VAAGESWARI COLLEGE OF ENGINEERING
(Sponsored by SREE VAAGESWARI EDUCATIONAL SOCIETY)
Accredited by NAAC with A+ Grade
Approved by AICTE New Delhi and Affiliated to JNTUH. Hyderabad
ESTD : 2005

SUB: Cryptography and Network Security

IV. B.Tech I Semster

FACULTY NAME:R.Sagar

(COMPUTER SCIENCE AND ENGINEERING)
VAAGESWARI COLLEGE OF ENGINEERING

CNS

UNIT-1

SECURITY CONCEPTS

INTRODUCTION

Cryptography is technique of securing information and communications through use of codes so that only those person for whom the information is intended can understand it and process it. Thus preventing unauthorized access to information. The prefix "crypt" means "hidden" and suffix graphy means "writing".One is confidentiality which basically means that we need to be sure that nobody will see our information as it travels across a network. Authentication and access control is also another capability provided by cryptography. Some other capabilities provided by cryptography are non-repudiation and integrity.

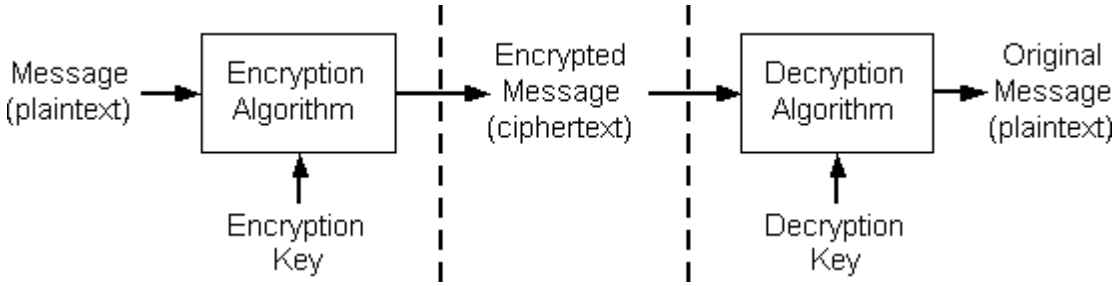
In Cryptography the techniques which are use to protect information are obtained from mathematical concepts and a set of rule based calculations known as algorithms to convert messages in ways that make it hard to decode it. These algorithms are used for cryptographic key generation, digital signing, verification to protect data privacy, web browsing on internet and to protect confidential transactions such as credit card and debit card transactions.

Basic Concepts

Cryptography The art or science encompassing the principles and methods of transforming anintelligible message into one that is unintelligible, and then retransforming that message back to itsoriginal form

Plaintext can refer to anything which humans can understand and/or relate to. This may be as simple as English sentences, a script, or Java code. If you can make sense of what is written, then it is in plaintext.

Ciphertext, or encrypted text, is a series of randomized letters and numbers which humans cannot make any sense of. An encryption algorithm takes in a plaintext message, runs the algorithm on the plaintext, and produces a ciphertext. The ciphertext can be reversed through the process of decryption, to produce the original plaintext.



Key Some critical information used by the cipher, known only to the sender& receiver.

The Basic Principles

1. Encryption

In a simplest form, encryption is to convert the data in some unreadable form. This helps in protecting the privacy while sending the data from sender to receiver. On the receiver side, the data can be decrypted and can be brought back to its original form. The reverse of encryption is called as decryption. The concept of encryption and decryption requires some extra information for encrypting and decrypting the data. This information is known as key. There may be cases when same key can be used for both encryption and decryption while in certain cases, encryption and decryption may require different keys.

2. Authentication

This is another important principle of cryptography. In a layman's term, authentication ensures that the message was originated from the originator claimed in the message. Suppose, Alice sends a message to Bob and now Bob wants proof that the message has been indeed sent by Alice. This can be made possible if Alice performs some action on message that Bob knows only Alice can do. Well, this forms the basic fundamental of Authentication.

3. Integrity

Now, one problem that a communication system can face is the loss of integrity of messages being sent from sender to receiver. This means that Cryptography should ensure that the messages that are received by the receiver are not altered anywhere on the communication path. This can be achieved by using the concept of cryptographic hash.

4. Non Repudiation

What happens if Alice sends a message to Bob but denies that she has actually sent the message? Cases like these may happen and cryptography should prevent the originator or sender to act this way. One popular way to achieve this is through the use of digital signatures.

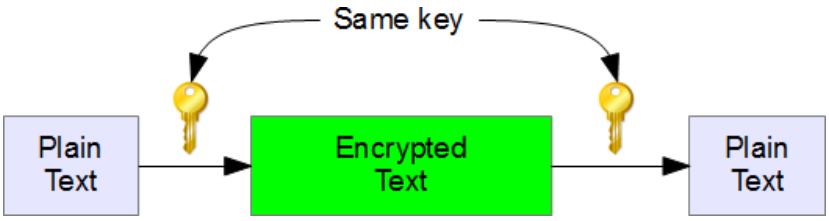
Types of Cryptography

There are three types of cryptography techniques :

1. Secret key Cryptography
2. Public key cryptography
3. Hash Functions

1. Secret Key Cryptography

This type of cryptography technique uses just a single key. The sender applies a key to encrypt a message while the receiver applies the same key to decrypt the message. Since only single key is used so we say that this is a symmetric encryption.



The biggest problem with this technique is the distribution of key as this algorithm makes use of single key for encryption or decryption.

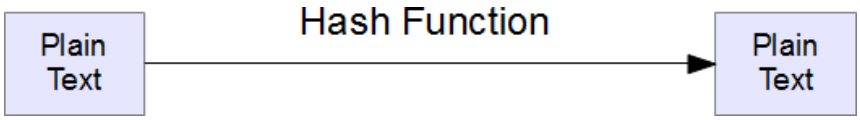
2. Public Key Cryptography

This type of cryptography technique involves two key crypto system in which a secure communication can take place between receiver and sender over insecure communication channel. Since a pair of keys is applied here so this technique is also known as asymmetric encryption.

In this method, each party has a private key and a public key. The private is secret and is not revealed while the public key is shared with all those whom you want to communicate with. If Alice wants to send a message to bob, then Alice will encrypt it with Bob's public key and Bob can decrypt the message with its private key.

This is what we use when we setup public key authentication in openssh to login from one server to another server in the backend without having to enter the password.

3. Hash Functions



This technique does not involve any key. Rather it uses a fixed length hash value that is computed on the basis of the plain text message. Hash functions are used to check the integrity of the message to ensure that the message has not be altered,compromised or affected by virus.

So we see that how different types of cryptography techniques (described above) are used to implement the basic principles that we discussed earlier. In the future article of this series, we'll cover more advanced topics on Cryptography.

THE NEED FOR SECURITY

Most initial computer applications had no or at best, very little security.

The need for security:

1. **Protecting the functionality of the organization:**

   The decision maker in organizations must set policy and operates their organization in compliance with the complex, shifting legislation, efficient and capable applications.

2. **Enabling the safe operation of applications:**

   The organization is under immense pressure to acquire and operates integrated, efficient and capable applications. The modern organization needs to create an environment that safeguards application using the organizations IT systems, particularly those application that serves as important elements of the infrastructure of the organization.

3. **Protecting the data that the organization collect and use:**

   Data in the organization can be in two forms are either in rest or in motion, the motion of data signifies that data is currently used or processed by the system. The values of the data motivated the attackers to steal or corrupts the data. This is essential for the integrity and the values of the organization's data. Information security ensures the protection of both data in motion as well as data in rest.

4. **Safeguarding technology assets in organizations:**

   The organization must add intrastate services based on the size and scope of the organization. Organizational growth could lead to the need for public key infrastructure, PKI an integrated system of the software, encryption methodologies. The information security mechanism used by large organizations is complex in comparison to a small organization. The small organization generally prefers symmetric key encryption of data.

SECURITY APPROACHES

1. **Trusted Systems:**

A trusted system is a computer system that can be trusted to a specified extent to enforce aspecified security policy.

Trusted systems were initially of primary interest to the military. However, these days, the concept hasspanned across various areas, most prominently in the banking and financial community, but the conceptnever caught on. Trusted systems often use the term reference monitor.

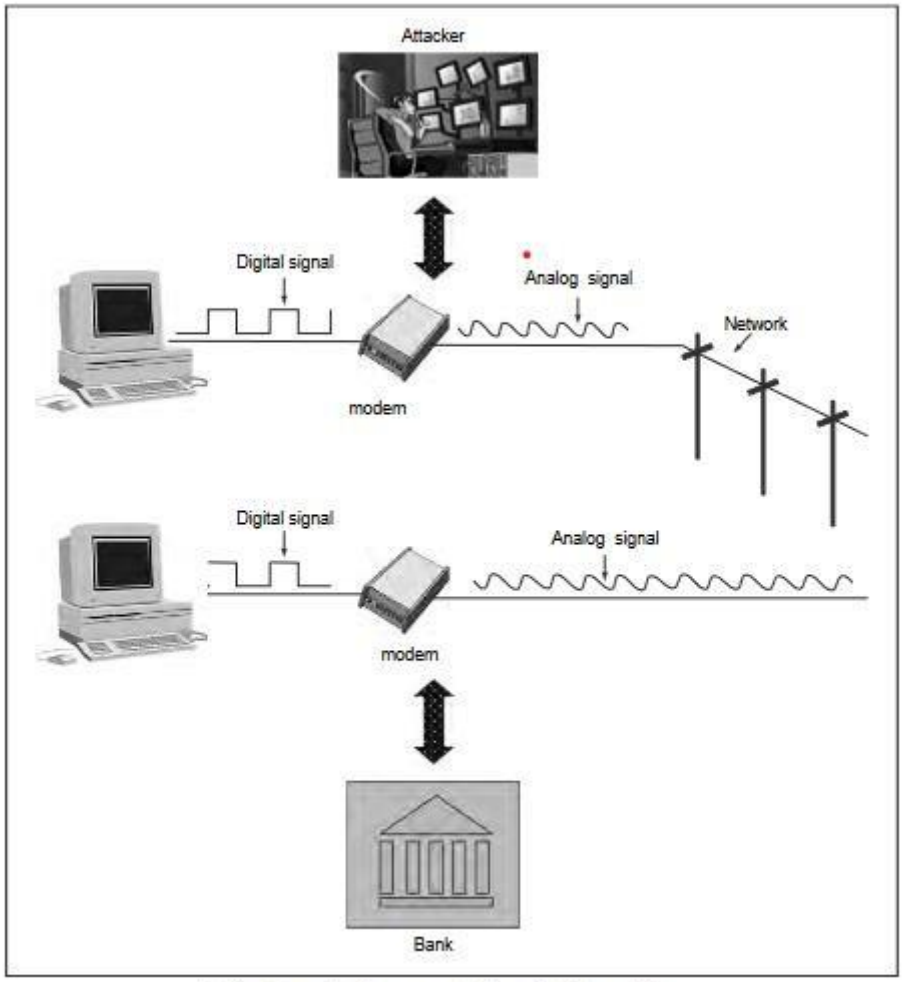


Fig. 1.3 *Attacks can now be launched from a distance*

It is mainly responsible for all the decisions related to access controls. Naturally, following are the expectations from the reference monitor:

(a) It should be tamperproof

(b) It should always be invoked

(c) It should be small enough so that it can be independently tested

2. Security Models

An organization can take several approaches to implement its security model. Let us summarize theseapproaches.

- **No security**In this simplest case, the approach could be a decision to implement no security at all.
- **Security through obscurity**In this model, a system is secure simply because nobody knows about its existence and contents. This approach cannot work for too long, as there are many ways an attackercan come to know about it.

- **Host security**In this scheme, the security for each host is enforced individually. This is a very safe approach, but the trouble is that it cannot scale well. The complexity and diversity of modern sites/organizations makes the task even harder.
- **Network security** Host security is tough to achieve as organizations grow and become more diverse. In this technique, the focus is to control network access to various hosts and their services, ratherthan individual host security. This is a very efficient and scalable model

3. Security Management Practices

Good security management practices always talk of a security policy being in place. Putting a securitypolicy in place is actually quite tough.

A good security policy generally takes careoffour key aspects, as follows:

- Affordability Cost and effort in security implementation.
- Functionality Mechanism of providing security.
- Cultural issues Whether the policy gels well with people's expectations, working style and beliefs.
- Legality Whether the policy meets the legal requirements.

Once a security policy is in place, the following points should be ensured.

(a) Explanation of the policy to all concerned.
(b) Outline everybody's responsibilities.
(c) Use simple language in all communications.
(d) Establishment of accountability.
(e) Provision for exceptions and periodic reviews.

PRINCIPLES OF SECURITY

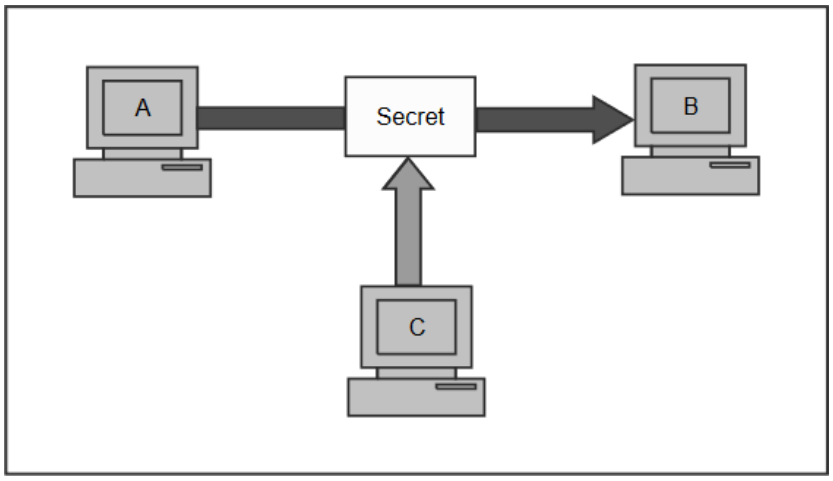
There are six principles

1. Confidentiality.
2. Authentication.
3. Integrity.
4. Non-repudiation.
5. Access control
6. Availability

.

1. **Confidentiality**

The principle of confidentiality specifies that only the sender and the intended recipient(s) should be ableto access the contents of a message. Confidentiality gets compromised if an unauthorized person is ableto access a message. Example of compromising the confidentiality of a message is shown in Fig. Here, the user of computer A sends a message to user of computer B.



Loss of confidentiality

Another user C gets access to this message, which is not desired andtherefore, defeats the purpose of confidentiality. Example of this could be a confidential email messagesent by A to B, which is accessed by C without the permission or knowledge of A and B. This type ofattack is called as interception.

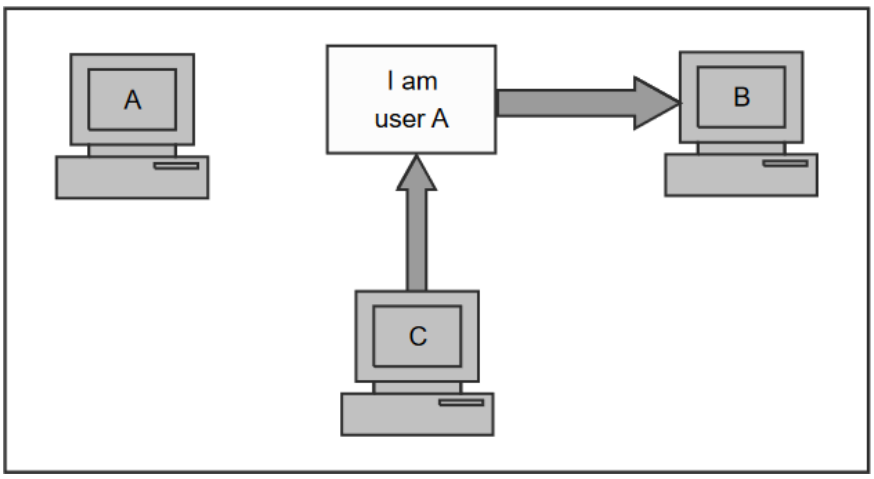
Interception causes loss of message confidentiality.

2. **Authentication**

Authentication mechanisms help establish proof of identities. The authentication process ensures thatthe origin of a electronic message or document is correctly identified.

suppose that user Csends an electronic document over the Internet to user B. However, the trouble is that user C had posedas user A when she sent this document to user B.

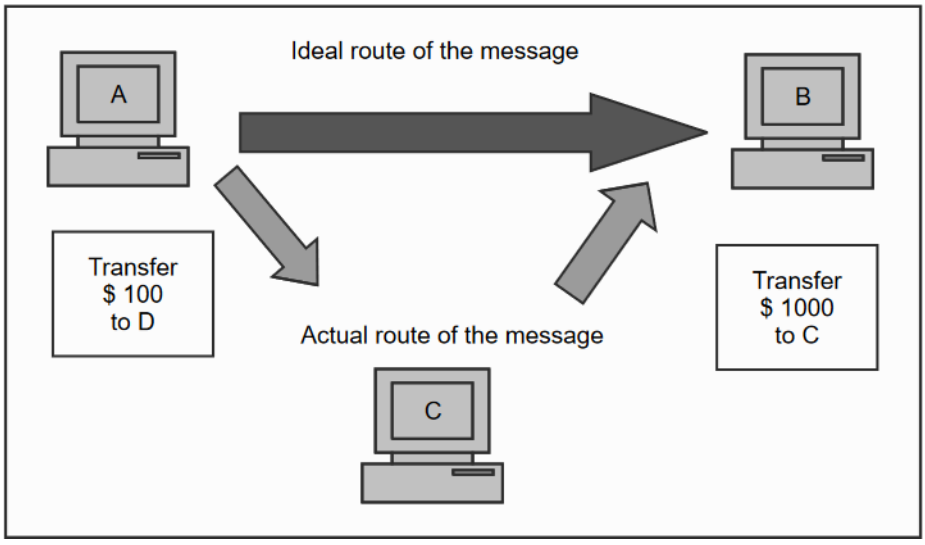
A real life example of this could be the case of a user C, posing asuser A, sending a funds transfer request (from A's account to C's account) to bank B. The bank mighthappily transfer the funds from A's account to C's account – after all, it would think that user A hasrequested for the funds transfer! This concept is shown in Fig.



Absence of authentication

3. Integrity

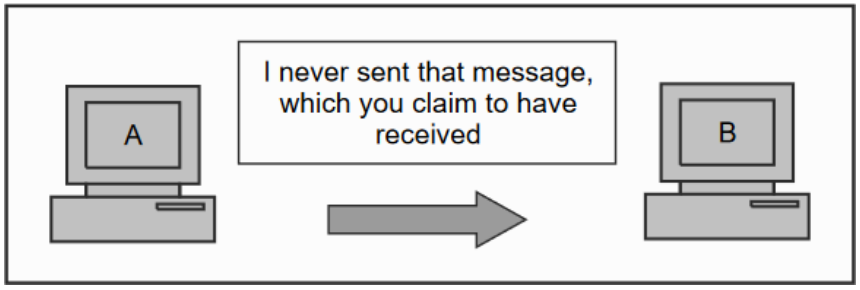
When the contents of a message are changed after the sender sends it, but before it reaches the intendedrecipient, we say that the integrity of the message is lost. For example, suppose you write a check forRs. 100 to pay for the goods bought from the US. However, when you see your next account statement, youare startled to see that the check resulted in a payment of Rs. 1000. This is the case for loss of messageintegrity. Conceptually, this is shown in Fig.



Loss of integrity

4. Non-repudiation

There are situations where a user sends a message and later on refuses that she had sent that message. Forinstance, user A could send a funds transfer request to bank B over the Internet. After the bank performsthe funds transfer as per A's instructions, A could claim that she never sent the funds transfer instructionto the bank! Thus, A repudiates or denies, her funds transfer instruction. The principle of non-repudiation defeats such possibilities of denying something, having done it. This is shown in Fig.



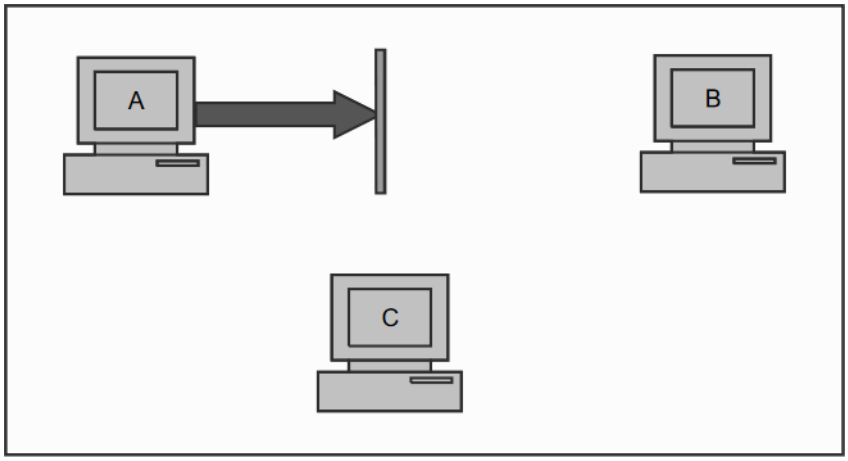
Establishing the non-repudiation

5. **Access Control**

The principle of access control determines who should be able to access what. For instance, we shouldbe able to specify that user A can view the records in a database, but cannot update them. However, userB might be allowed to make updates as well. An access control mechanism can be set up to ensure this.Access control is broadly related to two areas: role management and rule management. Rolemanagement concentrates on the user side (which user can do what), whereas rule management focuseson the resources side (which resource is accessible and under what circumstances).

6. **Availability**

The principle of availability states that resources (i.e. information) should be available to authorizedparties at all times. For example, due to the intentional actions of an unauthorized user C, an authorizeduser A may not be able to contact a server computer B, as shown in Fig.



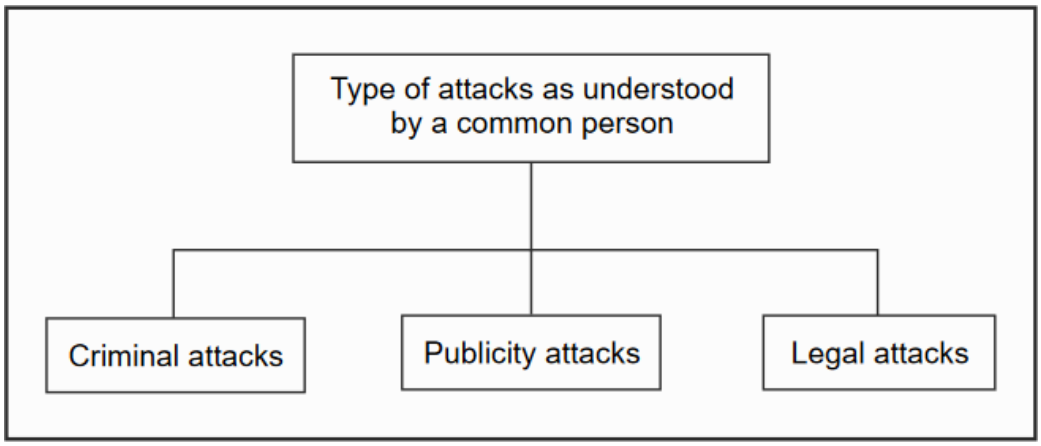
Attack on availability

TYPES OF SECURITY ATTACKS

We shall classify attacks with respect to two views: the common **person's view** and a **technologist's view**.

1. **General Attacks:**

   A General ViewFrom a common person's point of view, we can classify attacks into three categories, as shown inFig.



Classification of attacks in general terms

Criminal Attacks Criminal attacks are the simplest to understand. Here, the sole aim of the attackersis to maximize financial gain by attacking computer systems. The **following table gives** some of the criminal attacks.

Publicity Attacks Publicity attacks occur because the attackers want to see their names appear ontelevision news channels and newspapers. History suggests that these types of attackers are usually nothardcore criminals. They are people such as students in universities or employees in large organizations,
who seek publicity by adopting a novel approach of attacking computer systems.

Legal Attacks This form of attack is quite novel and unique. Here, the attacker tries to make the judgeor the jury doubtful about the security of a computer system. This works as follows. The attacker attacksthe computer system and the attacked party (say a bank or an organization) manages to take the attacker
to the court.

Attack	*Description*
Fraud	Modern fraud attacks concentrate on manipulating some aspects of electronic currency, credit cards, electronic stock certificates, checks, letters of credit, purchase orders, ATMs, etc.
Scams	Scams come in various forms, some of the most common ones being sale of services, auctions, multi-level marketing schemes, general merchandise and business opportunities, etc. People are enticed to send money in return of great profits, but end up losing their money. A very common example is the *Nigeria scam*, where an email from Nigeria (and other African countries) entices people to deposit money into a bank account with a promise of hefty gains. Whosoever gets caught in this scam loses money heavily.
Destruction	Some sort of grudge is the motive behind such attacks. For example, unhappy employees attack their own organization, whereas terrorists strike at much bigger levels. For example, in the year 2000, there was an attack against popular Internet sites such as Yahoo!, CNN, eBay, Buy.com, Amazon.com and e*Trade where authorized users of these sites failed to log in or access these sites.
Identity theft	This is best understood with a quote from Bruce Schneier: *Why steal from someone when you can just become that person?* In other words, an attacker does not steal anything from a legitimate user – he *becomes* that legitimate user! For example, it is much easier to manage to get the password of someone else's bank account or to actually be able to get a credit card on someone else's name. Then that privilege can be misused until it gets detected.
Intellectual property theft	Intellectual property theft ranges from stealing companies' trade secrets, databases, digital music and videos, electronic documents and books, software and so on.
Brand theft	It is quite easy to set up fake Web sites that look like real Web sites. How would a common user know if she is visiting the HDFC Bank site or an attacker's site? Innocent users end up providing their secrets and personal details on these fake sites to the attackers. The attackers use these details to then access the real site, causing an *identity theft*.

2. ATTACKS: A TECHNICAL VIEW

From the technical point of view, we can classify the types of attacks on computers and network systemsinto two categories for better understanding: (a) Theoretical concepts behind these attacks.

(b)Practical approaches used by the attackers.

(a) Theoretical Concepts

These attacks are generally classified into four categories.

- **Interception** –It means that an unauthorizedparty has gained access to a resource. The party can be a person, program or computer-basedsystem. Examples of interception are copying of data or programs and listening to network traffic.
- **Fabrication** –This involves creation of illegalobjects on a computer system. For example, the attacker may add fake records to a database.
    - **Modification** –For example the attacker may modifythe values in a database.

- **Interruption** – Here, the resource becomesunavailable, lost or unusable. Examples of interruption are causing problems to a hardware device,erasing program, data or operating system components.

These attacks are further grouped into twotypes:

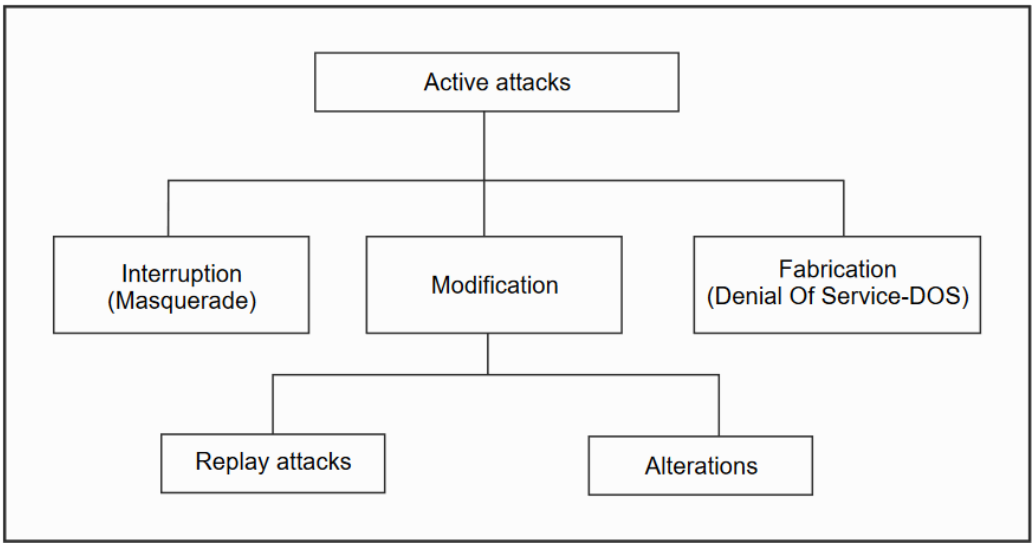
- Passive attacks.
- Active attacks.

Passive attacks: Passive attacks are those,wherein the attacker indulges in eavesdropping ormonitoring of data transmission. In other words,the attacker aims to obtain information that is intransit. The term passive indicates that the attackerdoes not attempt to perform any modifications tothe data.

Passive attacks do not involve any modifications to the contents of an original message.

Active attacks Unlike passive attacks, the active attacks are based on modification of the originalmessage in some manner or the creation of a false message. These attacks cannot be prevented easily.However, they can be detected with some effort and attempts can be made to recover from them. Theseattacks can be in the form of interruption, modification and fabrication.

In active attacks, the contents of the original message are modified in some way.



Active attacks

Masquerade is caused when an unauthorized entity pretends to be another entity.

Replay attack, a user captures a sequence of events or some data units and re-sends them.

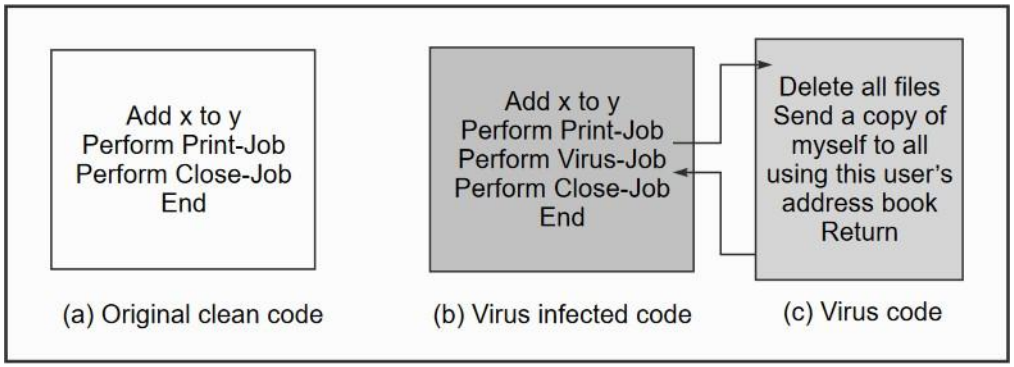
Alteration of messages involves some change to the original message. For instance, suppose user Asends an electronic message Transfer $1000 to D's account to bank B. User C might capture this andchange it to Transfer $10000 to C's account.

Denial Of Service (DOS) attacks make an attempt to prevent legitimate users from accessing someservices, which they are eligible for. For instance, an unauthorized user might send too many loginrequests to a server using random user ids one after the other in quick succession, so as to flood thenetwork and deny other legitimate users from using the network facilities.

3. PROGRAMS THAT ATTACK

Let us now discuss a few programs that attack computer systems to cause some damage or to createconfusion.

Virus One can launch an application-level attack or a network level attack using a virus. In simpleterms, a ***virus*** *is a piece of program code that attaches itself to legitimate program code and runs whenthe legitimate program runs. It can then infect other programs in that computer or programs that are inother computers but on the same network.*



Virus

A virus is a computer program that attaches itself to another legitimate program and causes damageto the computer system or to the network.

During its lifetime, a virus goes through four phases:

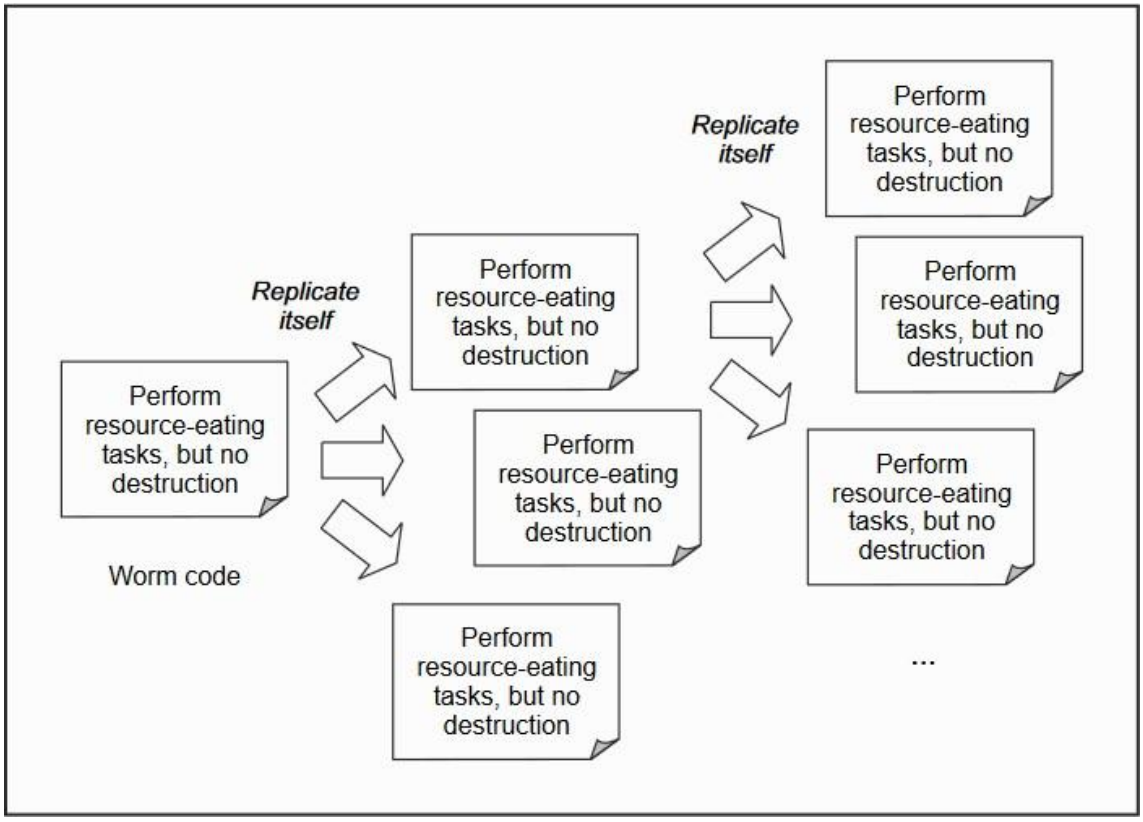
(a) **Dormant phase:** Here, the virus is idle. It gets activated based on certain action or event (e.g. theuser typing a certain key or certain date or time is reached, etc). This is an optional phase.

