same A, B, Q, and R as above, we would have: *A* mod *B*=*R*
- We would say this as A *modulo* B *is equal to* R. Where B is referred to as the **modulus**.

Ex. 13/5= 2 remainder of 3 then, **13 mod 5 = 3**

CONGRUENT MODULO:

- Consider two integers a and b
- a and b said to be congruent to n for
- a (mod n) = b (mod n) then
- **a ≡ b (mod n) (OR) a (mod n) = b**
- **example:**
- let a=73, b=4 and n=23
- ❑ find a mod n
- 73 mod 23 =4 (reminder of 73/23)
- ❑ find b mod n
- since 23 is larger than 4 then,
- 4 mod 23 = 4
- ❑ here 73 mod 23 = 4 and 4 mod 23=4, this can be written as
- 73 ≡ 4 (mod 23) ==> a ≡ b (mod n)

Properties of Congruences

Congruences have the following properties:

- **Property 1:** a ≡ b (mod n) if n is multiple of (a-b)
- Example: let a=30, b=10 and n=5
- a-b = 30-10 = 20
- Since 20 is multiple of 5 then 30 ≡ 10 (mod 5)
- **Property 2:** a (mod n) = b (mod n) ==> a ≡ b (mod n)
- **Property 3:** a mod n=b
- and b mod n=c, → b = c (mod n) sub it in a mod n
- then **a mod n= c mod n and a ≡ c (mod n)**
- **Arithmetic Property:** ((a mod n) + (b mod n))mod n= (a+b) mod n **[same for -,*,/]**
- **Commutative Property: (a+b) mod n = (b+a) mod n [same for *]**
- **Associative Property:** ((a + b)+c) mod n = (a+(b + c)) mod n

- **Identity Property:**
- (0+a) mod n = a mod n
- (1 * a) mod n = a mod n

Modular Arithmetic Operations

The (mod n) operator maps all integers into the set of integers {0, 1, c, (n - 1)}. This technique is known as **modular arithmetic**.

Modular arithmetic exhibits the following properties:

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

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

$[(a \bmod n) \qquad (b \bmod n)] \bmod n = (ab) \bmod n$

First property:

Define $(a \bmod n) = r_a$and $(b \bmod n) = r_b$. Then we can write $a = r_a + jn$for some integer j and $b = r_b + kn$for some integer k.

Then

$(a + b) \bmod n = (ra + jn + rb + kn) \bmod n = (ra + rb + (k + j)n) \bmod n$

$= (ra + rb) \bmod n$

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

Define $(a \bmod n) = r_a$and $(b \bmod n) = r_b$. Then we can write $a = r_a + jn$for some integer j and $b = r_b + kn$for some integer k.

Then

$(a + b) \bmod n = (ra + jn + rb + kn) \bmod n = (ra + rb + (k + j)n) \bmod n$

$= (ra + rb) \bmod n$

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

Examples of the three properties:

11 mod 8 = 3; 15 mod 8 = 7

[(11 mod 8) + (15 mod 8)] mod 8 = 10 mod 8 =2

(11 + 15) mod 8 =26 mod 8 =2

[(11 mod 8) – (15 mod 8)] mod 8 = -4 mod 8 = 4

(11-15) mod 8 = -4 mod 8 = 4

[(11 mod 8)× (15 mod 8)] mod 8 = 21 mod 8 =5

(11 × 15) mod 8 = 165 mod 8 =5

Exponentiation is performed by repeated multiplication, as in ordinary arithmetic.

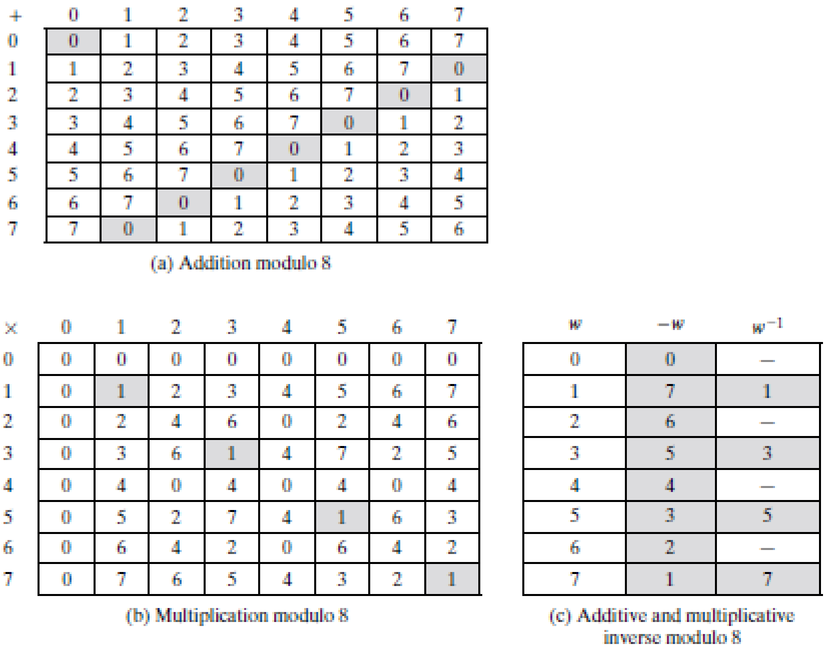
To find 11^7 mod 13,

11^2 = 121 = 4 (mod 13)

11^4 =$(11^2)^2$ = 4^2 = 3 (mod 13)

11^7 = 11 × 4 × 3 = 132 = 2 (mod 13)

Thus, the rules for ordinary arithmetic involving addition, subtraction, and multiplication carry over into modular arithmetic. The following table below provides an illustration of modular addition and multiplication modulo 8



+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	0
2	2	3	4	5	6	7	0	1
3	3	4	5	6	7	0	1	2
4	4	5	6	7	0	1	2	3
5	5	6	7	0	1	2	3	4
6	6	7	0	1	2	3	4	5
7	7	0	1	2	3	4	5	6

(a) Addition modulo 8

×	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	0	2	4	6
3	0	3	6	1	4	7	2	5
4	0	4	0	4	0	4	0	4
5	0	5	2	7	4	1	6	3
6	0	6	4	2	0	6	4	2
7	0	7	6	5	4	3	2	1

(b) Multiplication modulo 8

w	$-w$	w^{-1}
0	0	—
1	7	1
2	6	—
3	5	3
4	4	—
5	3	5
6	2	—
7	1	7

(c) Additive and multiplicative inverse modulo 8

Both matrices are symmetric about the main diagonal in conformance to the commutative property of addition and multiplication.

As in ordinary addition, there is an additive inverse, or negative, to each integer in modular arithmetic.

In this case, the negative of an integer *x* is the integer *y* such that (*x* + *y*) mod 8 = 0.

To find the additive inverse of an integer in the left-hand column, scan across the corresponding row of the matrix to find the value 0; the integer at the top of that column is the additive inverse; thus, (2 + 6) mod 8 = 0. Similarly, the entries in the multiplication table are straightforward.

In modular arithmetic mod 8, the multiplicative inverse of *x* is the integer *y*

such that (*x y*) mod 8 = 1 × mod 8.

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 theorem states the following: If *p* is prime and *a* is a positive integer not divisible by *p*, then

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

Proof: Consider the set of positive integers less than *p*: {1, 2, c, *p* - 1} and multiply each element by *a*, modulo *p*, to get the set *X* = {*a* mod *p*, 2*a* mod *p*, c, (*p* - 1)amod *p*}. None of the elements of *X* is equal to zero because *p* does not divide *a*. Furthermore, no two of the integers in *X* are equal.

- To see this, assume that *ja* ≡ *ka*(mod *p*), where 1 ≤ *j* < *k* ≤ *p* - 1. Because *a* is relatively prime to*p*, Eliminate *a* from both sides of the ≡equation resulting in *j k* (mod *p*).
- This last equality is impossible, because *j* and *k* are both positive integers less than *p*. Therefore, (*p* - 1) elements of *X* are all positive integers with no two elements equal.
- We can conclude the *X* consists of the set of integers {1, 2, ...,*p* - 1} in some order. Multiplying the numbers in both sets (*p* and *X*) and taking the result mod *p* yields

$$a \times 2a \times \ldots\ldots \times (p - 1)a \equiv [(1 \times 2 \times \ldots\ldots \times (p - 1)] \pmod{p}$$

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

- We can cancel the (*p* - 1)! term because it is relatively prime to *p*. This yields Equation, which completes the proof.

$$a = 7, p = 19$$

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

$$7^4 \equiv 121 \equiv 7 \pmod{19}$$

$$7^8 \equiv 49 \equiv 11 \pmod{19}$$

$$7^{16} \equiv 121 \equiv 7 \pmod{19}$$

$$a^{p-1} = 7^{18} = 7^{16} \times 7^2 \equiv 7 \times 11 \equiv 1 \pmod{19}$$

- An alternative form of Fermat‟s theorem is also useful: If *p* is prime and *a* is a positive integer, then

$$a^p \equiv a (\mathrm{mod}\ p)$$

Euler’s Theorem

Euler‟s theorem states that for every *a* and *n* that are relatively prime:

$$a\Phi^{(n)} \equiv 1(\mathrm{mod}\ n)$$

Proof: The above equation is true if *n* is prime, because in that case,

□(n) = (n- 1) and Fermat‟s theorem holds. However, it also holds for any integer n. □(n) is the number of positive integers less than *n* that are relatively primeto *n*.

Consider the set of such integers, labeled as

$R = \{x_1, x_2, \ldots\ldots, x_{\Phi(n)}\}$

That is, each element $x_i$of *R* is a unique positive integer less than *n* with gcd(x_i, *n*) = 1.

Now multiply each element by *a*, modulo *n*:

$S = \{(ax_1 \bmod n), (ax_2 \bmod n), \ldots \quad , (ax_{\phi(n)} \bmod n)\}$

The set *S* is a permutation of *R*, by the following line of reasoning:

Because *a* is relatively prime to *n* and $x_i$is relatively prime to *n*, ax_imust also be relatively prime to *n*. Thus, all the members of *S* are integers that are less than *n* and that are relatively primeto *n*.

If ax_imod n = ax_jmod *n*, then $x_i = x_j$.

Therefore,

$$\prod_{i=1}^{\phi n}(ax_i \bmod n) = \prod_{i=1}^{\phi n} x_i$$

$$\prod_{i=1}^{\phi n} x_i = \sum_{i=1}^{\phi n}(ax_i \bmod n)$$

$$a\phi(n) \times \left[\prod_{i=1}^{\phi n} x_i\ (mod\ n)\right]$$

$$a^{\phi(n)} \equiv 1\ mod\ n$$

which completes the proof. This is the same line of reasoning applied to the proof of Fermat‟s theorem.

a=3; n=10; ϕ(10)= 4$a^{\phi(n)}$ = 3^4 = 81 = 1(mod 10) = 1(mod n) a=2; n=11;

ϕ(11)= 10$a^{\phi(n)}$ = 2^{10} = 1024 = 1(mod 11) = 1(mod n)

As is the case for Fermat‟s theorem, an alternative form of the theorem is also useful:

$a\phi^{\,n+1} \equiv a\ mod\ n$

CHINESE REMINDER THEOREM:

One of the most useful results of number theory is the **Chinese remainder theorem** (CRT). In essence, the CRT says it is possible to reconstruct integers in a certain range from their residues modulo a set of pairwise relatively prime moduli.

The CRT can be stated in several ways. Let

$$M = \prod_{i=1}^{k} m_i$$

where *A* Z_M,*ai* Z_{mi}, and a_i= *A* mod m_i for 1   i   *k*.

The CRT makes two assertions. The mapping of the above equation is a one-to-one correspondence (called a **bijection**) between Z_M and the Cartesian product Z*m*1× Z*m*2 ×….×Z*mk*. That is, for every integer *A* such that 0≤*A* ≤*M*, there is a unique *k*-tuple (*a*1, *a*2,…., *ak*) with 0≤ *ai* < *mi* that represents it, and for every such *k*-tuple (*a*1, *a*2,….. *ak*), there is a unique integer *A* in Z*M*.

Operations performed on the elements of Z*M* can be equivalently performed on the corresponding *k*-tuples by performing the operation independently in each coordinate position in the appropriate system.

FINITE FIELDS

Groups, Rings and Field:

□ **Group**: A set of elements that is closed with respect to some operation.

□ Closed-> The result of the operation is also in the set

□ The operation obeys:

□ Obeys associative law: **(a.b).c = a.(b.c)**

□ Has identity e: **e.a = a.e = a**

□ Has inverses **a-1: a.a-1 = e**

□ **Abelian Group**: The operation is commutative

a.b = b.a

□ Example: Z8, + modular addition, identity =0

Cyclic Group

Exponentiation: Repeated application of operator

□ example: **a3 = a.a.a**

□ Cyclic Group: Every element is a power of some fixed element, i.e., **b = ak**

for some a and every b in group a is said to be a generator of the group

□ Example: {1, 2, 4, 8} with mod 12 multiplication, the generator is 2.

□ 20=1, 21=2, 22=4, 23=8, 24=4, 25=8

Ring:

□ A group with two operations: addition and multiplication

□ The group is abelian with respect to addition: **a+b=b+a**

□ Multiplication and additions are both associative:

a+(b+c)=(a+b)+c

a.(b.c)=(a.b).c

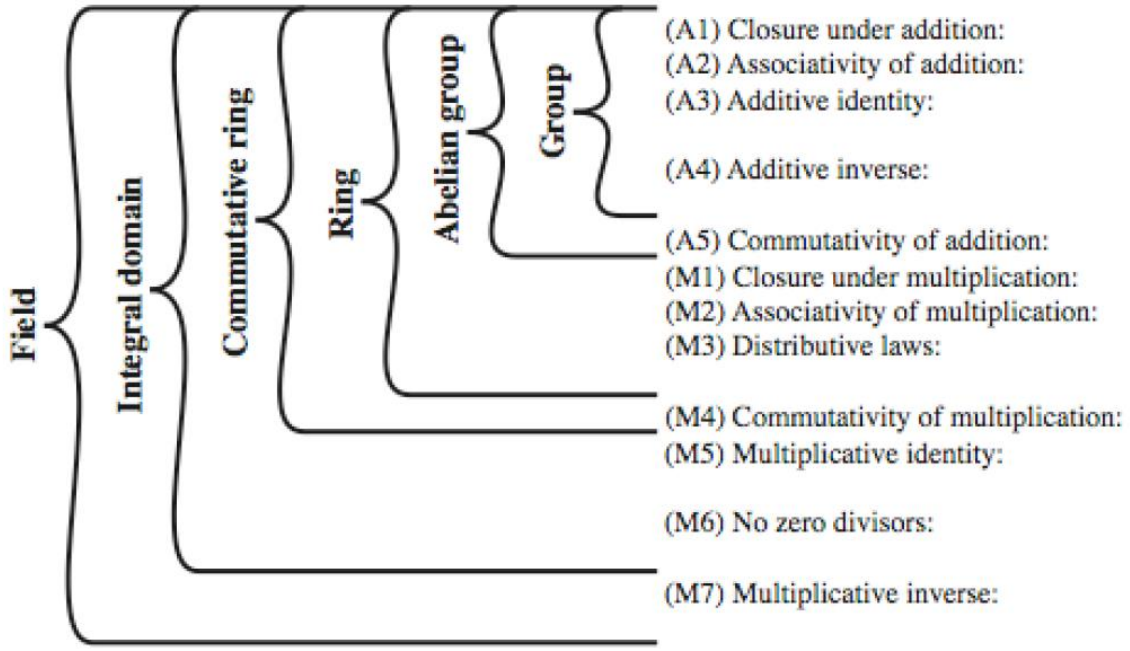
□ Multiplication distributes over addition, **a.(b+c)=a.b+a.c**

 Commutative Ring: Multiplication is commutative, i.e., **a.b = b.a**

 Integral Domain: Multiplication operation has an identity and no zero divisors

Field:

An integral domain in which each element has a multiplicative inverse.


Field
Integral domain
Commutative ring
Ring
Abelian group
Group
(A1) Closure under addition:
(A2) Associativity of addition:
(A3) Additive identity:
(A4) Additive inverse:
(A5) Commutativity of addition:
(M1) Closure under multiplication:
(M2) Associativity of multiplication:
(M3) Distributive laws:
(M4) Commutativity of multiplication:
(M5) Multiplicative identity:
(M6) No zero divisors:
(M7) Multiplicative inverse:

Polynomial Arithmetic

***f*(*x*) = an*x*n + an-1*x*n-1 + …+ a1*x* + a0 = Σ ai*x*i**

1. Ordinary polynomial arithmetic:

- Add, subtract, multiply, divide polynomials,
- Find remainders, quotient.
- Some polynomials have no factors and are prime.

2. Polynomial arithmetic with mod p coefficients

3. Polynomial arithmetic with mod p coefficients and mod m(x) operations

Polynomial Arithmetic with Mod 2 Coefficients

- All coefficients are 0 or 1, e.g.,

  let *f*(*x*) = *x*3 + *x*2 and *g*(*x*) = *x*2 + *x* + 1

  f(*x*) + *g*(*x*) = *x*3 + *x* + 1

  f(*x*) x *g*(*x*) = *x*5 + *x*2
- Polynomial Division: *f*(*x*) = *q*(*x*) *g*(*x*) + *r*(*x*)
- can interpret *r*(*x*) as being a remainder
- *r*(*x*) = *f*(*x*) mod *g*(*x*)

- if no remainder, say *g*(*x*) divides *f*(*x*)
- if *g*(*x*) has no divisors other than itself & 1 say it is irreducible (or prime) polynomial
- Arithmetic modulo an irreducible polynomial form a finite field
- Can use Euclid"s algorithm to find gcd and inverses.

Discrete Logarithm:

The inverse problem to exponential is to find the discreate logarithm of a number modulo P, that is to find i

$$b = a^i \ (mod\ p)$$

Written as

$$i = dlog_a b \ (mod\ p)$$

If a is a primitive root then it always exists, otherwise it may not.

Eg. $x = log_3 4 \ \ mod$ 13 has no answer

$x = log_2 \ \ 3 \ \ mod$ 13 =4 by typing successive power

References

1. William Stallings, Cryptography and Network Security, 6th Edition, Pearson Education, March 2013.
2. Behrouz A. Ferouzan, “Cryptography & Network Security”, Tata McGraw Hill, 2007.
3. Man Young Rhee, “Internet Security: Cryptographic Principles”, “Algorithms and Protocols”, Wiley Publications, 2003.
4. Charles Pfleeger, “Security in Computing”, 4th Edition, Prentice Hall of India, 2006.

Unit-2

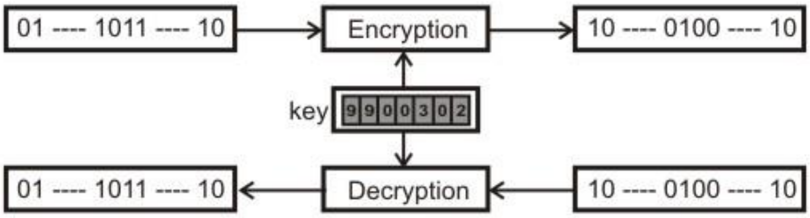
Block Cipher and Public Key Cryptography

Block Cipher:

A block cipher takes a block of plaintext bits and generates a block of ciphertext bits, generally of same size. The size of block is fixed in the given scheme. The choice of block size does not directly affect to the strength of encryption scheme. The strength of cipher depends up on the key length.

Block Size:

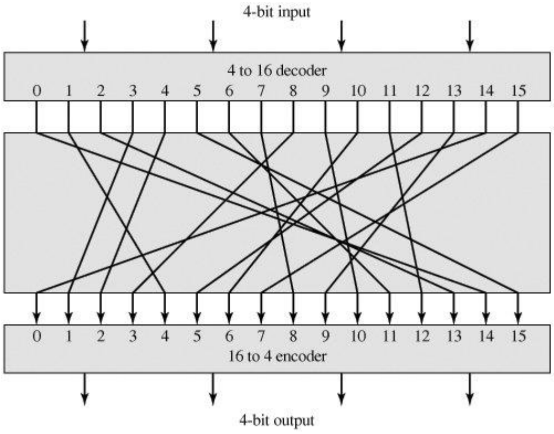
Block ciphers use a block of bits as the unit of encryption and decryption. To encrypt a 64-bit block, one has to take each of the 2^{64} input values and map it to one of the 2^{64} output values. The mapping should be one-to-one. Encryption and decryption operations of a block cipher are shown in Fig.



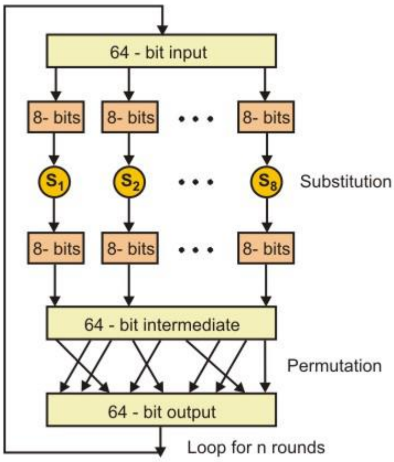
Some operations, such as permutation and substitution, are performed on the block of bits based on a key (a secret number) to produce another block of bits.

Permutation: The permutation is performed by a permutation box at the bit-level, which keeps the number of 0s and 1s same at the input and output. Although it can be implemented either by a hardware or a software, the hardware implementation is faster.

Substitution: Fig. shows the substitution is implemented with the help of three building blocks – a decoder, one p-box and an encoder. For an n-bit input, the decoder produces an 2_n bit output having only one 1, which is applied to the P-box. The P-box permutes the output of the decoder and it is applied to the encoder. The encoder, in turn, produces an n-bit output. For example, if the input to the decoder is 011, the output of the decoder is 00001000. Let the permuted output is 01000000, the output of the encoder is 011.



Most symmetric block ciphers are based on a Feistel Cipher Structure needed since must be able to decrypt ciphertext to recover messages efficiently. block ciphers look like an extremely It devid input into 8-Bit pieces Substitute each 8-bit based on functions derived from the key. Permute the bits based on the key



• Requires table of 264 entries for a 64-bit block

• Instead create from smaller building blocks

• using idea of a product cipher in 1949 Claude Shannon introduced idea of substitution-

permutation (S-P) networks called modern substitution-transposition product cipher

these form the basis of modern block ciphers

• S-P networks are based on the two primitive cryptographic operations we have seen before:

• *substitution* (S-box)

• *permutation* (P-box)

• provide *confusion* and *diffusion* of message

• diffusion – dissipates statistical structure of plaintext over bulk of ciphertext

• confusion – makes relationship between ciphertext and key as complex as possible

Block Cipher Schemes:

There is a vast number of block ciphers schemes that are in use. Many of them are publicly known. Most popular and prominent block ciphers are listed below.

Digital Encryption Standard (DES) − The popular block cipher of the 1990s. It is now considered as a 'broken' block cipher, due primarily to its small key size.

Triple DES − It is a variant scheme based on repeated DES applications. It is still a respected block ciphers but inefficient compared to the new faster block ciphers available.

Advanced Encryption Standard (AES) − It is a relatively new block cipher based on the encryption algorithm

Feistel cipher structure:

The plaintext block is divided into two halves L0 and R0. 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 Li-1 and Ri-1, derived from the previous round, as well as the subkey Ki, derived from the overall key K. in general, the subkeys Ki 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

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 ki. Following this substitution, a permutation is performed that consists of the interchange of the two halves of the data.

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 ki in reverse order. i.e., kn in the first round, kn-1 in second round and so on.

The exact realization of a Feistel network depends on the choice of the following parameters and design features:

· Block size - Increasing size improves security, but slows cipher

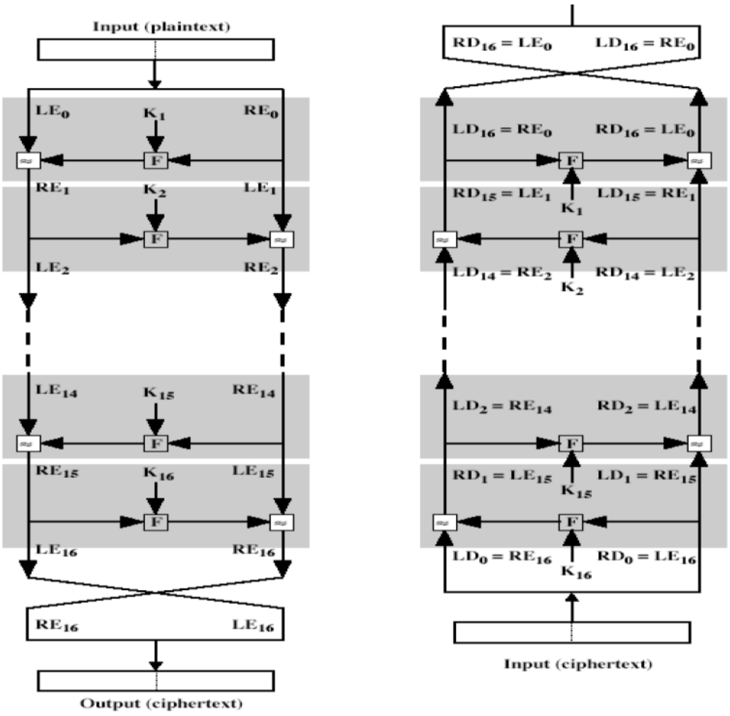
· Key size - Increasing size improves security, makes exhaustive key searching harder, but may slow cipher

· 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

· Fast software en/decryption & ease of analysis - are more recent concerns for practical use and testing.



The exact realization of a Feistel network depends on the choice of the following parameters and design features:

- Block size: Larger block sizes mean greater security (all other things being equal) but reduced encryption/decryption speed for a given algorithm. The greater security is achieved by greater diffusion. Traditionally, a block size of 64 bits has been considered a reasonable tradeoff and was nearly universal in block cipher design. However, the new AES uses a 128-bit block size.

- Key size: Larger key size means greater security but may decrease encryption/decryption speed. The greater security is achieved by greater resistance to brute-force attacks and greater confusion. Key sizes of 64 bits or less are now widely considered to be inadequate, and 128 bits has become a common size.

- Number of rounds: The essence of the Feistel cipher is that a single round offers inadequate security but that multiple rounds offer increasing security. A typical size is 16 rounds.

- Subkey generation algorithm: Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis.

- Round function F: Again, greater complexity generally means greater resistance to cryptanalysis.

There are two other considerations in the design of a Feistel cipher:

- Fast software encryption/decryption: In many cases, encryption is embedded in applications or utility functions in such a way as to preclude a hardware implementation.

Accordingly, the speed of execution of the algorithm becomes a concern.

- Ease of analysis: Although we would like to make our algorithm as difficult as possible to cryptanalyze, there is great benefit in making the algorithm easy to analyze. That is, if the algorithm can be concisely and clearly explained, it is easier to analyze that algorithm for cryptanalytic vulnerabilities and therefore develop a higher level of assurance as to its strength. DES, for example, does not have an easily analyzed functionality.

Block Cipher Schemes

There is a vast number of block ciphers schemes that are in use. Many of them are publicly known. Most popular and prominent block ciphers are listed below.

Digital Encryption Standard (DES) − The popular block cipher of the 1990s. It is now considered as a ‘broken’ block cipher, due primarily to its small key size.

Triple DES − It is a variant scheme based on repeated DES applications. It is still a respected block ciphers but inefficient compared to the new faster block ciphers available.

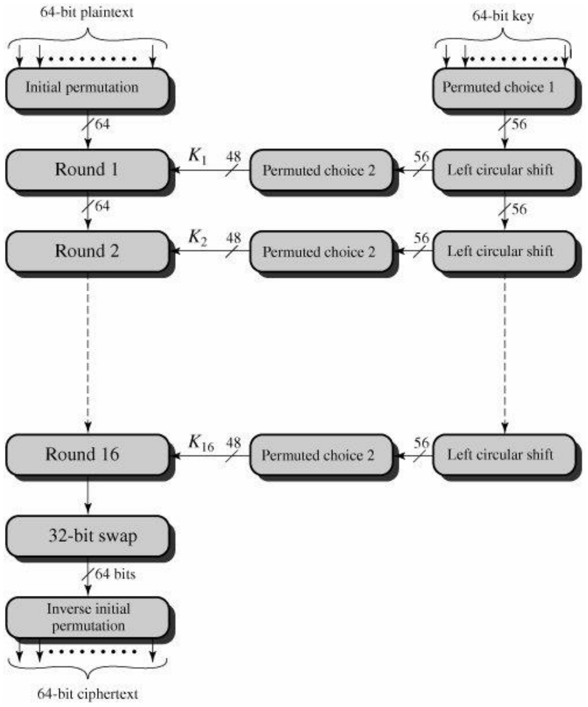
Advanced Encryption Standard (AES) − It is a relatively new block cipher based on the encryption algorithm

Data Encryption

In May 1973, and again in Aug 1974 the NBS (now NIST) called for possible encryption algorithms for use in unclassified government applications response was mostly disappointing, however, IBM submitted their Lucifer design following a period of redesign and comment it became the Data Encryption Standard (DES) it was adopted as a (US) federal standard in Nov 76, published by NBS as a hardware only scheme in Jan 77 and by ANSI for both hardware and software standards in ANSI X3.92-

1981 (also X3.106-1983 modes of use) subsequently it has been widely adopted and is now published in many standards around the world cf Australian Standard AS2805.5-1985 one of the largest users of the DES is the banking industry, particularly with EFT, and EFTPOS it is for this use that the DES has primarily been standardized, with ANSI having twice reconfirmed its recommended use for 5 year periods - a further extension is not expected however although the standard is public, the design criteria used are classified and have yet to be released there has been considerable controversy over the design, particularly in the choice of a 56-bit key, recent analysis has shown despite this that the choice was appropriate, and that DES is well designed, rapid advances in computing speed though have rendered the 56-bit key susceptible to exhaustive key search, as predicted by Diffie & Hellman.

The overall scheme for DES encryption is illustrated in Fig. As with any encryption scheme, there are two inputs to the encryption function: the plaintext to be encrypted and the key. In this case, the plaintext must be 64 bits in length and the key is 56 bits in length.



Looking at the left-hand side of the figure, we can see that the processing of the plaintext proceeds in three phases. First, the 64-bit plaintext passes through an initial permutation (IP) that rearranges the bits to produce the *permuted input*. This is followed by a phase consisting of 16 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. With the exception of the initial and final permutations, DES has the exact structure of a Feistel cipher.

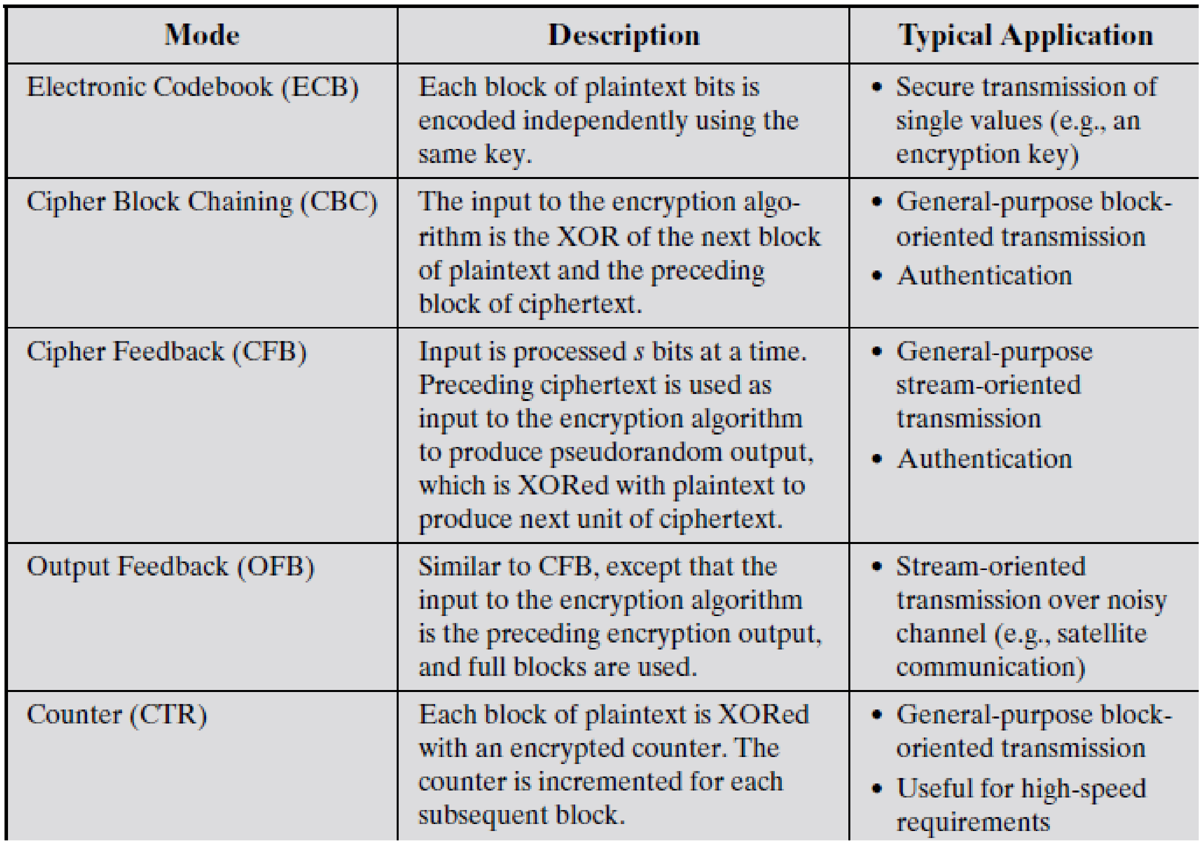
Block Cipher Modes of Operation

Block cipher mode of operation is an algorithm that uses a block cipher to processes the data blocks of fixed size provide information security such as confidentiality or authenticity.

the long message is divided into a series of sequential message blocks, and the cipher operates on these blocks one at a time.

A block cipher by itself is only suitable for the secure cryptographic transformation (encryption or decryption) of one fixed-length group of bits called a block.

A mode of operation describes how to repeatedly apply a cipher's single-block operation to securely transform amounts of data larger than a block.



Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of plaintext bits is encoded independently using the same key.	• Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next block of plaintext and the preceding block of ciphertext.	• General-purpose block-oriented transmission • Authentication
Cipher Feedback (CFB)	Input is processed *s* bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	• General-purpose stream-oriented transmission • Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding encryption output, and full blocks are used.	• Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	• General-purpose block-oriented transmission • Useful for high-speed requirements

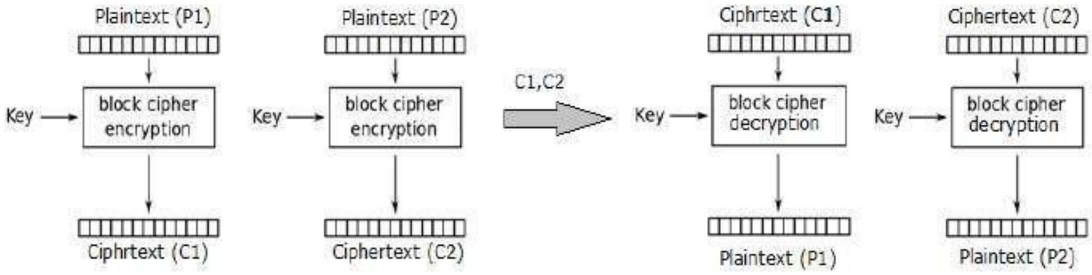
Electronic Code Book (ECB) Mode

This mode is a most straightforward way of processing a series of sequentially listed message blocks.

Operation:

The user takes the first block of plaintext and encrypts it with the key to produce the first block of ciphertext. He then takes the second block of plaintext and follows the same process with same key and so on so forth. The ECB mode is deterministic, that is, if plaintext block P1, P2,⋯, Pm are encrypted twice under the same key, the output ciphertext blocks will be the same.

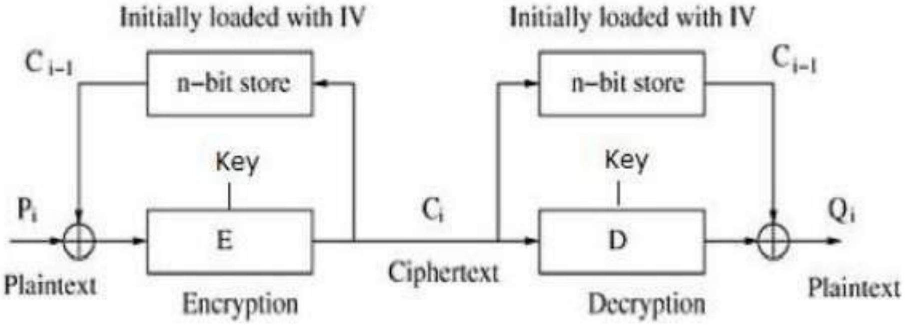
In fact, for a given key technically we can create a codebook of ciphertexts for all possible plaintext blocks. Encryption would then entail only looking up for required plaintext and select the corresponding ciphertext. Thus, the operation is analogous to the assignment of code words in a codebook, and hence gets an official name − Electronic Codebook mode of operation (ECB). It is illustrated as follows



Analysis of ECB Mode:
In reality, any application data usually have partial information which can be guessed. For example, the range of salary can be guessed. A ciphertext from ECB can allow an attacker to guess the plaintext by trial-and-error if the plaintext message is within predictable.
For example, if a ciphertext from the ECB mode is known to encrypt a salary figure, then a small number of trials will allow an attacker to recover the figure. In general, we do not wish to use a deterministic cipher, and hence the ECB mode should not be used in most applications.