(b) **Propagation phase:** In this phase, a virus copies itself and each copy starts creating more copies ofself, thus propagating the virus.

(c) **Triggering phase:** A dormant virus moves into this phase when the action/event for which it waswaiting is initiated.

(d) **Execution phase:** This is the actual work of the virus, which could be harmless (display somemessage on the screen) or destructive (delete a file on the disk).

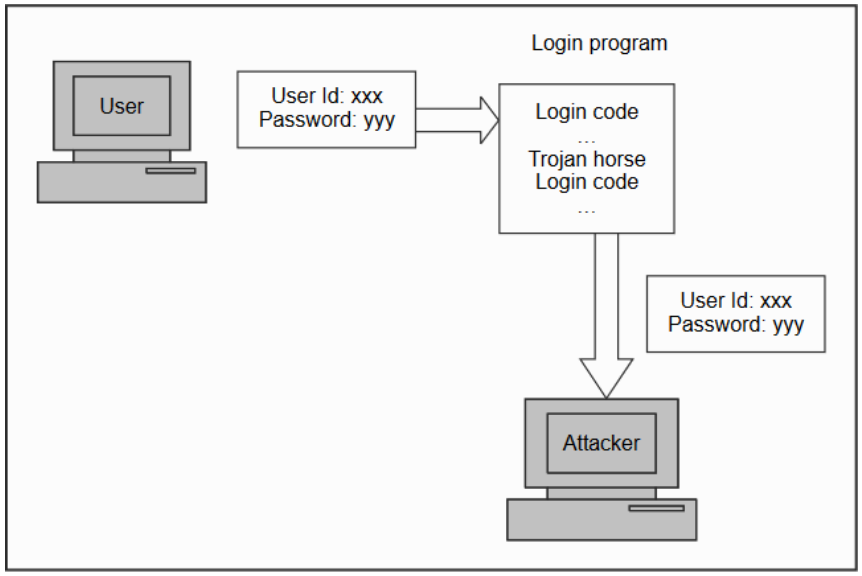
Worm Similar in concept to a virus, a worm is actually different in implementation. A virus modifiesa program (i.e. it attaches itself to the program under attack). A worm, however, does not modify aprogram. Instead, it replicates itself again and again.



Worm

Trojan HorseA Trojan horse is a hidden piece of code, like a virus. However, the purpose of aTrojan horse is different. Whereas the main purpose of a virus is to make some sort of modifications tothe target computer or network, a Trojan horse attempts to reveal confidential information to an attacker.

A Trojan horse allows an attacker to obtain some confidential information about a computeror a network.



Trojan horse

4. Specific Attacks

There are two specific attacks.

1. Sniffing
2. Spoofing

On the Internet, computers exchange messages with each other in the form of small blocks of data, called as packets. A packet, like a postal envelope contains the actual data to besent and the addressing information. Attackers target these packets, as they travel from the sourcecomputer to the destination computer over the Internet.

These attacks take two main forms:

(a) Packetsniffing

(b) Packet spoofing

(a) Packet sniffing: Packet sniffing is a passive attack on an on-going conversation. An attacker neednot hijack a conversation, but instead, can simply observe (i.e. sniff) packets as they pass by.

Clearly, to prevent an attacker from sniffing packets, the information that is passing needs to beprotected in some ways.

This can be done at two levels:

(i) The data that is traveling can be encoded in some ways

(ii) The transmission link itself can be encoded.

To read a packet, theattacker somehow needs to access it in the first place.

(B) Packet spoofing: In this technique, an attacker sends packets with a false source address. Whenthis happens, the receiver (i.e. the party who receives these packets containing false address)would inadvertently send replies back to this forged address (called as spoofed address) and not tothe attacker.

This can lead to three possible cases:

(i) The attacker can intercept the reply – If the attacker is between the destination and theforged source, the attacker can see the reply and use that information for hijacking attacks.

(ii) The attacker need not see the reply – If the attacker's intention was a Denial Of Service(DOS) attack, the attacker need not bother about the reply.

(iii) The attacker does not want the reply – The attacker could simply be angry with the host,so it may put that host's address as the forged source address and send the packet to thedestination.

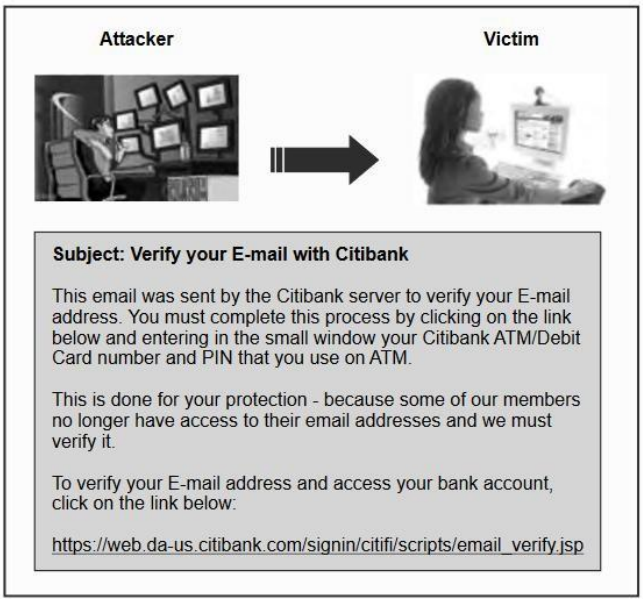
Phishing has become a big problem in recent times.

The attacker's module works as follows

- The attacker decides to create her own Web site, which looks very identical to a real Web site.Forexample, the attacker can clone Citibank's

Web site. The cloning is so clever that human eye willnot be able to distinguish between the real (Citibank's) and fake (attacker's) sites now

- The attacker can use many techniques to attack the bank's customers.
- When the customer (i.e. the victim) innocently clicks on the URL specified in the email, she istaken to the attacker's site and not the bank's original site.



- There, the customer is prompted to enterconfidential information, such as her password or PIN. Since the attacker's fake site looks exactlyike the original bank site, the customer provides this information.

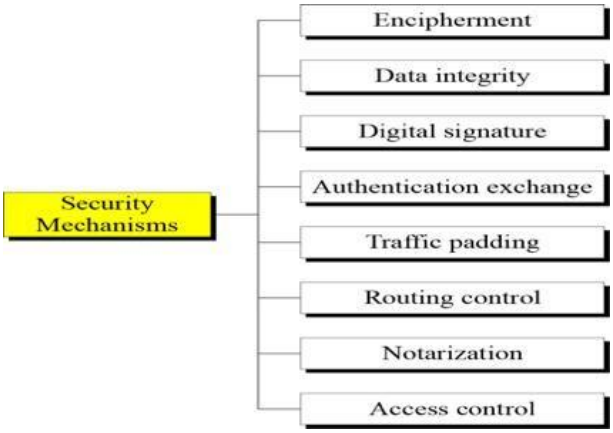
SECURITY SERVICES

- **Authentication:** assures recipient that the **message is from the source** that it **claims to** be from.
- **Access Control:** controls who can have **access to resource** under what **condition**
- **Availability:** available to authorized entities for 24/7.
- **Confidentiality:** information is not made available to unauthorized individual
- **Integrity:** assurance that the message is unaltered

- **Non-Repudiation:** protection against denial of sending or receiving in the communication

SECURITY MECHANISMS

Network Security is field in computer technology that deals with ensuring security of computer network infrastructure. As the network is very necessary for sharing of information whether it is at hardware level such as printer, scanner, or at software level.



1. **Encipherment :**

   This security mechanism deals with hiding and covering of data which helps data to become confidential. It is achieved by applying mathematical calculations or algorithms which reconstruct information into not readable form. It is achieved by two famous techniques named Cryptography and Encipherment. Level of data encryption is dependent on the algorithm used for encipherment.

2. **Access Control :**

   This mechanism is used to stop unattended access to data which you are sending. It can be achieved by various techniques such as applying passwords, using firewall, or just by adding PIN to data.

3. **Notarization :**

   This security mechanism involves use of trusted third party in communication. It acts as mediator between sender and receiver so that if any chance of conflict is reduced. This mediator keeps record of requests made by sender to receiver for later denied.

4. **Data Integrity :**

This security mechanism is used by appending value to data to which is created by data itself. It is similar to sending packet of information known to both sending and receiving parties and checked before and after data is received. When this packet or data which is appended is checked and is the same while sending and receiving data integrity is maintained.

5. **Authentication exchange :**

   This security mechanism deals with identity to be known in communication. This is achieved at the TCP/IP layer where two-way handshaking mechanism is used to ensure data is sent or not

6. **Bit stuffing :**

   This security mechanism is used to add some extra bits into data which is being transmitted. It helps data to be checked at the receiving end and is achieved by Even parity or Odd Parity.

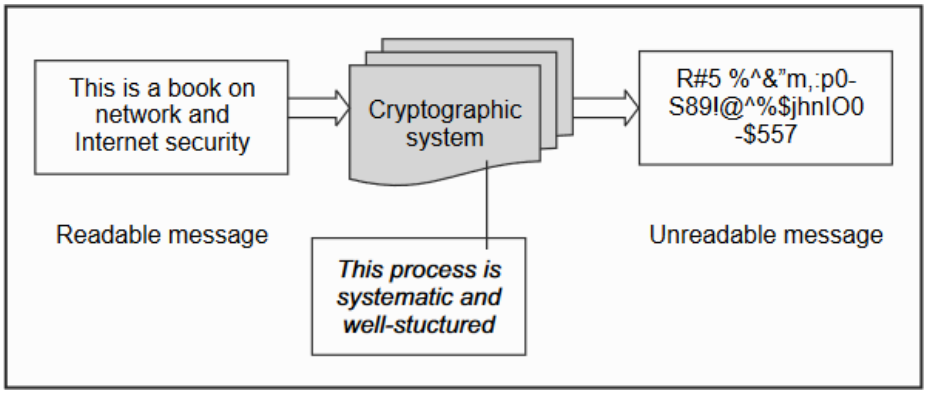
7. **Digital Signature :**

   This security mechanism is achieved by adding digital data that is not visible to eyes. It is form of electronic signature which is added by sender which is checked by receiver electronically. This mechanism is used to preserve data which is not more confidential but sender's identity is to be notified.

CRYPTOGRAPHY CONCEPTS AND TECHNIQUES

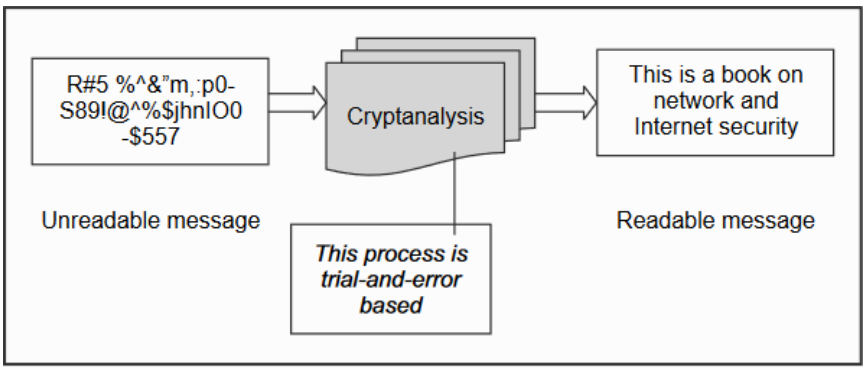
INTRODUCTION:

Cryptography is the art and science of achieving security by encoding messages to makethem non-readable.



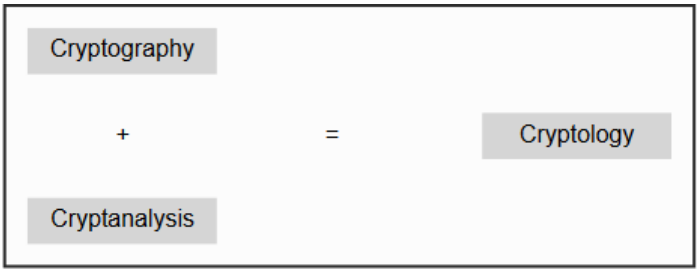
Cryptographic system

Cryptanalysis is the technique of decoding messages from a non-readable format back toreadable format without knowing how they were initially converted from readable format tonon-readable format.



Cryptanalysis

Cryptology is a combination of cryptography and cryptanalysis.



PLAIN TEXT AND CIPHER TEXT

Plain text or clear text is a message that can be understood by anybody knowing the languageas long as the message is not codified in any manner.

Clear text or plain text signifies a message that can be understood by the sender, the recipientand also by anyone else who gets an access to that message.

an example, they replace each alphabet with the alphabet that is actually three alphabetsdown the order. So, each A will be replaced by D, B will be replaced by E, C will be replaced by F andso on. To complete the cycle, each W will be replaced by Z, each X will be replaced by A, each Y will bereplaced by B and each Z will be replaced by C. We can summarize this scheme as shown in Fig.Thefirst row shows the original alphabets and the second row shows what each original alphabet will bereplaced with.

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

A scheme for codifying messages by replacing each alphabet with an alphabet threeplaces down the line

ANNAMACHARYA can be coded as ***DQQDPDFKDUBD***

A	N	N	A	M	A	C	H	A	R	Y	A
D	Q	Q	D	P	D	F	K	D	U	B	D

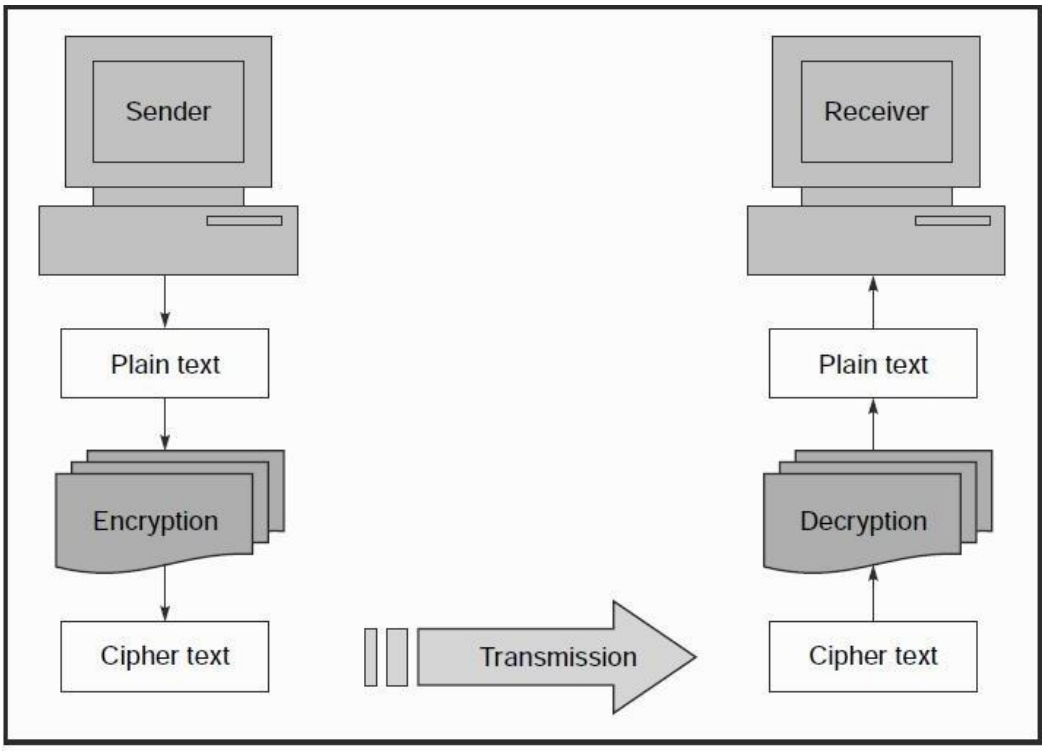
Each alphabet in the original message can be replaced by another to hidethe original contents of the message. The codified message is called as **cipher text.** Cipher means a codeor a secret message.

When a plain text message is codified using any suitable scheme, the resulting message iscalled as cipher text.

SUBSTITUTION TECHNIQUES

1. CAESAR CIPHER

This was first proposed by **Julius Caesar** and is termed as **Caesar Cipher.** Caesar Cipher is a special case of substitution techniques wherein eachalphabet in a message is replaced by an alphabet three places down the line. For instance, using theCaesar Cipher, the plain text ATUL will become cipher text DWXO.



Elements of cryptographic operations

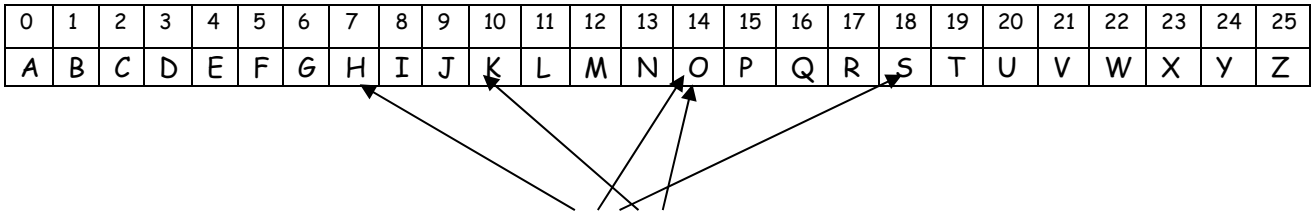
In the **substitution cipher technique**, the characters of a plain text message are replaced byother *characters, numbers or symbols.*

An attack on a cipher text message,wherein the attacker attempts to use all possible permutations and combinations, is called as a **Bruteforceattack**. The process of trying to break any cipher text message to obtain the original plain textmessage itself is called as **Cryptanalysis** and the person attempting a cryptanalysis is called as a**cryptanalyst**.

MONO-ALPHABETIC CIPHER

Mono-alphabetic ciphers pose a difficult problem for a cryptanalyst because it can be verydifficult to crack thanks to the high number of possible permutations and combinations.

Use random substitution. This means that in a given plain text message**, each A can be replaced by any other alphabet (B through Z), each B can also be replaced by any other random alphabet (A or C through Z) and so on.** The crucial difference being, there is no relation between thereplacement of B and replacement of A. That is, if we have decided to replace each A with D, we neednot necessarily replace each B with E – we can replace each B with any other character!



the Cipher text is : *HOSKO*

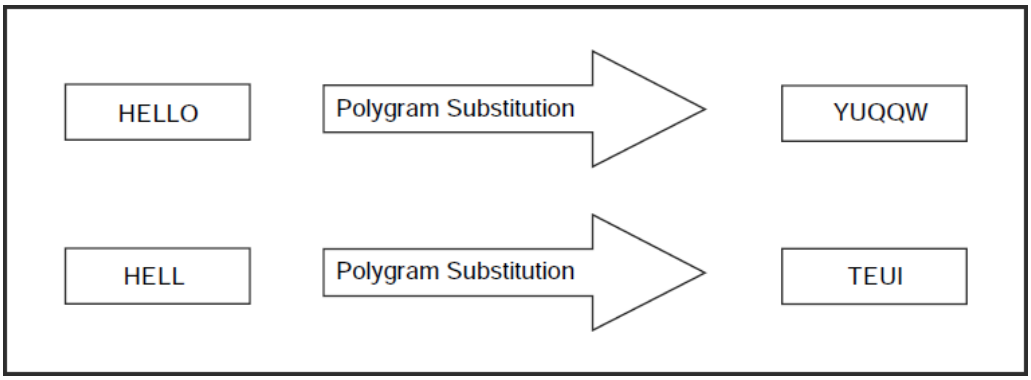
Homophonic Substitution Cipher

Homophonic Substitution Cipher also involves substitution of one plain text character with acipher text character at a time, however the cipher text character can be any one of thechosen set.

The Homophonic Substitution Cipher is very similar to Mono-alphabetic Cipher. Like a plainsubstitution cipher technique, we replace one alphabet with another in this scheme. However, thedifference between the two techniques is that whereas the replacement alphabet set in case of the simplesubstitution techniques is fixed (e.g. replace A with D, B with E, etc.), in the case of HomophonicSubstitution Cipher**, one plain text alphabet can map to more than one cipher text alphabet**. For instance,A can be replaced by D, H, P, R; B can be replaced by E, I, Q, S, etc.

Polygram Substitution Cipher

In Polygram Substitution Cipher technique, **rather than replacing one plain text alphabet with one cipher text alphabet at a time, a block of alphabets** is replaced with another block. For instance, HELLOcould be replaced by YUQQW, but HELL could be replaced by a totally different cipher text block TEUI, as shown in Fig.



Polyalphabetic Substitution Cipher

A poly-alphabetic cipher is any cipher based on substitution, using several substitution alphabets. In polyalphabetic substitution ciphers, the plaintext letters are enciphered differently based upon their installation in the text. Rather than being a one-to-one correspondence, there is a one-to-many relationship between each letter and its substitutes.

For example, 'a' can be enciphered as 'd' in the starting of the text, but as 'n' at the middle. The polyalphabetic ciphers have the benefit of hiding the letter frequency of the basic language. Therefore attacker cannot use individual letter frequency static to divide the ciphertext.

As the name polyalphabetic recommend this is achieved by **using multiple keys rather than only one key**. This implies that the key should be a stream of subkeys, in which each subkey depends somehow on the position of the plaintext character that needs subkey for encipherment.

Vigenere cipher is one of the simplest and popular algorithms in polyalphabetic cipher. In this approach, the alphabetic text is encrypted using a sequence of **multiple Caesar ciphers** based on the letters of a keyword.
The Vigenère cipher includes several simple substitution ciphers in sequence with several shift values. In this cipher, the keyword is repeated just before it connects with the duration of the plaintext.

Encryption Process:

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

In this process sum of i^{th} position of plain text and i^{th} position of key will be added and applied modulus 26 on the result , the generated positional value will be considered as Cipher text.

Decryption Process

$P_i=(C_i-K_i) \bmod 26$

In this process sum of i^{th} position of Cipher text and i^{th} position of key will be subtracted and applied modulus 26 on the result , the generated positional value will be considered as Plain text.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
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

Key: samba

Plain text: hello students how are you

Cipher text:

Key	s	a	m	B	a	s	a	m	b	a	s	a	m	b	a	s	a	m	b	a	s	A
PT	h	e	l	L	o	s	t	u	d	e	n	t	s	h	o	w	a	r	e	y	o	U
CT	z	e	x	M	o	k	t	g	E	e	f	t	e	i	o	o	a	c	f	y	g	U

Apply Encryption process to generate cipher text

That is 's' position is 18 and 'h' position is 7 so now

$C_1=(p_1+k_1) \bmod 26$

$= (18+7) \text{mpd} 26$

$=(25) \bmod 26$

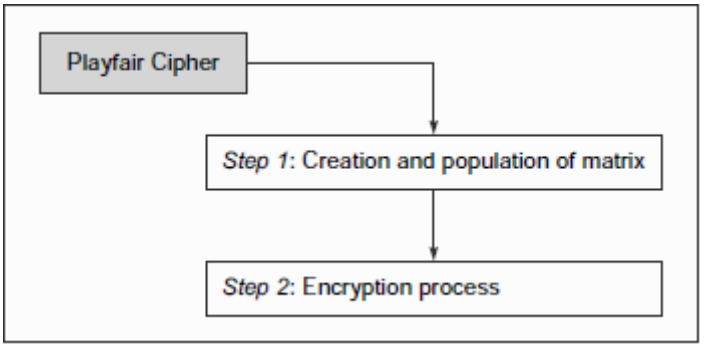
=25 (which is equalent to 'z')

Like wise generate the table accordingly.

Playfair Cipher:

The Playfair Cipher, also called as **Playfair Square**, is a cryptographic technique that is used formanual encryption of data.

The Playfair encryption scheme uses two main processes, as shown in Fig



Playfair cipher steps

Step 1: Creation Population of Matrix and The Playfair Cipher makes use of a 5 x 5 matrix(table), which is used to store a keyword or phrase that becomes the key for encryption and decryption.
The way this is entered into the 5 x 5 matrix is based on some simple rules, as shown below

1. Enter the keyword in the matrix row-wise: left-to-right, and then top-to-bottom.
2. Drop duplicate letters.
3. Fill the remaining spaces in the matrix with the rest of the English alphabets (A-Z)that were not a part of our keyword. While doing so, combine I and J in the same cell of the table. In otherwords, if I or J is a part of the keyword, disregard both I and J while filling the remaining slots.

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

PlainText: "instruments"
After Split: 'in' 'st' 'ru' 'me' 'nt' 'sz'

1. Pair cannot be made with same letter. Break the letter in single and add a bogus letter to the previous letter.
Plain Text: "hello"
After Split: 'he' 'lx' 'lo'
Here **'x'** is the bogus letter.

2. If the letter is standing alone in the process of pairing, then add an extra bogus letter with the alone letter
Plain Text: "helloe"
AfterSplit: 'he' 'lx' 'lo' 'ez'

Here '**z**' is the bogus letter.

Rules for Encryption:

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

Diagraph: "me"
Encrypted Text: cl
Encryption:
m -> c
e -> l

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

If both the letters are in the same row: Take the letter to the right of each one (going back to the leftmost if at the rightmost position).
Diagraph: "ST"
Encrypted Text: TL
Encryption:
S -> T
T -> L

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

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

Diagraph: "nt"
Encrypted Text:rq
Encryption:
n -> r
t -> q

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

Plain Text: "instrumentsz"

Encrypted Text: gatlmzclrqtx

Encryption:

i -> g

n -> a

s -> t

t -> l

r -> m

u -> z

m -> c

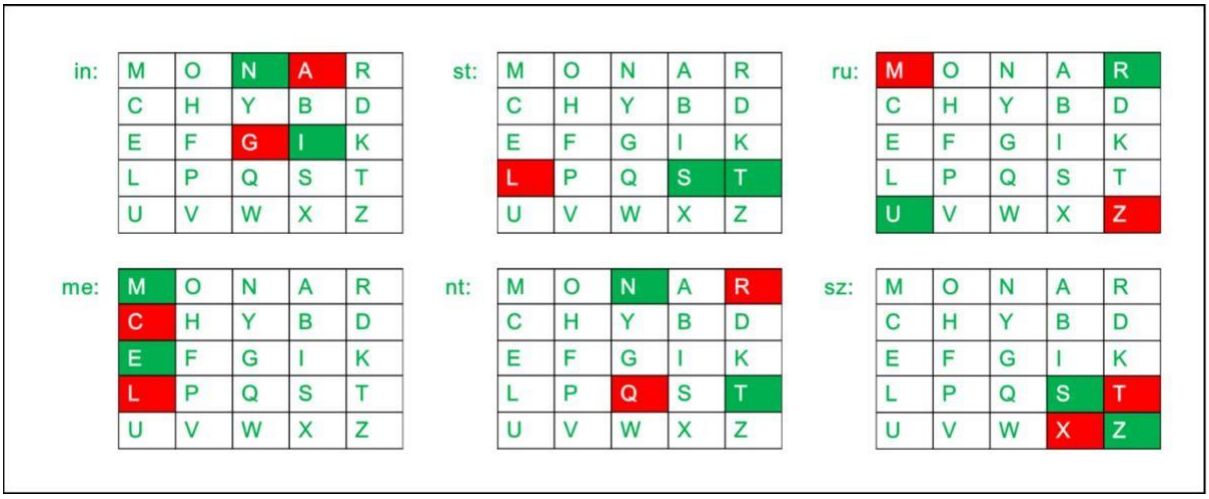
e -> l

n -> r

t -> q

s -> t

z -> x

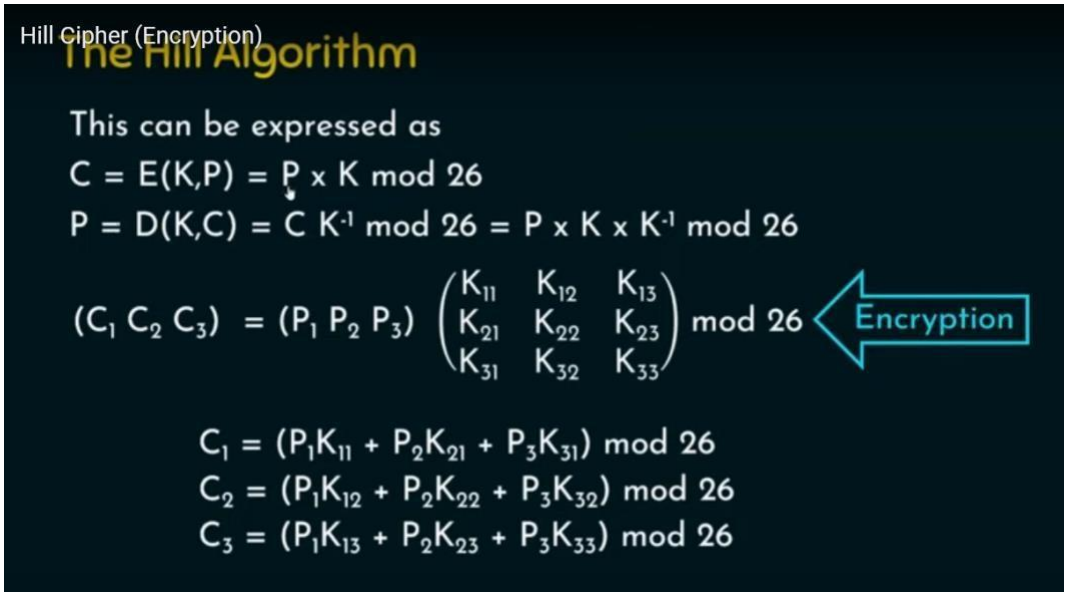


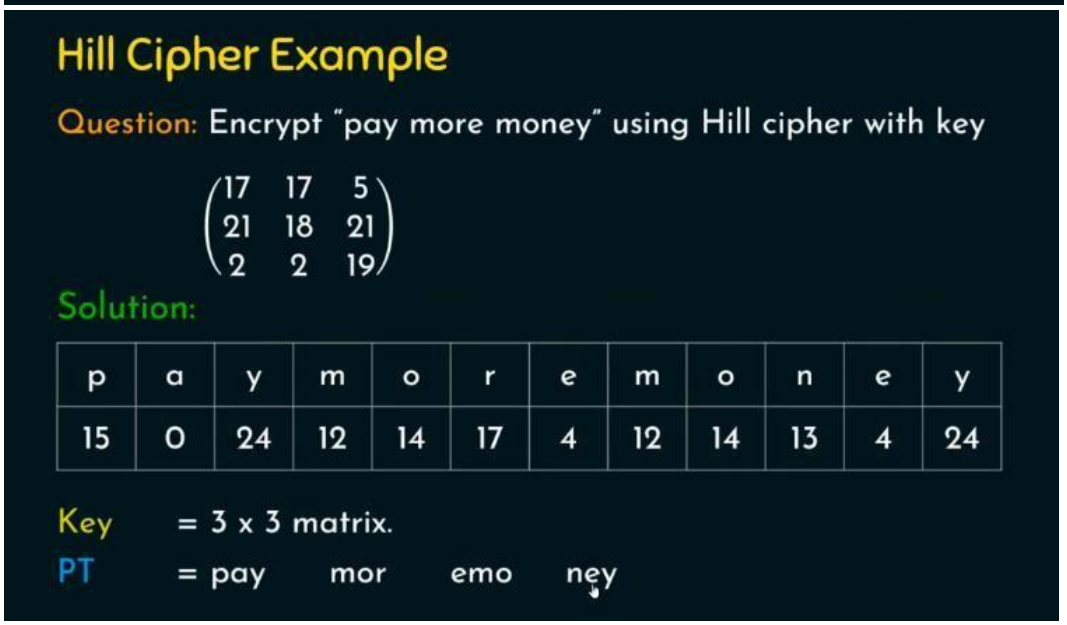
Hill Cipher

Hill cipher is a polygraphic substitution cipher based on linear algebra.Each letter is represented by a number modulo 26. Often the simple scheme A = 0, B= 1, …, Z = 25 is used, but this is not an essential feature of the cipher. To encrypt a message, each block of n letters (considered as an n-component vector) is multiplied by an invertible n × n matrix, against modulus 26. To decrypt the

message, each block is multiplied by the inverse of the matrix used for encryption.

The matrix used for encryption is the cipher key, and it should be chosen randomly from the set of invertible n × n matrices (modulo 26).





Hill Cipher Example

Encrypting: pay

$$(C_1\ C_2\ C_3) = (P_1\ P_2\ P_3) \begin{pmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{pmatrix} \bmod 26$$

$$(C_1\ C_2\ C_3) = (15\ 0\ 24) \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \bmod 26$$

$$= (15x17+0x21+24x2 \quad 15x17+0x18+24x2 \quad 15x5+0x21+24x19) \bmod 26$$

$$= (303\ 303\ 531) \bmod 26$$

$$= (17\ 17\ 11)$$

$$= (R\ R\ L)$$

Hill Cipher Example

Encrypting: mor

$$(C_1\ C_2\ C_3) = (P_1\ P_2\ P_3) \begin{pmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{pmatrix} \bmod 26$$

$$(C_1\ C_2\ C_3) = (12\ 14\ 17) \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \bmod 26$$

$$= (12x17+14x21+17x2 \quad 12x17+14x18+17x2 \quad 12x5+14x21+17x19) \bmod 26$$

$$= (532\ 490\ 677) \bmod 26$$

$$= (12\ 22\ 1)$$

$$= (M\ W\ B)$$

NESO ACADEMY

Hill Cipher Example

Encrypting: emo

$$(C_1\ C_2\ C_3) = (P_1\ P_2\ P_3) \begin{pmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{pmatrix} \bmod 26$$

$$(C_1\ C_2\ C_3) = (4\ 12\ 14) \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \bmod 26$$

$$= (4x17+12x21+14x2 \quad 4x17+12x18+14x2 \quad 4x5+12x21+14x19) \bmod 26$$

$$= (348\ 312\ 538) \bmod 26$$

$$= (10\ 0\ 18)$$

$$= (K\ A\ S)$$

NESO ACADEMY

Hill Cipher Example

Encrypting: ney

$$(C_1\ C_2\ C_3) = (P_1\ P_2\ P_3) \begin{pmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{pmatrix} \bmod 26$$

$$(C_1\ C_2\ C_3) = (13\ 4\ 24) \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \bmod 26$$

$$= (13x17+4x21+24x2 \quad 13x17+4x18+24x2 \quad 13x5+4x21+24x19) \bmod 26$$

$$= (348\ 312\ 538) \bmod 26$$

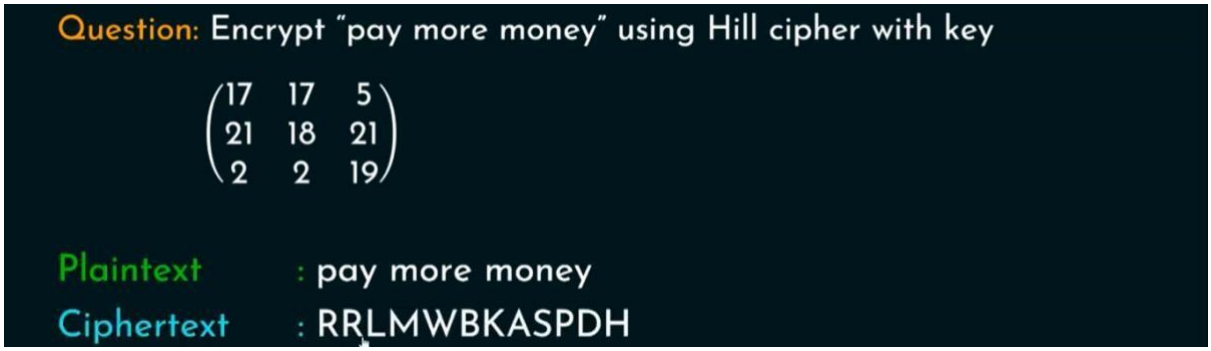
$$= (15\ 3\ 7)$$

$$= (P\ D\ H)$$

NESO ACADEMY

Hill Cipher Example

PT	p	a	y	m	o	r	e	m	o	n	e	y
CT	R	R	L	M	W	B	K	A	S	P	D	H



TRANSPOSITION TECHNIQUES

Transposition techniques differ from substitution techniques in the way that they do not simplyreplace one alphabet with another: they also perform some permutation over the plain text alphabets.

Rail Fence Technique

Rail fence technique involves writing plain text as sequence of diagonals and then reading itrow-by-row to produce cipher text.
Suppose that we have a plain textmessage Come home tomorrow. How would we transform that into a cipher text message using the RailFence Technique? This is shown in Fig.

1. Write down the plain text message as a sequence of diagonals.
2. Read the plain text written in Step 1 as a sequence of rows.
3. here depth=2

Original plain text message: ***Come home tomorrow***

1. After we arrange the plain text message as a sequence of diagonals, it would look as follows (write the first character on the first line i.e. *C*, then second character on the second line, i.e. *o*, then the third character on the first line, i.e. *m*, then the fourth character on the second line, i.e. *e*, and so on). This creates a zigzag sequence, as shown below.

C m h m t m r o
o e o e o o r w

2. Now read the text row-by-row, and write it sequentially. Thus, we have: ***Cmhmtmrooeoeoorw*** as the cipher text.

Example of rail technique

Simple Columnar Transposition Technique

Basic Technique Variations of the basic transposition technique such as Rail Fence Technique exist.Such a scheme call as Simple Columnar Transposition Technique.

- Write the plain text message row-by-row in a rectangle of a pre-defined size.
- Read the message column-by-column. However, it need not be in the order of columns 1,2, 3 etc. It can be any random order such as 2, 3, 1, etc.
- The message thus obtained is the cipher text message.

The Simple Columnar Transposition Technique simply arranges the plain text as a sequenceof rows of a rectangle that are read in columns randomly.

Original plain text message: ***Come home tomorrow***

1. Let us consider a rectangle with six columns. Therefore, when we write the message in the rectangle row-by-row (suppressing spaces), it would look as follows:

Column 1	Column 2	Column 3	Column 4	Column 5	Column 6
C	o	m	e	h	o
m	e	t	o	m	o
r	r	o	w		

2. Now, let us decide the order of columns as some random order, say 4, 6, 1, 2, 5 and 3. Then read the text in the order of these columns.
3. The cipher text thus obtained would be ***eowoocmroerhmmto***.

Example of simple columnar technique

A MODEL FOR NETWORK SECURITY

A message is to be transferred from one party to another across some sort of internet. Thetwo parties, who are the principals in this transaction, must cooperate for the exchange to takeplace. A logical information channel is established by defining a route through the internetfrom 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 ordesirable to protect the information transmission from an opponent who may present a threatto confidentiality, authenticity, and so on.

All the techniques for providing security have twocomponents:

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 bythe opponent, and the addition of a code based on the contents of the message, whichcan be used to verify the identity of the sender Some secret

information shared by thetwo principals and, it is hoped, unknown to the opponent. An example is an encryptionkey used in conjunction with the transformation to scramble the message beforetransmission and unscramble it on reception.

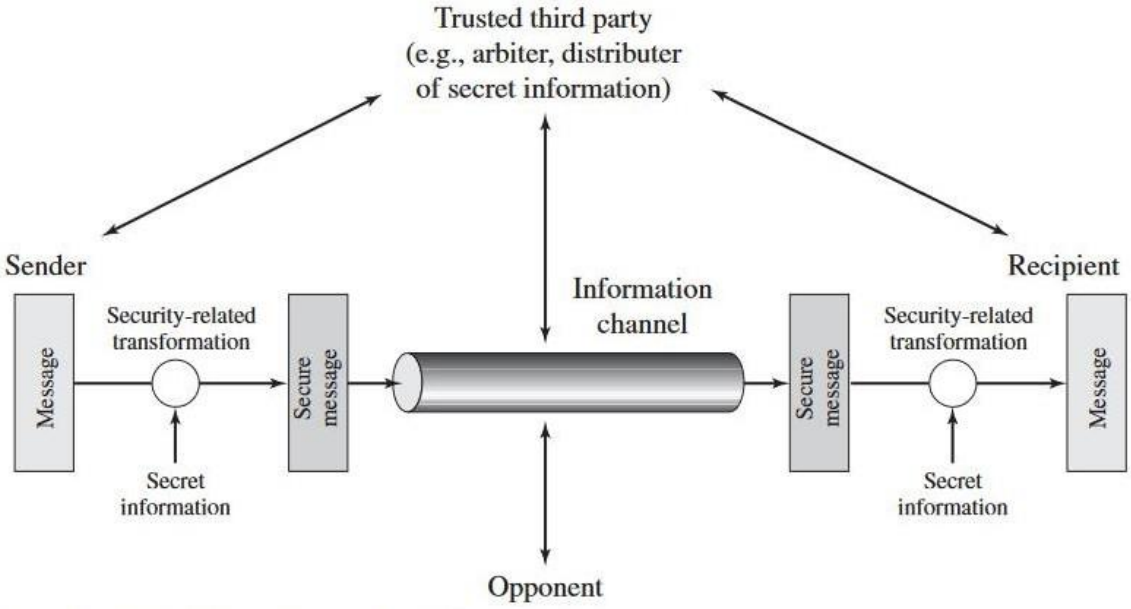


Figure 1.4 Model for Network Security

The general model shows that there are four basic tasks in designing a particular securityservice:

1. Design an algorithm for performing the security-related transformation. Thealgorithm 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 securityalgorithm and the secret information to achieve a particular security service

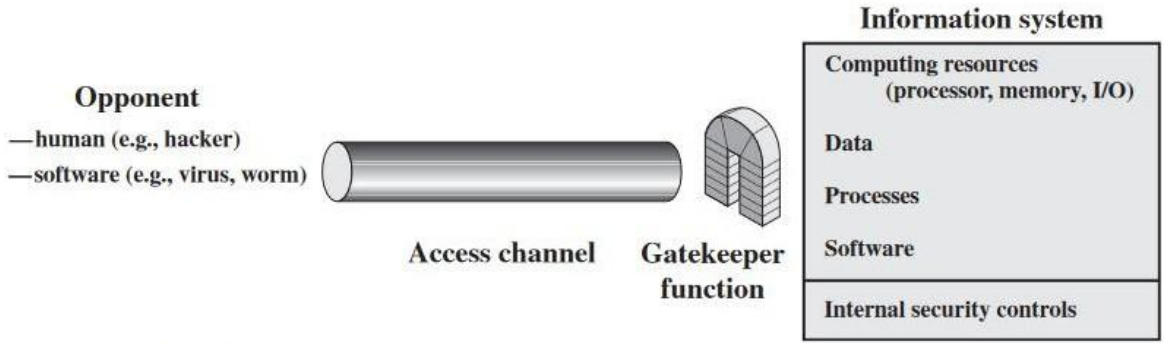


Figure 1.5 Network Access Security Model

A general model is illustrated by the above Figure 1.6, which reflects a concern forprotecting an information system from unwanted access. Most readers are

familiar with theconcerns caused by the existence of hackers, who attempt to penetrate systems that can beaccessed over a network. The hacker can be someone who, with no malign intent, simply getssatisfaction from breaking and entering a computer system. Or, the intruder can be adisgruntled employee who wishes to do damage, or a criminal who seeks to exploit computerassets for financial gain.

ENCRYPTION AND DECRYPTION

The process of encoding plain text messagesinto cipher text messages is called as **encryption**.

The process of transforming cipher textmessages back to plain text messages is called as**decryption**.

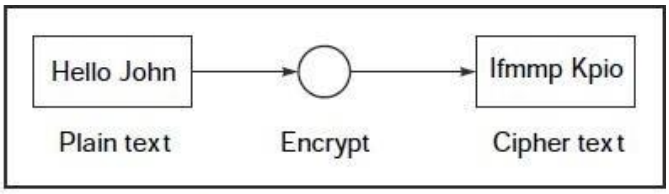


Fig. 2.40 *Encryption*

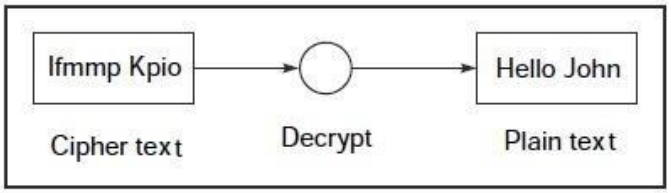
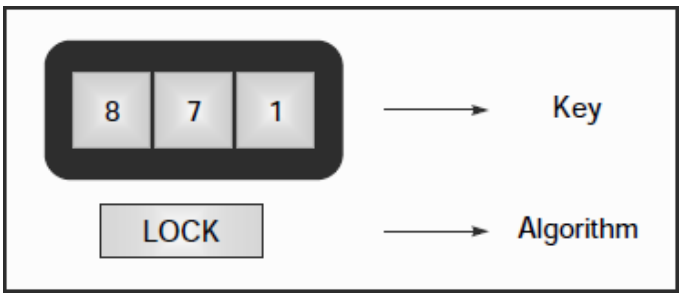


Fig. 2.41 *Decryption*

Every encryption and decryption process has two aspects: the **algorithm** and the **key** used forencryption and decryption.

let us take the example ofa combination lock, which we use in real life. We need to remember the combination (which is a number, suchas 871) needed to open up the lock. The facts that it is acombination lock and how to open it (algorithm) arepieces of public knowledge. However, the actual valueof the key required for opening a specific lock (key),which is 871 in this case, is kept secret. The idea isillustrated in Fig

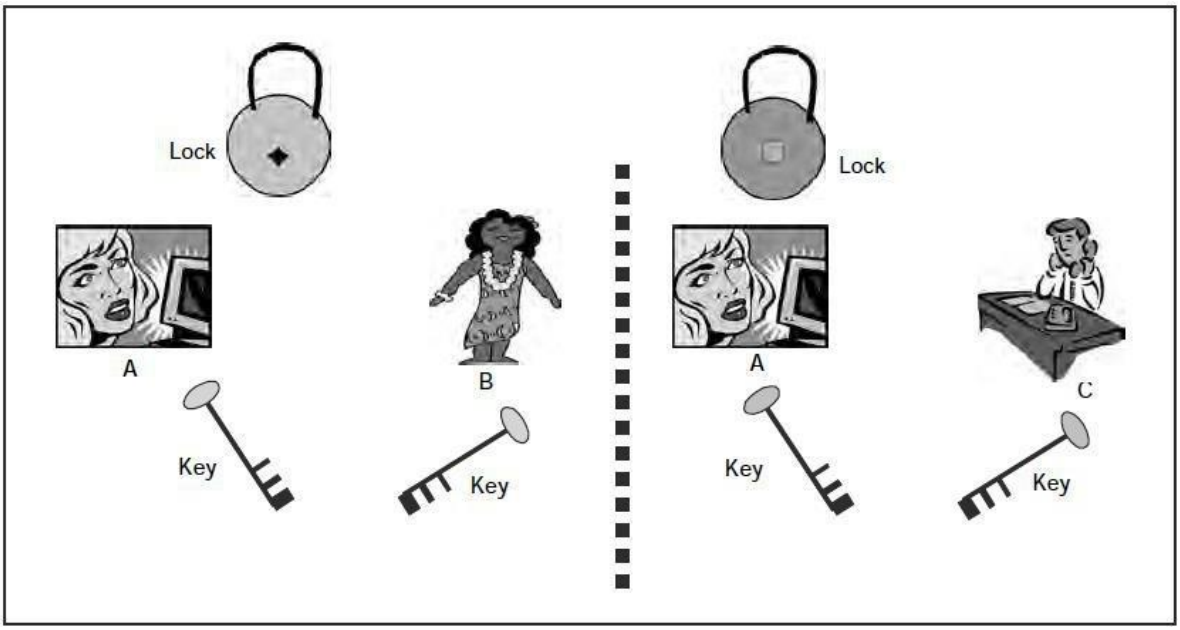


Broadly, there are two cryptographic mechanisms, depending on **what keys are used.** If the same keyis used for encryption and decryption, we call the mechanism as **Symmetric Key Cryptography**.However, if two different keys are used in a cryptographic mechanism, wherein one key is used forencryption and another, different key is used for decryption; we call the mechanism as **Asymmetric KeyCryptography**.

Symmetric and Asymmetric Key Cryptography

The sender andthe receiver will use the same key to lock and unlock, this is called as symmetric key operation (whenused in the context of cryptography, this operation is called as **symmetric key cryptography**).
Person A wants to send a highly confidential letter to another person B. A and B both residein the same city, but are separated by a few miles and for some reason, cannot meet eachother.
With the symmetric key cryptography A can send securely to the B.

Let us now imagine that not only A and B but also thousands of people want to send such confidentialletters securely to each other. What would happen if they decide to go for symmetric key operation? Ifwe examine this approach more closely, we can see that it has one big drawback if the number of peoplethat want to avail of its services is very large.



Use of separate locks and keys per communication pair
we have the following situation:

- When A wanted to communicate only with B, we needed one lock-and-key pair (A-B).
- When A wants to communicate with B and C, we need two lock-and-key pairs (A-B and A-C).

Thus, we need one lock-and-key pair per person with whom A wants to communicate. If B alsowants to communicate with C, we have B-C as the third communicating pair, requiring its ownlock-and-key pair. Thus, we would need three lock-and-key pairs to serve the needs of threecommunicating pairs.

Therefore, can we see that, in general, for n persons, the number of lock-and-key pairs is$\frac{n*(n-1)}{2}$

Parties involved	Number of lock-and-key pairs required
2 (A, B)	1 (A-B)
3 (A, B, C)	3 (A-B, A-C, B-C)
4 (A, B, C, D)	6 (A-B, A-C, A-D, B-C, B-D, C-D)
5 (A, B, C, D, E)	10 (A-B, A-C, A-D, A-E, B-C, B-D, B-E, C-D, C-E, D-E)

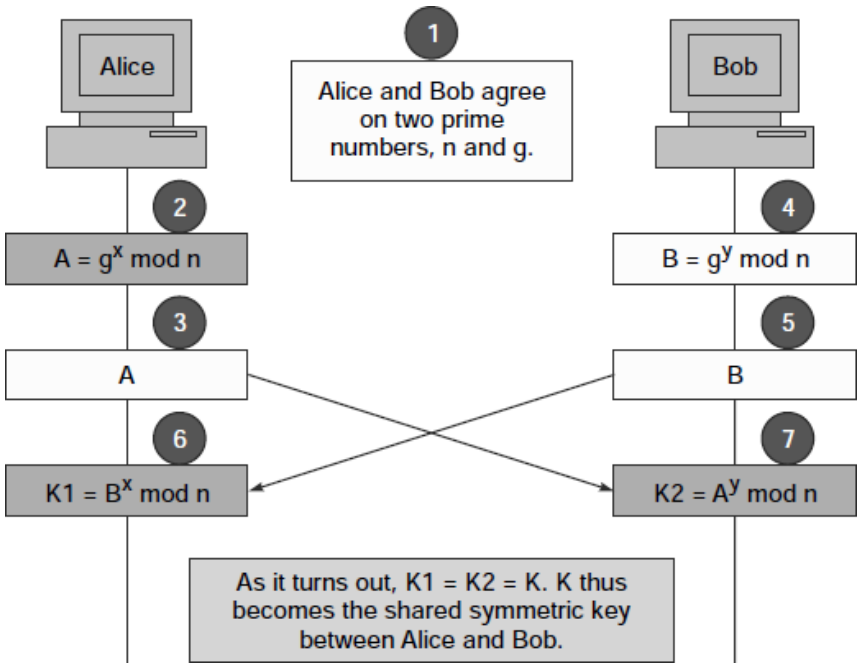
Diffie–Hellman Key Exchange/Agreement Algorithm

In this scheme the two parties, who want to communicatesecurely, can agree on a **symmetric key** using this technique. This key can then be used for encryption/ decryption. However, we must note that Diffie–Hellman key exchange algorithm can be used only forkey agreement, but not for encryption or decryption of messages. Once both the parties agree on the keyto be used, they need to use other symmetric key encryption algorithms for actual encryption or decryption of messages.

Description of the Algorithm

1. Firstly, Alice and Bob agree on two large prime numbers, n and g. These two integers need not be kept secret. Alice and Bob can use an insecure channel to agree on them.
2. Alice chooses another large random number x, and calculates A such that:
   $A = g^x \bmod n$
3. Alice sends the number A to Bob.
4. Bob independently chooses another large random integer y and calculates B such that:
   $B = g^y \bmod n$
5. Bob sends the number B to Alice.
6. A now computes the secret key K1 as follows:
   $K1 = B^x \bmod n$
7. B now computes the secret key K2 as follows:
   $K2 = A^y \bmod n$

Diffie–Hellman key exchange algorithm



Example of the Algorithm

1. Firstly, Alice and Bob agree on two large prime numbers, n and g. These two integers need not be kept secret. Alice and Bob can use an insecure channel to agree on them.

   Let $n = 11, g = 7$.

2. Alice chooses another large random number x, and calculates A such that:
   $A = g^x \bmod n$

   Let $x = 3$. Then, we have, $A = 7^3 \bmod 11 = 343 \bmod 11 = 2$.

3. Alice sends the number A to Bob.

   Alice sends 2 to Bob.

4. Bob independently chooses another large random integer y and calculates B such that:
   $B = g^y \bmod n$

   Let $y = 6$. Then, we have, $B = 7^6 \bmod 11 = 117649 \bmod 11 = 4$.

5. Bob sends the number B to Alice.

   Bob sends 4 to Alice.

6. A now computes the secret key K1 as follows:
   $K1 = B^x \bmod n$

   We have, $K1 = 4^3 \bmod 11 = 64 \bmod 11 = 9$.

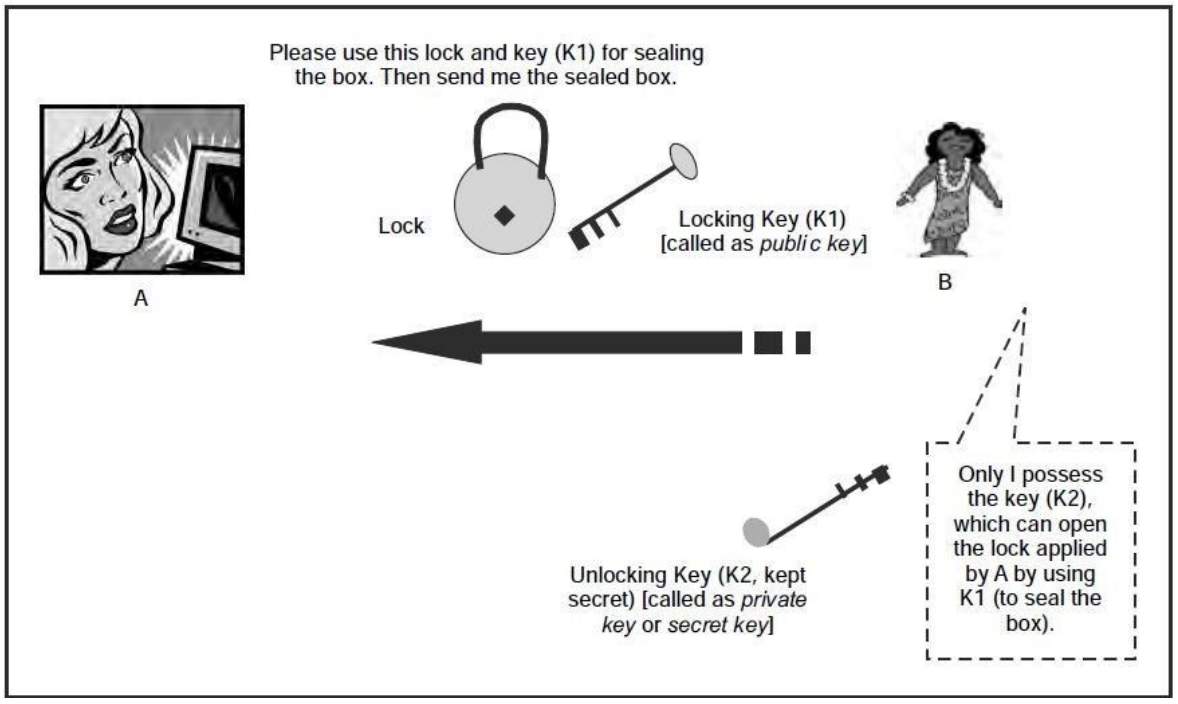
7. B now computes the secret key K2 as follows:
   $K2 = A^y \bmod n$

   We have, $K2 = 2^6 \bmod 11 = 64 \bmod 11 = 9$.

Asymmetric Key Operation

In this scheme, (Alice) A and (Bob) B do not have to jointly approach (Tom) T for a lock-and-key pair. Instead, B aloneapproaches T, obtains a lock and a key (K1) that can seal the lock and sends the lock and key K1 to A. Btells A that A can use that lock and key to seal the box before sending the sealed box to B.

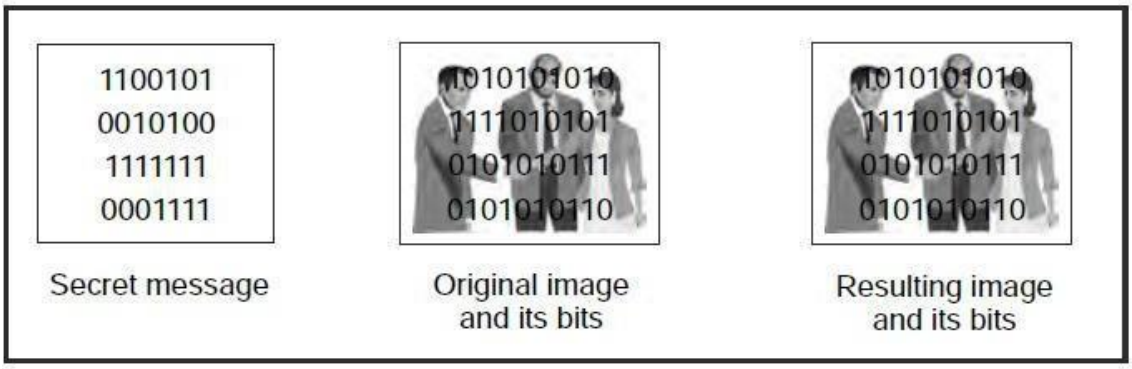
An interesting property of this scheme is that B possesses a different but related key (K2), which isobtained by B from T along with the lock and key K1, only which can open the lock. It is guaranteed thatno other key and of course, including the one used by A (i.e. K1) for locking, can open the lock. Sinceone key (K1) is used for locking and another, different key (K2) is used for unlocking; we will call thisscheme as asymmetric key operation. Also, T is clearly defined here as a **trusted third party**. T iscertified as a highly trustworthy and efficient agency by the government.



STEGANOGRAPHY

Steganography is a technique that facilitates hiding of a message that is to be kept secret inside othermessages.

The sender usedmethods such as invisible ink, tiny pin punctures on specific characters, minute variations betweenhandwritten characters, pencil marks on handwritten characters, etc.

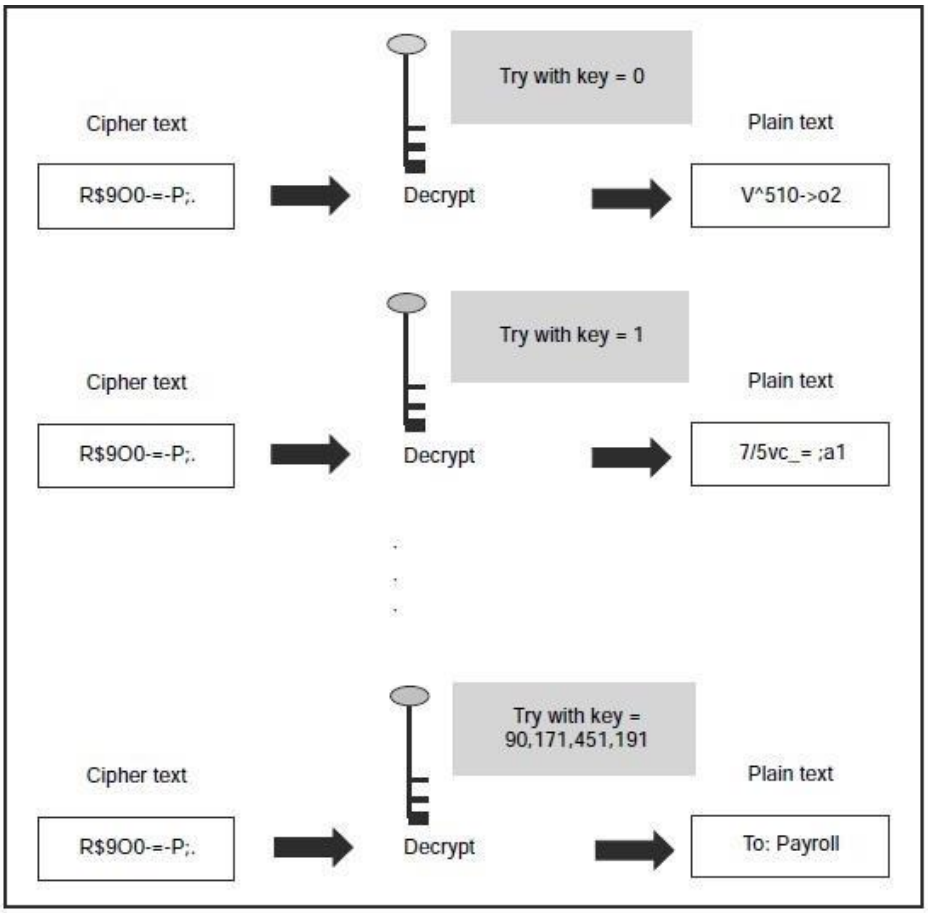


KEY RANGE AND KEY SIZE

The cryptanalystis armed with the following information:

- The encryption/decryption algorithm
- The encrypted message
- Knowledge about the key size (e.g. the value of the key is a number between 0 and 100 billion).

For example consider the **brute force attack** here, which works onthe principle of trying every possible key in the key range, until you get the right key.



Brute forceattack

A 2-bit binary number has four possible states:

00
01
10
11

If we have one more bit to make it a 3-bit binary number, the number of possible states also doubles to eight, as follows:

000
001
010
011
100
101
110
111

In general, if an *n bit* binary number has *k* possible states, an *n+1 bit* binary number will have *2k* possible states.

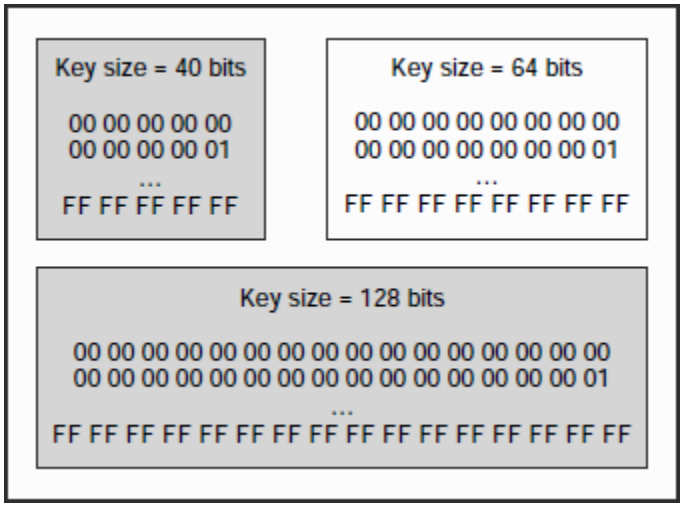
Understanding key range

With every incremental bit, the attacker has to perform double the number of operations ascompared to the previous key size. It is found that for a 56-bit key,

it takes 1 second to search 1 percentof the key range. Taking this argument further, it takes about 1 minute to search about half of the keyrange (which is what is required, on an average, to crack a key). Using this as the basis, let us have a lookat the similar values (time required for a search of 1 percent and 50 percent of the key space) for variouskey sizes. This is shown in Table

Key size on bits	*Time required to search 1 percent of the key space*	*Time required to search 50 percent of the key space*
56	1 second	1 minute
57	2 seconds	2 minutes
58	4 seconds	4 minutes
64	4.2 minutes	4.2 hours
72	17.9 hours	44.8 days
80	190.9 days	31.4 years
90	535 years	321 centuries
128	146 billion millennia	8 trillion millennia

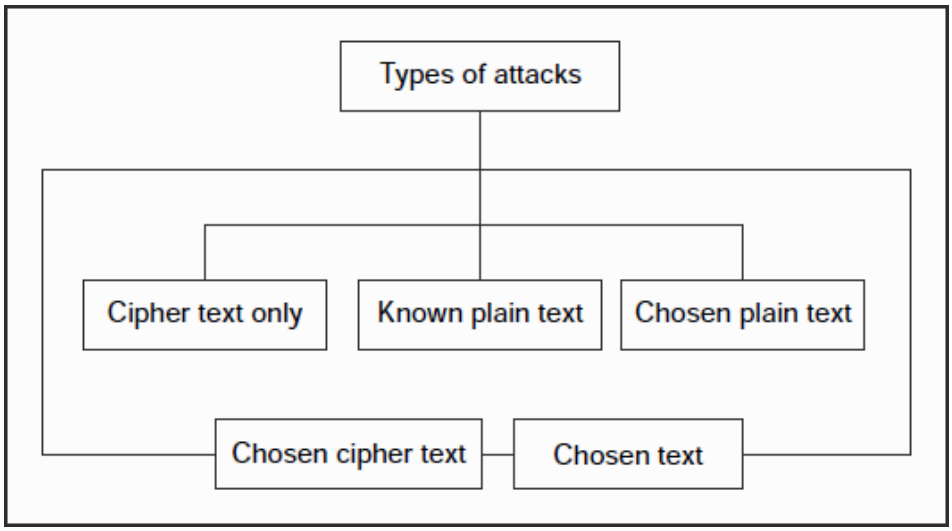
We can represent the possible values in the key range using hexadecimal notation and see visuallyhow an increase in the key size increases the key range and therefore, the complexity for an attacker.