Cipher Block Chaining (CBC) Mode

CBC mode of operation provides message dependence for generating ciphertext and makes the system non-deterministic.
Operation:
The operation of CBC mode is depicted in the following illustration. The steps are as follows. Load the n-bit Initialization Vector (IV) in the top register. XOR the n-bit plaintext block with data value in top register. Encrypt the result of XOR operation with underlying block cipher with key K. Feed ciphertext block into top register and continue the operation till all plaintext blocks are processed.
For decryption, IV data is XORed with first ciphertext block decrypted. The first ciphertext block is also fed into to register replacing IV for decrypting next ciphertext block.



Analysis of CBC Mode

In CBC mode, the current plaintext block is added to the previous ciphertext block, and then the result is encrypted with the key. Decryption is thus the reverse process, which involves decrypting the current ciphertext and then adding the previous ciphertext block to the result.

Advantage of CBC over ECB is that changing IV results in different ciphertext for identical message. On the drawback side, the error in transmission gets propagated to few further blocks during decryption due to chaining effect.

It is worth mentioning that CBC mode forms the basis for a well-known data origin authentication mechanism. Thus, it has an advantage for those applications that require both symmetric encryption and data origin authentication.

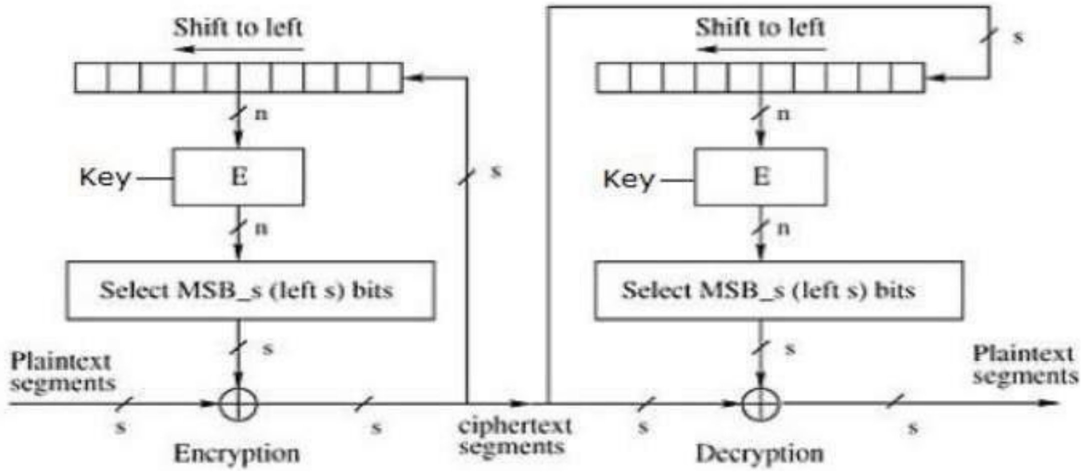
Cipher Feedback (CFB) Mode

In this mode, each ciphertext block gets 'fed back' into the encryption process in order to encrypt the next plaintext block.

Operation

The operation of CFB mode is depicted in the following illustration. For example, in the present system, a message block has a size 's' bit where 1 < s < n. The CFB mode requires an initialization vector (IV) as the initial random n-bit input block. The IV need not be secret.

Steps of operation are, Load the IV in the top register. Encrypt the data value in top register with underlying block cipher with key K. Take only 's' number of most significant bits (left bits) of output of encryption process and XOR them with 's' bit plaintext message block to generate ciphertext block. Feed ciphertext block into top register by shifting already present data to the left and continue the operation till all plaintext blocks are processed. Essentially, the previous ciphertext block is encrypted with the key, and then the result is XORed to the current plaintext block. Similar steps are followed for decryption. Pre-decided IV is initially loaded at the start of decryption.



Analysis of CFB Mode

CFB mode differs significantly from ECB mode, the ciphertext corresponding to a given plaintext block depends not just on that plaintext block and the key, but also on the previous ciphertext block. In other words, the ciphertext block is dependent of message.

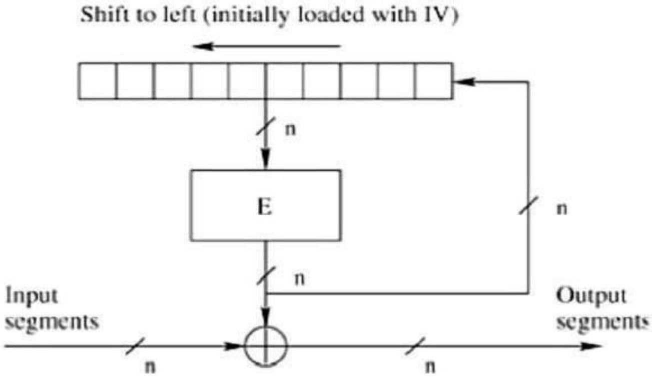
CFB has a very strange feature. In this mode, user decrypts the ciphertext using only the encryption process of the block cipher. The decryption algorithm of the underlying block cipher is never used.

Apparently, CFB mode is converting a block cipher into a type of stream cipher. The encryption algorithm is used as a key-stream generator to produce key-stream that is placed in the bottom register. This key stream is then XORed with the plaintext as in case of stream cipher.

By converting a block cipher into a stream cipher, CFB mode provides some of the advantageous properties of a stream cipher while retaining the advantageous properties of a block cipher. On the flip side, the error of transmission gets propagated due to changing of blocks.

Output Feedback (OFB) Mode

It involves feeding the successive output blocks from the underlying block cipher back to it. These feedback blocks provide string of bits to feed the encryption algorithm which act as the key-stream generator as in case of CFB mode. The key stream generated is XOR-ed with the plaintext blocks. The OFB mode requires an IV as the initial random n-bit input block. The IV need not be secret. The operation is depicted in the following illustration



Counter (CTR) Mode:

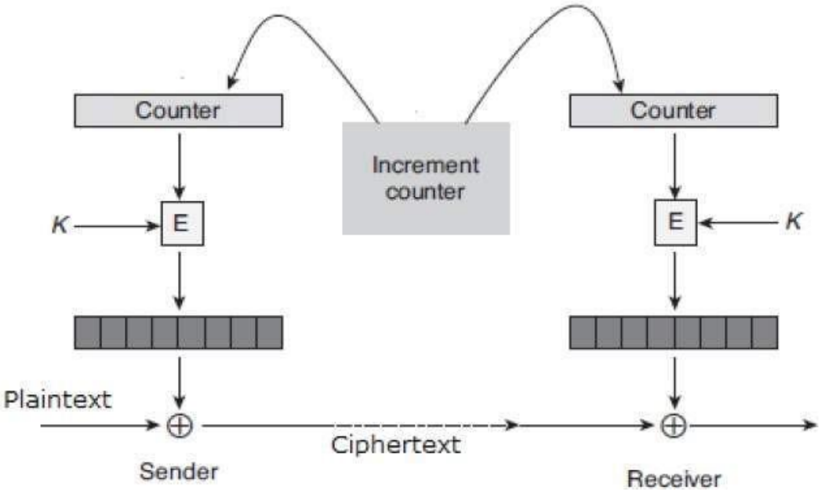
It can be considered as a counter-based version of CFB mode without the feedback. In this mode, both the sender and receiver need to access to a reliable counter, which computes a new shared value each time a ciphertext block is exchanged. This shared counter is not necessarily a secret value, but challenge is that both sides must keep the counter synchronized.

Operation

Both encryption and decryption in CTR mode are depicted in the following illustration. Steps in operation are, Load the initial counter value in the top register is the same for both the sender and the receiver. It plays the same role as the IV in CFB (and CBC) mode.

Encrypt the contents of the counter with the key and place the result in the bottom register. Take the first plaintext block P1 and XOR this to the contents of the bottom register. The result of this is C1. Send C1 to the receiver and update the counter. The counter update replaces the ciphertext feedback in CFB mode.

Continue in this manner until the last plaintext block has been encrypted. The decryption is the reverse process. The ciphertext block is XORed with the output of encrypted contents of counter value. After decryption of each ciphertext block counter is updated as in case of encryption.



Analysis of Counter Mode

It does not have message dependency and hence a ciphertext block does not depend on the previous plaintext blocks. Like CFB mode, CTR mode does not involve the decryption process of the block cipher. This is because the CTR mode is really using the block cipher to generate a key-stream, which is encrypted using the XOR function. In other words, CTR mode also converts a block cipher to a stream cipher.

The serious disadvantage of CTR mode is that it requires a synchronous counter at sender and receiver. Loss of synchronization leads to incorrect recovery of plaintext. However, CTR mode has almost all advantages of CFB mode. In addition, it does not propagate error of transmission at all.

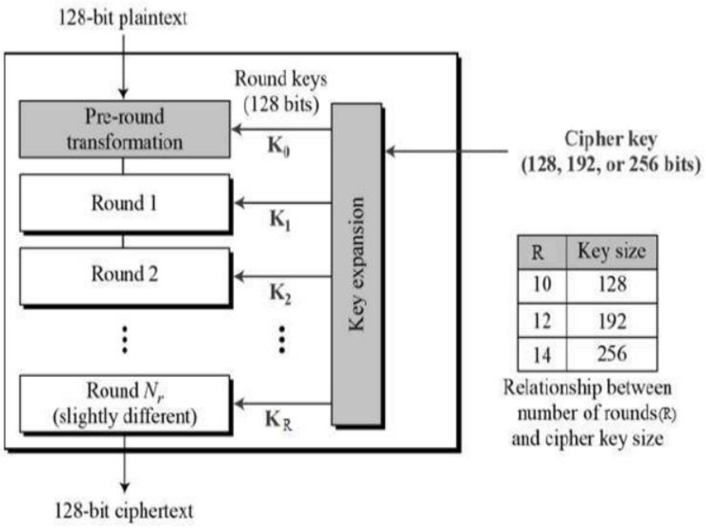
The Advanced Encryption Standard (AES):

The Advanced Encryption Standard (AES) was published by the National Institute of Standards and Technology (NIST) in 2001. AES is a symmetric block cipher that is intended to replace DES as the approved standard for a wide range of applications. Compared to public-key ciphers such as RSA, the structure of AES and most symmetric ciphers is quite complex and cannot be explained as easily as many other cryptographic algorithms.

AES Parameters

- Block Size is 128
- No. of Rounds is 10
- Key Size is 128 bits (4 Words/16 Bytes)
- No. of Sub keys is 44 (128 bit)
- Each subkey Size is 32 bits/ 1 word/ 4 bytes
- Each Round there is 4 subkeys, total 40 subkeys
- Pre round calculation, 4 Subkeys were used
- Cipher Text is 128 bits

Figure shows the overall structure of the AES encryption process.

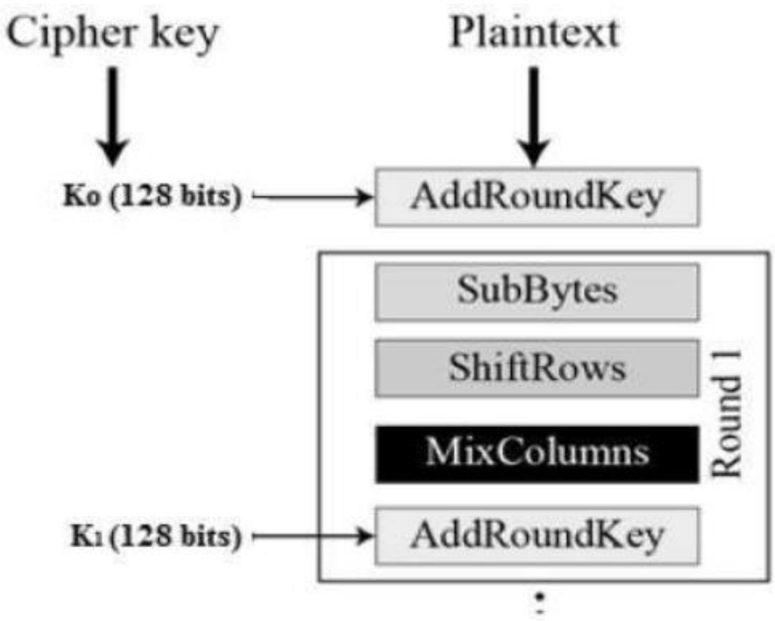


R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

Encryption Process:

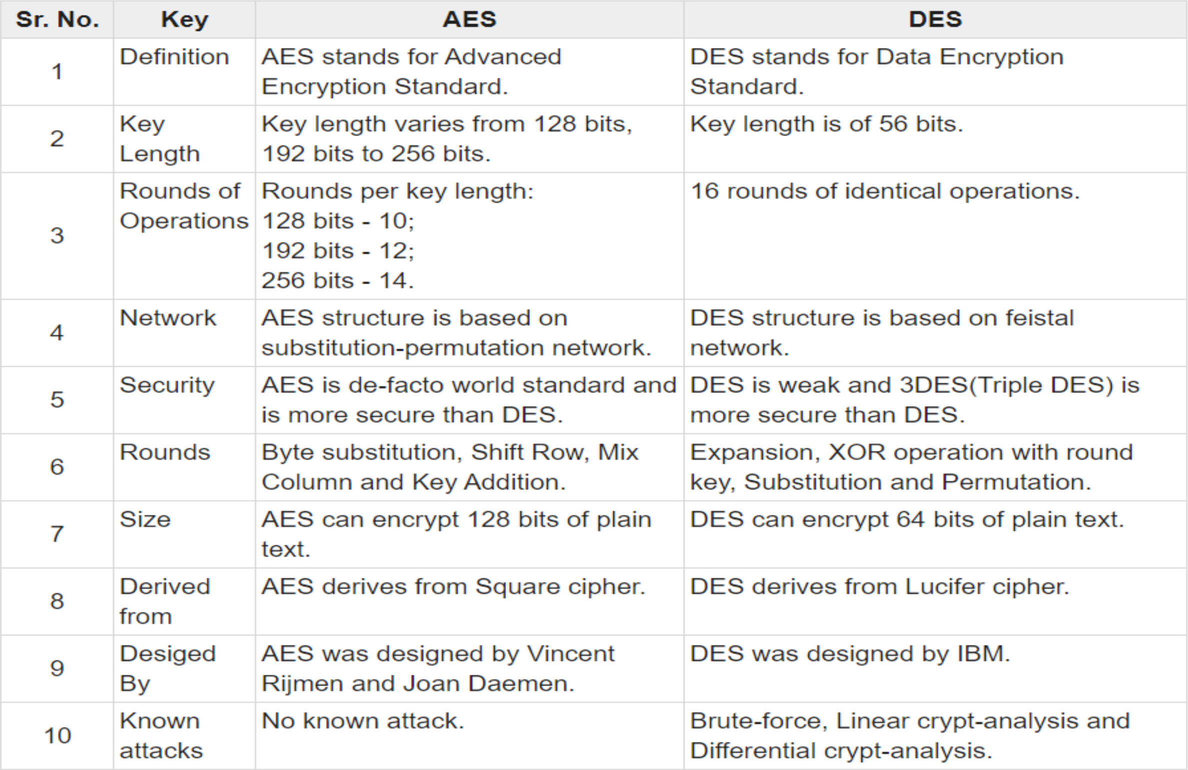
Here, we restrict to description of a typical round of AES encryption. Each round comprises of four sub-processes. The first-round process is depicted below



- **Byte Substitution (Sub Bytes):**
- The 16 input bytes are substituted by looking up a fixed table (S-box) given in design. The result is in a matrix of four rows and four columns.
- **Shift rows:**
- Each of the four rows of the matrix is shifted to the left. Any entries that ‘fall off’ are re-inserted on the right side of row. Shift is carried out as follows.

- First row is not shifted.
- Second row is shifted one (byte) position to the left.
- Third row is shifted two positions to the left.
- Fourth row is shifted three positions to the left.
- The result is a new matrix consisting of the same 16 bytes but shifted with respect to each other.

Comparison Between DES and AES:



Sr. No.	Key	AES	DES
1	Definition	AES stands for Advanced Encryption Standard.	DES stands for Data Encryption Standard.
2	Key Length	Key length varies from 128 bits, 192 bits to 256 bits.	Key length is of 56 bits.

3	Rounds of Operations	Rounds per key length:
		128 bits - 10;
		192 bits - 12;
		256 bits - 14.
			16 rounds of identical operations.

4	Network	AES structure is based on substitution-permutation network.	DES structure is based on feistal network.
5	Security	AES is de-facto world standard and is more secure than DES.	DES is weak and 3DES(Triple DES) is more secure than DES.
6	Rounds	Byte substitution, Shift Row, Mix Column and Key Addition.	Expansion, XOR operation with round key, Substitution and Permutation.
7	Size	AES can encrypt 128 bits of plain text.	DES can encrypt 64 bits of plain text.
8	Derived from	AES derives from Square cipher.	DES derives from Lucifer cipher.
9	Desiged By	AES was designed by Vincent Rijmen and Joan Daemen.	DES was designed by IBM.
10	Known attacks	No known attack.	Brute-force, Linear crypt-analysis and Differential crypt-analysis.

BLOWFISH Algorithm:

- One of the most popular Feistel network cipher is Blowfish. Blowfish is a symmetric-key block cipher proposed as a new encryption standard.
- It is a 16- round Feistel system, which uses large key-dependent S-boxes and iterates a simple encryption 16 times.
- The block size is 64 bits and the key-length may vary from 32 bits up to 448 bits.
- Although there is a complex initialization phase required before any encryption can take place, the actual encryption of data is very efficient on large microprocessors.
- Blowfish provides a good encryption rate in software and no effective cryptanalysis of it has been found to date.
- Blowfish symmetric block cipher algorithm encrypts block data of 64-bits at a time.
- The algorithm follows Feistel network

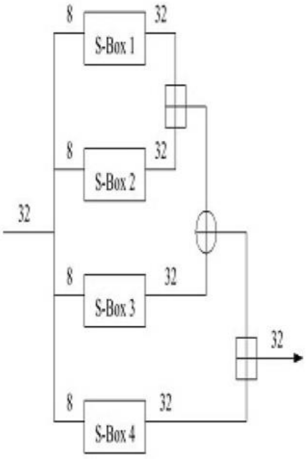
1. Key-expansion

2. Data Encryption

3. Data Decryption

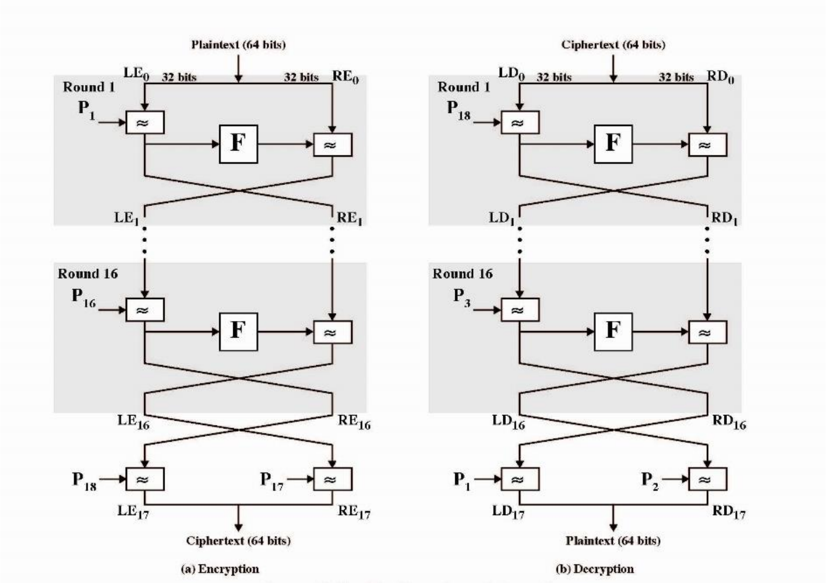
- Key Expansion
- uses a 32 to 448 bit key
- used to generate
- *18 32-bit subkeys stored in P-array: P1 to P18*
- *S-boxes stored in* $S_{i,j,}$
- i=1..4

- j=0..255



Data Encryption

It is having a function to iterate 16 times of network. Each round consists of key- dependent permutation and a key and data-dependent substitution. All operations are XORs and additions on 32-bit words. The only additional operations are four indexed array data lookup tables for each round.



(a) Encryption

(b) Decryption

- uses two primitives: addition & XOR
- data is divided into two 32-bit halves L_0 & R_0

```
for i = 1 to 16 do

$R_i$ = $L_{i-1}$ XOR $P_i$;

$L_i$ = F[$R_i$] XOR $R_{i-1}$;

$L_{17}$ = $R_{16}$ XOR $P_{18}$;
```

$R_{17} = L_{16}$ XOR i_{17};

- where

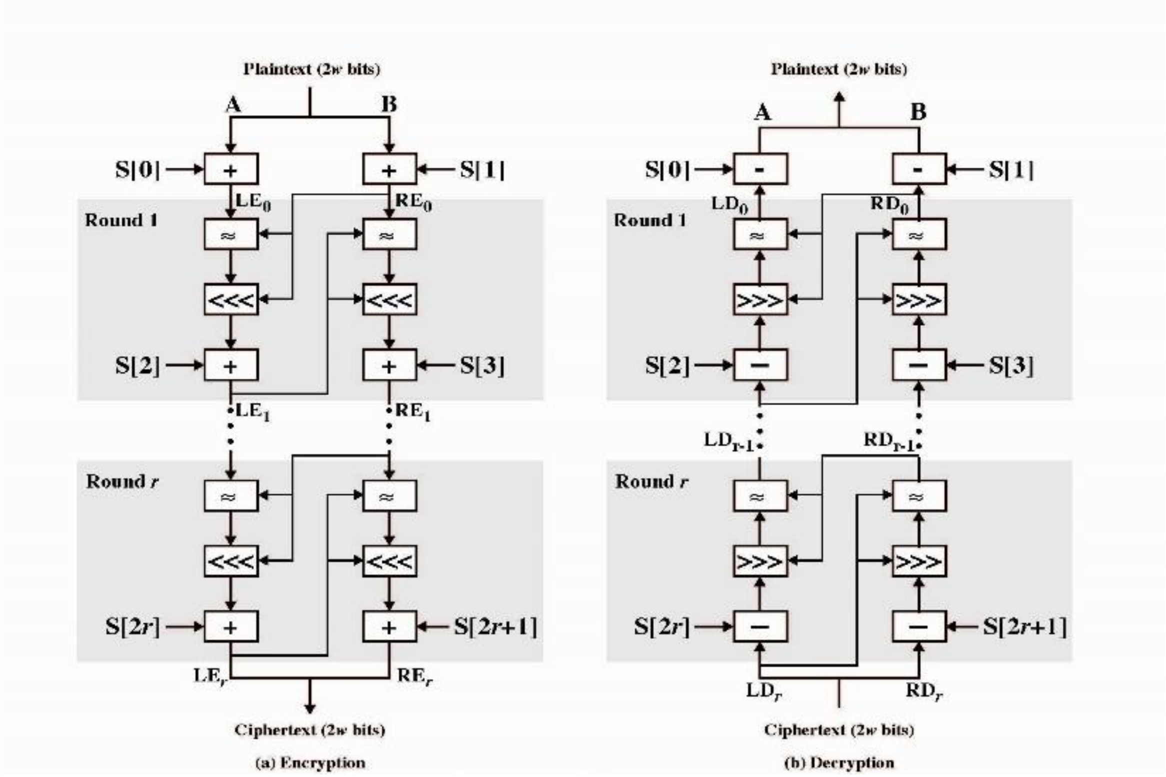
$F[a,b,c,d] = ((S_{1,a} + S_{2,b})$ XOR $S_{3,c}) + S_{4,a}$

Break 32-bit R_i into (a,b,c,d)

RC5 Encryption Algorithm

RC5 is a **symmetric** key block **encryption** algorithm designed by Ron Rivets in 1994. It is notable for being simple, fast (on account of using only primitive computer operations like XOR, shift, etc.) and consumes less memory. RC5 is a **block cipher** and addresses two-word blocks at a time.

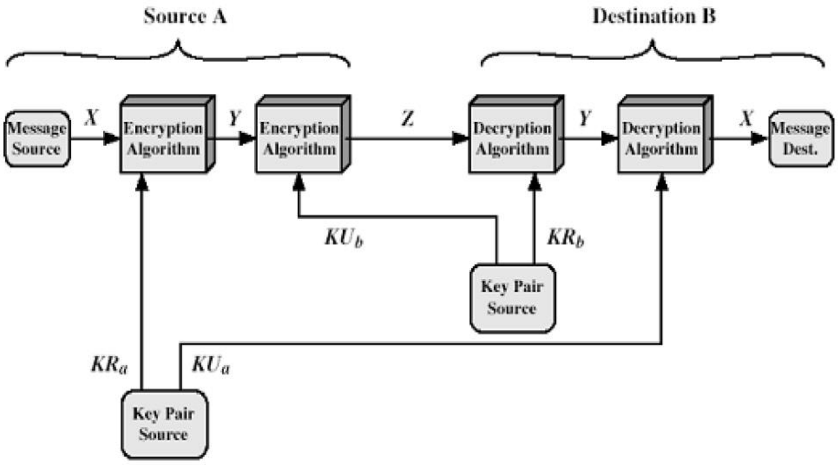
- It can vary key size / input data size / #rounds
- very clean and simple design
- easy implementation on various CPUs
- yet still regarded as secure
  - *Vary parameters to achieve tradeoffs*
- RC5 is a family of ciphers RC5-w/r/b
  - *w = word size in bits (16/32/64) data=2w*
  - *r = number of rounds (0..255)*
  - *b = number of bytes in key (0..255)*
- nominal version is RC5-32/12/16
  - *ie 32-bit words so encrypts 64-bit data blocks*
  - *using 12 rounds*
  - *with 16 bytes (128-bit) secret key*



- RC5 uses 2r+2 subkey words (w-bits)
    - *Two subkeys for each round*
    - *2 subkeys for additional operations*
- subkeys are stored in array S[i], i=0..t-1
- Key expansion: fill in pseudo-random bits to the original key K
- Certain amount of *one-wayness*
    - *Difficult to determine K from S*

Public Key Cryptography

- Public key cryptography (PKC) is an encryption technique that uses a paired public and private key (or asymmetric key) algorithm for secure data communication.
- PKC is also known as public key encryption, asymmetric encryption, asymmetric cryptography, asymmetric cipher, asymmetric key encryption and Diffie-Hellman encryption.
- A message sender uses a recipient's public key to encrypt a message. To decrypt the sender's message, only the recipient's private key may be used.
- The two types of PKC algorithms are RSA, which is an acronym named after this algorithm's inventors: Rivest, Shamir and Adelman, and Digital Signature Algorithm (DSA).
- PKC encryption evolved to meet the growing secure communication demands of multiple sectors and industries, such as the military.
- Features:
- Use two different keys, one for encryption another for decryption
- Plain text
- Encryption Algorithm
- Keys (two types)
- *Public key*
- *Private Key*
- Decryption Algorithm
- Cipher Key



$$\textbf{Ciphertext } Z = EKU_b\,[EKR_a\,(X)]$$

$$\textbf{Plaintext} \quad X = EKU_a[EKR_b\,(Y)]$$

- Each user generates a pair of keys to be used for encryption and decryption of messages.

- Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private.
- If A wishes to send a confidential message to B, A encrypts the message using B‟s public key.
- When B receives the message, it decrypts using its private key. No other recipient can decrypt the message because only B knows B‟s private key.
- Let the plaintext be X=[X1, X2, X3, …,Xm] where m is the number of letters in some finite alphabets.
- Suppose A wishes to send a message to B. B generates a pair of keys: a public key Kub and a private key KRb. KRb is known only to B, whereas KUb is publicly available and therefore accessible by A.
- With the message X and encryption key KUb as input, A forms the cipher text
- **Y=[Y1, Y2, Y3, … Yn]., i.e., Y=E KUb(X)**
- The receiver can decrypt it using the private key KRb. i.e., X=D KRb(). The encrypted message serves as a **digital signature.**
- It is important to emphasize that the encryption process just described does not provide confidentiality.
- There is no protection of confidentiality because any observer can decrypt the message by using the sender‟s public key.
- It is however, possible to provide both the authentication and confidentiality by a double use of the public scheme.
- Initially, the message is encrypted using the sender‟s private key. This provides the digital signature.
- Next, we encrypt again, using the receiver‟s public key. The final ciphertext can be decrypted only by the intended receiver, who alone has the matching private key. Thus, confidentiality is provided.
- **Requirements for public key cryptography**:
- The computation of the pair of keys i.e. private key and the public key must be easy.
- Knowing the encryption algorithm and public key of the intended receiver, computation of cipher text must be easy.
- For a receiver of the message, it should be computationally easy to decrypt the obtained cipher text using his private key.
- It is also required that any opponent in the network knowing the public key should be unable to determine its corresponding private key.
- Having the cipher text and public key an opponent should be unable to determine the original message.
- The two keys i.e. public and private key can be implemented in both orders D[PU, E(PR, M)] = D[PR, E(PU, M)]

RSA:

The pioneering paper by Diffie and Hellman [DIFF76b] introduced a new approach to cryptography and, in effect, challenged cryptologists to come up with a cryptographic algorithm that met the requirements for public-key systems. One of the first of the responses to the challenge was developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman at MIT and first published in 1978. The Rivest-Shamir- Adleman (RSA) scheme has since that time reigned supreme as the most widely accepted and implemented general-purpose approach to public-key encryption.

Description of the Algorithm:

The RSA scheme is a block cipher in which the plaintext and ciphertext are integers between 0 and *n* 1 for some *n*. A typical size for *n* is 1024 bits, or 309 decimal digits. That is, *n* is less than 2^{1024}. We examine RSA in this section in some detail, beginning with an explanation of the algorithm. Then we examine some of the computational and cryptanalytical implications of RSA.

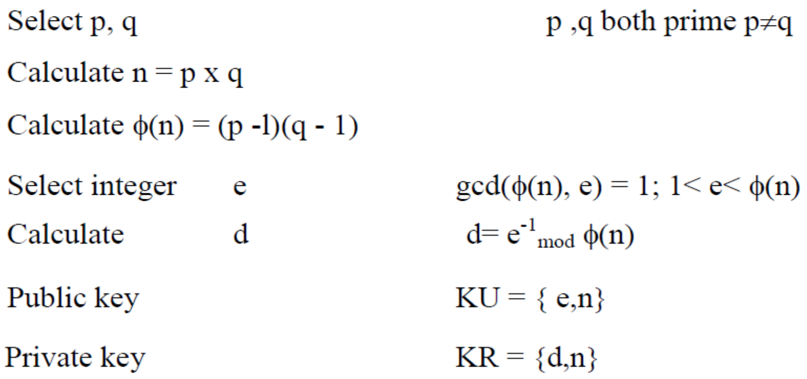
RSA makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number n. That is, the block size must be less than or equal to $\log_2(n) + 1$. Encryption and decryption for some plaintext block M and ciphertext block C
are:
$C = M^e \text{mod } n$
$M = C^d \text{mod } n = (M^e)^d \text{mod } n = M^{ed} \text{mod } n$
Both sender and receiver must know the value of n. The sender knows the value of e, and only the receiver knows the value of d. Thus, this is a public-key encryption algorithm with a public key of
$PU = \{e, n\}$ and a private key of $PR = \{d, n\}$.
For this algorithm to be satisfied public key encryption, the following requirements must be met.
1. It is possible to find values of e, d, and n such that $Med\text{mod } n = M$ for all $M < n$.
2. It is relatively easy to calculate $Me\text{mod } n$ and $Cd \text{ mod } n$ for all values of $M < n$.
3. It is infeasible to determine d given e and n.
By the first requirement, there is a need to find a relationship of the form
$M^{ed}\text{mod } n = M$
The preceding relationship holds if e and d are multiplicative inverses modulo $\phi(n)$, where $\phi(n)$ is the Euler totient function.
For any prime numbers p, q,
$\phi(pq) = (p - 1)(q - 1)$. The relationship between e and d can be expressed as $ed\text{mod } \phi(n) = 1$
This is equivalent to saying
$ed \equiv 1 \; mod \; \phi(n)$
$d \equiv e{-}1 mod \; \phi(n)$
That is, e and d are multiplicative inverses mod f(n). Note that, according to the rules of modular arithmetic, this is true only if d (and therefore e) is relatively prime to $\phi(n)$.
Equivalently, gcd($\phi(n)$, d) = 1.

The ingredients of the RSA schene are the following:

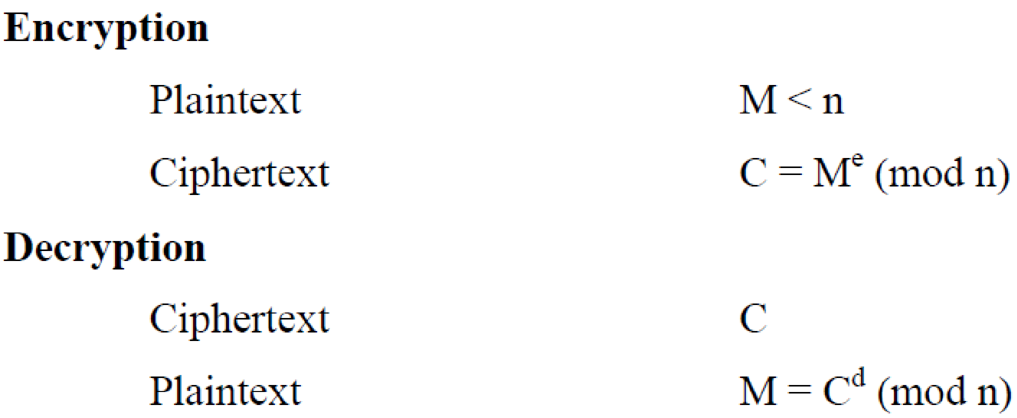
p, q, two prime numbers	(private, chosen)
$n = pq$	(public, calculated)
e, with gcd($\phi(n)$, e) = 1; $1 < e < \phi(n)$	(public, chosen)
$d \equiv e{-}1 mod \; \phi(n)$	(private, calculated)

The private key consists of {d, n} and the public key consists of {e, n}. Suppose that user A has published its public key and that user B wishes to send the message M to A. Then B calculates C = Me mod n and transmits C. On receipt of this ciphertext, user A decrypts by calculating M = C^d mod n.