Key sizes and ranges

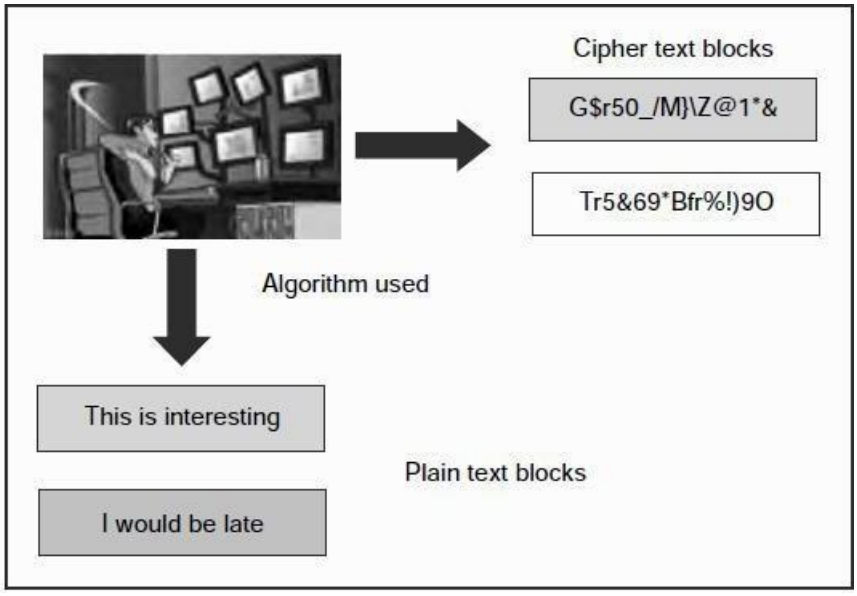
POSSIBLE TYPES OF ATTACKS

When the sender of a message encrypts a plain text message into itscorresponding cipher text, there are five possibilities for an attack on this message.



Cipher text only attack: In this type of attack, the attacker does not have any clue about the plaintext and has some or all of the cipher text.The attacker analyzes the cipher text at leisure to try andfigure out the original plain text.

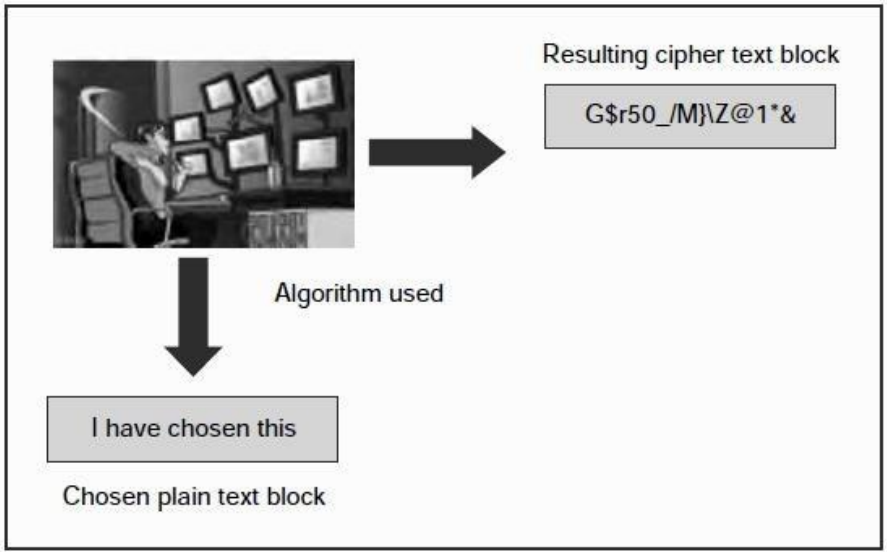
Known plain text attack: In this case, the attacker knows about some pairs of plain text andcorresponding cipher text for those pairs. Using this information, the attacker tries to find otherpairs and therefore, know more and more of the plain text. Examples of such known plain texts arecompany banners, file headers, etc. which are found commonly in all the documents of a particularcompany.



Known plain text attack

Chosen plain text attack: Here, the attacker selects a plain text block and tries to look for theencryption of the same in the cipher text. Here, the attacker is able to choose the messages toencrypt. Based on this, the attacker intentionally

picks patterns of cipher text that result inobtaining more information about the key.



Chosen plain text attack

Chosen cipher text attack: In the chosen cipher text attack, the attacker knows the cipher text tobe decrypted, the encryption algorithm that was used to produce this cipher text and thecorresponding plain text block. The attacker's job is to discover the key used for encryption.

Chosen text attack: The chosen text attack is essentially a combination of chosen plain textattack and chosen cipher text attack.

UNIT-2

SYMMETRIC KEY CIPHERS

BLOCK CIPHER PRICNIPLLES

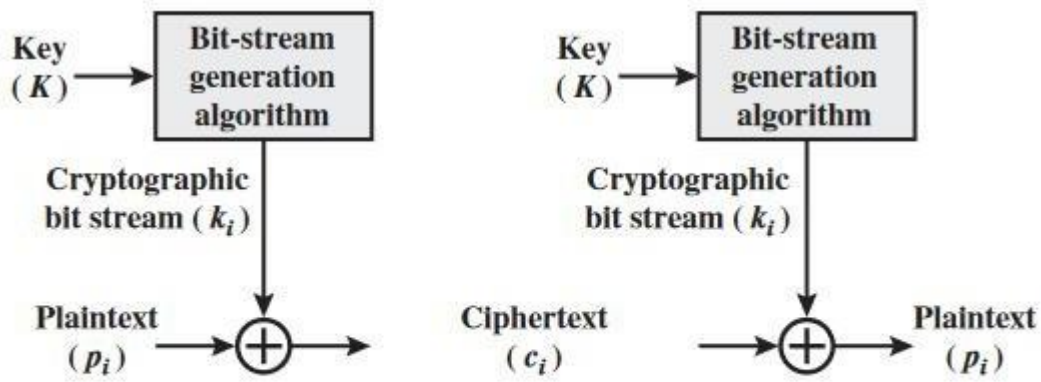
Stream Ciphers and Block Ciphers

A stream cipher is one that encrypts a digital data stream one bit or one byte at a time.

Examples of classical stream ciphers are the autokeyed Vigenère cipher and the Vernam cipher.

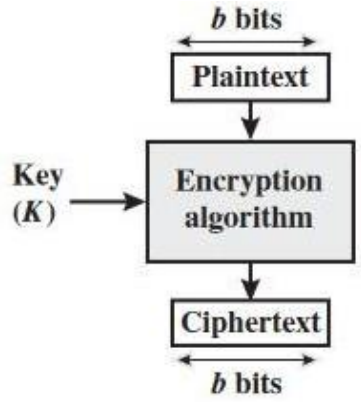
In the ideal case, a one-time pad version of the Vernam cipher wouldbe used, in which the keystream (K_i) is as long as the plaintext bit stream (P_i). If the cryptographic keystream is random, then this cipher is unbreakable by anymeans other than acquiring the keystream.

The bit-stream generator is akey-controlled algorithm and must produce a bit stream that is cryptographicallystrong.



(a) Stream cipher using algorithmic bit-stream generator

A block cipher is one in which a block of plaintext is treated as a whole andused to produce a ciphertext block of equal length. Typically, a block size of 64 or128 bits is used. As with a stream cipher, the two users share a symmetric encryptionkey.



(b) Block cipher

Motivation for the Feistel Cipher Structure

A block cipher operates on a plaintext block of n bits to produce a ciphertextblock of n bits. There are 2^npossible different plaintext blocks and, forthe encryption to be reversible (i.e., for decryption to be possible), each mustproduce a unique ciphertext block.

Reversible Mapping

Plaintext	Ciphertext
00	11
01	10
10	00
11	01

Irreversible Mapping

Plaintext	Ciphertext
00	11
01	10
10	01
11	01

The logic of a general substitution cipher for A 4-bit input produces one of 16 possible input states, which is mapped by thesubstitution cipher into a unique one of 16 possible output states, each of which isrepresented by 4 ciphertext bits.

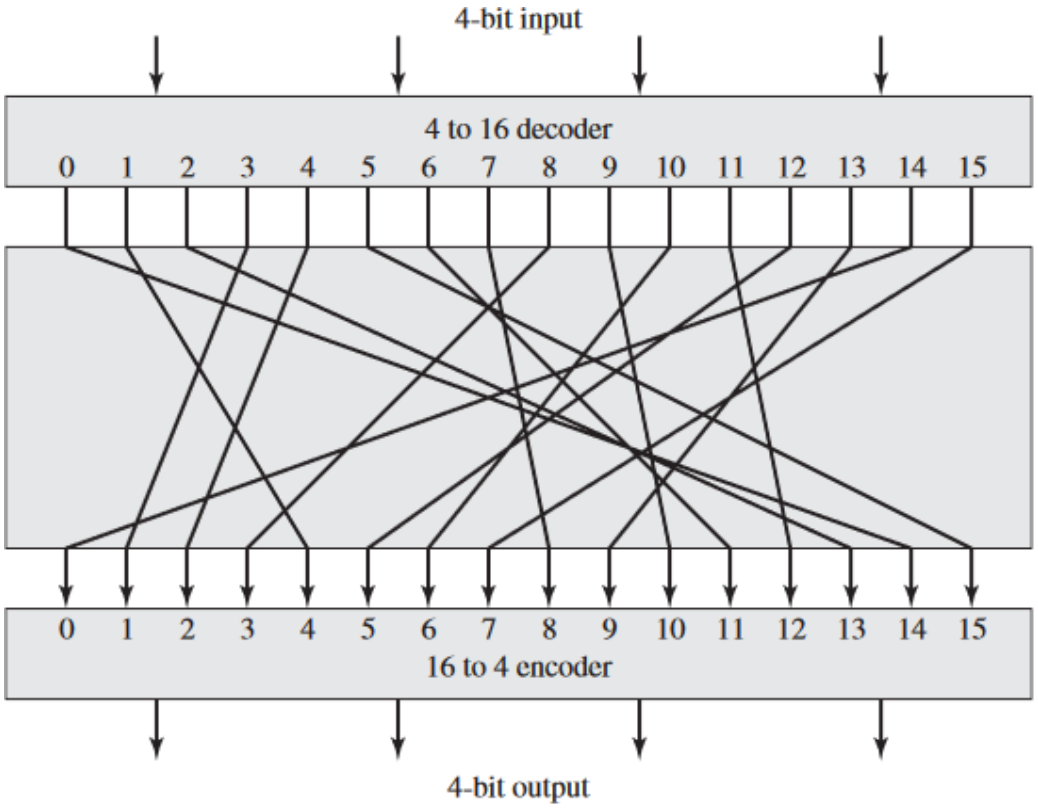


Figure 3.2 General n-bit-n-bit Block Substitution (shown with $n = 4$)

Table 3.1 Encryption and Decryption Tables for Substitution Cipher of Figure 3.2

Plaintext	Ciphertext
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

Ciphertext	Plaintext
0000	1110
0001	0011
0010	0100
0011	1000
0100	0001
0101	1100
0110	1010
0111	1111
1000	0111
1001	1101
1010	1001
1011	0110
1100	1011
1101	0010
1110	0000
1111	0101

Block Cipher Principles

A block cipher is designed by considering its three critical aspects which are listed as below:

1. Number of Rounds
2. Design of Function F
3. Key Schedule Algorithm

1. Number of Rounds

The number of rounds judges the strength of the block cipher algorithm. It is considered that more is the number of rounds, difficult is for cryptanalysis to break the algorithm.

It is considered that even if the function F is relatively weak, the number of rounds would make the algorithm tough to break.

2. Design of Function F

The function F of the block cipher must be designed such that it must be impossible for any cryptanalysis to unscramble the substitution. The criterion that strengthens the function F is it non-linearity.

More the function F is nonlinear, more it would be difficult to crack it. Well, while designing the function F it should be confirmed that it has a good avalanche property which states that a change in one-bit of input must reflect the change in many bits of output.

The Function F should be designed such that it possesses a bit independence criterion which states that the output bits must change independently if there is any change in the input bit.

3. Key Schedule Algorithm

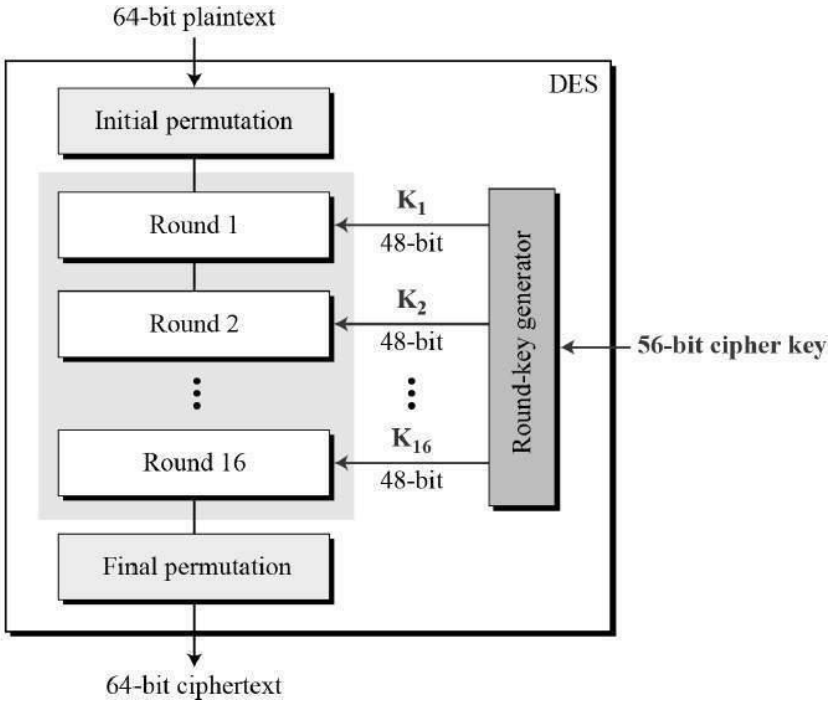
It is suggested that the key schedule should confirm the strict avalanche effect and bit independence criterion.

DATA ENCRYPTION STANDARD

The Data Encryption Standard (DES) is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).

DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure. The block size is 64-bit. Though, key length is 64-bit, DES has an effective key

length of 56 bits, since 8 of the 64 bits of the key are not used by the encryption algorithm (function as check bits only). General Structure of DES is depicted in the following illustration –

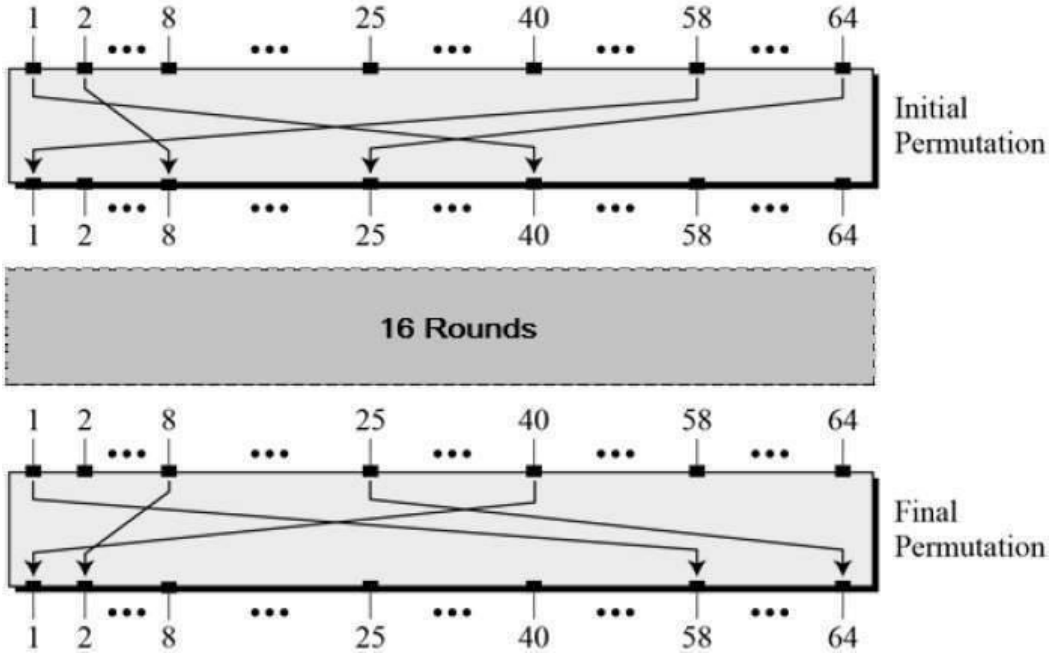


Since DES is based on the Feistel Cipher, all that is required to specify DES is –

- Round function
- Key schedule
- Any additional processing – Initial and final permutation

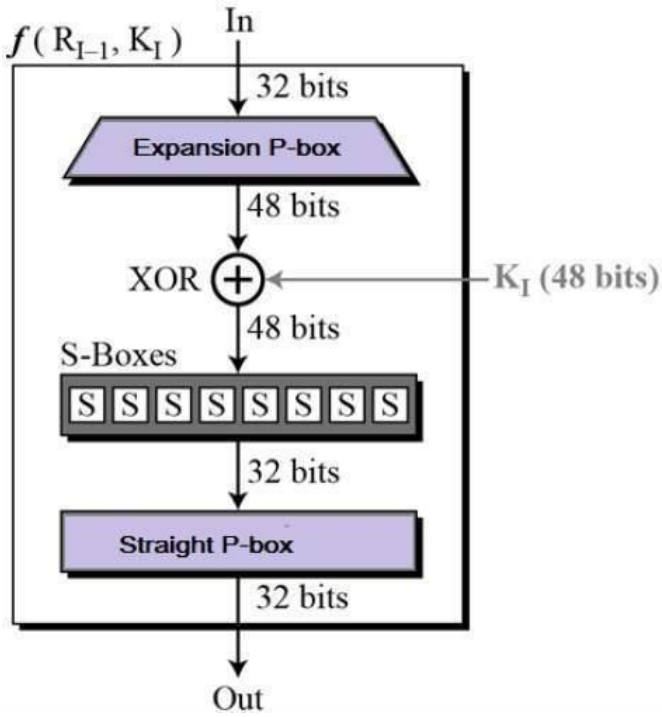
Initial and Final Permutation

The initial and final permutations are straight Permutation boxes (P-boxes) that are inverses of each other. They have no cryptography significance in DES. The initial and final permutations are shown as follows –

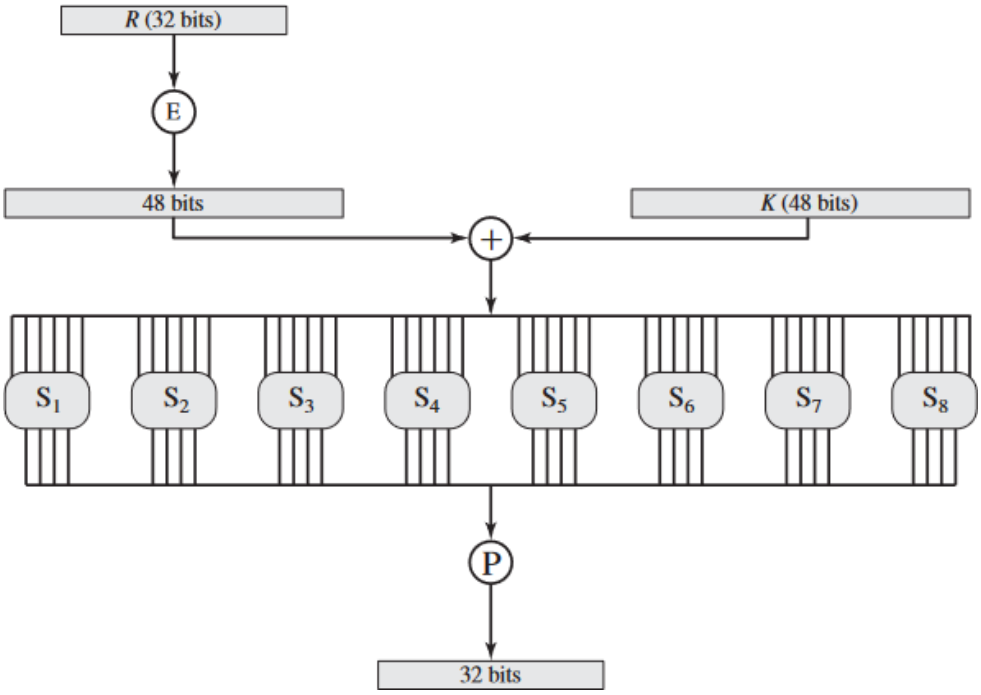


Round Function

The heart of this cipher is the DES function, f. The DES function applies a 48-bit key to the rightmost 32 bits to produce a 32-bit output.



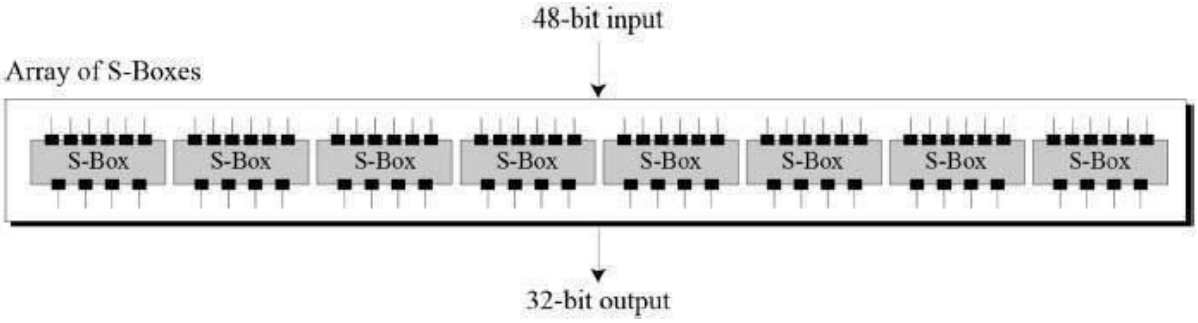
Expansion Permutation Box –Since right input is 32-bit and round key is a 48-bit, we first need to expand right input to 48 bits. Permutation logic is graphically depicted in the following illustration



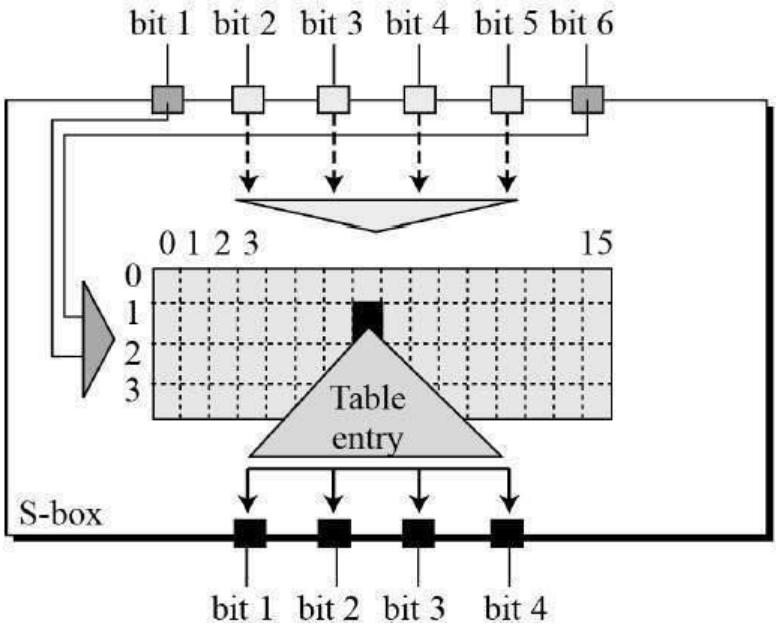
The graphically depicted permutation logic is generally described as table in DES specification illustrated as shown

32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	31	31	32	01

- **XOR (Whitener).** – After the expansion permutation, DES does XOR operation on the expanded right section and the round key. The round key is used only in this operation.
- **Substitution Boxes.** – The S-boxes carry out the real mixing (confusion). DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output. Refer the following illustration –



The S-box rule is illustrated below

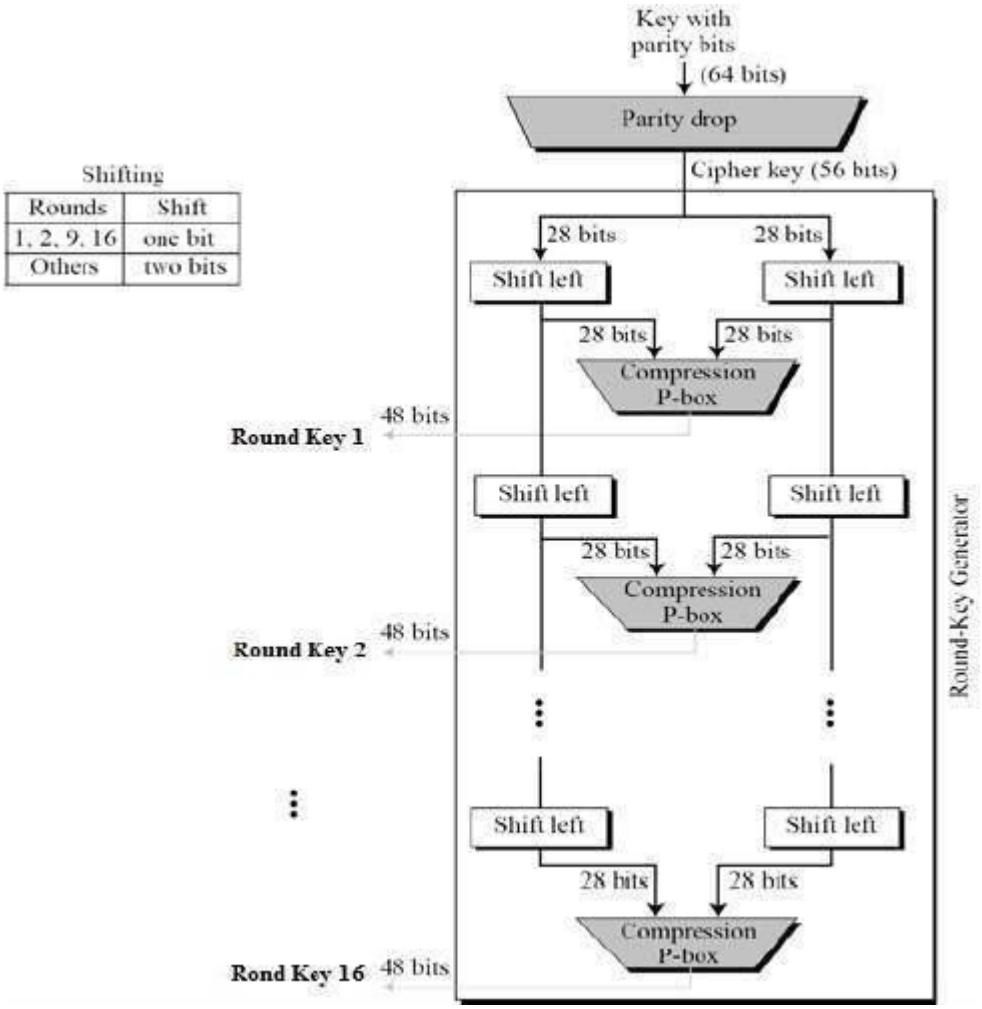


 There are a total of eight S-box tables. The output of all eight s-boxes is then combined in to 32 bit section.

 Straight Permutation – The 32 bit output of S-boxes is then subjected to the straight permutation with rule shown in the following illustration:

Key Generation

The round-key generator creates sixteen 48-bit keys out of a 56-bit cipher key. The process of key generation is depicted in the following illustration



ADVANCED ENCRYPTION STANDARD

The Advanced Encryption Standard (AES) was published by the National Institute ofStandards and Technology (NIST) in 2001. AES is a symmetric block cipher that isintended to replace DES as the approved standard for a wide range of applications.

AES is a block cipher intended to replace DES for commercial applications. It uses a 128-bit block size and a key size of 128, 192, or 256 bits.

AES does not use a Feistel structure. Instead, each full round consists offour separate functions: byte substitution, permutation, arithmetic operations over a finite field, and XOR with a key.

General Structure

Figure shows the overall structure of the AES encryption process. The cipher takes a plaintext block size of 128 bits, or 16 bytes. The key length can be 16, 24, or 32bytes (128, 192, or 256 bits). The algorithm is referred to as AES-128, AES-192, orAES-256, depending on the key length.

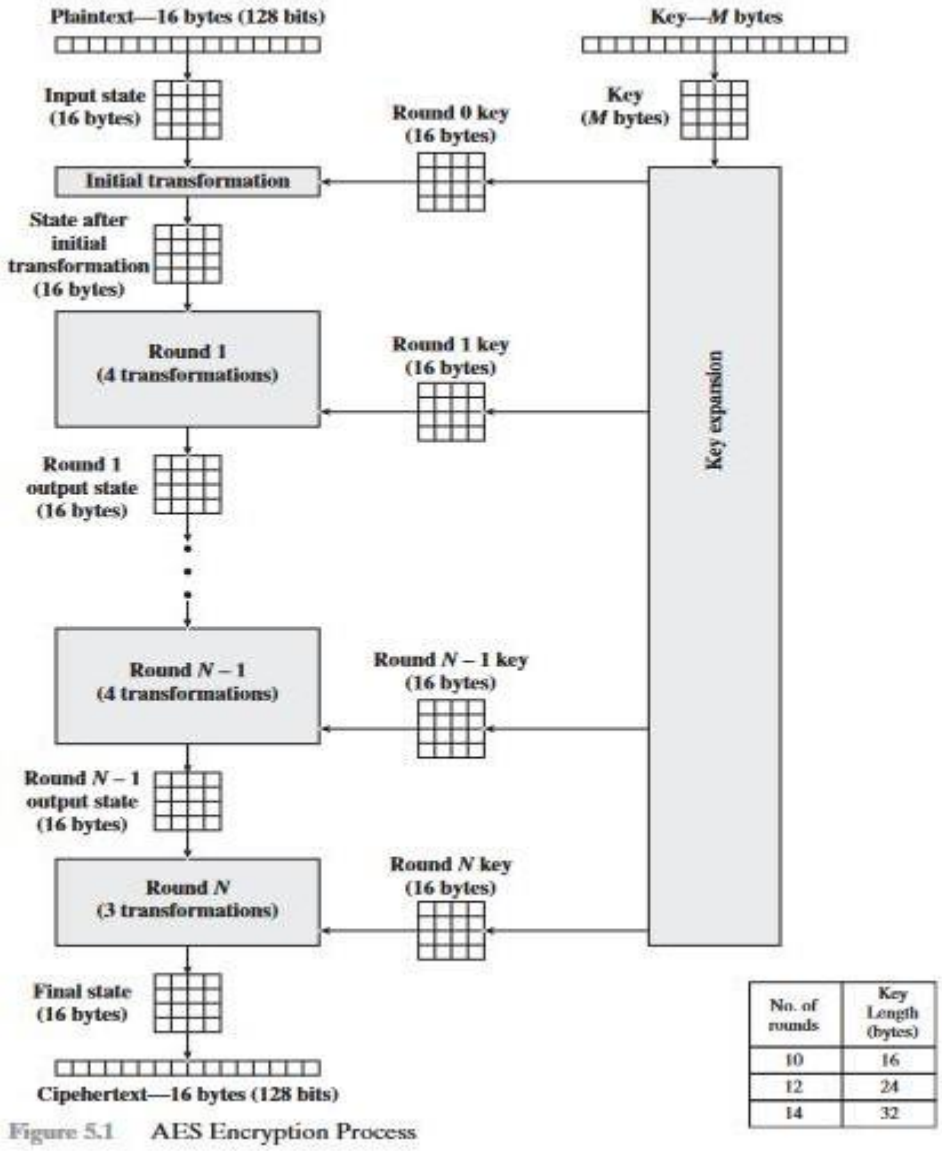


Figure 5.1 AES Encryption Process

Advanced Encryption Standard is found at least six time faster than triple DES. A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.

The features of AES are as follows –

- Symmetric key symmetric block cipher
- 128-bit data, 128/192/256-bit keys
- Stronger and faster than Triple-DES
- Provide full specification and design details
- Software implementable in C and Java

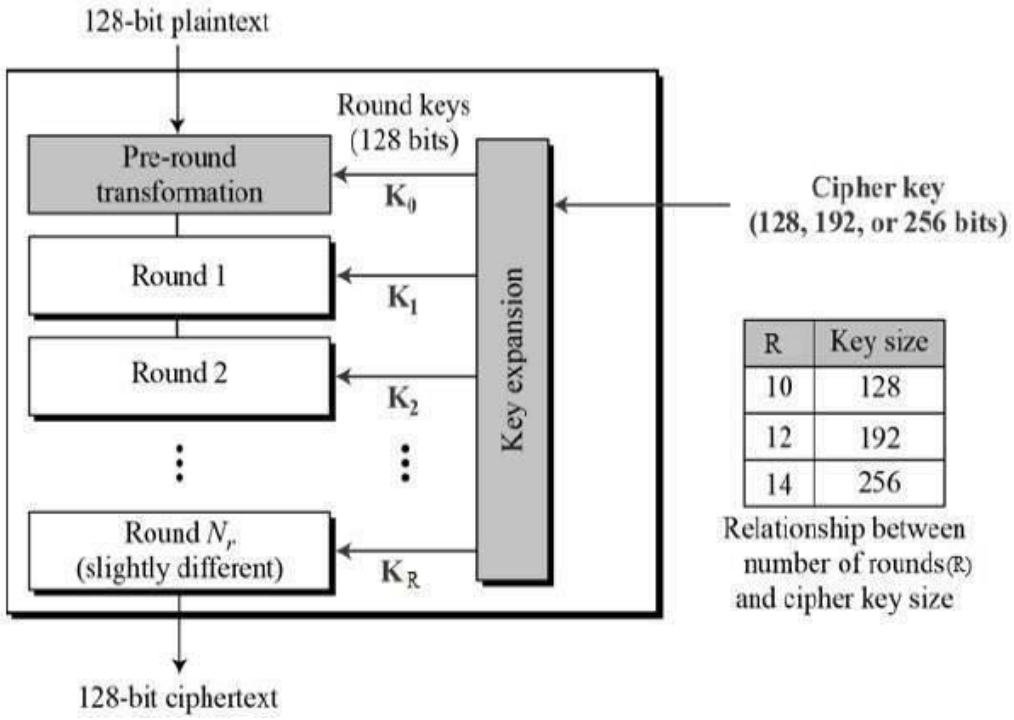
Operation of AES

AES is an iterative rather than Feistel cipher. It is based on 'substitution-permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix –

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.

The schematic of AES structure is given in the following illustration

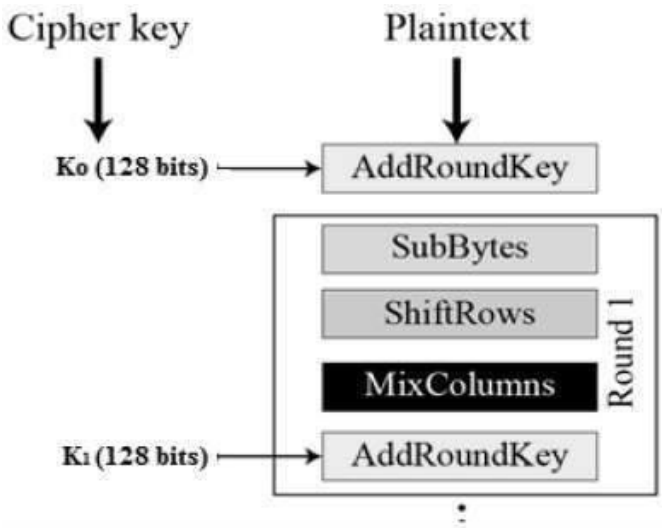


R	Key size
10	128
12	192
14	256

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 comprise of four sub-processes. The first round process is depicted below –



Byte Substitution (SubBytes)

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.

Shiftrows

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.

```
[ b0  | b1  | b2  | b3  ]          [ b0  | b1  | b2  | b3  ]
| b4  | b5  | b6  | b7  |    ->    | b5  | b6  | b7  | b4  |
| b8  | b9  | b10 | b11 |          | b10 | b11 | b8  | b9  |
[ b12 | b13 | b14 | b15 ]          [ b15 | b12 | b13 | b14 ]
```

MixColumns

Each column of four bytes is now transformed using a special mathematical function. This function takes as input the four bytes of one column and outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

```
[ c0 ]          [ 2  3  1  1 ] [ b0 ]
| c1 |  =       | 1  2  3  1 || b1 |
| c2 |          | 1  1  2  3 || b2 |
[ c3 ]          [ 3  1  1  2 ][ b3 ]
```

Addroundkey

The 16 bytes of the matrix are now considered as 128 bits and are XORed to the 128 bits of the round key. If this is the last round then the output is the ciphertext. Otherwise, the resulting 128 bits are interpreted as 16 bytes and we begin another similar round.

Decryption Process

The process of decryption of an AES ciphertext is similar to the encryption process in the reverse order. Each round consists of the four processes conducted in the reverse order –

- Add round key
- Mix columns
- Shift rows
- Byte substitution

Since sub-processes in each round are in reverse manner, unlike for a Feistel Cipher, the encryption and decryption algorithms needs to be separately implemented, although they are very closely related.

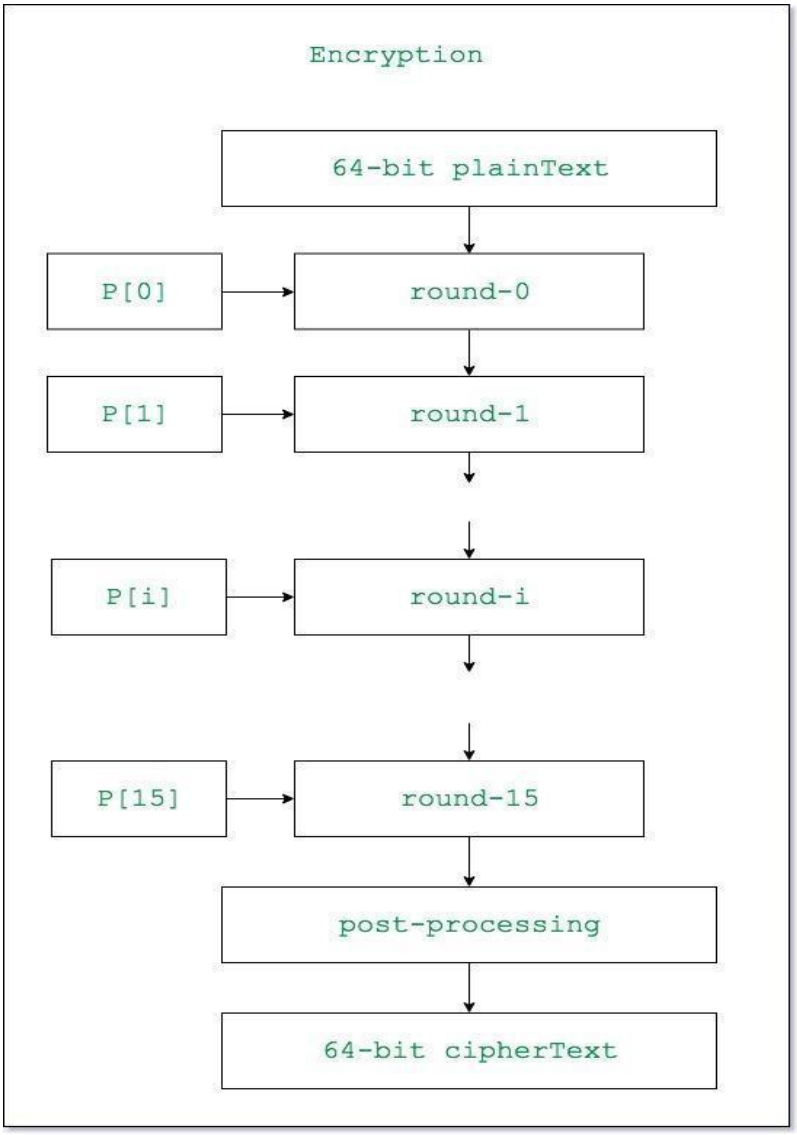
BLOWFISH ALGORITHM

Blowfish is an encryption technique designed by Bruce Schneier in 1993 as an alternative to DES Encryption Technique. It is significantly faster than DES and provides a good encryption rate with no effective cryptanalysis technique found to date. It is one of the first, secure block cyphers not subject to any patents and hence freely available for anyone to use.

1. blockSize: 64-bits
2. keySize: 32-bits to 448-bits variable size
3. number of subkeys: 18 [P-array]
4. number of rounds: 16
5. number of substitution boxes: 4 [each having 512 entries of 32-bits each]

Blowfish Encryption Algorithm

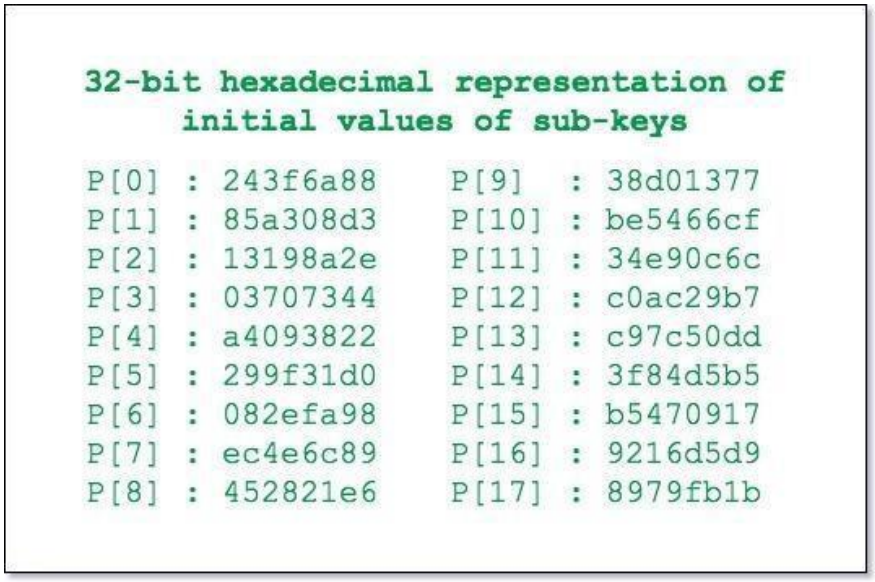
The entire encryption process can be elaborated as:



Lets see each step one by one:

Step1: Generation of subkeys:

- 18 subkeys{P[0]…P[17]} are needed in both encryption as well as decryption process and the same subkeys are used for both the processes.
- These 18 subkeys are stored in a P-array with each array element being a 32-bit entry.
- It is initialized with the digits of pi(?).
- The hexadecimal representation of each of the subkeys is given by:



Now each of the subkey is changed with respect to the input key as:

P[0] = P[0] xor 1st 32-bits of input key
P[1] = P[1] xor 2nd 32-bits of input key
.
.
.
P[i] = P[i] xor (i+1)th 32-bits of input key
(roll over to 1st 32-bits depending on the key length)
.
.
.
P[17] = P[17] xor 18th 32-bits of input key
(roll over to 1st 32-bits depending on key length)

The resultant P-array holds 18 subkeys that is used during the entire encryption process

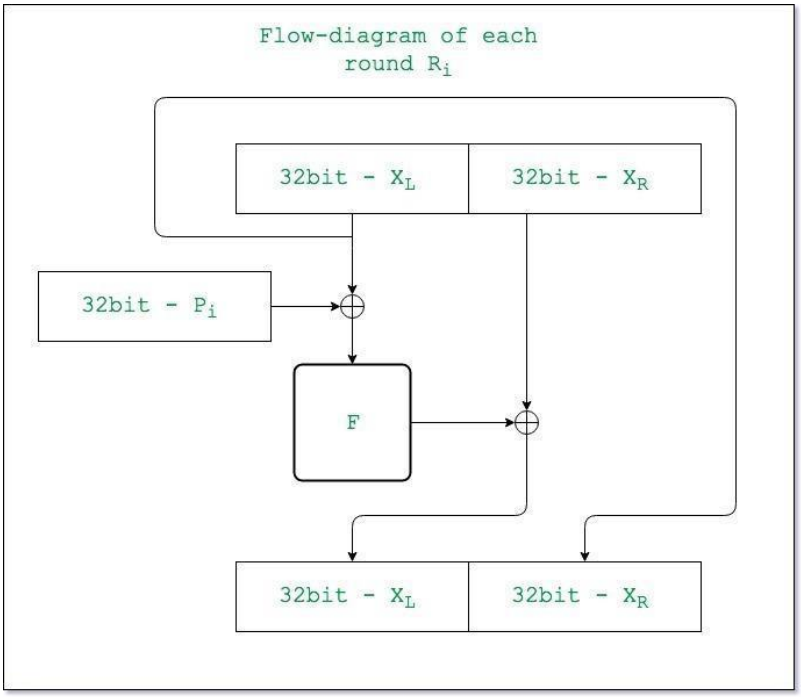
Step2: initialise Substitution Boxes:

- 4 Substitution boxes(S-boxes) are needed{S[0]…S[4]} in both encryption aswell as decryption process with each S-box having 256 entries{S[i][0]…S[i][255], 0&lei&le4} where each entry is 32-bit.
- It is initialized with the digits of pi(?) after initializing the P-array.

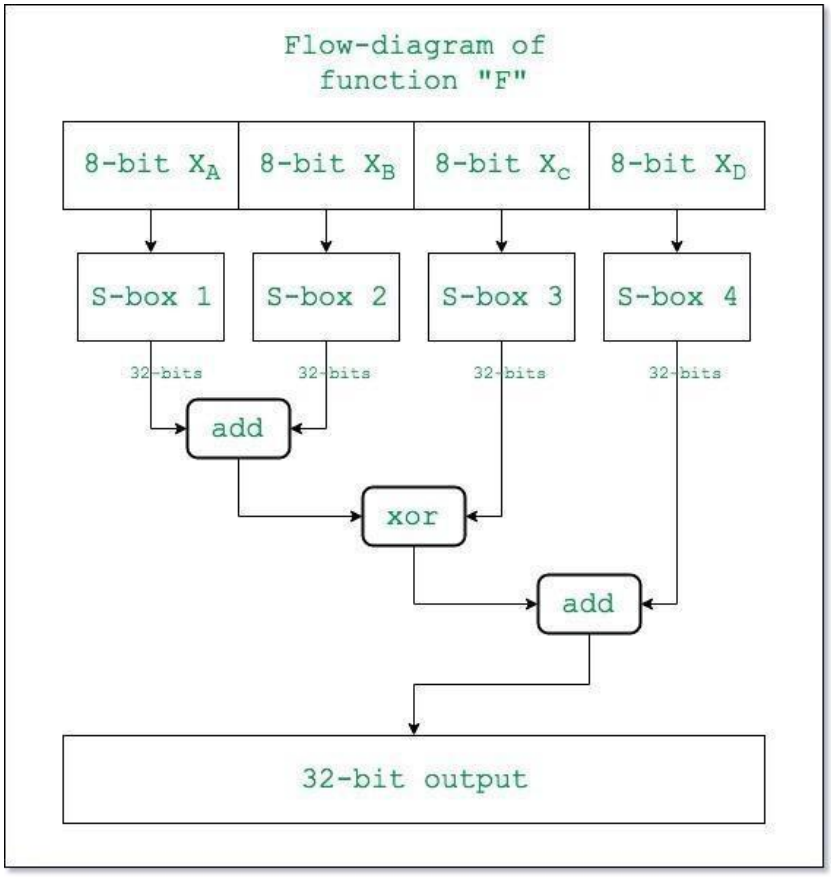
Step3: Encryption:

- The encryption function consists of two parts:
  a. Rounds: The encryption consists of 16 rounds with each round(Ri)

taking inputs the plainText(P.T.) from previous round and corresponding subkey(Pi). The description of each round is as follows:

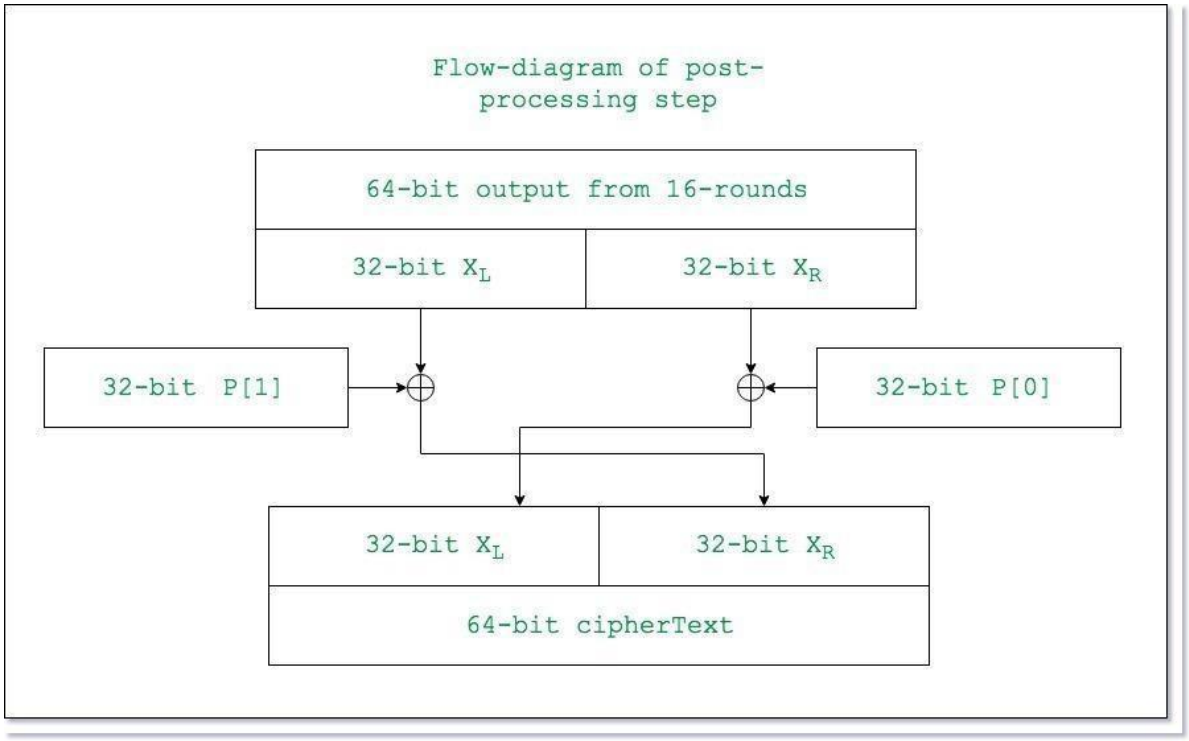


The description of the function " F " is as follows:



Here the function "add" is addition modulo 2^32.

b. Post-processing: The output after the 16 rounds is processed as follows:



DIFFERENTIAL AND LINEAR CRYPTANALYSIS

One of the most significant advances in cryptanalysis in recent years is differentialcryptanalysis.

DIFFERENTIAL CRYPTANALYSIS ATTACK The differential cryptanalysis attack is complex, provides a complete description. The rationale behind differentialcryptanalysis is to observe the behaviour of pairs of text blocks evolving along eachround of the cipher, instead of observing the evolution of a single text block.

We begin with a change in notation for DES. Consider the original plaintext block m to consist of two halves m_0, m_1. Each round of DES maps the right-hand input into the left-hand output and sets the right-hand output to be a function of the left-hand input and the subkey for this round. So, at each round, only one new 32-bit block is created. If we label each new block $m_i (2 \leq i \leq 17)$, then the intermediate message halves are related as follows:

$$m_{i+1} = m_{i-1} \oplus \mathrm{f}(m_i, K_i), \qquad i = 1, 2, \ldots, 16$$

In differential cryptanalysis, we start with two messages, m and m', with a known XOR difference $\Delta m = m \oplus m'$, and consider the difference between the intermediate message halves: $\Delta m_i = m_i \oplus m'_i$. Then we have

$$\begin{aligned} \Delta m_{i+1} &= m_{i+1} \oplus m'_{i+1} \\ &= [m_{i-1} \oplus \mathrm{f}(m_i, K_i)] \oplus [m'_{i-1} \oplus \mathrm{f}(m'_i, K_i)] \\ &= \Delta m_{i-1} \oplus [\mathrm{f}(m_i, K_i) \oplus \mathrm{f}(m'_i, K_i)] \end{aligned}$$

Linear Cryptanalysis

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

We now give a brief summary of the principle on which linear cryptanalysis is based. For a cipher with n-bit plaintext and ciphertext blocks and an m-bit key, let the plaintext block be labeled P[1], ... P[n], the cipher text block C[1], ... C[n], and the key K[1], ..., K[m]. Then define

$$A[i, j, \ldots, k] = A[i] \oplus A[j] \oplus \ldots \oplus A[k]$$

The objective of linear cryptanalysis is to find an effective *linear* equation of the form:

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

(where $x = 0$ or 1; $1 \leq a$; $b \leq n$; $c \leq m$; and where the α, β, and γ terms represent fixed, unique bit locations) that holds with probability $p \neq 0.5$. The further p is from 0.5, the more effective the equation. Once a proposed relation is determined, the procedure is to compute the results of the left-hand side of the preceding equation for a large number of plaintext–ciphertext pairs. If the result is 0 more than half the time, assume $K[\gamma_1, \gamma_2, \ldots, \gamma_c] = 0$. If it is 1 most of the time, assume $K[\gamma_1, \gamma_2, \ldots, \gamma_c] = 1$.

BLOCK CIPHER MODES OF OPERATION

A block cipher processes the data blocks of fixed size. Usually, the size of a message is larger than the block size. Hence, the long message is divided into a series of sequential message blocks, and the cipher operates on these blocks one at a time.

Electronic Code Book (ECB) Mode

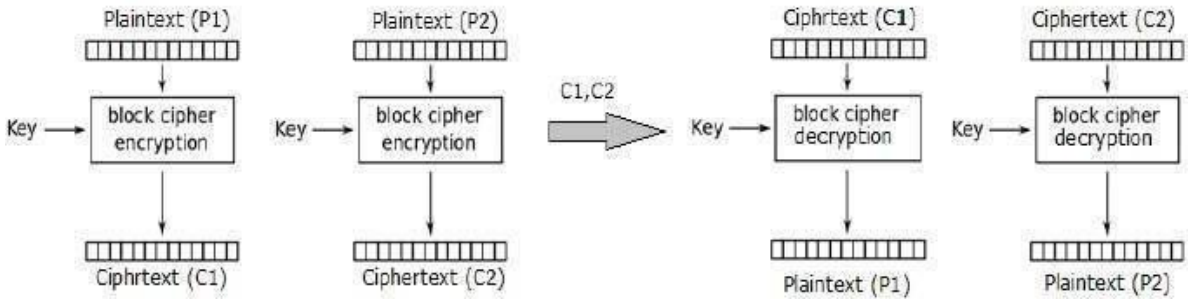
This mode is a most straightforward way of processing a series of sequentially listed message blocks.

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

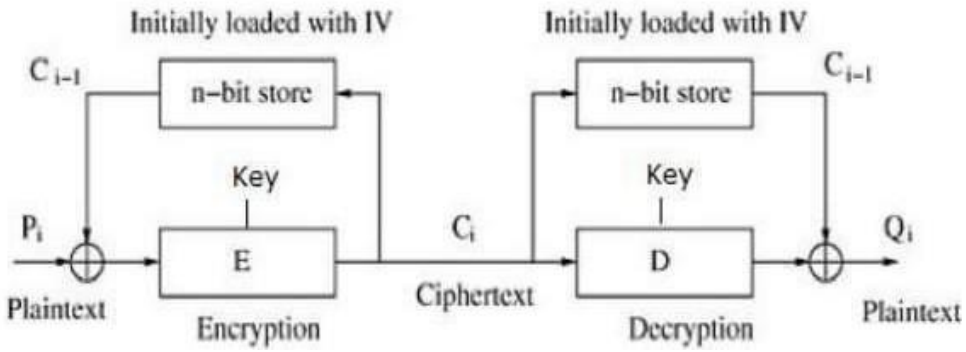
CBC mode of operation provides message dependence for generating ciphertext and makes the system non-deterministic.

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 block 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

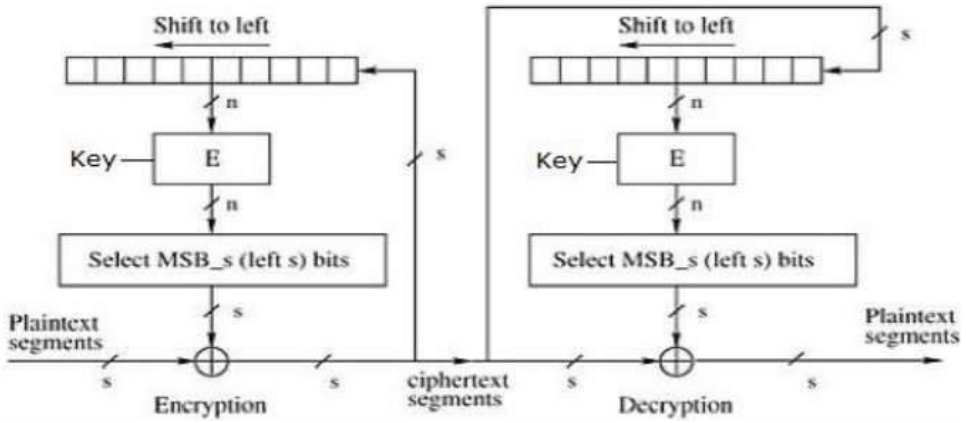
In this mode, each ciphertext block gets 'fed back' into the encryption process in order to encrypt the next plaintext block.

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' bits 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.

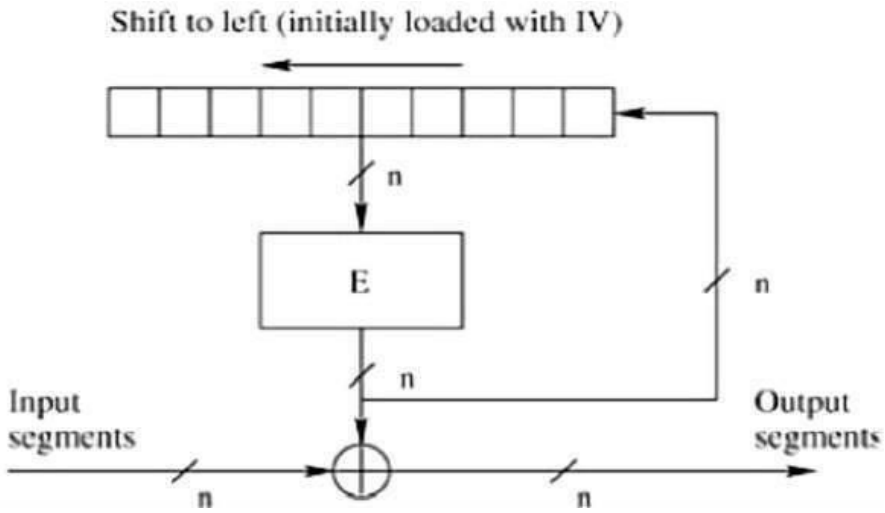
On the flip side, the error of transmission gets propagated due to changing of blocks.

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

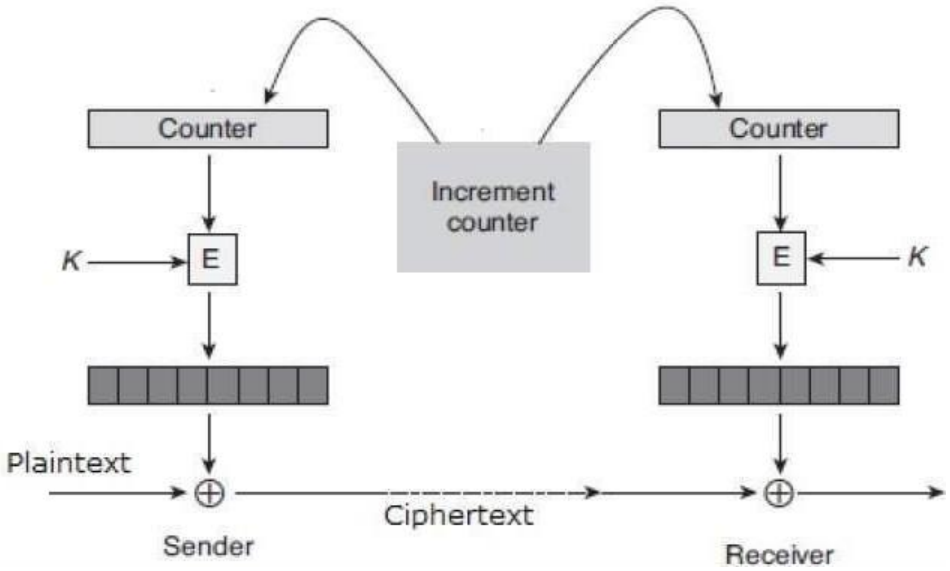
It can be considered as a counter-based version of CFB mode without the feedback. In this mode, both the sender and receiver need to access to a reliable counter, which computes a new shared value each time a ciphertext block is exchanged. This shared counter is not necessarily a secret value, but challenge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

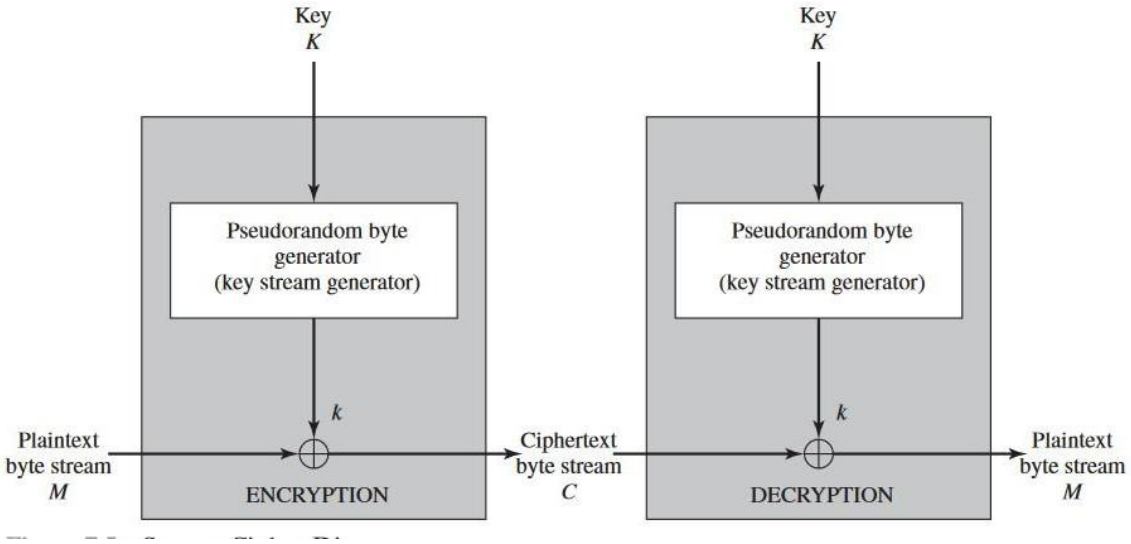
- Take the first plaintext block P1 and XOR this to the contents of the bottom register. The result of this is C1. Send C1 to the receiver and update the counter. The counter update replaces the ciphertext feedback in CFB mode.
- Continue in this manner until the last plaintext block has been encrypted.
- 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.



STREAM CIPHERS

A typical stream cipher encrypts plaintext one byte at a time, although a stream cipher may be designed to operate on one bit at a time or on units larger than a byteat a time. A key is input to a pseudorandom bit generator that produces a stream of8-bit numbers that are apparently random. The output of the generator, called akeystream, is combined one byte at a time with the plaintext stream using the bit-wise exclusive-OR (XOR) operation. For example, if the next byte generated by thegenerator is 01101100 and the next plaintext byte is 11001100, then the resultingciphertext byte is

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



STREAM CIPHERS

Decryption requires the use of the same pseudorandom sequence

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

RC4

RC4 is a stream cipher designed in 1987 by Ron Rivest for RSA Security. It is a variable key size stream cipher with byte-oriented operations. The algorithm is based onthe use of a random permutation. Analysis shows that the period of the cipher is overwhelmingly likely to be greater than 10^{100}. Eight to sixteen machine operations are required per output byte, and the cipher can be expected to run veryquickly in software. RC4 is used in the Secure Sockets Layer/Transport Layer Security(SSL/TLS) standards that have been defined for communication between Webbrowsers and servers.

The RC4 algorithm is remarkably simple and quite easy to explain. A variable length key of from 1 to 256 bytes (8 to 2048 bits) is used to initialize a 256-bytestate vector S, with elementsS[0],S[1],S[2], S[255].

Initialization of S

To begin, the entries are set equal to the values from 0 through 255 in ascendingorder; that is, S[0],S[1],S[2], S[255]=255.