Key Generation:



Select p, q		p ,q both prime p≠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	$d= e^{-1}{}_{\text{mod}}\ \phi(n)$
Public key		$KU = \{ e,n\}$
Private key		$KR = \{d,n\}$



Encryption

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

Decryption

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

- **Security of RSA**
- There are three approaches to attack the RSA:

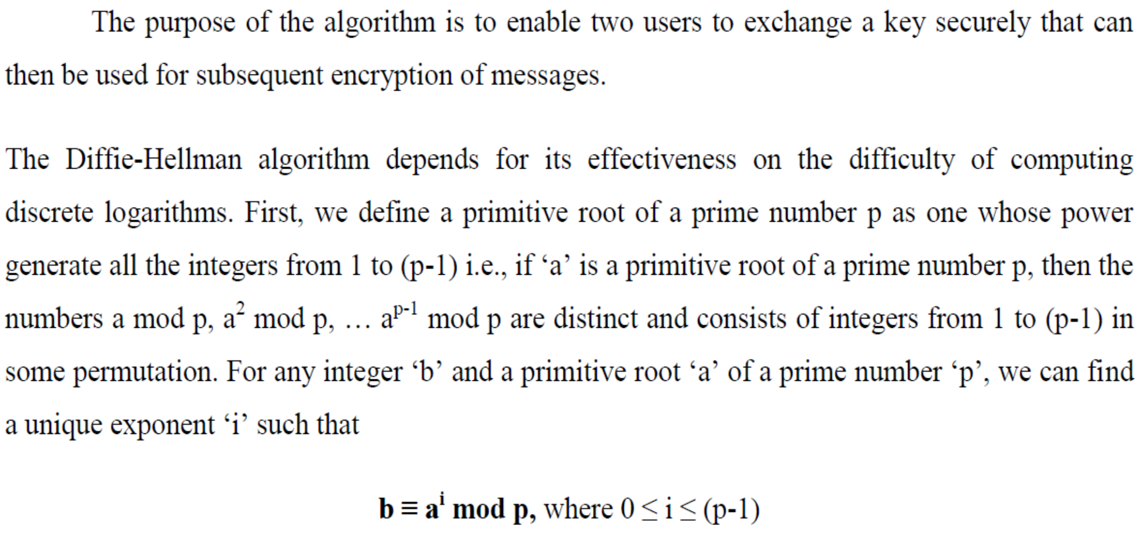
· brute force key search (infeasible given size of numbers)

· mathematical attacks (based on difficulty of computing ø(N), by factoring modulusN)

- timing attacks (on running time of decryption)

- **Factoring Problem**
- Mathematical approach takes 3 forms:
- · Factor n = p*q, hence find Φ(n) and then d.
- · Determine Φ(n)directly without determining p and q and find d.
- · Find d directly, without first determination Φ(n).
- **Timing attacks**
- It has been proved that the opponent can determine a private key by keeping track of how long a computer takes to decipher messages.
- Although the timing attack is a serious threat, there are simple countermeasures that can be used:

- · Constant exponentiation time – ensures that all exponentiations take the same amount of time before returning a result.
- · Random delay – better performance could be achieved by adding a random delay to the exponentiation algorithm to confuse the timing attack.
- · Blinding – multiply the ciphertext by a random number before performing exponentiation.
- **DIFFIE-HELLMAN KEY EXCHANGE**

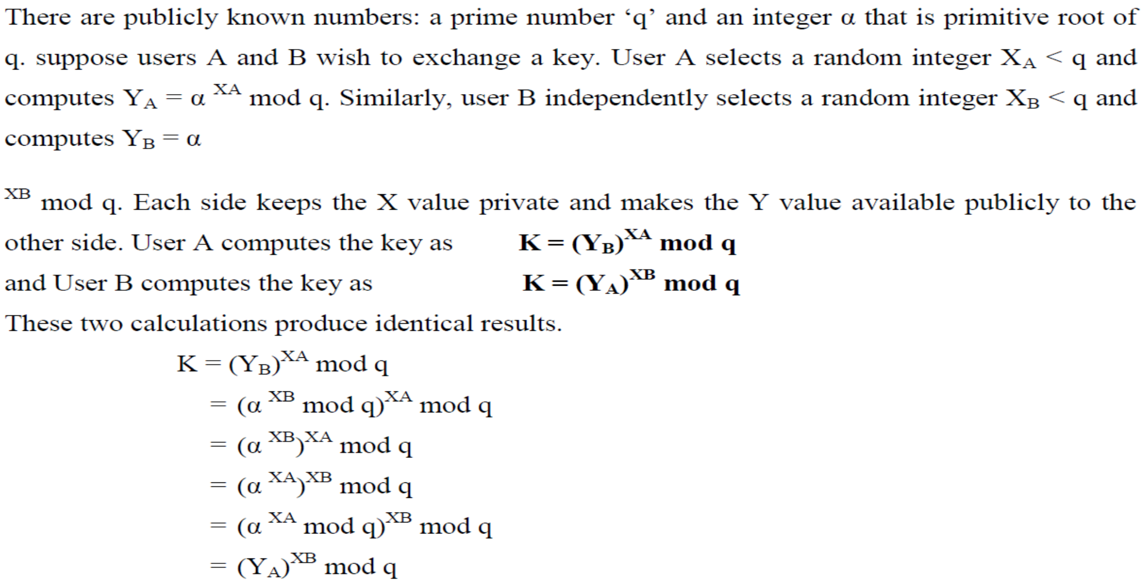


The purpose of the algorithm is to enable two users to exchange a key securely that can then be used for subsequent encryption of messages.

The Diffie-Hellman algorithm depends for its effectiveness on the difficulty of computing discrete logarithms. First, we define a primitive root of a prime number p as one whose power generate all the integers from 1 to (p-1) i.e., if ‘a’ is a primitive root of a prime number p, then the numbers a mod p, a^2 mod p, … a^{p-1} mod p are distinct and consists of integers from 1 to (p-1) in some permutation. For any integer ‘b’ and a primitive root ‘a’ of a prime number ‘p’, we can find a unique exponent ‘i’ such that

$$\mathbf{b \equiv a^i \bmod p,} \text{ where } 0 \leq i \leq (p-1)$$

- The exponent „i" is referred to as discrete logarithm



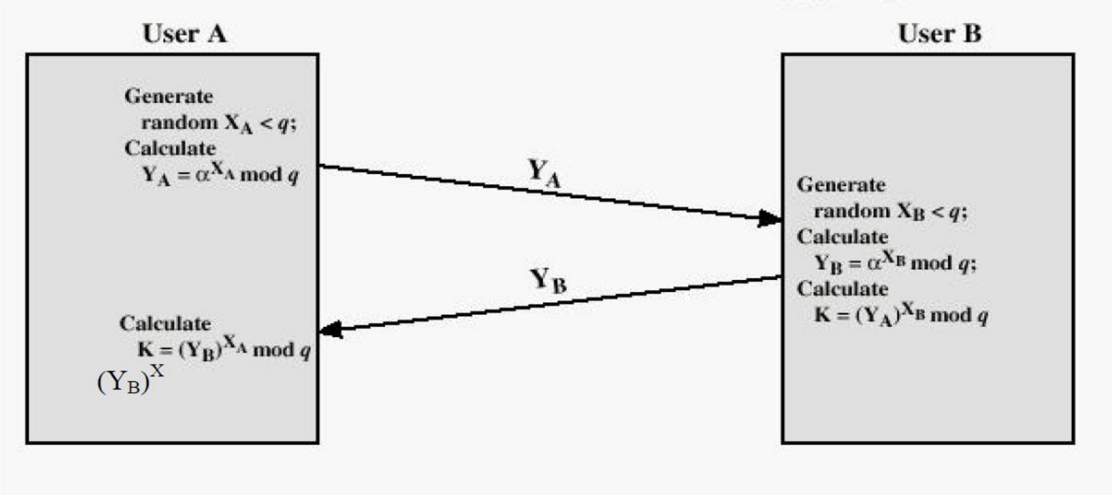
There are publicly known numbers: a prime number ‘q’ and an integer α that is primitive root of q. suppose users A and B wish to exchange a key. User A selects a random integer $X_A < q$ and computes $Y_A = \alpha^{XA} \bmod q$. Similarly, user B independently selects a random integer $X_B < q$ and computes $Y_B = \alpha$

^{XB} mod q. Each side keeps the X value private and makes the Y value available publicly to the other side. User A computes the key as $\mathbf{K = (Y_B)^{XA} \bmod q}$

and User B computes the key as $\mathbf{K = (Y_A)^{XB} \bmod q}$

These two calculations produce identical results.

$$\begin{aligned} K &= (Y_B)^{XA} \bmod q \\ &= (\alpha^{XB} \bmod q)^{XA} \bmod q \\ &= (\alpha^{XB})^{XA} \bmod q \\ &= (\alpha^{XA})^{XB} \bmod q \\ &= (\alpha^{XA} \bmod q)^{XB} \bmod q \\ &= (Y_A)^{XB} \bmod q \end{aligned}$$



The protocol depicted in figure is insecure against a **man-in-the-middle attack**. Suppose Alice and Bob wish to exchange keys, and Darth is the adversary. The attack proceeds as follows:

1. Darth prepares for the attack by generating two random private keys X_{D1} and X_{D2} and then computing the corresponding public keys Y_{D1} and Y_{D2}.
2. Alice transmits Y_A to Bob.
3. Darth intercepts Y_A and transmits Y_{D1} to Bob. Darth also calculates K2 = $(Y_A)^{X}{}_{D2}$ mod q.
4. Bob receives Y_{D1} and calculates K1 = $(Y_{D1})^{X}{}_{E}$ mod q. 5. Bob transmits X_A to Alice.
6. Darth intercepts X_A and transmits Y_{D2} to Alice. Darth calculates K1 =D1 mod q.
7. Alice receives Y_{D2} and calculates K2 = $(Y_{D2})^{X}{}_{A}$ mod q.

- At this point, Bob and Alice think that they share a secret key, but instead Bob and Darth share secret key K1 and Alice and Darth share secret key K2.
- All future communication between Bob and Alice is compromised in the following way:
- 1. Alice sends an encrypted message M: E(K2, M).
- 2. Darth intercepts the encrypted message and decrypts it, to recover M.
- 3. Darth sends Bob E(K1, M) or E(K1, M'), where M' is any message. In the first case, Darth simply wants to eavesdrop on the communication without altering it.

ELLIPTIC CURVE CRYPTOGRAPHY

Why ECC?

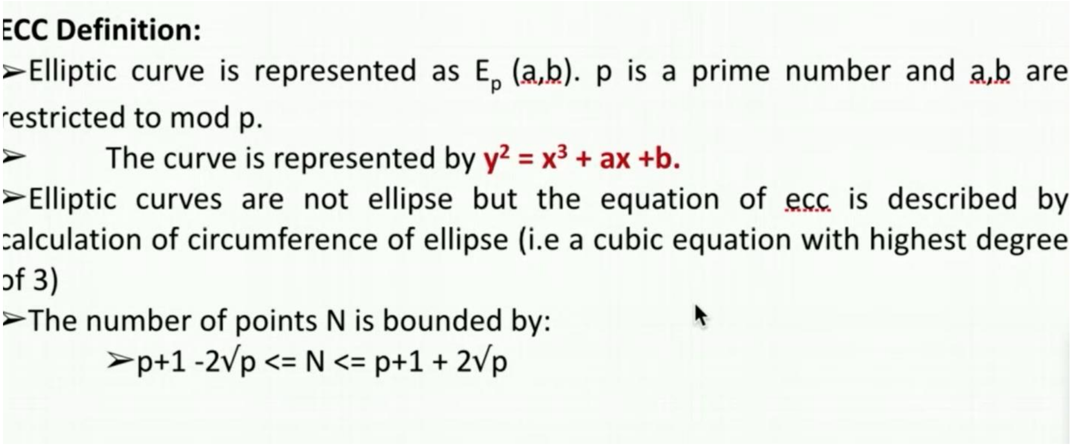
- ➢Good security even with far smallest key than RSA
- ➢Reduce the processing overhead

Basics for ECC:

Abelian Groups:

It should satisfy the following properties:

- ➢**Closure**
- ➢ **Associativity**
- ➢**Identity element**
- ➢**Inverse element**
- ➢**Commutativity**



ECC Definition:

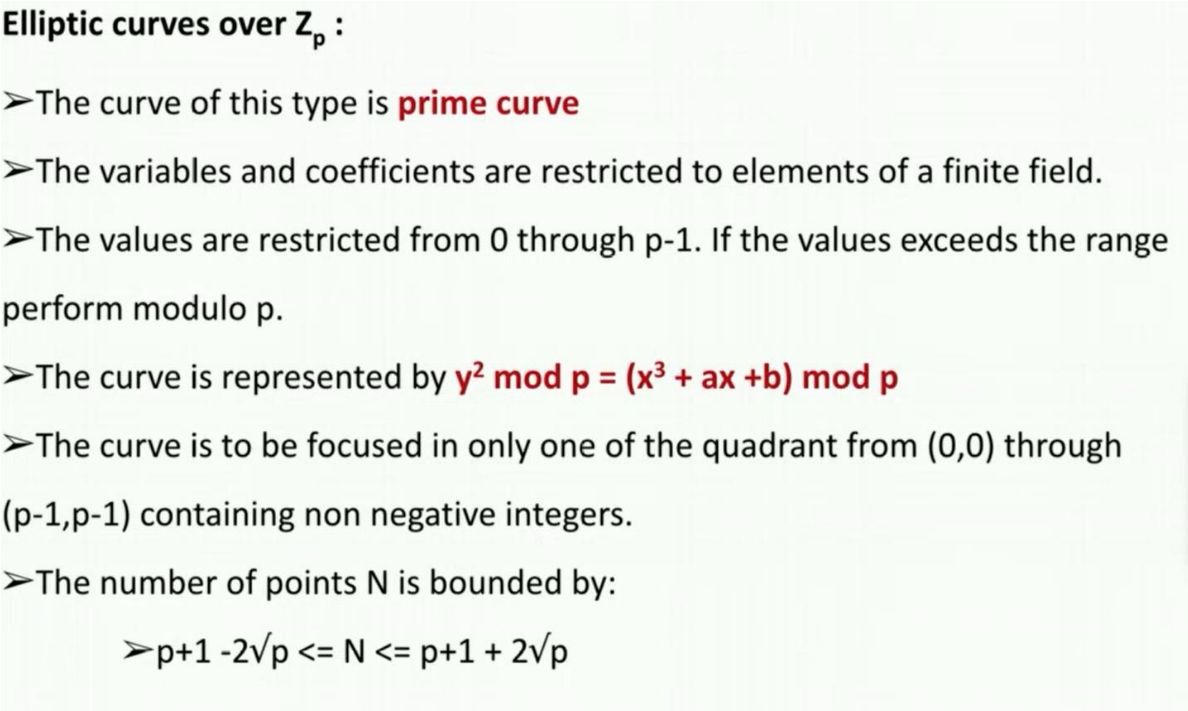
➢Elliptic curve is represented as E_p (a,b). p is a prime number and a,b are restricted to mod p.

➢ The curve is represented by $\mathbf{y^2 = x^3 + ax + b}$**.**

➢Elliptic curves are not ellipse but the equation of ecc is described by calculation of circumference of ellipse (i.e a cubic equation with highest degree of 3)

➢The number of points N is bounded by:

➢$p+1 - 2\sqrt{p} <= N <= p+1 + 2\sqrt{p}$



Elliptic curves over Z_p :

➢The curve of this type is **prime curve**

➢The variables and coefficients are restricted to elements of a finite field.

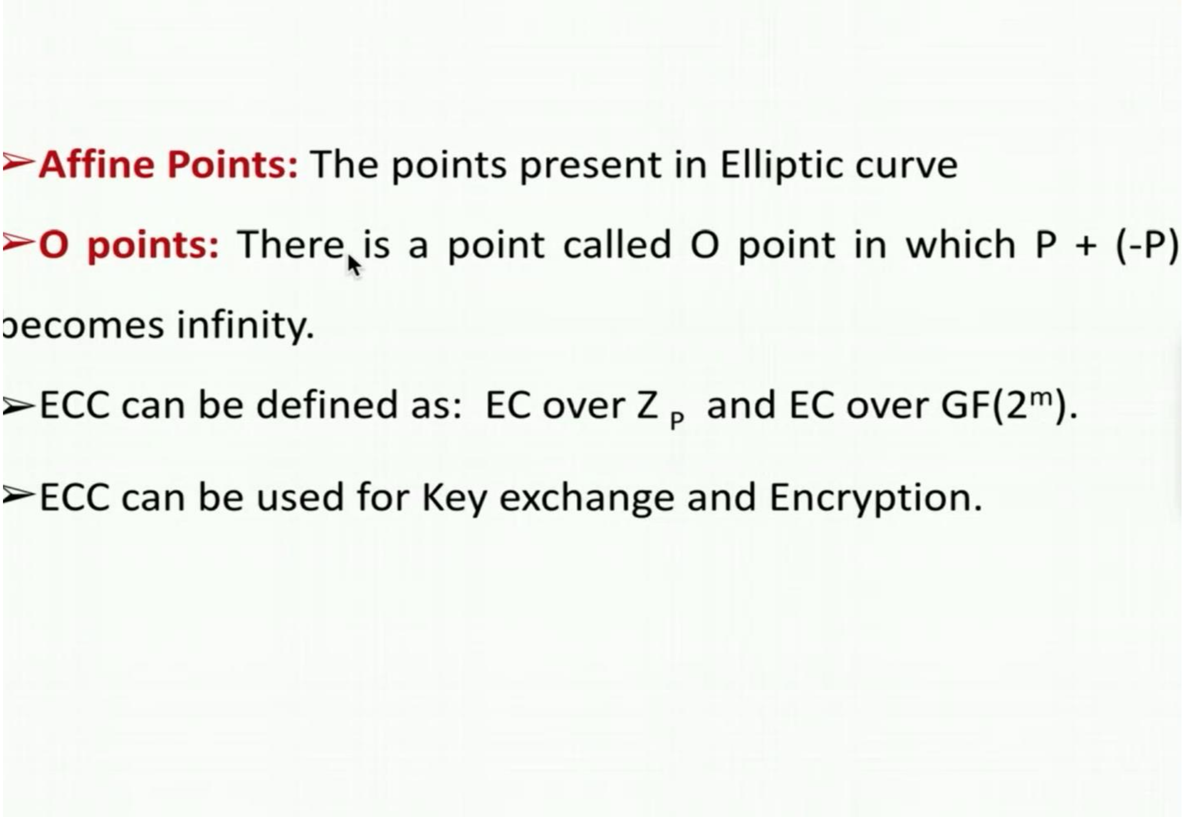
➢The values are restricted from 0 through p-1. If the values exceeds the range perform modulo p.

➢The curve is represented by **y^2 mod p = (x^3 + ax +b) mod p**

➢The curve is to be focused in only one of the quadrant from (0,0) through (p-1,p-1) containing non negative integers.

➢The number of points N is bounded by:

➢$p+1 -2\sqrt{p} <= N <= p+1 + 2\sqrt{p}$

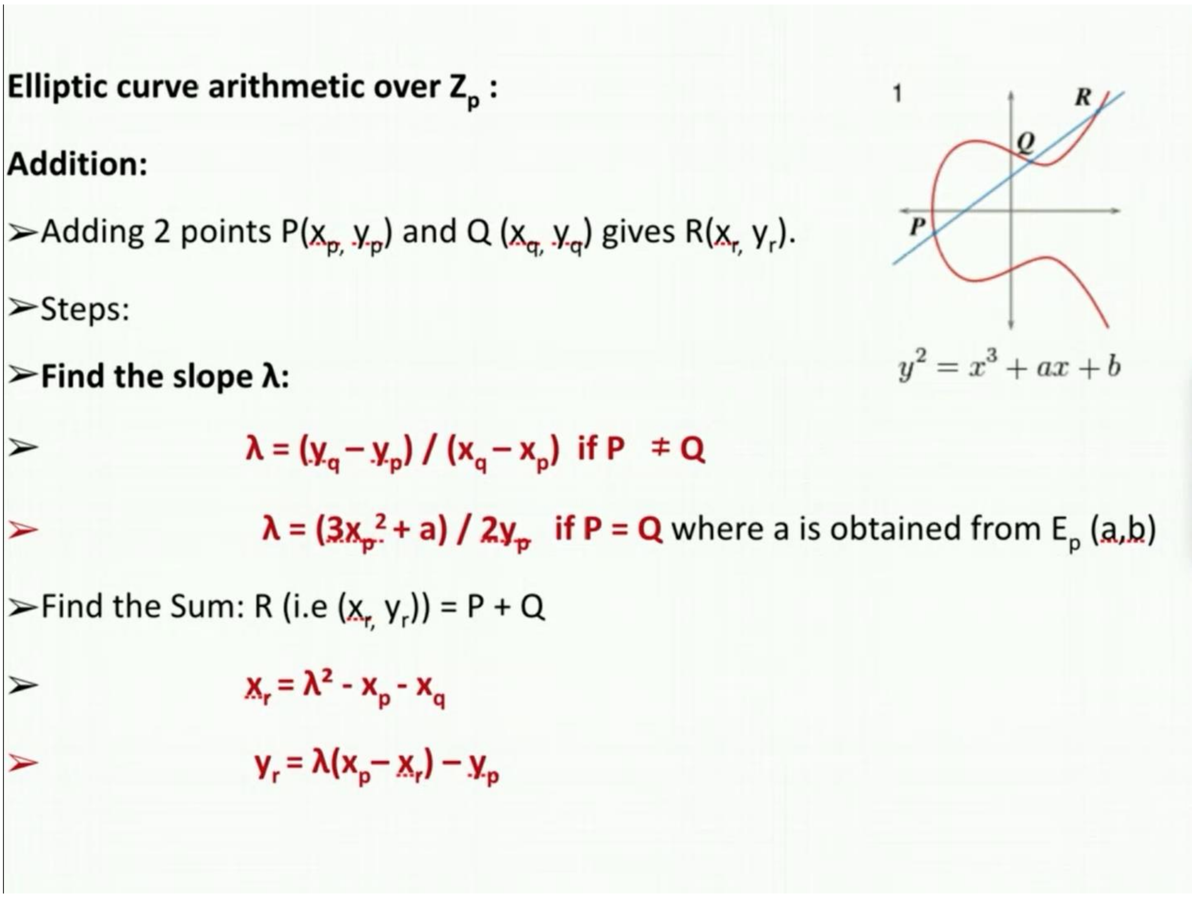


➢**Affine Points:** The points present in Elliptic curve

➢**O points:** There is a point called O point in which P + (-P) becomes infinity.

➢ECC can be defined as: EC over Z_P and EC over GF(2^m).

➢ECC can be used for Key exchange and Encryption.



Elliptic curve arithmetic over Z_p :

Addition:

➢Adding 2 points P(x_p, y_p) and Q (x_q, y_q) gives R(x_r, y_r).

➢Steps:

➢**Find the slope λ:**

➢ **$\lambda = (y_q - y_p) / (x_q - x_p)$ if P ∓ Q**

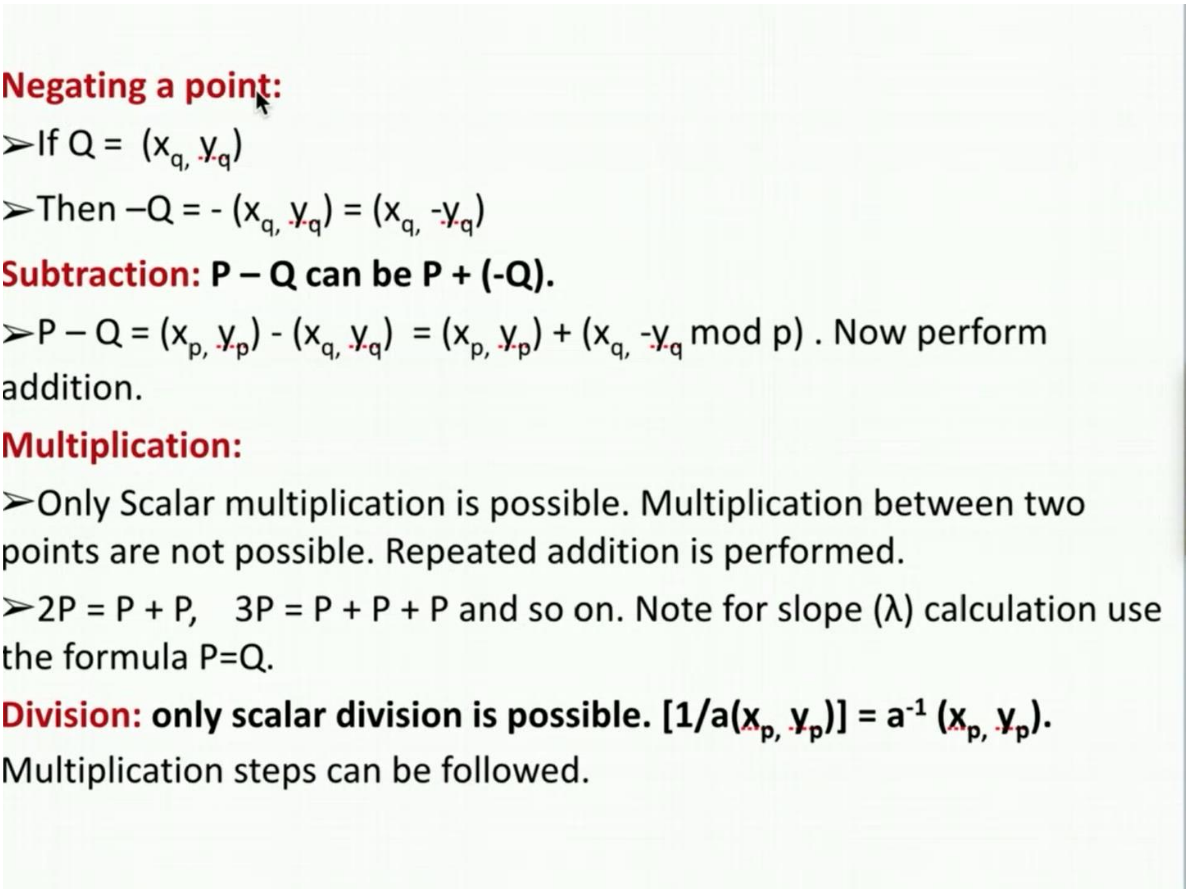
➢ **$\lambda = (3x_p^2 + a) / 2y_p$ if P = Q** where a is obtained from E_p (a,b)

➢Find the Sum: R (i.e (x_r, y_r)) = P + Q

➢ **$x_r = \lambda^2 - x_p - x_q$**

➢ **$y_r = \lambda(x_p - x_r) - y_p$**

1
R
Q
P
$y^2 = x^3 + ax + b$



Negating a point:

➢If Q = (x_q, y_q)

➢Then –Q = - (x_q, y_q) = (x_q, -y_q)

Subtraction: P – Q can be P + (-Q).

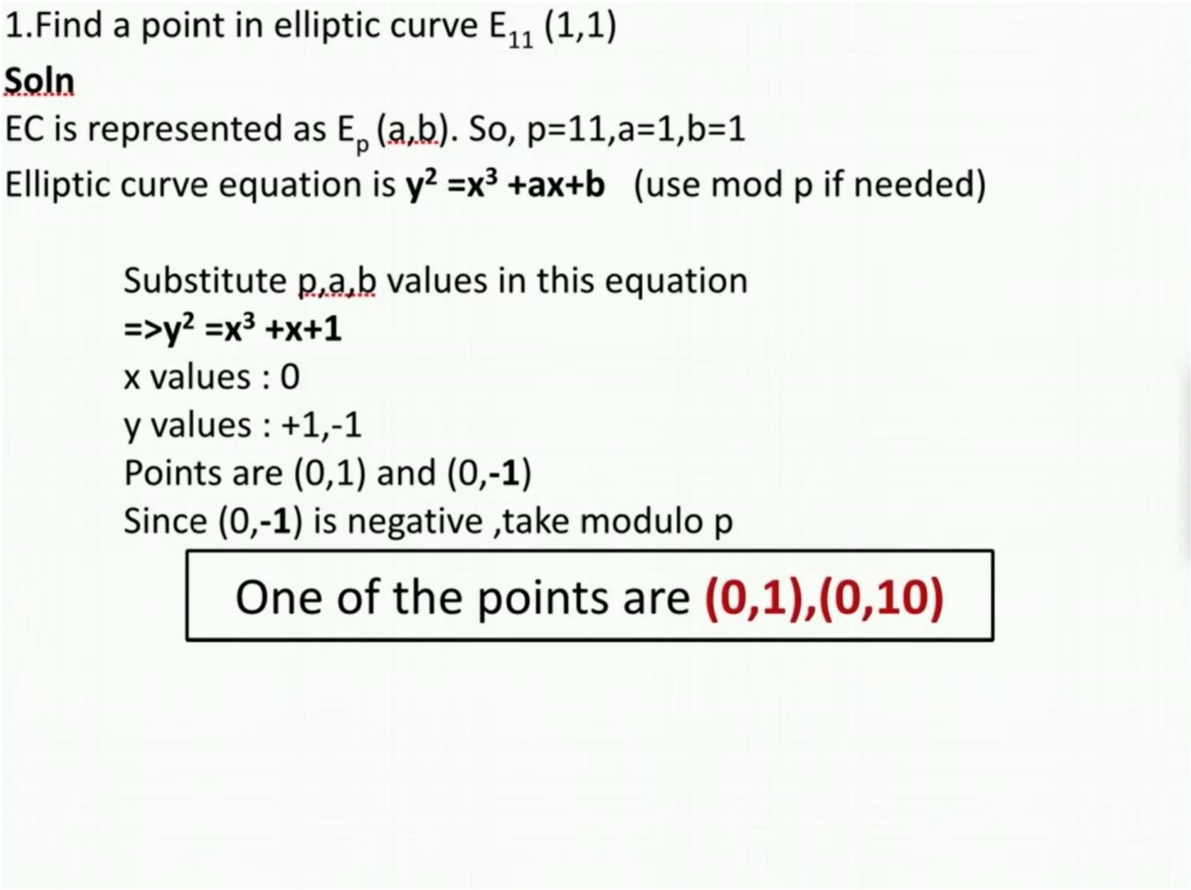
➢P – Q = (x_p, y_p) - (x_q, y_q) = (x_p, y_p) + (x_q, -y_q mod p) . Now perform addition.

Multiplication:

➢Only Scalar multiplication is possible. Multiplication between two points are not possible. Repeated addition is performed.

➢2P = P + P, 3P = P + P + P and so on. Note for slope (λ) calculation use the formula P=Q.

Division: only scalar division is possible. [1/a(x_p, y_p)] = a^{-1} (x_p, y_p). Multiplication steps can be followed.



1.Find a point in elliptic curve E_{11} (1,1)

Soln

EC is represented as E_p (a,b). So, p=11,a=1,b=1

Elliptic curve equation is $\mathbf{y^2 = x^3 + ax + b}$ (use mod p if needed)

Substitute p,a,b values in this equation

$\mathbf{\Rightarrow y^2 = x^3 + x + 1}$

x values : 0

y values : +1,-1

Points are (0,1) and (0,**-1**)

Since (0,**-1**) is negative ,take modulo p

One of the points are **(0,1),(0,10)**

UNIT – III HASH FUNCTIONS & DIGITAL SIGNATURES

Digests – Requirements – MAC – Hash function – Security of Hash and MAC – Birthday Attack – MD5 – SHA – RIPEMD – Digital Signature Standard – Proof of DSS

Authentication Requirements

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.
- the number and length of messages between parties could be determined on both environments

Masquerade

- Insertion of messages into the network from a fraudulent source.
- includes the creation of messages by an opponent that are purported to come from an authorized entity.
- Also included are fraudulent acknowledgments of message receipt or nonreceipt by someone else

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 a replay of some previous valid session, or individual messages in the sequence could be delayed or replayed.
- In a connectionless application, an individual message (e.g., datagram) could be delayed or replayed

Source repudiation

- Denial of transmission of message by source.

Destination repudiation

- Denial of receipt of message by destination

Authentication Functions

Message Authentication

- a mechanism or service used to verify the integrity of a message.
- assures that data received are exactly as sent (i.e., contain no modification, insertion, deletion, or

replay).

- assures that purported identity of the sender is valid.
- When a hash function is used to provide message authentication, the hash function value is often referred to as a message digest.

Authentication function is of two levels of functionality

Lower Level

produces an authenticator: a value to be used to authenticate a message.

Higher-Level

enables a receiver to verify the authenticity of a message

Grouped Into Three Classes

Message Encryption

The ciphertext of the entire message serves as its authenticator

Message authentication code (MAC)

A function of the message and a secret key that produces a fixed-length value that serves as the authenticator

Hash function

A function that maps a message of any length into a fixed-length hash value, which serves as the authenticator

Message Encryption

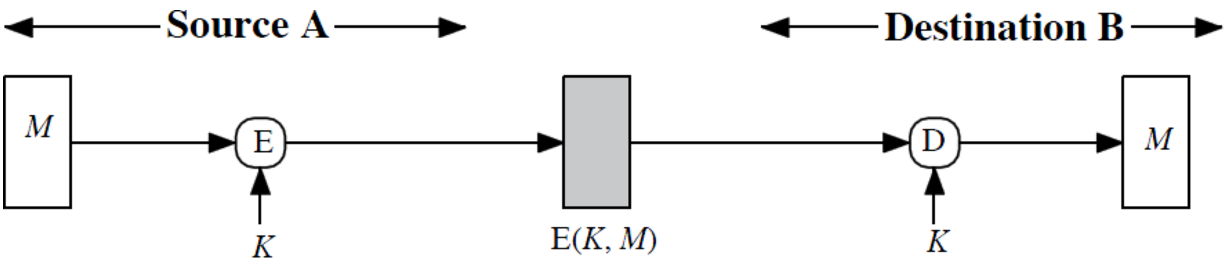
- Message encryption by itself can provide a measure of authentication.
- The analysis differs for symmetric and public-key encryption schemes

Topics

- Basic Uses of Message Encryption
- Symmetric Encryption
  - Internal Error Control
  - External Error Control
- Public-Key Encryption

Basic Uses of Message Encryption

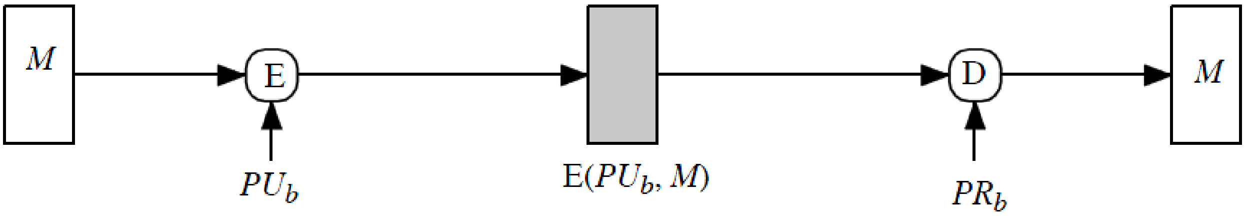
a) *Symmetric encryption: confidentiality and authentication: A --⍰ B:E(K, M)*



- Provides confidentiality
  - Only A and B share K
- Provides a degree of authentication
  - Could come only from A

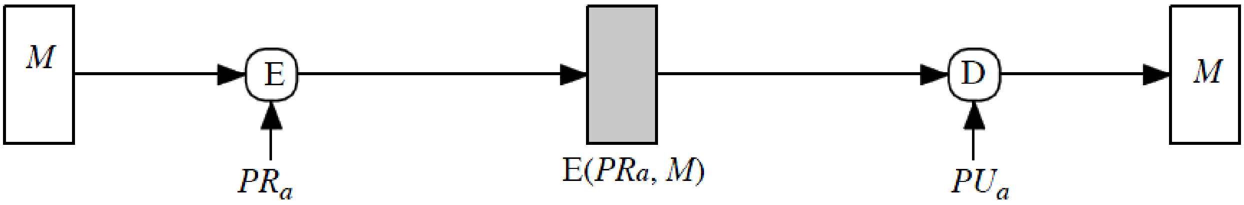
  - Has not been altered in transit
  - Requires some formatting/redundancy
- Does not provide signature
  - Receiver could forge message
  - Sender could deny message

***b)** Public-key encryption: confidentiality: A ⍰ B:E(PUb, M)*



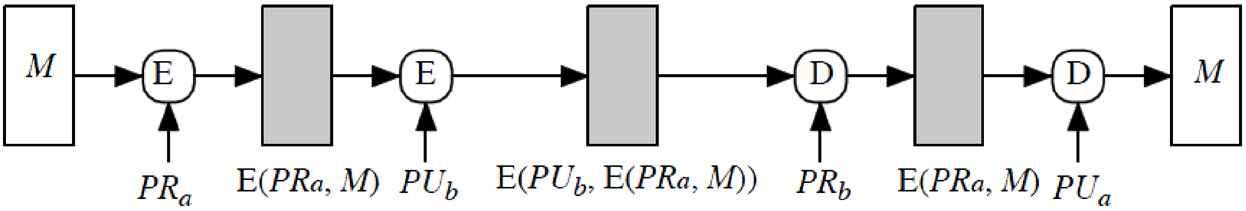
- Provides confidentiality
  - Only B has PRb to decrypt
- Provides no authentication
  - Any party could use PUb to encrypt message and claim to be A

***c)** Public-key encryption: authentication and signature: A ⍰ B:E(PRa, M)*



- Provides authentication and signature
  - Only A has PRa to encrypt
  - Has not been altered in transit
  - Requires some formatting/redundancy
  - Any party can use PUa to verify signature

***d)** Public-key encryption: confidentiality, authentication, and signature: A ⍰B:E(PUb, E(PRa, M))*



- Provides confidentiality because of Pub
- Provides authentication and signature because of Pra

Symmetric Encryption

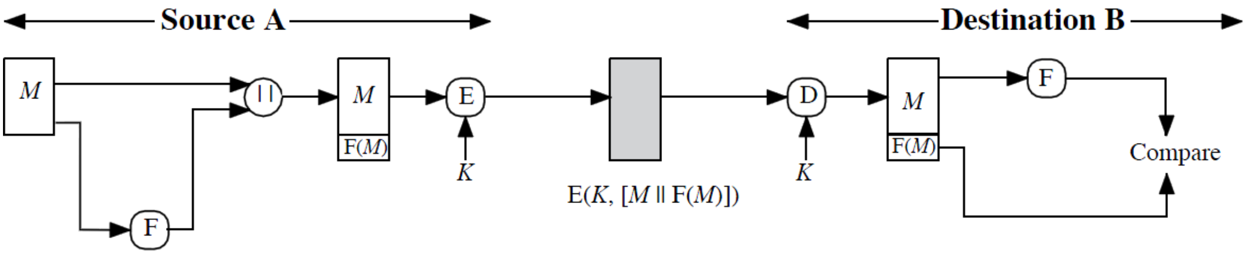
- A message M transmitted from source A to destination B is encrypted using a secret key K shared by both
- If no other party knows the key, then confidentiality is provided
- B is assured that the message was generated by A because A is the only other party that possesses K. Hence, authentication is provided.
- Hence, symmetric encryption provides authentication as well as confidentiality

- It may be difficult to determine automatically if incoming ciphertext decrypts to intelligible plaintext or not
    - an opponent could achieve a certain level of disruption

Solution to this problem

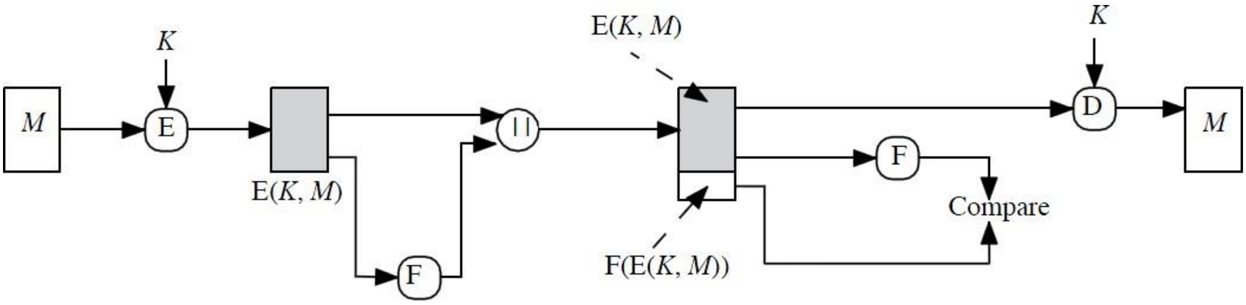
- force the plaintext to have some structure that is easily recognized but that cannot be replicated without recourse to the encryption function
- for example, append an error-detecting code, also known as a frame check sequence (FCS) or checksum, to each message before encryption
- the order in which the FCS and encryption functions are performed is critical
- Two classifications: Internal, External

Internal Error Control



- With internal error control, authentication is provided because an opponent would have difficulty generating ciphertext that, when decrypted, would have valid error control bits.
- If instead the FCS is the outer code, an opponent can construct messages with valid errorcontrol codes
- he or she can still hope to create confusion and disrupt operations

External Error Control



TCP Segment

- any sort of structuring added to the transmitted message serves to strengthen the authentication capability
- Such structure is provided by the use of a communications architecture consisting of layered protocols.
- As an example, consider the structure of messages transmitted using the TCP/IP protocol architecture
- each pair of hosts shared a unique secret key, so that all exchanges between a pair of hosts used the same key, regardless of application
- header includes not only a checksum (which covers the header) but also other useful information, such as the sequence number

Message Authentication Code

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

- This technique assumes that two communicating parties, say A and B, share a common secret key K.