A temporary vector, T, is alsocreated. If the length of the key K is 256 bytes, then T is transferred to T. Otherwise,for a key of length keylen bytes, the first keylen elements of T are copied from K,and then K is repeated as many times as necessary to fill out T. These preliminaryoperations can be summarized as

```
/* Initialization */
for i = 0 to 255 do
S[i] = i;
T[i] = K[i mod keylen];
```

Next we use T to produce the initial permutation of S. This involves starting with S[0] and going through to S[255], and for each S[i], swapping S[i] with another byte in S according to a scheme dictated by T[i]:

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

Stream Generation

Once the S vector is initialized, the input key is no longer used. Stream generationinvolves cycling through all the elements of S[i] , and for each S[i], swapping S[i] withanother byte in S according to a scheme dictated by the current configuration of S.After S[255]is reached, the process continues, starting over again at S[0].

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

LOCATION AND PLACEMENT OF ENCRYPTION FUNCTION

If encryption is to be used to counter attacks on confidentiality, we need to decide what to encrypt and where the encryption function should be located. To begin, this section examines the potential locations of security attacks and then looks at the two major approaches to encryption placement: link and end to end.

Potential Locations for Confidentiality Attacks

As an example, consider a user workstation in a typical business organization. Figure 7.1 suggests the types of communications facilities that might be employed by such a workstation and therefore gives an indication of the points of vulnerability.

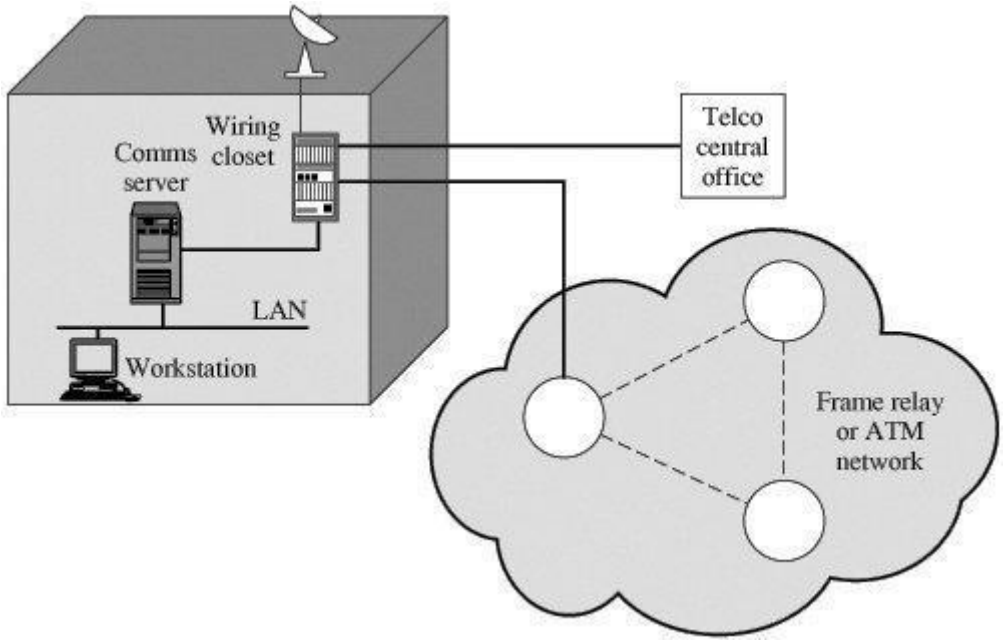


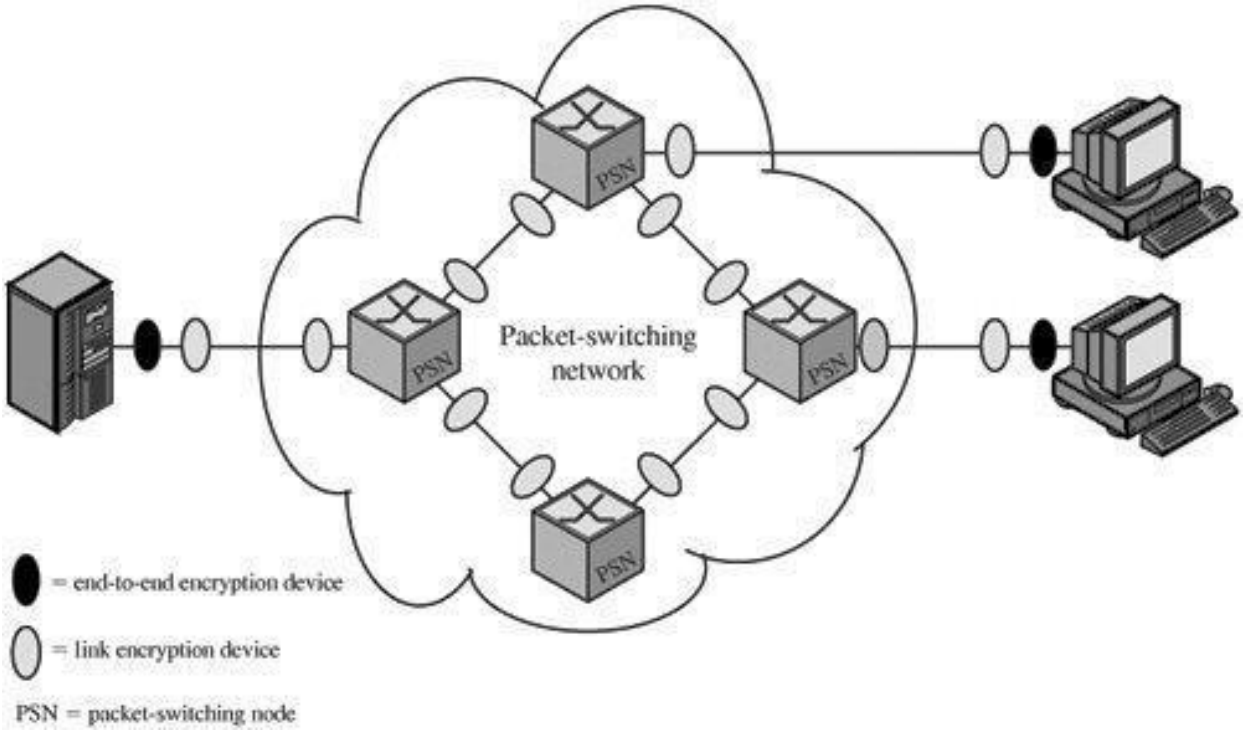
Figure 7.1. Points of Vulnerability

In most organizations, workstations are attached to local area networks (LANs). Typically, the user can reach other workstations, hosts, and servers directly on the LAN or on other LANs in the same building that are interconnected with bridges and routers. Here, then, is the first point of vulnerability. In this case, the main concern is eavesdropping by another employee. Typically, a LAN is a broadcast network: Transmission from any station to any other station is visible on the LAN medium to all stations. Data are transmitted in the form of frames, with each frame containing the source and destination address. An eavesdropper can monitor the traffic on the LAN and capture any traffic desired on the basis of source and destination addresses. If part or all of the LAN is wireless, then the potential for eavesdropping is greater.

Link versus End-to-End Encryption

The most powerful and most common approach to securing the points of vulnerability highlighted in the preceding section is encryption. If encryption is to be used to counter these attacks, then we need to decide what to encrypt and

where the encryption gear should be located. As Figure indicates, there are two fundamental alternatives: link encryption and end-to-end encryption.



Encryption Across a Packet-Switching Network

Table 7.1. Characteristics of Link and End-to-End Encryption [PFLE02]

Link Encryption	End-to-End Encryption
Security within End Systems and Intermediate Systems	
Message exposed in sending host	Message encrypted in sending host
Message exposed in intermediate nodes	Message encrypted in intermediate nodes
Role of User	
Applied by sending host	Applied by sending process
Transparent to user	User applies encryption
Host maintains encryption facility	User must determine algorithm
One facility for all users	Users selects encryption scheme
Can be done in hardware	Software implementation
All or no messages encrypted	User chooses to encrypt, or not, for each message
Implementation Concerns	
Requires one key per (host-intermediate node) pair and (intermediate node-intermediate node) pair	Requires one key per user pair
Provides host authentication	Provides user authentication

PRINCIPLES OF PUBLIC KEY CRYPTOSYSTEMS

The concept of public-key cryptography evolved from an attempt to attack two ofthe most difficult problems associated with symmetric encryption. The first problem is that of key distribution.

The second problem that Diffie pondered, and one that was apparentlyunrelated to the first, was that of digital signatures.

Public key Cryptosystem – Asymmetric algorithms depends on one key for encryption and a distinct but related key for decryption. These algorithms have the following characteristics which are as follows –

- It is computationally infeasible to decide the decryption key given only information of the cryptographic algorithm and the encryption key.
- There are two related keys such as one can be used for encryption, with the other used for decryption.

A public key encryption scheme has the following ingredients which are as follows

- **Plaintext** – This is the readable message or information that is informer into the algorithm as input.
- **Encryption algorithm** – The encryption algorithm performs several conversion on the plaintext.
- **Public and Private keys** – This is a set of keys that have been selected so that if one can be used for encryption, and the other can be used for decryption.
- **Ciphertext** – This is scrambled message generated as output. It based on the plaintext and the key. For a given message, there are two specific keys will create two different ciphertexts.
- **Decryption Algorithm** – This algorithm get the ciphertext and the matching key and create the original plaintext.

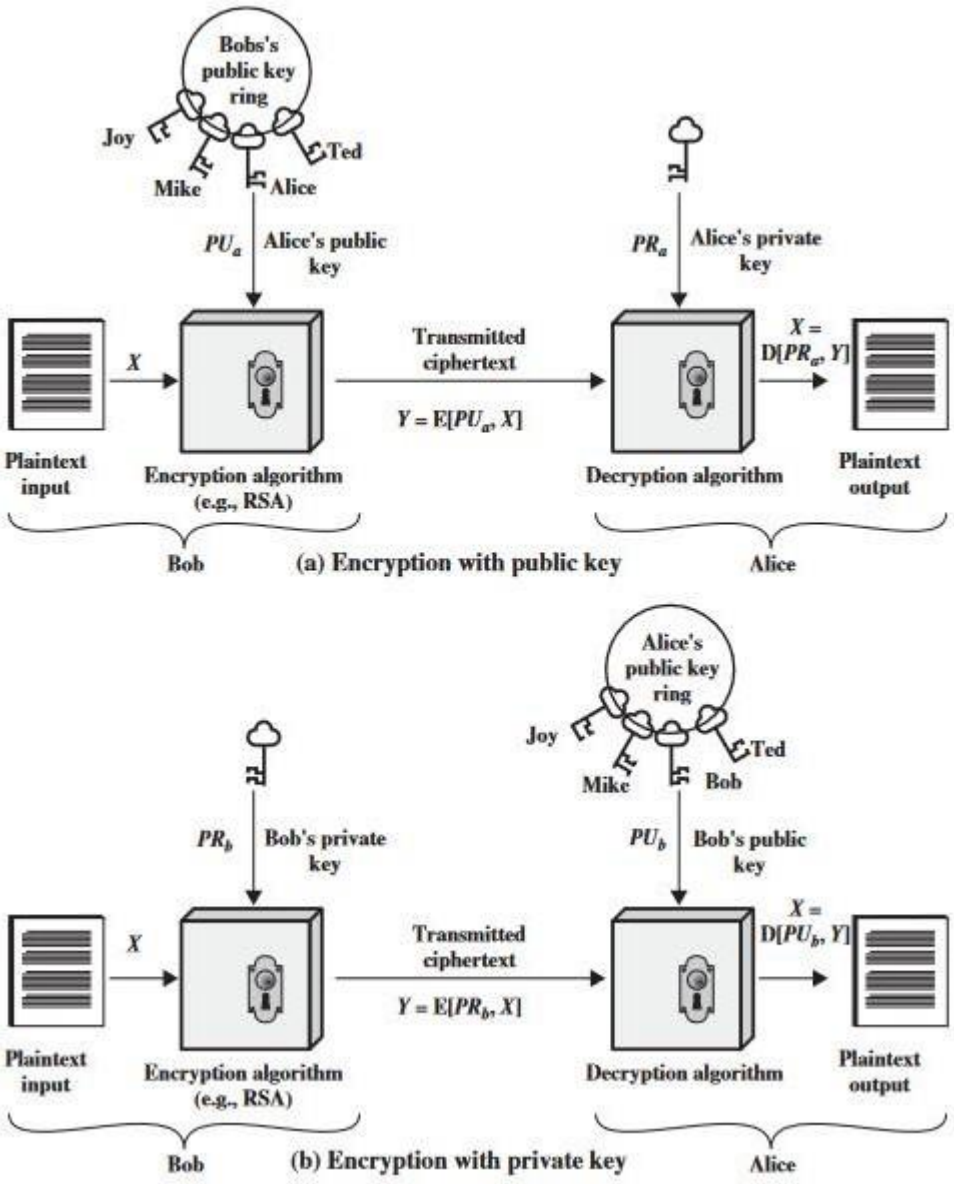
The essential steps are the following.

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

2. 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. As in Figure suggests, each user maintains a collection of public keys obtained from others.

3. If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.

4. When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows Alice'sprivate key.



Public Key Cryptography Requirements

To accomplish the public key cryptography there are following requirements as discussed below.

- 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 ALGORITHM

In this algorithm two keys were used. One is private key and another one is public key.

RSA Algorithm

- Ron Rivest, Adi Shamir and Len Adleman have developed this algorithm (Rivest-Shamir-Adleman). It is a block cipher which converts plain text into cipher text and vice versa at receiver side.

- **The algorithm works as follow:**
  1. Select two prime numbers p and q where $p \neq q$.
  2. Calculate $n = p * q$.
  3. Calculate $\Phi(n) = (p-1) * (q-1)$.
  4. Select e such that, e is relatively prime to $\Phi(n)$
     i.e. $(e, \Phi(n)) = 1$ and $1 < e < \Phi(n)$
  5. Calculate $d = e^{-1} \bmod \Phi(n)$ or $ed = 1 \bmod \Phi(n)$.
  6. Public key = {e, n}, private key = {d, n}.
  7. Find out cipher text using the formula,
     $C = P^e \bmod n$ where, $P < n$ and
     C = Cipher text, P = Plain text, e = Encryption key and n=block size.
  8. $P = C^d \bmod n$. Plain text P can be obtain using the given formula.
     where, d = decryption key.

RSA Algorithm step by step explanation

- **Step – 1:** Select two prime numbers p and q where $p \neq q$.
- **Step – 2:** Calculate $n = p * q$.
- **Step – 3:** Calculate $\Phi(n) = (p-1) * (q-1)$.
- **Step – 4:** Select e such that, e is relatively prime to $\Phi(n)$
  i.e. $(e, \Phi(n)) = 1$ and $1 < e < \Phi(n)$

❖**Explanation with example:**

1. Two prime numbers $p = 13$, $q = 11$.
2. $n = p * q = 13 * 11 = 143$.
3. $\Phi(n) = (13 - 1) * (12 - 1) = 12 * 10 = 120$.
4. Select $e = 13$, gcd $(13, 120) = 1$.

➢ **Step – 5:** Calculate $d = e^{-1} \bmod \Phi(n)$ or $ed = 1 \bmod \Phi(n)$.

❖**Explanation with example:**

5. Finding d:

➔ $e * d \bmod \Phi(n) = 1$

➔ $13 * d \bmod 120 = 1$

(How to find: $d * e = 1 \bmod \Phi(n)$ ➔ $d = ((\Phi(n) * i) + 1) / e$

$d = (120 + 1) / 13 = 9.30$ ($\because i = 1$)

$d = (240 + 1) / 13 = 18.53$ ($\because i = 2$)

$d = (360 + 1) / 13 = 27.76$ ($\because i = 3$)

$d = (480 + 1) / 13 = 37$ ($\because i = 4$))

➢ **Step – 6:** Public key = {e, n}, private key = {d, n}.

➢ **Step – 7:** Find out cipher text using the formula,

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

C = Cipher text, P = Plain text, e = Encryption key and n=block size.

➢ **Step – 8:** $P = C^d \bmod n$. Plain text P can be obtain using the given formula.

where, d = decryption key.

❖**Explanation with example:**

6. Public key = {13, 143} and private key = {37, 143}.

7. **Encryption :** Plain text ***P = 13***. (where, $P < n$)

$C = P^e \bmod n = 13^{13} \bmod 143 = 52$. ***C = 52***

8. **Decryption:**

$P = C^d \bmod n = 52^{37} \bmod 143 = 13$. ***P = 13***

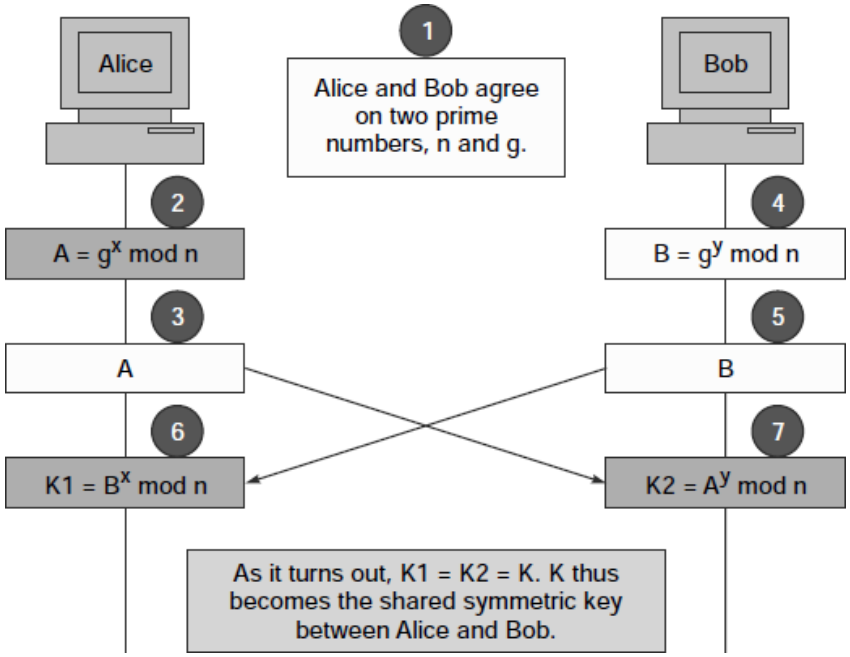
Diffie–Hellman Key Exchange/Agreement Algorithm

In this scheme the two parties, who want to communicatesecurely, can agree on a **symmetric key** using this technique. This key can then be used for encryption/ decryption. However, we must note that Diffie–Hellman key exchange algorithm can be used only forkey agreement, but not for encryption or decryption of messages. Once both the parties agree on the keyto be used, they need to use other symmetric key encryption algorithms for actual encryption or decryption of messages.

Description of the Algorithm

1. Firstly, Alice and Bob agree on two large prime numbers, n and g. These two integers need not be kept secret. Alice and Bob can use an insecure channel to agree on them.
2. Alice chooses another large random number x, and calculates A such that:
   $A = g^x \bmod n$
3. Alice sends the number A to Bob.
4. Bob independently chooses another large random integer y and calculates B such that:
   $B = g^y \bmod n$
5. Bob sends the number B to Alice.
6. A now computes the secret key K1 as follows:
   $K1 = B^x \bmod n$
7. B now computes the secret key K2 as follows:
   $K2 = A^y \bmod n$

Diffie–Hellman key exchange algorithm



Example of the Algorithm

1. Firstly, Alice and Bob agree on two large prime numbers, n and g. These two integers need not be kept secret. Alice and Bob can use an insecure channel to agree on them.

   Let n = 11, g = 7.

2. Alice chooses another large random number x, and calculates A such that:
   $A = g^x \bmod n$

   Let x = 3. Then, we have, $A = 7^3 \bmod 11 = 343 \bmod 11 = 2$.

3. Alice sends the number A to Bob.

   Alice sends 2 to Bob.

4. Bob independently chooses another large random integer y and calculates B such that:
   $B = g^y \bmod n$

   Let y = 6. Then, we have, $B = 7^6 \bmod 11 = 117649 \bmod 11 = 4$.

5. Bob sends the number B to Alice.

   Bob sends 4 to Alice.

6. A now computes the secret key K1 as follows:
   $K1 = B^x \bmod n$

   We have, $K1 = 4^3 \bmod 11 = 64 \bmod 11 = 9$.

7. B now computes the secret key K2 as follows:
   $K2 = A^y \bmod n$

   We have, $K2 = 2^6 \bmod 11 = 64 \bmod 11 = 9$.

ELAGAMAL CRYPTOGRAPHY (ECC)

In this ECC we have three phases

1. Key generation
2. Encryption
3. Decryption

1. Generate a random integer X_A, such that $1 < X_A < q - 1$.
2. Compute $Y^A = \alpha^{X_A} \bmod q$.
3. A's private key is X_A; A's pubic key is $\{q, \alpha, Y_A\}$.

Any user B that has access to A's public key can encrypt a message as follows:

1. Represent the message as an integer M in the range $0 \leq M \leq q - 1$. Longer messages are sent as a sequence of blocks, with each block being an integer less than q.
2. Choose a random integer k such that $1 \leq k \leq q - 1$.
3. Compute a one-time key $K = (Y_A)^k \bmod q$.
4. Encrypt M as the pair of integers (C_1, C_2) where

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

User A recovers the plaintext as follows:

1. Recover the key by computing $K = (C_1)^{X_A} \bmod q$.
2. Compute $M = (C_2 K^{-1}) \bmod q$.

ElGamal process as follows,

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

Key Generation by Alice	
Select private X_A	$X_A < q - 1$
Calculate Y_A	$Y_A = \alpha^{XA} \bmod q$
Public key	$PU = \{q, \alpha, Y_A\}$
Private key	X_A

Encryption by Bob with Alice's Public Key	
Plaintext:	$M < q$
Select random integer k	$k < q$
Calculate K	$K = (Y_A)^k \bmod q$
Calculate C_1	$C_1 = \alpha^k \bmod q$
Calculate C_2	$C_2 = KM \bmod q$
Ciphertext:	(C_1, C_2)

Decryption by Alice with Alice's Private Key	
Ciphertext:	(C_1, C_2)
Calculate K	$K = (C_1)^{X_A} \bmod q$
Plaintext:	$M = (C_2 K^{-1}) \bmod q$

Thus, K functions as a one-time key, used to encrypt and decrypt the message.

For example, let us start with the prime field GF(19); that is, $q = 19$. It has primitive roots $\{2, 3, 10, 13, 14, 15\}$, as shown in Table 8.3. We choose $\alpha = 10$.

Alice generates a key pair as follows:

1. Alice chooses $X_A = 5$.
2. Then $Y_A = \alpha^{X_A} \bmod q = \alpha^5 \bmod 19 = 3$ (see Table 8.3).
3. Alice's private key is 5; Alice's pubic key is $\{q, \alpha, Y_A\} = \{19, 10, 3\}$.

Suppose Bob wants to send the message with the value $M = 17$. Then,

1. Bob chooses $k = 6$.
2. Then $K = (Y_A)^k \bmod q = 3^6 \bmod 19 = 729 \bmod 19 = 7$.
3. So
   $C_1 = \alpha^k \bmod q = \alpha^6 \bmod 19 = 11$
   $C_2 = KM \bmod q = 7 \times 17 \bmod 19 = 119 \bmod 19 = 5$
4. Bob sends the ciphertext $(11, 5)$.

For decryption:

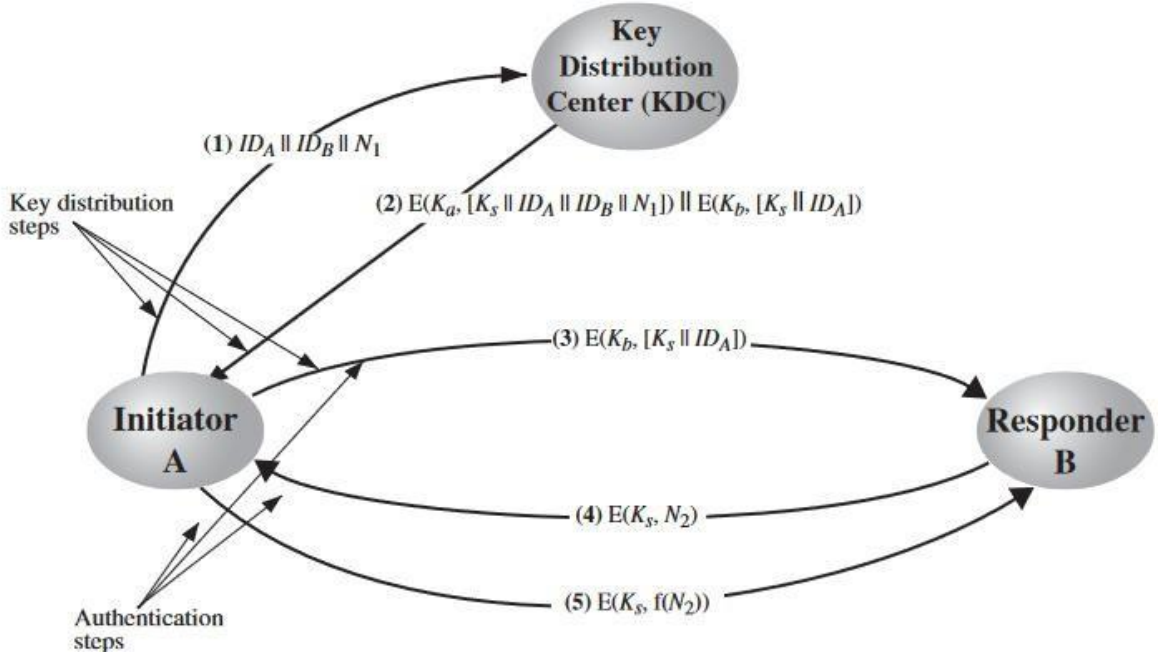
1. Alice calculates $K = (C_1)^{X_A} \bmod q = 11^5 \bmod 19 = 161051 \bmod 19 = 7$.
2. Then K^{-1} in GF(19) is $7^{-1} \bmod 19 = 11$.
3. Finally, $M = (C_2 K^{-1}) \bmod q = 5 \times 11 \bmod 19 = 55 \bmod 19 = 17$.

KEY DISTRIBUTION

- Key distribution is the function that delivers a key to two parties who wish to exchange secure encrypted data. Some sort of mechanism or protocol is needed to provide for the secure distribution of keys.
- Key distribution often involves the use of master keys, which are infrequently used and are long lasting, and session keys, which are generated and distributed for temporary use between two parties.
- Public-key encryption schemes are secure only if the authenticity of the public key is assured. A public-key certificate scheme provides the necessary security.
- X.509 defines the format for public-key certificates. This format is widely used in a variety of applications.

- A public-key infrastructure (PKI) is defined as the set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates based on asymmetric cryptography.
- Typically, PKI implementations make use of X.509 certificates

A Key Distribution Scenario



User A wishes to establish a logical connection with B andrequires a one-time session key to protect the data transmitted over the connection.A has a master key, K_a, known only to itself and the KDC; similarly, B shares themaster key $K_b$with the KDC.

Hierarchical Key Control

It is not necessary to limit the key distribution function to a single KDC. Indeed, forvery large networks, it may not be practical to do so. As an alternative, a hierarchy ofKDCs can be established. For example, there can be local KDCs, each responsiblefor a small domain of the overall internetwork, such as a single LAN or a singlebuilding.

A hierarchical scheme minimizes the effort involved in master key distribution, because most master keys are those shared by a local KDC with its local entities. Furthermore, such a scheme limits the damage of a faulty or subverted KDC to its local area only.

Session Key Lifetime

The more frequently session keys are exchanged, the more secure they are, becausethe opponent has less ciphertext to work with for any given session key. On theother hand, the distribution of session keys delays the start of any exchange andplaces a burden on network capacity. A security manager must try to balance thesecompeting considerations in determining the lifetime of a particular session key.

Decentralized Key Control

The use of a key distribution center imposes the requirement that the KDC be trusted and be protected from subversion. This requirement can be avoided if keydistribution is fully decentralized. Although full decentralization is not practical forlarger networks using symmetric encryption only, it may be useful within a localcontext.

A decentralized approach requires that each end system be able to communicate in a secure manner with all potential partner end systems for purposes of session key distribution. Thus, there may need to be as many as n(n-1)/2 master keys for a configuration with n end systems.

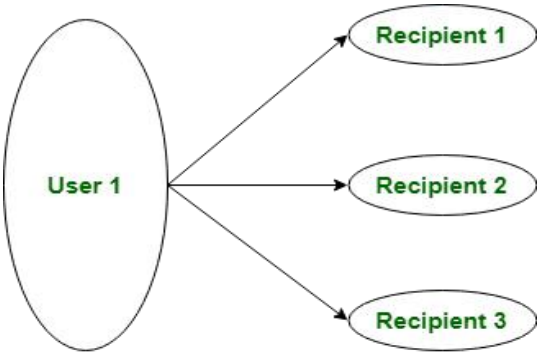
Distribution of Public Key:

The public key can be distributed in four ways:

1. Public announcement
2. Publicly available directory
3. Public-key authority
4. Public-key certificates.

These are explained as following below:

1. Public Announcement: Here the public key is broadcasted to everyone. The major weakness of this method is a forgery. Anyone can create a key claiming to be someone else and broadcast it. Until forgery is discovered can masquerade as claimed user.



Public Key Announcement

2. Publicly Available Directory: In this type, the public key is stored in a public directory. Directories are trusted here, with properties like Participant Registration, access and allow to modify values at any time, contains entries like {name, public-key}. Directories can be accessed electronically still vulnerable to forgery or tampering.

3. Public Key Authority: It is similar to the directory but, improves security by tightening control over the distribution of keys from the directory. It requires users to know the public key for the directory. Whenever the keys are needed, real-time access to the directory is made by the user to obtain any desired public key securely.

4. Public Certification: This time authority provides a certificate (which binds an identity to the public key) to allow key exchange without real-time access to the public authority each time. The certificate is accompanied by some other info such as period of validity, rights of use, etc. All of this content is signed by the private key of the certificate authority and it can be verified by anyone possessing the authority's public key.

UNIT-3

MESSAGE AUTHENTICATION ALGORITHMS AND HASH FUNCTIONS

Message authentication is a mechanism or service used to verify theintegrity of a message. Message authentication assures that data receivedare exactly as sent by (i.e., contain no modification, insertion, deletion, orreplay) and that the purported identity of the sender is valid.

AUTHENTICATION REQUIREMENTS

In the context of communications across a network, the following attacks can beidentified.

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

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

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

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

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

6. Timing modification: Delay or replay of messages. 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.

7. Source repudiation: Denial of transmission of message by source.

8. Destination repudiation: Denial of receipt of message by destination.

MESSAGE AUTHENTICATION FUNCTIONS

Any message authentication or digital signature mechanism has two levels of functionality. At the lower level, there must be some sort of function that produces anauthenticator: a value to be used to authenticate a message.

These may be grouped into three classes.

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

MessageEncryption

Messageencryptionbyitselfcanprovideameasureofauthentication.Theanaly sis differsforsymmetricandpublic-keyencryptionschemes.

SYMMETRICENCRYPTION: A message *M* transmitted from source *A* to destination B is encryptedusingasecretkey*K*sharedby*A*and*B*.Ifnootherpartyknowsthe key, then confidentiality is provided: No other party can recover the plaintext of the message.

So we may say that symmetric encryption provides authentication as well as confidentiality.

Given a decryption function D and a secret key *K*, the destination will accept *any* input *X* and produce output *Y* = *D*(*K*, *X*). If *X* is the ciphertext of a legitimate message *M* produced by the corresponding encryption function, then *Y* is some plaintext message *M*. Otherwise, *Y* will likely be a meaningless sequence of bits. There may need to be some automated means of determining at B whether *Y* is legitimate plaintext and therefore must have come from*A*.