Theory of operation

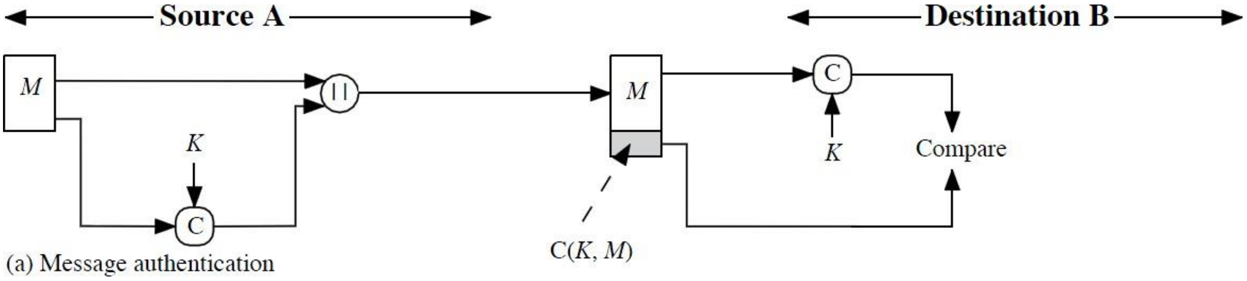
- When A has a message to send to B, it calculates the MAC as a function of the message and the key:
- MAC = C(K, M), where
  - M = input message
  - C = MAC function
  - K = shared secret key
  - MAC = message authentication code
- The message plus MAC are transmitted to the intended recipient.
- The recipient performs the same calculation on the received message, using the same secret key, to generate a new MAC.
- The received MAC is compared to the calculated MAC
- if the received MAC matches the calculated MAC, then
  - The receiver is assured that the message has not been altered
  - The receiver is assured that the message is from the alleged sender
  - If the message includes a sequence number (such as is used with HDLC, X.25, and TCP), then the receiver can be assured of the proper sequence

MAC function

- similar to encryption, difference is that the MAC algorithm need not be reversible
- many-to-one function
- The domain of the function consists of messages of some arbitrary length, whereas the range consists of all possible MACs and all possible keys
  - If an n-bit MAC is used, then there are 2^n possible MACs, whereas there are N possible messages with N >> 2^n
  - with a k-bit key, there are 2^k possible keys
- MAC does not provide a digital signature because both sender and receiver share the same key

Basic Uses of Message Authentication Code (MAC)

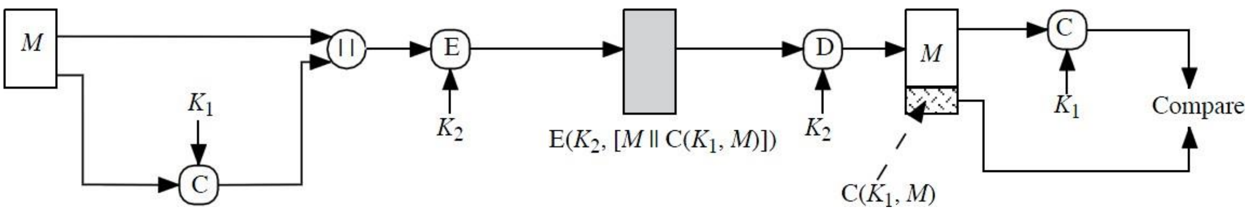
(a) Message authentication: A⍰B: M||C(K, M)



(a) Message authentication

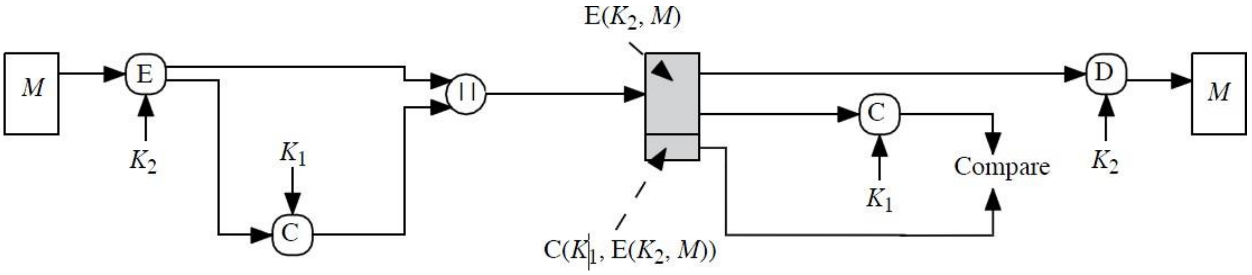
- Provides authentication: Only A and B share K

(b) Message authentication and confidentiality; authentication tied to plaintext



- A B:E(K2, [M||C(K, M)])
- Provides authentication
  - Only A and B share K1
- Provides confidentiality
  - Only A and B share K2

(c) Message authentication and confidentiality; authentication tied to ciphertext



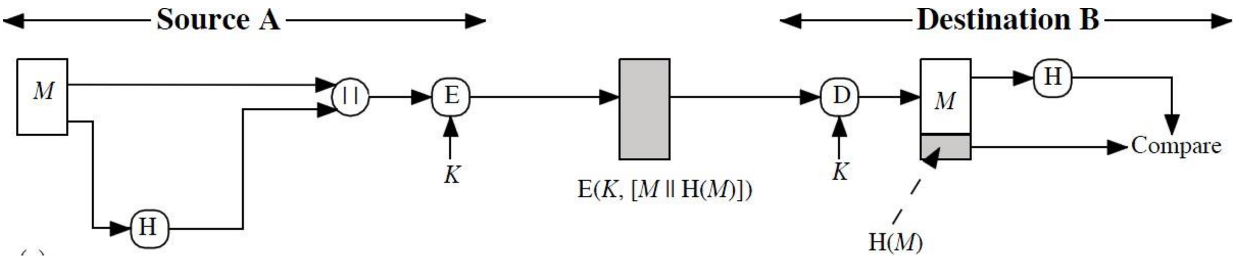
- A → B:E(K2, M)||C(K1, E(K2, M))
- Provides authentication
  - Using K1
- Provides confidentiality
  - Using K2

Hash Function

- a hash function accepts a variable-size message M as input and produces a fixedsize output, referred to as a hash code H(M).
- 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
- The hash code is a function of all the bits of the message and provides an error-detection capability:
- A change to any bit or bits in the message results in a change to the hash code

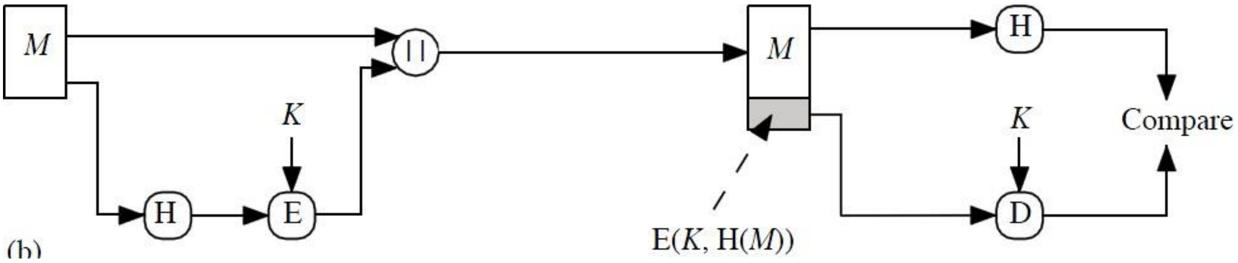
Basic Uses of Hash Function

a) Encrypt message plus hash code



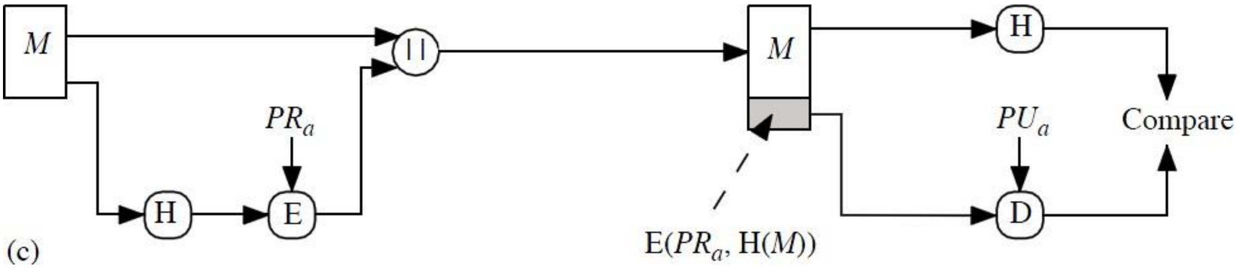
- A → B:E(K, [M||H(M)])
- Provides confidentiality
  - Only A and B share K
- Provides authentication
  - H(M) is cryptographically protected

(b) Encrypt hash code shared secret key



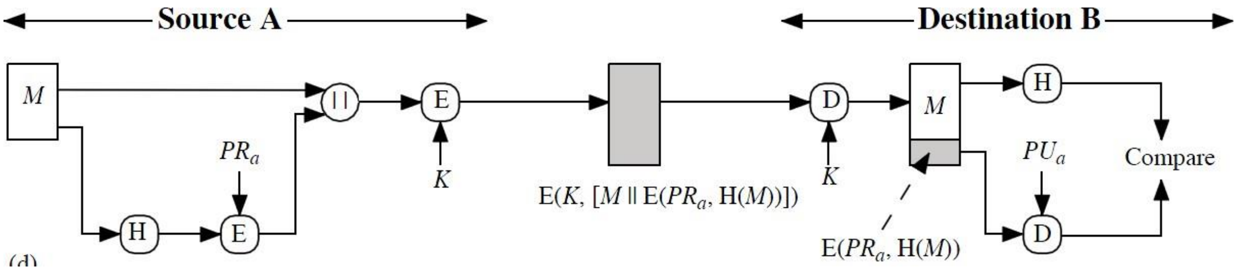
- A → B: M||E(K, H(M))
- Provides authentication
  - H(M) is cryptographically protected

(c) Encrypt hash code sender's private key



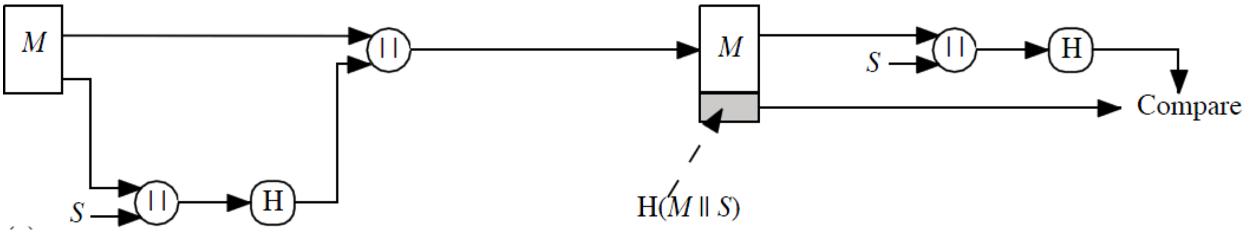
- A → B: M||E(PRa, H(M))
- Provides authentication and digital signature
  - H(M) is cryptographically protected
- Only A could create E(PRa, H(M))

(d) Encrypt result of (c)shared secret key



- A → B: E(K, [M||E(PRa, H(M))])
- Provides authentication and digital signature
  - Provides confidentiality
    - Only A and B share K

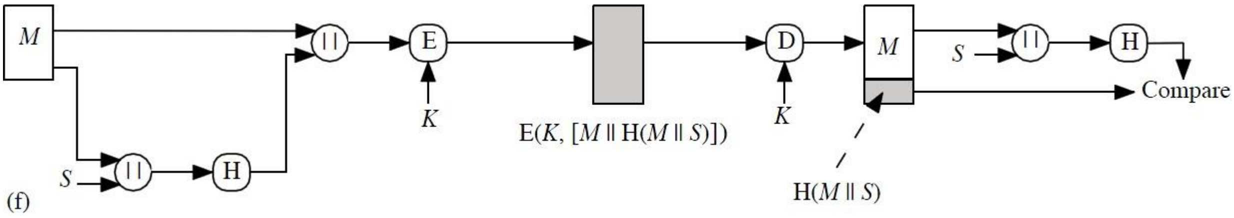
(e) Compute hash code of message plus secret value



- A → B: M||H(M||S)

- Provides authentication
    - Only A and B share S

(f) Encrypt result of (e)



- A B: E(K, [M||H(M||S])
- Provides authentication
    - Only A and B share S
- Provides confidentiality
    - Only A and B share K

Message Authentication Codes

- General Description
- Requirements for MACs
- Message Authentication Code Based on DES
- Data Authentication Algorithm

General Description

- A MAC, is also known as a cryptographic checksum
- It is generated by a function C of the form MAC = C(K, M)
- M is a variable-length message,
- K is a secret key shared only by sender and receiver
- C(K, M) is the fixed-length authenticator
- The MAC is appended to the message at the source
- The receiver authenticates that message by recomputing the MAC

Requirements for MACs

- the security of the scheme generally depends on the bit length of the key
- an attack will require $2^{(k-1)}$ attempts for a k-bit key
- for a ciphertext-only attack, the opponent, given ciphertext C, would perform Pi = D(Ki, C) for all possible key values Ki until a Pi was produced that matched the form of acceptable plaintext.
- the MAC function is a manyto-one function

MAC function should satisfy the following requirements

- 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

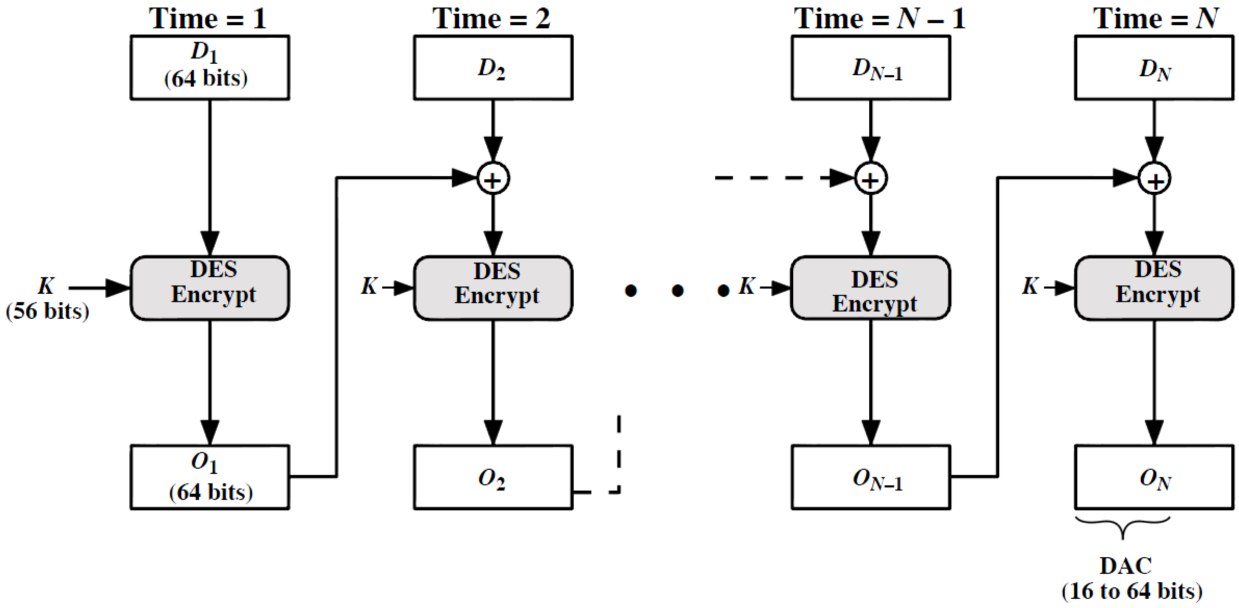
number of bits in the MAC.

- Let M' be equal to some known transformation on M. That is, M' = f(M).
  - For example, f may involve inverting one or more specific bits. In that case, Pr[C(K, M) = C(K, M')] = 2^n

Message Authentication Code Based on DES

- The Data Authentication Algorithm, based on DES, has been one of the most widely used MACs
- The algorithm can be defined as using the cipher block chaining (CBC) mode of operation of DES with an initialization vector of zero
- Data is grouped into contiguous 64-bit blocks: D1, D2,..., DN.
- the final block is padded on the right with zeroes to form a full 64-bit block
- Using the DES encryption algorithm, E, secret key, K, a data authentication code (DAC) is calculated as
  - O1 = E(K, D1)
  - O2 = E(K, [D2 + O1])
  - ...
  - O_N = E(K, [D_N + O_{N1}])

Data Authentication Algorithm (DAA)



The DAC consists of either the entire block O_N or the leftmost M bits of the block, with $16 \leq M \leq 64$.

Hash Functions

- Introduction
- Requirements for a Hash Function
- Simple Hash Functions
- Two Simple Hash Functions

- Birthday Attacks
- Block Chaining Techniques

Introduction

- A hash value h is generated by a function H of the form h = H(M)
- 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.
  - Because the hash function itself is not considered to be secret, some means is required to protect 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

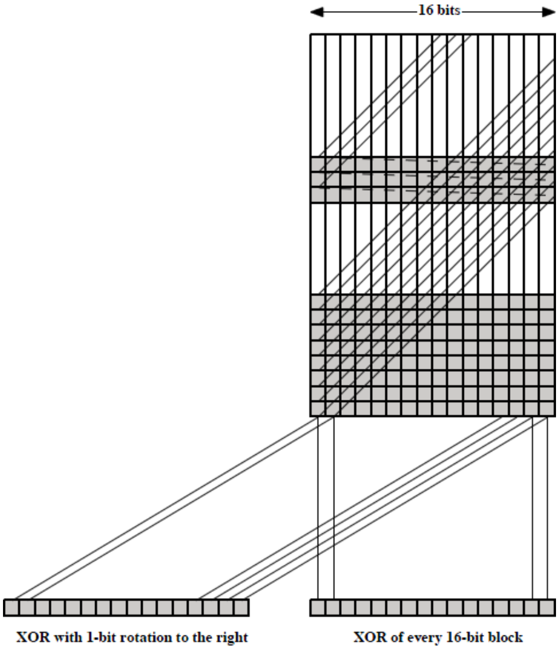
Simple Hash Functions

- 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.

Two Simple Hash Functions

Method 1

- bit-by-bit exclusive-OR (XOR) of every block.
- This can be expressed as follows
  - $C_i = b_{i1} \wedge b_{i2} .. \wedge . b_{im}$
  - where
    - Ci = ith bit of the hash code, 1≤ i ≤ n
    - m = number of n-bit blocks in the input
    - bij = ith bit in jth block
    - ^ = XOR operation


16 bits
XOR with 1-bit rotation to the right
XOR of every 16-bit block

Method 2

- Initially set the -bit hash value to zero.
- Process each successive -bit block of data as follows:
    - Rotate the current hash value to the left by one bit.
    - XOR the block into the hash value

Birthday Attacks

Birthday paradox

- The birthday paradox is often presented in elementary probability courses to demonstrate that probability results are sometimes counterintuitive.
- What is the minimum value of k such that the probability is greater than 0.5 that at least two people in a group of k people have the same birthday?
- Ignore February 29 and assume that each birthday is equally likely

Birthday Attack

- 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
- The opponent generates $2^{m/2}$ variations on the message, all of which convey essentially the same meaning
    - The opponent prepares an equal number of messages, all of which are variations on the fraudulent message to be substituted for the real one.
- 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
- 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

Meet in the middle attack possible

Security of Hash Functions and Macs

- Brute-force Attack
    - Hash Function
    - Message Authentication Codes
- Cryptanalysis
    - Hash Functions
    - Message Authentication Codes

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

there are three desirable properties

Property	Description	Effort Needed
One-way	For any given code h, it is computationally infeasible to find x such that H(x) = h	2^n
Weak collision resistance	For any given block x, it is computationally infeasible to find y ≠ x with H(y) = H(x).	2^n
Strong collision resistance	It is computationally infeasible to find any pair (x, y) such that H (x) = H(y).	$2^{n/2}$

Message Authentication Codes

- A brute-force attack on a MAC is a difficult undertaking because it requires known message-MAC pairs
- Security of the MAC algorithm depends on the relative size of the key and the MAC.
- we need to state the desired security property of a MAC algorithm

Computation resistance

Given one or more text-MAC pairs [xi, C(K, xi)], it is computationally infeasible to compute any text-MAC pair [x, C(K, x)] for any new input x ≠ xi

There are two lines of attack possible:

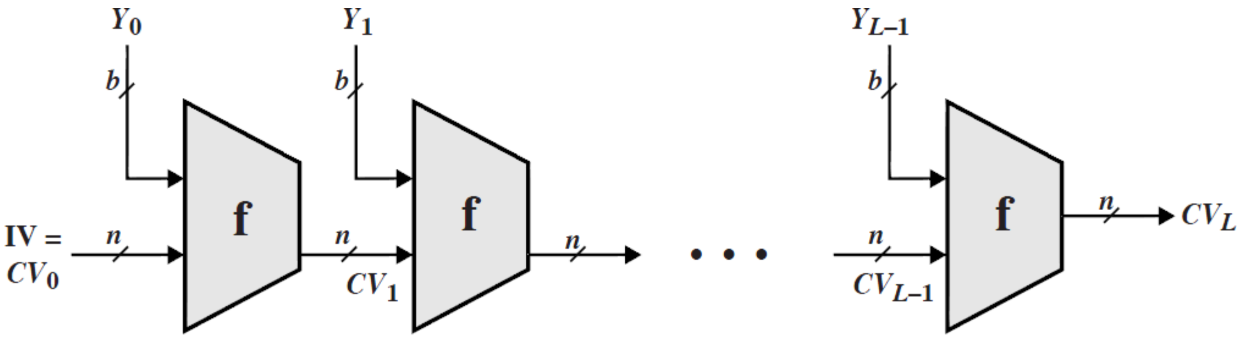
- Attack the key space
- attack the MAC value

Cryptanalysis

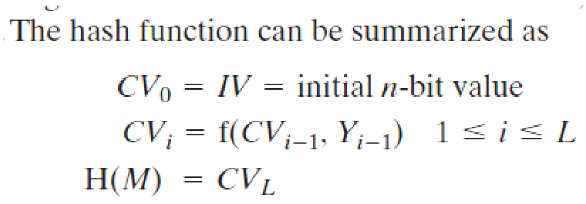
an ideal hash or MAC algorithm will require a cryptanalytic effort greater than or equal to the brute-force effort

Hash Functions

General Structure of Secure Hash Code



- The hash function takes an input message and partitions it into L fixed-sized blocks of b bits each.
- If necessary, the final block is padded to b bits.
- The final block also includes the value of the total length of the input to the hash function.
- The inclusion of the length makes the job of the opponent more difficult.
- 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.
- The hash algorithm involves repeated use of a compression function, f,


The hash function can be summarized as

$$CV_0 = IV = \text{initial } n\text{-bit value}$$

$$CV_i = \text{f}(CV_{i-1}, Y_{i-1}) \quad 1 \leq i \leq L$$

$$\text{H}(M) = CV_L$$

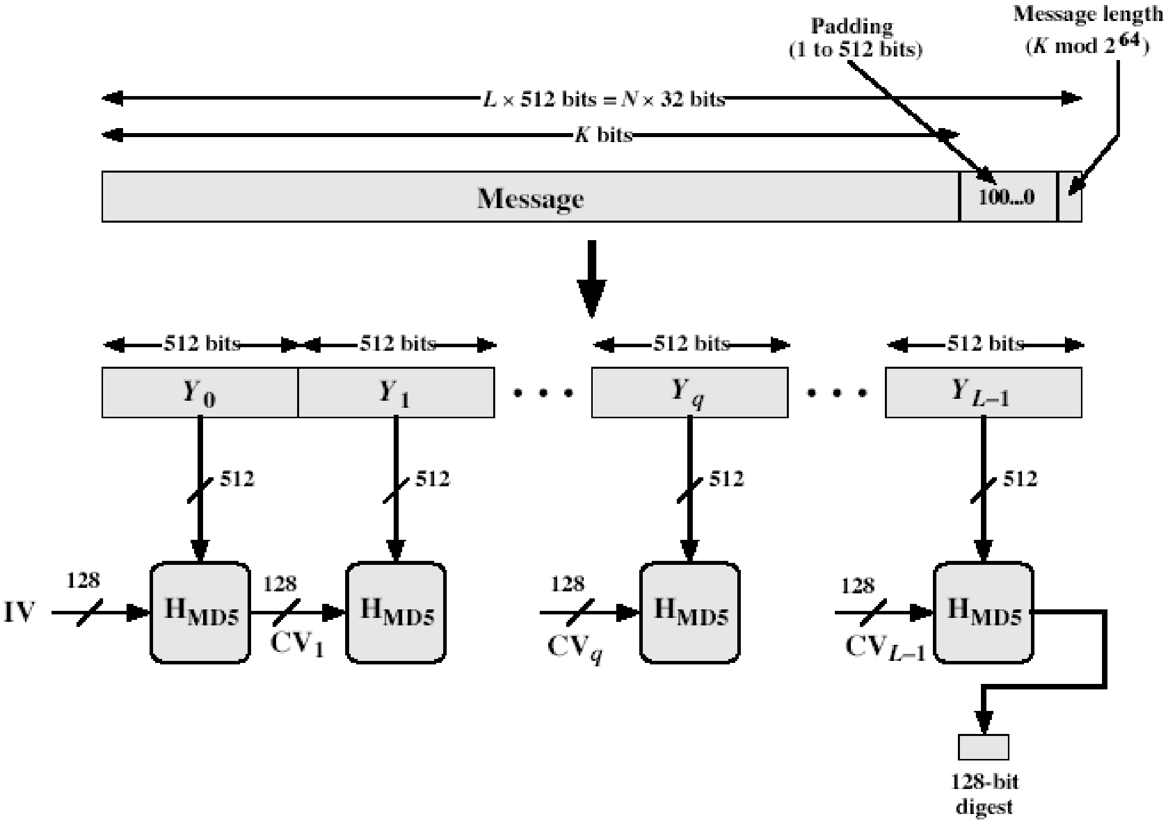
Message Authentication Codes

- There is much more variety in the structure of MACs than in hash functions
- Hence it is difficult to generalize about the cryptanalysis of MACs.
- far less work has been done on developing such attacks

MD5

MD5 Overview

- pad message so its length is 448 mod 512
- append a 64-bit length value to message
- initialise 4-word (128-bit) MD buffer (A,B,C,D)
- process message in 16-word (512-bit) blocks:
  - using 4 rounds of 16 bit operations on message block & buffer
  - add output to buffer input to form new buffer value
- output hash value is the final buffer value



MD5 Compression Function

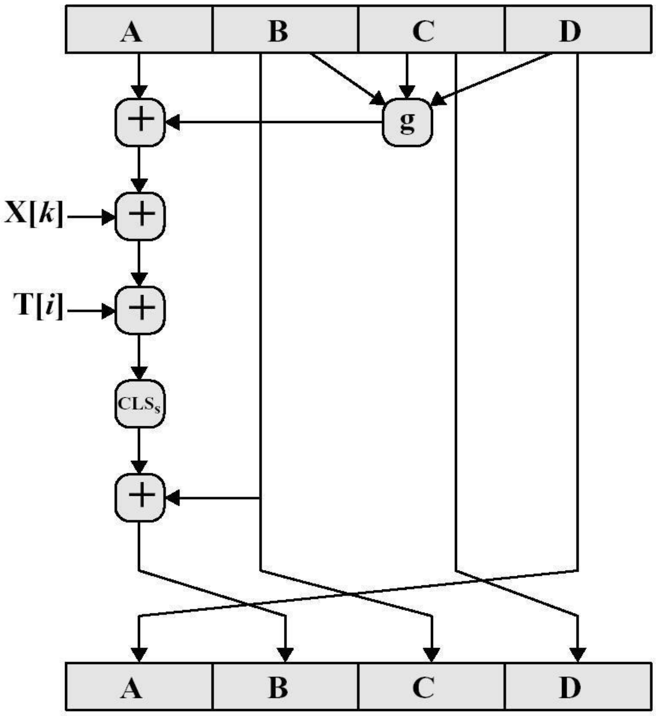


Figure 9.3 Elementary MD5 Operation (single step)

- each round has 16 steps of the form:
- a = b+((a+g(b,c,d)+X[k]+T[i])<<<s)
- a,b,c,d refer to the 4 words of the buffer
  - o this updates 1 word only of the buffer
  - o after 16 steps each word is updated 4 times
- where g(b,c,d) is a different nonlinear function in each round (F,G,H,I)
- T[i] is a constant value derived from sin

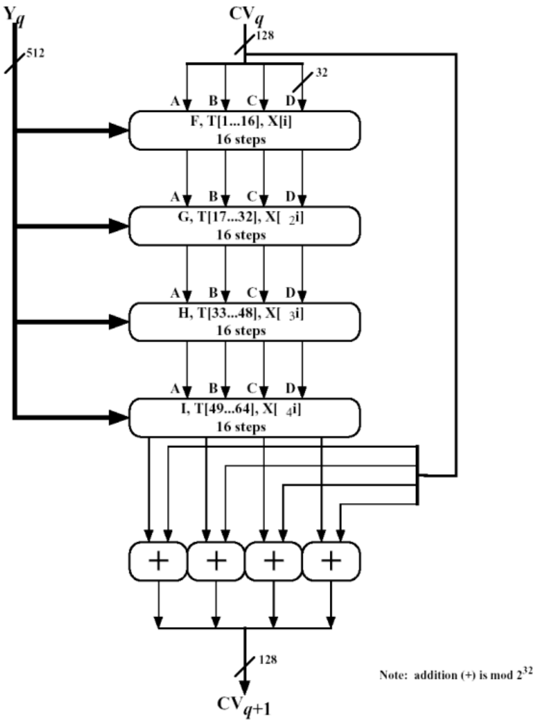
Strength of MD5

- MD5 hash is dependent on all message bits
- Rivest claims security is good as can be
- known attacks are:
  - o Berson 92 attacked any 1 round using differential cryptanalysis (but can't extend)
  - o Boer & Bosselaers 93 found a pseudo collision (again unable to extend)
  - o Dobbertin 96 created collisions on MD compression function (but initial constants prevent

exploit)

- conclusion is that MD5 looks vulnerable soon

Processing 512 Bit Block

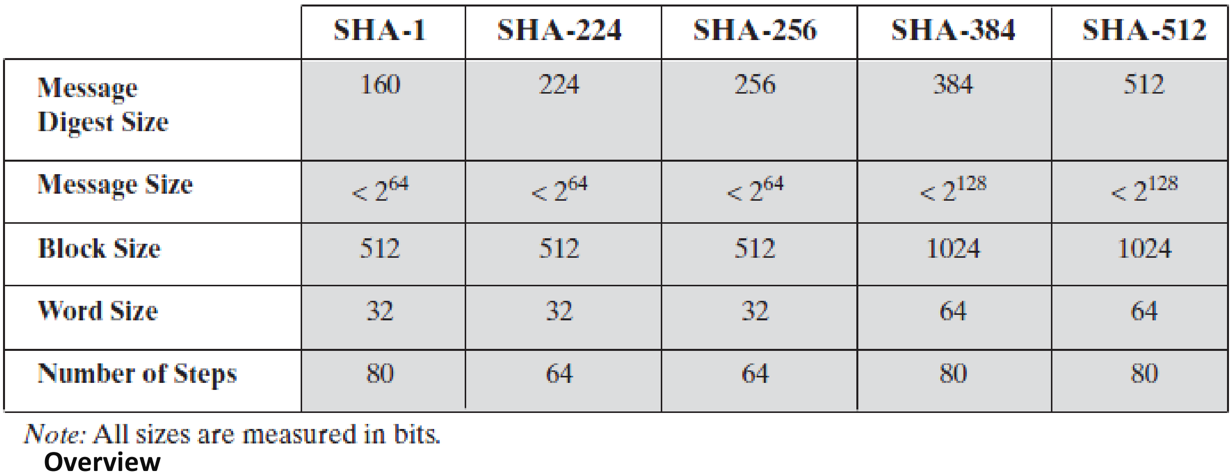


Note: addition (+) is mod 2^{32}

Figure 9.2 MD5 Processing of a Single 512-bit Block (MD5 Compression Function)

Secure Hash Algorithm (SHA)

Comparison of SHA Parameters



	SHA-1	SHA-224	SHA-256	SHA-384	SHA-512
Message Digest Size	160	224	256	384	512
Message Size	$< 2^{64}$	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block Size	512	512	512	1024	1024
Word Size	32	32	32	64	64
Number of Steps	80	64	64	80	80

Note: All sizes are measured in bits.