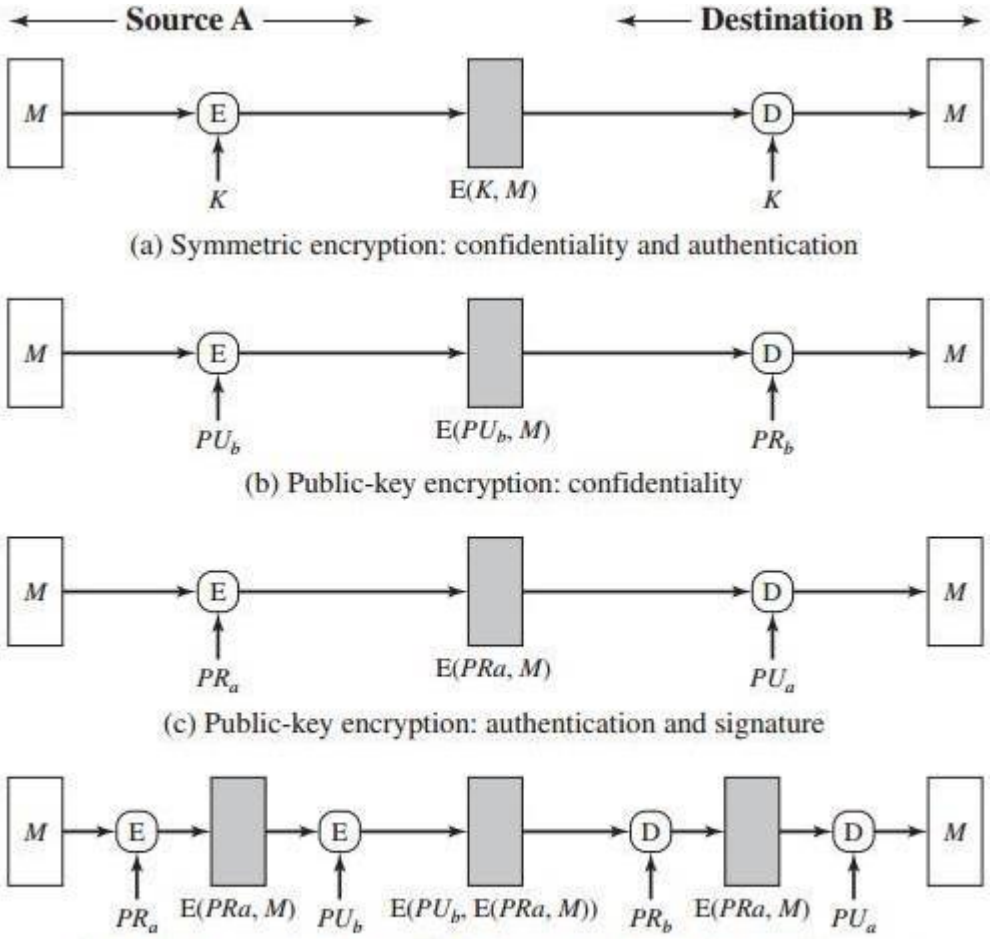


Figure 12.1 Basic Uses of Message Encryption

The implications of the line of reasoning in the preceding paragraph are profoundfromthepointofviewofauthentication.Supposethemessage*M*canb eany arbitrarybitpattern.Inthatcase,thereisnowaytodetermineautomatically,a tthe destination,whetheranincomingmessageisthecipher textofalegitimatemess age. This conclusion is incontrovertible: If *M* can be any bit pattern, then regardless of the value of *X*, the value *Y* = D(*K*, *X*) is *some* bit pattern and therefore must be accepted as authenticplaintext.

Thus, in general, we require that only a small subset of all possible bit patterns be considered legitimate plaintext. In that case, any spurious

ciphertext is unlikely to produce legitimate plaintext. For example, suppose that only one bit pattern in 10^6 is legitimate plaintext. Then the probability that any randomly chosen bit pattern, treated as ciphertext, will produce a legitimate plaintext message isonly10^{-6}.

For a number of applications and encryption schemes, the desiredconditions prevailasamatterofcourse.Forexample,supposethatwearetransmittingEnglish- language messages using a Caesar cipher with a shift of one (*K* = 1). *A* sends the following legitimateciphertext:

Nbsftfbupbutboeepftfbupbutboemjuumfmbnctfbujwz

B decrypts to produce the following plaintext:

mareseatoatsanddoeseatoatsandlittlelambseativy

Asimplefrequencyanalysisconfirmsthatthismessagehastheprofileofordinary English. On the other hand, if an opponent generates the following random sequence ofletters:

zuvrsoevgqxlzwigamdvnmhpmccxiuureosfbcebtqxsxq

this decrypts to

ytuqrndufpwkyvhfzlcumlgolbbwhttqdnreabdaspwrwp

which does not fit the profile of ordinary English.Itmaybedifficulttodetermine*automatically*ifincomingciphertextdecrypts tointelligibleplaintext.Iftheplaintextis,say,abinaryobjectfileordigitizedX-rays, determination of properly formed and therefore authentic plaintext may be diffi-

cult.Thus,anopponentcouldachieveacertainlevelofdisruptionsimplybyissuin g messageswithrandomcontentpurportingtocomefromalegitimateuser.

One solution to this problem is to force the plaintext to have some structure that is easily recognized but that cannot be replicated without recourse to the encryption function. We could, for example, append an error-detecting code, also known as a frame check sequence (FCS) or checksum, to each message before encryption. A prepares a plaintext message M and then provides this as input to a function F that produces an FCS. The FCS is appended to M and the entire block is then encrypted. At the destination, B

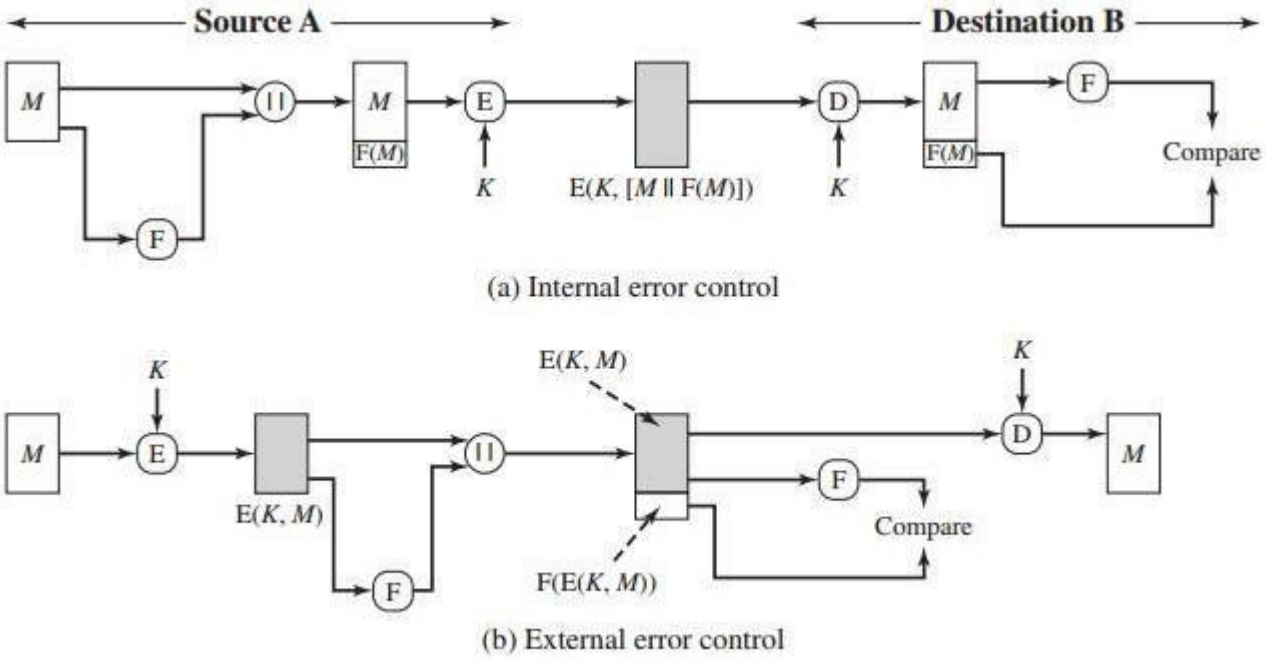


Figure 12.2 Internal and External Error Control

decrypts the incoming block and treats the results as a message withanappendedFCS. B applies the same function F to attempt to reproduce the FCS. IfthecalculatedFCSisequaltotheincomingFCS,thenthemessageisconsidere

dauthentic.Itisunlikelythatanyrandomsequenceofbitswouldexhibitthedesi redrelationship.

NotethattheorderinwhichtheFCSandencryptionfunctionsareperformed is critical. With internal error control, authentication is provided because an opponentwouldhavedifficultygeneratingciphertextthat,whendecrypted,w ould have valid error control bits. If instead the FCS is the outer code, an opponent can construct messages with valid error-control codes. Although the opponent cannot knowwhatthedecryptedplaintextwillbe,heorshecanstillhopetocreateconfu - sion and disruptoperations.

An error-control code is just one example; in fact, any sort of structuring addedtothetransmittedmessageservestostrengthentheauthenticationcap ability.
Suchstructureisprovidedbytheuseofacommunicationsarchitectureconsisti ngof layered protocols.

PUBLIC-KEY ENCRYPTION The straightforward use of public-key encryption providesconfidentialitybutnotauthentication.Thesource(*A*)uses thepublickey*PUb*ofthedestination(B)toencrypt*M*.BecauseonlyBhasthecorr esponding private key *PRb*, only B can decrypt the message. This scheme providesnoauthentication,becauseanyopponentcouldalsouseB'spublickeyto encryptamessageandclaimtobe*A*.

Toprovideauthentication,*A*usesitsprivatekeytoencryptthemessage,andB uses*A*'spublickeytodecrypt. Thisprovidesauthenticationusingthe sametypeofreasoningasinthesymmetricencryptioncase:Themessagemusth

ave comefromAbecauseAistheonlypartythatpossessesPR_aandthereforetheonly party with the information necessary to construct ciphertext that can be decrypted withPU_a.Again,thesamereasoningasbeforeapplies:Theremustbesomeinternal structure to the plaintext so that the receiver can distinguish between well-formed plaintext and randombits.

Assumingthereissuchstructure,thentheschemeofFigure12.1cdoesprovide authentication. It also provides what is known as digital signature.[1] Only A could have constructed the ciphertext because only A possesses PR_a. Not even B, the recipient,couldhaveconstructedtheciphertext.Therefore,ifBisinpossessionof theciphertext,Bhasthemeanstoprovethatthemessagemusthavecomefrom A. Ineffect,Ahas"signed"themessagebyusingitsprivatekeytoencrypt.Noteth at thisschemedoesnotprovideconfidentiality.AnyoneinpossessionofA'spublick ey can decrypt theciphertext.

To provide both confidentiality and authentication, A can encrypt M first usingitsprivatekey,whichprovidesthedigitalsignature,andthenusingB'spubli c key, which provides confidentiality. The disadvantage of this approachisthatthepublic-keyalgorithm,whichiscomplex,mustbeexercisedfour timesratherthantwoineachcommunication.

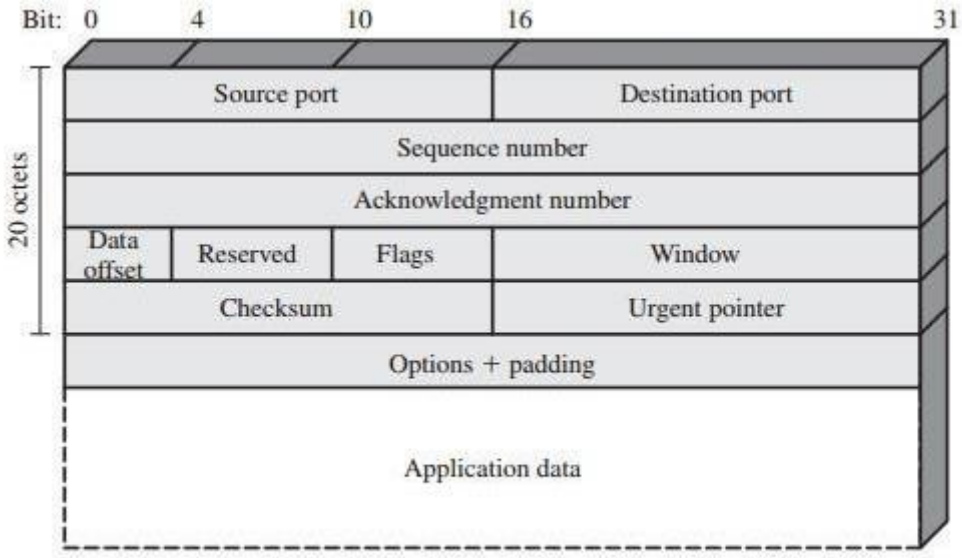


Figure 12.3 TCP Segment

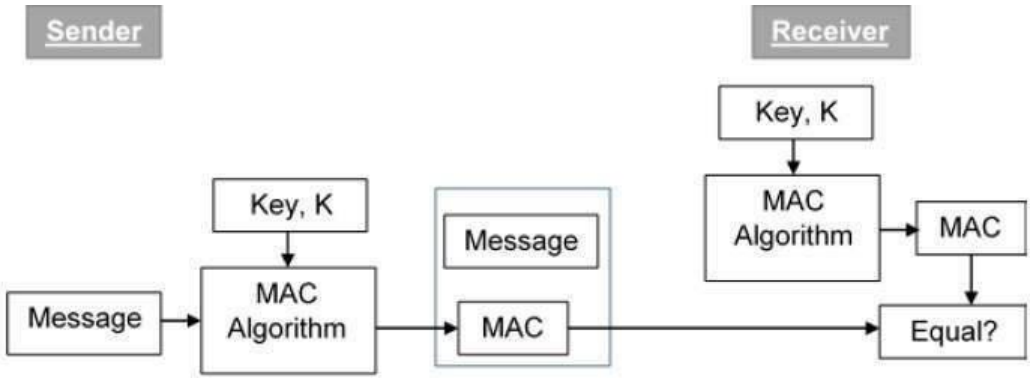
As an example, consider the structure of messages transmitted usingtheTCP/IPprotocolarchitecture.The above figureshowstheformatofaTCPsegment, illustrating the TCP header. Now suppose that each pair of hosts shared a unique secret key, so that all exchanges between a pair of hosts used the samekey, regardlessofapplication.Thenwecouldsimplyencryptallofthedatagramexcept the IP header. Again, if an opponent substituted some arbitrary bit pattern for the encrypted TCP segment, the resulting plaintext would not include a meaningful header. In this case, the header includes not only a checksum (which covers the header) but also other useful information, such as the sequence number. Because successiveTCPsegmentsonagivenconnectionarenumberedsequentially,encryptionassuresthatanopponentdoesnotdelay,misorder,ordeleteanysegments.

Message Authentication Code (MAC)

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

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

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



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

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

Limitations of MAC

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

- **Establishment of Shared Secret.**

- It can provide message authentication among pre-decided legitimate users who have shared key.
- This requires establishment of shared secret prior to use of MAC.

- **Inability to Provide Non-Repudiation**

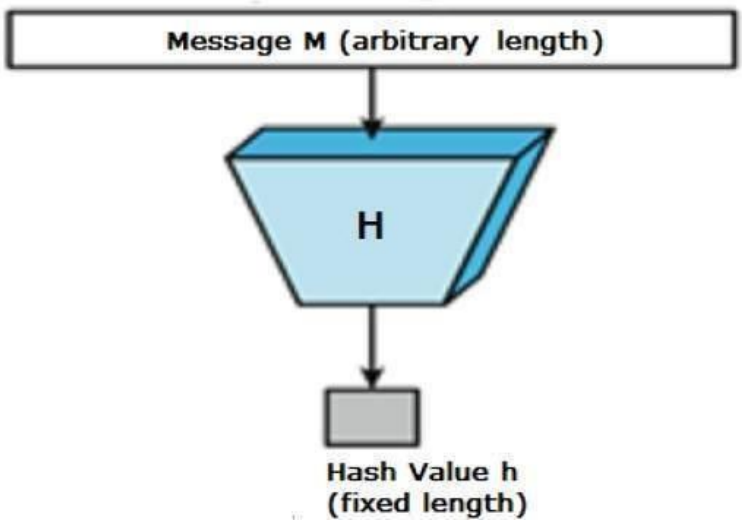
  - Non-repudiation is the assurance that a message originator cannot deny any previously sent messages and commitments or actions.
  - MAC technique does not provide a non-repudiation service. If the sender and receiver get involved in a dispute over message origination, MACs cannot provide a proof that a message was indeed sent by the sender.
  - Though no third party can compute the MAC, still sender could deny having sent the message and claim that the receiver forged it, as it is impossible to determine which of the two parties computed the MAC.

Both these limitations can be overcome by using the public key based digital signatures discussed in following section.

HASH FUNCTIONS

A hash function is a mathematical function that converts a numerical input value into another compressed numerical value. The input to the hash function is of arbitrary length but output is always of fixed length.

Values returned by a hash function are called **message digest** or simply **hash values**. The following picture illustrated hash function –



Features of Hash Functions

The typical features of hash functions are –

- **Fixed Length Output (Hash Value)**
  - Hash function coverts data of arbitrary length to a fixed length. This process is often referred to as **hashing the data**.
  - In general, the hash is much smaller than the input data, hence hash functions are sometimes called **compression functions**.
  - Since a hash is a smaller representation of a larger data, it is also referred to as a **digest**.
  - Hash function with n bit output is referred to as an **n-bit hash function**. Popular hash functions generate values between 160 and 512 bits.
- **Efficiency of Operation**
  - Generally for any hash function h with input x, computation of h(x) is a fast operation.
  - Computationally hash functions are much faster than a symmetric encryption.

Properties of Hash Functions

In order to be an effective cryptographic tool, the hash function is desired to possess following properties –

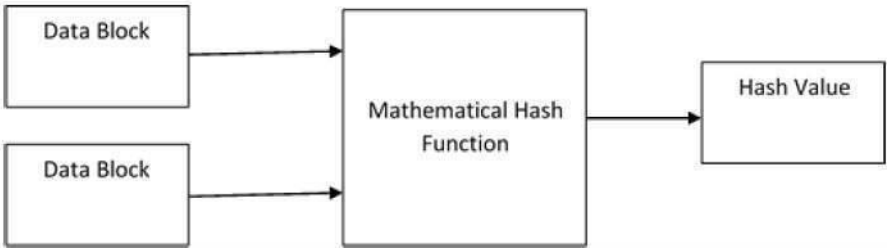
- **Pre-Image Resistance**
  - This property means that it should be computationally hard to reverse a hash function.
  - In other words, if a hash function h produced a hash value z, then it should be a difficult process to find any input value x that hashes to z.
  - This property protects against an attacker who only has a hash value and is trying to find the input.
- **Second Pre-Image Resistance**
  - This property means given an input and its hash, it should be hard to find a different input with the same hash.
  - In other words, if a hash function h for an input x produces hash value h(x), then it should be difficult to find any other input value y such that h(y) = h(x).
  - This property of hash function protects against an attacker who has an input value and its hash, and wants to substitute different value as legitimate value in place of original input value.
- **Collision Resistance**
  - This property means it should be hard to find two different inputs of any length that result in the same hash. This property is also referred to as collision free hash function.

- In other words, for a hash function h, it is hard to find any two different inputs x and y such that h(x) = h(y).
- Since, hash function is compressing function with fixed hash length, it is impossible for a hash function not to have collisions. This property of collision free only confirms that these collisions should be hard to find.
- This property makes it very difficult for an attacker to find two input values with the same hash.
- Also, if a hash function is collision-resistant **then it is second pre-image resistant.**

Design of Hashing Algorithms

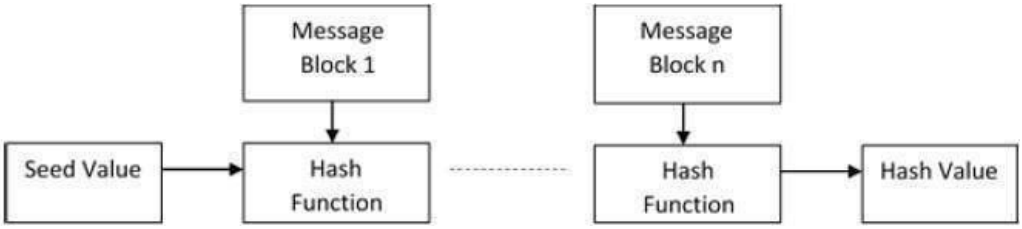
At the heart of a hashing is a mathematical function that operates on two fixed-size blocks of data to create a hash code. This hash function forms the part of the hashing algorithm.

The size of each data block varies depending on the algorithm. Typically the block sizes are from 128 bits to 512 bits. The following illustration demonstrates hash function –



Hashing algorithm involves rounds of above hash function like a block cipher. Each round takes an input of a fixed size, typically a combination of the most recent message block and the output of the last round.

This process is repeated for as many rounds as are required to hash the entire message. Schematic of hashing algorithm is depicted in the following illustration –



Since, the hash value of first message block becomes an input to the second hash operation, output of which alters the result of the third operation, and so on. This effect, known as an **avalanche** effect of hashing.

Avalanche effect results in substantially different hash values for two messages that differ by even a single bit of data.

Understand the difference between hash function and algorithm correctly. The hash function generates a hash code by operating on two blocks of fixed-length binary data.

Hashing algorithm is a process for using the hash function, specifying how the message will be broken up and how the results from previous message blocks are chained together.

SECURE HASHING ALGORITHM

Secure Hashing Algorithm (SHA) is the cryptographic algorithm adopted for digital signatures. It produces a unique hash in an unreadable format. This is to make your data secure and unhackable.

Additionally, SHA uses MD5, SHA 1, or SHA 256 for symmetric cryptography. They generate hash values to encrypt and decrypt data securely.

Some of the SHA algorithms

SHA-0: A retronym applied to the original version of the 160-bit hash function published in 1993 under the name "SHA". It was withdrawn shortly after publication due to an undisclosed "significant flaw" and replaced by the slightly revised version SHA-1.

SHA-1: A 160-bit hash function which resembles the earlier MD5 algorithm. This was designed by the National Security Agency (NSA) to be part of the Digital Signature Algorithm. Cryptographic weaknesses were discovered in SHA-1, and the standard was no longer approved for most cryptographic uses after 2010.

SHA-2: A family of two similar hash functions, with different block sizes, known as SHA-256 and SHA-512. They differ in the word size; SHA-256 uses 32-bit words where SHA-512 uses 64-bit words. There are also truncated versions of each standard, known as SHA-224, SHA-384, SHA-512/224 and SHA-512/256. These were also designed by the NSA.

SHA-3: A hash function formerly called Keccak, chosen in 2012 after a public competition among non-NSA designers. It supports the same hash lengths as SHA-2, and its internal structure differs significantly from the rest of the SHA family.

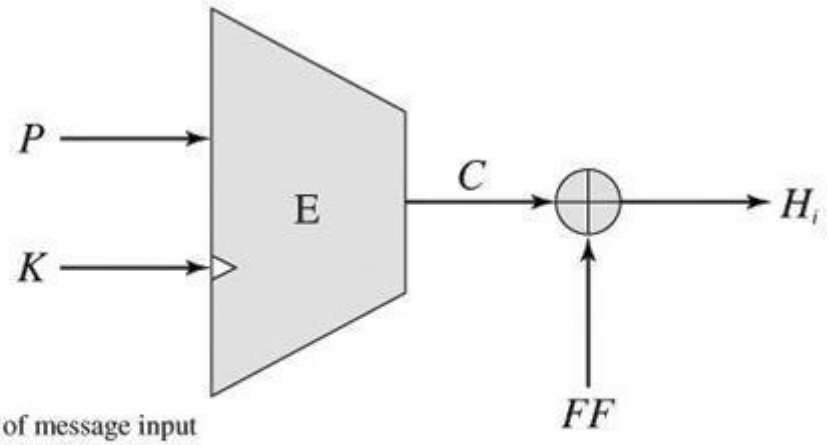
Difference between SHA1 and SHA2

SHA 1	SHA 2
SHA 1 was released in 1995.	SHA 2 was released recently in 2001.
It is the advanced version of SH0.	It is the advanced version of SHA1.
SHA 2 is SHA 1's upgraded algorithm.	SHA 3 is SHA 2's upgraded algorithm.
SHA 1 is a standalone Hash entity.	SHA 2 has many variations.
SHA 1 generates 160 bits hash value.	SHA 2 generates 224-, 256-, 384- or 512-bits hash values.
The length output value of SHA 1 is 40 digits.	The length output value of SHA 2 is 64 digits
SHA 1 is less secured when compared to SHA 2.	SHA 2 is more secured than SHA 1 but less secure than SHA 3.
SHA 1 certificates are not reliable.	SHA 2 is more reliable because of its improved certificates.
SHA 1 is not widely used.	SHA 2 Family is widely used today.

WHIRLPOOL HASH FUNCTION

The general iterated hash structure proposed by Merkle is used in virtually all secure hash functions. Preneelperformedia systematic analysis of block-cipher-based hash functions. In this model, the hash code length equals the cipher block length. Additional security problems are introduced and the analysis is more difficult if the hash code length exceeds the cipher block length. Preneel devised 64 possible permutations of the basic model, based on which input served as the encryption key and which served as plaintext and on what input, if any, was combined with the ciphertext to produce the intermediate hash code. Based on his analysis, he concluded that only schemes in which the plaintext was fed forward and combined with the ciphertext were secure.

Model of Single Iteration



m_i = ith block of message input
H_i = ith intermediate hash value
P = plaintext; K = encryption key; C = ciphertext
FF = feed forward value
P, K, and FF can be chosen from the set $(0, m_i, H_{i-1}, m_i \oplus H_{i-1})$

Whirlpool Logic:

Given a message consisting of a sequence of blocks m1, m2,…,mt the Whirlpool hash function is expressed as follows:

H_0 = initial value

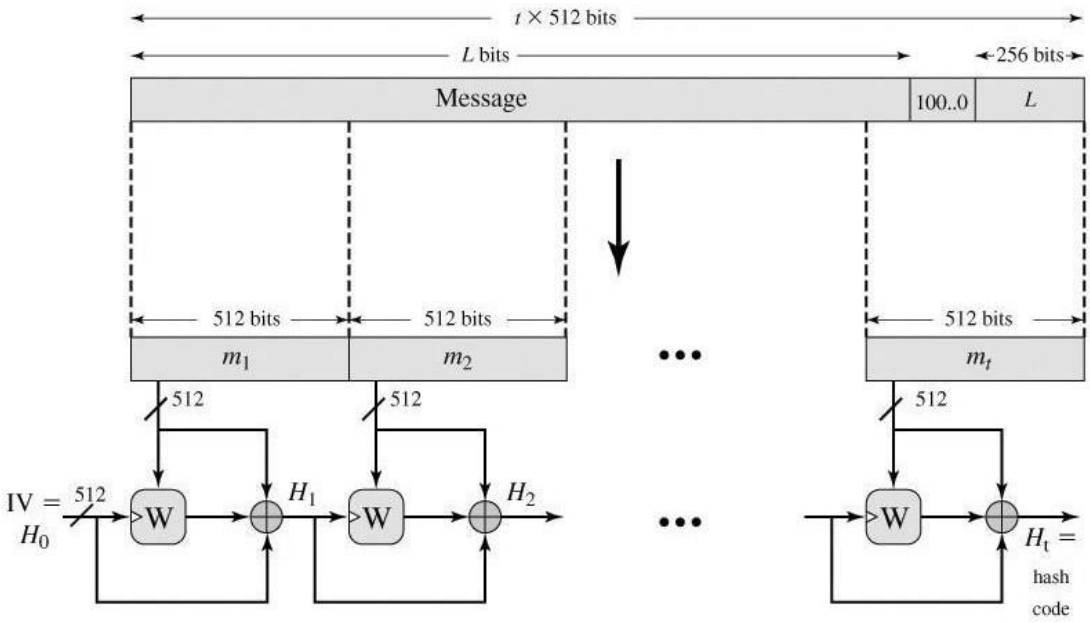
$H_i = E(H_{i-1}, m_i) \oplus H_{i-1} \oplus m_i$ = intermediate value

H_t = hash code value

The encryption key input for each iteration is the intermediate hash value from the previous iteration; the plaintext is the current message block; and the feed forward value is the bitwise XOR of the current message block and the intermediate hash value from the previous iteration.

The algorithm takes as input a message with a maximum length of less than 2^{256} bits and produces as output a 512-bit message digest. The input is processed in 512-bit blocks.. The processing consists of the following steps:

Message Digest Generation Using Whirlpool



Step 1:Append padding bits.The message is padded so that its length in bits is an odd multiple of 256. Padding is always added, even if the message is already of the desired length.

Step 2:Append length. A block of 256 bits is appended to the message. This block is treated as an unsigned 256-bit and contains the length in bits of the original message.

Step 3:Initialize hash matrix. An 8 x 8 matrix of bytes is used to hold intermediate and final results of the hash function. The matrix is initialized as consisting of all 0-bits.

Step 4: Process message in 512-bit (64-byte) blocks. The heart of the algorithm is the block cipher W.

HMAC

A hash function such as SHA was not designed for use as a MAC and cannot be used directly for that purpose because it does not rely on a secret key. There have been a number of proposals for the incorporation of a secret key into an existing hash algorithm. The approach that has received the most support is HMAC. HMAC has been issued as RFC 2104, has been chosen as the mandatory-to-implement MAC for IP security, and is used in other Internet protocols, such as SSL.

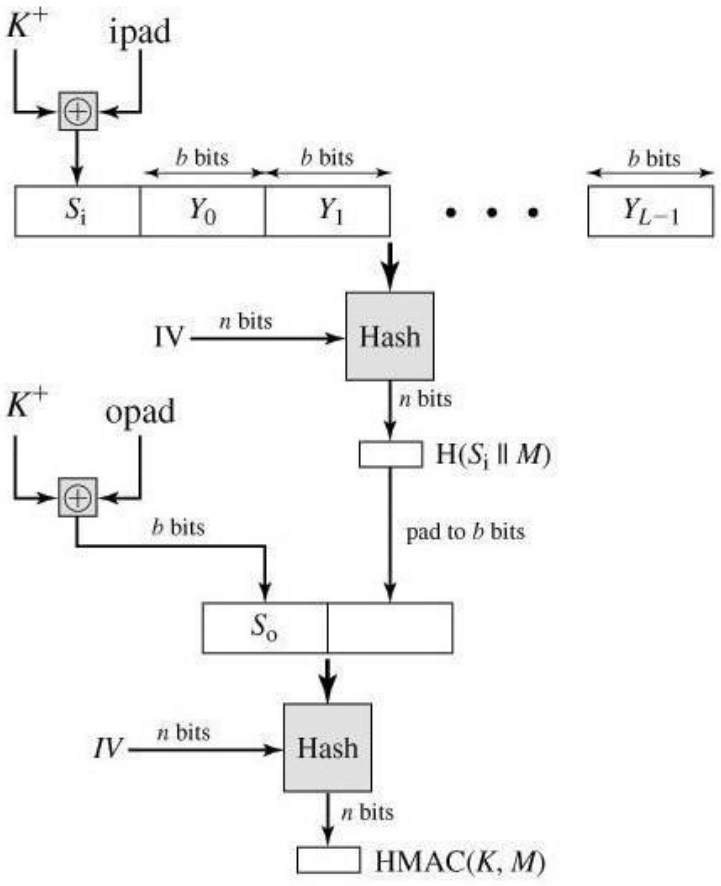
The following design objectives for HMAC:-

- To use, without modifications, available hash functions. In particular, hash functions that perform well in software, and for which code is freely and widely available.
- To allow for easy replace ability of the embedded hash function in case faster or more secure hash functions are found or required.

- To preserve the original performance of the hash function without incurring a significant degradation.
- To use and handle keys in a simple way.
- To have a well understood cryptographic analysis of the strength of the authentication mechanism based on reasonable assumptions about the embedded hash function.

The first two objectives are important to the acceptability of HMAC. HMAC treats the hash function as a "black box." This has two benefits. First, an existing implementation of a hash function can be used as a module in implementing HMAC. In this way, the bulk of the HMAC code is prepackaged and ready to use without modification. Second, if it is ever desired to replace a given hash function in an HMAC implementation, all that is required is to remove the existing hash function module and drop in the new module. This could be done if a faster hash function were desired.

The last design objective in the preceding list is, in fact, the main advantage of HMAC over other proposed hash-based schemes. HMAC can be proven secure provided that the embedded hash function has some reasonable cryptographic strength.



HMAC Algorithm

H	= embedded hash function (e.g., MD5, SHA-1, RIPEMD-160)
IV	= initial value input to hash function
M	= message input to HMAC(including the padding specified in the embedded hash function)
Y_i	= ith block of M, $0 \leq i \leq (L - 1)$
L	= number of blocks in M
b	= number of bits in a block
n	= length of hash code produced by embedded hash function
K	= secret key recommended length is $\geq n$; if key length is greater than b; the key is input to the hash function to produce an n-bit key
K^+	= K padded with zeros on the left so that the result is b bits in length
ipad	= 00110110 (36 in hexadecimal) repeated $b/8$ times
opad	= 01011100 (5C in hexadecimal) repeated $b/8$ times

Then HMAC can be expressed as follows:

HMAC (K, M) = H [(K⊕opad)||H[(K⊕ipad)||M]]

In words,

1. Append zeros to the left end of K to create a b-bit string K(e.g., if K is of length 160 bits and b = 512 then K will be appended with 44 zero bytes 0 x 00).
2. XOR (bitwise exclusive-OR) Kwith ipad to produce the b-bit block S_i.
3. Append M to S_i.
4. Apply H to the stream generated in step 3.
5. XOR Kwith opad to produce the b-bit block S_o
6. Append the hash result from step 4 to S_o
7. Apply H to the stream generated in step 6 and output the result.