Overview

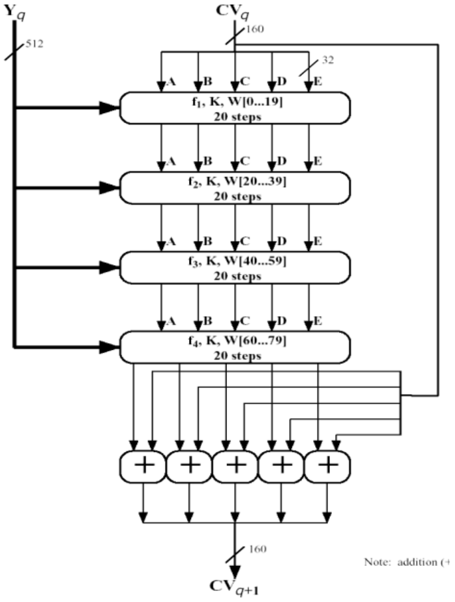
- most widely used hash function
- produces 160-bit hash values
- based on MD5
- The input is a message with length is < 2^64 bits

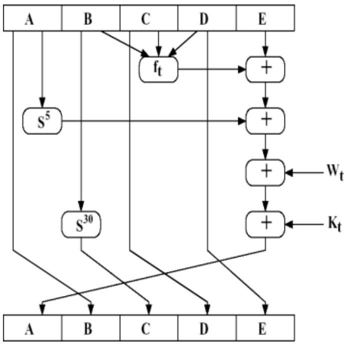
- The output is a message digest of length 160 bit
- The processing is done 512 bit blocks

Algorithm

Processing of a single 512-bit block

Single Round Function





Steps

Step 1: Append Padding Bits

Message is “padded” with a 1 and as many 0’s as necessary to bring the message length to 64 bits fewer than an even multiple of 512

Step 2: Append Length

64 bits are appended to the end of the padded message. These bits hold the binary format of 64 bits indicating the length of the original message

Step 3: Initialize MD buffer

- 160 bit buffer is used to hold the intermediate and final results
- The buffer is represented as 5 – 32 bit registes A, B, C, D, E
- Initialized as A = 0x67452301, B = 0xEFCDAB89, C = 0x98BADCFE, D = 0x10325476, E = 0xC3D2E1F0

Step 4: Process the message in 512 bit word blocks

Step 1: Append Padding Bits

Message is “padded” with a 1 and as many 0’s as necessary to bring the message length to 64 bits fewer than an even multiple of 512

Step 2: Append Length

64 bits are appended to the end of the padded message. These bits hold the binary

format of 64 bits indicating the length of the original message

Step 3: Initialize MD buffer

- 160 bit buffer is used to hold the intermediate and final results
- The buffer is represented as 5 – 32 bit registes A, B, C, D, E
- Initialized as A = 0x67452301, B = 0xEFCDAB89, C = 0x98BADCFE, D = 0x10325476, E = 0xC3D2E1F0

Step 4: Process the message in 512 bit word blocks

- The number of rounds = 4
- Each round has 20 steps
- Four different logical functions f1, f2, f3 and f4
- Each round makes use of additive constant Kt where 0 < t < 79
- 4 different constants are used 0x5A827999, 0x6ED9EBA1, 0x8F1BBCDC, 0xCA62C1D6
- The output of the 80th round is added to the input to the 1st round to produce the output

Step 5: Output

- After all 512 bit block is processed, the output from the Lth stage is 160 bit message digest
- CV0 = IV
- Yq+1 = sum32 (CVq, ABCDEq)
- MD = CVL
- IV: Initial value of the ABCDE buffer
- ABCDEq = Output of the last round of processing of the qth block
- L = The no of blocks (after padding and appending length)
- Sum32 = Addition modulo 2^32 done on each word separately
- MD = Final Message Digest

SHA-1 Compression Function

Describe the picture

Comparison of SHA-1 and MD5

Security against Brute Force Attack

- SHA, operations are 2 ^ 160
- MD5, operations are 2 ^ 128
- It is difficult to have same message digest for 2 different messages
- brute force attack is harder for SHA-1(160 vs 128 bits for MD5)

Security against crypt analysis

- MD5 is vulnerable, SHA1 is not vulnerable

Speed

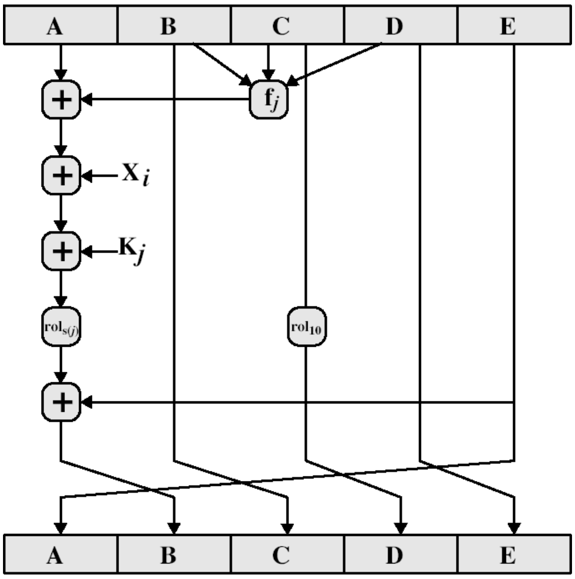
- MD5 executes faster when compared to SHA1 due to less bits needed, 64 steps versus 80 steps
- Both depend on addition modulo of 2^32 and could run well on 32 bit processors

Simplicity and Compactness

- Both are simple to describe and implement

- MD5 uses little endian, SHA 1 uses big endian

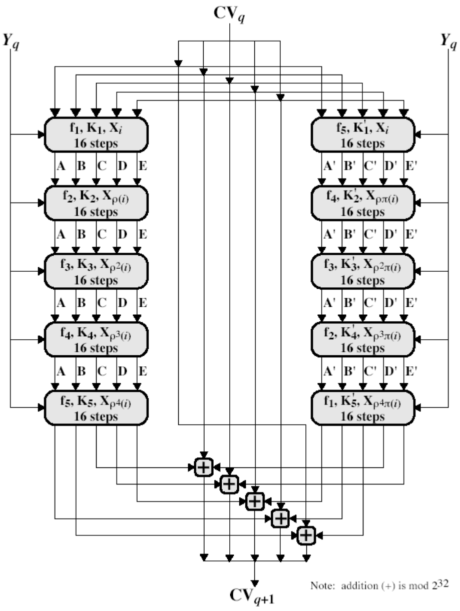
RACE Integrity Primitives Evaluation Message Digest (RIPEMD)



- RIPEMD-160 was developed in Europe as part of RIPE project in 96
- by researchers involved in attacks on MD4/5
- initial proposal strengthen following analysis to become RIPEMD-160
- somewhat similar to MD5/SHA
- uses 2 parallel lines of 5 rounds of 16 steps
- creates a 160-bit hash value
- slower, but probably more secure, than SHA

RIPEMD-160 Overview

- pad message so its length is 448 mod 512
- append a 64-bit length value to message
- initialise 5-word (160-bit) buffer (A,B,C,D,E) to
- (67452301, efcdab89,98badcfe,
- 10325476, c3d2e1f0)
- process message in 16-word (512-bit) chunks:
- use 10 rounds of 16-bit operations on message block & buffer – in 2 parallel lines of 5
- add output to input to form new buffer value
- output hash value is the final buffer value



RIPEMD-160 Compression Function

Refer Picture

RIPEMD-160 Design Criteria

- use 2 parallel lines of 5 rounds for increased complexity
- for simplicity the 2 lines are very similar
- step operation very close to MD5
- permutation varies parts of message used
- circular shifts designed for best results

RIPEMD-160 verses MD5 & SHA-1

- brute force attack harder (160 like SHA-1 vs 128 bits for MD5)
- not vulnerable to known attacks, like SHA- 1 though stronger (compared to MD4/5)
- slower than MD5 (more steps)
- all designed as simple and compact
- SHA-1 optimised for big endian CPU's vs
  RIPEMD-160 & MD5 optimised for little endian CPU’s

Digital Signature Standard

- Introduction
- Digital Signature
    - Requirements

    - Categories
- Digital Signature Standard
- Approaches to Digital Signatures
    - DSS Approach
    - RSA Approach
- The Digital Signature Algorithm
- DSS Signing and Verifying

Introduction

- A digital signature is an authentication mechanism that enables the creator of a message to attach a code that acts as a signature. The signature is formed by taking the hash of the message and encrypting the message with the creator's private key.
- The signature guarantees the source and integrity of the message.
- Mutual authentication protocols enable communicating parties to satisfy themselves mutually about each other's identity and to exchange session keys.
- In one-way authentication, the recipient wants some assurance that a message is from the alleged sender.
- The digital signature standard (DSS) is an NIST standard that uses the secure hash algorithm (SHA).

Digital Signature

Requirements

- 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.

Two categories

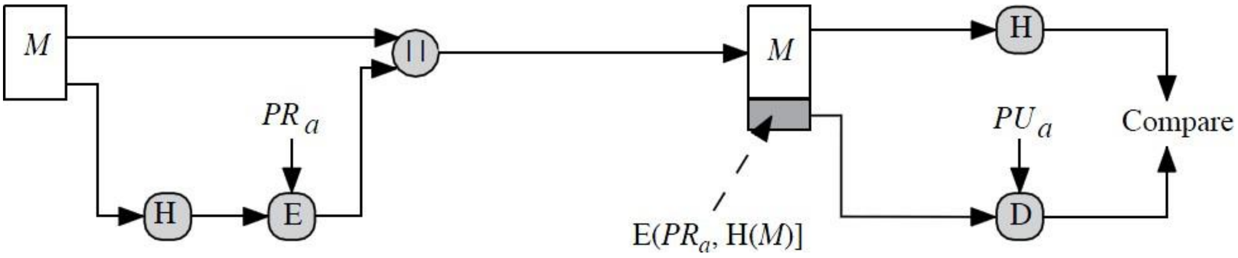
- Direct Digital Signature
- Arbitrated Digital Signature

Digital Signature Standard

- The National Institute of Standards and Technology (NIST) has published Federal Information Processing Standard FIPS 186, known as the Digital Signature Standard (DSS).
- The DSS makes use of the Secure Hash Algorithm (SHA)
- DSS presents a new digital signature technique, the Digital Signature Algorithm (DSA)
- latest version also incorporates digital signature algorithms based on RSA and on elliptic curve cryptography

Two Approaches to Digital Signatures

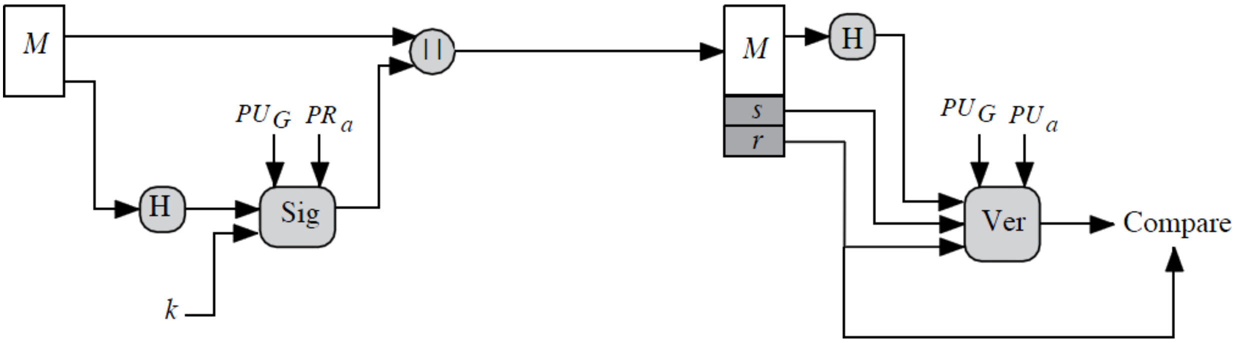
RSA Approach



- In the RSA approach, the message to be signed is input to a hash function that produces a secure hash code of fixed length.
- This hash code is then encrypted using the sender's private key to form the signature.
- Both the 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

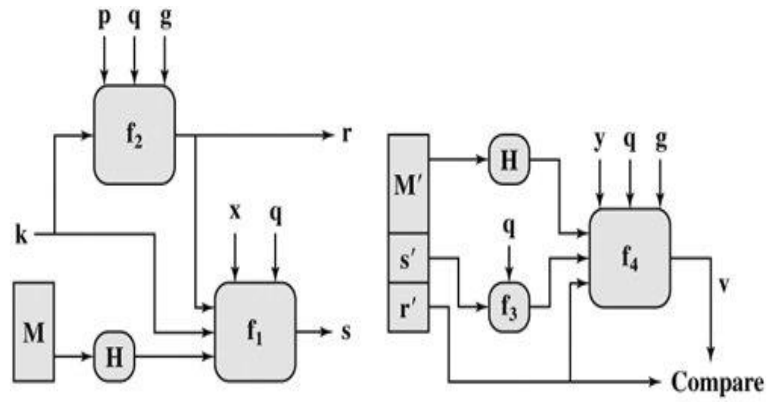
- The DSS uses an algorithm that is designed to provide only the digital signature function
- cannot be used for encryption or key exchange
- it is a public-key technique



The Digital Signature Algorithm

- Global Public-Key Components
- User's Private Key: x
  - random or pseudorandom integer with 0 < x < q
- User's Public Key: y
  - g^x mod p
- User's Per-Message Secret Number: k
  - random or pseudorandom integer with 0 < k < q
- Signing
  - r = (g^k mod p) mod q
  - s = [k^-1 (H(M) + xr)] mod q
  - Signature = (r, s)
- Verifying

DSS Signing and Verifying



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

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

(a) Signing

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

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

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

(b) Verifying

UNIT – IV SECURITY PRACTICE & SECURITY SYSTEM

Authentication applications

- authentication functions developed to support application-level authentication & digital signatures
- Kerberos – a private-key authentication service
- X.509 - a public-key directory authentication service

Kerberos

Overview

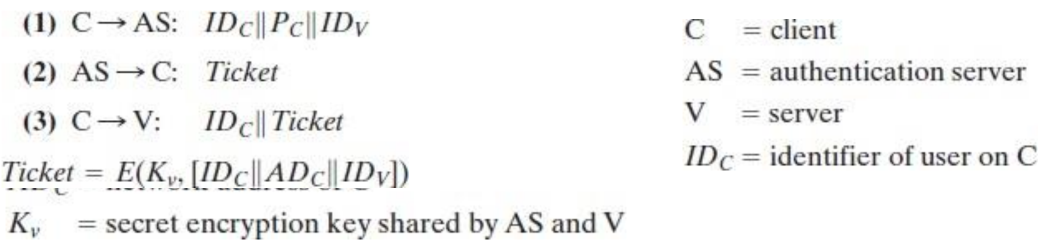
- authentication service designed for use in a **distributed** environment.
- makes use of a trusted **third-party authentication** service
  - enables clients and servers to establish authenticated communication.
- developed as part of Project Athena at MIT
- Addresses the following threats
  - user pretend to be another user operating from that workstation
  - user may alter the network address and impersonate the workstation
  - eavesdrop on exchanges and use a replay attack for gaining entry or disrupt
- provides a centralized authentication server to authenticate users to servers and servers to users
- relies exclusively on symmetric encryption, making no use of public-key encryption
- two versions in use 4 & 5

Motivation / Requirements (SRTS)

- **Secure**
  - A network eavesdropper should not be able to obtain the necessary information to impersonate a user.
  - strong enough such that a potential opponent does not find it to be the weak link.
- Reliable
  - should be highly reliable
  - should employ a distributed server architecture,
    - one system able to back up another.
- Transparent
  - user should not be aware that authentication is taking place
    - beyond the requirement to enter a password.
- Scalable
  - capable of supporting large numbers of clients and servers
    - modular, distributed architecture.

Kerberos Encryption Techniques

Simple Kerberos Dialogue



(1) C → AS: $ID_C \| P_C \| ID_V$

(2) AS → C: $Ticket$

(3) C → V: $ID_C \| Ticket$

$Ticket = E(K_v, [ID_C \| AD_C \| ID_V])$

K_v = secret encryption key shared by AS and V

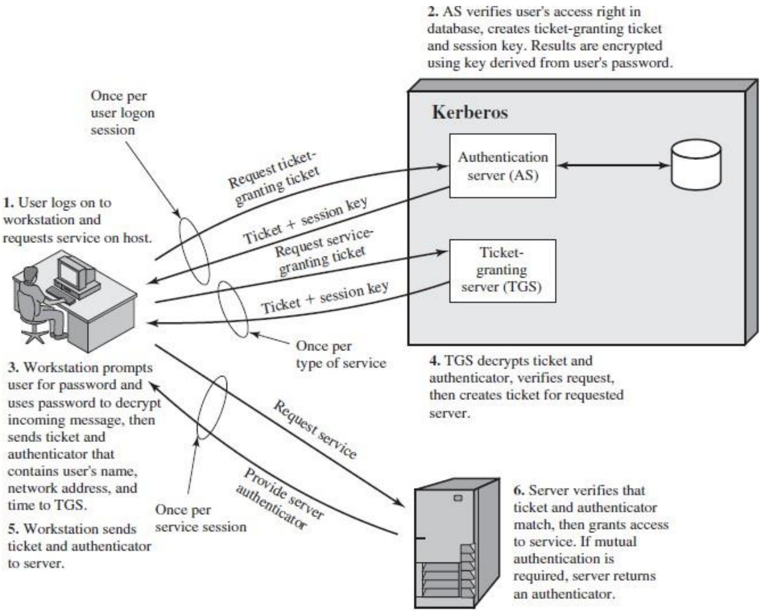
C = client

AS = authentication server

V = server

ID_C = identifier of user on C

Kerberos Overview

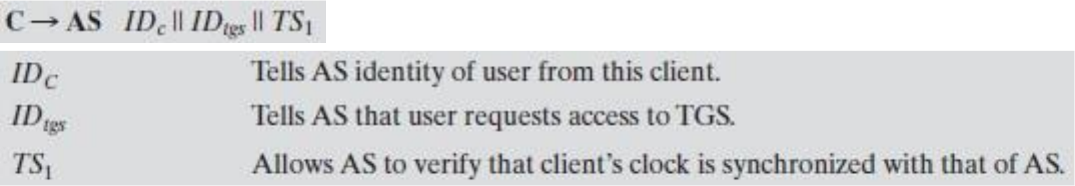

2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password.
Kerberos
Authentication server (AS)
Ticket-granting server (TGS)
Once per user logon session
1. User logs on to workstation and requests service on host.
Request ticket-granting ticket
Ticket + session key
Request service-granting ticket
Ticket + session key
Once per type of service
3. Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS.
4. TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server.
Request service
Provide server authenticator
Once per service session
5. Workstation sends ticket and authenticator to server.
6. Server verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator.

Kerberos Version 4 Message Exchanges

i) Authentication Service Exchange to obtain ticket-granting ticket

1) Client requests ticket-granting ticket

- User logs on to workstation and requests service on host

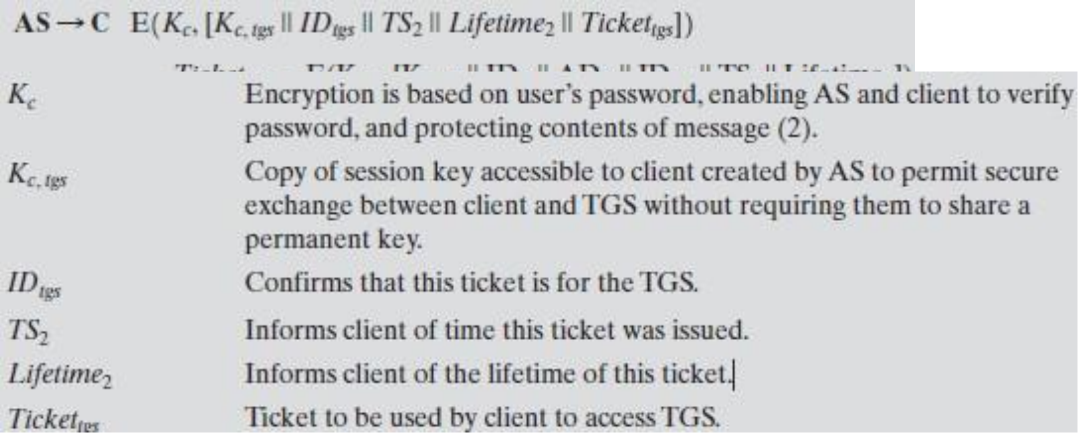


C → AS $ID_c \| ID_{tgs} \| TS_1$

ID_C	Tells AS identity of user from this client.
ID_{tgs}	Tells AS that user requests access to TGS.
TS_1	Allows AS to verify that client's clock is synchronized with that of AS.

2) AS returns ticket-granting ticket

- AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password



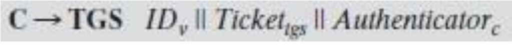
AS → C $E(K_c, [K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}])$

K_c	Encryption is based on user's password, enabling AS and client to verify password, and protecting contents of message (2).
$K_{c,tgs}$	Copy of session key accessible to client created by AS to permit secure exchange between client and TGS without requiring them to share a permanent key.
ID_{tgs}	Confirms that this ticket is for the TGS.
TS_2	Informs client of time this ticket was issued.
$Lifetime_2$	Informs client of the lifetime of this ticket.
$Ticket_{tgs}$	Ticket to be used by client to access TGS.

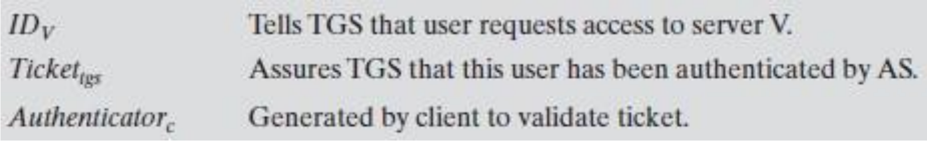
ii) Ticket-Granting Service Exchange to obtain service-granting ticket

3) Client requests service-granting ticket

- Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS



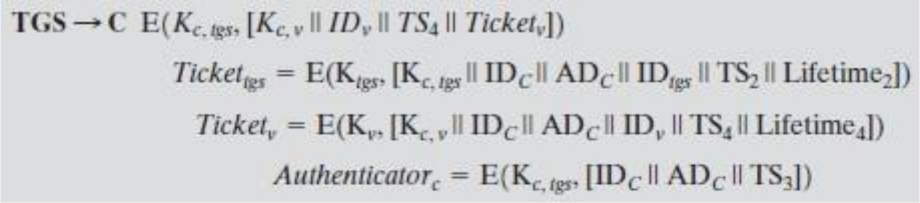
C → TGS $ID_v \| Ticket_{tgs} \| Authenticator_c$



ID_V	Tells TGS that user requests access to server V.
$Ticket_{tgs}$	Assures TGS that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket.

4) TGS returns service-granting ticket

- TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server



TGS → C $E(K_{c,tgs}, [K_{c,v} \| ID_v \| TS_4 \| Ticket_v])$

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$$

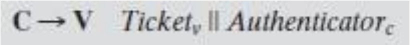
$$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$$

$$Authenticator_c = E(K_{c,tgs}, [ID_C \| AD_C \| TS_3])$$

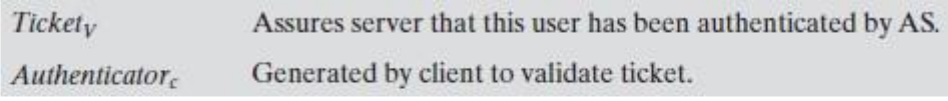
iii) Client/Server Authentication Exchange to obtain service

5) Client requests service

- Workstation sends ticket and authenticator to server



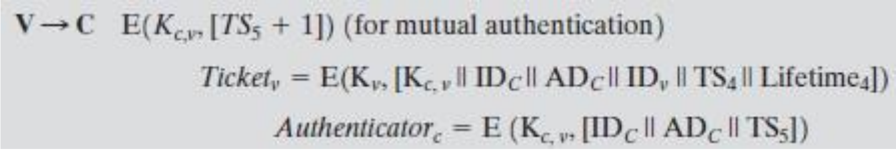
C → V $Ticket_v \| Authenticator_c$



$Ticket_V$	Assures server that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket.

6) Optional authentication of server to client

- Server verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator



V → C $E(K_{c,v}, [TS_5 + 1])$ (for mutual authentication)

$$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$$

$$Authenticator_c = E(K_{c,v}, [ID_C \| AD_C \| TS_5])$$

Kerberos Realms And Multiple Kerberi

Kerberos realm

- a set of managed nodes that share the same Kerberos database.
- database resides on the Kerberos master computer system, kept in a physically secure room.
  - A read-only copy on other Kerberos computer systems.
- all changes to the database must be made on the master computer system.
  - requires the Kerberos master password

Kerberos principal

- 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

Requirements

1. 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.

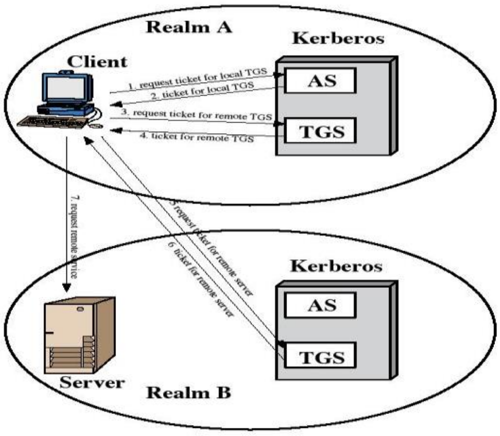
2. The Kerberos server must share a secret key with each server. All servers are registered with the Kerberos server

3. 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

Exchanges

$$\begin{array}{lll}
\textbf{(1)} & \mathrm{C} \rightarrow \mathrm{AS}: & ID_c \parallel ID_{tgs} \parallel TS_1 \\
\textbf{(2)} & \mathrm{AS} \rightarrow \mathrm{C}: & \mathrm{E}(K_c, [K_{c,tgs} \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2 \parallel Ticket_{tgs}]) \\
\textbf{(3)} & \mathrm{C} \rightarrow \mathrm{TGS}: & ID_{tgsrem} \parallel Ticket_{tgs} \parallel Authenticator_c \\
\textbf{(4)} & \mathrm{TGS} \rightarrow \mathrm{C}: & \mathrm{E}(K_{c,tgs}, [K_{c,tgsrem} \parallel ID_{tgsrem} \parallel TS_4 \parallel Ticket_{tgsrem}]) \\
\textbf{(5)} & \mathrm{C} \rightarrow \mathrm{TGS_{rem}}: & ID_{vrem} \parallel Ticket_{tgsrem} \parallel Authenticator_c \\
\textbf{(6)} & \mathrm{TGS_{rem}} \rightarrow \mathrm{C}: & \mathrm{E}(K_{c,tgsrem}, [K_{c,vrem} \parallel ID_{vrem} \parallel TS_6 \parallel Ticket_{vrem}]) \\
\textbf{(7)} & \mathrm{C} \rightarrow \mathrm{V_{rem}}: & Ticket_{vrem} \parallel Authenticator_c
\end{array}$$

Request for Service in another Realm


Realm A
Kerberos
Client
AS
TGS
1. request ticket for local TGS
2. ticket for local TGS
3. request ticket for remote TGS
4. ticket for remote TGS
5. request ticket for remote server
6. ticket for remote server
7. request remote service
Kerberos
AS
TGS
Server
Realm B

Environmental shortcomings of Kerberos version 4

Encryption system dependence

- Export restriction on DES as well as doubts about

the strength of DES In version 5,

- ciphertext is tagged with an encryption type identifier so that any encryption technique may be used
- Encryption keys are tagged with a type and a length,
  - allowing the same key to be used in different algorithms
  - allowing the specification of different variations on a given algorithm.

Internet protocol dependence:

- Version 4 requires the use of Internet Protocol (IP) addresses. Other address types, such as the ISO network address, are not accommodated.
- Version 5 network addresses are tagged with type and length, allowing any network address

Message byte ordering

- In version 4, the sender of a message employs a byte ordering of its Own
- 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.

Ticket lifetime:

- Lifetime values in version 4 are encoded in an 8-bit quantity in units of five minutes.
  - 1280 minutes, or over 21 hours
- In version 5, tickets include an explicit start time and end time, allowing tickets with arbitrary lifetimes

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

Interrealm authentication:

- In version 4, interoperability among N realms requires on the order of N2 Kerberos-to-Kerberos relationships
- Version 5 supports a method that requires fewer relationships

Technical deficiencies

Double encryption

- second encryption is not necessary and is computationally wasteful

PCBC encryption

- Version 4 uses nonstandard mode of DES known as propagating cipher block chaining (PCBC)
- Version 5 provides explicit integrity mechanisms, allowing the standard CBC mode to be used

- a checksum or hash code is attached to the message prior to encryption

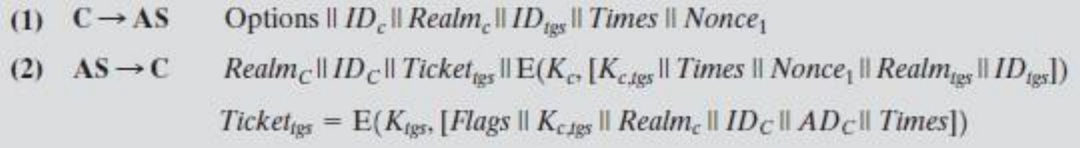
Session keys

Password attacks

- Both versions are vulnerable to a password attack
- Version 5 does provide a mechanism known as Preauthentication

Kerberos Version 5 Message Exchanges

Authentication Service Exchange to obtain ticket-granting ticket

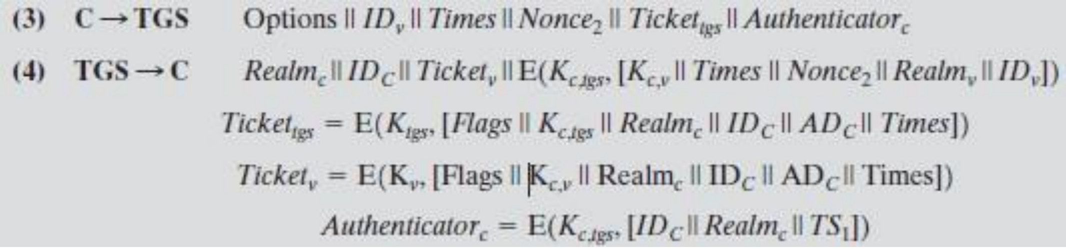


$$\textbf{(1)}\quad \mathbf{C \rightarrow AS}\quad \text{Options} \,\|\, ID_c \,\|\, Realm_c \,\|\, ID_{tgs} \,\|\, Times \,\|\, Nonce_1$$

$$\textbf{(2)}\quad \mathbf{AS \rightarrow C}\quad Realm_C \,\|\, ID_C \,\|\, Ticket_{tgs} \,\|\, \mathrm{E}(K_c, [K_{c,tgs} \,\|\, Times \,\|\, Nonce_1 \,\|\, Realm_{tgs} \,\|\, ID_{tgs}])$$

$$Ticket_{tgs} = \mathrm{E}(K_{tgs}, [Flags \,\|\, K_{c,tgs} \,\|\, Realm_c \,\|\, ID_C \,\|\, AD_C \,\|\, Times])$$

Ticket-Granting Service Exchange to obtain service-granting ticket



$$\textbf{(3)}\quad \mathbf{C \rightarrow TGS}\quad \text{Options} \,\|\, ID_v \,\|\, Times \,\|\, Nonce_2 \,\|\, Ticket_{tgs} \,\|\, Authenticator_c$$

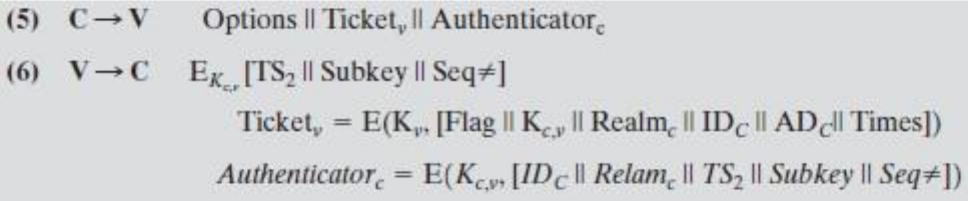
$$\textbf{(4)}\quad \mathbf{TGS \rightarrow C}\quad Realm_c \,\|\, ID_C \,\|\, Ticket_v \,\|\, \mathrm{E}(K_{c,tgs}, [K_{c,v} \,\|\, Times \,\|\, Nonce_2 \,\|\, Realm_v \,\|\, ID_v])$$

$$Ticket_{tgs} = \mathrm{E}(K_{tgs}, [Flags \,\|\, K_{c,tgs} \,\|\, Realm_c \,\|\, ID_C \,\|\, AD_C \,\|\, Times])$$

$$Ticket_v = \mathrm{E}(\mathrm{K}_v, [\mathrm{Flags} \,\|\, \mathrm{K}_{c,v} \,\|\, \mathrm{Realm}_c \,\|\, \mathrm{ID}_C \,\|\, \mathrm{AD}_C \,\|\, \mathrm{Times}])$$

$$Authenticator_c = \mathrm{E}(K_{c,tgs}, [ID_C \,\|\, Realm_c \,\|\, TS_1])$$

Client/Server Authentication Exchange to obtain service



$$\textbf{(5)}\quad \mathbf{C \rightarrow V}\quad \text{Options} \,\|\, \text{Ticket}_v \,\|\, \text{Authenticator}_c$$

$$\textbf{(6)}\quad \mathbf{V \rightarrow C}\quad \mathrm{E}_{K_{c,v}}[\mathrm{TS}_2 \,\|\, \text{Subkey} \,\|\, \text{Seq}\neq]$$

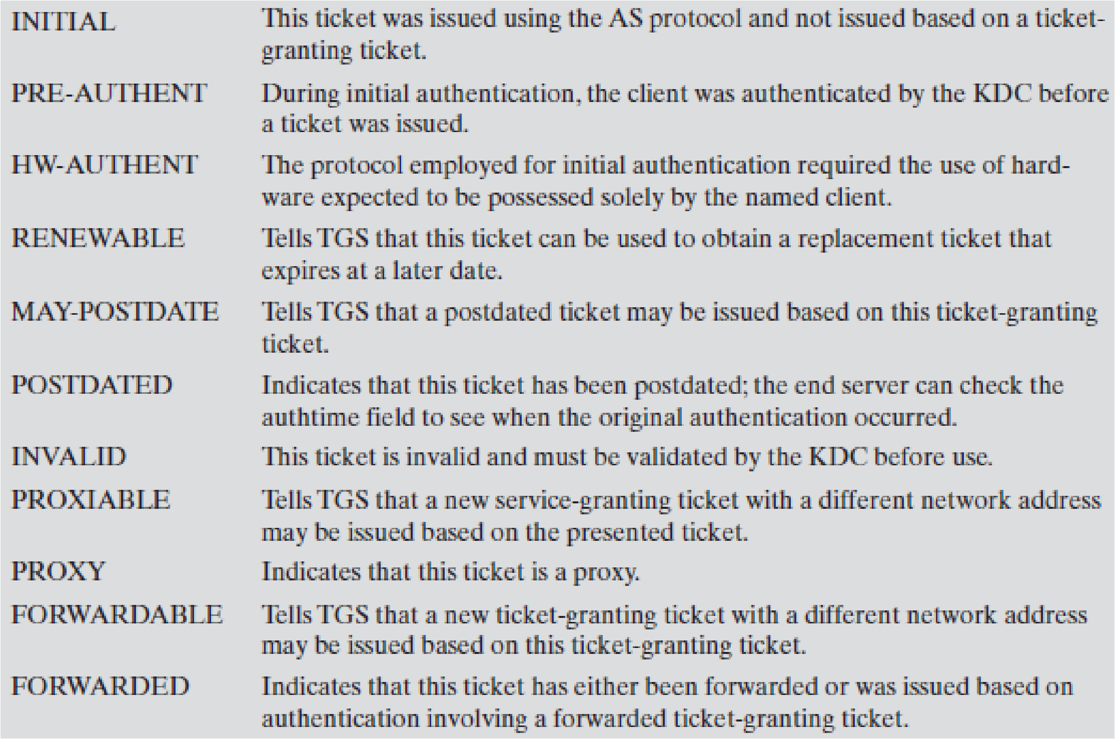
$$\text{Ticket}_v = \mathrm{E}(\mathrm{K}_v, [\text{Flag} \,\|\, \mathrm{K}_{c,v} \,\|\, \text{Realm}_c \,\|\, \mathrm{ID}_C \,\|\, \mathrm{AD}_C \,\|\, \text{Times}])$$

$$Authenticator_c = \mathrm{E}(K_{c,v}, [ID_C \,\|\, Relam_c \,\|\, TS_2 \,\|\, Subkey \,\|\, Seq\neq])$$

New Elements

- **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
  - **rtime**: requested renew-till time
- **Nonce**: A random value to be repeated in message (2) to assure that the response is fresh

Kerberos Version 5 Flags

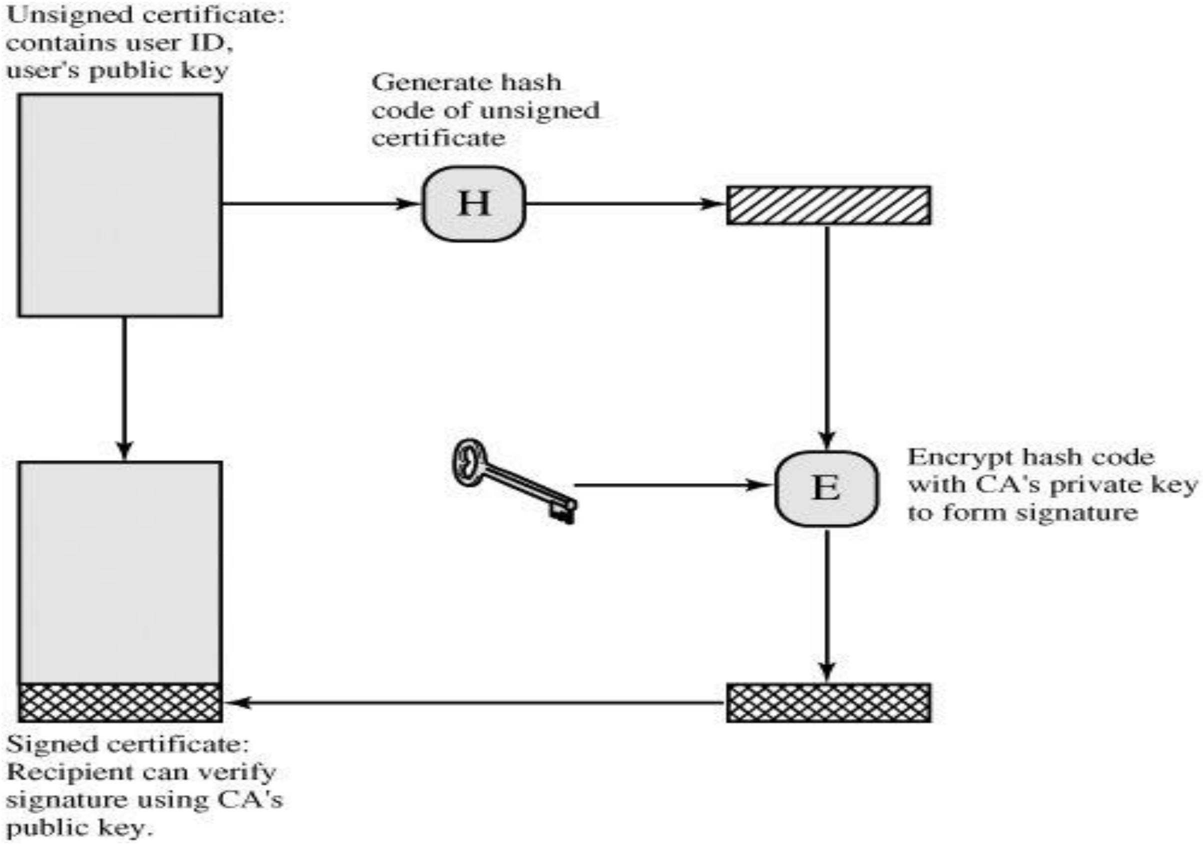


INITIAL	This ticket was issued using the AS protocol and not issued based on a ticket-granting ticket.
PRE-AUTHENT	During initial authentication, the client was authenticated by the KDC before a ticket was issued.
HW-AUTHENT	The protocol employed for initial authentication required the use of hardware expected to be possessed solely by the named client.
RENEWABLE	Tells TGS that this ticket can be used to obtain a replacement ticket that expires at a later date.
MAY-POSTDATE	Tells TGS that a postdated ticket may be issued based on this ticket-granting ticket.
POSTDATED	Indicates that this ticket has been postdated; the end server can check the authtime field to see when the original authentication occurred.
INVALID	This ticket is invalid and must be validated by the KDC before use.
PROXIABLE	Tells TGS that a new service-granting ticket with a different network address may be issued based on the presented ticket.
PROXY	Indicates that this ticket is a proxy.
FORWARDABLE	Tells TGS that a new ticket-granting ticket with a different network address may be issued based on this ticket-granting ticket.
FORWARDED	Indicates that this ticket has either been forwarded or was issued based on authentication involving a forwarded ticket-granting ticket.

X.509 Authentication services

- ITU-T recommendation X.509 is part of the X.500 series of recommendations that define a directory service.
- The directory is, in effect, a server or distributed set of servers that maintains a database of information about users.
- The information includes a mapping from user name to network address, as well as other attributes and information about the users.
- X.509 defines a framework for the provision of authentication services by the X.500 directory to its users.
- The directory may serve as a repository of public-key certificates of the type
- Each certificate contains the public key of a user and is signed with the private key of a trusted certification authority.
- In addition, X.509 defines alternative authentication protocols based on the use of public-key certificates.
- X.509 is based on the use of public-key cryptography and digital signatures.
- The standard does not dictate the use of a specific algorithm but recommends RSA.
- The digital signature scheme is assumed to require the use of a hash function. Again, the standard does not dictate a specific hash algorithm.

- The 1988 recommendation included the description of a recommended hash algorithm; this algorithm has since been shown to be insecure and was dropped from the 1993 recommendation.


Unsigned certificate: contains user ID, user's public key
Generate hash code of unsigned certificate
H
E
Encrypt hash code with CA's private key to form signature
Signed certificate: Recipient can verify signature using CA's public key.

- **issued by a Certification Authority (CA), containing:**

– version (1, 2, or 3)

– serial number (unique within CA) identifying certificate

– signature algorithm identifier

– issuer X.500 name (CA)

– period of validity (from - to dates)

– subject X.500 name (name of owner)

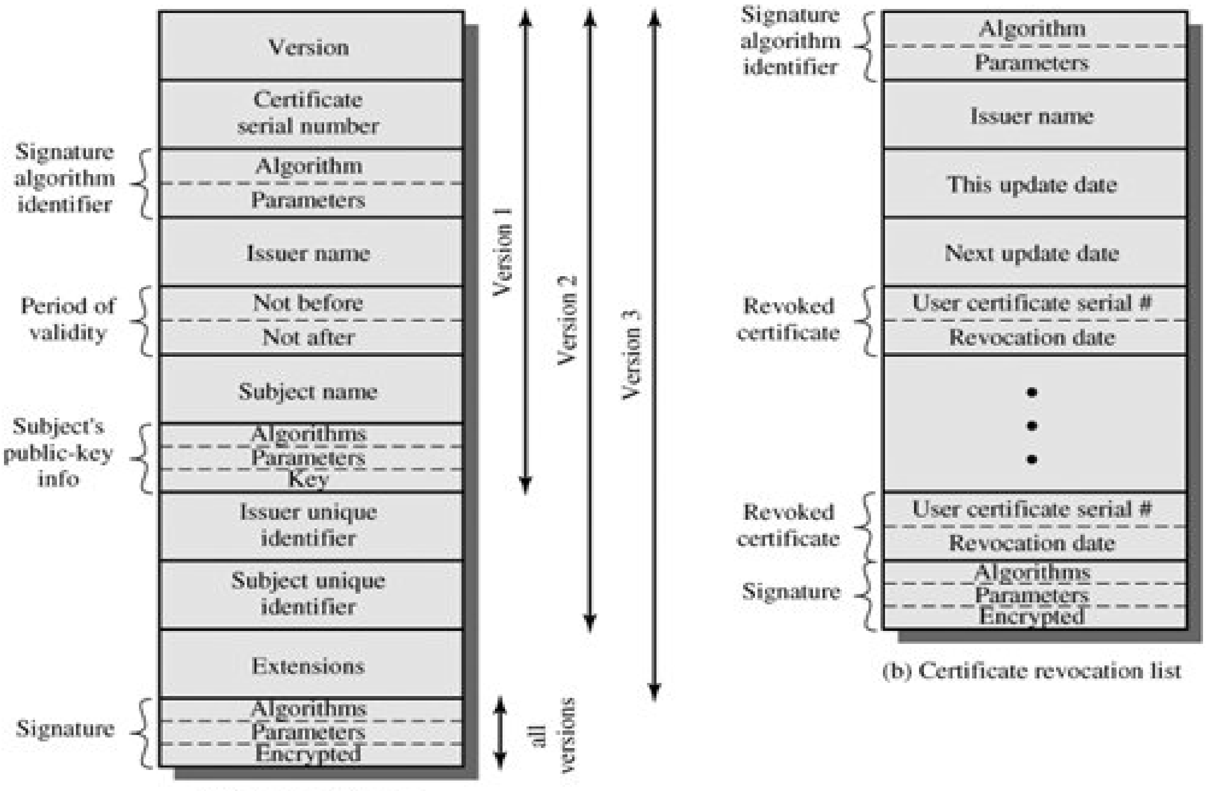
– subject public-key info (algorithm, parameters, key)

– issuer unique identifier (v2+)

– subject unique identifier (v2+)

– extension fields (v3)

– signature (of hash of all fields in certificate)

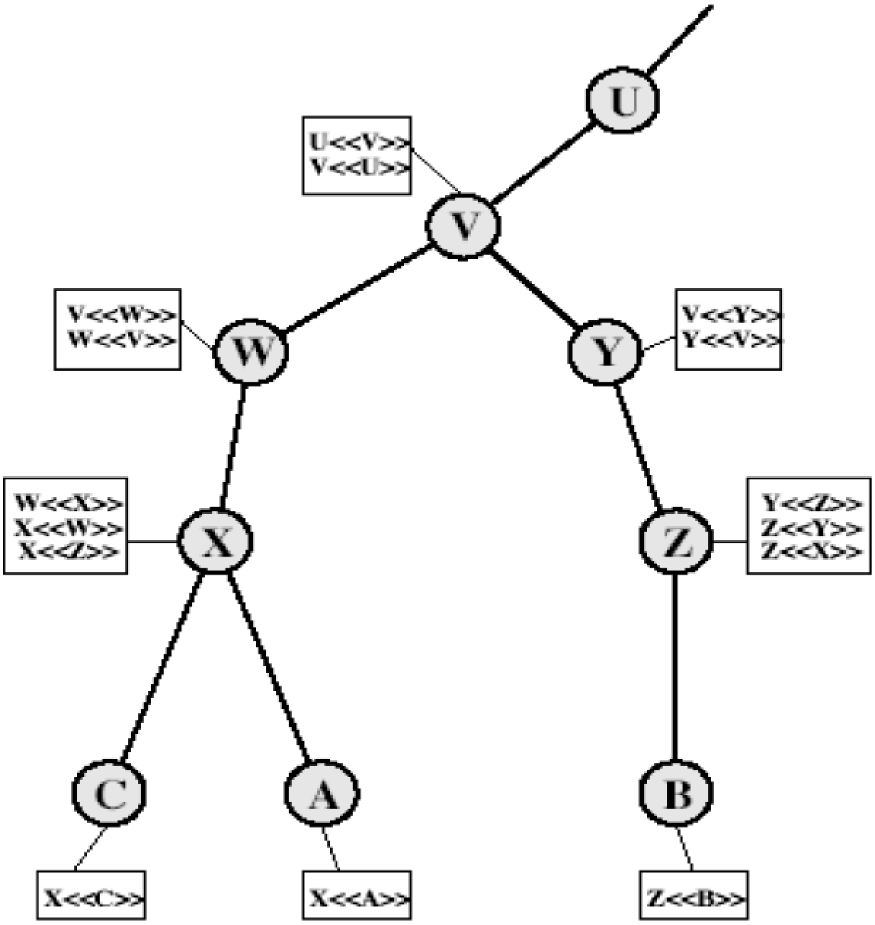


(b) Certificate revocation list

(a) X.509 certificate

- **Signature:**
- Covers all of the other fields of the certificate; it contains the hash code of the other fields, encrypted with the CA's private key.
- This field includes the signature algorithm identifier.
- The standard uses the following 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.
- **Obtaining a User's Certificate**
- User certificates generated by a CA have the following characteristics:
- · Any user with access to the public key of the CA can verify the user public key that was certified.
- · No party other than the certification authority can modify the certificate without this being detected.

- **CA Hierarchy Use**



- 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>>
- When A has obtained these certificates, it can unwrap the certification path in sequence to recover a trusted copy of B's public key.
- Using this public key, A can send encrypted messages to B.
- If A wishes to receive encrypted messages back from B, or to sign messages sent to B, then B will require A’s public key, which can be obtained from the following certification path:
- Z<<Y>> Y <<V>> V <<W>> W <<X>>X <<A>>
- B can obtain this set of certificates from the directory, or A can provide them as part of its initial message to B.
- **Certificate Revocation**

• certificates have a period of validity

• may need to revoke before expiry, for the following reasons eg:

1. user's private key is compromised

2. user is no longer certified by this CA

3. CA's certificate is compromised

• CA‟s maintain list of revoked certificates

1. the Certificate Revocation List (CRL)

• users should check certs with CA‟s CRL

- **Authentication Procedures**

- X.509 includes three alternative authentication procedures:

• **One-Way Authentication**

• **Two-Way Authentication**

• **Three-Way Authentication**

• all use public-key signatures

- **One-Way Authentication**

• 1 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**

• 2 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

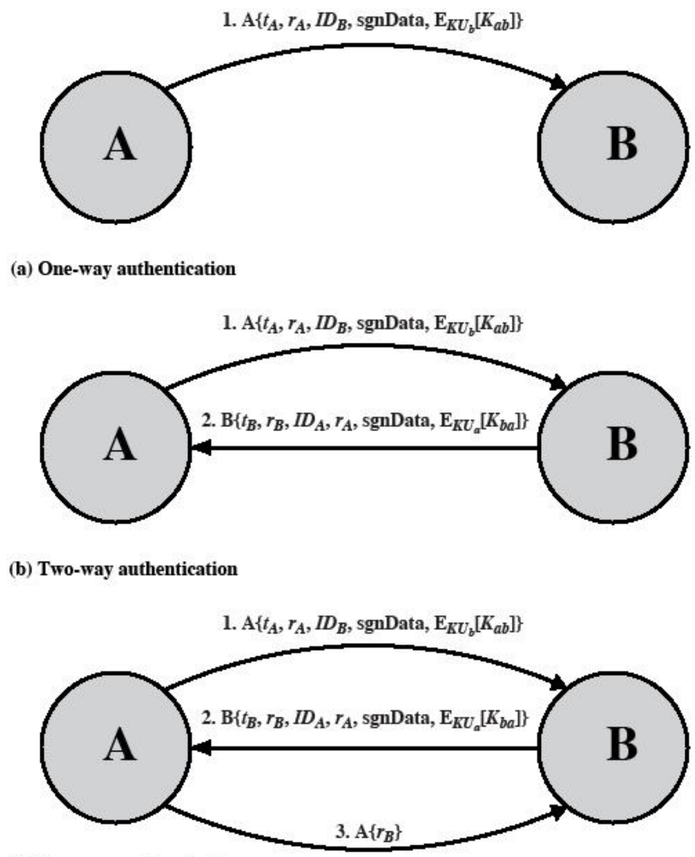
• reply includes original nonce from A, also timestamp and nonce from B

- **Three-Way Authentication**

• 3 messages (A->B, B->A, A->B) which enables above authentication without synchronized clocks

• has reply from A back to B containing signed copy of nonce from B

• means that timestamps need not be checked or relied upon


1. $A\{t_A, r_A, ID_B, \text{sgnData}, E_{KU_b}[K_{ab}]\}$
A
B
1. $A\{t_A, r_A, ID_B, \text{sgnData}, E_{KU_b}[K_{ab}]\}$
2. $B\{t_B, r_B, ID_A, r_A, \text{sgnData}, E_{KU_a}[K_{ba}]\}$
A
B
1. $A\{t_A, r_A, ID_B, \text{sgnData}, E_{KU_b}[K_{ab}]\}$
2. $B\{t_B, r_B, ID_A, r_A, \text{sgnData}, E_{KU_a}[K_{ba}]\}$
3. $A\{r_B\}$
A
B

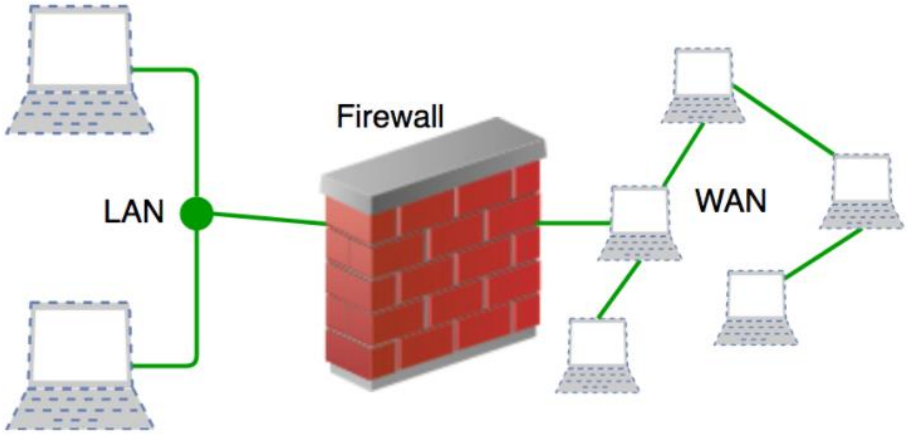
(a) One-way authentication

(b) Two-way authentication

(c) Three-way authentication

FIREWALLS

- A firewall is a network security device, either hardware or software-based, which monitors all incoming and outgoing traffic and based on a defined set of security rules it accepts, rejects or drops that specific traffic.
- **Accept :** allow the traffic
  Reject : block the traffic but reply with an "unreachable error"
  Drop : block the traffic with no reply
- A firewall establishes a barrier between secured internal networks and outside untrusted network, such as the Internet.



- **Firewall design principles**
- The firewall is inserted between the premise network and internet to establish a controlled link and to erect an outer security wall or perimeter.
- The aim of this perimeter is to protect the premises network from internet based attacks and to provide a single choke point where security and audit can be imposed.
- The firewall can be a single computer system or a set of two or more systems that cooperate to perform the firewall function.
- **Firewall characteristics:**
- All traffic from inside to outside, and vice versa, must pass through the firewall. This is achieved by physically blocking all access to the local network except via the firewall. Various configurations are possible.
- Only authorized traffic, as defined by the local security policy, will be allowed to pass.
- Various types of firewalls are used, which implement various types of security policies.
- The firewall itself is immune to penetration. This implies that use of a trusted system with a secure operating system.
- This implies that use of a trusted system with a secure operating system.
- Four techniques that firewall use to control access and enforce the site‟s security policy is as follows:
- Service control – determines the type of internet services that can be accessed, inbound or outbound. The firewall may filter traffic on this basis of IP address and TCP port number; may provide proxy software that receives and interprets each service request before passing it on; or may host the server software itself, such as web or mail service.
- Direction control – determines the direction in which particular service request may be initiated and allowed to flow through the firewall.
- User control – controls access to a service according to which user is attempting to access it.

Behavior control – controls how particular services are used.

- **Capabilities of firewall**
- A firewall defines a single choke point that keeps unauthorized users out of the protected network, prohibits potentially vulnerable services from entering or leaving the network, and provides protection from various kinds of IP spoofing and routing attacks.
- A firewall provides a location for monitoring security related events. Audits and alarms can be implemented on the firewall system.
- A firewall is a convenient platform for several internet functions that are not security related.
- A firewall can serve as the platform for IPsec.
- **Types of firewalls**

- There are 3 common types of firewalls.

- Packet filters
- Application-level gateways
- Circuit-level gateways

Packet filtering router

- A packet filtering router applies a set of rules to each incoming IP packet and then forwards or discards the packet.
- The router is typically configured to filter packets going in both directions.
- Filtering rules are based on the information contained in a network packet:

· Source IP address – IP address of the system that originated the IP packet.

· Destination IP address – IP address of the system, the IP is trying to reach.

· Source and destination transport level address – transport level port number.

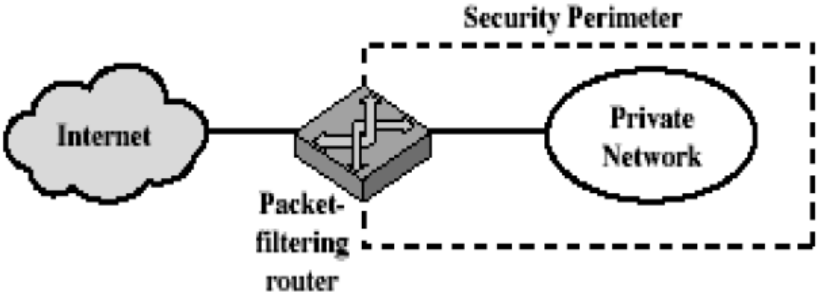
· IP protocol field – defines the transport protocol.

· Interface – for a router with three or more ports, which interface of the router the packet come from or which interface of the router the packet is destined for.

- The packet filter is typically set up as a list of rules based on matches to fields in the IP or TCP header.
- If there is a match to one of the rules, that rule is invoked to determine whether to forward or discard the packet. If there is no match to any rule, then a default action is taken.
- Two default policies are possible:

· Default = discard: That which is not expressly permitted is prohibited.

· Default = forward: That which is not expressly prohibited is permitted.



(a) Packet-filtering router

- **Advantages of packet filter router**

· Simple

· Transparent to users

· Very fast

- **Weakness of packet filter firewalls**

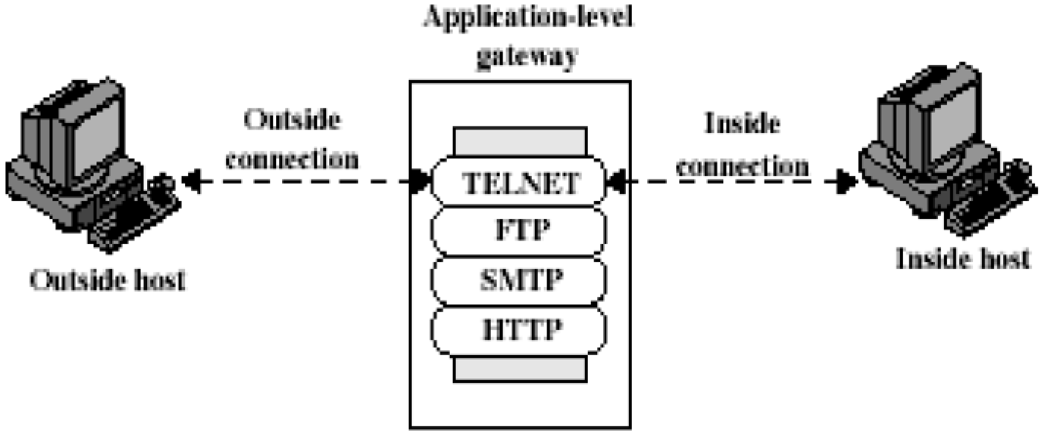
· Because packet filter firewalls do not examine upper-layer data, they cannot prevent attacks that employ application specific vulnerabilities or functions.

· Because of the limited information available to the firewall, the logging functionality present in packet filter firewall is limited.

· It does not support advanced user authentication schemes.

· They are generally vulnerable to attacks such as layer address spoofing.

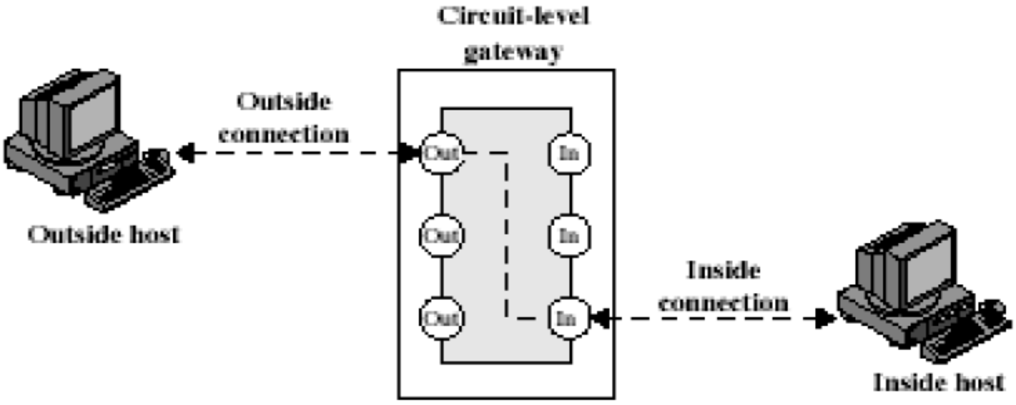
- **Application level gateway**
- An Application level gateway, also called a proxy server, acts as a relay of application level traffic.
- The user contacts the gateway using a TCP/IP application, such as Telnet or FTP, and the gateway asks the user for the name of the remote host to be accessed.
- When the user responds and provides a valid user ID and authentication information, the gateway contacts the application on the remote host and relays TCP segments containing the application data between the two endpoints.
- Application level gateways tend to be more secure than packet filters.
- It is easy to log and audit all incoming traffic at the application level.
- A prime disadvantage is the additional processing overhead on each connection.



(b) Application-level gateway

- **Circuit level gateway**
- Circuit level gateway can be a stand-alone system or it can be a specified function performed by an application level gateway for certain applications.

- A Circuit level gateway does not permit an end-to-end TCP connection; rather, the gateway sets up two TCP connections, one between itself and a TCP user on an inner host and one between itself and a TCP user on an outer host.
- Once the two connections are established, the gateway typically relays TCP segments from one connection to the other without examining the contents.
- The security function consists of determining which connections will be allowed.



(c) Circuit-level gateway

- **Bastion host**
- It is a system identified by the firewall administrator as a critical strong point in the network‟s security.
- The Bastion host serves as a platform for an application level and circuit level gateway.
- Common characteristics of a Bastion host are as follows:

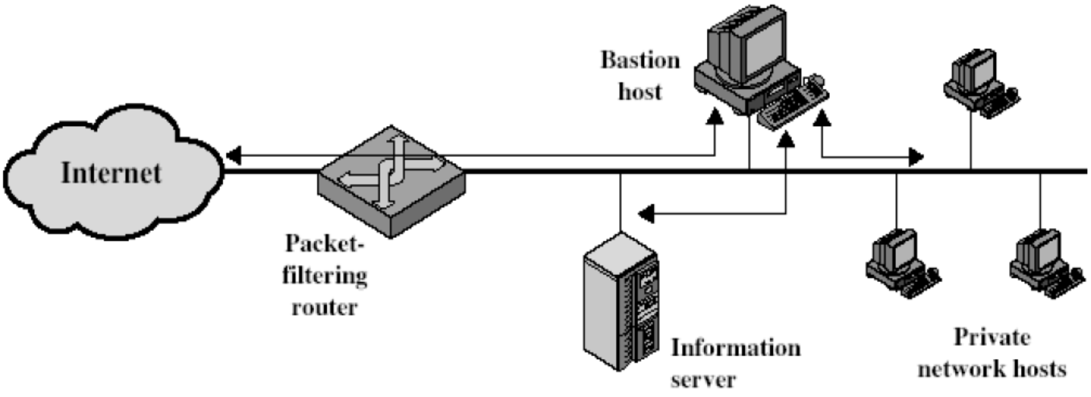
· The Bastion host hardware platform executes a secure version of its operating system, making it a trusted system.

- Only the services that the network administrator considers essential are installed on the Bastion host.
- It may require additional authentication before a user is allowed access to the proxy services.

· Each proxy is configured to support only a subset of standard application‟s command set.

- Each proxy is configured to allow access only to specific host systems.
- Each proxy maintains detailed audit information by logging all traffic, each connection and the duration of each connection.
- Each proxy is independent of other proxies on the Bastion host.
- A proxy generally performs no disk access other than to read its initial configuration file.
- Each proxy runs on a non privileged user in a private and secured directory on the Bastion host.

- **1. Screened host firewall, single-homed bastion configuration**



(a) Screened host firewall system (single-homed bastion host)

- In this configuration, the firewall consists of two systems: a packet filtering router and a bastion host. Typically, the router is configured so that

· For traffic from the internet, only IP packets destined for the bastion host are allowed in.

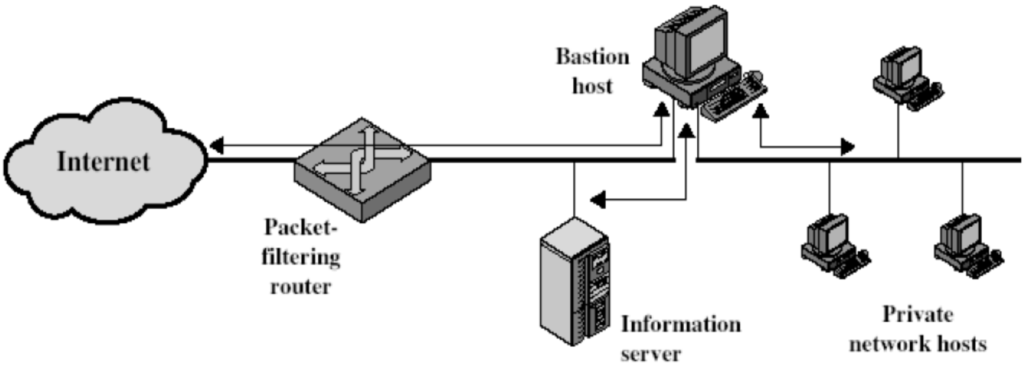
· For traffic from the internal network, only IP packets from the bastion host are allowed out.

- The bastion host performs authentication and proxy functions. This configuration has greater security than simply a packet filtering router or an application level gateway alone, for two reasons:

· This configuration implements both packet level and application level filtering, allowing for considerable flexibility in defining security policy.

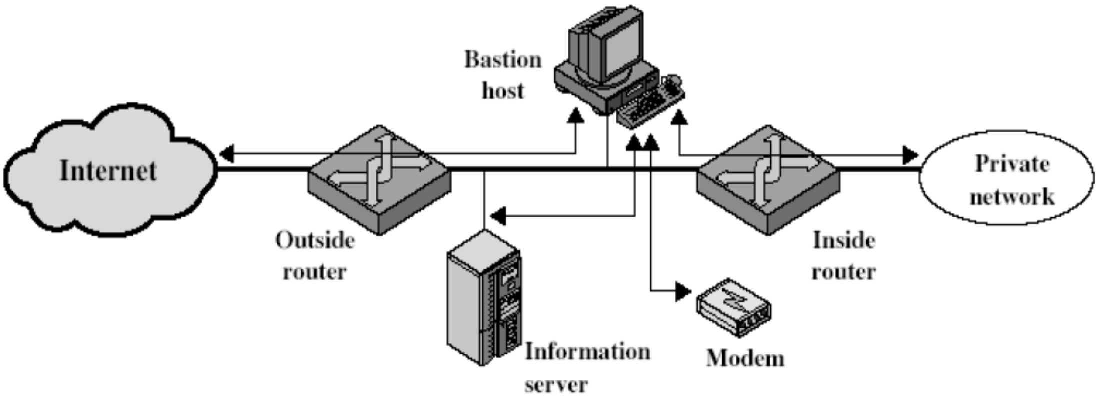
· An intruder must generally penetrate two separate systems before the security of the internal network is compromised.

Screened host firewall, dual homed bastion configuration



(b) Screened host firewall system (dual-homed bastion host)

- **Screened subnet firewall configuration**



(c) Screened-subnet firewall system

- In this configuration, two packet filtering routers are used, one between the bastion host and internet and one between the bastion host and the internal network.
- This configuration creates an isolated subnetwork, which may consist of simply the bastion host but may also include one or more information servers and modems for dial-in capability.
- Typically both the internet and the internal network have access to hosts on the screened subnet, but traffic across the screened subnet is blocked.

SET E-commerce Transaction

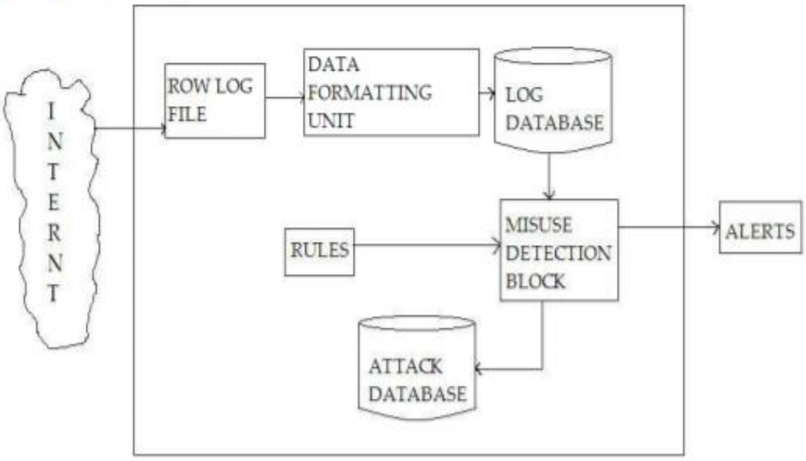
- **Secure Electronic Transaction** (**SET**) is a communications protocol standard for securing credit card transactions over networks, specifically, the Internet.
- SET was not itself a payment system, but rather a set of security protocols and formats that enabled users to employ the existing credit card payment infrastructure on an open network in a secure fashion.
- Secure Electronic Transaction (SET) is a system for ensuring the security of financial transactions on the Internet. It was supported initially by Mastercard, Visa, Microsoft, Netscape, and others.
- With SET, a user is given an electronic wallet (digital certificate) and a transaction is conducted and verified using a combination of digital certificates and digital signatures among the purchaser, a merchant, and the purchaser's bank in a way that ensures privacy and confidentiality

INTRUDERS

- One of the most publicized attacks to security is the intruder, generally referred to as hacker or cracker. Three classes of intruders are as follows:
- **Masquerader** – an individual who is not authorized to use the computer and who penetrates a system’s access controls to exploit a legitimate user’s account.
- **Misfeasor** – a legitimate user who accesses data, programs, or resources for which such access is not authorized, or who is authorized for such access but misuse his or her privileges.

- **Clandestine user** – an individual who seizes supervisory control of the system and uses this control to evade auditing and access controls or to suppress audit collection.
- **INTRUSION DETECTION:**
- The masquerader is likely to be an outsider; the misfeasor generally is an insider; and the clandestine user can be either an outsider or an insider.
- Inevitably, the best intrusion prevention system will fail. A system's second line of defense is intrusion detection, and this has been the focus of much research in recent years.
- This interest is motivated by a number of considerations, including the following:
- If an intrusion is detected quickly enough, the intruder can be identified and ejected from the system before any damage is done or any data are compromised.
- An effective intrusion detection system can serve as a deterrent, so acting to prevent intrusions.
- Intrusion detection enables the collection of information about intrusion techniques that can be used to strengthen the intrusion prevention facility.
- Intrusion detection is based on the assumption that the behavior of the intruder differs from that of a legitimate user in ways that can be quantified.

Architecture of Intrusion Detection System



- It consists of following blocks: **Log File:** Packet sniffer Win Dump collects packet headers of data coming from internet or LAN. Data captured from WinDump is redirected to a file. This file is called as log file.
- **Data Formatting Unit:** Data collected in log file is classified according to various fields in the packet header.
- Protocols used for different packets are identified using some specific fields or predefined values of these fields.
- **Log Database:** It contains different tables according to different protocols (like TCP/IP, UDP, ICMP, and ARP).
- For each protocol there is one table. Each table consists of attributes related to that particular protocol. Formatted Data is stored in the database.

- **Misuse Detection Block:** Misuse Detection technique is used for detection of known attacks. Many computer attacks have fix signature.
- These attack signatures can be used to identify particular attack. We use predefined rules and compare the captured data packet header with them. If pattern matches, intrusion detection system declares it as intrusion and alerts administrator about it.
- **Attack Database:** Attack database also contains tables for different protocols as in case of log database. The entries from log database which are declared as attacks are stored in attack database. This database can be referred in future for drawing some conclusions or as a table showing statistics of past attacks on the system.

Trusted systems

- One way to enhance the ability of a system to defend against intruders and malicious programs is to implement trusted system technology.
- **Data access control**
- Following successful logon, the user has been granted access to one or set of hosts and applications. This is generally not sufficient for a system that includes sensitive data in its database. Through the user access control procedure, a user can be identified to the system. Associated with each user, there can be a profile that specifies permissible operations and file accesses. The operating system can then enforce rules based on the user profile. The database management system, however, must control access to specific records or even portions of records. The operating system may grant a user permission to access a file or use an application, following which there are no further security checks, the database management system must make a decision on each individual access attempt. That decision will depend not only on the user's identity but also on the specific parts of the data being accessed and even on the information already divulged to the user.
- A general model of access control as exercised by an file or database management system is that of an access matrix. The basic elements of the model are as follows:
- **Subject**: An entity capable of accessing objects. Generally, the concept of subject equates with that of process.
- **Object**: Anything to which access is controlled. Examples include files, portion of files, programs, and segments of memory.
- **Access right:** The way in which the object is accessed by a subject. Examples are read, write and execute. One axis of the matrix consists of identified subjects that may attempt data access.
- Typically, this list will consist of individual users or user groups. The other axis lists the objects that may be accessed. Objects may be individual data fields. Each entry in the matrix indicates the access rights of that subject for that object. The matrix may be decomposed by columns, yielding **access control lists.** Thus, for each object, an access control list lists users and their permitted access rights. The access control list may contain a default, or public, entry.
- **The concept of Trusted Systems:**

- When multiple categories or levels of data are defined, the requirement is referred to as multilevel security.
- The general statement of the requirement for multilevel security is that a subject at a high level may not convey information to a subject at a lower or noncomparable level unless that flow accurately reflects the will of an authorized user.
- For implementation purposes, this requirement is in two parts and is simply stated. A multilevel secure system must enforce:
- · **No read up:** A subject can only read an object of less or equal security level. This is referred to as **simple security property.**
- · **No write down:** A subject can only write into an object of greater or equal security level.
- **Reference Monitor concept**
- The reference monitor is a controlling element in the hardware and operating system of a computer that regulates the access of subjects to objects on the basis of security parameters of the subject and object.
- The reference monitor has access to a file, known as the security kernel database that lists the access privileges (security clearance) of each subject and the protection attributes (classification level) of each object.
- The reference monitor enforces the security rules and has the following properties:
- Complete mediation: The security rules are enforced on every access, not just, for example, when a file is opened.
- Isolation: The reference monitor and database are protected from unauthorised modification.
- Verifiability: The reference monitor's correctness must be provable. That is, it must be possible to demonstrate mathematically that the reference monitor enforces the security rules and provides complete mediation and isolation. Important security events, such as detected security violations and authorized changes to the security kernel database, are stored in the audit file.

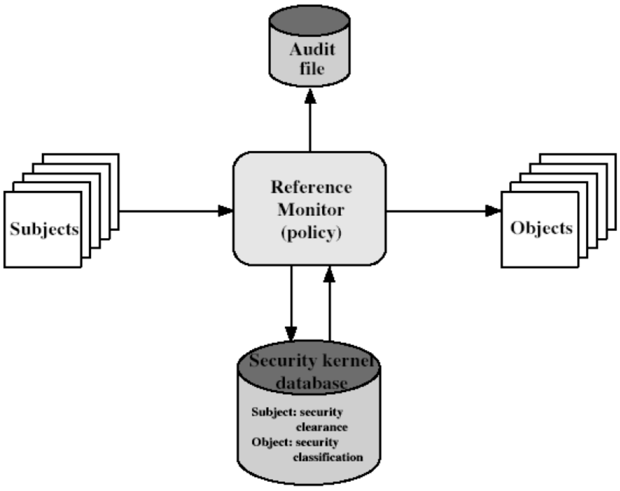

Audit file
Subjects
Reference Monitor (policy)
Objects
Security kernel database
Subject: security clearance
Object: security classification

Fig:Reference Monitor Concept

VIRUSES AND RELATED THREATS

- Perhaps the most sophisticated types of threats to computer systems are presented by programs that exploit vulnerabilities in computing systems.

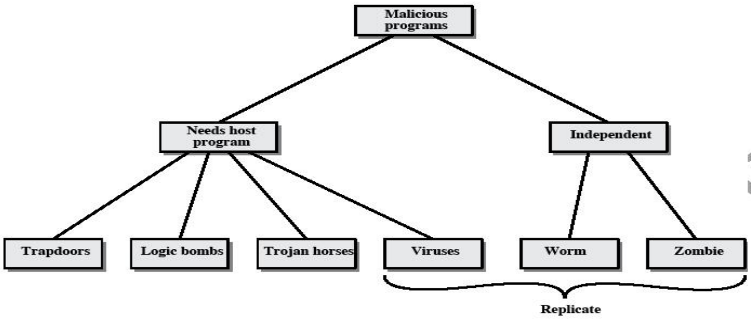


Figure 19.1 Taxonomy of Malicious Programs

Name	Description
Virus	Attaches itself to a program and propagates copies of itself to other programs
Worm	Program that propagates copies of itself to other computers
Logic bomb	Triggers action when condition occurs
Trojan horse	Program that contains unexpected additional functionality
Backdoor (trapdoor)	Program modification that allows unauthorized access to functionality
Exploits	Code specific to a single vulnerability or set of vulnerabilities
Downloaders	Program that installs other items on a machine that is under attack. Usually, a downloader is sent in an e-mail.
Auto-rooter	Malicious hacker tools used to break into new machines remotely
Kit (virus generator)	Set of tools for generating new viruses automatically
Spammer programs	Used to send large volumes of unwanted e-mail
Flooders	Used to attack networked computer systems with a large volume of traffic to carry out a denial of service (DoS) attack
Keyloggers	Captures keystrokes on a compromised system
Rootkit	Set of hacker tools used after attacker has broken into a computer system and gained root-level access
Zombie	Program activated on an infected machine that is activated to launch attacks on other machines

- **The Nature of Viruses**
- A virus is a piece of software that can "infect" other programs by modifying them; the modification includes a copy of the virus program, which can then go on to infect other programs.
- During its lifetime, a typical virus goes through the following four phases:
- · **Dormant phase**: The virus is idle. The virus will eventually be activated by some event, such as a date, the presence of another program or file, or the capacity of the disk exceeding some limit. Not all viruses have this stage.
- **Propagation phase:** The virus places an identical copy of itself into other programs or into certain system areas on the disk. Each infected program will now contain a clone of the virus, which will itself enter a propagation phase.
- **Triggering phase:** The virus is activated to perform the function for which it was intended. As with the dormant phase, the triggering phase can be caused by a variety of system events, including a count of the number of times that this copy of the virus has made copies of itself.
- · **Execution phase:** The function is performed. The function may be harmless, such as a message on the screen, or damaging, such as the destruction of programs and data files.
- ***Virus Structure:***
- A virus can be prepended or postpended to an executable program, or it can be embedded in some other fashion. The key to its operation is that the infected program, when invoked, will first execute the virus code and then execute the original code of the program.
- **An infected program begins with the virus code and works as follows.**
- The first line of code is a jump to the main virus program. The second line is a special marker that is used by the virus to determine whether or not a potential victim program has already been infected with this virus.
- When the program is invoked, control is immediately transferred to the main virus program. The virus program first seeks out uninfected executable files and infects them. Next, the virus may perform some action, usually detrimental to the system.
- This action could be performed every time the program is invoked, or it could be a logic bomb that triggers only under certain conditions.
- Finally, the virus transfers control to the original program. If the infection phase of the program is reasonably rapid, a user is unlikely to notice any difference between the execution of an infected and uninfected program.
- **Types of Viruses**
- **Parasitic virus**: The traditional and still most common form of virus. A parasitic virus attaches itself to executable files and replicates, when the infected program is executed, by finding other executable files to infect.
- **Memory-resident virus**: Lodges in main memory as part of a resident system program.

- **Boot sector virus**: Infects a master boot record or boot record and spreads when a system is booted from the disk containing the virus.
- **Stealth virus**: A form of virus explicitly designed to hide itself from detection by antivirus software
- **Polymorphic virus**: A virus that mutates with every infection, making detection by the "signature" of the virus impossible.
- **E-mail Viruses**
- A more recent development in malicious software is the e-mail virus. The first rapidly spreading e-mail viruses, such as Melissa, made use of a Microsoft Word macro embedded in an attachment. If the recipient opens the e-mail attachment, the Word macro is activated.
- 1. The e-mail virus sends itself to everyone on the mailing list in the user's e-mail package.
- 2. The virus does local damage.
- **Worms**
- A worm is a program that can replicate itself and send copies from computer to computer across network connections. Upon arrival, the worm may be activated to replicate and propagate again.
- **Metamorphic virus**: As with a polymorphic virus, a metamorphic virus mutates with every infection. The difference is that a metamorphic virus rewrites itself completely at each iteration, increasing the difficulty of detection. Metamorphic viruses my change their behavior as well as their appearance.
- **Macro Viruses :** In the mid-1990s, macro viruses became by far the most prevalent type of virus.
- 1. A macro virus is platform independent. Virtually all of the macro viruses infect Microsoft Word documents. Any hardware platform and operating system that supports Word can be infected.
- 2. Macro viruses infect documents, not executable portions of code. Most of the information introduced onto a computer system is in the form of a document rather than a program.
- 3. Macro viruses are easily spread. A very common method is by electronic mail.
- **Antivirus Approaches**
- The ideal solution to the threat of viruses is prevention: The next best approach is to be able to do the following:

· **Detection:** Once the infection has occurred, determine that it has occurred and locate the virus.

· **Identification**: Once detection has been achieved, identify the specific virus that has infected a program.

· **Removal**: Once the specific virus has been identified, remove all traces of the virus from the infected program and restore it to its original state. Remove the virus from all infected systems so that the disease cannot spread further.

Unit-V

E-MAIL, IP & WEB SECURITY

Email security is a term for describing different procedures and techniques for protecting **email** accounts, content, and communication against unauthorized access, loss or compromise. **Email** is often used to spread malware, spam and phishing attacks.

Attacks possible through E-mail

- E-mail Hacking
- Email hacking can be done in any of the following ways:
- **Spam**
- E-mail spamming is an act of sending **Unsolicited Bulk E-mails (UBI)** which one has not asked for. Email spams are the junk mails sent by commercial companies as an advertisement of their products and services.
- **Virus**
- Some emails may incorporate with files containing malicious script which when run on your computer may lead to destroy your important data.
- **Phishing**
- Email phishing is an activity of sending emails to a user claiming to be a legitimate enterprise. Its main purpose is to steal sensitive information such as usernames, passwords, and credit card details.
- Such emails contains link to websites that are infected with malware and direct the user to enter details at a fake website whose look and feels are same to legitimate one.

Pretty Good Privacy

- PGP is a remarkable phenomenon. Largely the effort of a single person, Phil Zimmermann.
- PGP provides a confidentiality, authentication and digital signature service that can be used for electronic mail and file storage applications.
- Also provides email compatibility and compression algorithm.
- The following symbols are used:

K_s =session key used in symmetric encryption scheme

PR_a =private key of user A, used in public-key encryption scheme

PU_a =public key of user A, used in public-key encryption scheme

EP = public-key encryption

DP = public-key decryption

EC = symmetric encryption

DC = symmetric decryption

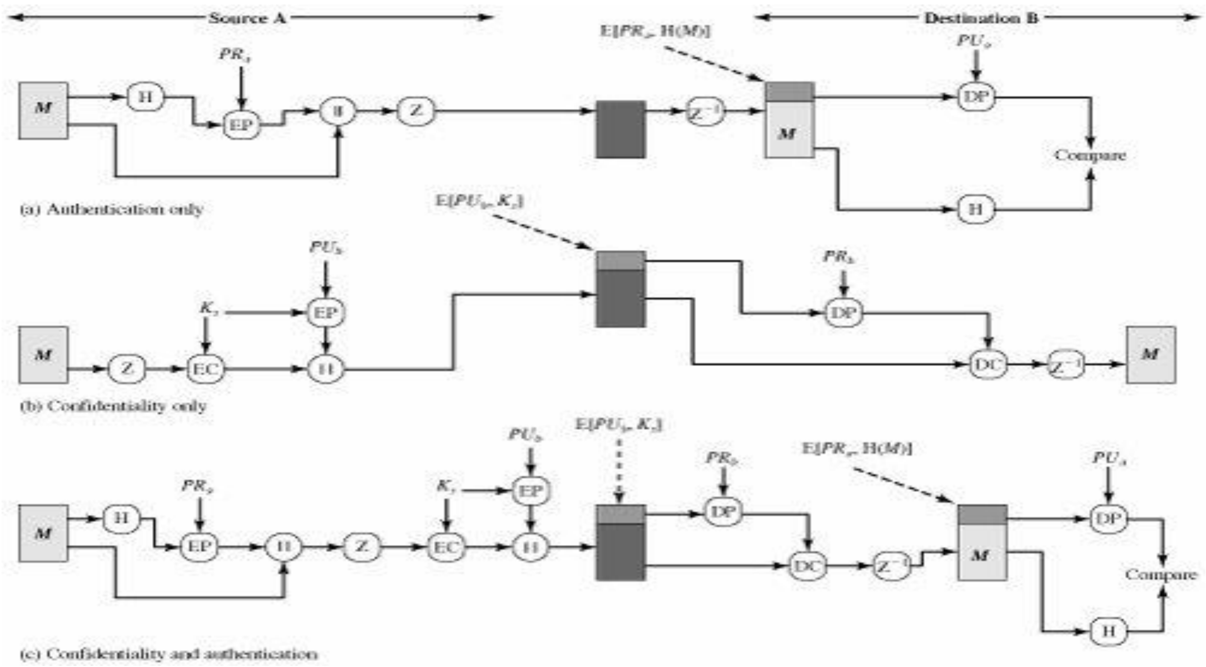
H = hash function

|| = concatenation

Z = compression using ZIP algorithm

R64 = conversion to radix 64 ASCII format

- **Authentication**
- The sender creates a message.
- SHA-1 is used to generate a 160-bit hash code of the message.
- The hash code is encrypted with RSA using the sender's private key, and the result is prepended to the message.
- The receiver uses RSA with the sender's public key to decrypt and recover the hash code.
- The receiver generates a new hash code for the message and compares it with the decrypted hash code. If the two match, the message is accepted as authentic.



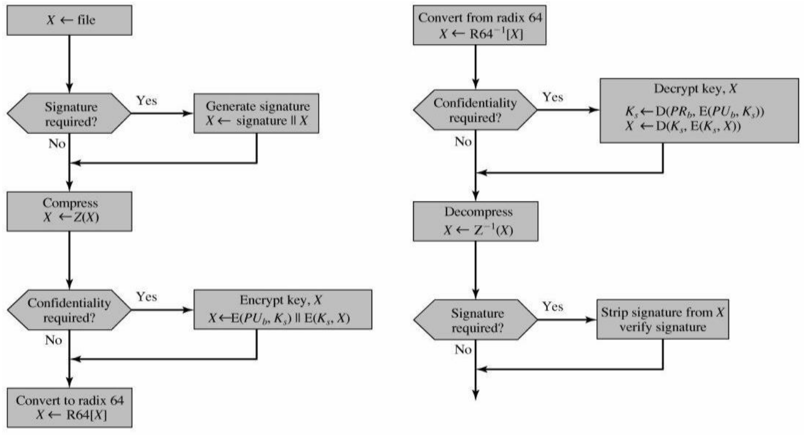
(a) Authentication only

(b) Confidentiality only

(c) Confidentiality and authentication

- **Confidentiality:** Another basic service provided by PGP is confidentiality, which is provided by encrypting messages to be transmitted or to be stored locally as files.
- The sender generates a message and a random 128-bit number to be used as a session key for this message only.
- The message is encrypted, using CAST-128 (or IDEA or 3DES) with the session key.
- The session key is encrypted with RSA, using the recipient's public key, and is prepended to the message.
- The receiver uses RSA with its private key to decrypt and recover the session key.
- The session key is used to decrypt the message.
- **Confidentiality and Authentication:**
- both services may be used for the same message.
- First, a signature is generated for the plaintext message and prepended to the message. Then the plaintext message plus signature is encrypted using CAST-128 (or IDEA or 3DES), and the session key is encrypted using RSA (or ElGamal).
- This sequence is preferable to the opposite: encrypting the message and then generating a signature for the encrypted message.
- It is generally more convenient to store a signature with a plaintext version of a message.
- Furthermore, for purposes of third-party verification, if the signature is performed first, a third party need not be concerned with the symmetric key when verifying the signature.

Transmission and Reception of PGP Messages


$X \leftarrow$ file
Signature required?
Yes
No
Generate signature $X \leftarrow$ signature $\|$ X
Compress $X \leftarrow Z(X)$
Confidentiality required?
Yes
No
Encrypt key, X $X \leftarrow E(PU_b, K_s) \| E(K_s, X)$
Convert to radix 64 $X \leftarrow R64[X]$
Convert from radix 64 $X \leftarrow R64^{-1}[X]$
Confidentiality required?
Yes
No
Decrypt key, X $K_s \leftarrow D(PR_b, E(PU_b, K_s))$ $X \leftarrow D(K_s, E(K_s, X))$
Decompress $X \leftarrow Z^{-1}(X)$
Signature required?
Yes
No
Strip signature from X verify signature

(a) Generic transmission diagram (from A)

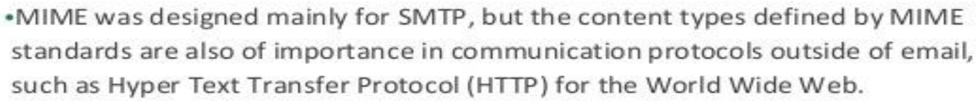
(b) Generic reception diagram (to B)

Multipurpose Internet Mail Extensions (MIME)

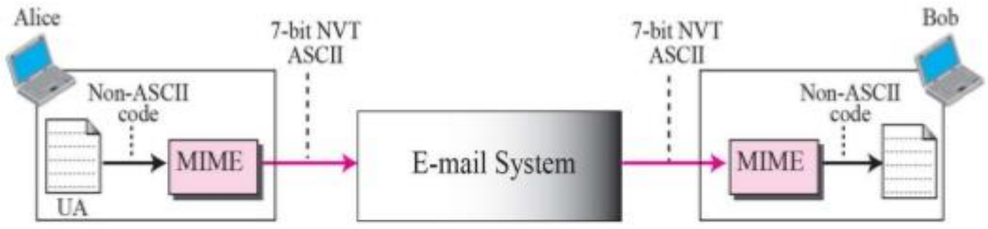
- **Multipurpose Internet Mail Extension (MIME)** is a standard which was proposed by Bell Communications in 1991 in order to expand limited capabilities of email.
- MIME is a kind of *add on or a supplementary protocol* which allows non-ASCII data to be sent through SMTP. It allows the users to exchange different kinds of data files on the Internet: audio, video, images, application programs as well.

- **Why do we need MIME?**
  Limitations of Simple Mail Transfer Protocol (SMTP):

1. SMTP has a very simple structure
2. It’s simplicity however comes with a price as it only send messages in NVT 7-bit ASCII format.
3. It cannot be used for languages that do not support 7-bit ASCII format such as- French, German, Russian, Chinese and Japanese, etc. so it cannot be transmitted using SMTP. So, in order *to make SMTP more broad we use MIME*.
4. It cannot be used to send binary files or video or audio data.



•MIME was designed mainly for SMTP, but the content types defined by MIME standards are also of importance in communication protocols outside of email, such as Hyper Text Transfer Protocol (HTTP) for the World Wide Web.



MIME *transforms non-ASCII data* at sender side to NVT 7-bit data and delivers it to the client SMTP. The message at receiver side is transferred back to the original data. As well as we can send video and audio data using MIME as it transfers them also in 7-bit ASCII data.

S/MIME (Secure/Multipurpose Internet Mail Extensions)

- S/MIME is an extension of the widely implemented Multipurpose Internet Mail Extensions (MIME) encoding standard
- S/MIME uses the RSA public key cryptography algorithm along with the Data Encryption Standard (DES) or Rivest-Shamir-Adleman (RSA) encryption algorithm.
- S/MIME provides the following functions:
- ● **Enveloped data:** This consists of encrypted content of any type and encrypted-content encryption keys for one or more recipients.
- **Signed data:** A digital signature is formed by taking the message digest of the content to be signed and then encrypting that with the private key of the signer. The content plus signature are then encoded using base64 encoding. A signed data message can only be viewed by a recipient with S/MIME capability.
- **Clear-signed data:** As with signed data, a digital signature of the content is formed. However, in this case, only the digital signature is encoded using base64. As a result,

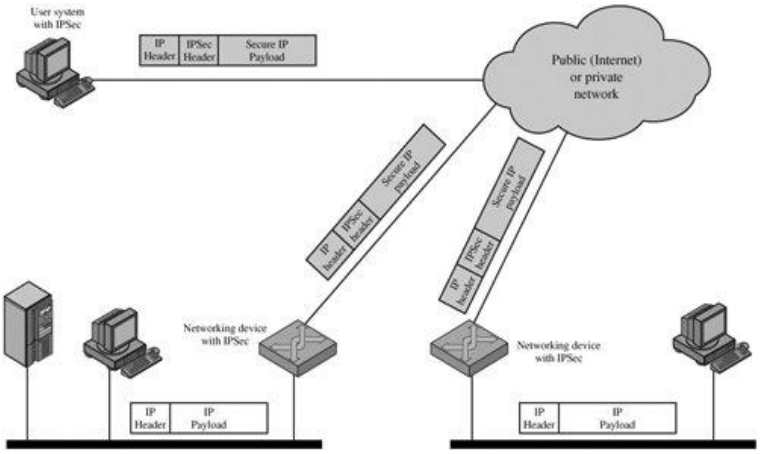
recipients without S/MIME capability can view the message content, although they cannot verify the signature.

- **Signed and enveloped data:** Signed-only and encrypted-only entities may be nested, so that encrypted data may be signed and signed data or clear-signed data may be encrypted.
- **Must:** The definition is an absolute requirement of the specification. An implementation must include this feature or function to be in conformance with the specification.
- ● **Should:** There may exist valid reasons in particular circumstances to ignore this feature or function, but it is recommended that an implementation include the feature or function.

IP security (IPSec)

- The **IP security (IPSec)** is an Internet Engineering Task Force (IETF) standard suite of protocols between 2 communication points across the IP network that provide data authentication, integrity, and confidentiality. It also defines the encrypted, decrypted and authenticated packets. The protocols needed for secure key exchange and key management are defined in it.
- To provide security, the IAB included authentication and encryption as necessary security features in the next-generation IP, which has been issued as IPv6. Fortunately, these security capabilities were designed to be usable both with the current IPv4 and the future IPv6.
- In 1994, the Internet Architecture Board (IAB) issued a report titled "Security in the Internet Architecture"
- **Applications of IPSec:** IPSec provides the capability to secure communications across a LAN, across private and public WANs, and across the Internet. Examples of its use include the following:
- **Secure branch office connectivity over the Internet:** A company can build a secure virtual private network over the Internet or over a public WAN.
- **Secure remote access over the Internet:** An end user whose system is equipped with IP security protocols can make a local call to an Internet service provider (ISP) and gain secure access to a company network.
- **Establishing extranet and intranet connectivity with partners:** IPSec can be used to secure communication with other organizations, ensuring authentication and confidentiality and providing a key exchange mechanism.
- **Enhancing electronic commerce security:** Even though some Web and electronic commerce applications have built-in security protocols, the use of IPSec enhances that security.
- For traffic offsite, through some sort of private or public WAN, IPSec protocols are used. These protocols operate in networking devices, such as a router or firewall, that connect each LAN to the outside world.
- The IPSec networking device will typically encrypt and compress all traffic going into the WAN, and decrypt and decompress traffic coming from the WAN; these operations are transparent to workstations and servers on the LAN.

- Secure transmission is also possible with individual users who dial into the WAN. Such user workstations must implement the IPSec protocols to provide security.
- **An IP Security Scenario**

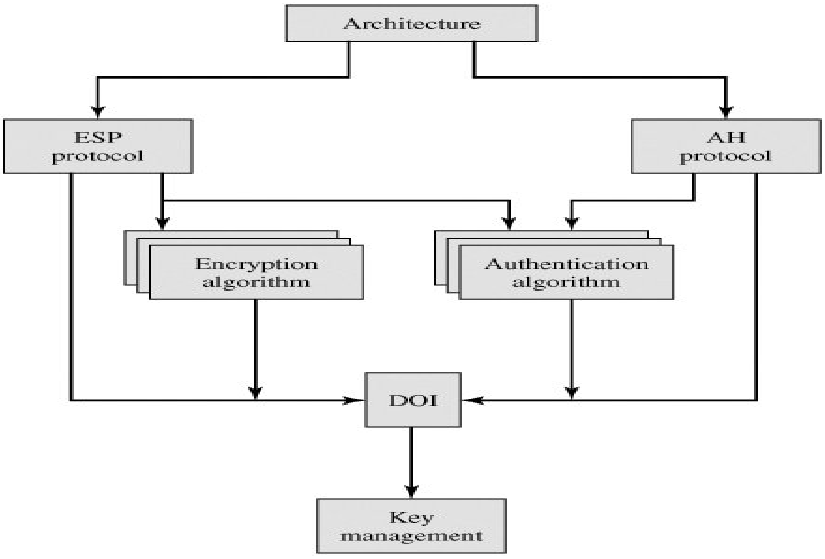


Benefits of IPSec

- When IPSec is implemented in a firewall or router, it provides strong security that can be applied to all traffic crossing the perimeter.
- IPSec in a firewall is resistant to bypass if all traffic from the outside must use IP, and the firewall is the only means of entrance from the Internet into the organization.
- IPSec can provide security for individual users if needed. This is useful for offsite workers and for setting up a secure virtual subnetwork within an organization for sensitive applications.
- IPSec can be transparent to end users. There is no need to train users on security mechanisms, issue keying material on a per-user basis, or revoke keying material when users leave the organization.

IP Security Architecture

- The IPSec specification consists of numerous documents. The most important of these, issued in November of 1998, are RFCs 2401, 2402, 2406, and 2408:
- ● RFC 2401: An overview of a security architecture
- ● RFC 2402: Description of a packet authentication extension to IPv4 and IPv6
- ● RFC 2406: Description of a packet encryption extension to IPv4 and IPv6
- ● RFC 2408: Specification of key management capabilities
- Support for these features is mandatory for IPv6 and optional for IPv4. In both cases, the security features are implemented as extension headers that follow the main IP header.
- The extension header for authentication is known as the Authentication header; that for encryption is known as the Encapsulating Security Payload (ESP) header.



- **Architecture:** Covers the general concepts, security requirements, definitions, and mechanisms defining IPSec technology.

- **Encapsulating Security Payload (ESP):** Covers the packet format and general issues related to the use of the ESP for packet encryption and, optionally, authentication.

● **Authentication Header (AH):** Covers the packet format and general issues related to the use of AH for packet authentication.

● **Encryption Algorithm:** A set of documents that describe how various encryption algorithms are used for ESP.

● **Authentication Algorithm:** A set of documents that describe how various authentication algorithms are used for AH and for the authentication option of ESP.

● **Key Management:** Documents that describe key management schemes.

● **Domain of Interpretation (DOI):** Contains values needed for the other documents to relate to each other. These include identifiers for approved encryption and authentication algorithms, as well as operational parameters such as key lifetime.

- **IPSec Services**

- IPSec provides security services at the IP layer by enabling a system to select required security protocols, determine the algorithm(s) to use for the service(s), and put in place any cryptographic keys required to provide the requested services.

- Two protocols are used to provide security: an authentication protocol designated by the header of the protocol, Authentication Header (AH); and a combined encryption/authentication protocol designated by the format of the packet for that protocol, Encapsulating Security Payload (ESP).

- The services are

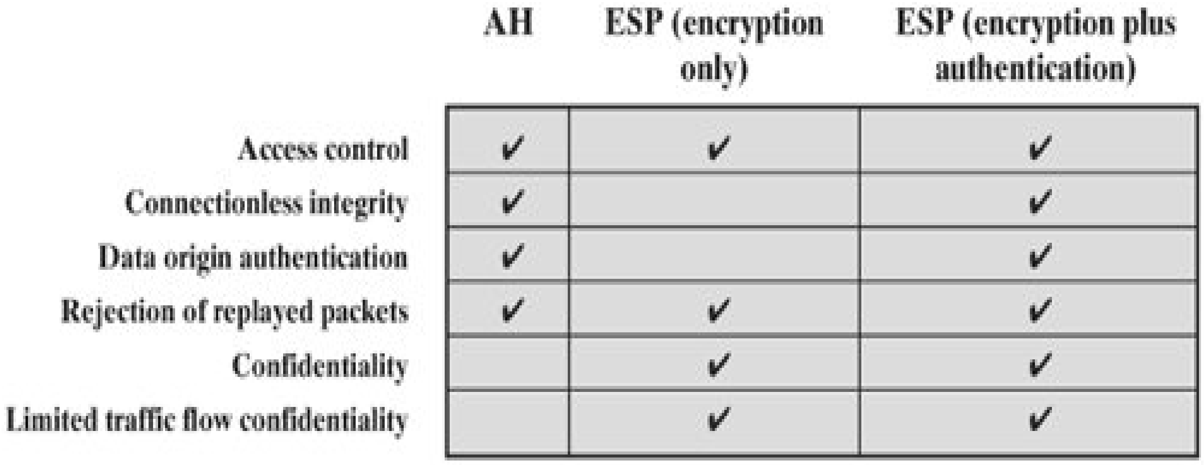
- Access control

● Connectionless integrity

● Data origin authentication

- Rejection of replayed packets (a form of partial sequence integrity)
- Confidentiality (encryption)
- Limited traffic flow confidentiality

- Table shows which services are provided by the AH and ESP protocols.



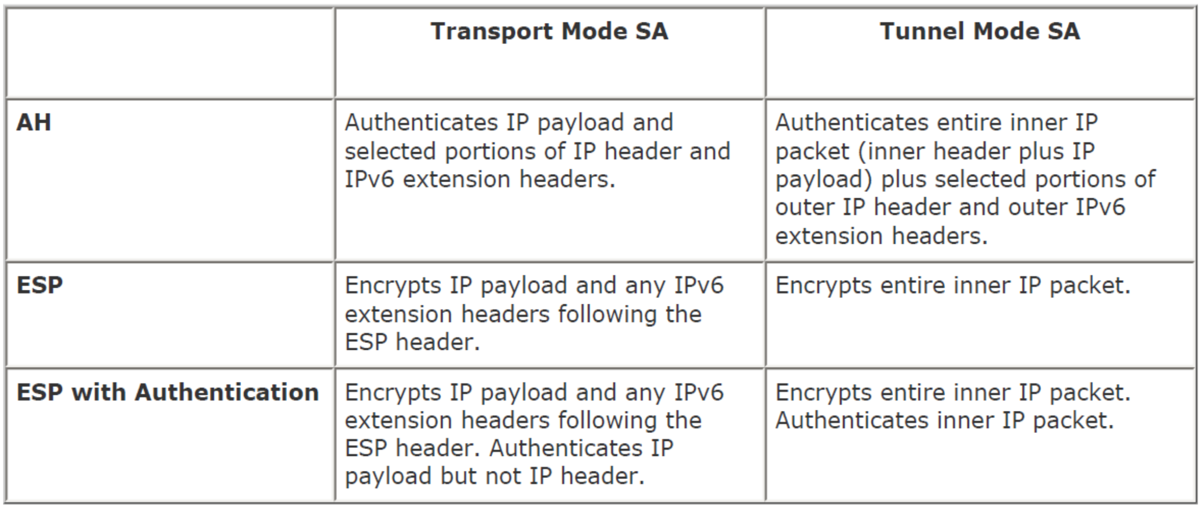
	AH	ESP (encryption only)	ESP (encryption plus authentication)
Access control	✓	✓	✓
Connectionless integrity	✓		✓
Data origin authentication	✓		✓
Rejection of replayed packets	✓	✓	✓
Confidentiality		✓	✓
Limited traffic flow confidentiality		✓	✓

- **Security Associations**
- A key concept that appears in both the authentication and confidentiality mechanisms for IP is the security association (SA).
- An association is a one-way relationship between a sender and a receiver that affords security services to the traffic carried on it.
- If a peer relationship is needed, for two-way secure exchange, then two security associations are required.
- Security services are afforded to an SA for the use of AH or ESP, but not both.
- A security association is uniquely identified by **three parameters:**
- **Security Parameters Index (SPI):** A bit string assigned to this SA and having local significance only. The SPI is carried in AH and ESP headers to enable the receiving system to select the SA under which a received packet will be processed.
- ● **IP Destination Address:** Currently, only unicast addresses are allowed; this is the address of the destination endpoint of the SA, which may be an end user system or a network system such as a firewall or router.
- ● **Security Protocol Identifier:** This indicates whether the association is an AH or ESP security association.
- Hence, in any IP packet, the security association is uniquely identified by the Destination Address inthe IPv4 or IPv6 header and the SPI in the enclosed extension header (AH or ESP).
- A security association is normally defined by the **following parameters:**
- **Sequence Number Counter:** A 32-bit value used to generate the Sequence Number field in AH or ESP headed.

- **Sequence Counter Overflow:** A flag indicating whether overflow of the Sequence Number Counter should generate an auditable event and prevent further transmission of packets on this SA (required for all implementations).
- **Anti-Replay Window:** Used to determine whether an inbound(duplicate packet) AH or ESP packet is a replay.
- **AH Information: is a** Authentication algorithm, keys, key lifetimes, and related parameters being used with AH (required for AH implementations).
- **ESP Information: is a** Encryption and authentication algorithm, keys, initialization values, key lifetimes, and related parameters being used with ESP (required for ESP implementations).
- **Lifetime of This Security Association:** A time interval or byte count after which an SA must be replaced with a new SA (and new SPI) or terminated, plus an indication of which of these actions should occur (required for all implementations).
- **IPSec Protocol Mode:** Tunnel, transport, or wildcard (required for all implementations).
- **Path MTU:** Any observed path maximum transmission unit (maximum size of a packet that can be transmitted without fragmentation) and aging variables (required for all implementations).
- **SA Selectors:**
- SPD (Security Policy Database): entry is defined by a set of IP and upper-layer protocol field values, called *selectors*. In effect, these selectors are used to filter outgoing traffic in order to map it into a particular SA.
- Outbound processing obeys the following general sequence for each IP packet:

1. Compare the values of the appropriate fields in the packet (the selector fields) against the SPD to find a matching SPD entry, which will point to zero or more SAs.
2. Determine the SA if any for this packet and its associated SPI.
3. Do the required IPSec processing (i.e., AH or ESP processing).
4. The following selectors determine an SPD entry:
5. **Destination IP Address:** This may be a single IP address, an enumerated list or range of addresses, or a wildcard (mask) address.
6. **Source IP Address:** This may be a single IP address, an enumerated list or range of addresses, or a wildcard (mask) address. The latter two are required to support more than one source system sharing the same SA (e.g., behind a firewall).
7. **Data Sensitivity Level:** Used for systems providing information flow security (e.g., Secret or Unclassified).
8. **Transport Layer Protocol:** Obtained from the IPv4 Protocol or IPv6 Next Header field. This may be an individual protocol number, a list of protocol numbers, or a range of protocol numbers.
9. **Source and Destination Ports:** These may be individual TCP or UDP port values, an enumerated list of ports, or a wildcard port.

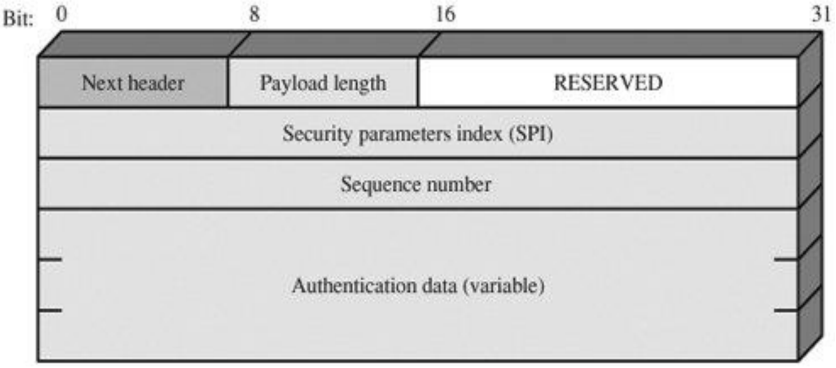
Tunnel Mode and Transport Mode Functionality



	Transport Mode SA	Tunnel Mode SA
AH	Authenticates IP payload and selected portions of IP header and IPv6 extension headers.	Authenticates entire inner IP packet (inner header plus IP payload) plus selected portions of outer IP header and outer IPv6 extension headers.
ESP	Encrypts IP payload and any IPv6 extension headers following the ESP header.	Encrypts entire inner IP packet.
ESP with Authentication	Encrypts IP payload and any IPv6 extension headers following the ESP header. Authenticates IP payload but not IP header.	Encrypts entire inner IP packet. Authenticates inner IP packet.

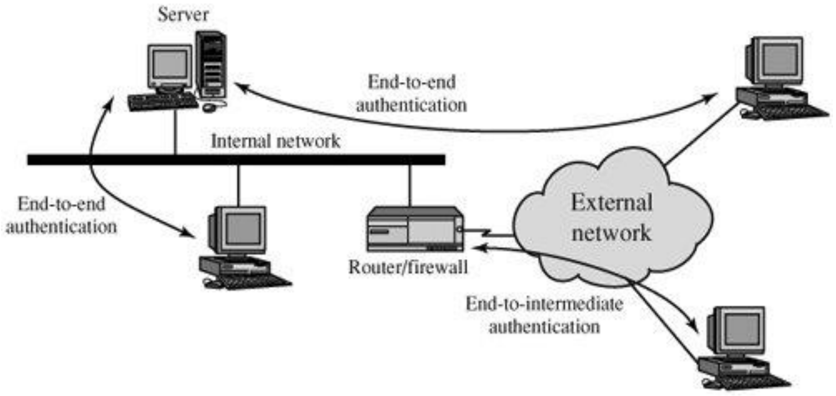
Authentication Header

- The Authentication Header provides support for data integrity and authentication of IP packets.
- The data integrity feature ensures that undetected modification to a packet's content in transit is not possible.
- The authentication feature enables an end system or network device to authenticate the user or application and filter traffic accordingly; it also prevents the address spoofing attacks observed in today's Internet.
- The Authentication Header consists of the following fields


Bit: 0
8
16
31
Next header
Payload length
RESERVED
Security parameters index (SPI)
Sequence number
Authentication data (variable)

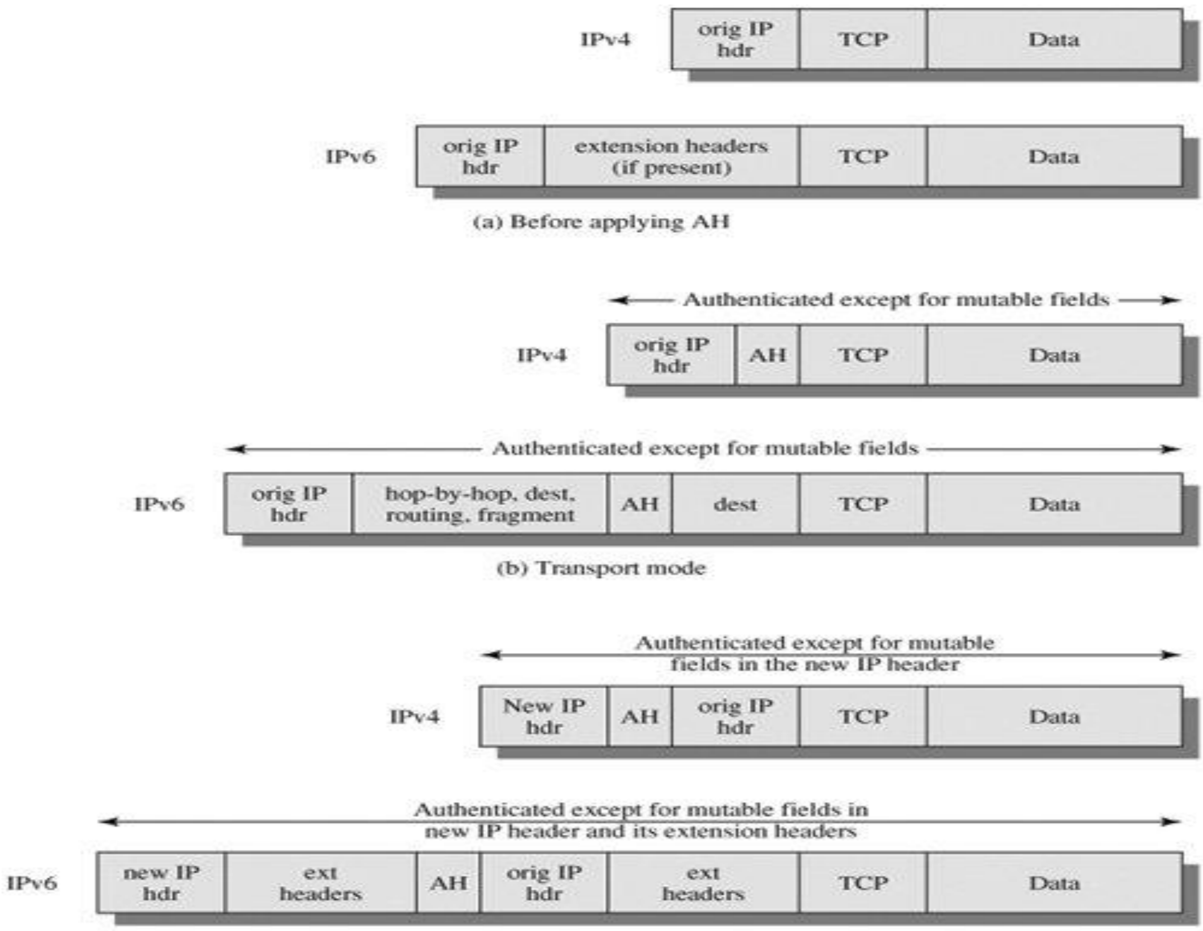
- **Next Header (8 bits):** Identifies the type of header immediately following this header.
- **Payload Length (8 bits):** Length of Authentication Header in 32-bit words, minus 2. For example, the default length of the authentication data field is 96 bits, or three 32-bit words. With a three-word fixed header, there are a total of six words in the header, and the Payload Length field has a value of 4.
- **Reserved (16 bits):** For future use.
- **Security Parameters Index (32 bits):** Identifies a security association.
- **Sequence Number (32 bits):** A monotonically increasing counter value

- **Authentication Data (variable):** A variable-length field (must be an integral number of 32-bit words) that contains the Integrity Check Value (ICV), or MAC, for this packet, discussed later.
- **Transport and Tunnel Modes**



Server
End-to-end authentication
Internal network
End-to-end authentication
Router/firewall
External network
End-to-intermediate authentication

- In one case, authentication is provided directly between a server and client workstations; the workstation can be either on the same network as the server or on an external network. As long as the workstation and the server share a protected secret key, the authentication process is secure. This case uses a transport mode SA.
- In the other case, a remote workstation authenticates itself to the corporate firewall, either for access to the entire internal network or because the requested server does not support the authentication feature. This case uses a tunnel mode SA.



IPv4
orig IP hdr
TCP
Data
IPv6
orig IP hdr
extension headers (if present)
TCP
Data
Authenticated except for mutable fields
IPv4
orig IP hdr
AH
TCP
Data
Authenticated except for mutable fields
IPv6
orig IP hdr
hop-by-hop, dest, routing, fragment
AH
dest
TCP
Data
Authenticated except for mutable fields in the new IP header
IPv4
New IP hdr
AH
orig IP hdr
TCP
Data
Authenticated except for mutable fields in new IP header and its extension headers
IPv6
new IP hdr
ext headers
AH
orig IP hdr
ext headers
TCP
Data

(a) Before applying AH

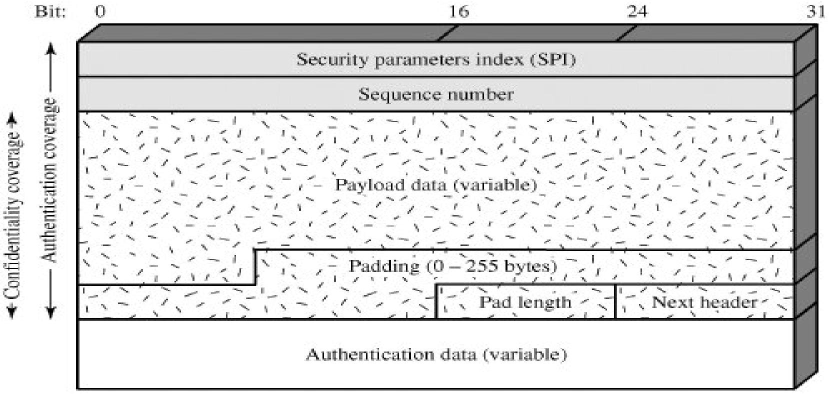
(b) Transport mode

(c) Tunnel mode

Encapsulating Security Payload

- The Encapsulating Security Payload provides confidentiality services, including confidentiality of message contents and limited traffic flow confidentiality. As an optional feature, ESP can also provide an authentication service.

- **ESP Packet Format**



- **Security Parameters Index (32 bits):** Identifies a security association.

- **Sequence Number (32 bits):** A monotonically increasing counter value; this provides an antireplay function, as discussed for AH.

- **Payload Data (variable):** This is a transport-level segment (transport mode) or IP packet (tunnel mode) that is protected by encryption

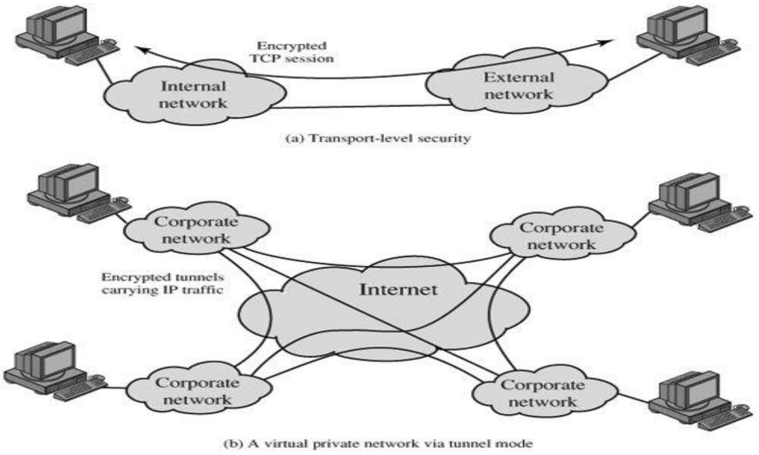
- **Padding (0255 bytes):** used to maintain the block of the any type of encryption algorithms

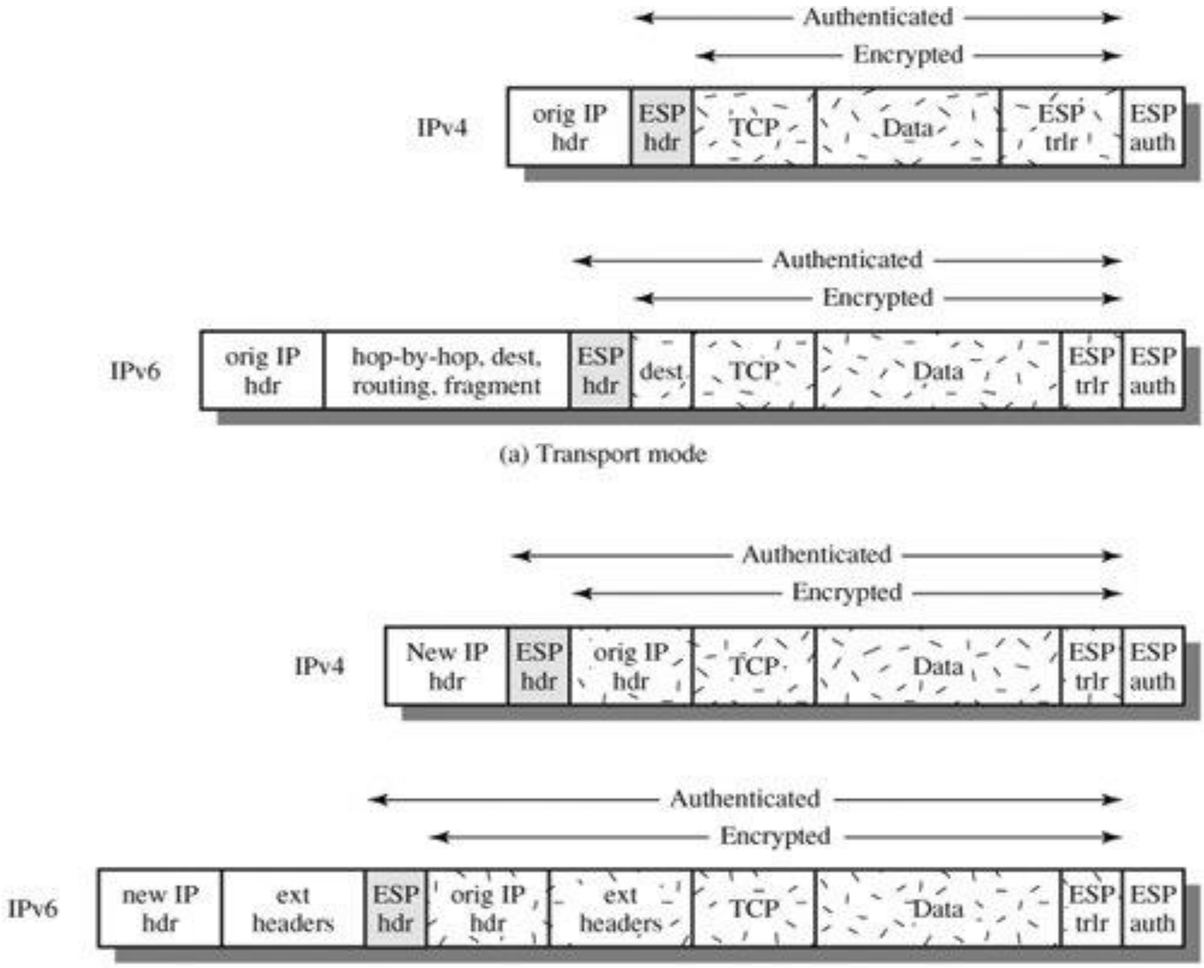
- **Pad Length (8 bits):** Indicates the number of pad bytes immediately preceding this field.

- **Next Header (8 bits):** Identifies the type of data contained in the payload data field by identifying the first header in that payload (for example, an extension header in IPv6, or an upper-layer protocol such as TCP).

- **Authentication Data (variable):** A variable-length field (must be an integral number of 32-bit words) that contains the Integrity Check Value computed over the ESP packet minus the Authentication Data field.

- **Transport-Mode vs. Tunnel-Mode Encryption**





(a) Transport mode

(b) Tunnel mode

Internet Key Exchange

- The key management portion of IPsec involves the determination and distribution of secret keys.

- A typical requirement is four keys for communication between two applications: transmit and receive pairs for both integrity and confidentiality.

- The IPsec Architecture document mandates support for two types of key management:

- **Manual:** A system administrator manually configures each system with its own keys and with the keys of other communicating systems. This is practical for small, relatively static environments.

- **Automated:** An automated system enables the on-demand creation of keys for SAs and facilitates the use of keys in a large distributed system with an evolving configuration.

- The default automated key management protocol for IPsec is referred to as ISAKMP/Oakley and consists of the following elements:

- **Oakley Key Determination Protocol:**

- Oakley is a key exchange protocol based on the Diffie-Hellman algorithm but providing added security. Oakley is generic in that it does not dictate specific formats.

- **• Internet Security Association and Key Management Protocol (ISAKMP):**

- ISAKMP provides a framework for Internet key management and provides the specific protocol support, including formats, for negotiation of security attributes.

- ISAKMP by itself does not dictate a specific key exchange algorithm; rather, ISAKMP consists of a set of message types that enable the use of a variety of key exchange algorithms. Oakley is the specific key exchange algorithm mandated for se with the initial version of ISAKMP.

- In IKEv2, the terms Oakley and ISAKMP are no longer used, and there are significant differences from the use of Oakley and ISAKMP in IKEv1. Nevertheless, the basic functionality is the same.

Web Security

- The World Wide Web is fundamentally a client/server application running over the Internet and TCP/IP intranets.

- A number of approaches to providing Web security are possible. The various approaches that have been considered are similar in the services they provide and, to some extent, in the mechanisms that they use, but they differ with respect to their scope of applicability and their relative location within the TCP/ IP protocol stack.

- Secure socket layer (SSL) provides security services between TCP and applications that use TCP. The Internet standard version is called transport layer service (TLS).

- SSL/TLS provides confidentiality using symmetric encryption and message integrity using a message authentication code.

- SSL/TLS includes protocol mechanisms to enable two TCP users to determine the security mechanisms and services they will use.

- ● Secure electronic transaction (SET) is an open encryption and security specification designed to protect credit card transactions on the Internet.

Secure Socket Layer and Transport Layer Security

- Two important SSL concepts are the SSL session and the SSL connection, which are defined in the specification as follows:

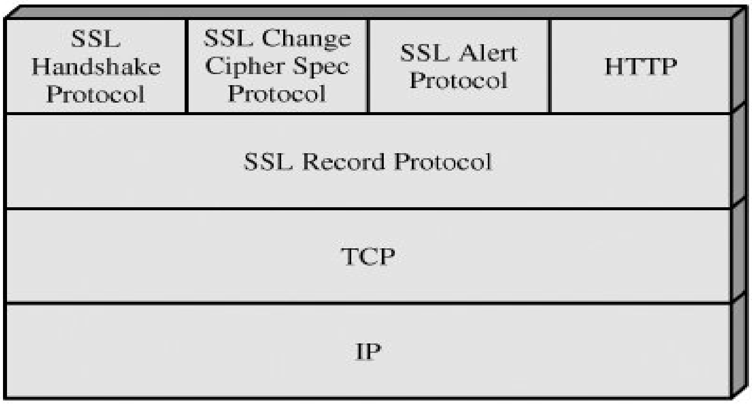
- **Connection:** Transport to provide the service between client and server

- A connection is a transport (in the OSI layering model definition) that provides a suitable type of service. For SSL, such connections are peer-to-peer relationships. The connections are transient. Every connection is associated with one session.

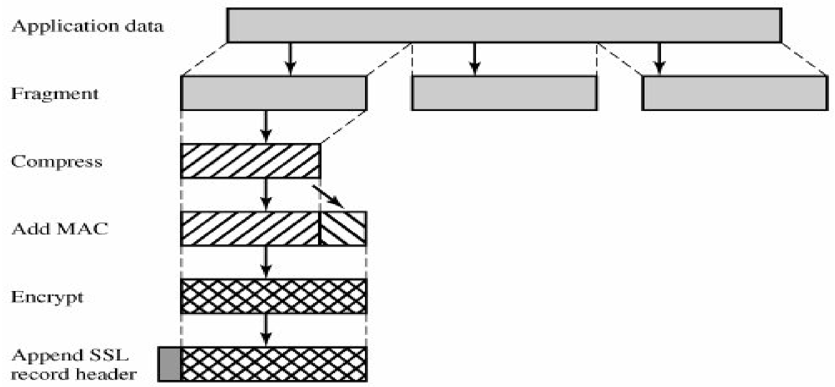
- **Session:** Association between client and server

- An SSL session is an association between a client and a server. Sessions are created by the Handshake Protocol. Sessions define a set of cryptographic security parameters, which can be shared among multiple connections. Sessions are used to avoid the expensive negotiation of new security parameters for each connection.

- **A session state is defined by the following parameters**
- **Session identifier:** An arbitrary byte sequence chosen by the server to identify an active or resumable session state.
- **Peer certificate:** An X509.v3 certificate of the peer. This element of the state may be null.
- **Compression method:** The algorithm used to compress data prior to encryption.
- **Cipher spec:** Specifies the bulk data encryption algorithm (such as null, AES, etc.) and a hash algorithm (such as MD5 or SHA-1) used for MAC calculation. It also defines cryptographic attributes such as the hash size.
- **Master secret:** 48-byte secret key shared between the client and server.
- **Is resumable:** A flag indicating whether the session can be used to initiate new connections.
- A connection state is defined by the following parameters:
- **Server and client random:** Byte sequences that are chosen by the server and client for each connection.
- **Server write MAC secret:** The secret key used in MAC operations on data sent by the server.
- **Client write MAC secret:** The secret key used in MAC operations on data sent by the client.
- **Server write key:** The conventional encryption key for data encrypted by the server and decrypted by the client.
- **Client write key:** The conventional encryption key for data encrypted by the client and decrypted by the server.
- **Initialization vectors:** When a block cipher in CBC mode is used, an initialization vector (IV) is maintained for each key. This field is first initialized by the SSL Handshake Protocol. Thereafter the final ciphertext block from each record is preserved for use as the IV with the following record.
- **Sequence numbers:** Each party maintains separate sequence numbers for transmitted and received messages for each connection.
- **SSL Architecture**
- SSL is designed to make use of TCP to provide a reliable end-to-end secure service. SSL is not a single protocol but rather two layers of protocols

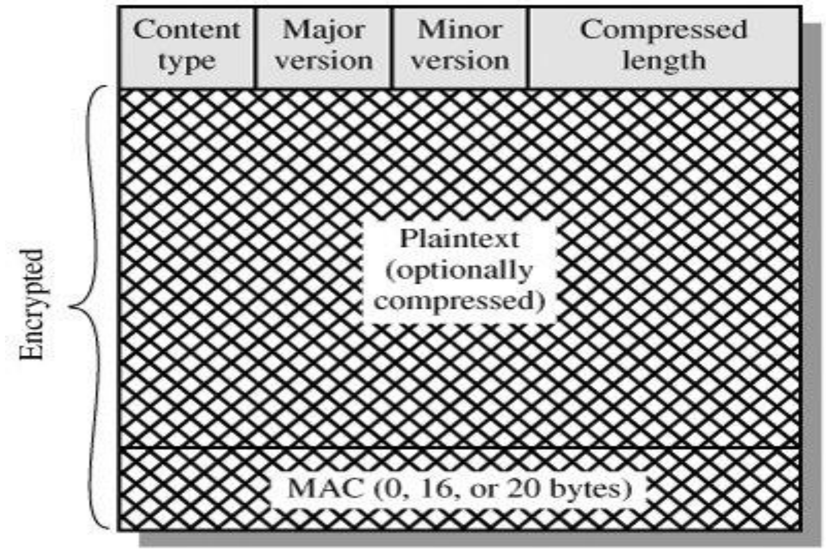


- The SSL Record Protocol provides basic security services to various higher-layer protocols.
- In particular, the Hypertext Transfer Protocol (HTTP), which provides the transfer service for Web client/server interaction, can operate on top of SSL.
- Three higher-layer protocols are defined as part of SSL: the Handshake Protocol, The Change Cipher Spec Protocol, and the Alert Protocol. These SSL-specific protocols are used in the management of SSL exchanges and are examined later in this section.
- **SSL Record Protocol**
- The SSL Record Protocol provides two services for SSL connections:
- **Confidentiality:** The Handshake Protocol defines a shared secret key that is used for conventional encryption of SSL payloads.
- **Message Integrity:** The Handshake Protocol also defines a shared secret key that is used to form a message authentication code (MAC).
- Fig. indicates the overall operation of the SSL Record Protocol.
-

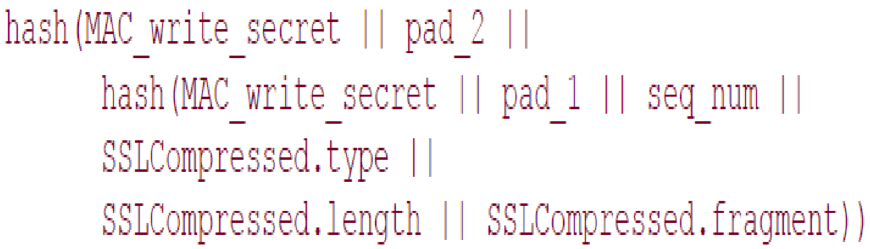


- The Record Protocol takes an application message to be transmitted, fragments the data into manageable blocks, optionally compresses the data, applies a MAC, encrypts, adds a header, and transmits the resulting unit in a TCP segment.
- Received data are decrypted, verified, decompressed, and reassembled and then delivered to higher-level users.
- The final step of SSL Record Protocol processing is to prepend a header, consisting of the following fields:
- • **Content Type (8 bits):** The higher layer protocol used to process the enclosed fragment.
- • **Major Version (8 bits):** Indicates major version of SSL in use. For SSLv3, the value is 3.
- **Minor Version (8 bits):** Indicates minor version in use. For SSLv3, the value is 0.
- • **Compressed Length (16 bits):** The length in bytes of the plaintext fragment (or compressed fragment if compression is used). The maximum value is 2 power**Minor Version (8 bits):** Indicates minor version in use. For SSLv3, the value is 0.

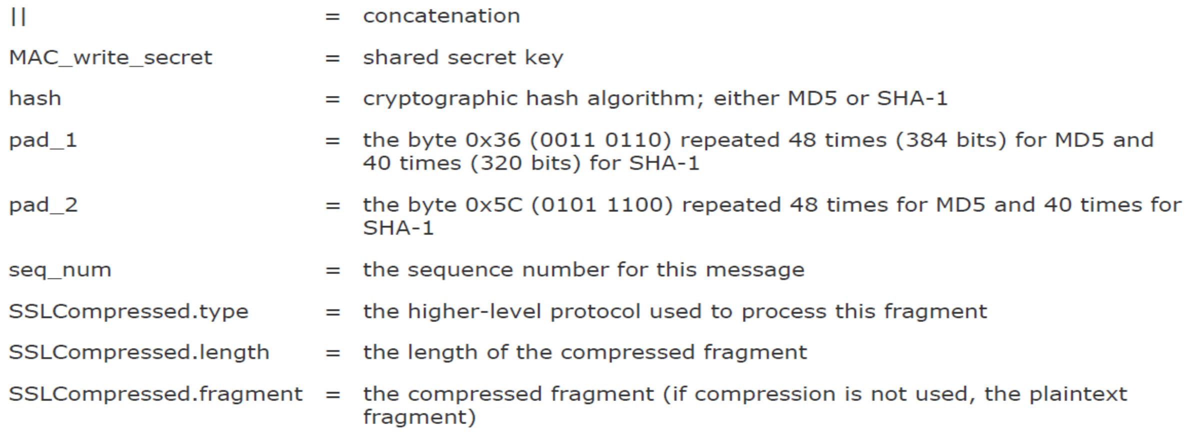
- ● **Compressed Length (16 bits):** The length in bytes of the plaintext fragment (or compressed fragment if compression is used). The maximum value is 2 power 14 + 2048.



- The next step in processing is to compute a **message authentication code** over the compressed data.
- For this purpose, a shared secret key is used. The calculation is defined as



```
hash(MAC_write_secret || pad_2 ||
    hash(MAC_write_secret || pad_1 || seq_num ||
    SSLCompressed.type ||
    SSLCompressed.length || SSLCompressed.fragment))
```

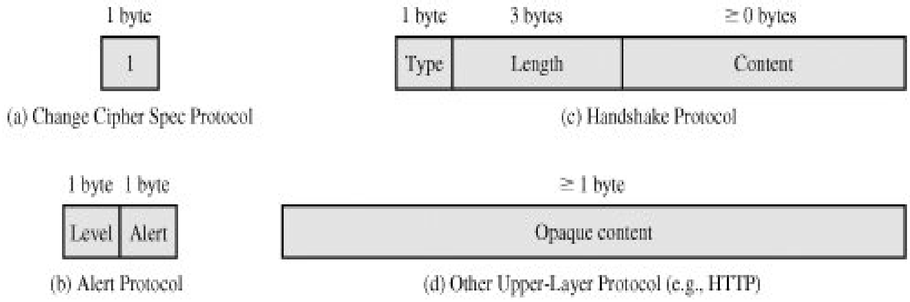


||	=	concatenation
MAC_write_secret	=	shared secret key
hash	=	cryptographic hash algorithm; either MD5 or SHA-1
pad_1	=	the byte 0x36 (0011 0110) repeated 48 times (384 bits) for MD5 and 40 times (320 bits) for SHA-1
pad_2	=	the byte 0x5C (0101 1100) repeated 48 times for MD5 and 40 times for SHA-1
seq_num	=	the sequence number for this message
SSLCompressed.type	=	the higher-level protocol used to process this fragment
SSLCompressed.length	=	the length of the compressed fragment
SSLCompressed.fragment	=	the compressed fragment (if compression is not used, the plaintext fragment)

Change Cipher Spec Protocol

- The Change Cipher Spec Protocol is one of the three SSL-specific protocols that use the SSL Record Protocol, and it is the simplest.

- This protocol consists of a single message fig below, which consists of a single byte with the value 1. The sole purpose of this message is to cause the pending state to be copied into the current state, which updates the cipher suite to be used on this connection.



(a) Change Cipher Spec Protocol

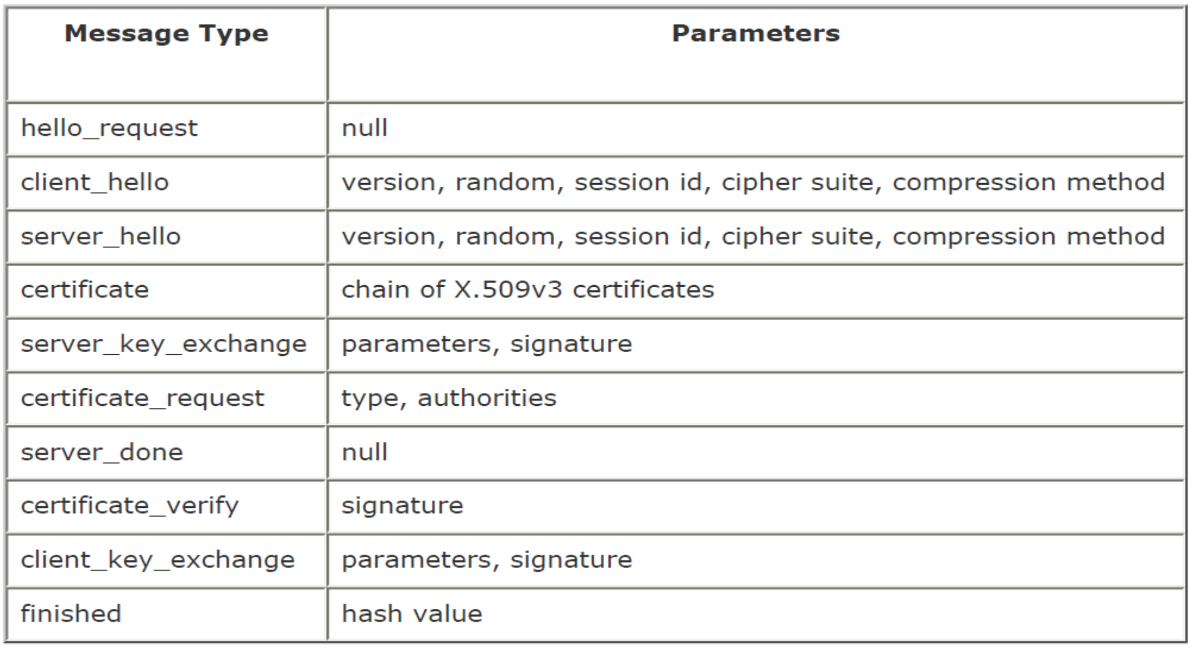
(c) Handshake Protocol

(b) Alert Protocol

(d) Other Upper-Layer Protocol (e.g., HTTP)

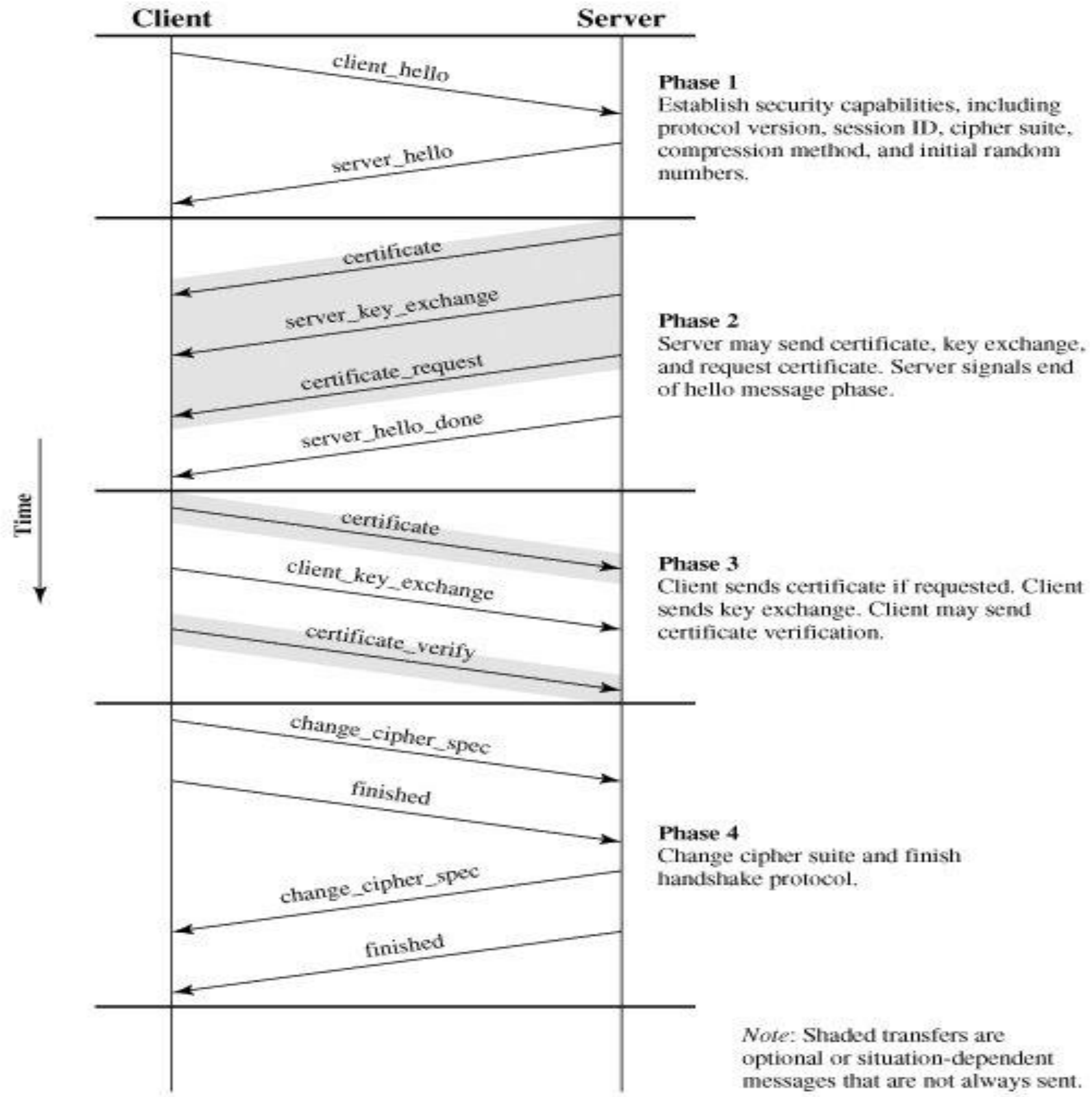
Handshake Protocol

- The Handshake Protocol consists of a series of messages exchanged by client and server.
- Each message has three fields:
- **Type (1 byte):** Indicates one of 10 messages.
- **Length (3 bytes):** The length of the message in bytes.
- **Content (0 bytes):** The parameters associated with this message; these are listed in below table.



Message Type	Parameters
hello_request	null
client_hello	version, random, session id, cipher suite, compression method
server_hello	version, random, session id, cipher suite, compression method
certificate	chain of X.509v3 certificates
server_key_exchange	parameters, signature
certificate_request	type, authorities
server_done	null
certificate_verify	signature
client_key_exchange	parameters, signature
finished	hash value

- The initial exchange needed to establish a logical connection between client and server. The exchange can be viewed as having four phases.


Client
Server
client_hello
server_hello
Phase 1
Establish security capabilities, including protocol version, session ID, cipher suite, compression method, and initial random numbers.
certificate
server_key_exchange
certificate_request
server_hello_done
Phase 2
Server may send certificate, key exchange, and request certificate. Server signals end of hello message phase.
Time
certificate
client_key_exchange
certificate_verify
Phase 3
Client sends certificate if requested. Client sends key exchange. Client may send certificate verification.
change_cipher_spec
finished
change_cipher_spec
finished
Phase 4
Change cipher suite and finish handshake protocol.
Note: Shaded transfers are optional or situation-dependent messages that are not always sent.

Alert Protocol

- The Alert Protocol is used to convey SSL-related alerts to the peer entity. As with other applications that use SSL, alert messages are compressed and encrypted, as specified by the current state.
- Each message in this protocol consists of two bytes.
- The first byte takes the value warning(1) or fatal(2) to convey the severity of the message.
- If the level is fatal, SSL immediately terminates the connection. Other connections on the same session may continue, but no new connections on this session may be established.
- The second byte contains a code that indicates the specific alert. First, we list those alerts that are always fatal (definitions from the SSL specification):
- **unexpected_message:** An inappropriate message was received.
- ● **bad_record_mac:** An incorrect MAC was received.

- ● **decompression_failure:** The decompression function received improper input (e.g., unable to decompress or decompress to greater than maximum allowable length).
- ● **handshake_failure:** Sender was unable to negotiate an acceptable set of security parameters given the options available.
- ● **illegal_parameter:** A field in a handshake message was out of range or inconsistent with other fields.
- The remainder of the alerts are the following:

● **close notify:** Notifies the recipient that the sender will not send any more messages on this connection. Each party is required to send a close notify alert before closing the write side of a connection.

● **no_certificate:** May be sent in response to a certificate request if no appropriate certificate is available.

● **bad_certificate:** A received certificate was corrupt (e.g., contained a signature that did not verify).

● **unsupported_certificate:** The type of the received certificate is not supported.

● **certificate_revoked:** A certificate has been revoked by its signer.

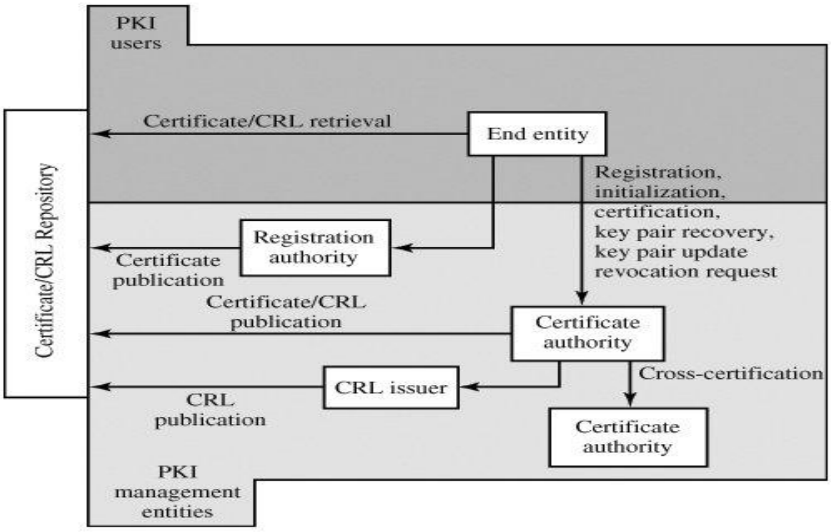
● **certificate_expired:** A certificate has expired.

● **certificate_unknown:** Some other unspecified issue arose in processing the certificate, rendering it unacceptable.

Public-Key Infrastructure

- public-key infrastructure (PKI) as the set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates based on asymmetric cryptography.
- The principal objective for developing a PKI is to enable secure, convenient, and efficient acquisition of public keys.
- The Elements of PKI Model:
- **End entity:** A generic term used to denote end users, devices (e.g., servers, routers), or any other entity that can be identified in the subject field of a public key certificate. End entities typically consume and/or support PKI-related services.
- **Certification authority (CA):** The issuer of certificates and (usually) certificate revocation lists (CRLs). It may also support a variety of administrative functions, although these are often delegated to one or more Registration Authorities.
- **Registration authority (RA):** An optional component that can assume a number of administrative functions from the CA. The RA is often associated with the End Entity registration process, but can assist in a number of other areas as well.
- **CRL issuer:** An optional component that a CA can delegate to publish CRLs.
- **Repository:** A generic term used to denote any method for storing certificates and CRLs so that they can be retrieved by End Entities.

PKI Architectural Model


PKI users
Certificate/CRL retrieval
End entity
Registration, initialization, certification, key pair recovery, key pair update revocation request
Certificate/CRL Repository
Registration authority
Certificate publication
Certificate/CRL publication
Certificate authority
Cross-certification
CRL issuer
CRL publication
Certificate authority
PKI management entities

- **Registration:** This is the process whereby a user first makes itself known to a CA (directly, or through an RA), prior to that CA issuing a certificate or certificates for that user. Registration begins the process of enrolling in a PKI.
- **Initialization:** Before a client system can operate securely, it is necessary to install key materials that have the appropriate relationship with keys stored elsewhere in the infrastructure.
- **Certification:** This is the process in which a CA issues a certificate for a user's public key, and returns that certificate to the user's client system and/or posts that certificate in a repository.
- ● **Key pair recovery:** Key pairs can be used to support digital signature creation and verification encryption and decryption, or both.
- **Key pair update:** All key pairs need to be updated regularly (i.e., replaced with a new key pair) and new certificates issued. Update is required when the certificate lifetime expires and as a result of certificate revocation.
- ● **Revocation request:** An authorized person advises a CA of an abnormal situation requiring certificate revocation. Reasons for revocation include private key compromise, change in affiliation, and name change.
- ● **Cross certification:** Two CAs exchange information used in establishing a cross-certificate. A cross-certificate is a certificate issued by one CA to another CA that contains a CA signature key used for issuing certificates.

Reference:

1. William Stallings, Cryptography and Network Security, 6th Edition, Pearson Education, March 2013.
2. Behrouz A. fFerouzan, “Cryptography & Network Security”, Tata McGraw Hill, 2007