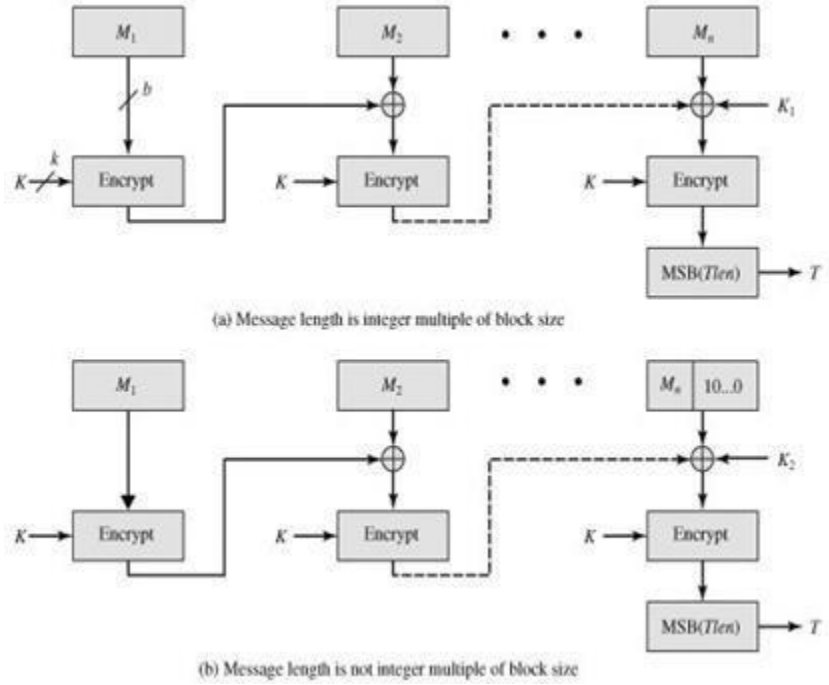
The XOR with ipad results in flipping one-half of the bits of K. Similarly, the XOR with opad results in flipping one-half of the bits of K, but a different set of bits. In effect, by passing S_i and S_o through the compression function of the hash algorithm, we have pseudorandomly generated two keys from K.

CMAC

The Data Authentication Algorithm defined in FIPS PUB 113, also known as the CBC-MAC (cipher block chaining message authentication code). This cipher-based MAC has been widely adopted in government and industry.MAC is secure under a reasonable set of security criteria, with the following restriction.

First, let us consider the operation of CMAC when the message is an integer multiple n of the cipher block length b. For AES, b = 128 and for triple DES, b=64. The message is divided into n blocks, M1, M2... Mn. The algorithm makes use of a k-bit encryption key K and an n-bit constant K1. For AES, the key size k is 128, 192, or 256 bits; for triple DES, the key size is 112 or 168 bits. CMAC is calculated as follows:

Cipher-Based Message Authentication Code (CMAC)



C1= E (K, M1)

C2 = E (K, [M2 ⊕C1])

C3 = E (K, [M3 ⊕C2])

.

.

.

$Cn = E(K, [Mn \oplus Cn1 \oplus K1])$

$T = MSB_{Tlen}(Cn)$

Where

T= message authentication code, also referred to as the tag

Tlen = bit length of T

$MSB_s(X)$ = the s leftmost bits of the bit string X

If the message is not an integer multiple of the cipher block length, then the final block is padded to the right (least significant bits) with a 1 and as many 0s as necessary so that the final block is also of length b. The CMAC operation then precedes as before, except that a different n-bit key K2 is used instead of K1.The two n-bit keys are derived from the k-bit encryption key as follows:

$L = E(K, 0n)$

$K1 = L \cdot x$

$K2 = L \cdot x2 = (L \cdot x) \cdot x$

where multiplication (·) is done in the finite field (2n) and x and x2 are first and second order polynomials that are elements of GF(2n) Thus the binary representation of x consists of n - 2 zeros followed by 10; the binary representation of x2 consists of n - 3 zeros followed by 100. The finite field is defined with respect to an irreducible polynomial that is lexicographically first among all such polynomials with the minimum possible number of nonzero terms. For the two approved block sizes, the polynomials are and x64 x4 x3 x 1 and x128 x7 x2 x 1. To generate K_1 and K_2 the block cipher is applied to the block that consists entirely of 0 bits.

DIGITAL SIGNATURE

A digital signature is an authentication mechanism that enables thecreator of a message to attach a code that acts as a signature. Typicallythe signature is formed by taking the hash of the message and encryptingthe message with the creator's private key. The signature guarantees thesource and integrity of the message.

Properties

Message authentication protects two parties who exchange messages from any thirdparty. However, it does not protect the two parties against each other. Several formsof dispute between the two are possible.

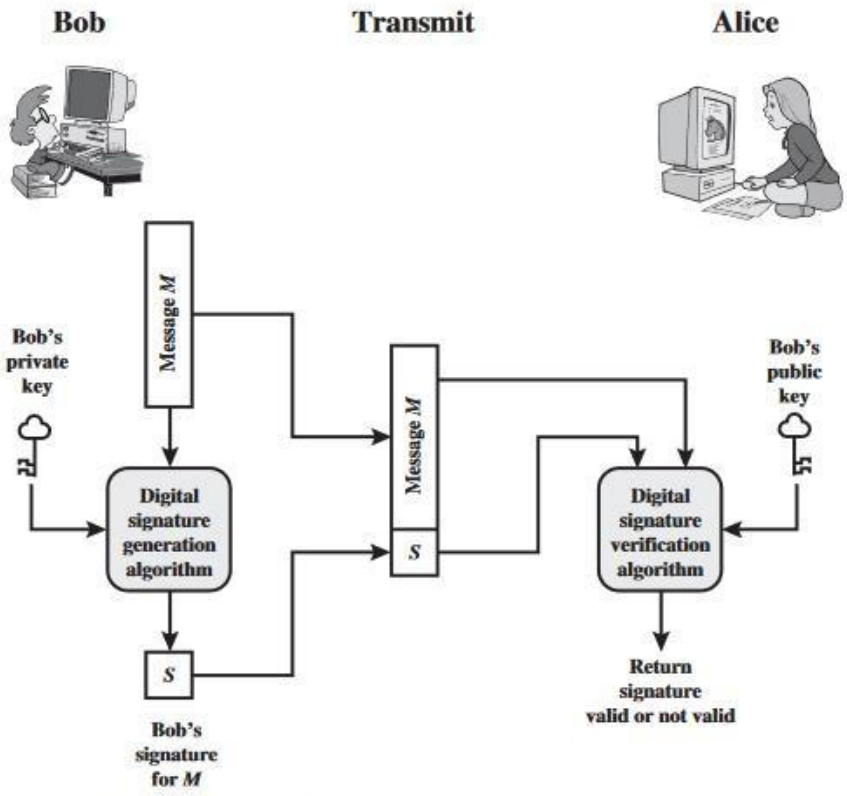


Figure 13.1 Generic Model of Digital Signature Process

In situations where there is not complete trust between sender and receiver,something more than authentication is needed. The most attractive solution tothis problem is the digital signature. The digital signature must have the followingproperties:

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

Thus, the digital signature function includes the authentication function

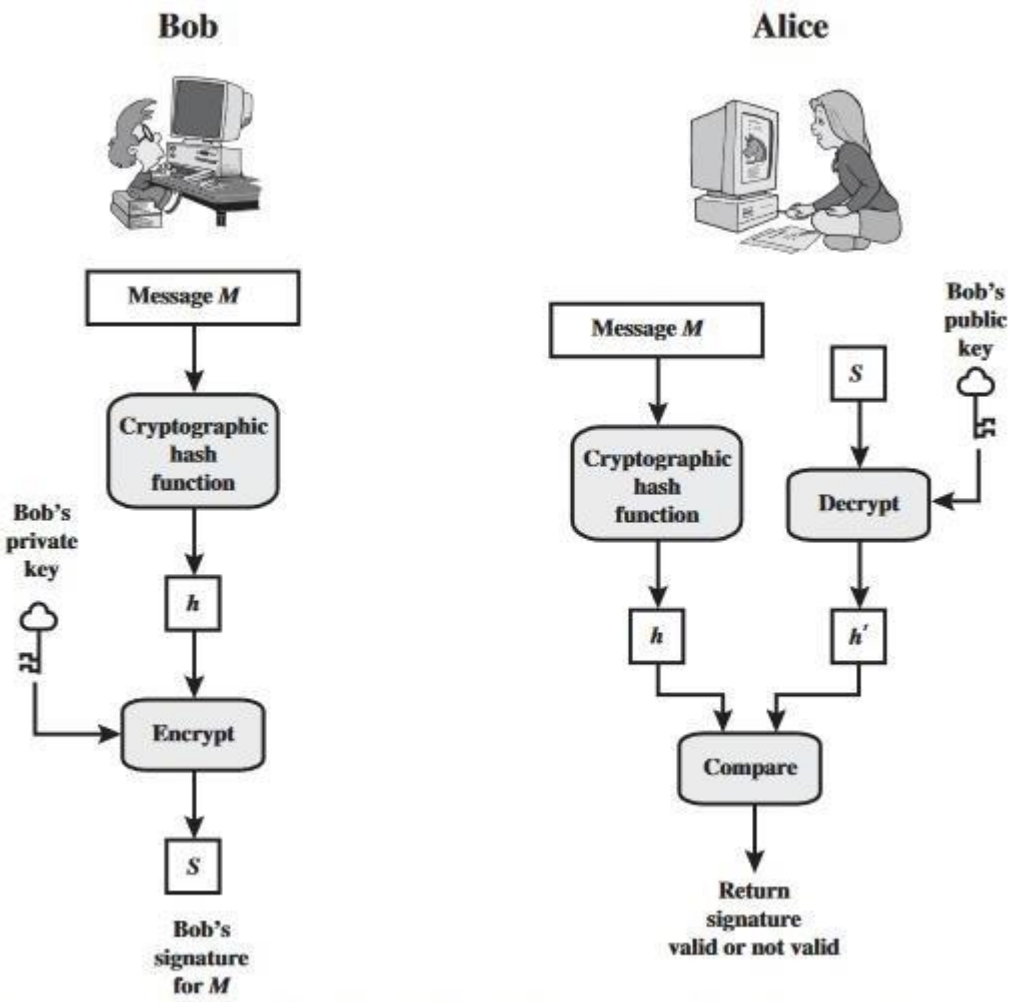


Figure 13.2 Simplified Depiction of Essential Elements of Digital Signature Process

Attacks and Forgeries

HereA denotes the user whose signature method is being attacked, and C denotes theattacker.

• **Key-only attack:** C only knows A's public key.

• **Known message attack:** C is given access to a set of messages and theirsignatures.

• **Generic chosen message attack:** C chooses a list of messages before attempting to breaks A's signature scheme, independent of A's public key. C thenobtains from A valid signatures for the chosen messages. The attack is generic,because it does not depend on A's public key; the same attack is used againsteveryone.

• **Directed chosen message attack:** Similar to the generic attack, except that thelist of messages to be signed is chosen after C knows A's public key but beforeany signatures are seen.

• **Adaptive chosen message attack:** C is allowed to use A as an "oracle." This means the A may request signatures of messages that depend on previously obtained message–signature pairs.

non-negligible probability:

- **Total break:** C determines A's private key.
- **Universal forgery:** C finds an efficient signing algorithm that provides an equivalent way of constructing signatures on arbitrary messages.
- **Selective forgery:** C forges a signature for a particular message chosen by C.
- **Existential forgery:** C forges a signature for at least one message. C has no control over the message. Consequently, this forgery may only be a minor nuisance to A.

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

Direct Digital Signature

The term direct digital signature refers to a digital signature scheme that involvesonly the communicating parties (source, destination). It is assumed that the destination knows the public key of the source.

Confidentiality can be provided by encrypting the entire message plus signature with a shared secret key (symmetric encryption). Note that it is important to perform the signature function first and then an outer confidentiality function. In case of dispute, some third party must view the message and its signature. If the signature is calculated on an encrypted message, then the third party also needs access to the decryption key to read the original message. However, if the

signature is the inner operation, then the recipient can store the plaintext message and its signature for later use in dispute resolution.

The validity of the scheme just described depends on the security of the sender's private key. If a sender later wishes to deny sending a particular message, the sender can claim that the private key was lost or stolen and that someone else forged his or her signature.

KNAPSACK ALGORITHM

Knapsack Encryption Algorithm is the first general public key cryptography algorithm. It is developed by Ralph Merkle and Mertin Hellman in 1978. As it is a Public key cryptography, it needs two different keys. One is Public key which is used for Encryption process and the other one is Private key which is used for Decryption process. In this algorithm we will use two different knapsack problems in which one is easy and other one is hard. The easy knapsack is used as the private key and the hard knapsack is used as the public key. The easy knapsack is used to derived the hard knapsack.

For the easy knapsack, we will choose a Super Increasing knapsack problem. Super increasing knapsack is a sequence in which every next term is greater than the sum of all preceding terms.

Example –

{1, 2, 4, 10, 20, 40} is a super increasing as

1<2, 1+2<4, 1+2+4<10, 1+2+4+10<20 and 1+2+4+10+20<40.

Derive the Public key

- **Step-1:**
  Choose a super increasing knapsack {1, 2, 4, 10, 20, 40} as the private key.

- **Step-2:**
  Choose two numbers n and m. Multiply all the values of private key by the number n and then find modulo m. The value of m must be greater than the sum of all values in private key, for example 110. And the number n should have no common factor with m, for example 31.

- **Step-3:**
  Calculate the values of Public key using m and n.

  1x31 mod(110) = 31
  2x31 mod(110) = 62
  4x31 mod(110) = 14
  10x31 mod(110) = 90
  20x31 mod(110) = 70
  40x31 mod(110) = 30

- Thus, our public key is {31, 62, 14, 90, 70, 30} And Private key is {1, 2, 4, 10, 20, 40}.

Now take an example for understanding the process of encryption and decryption.

Example –
Lets our plain text is 100100111100101110.

1. **Encryption :**
   As our knapsacks contain six values, so we will split our plain text in a groups of six:

   100100 111100 101110

   Multiply each values of public key with the corresponding values of each group and take their sum.

   100100 {31, 62, 14, 90, 70, 30}
   1x31+0x62+0x14+1x90+0x70+0x30 = 121

   111100 {31, 62, 14, 90, 70, 30}
   1x31+1x62+1x14+1x90+0x70+0x30 = 197

   101110 {31, 62, 14, 90, 70, 30}
   1x31+0x62+1x14+1x90+1x70+0x30 = 205

   So, our cipher text is 121 197 205.

2. **Decryption :**
   The receiver receive the cipher text which has to be decrypt. The receiver also know as the values of m and n.
   So, first we need to find the n^{-1} , which is multiplicative inverse of n mod m i.e.,

Gcd of 110 & 31 is
110=3(31)+17
31 =1(17) + 14
17 = 1(14) + 3
14 = 4(3) + 2
3 = 1(2) + 1

By Euclidean algorithm
1 = 3- 1(2)
1 = 5(3) -1(14)
1 = 11(14) –

9(17)

1 = 11(17) – 6(31)
1 = **71**(31) – 20 (110)

$n \times n^{-1} \bmod(m) = 1$

$31 \times n^{-1} \bmod(110) = 1$

$n^{-1} = 71$

Now, we have to multiply 71 with each block of cipher text take modulo m.

121 x 71 mod(110) = 11

Then, we will have to make the sum of 11 from the values of private key {1, 2, 4, 10, 20, 40} i.e.,
1+10=11 so make that corresponding bits 1 and others 0 which is 100100.
Similarly,

197 x 71 mod(110) = 17
1+2+4+10=17 = 111100

And, 205 x 71 mod(110) = 35
1+4+10+20=35 = 101110

After combining them we get the decoded text.
100100111100101110 which is our plain text.

UNIT – 4
E-Mail SECURITY

PRETTY GOOD PRIVACY

PGP is an open-source, freely available software package for e-mail security. It provides authentication through the use of digital signature, confidentiality through the use of symmetric block encryption, compressionusing the ZIP algorithm, and e-mail compatibility using the radix-64encoding scheme.

Notations:

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

618

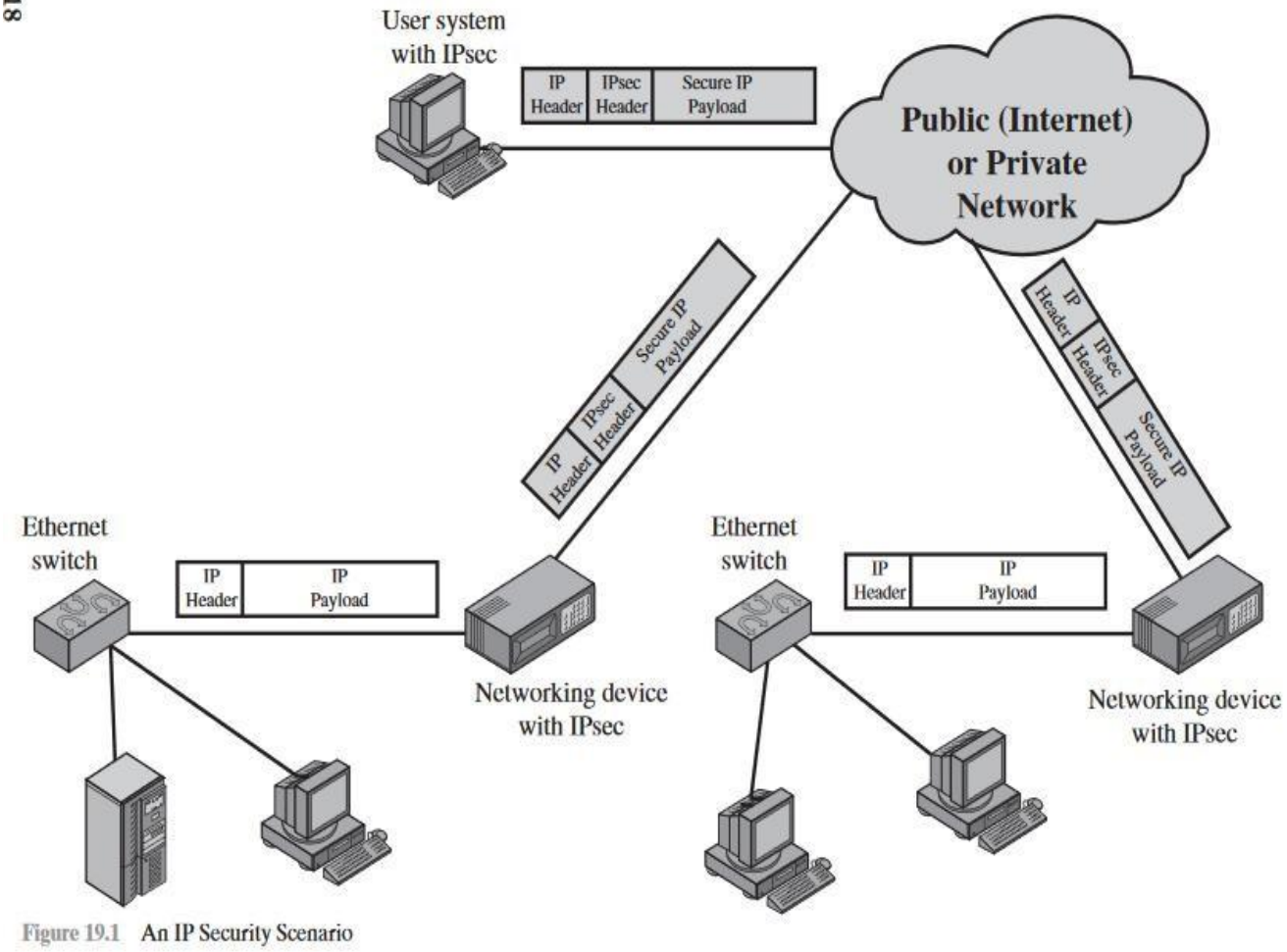
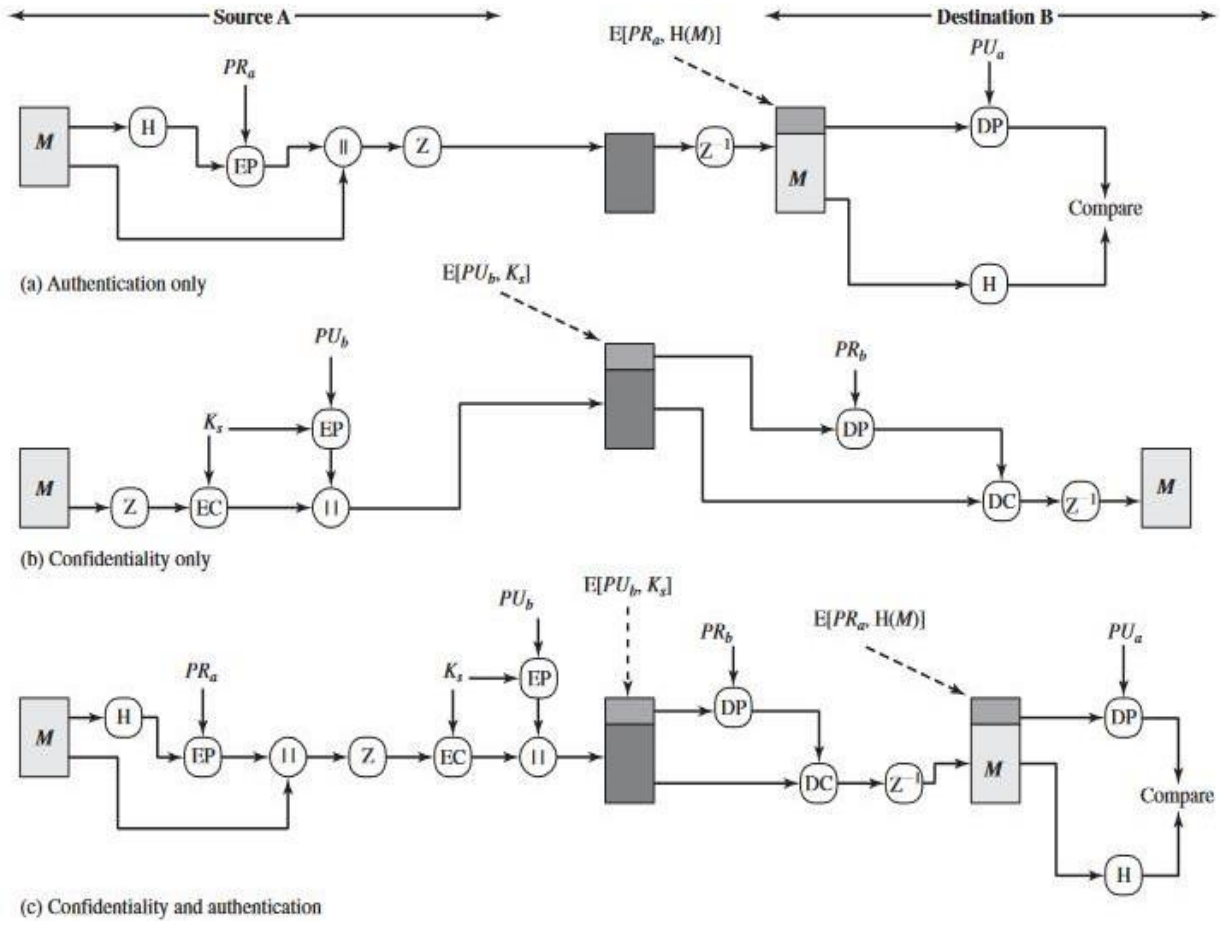


Figure 19.1 An IP Security Scenario

Z = compression using ZIP algorithm
R64 = conversion to radix 64 ASCII format



Operational description:

The actual operation of PGP, as opposed to the management of keys, consists of fourservices: authentication, confidentiality, compression, and e-mail compatibility.

Authentication:

The sequence of steps as follows

1. The sender creates a message.
2. SHA-1 is used to generate a 160-bit hash code of the message.
3. The hash code is encrypted with RSA using the sender's private key, and theresult is prepended to the message.
4. The receiver uses RSA with the sender's public key to decrypt and recover the hash code.
5. 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.

CONFIDENTIALITY Another basic service provided by PGP is confidentiality, whichis provided by encrypting messages to be transmitted or to be stored locally as files

The sequence is as follows.

1. The sender generates a message and a random 128-bit number to be used as a session key for this message only.

2. The message is encrypted using CAST-128 (or IDEA or 3DES) with the session key.
3. The session key is encrypted with RSA using the recipient's public key and is prepended to the message.
4. The receiver uses RSA with its private key to decrypt and recover the session key.
5. The session key is used to decrypt the message.

CONFIDENTIALITY AND AUTHENTICATION As both servicesmay 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.

COMPRESSION As a default, PGP compresses the message after applying the signature but before encryption. This has the benefit of saving space both for e-mail transmission and for file storage.

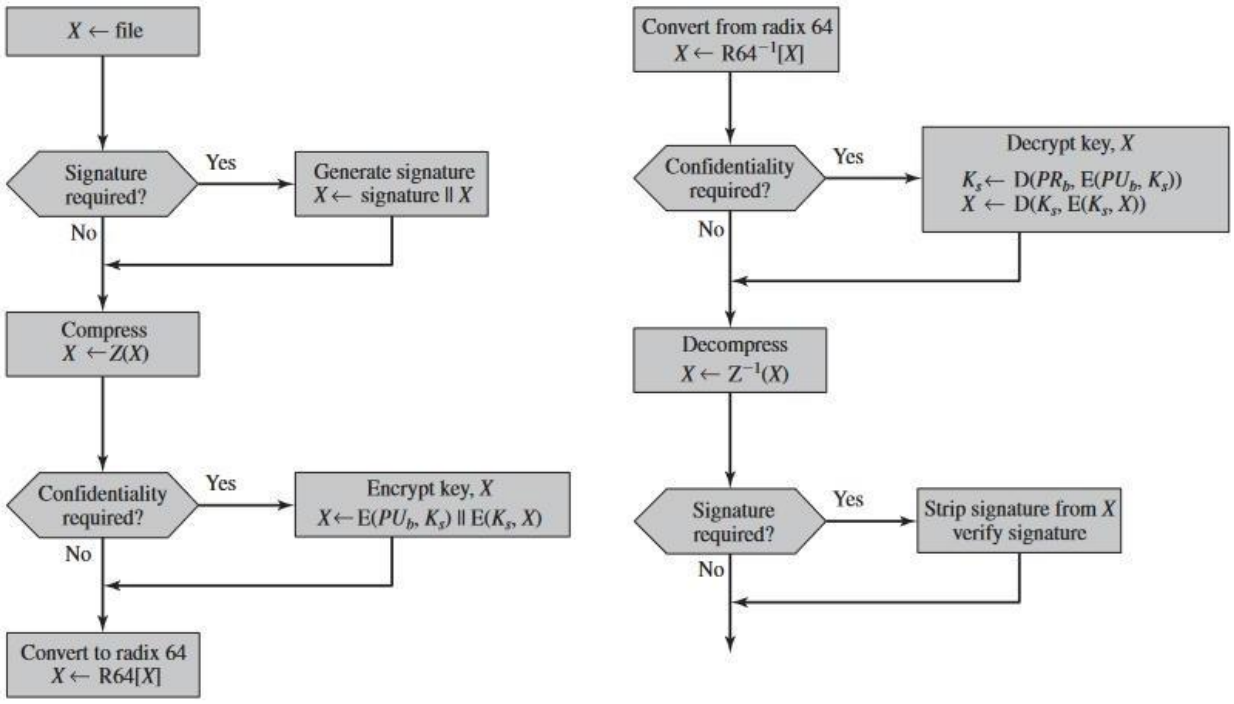
1. The signature is generated before compression for two reasons:
   - It is preferable to sign an uncompressed message so that one can store onlythe uncompressed message together with the signature for future verification.
   - Even if one were willing to generate dynamically a recompressed messagefor verification, PGP's compression algorithm presents a difficulty. The algorithm is not deterministic; various implementations of the algorithm achievedifferent tradeoffs in running speed versus compression ratio and, as a result,produce different compressed forms.

2. Message encryption is applied after compression to strengthen cryptographic security.

E-MAIL COMPATIBILITY When PGP is used, at least part of the block to be transmitted is encrypted. If only the signature service is used, then the message digest is encrypted (with the sender's private key).

- If the confidentiality service is used, the message plus signature (if present) are encrypted (with a one-time symmetric key).
- Thus, part or all of the resulting block consists of a stream of arbitrary 8-bit octets.

- However, many electronic mail systems only permit the use of blocks consisting of ASCII text.
- To accommodate this restriction, PGP provides the service of converting the raw 8-bit binary stream to a stream of printable ASCII characters.



(a) Generic transmission diagram (from A) (b) Generic reception diagram (to B)

Figure 18.2 Transmission and Reception of PGP Messages

PGP Message Generation:

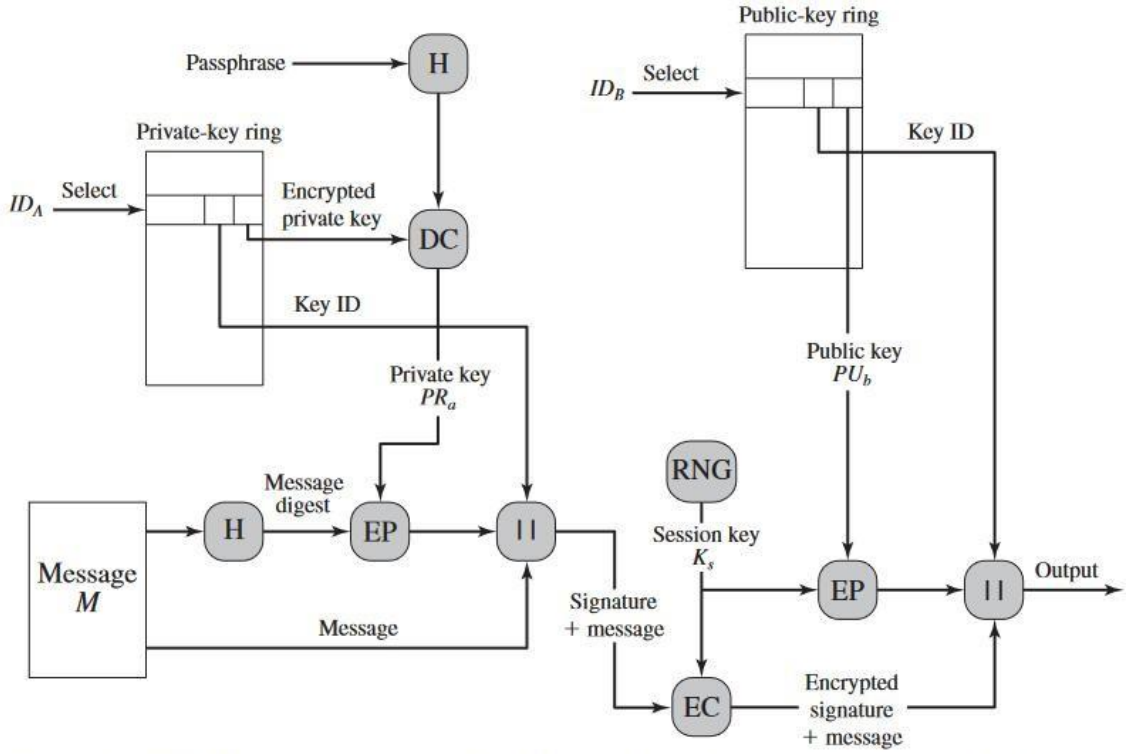


Figure 18.5 PGP Message Generation (from User A to User B: no compression or radix-64 conversion)

S/MIME (Secure/Multipurpose Internet Mail Extensions)

S/MIME is a security enhancementto the MIME Internet e-mail format standard based on technology from RSA DataSecurity.

Multipurpose Internet Mail Extension (MIME) is an extension to the RFC 5322 framework that is intended to address some of the problems and limitations of theuse of Simple Mail Transfer Protocol (SMTP), defined in RFC 821, or some othermail transfer protocol and RFC 5322 for electronic mail. [PARZ06] lists the following limitations of the SMTP/5322 scheme.

- SMTP cannot transmit executable files or other binary objects.
- SMTP cannot transmit text data that includes national language characters,because these are represented by 8-bit codes with values of 128 decimal orhigher, and SMTP is limited to 7-bit ASCII.
- SMTP servers may reject mail message over a certain size.
- SMTP gateways that translate between ASCII and the character code EBCDICdo not use a consistent set of mappings, resulting in translation problems.
- SMTP gateways to X.400 electronic mail networks cannot handle non-textualdata included in X.400 messages.

Header fields in MIME:

The five header fields defined in MIME are

- **MIME-Version:** Must have the parameter value 1.0. This field indicates thatthe message conforms to RFCs 2045 and 2046.
- **Content-Type:** Describes the data contained in the body with sufficient detailthat the receiving user agent can pick an appropriate agent or mechanism torepresent the data to the user or otherwise deal with the data in an appropriate manner.
- **Content-Transfer-Encoding:** Indicates the type of transformation that has been used to represent the body of the message in a way that is acceptable formail transport.
- **Content-ID:** Used to identify MIME entities uniquely in multiple contexts.
- **Content-Description:** A text description of the object with the body; this isuseful when the object is not readable (e.g., audio data).

IP SECURITY OVERVIEW

IP security (IPsec) is a capability that can be added to either current version of the Internet Protocol (IPv4 or IPv6) by means of additional headers. IPsec encompasses three functional areas: authentication, confidentiality, and key management.

In 1994, the Internet Architecture Board (IAB) issued a report titled "Security inthe Internet Architecture"

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 bothwith the current IPv4 and the future IPv6.

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:

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.

The principal feature of IPsec that enables it to support these varied applications is that it can encrypt and/or authenticate all traffic at the IP level. Thus, all distributed applications (including remote logon, client/server, e-mail, file transfer, Web access, and so on) can be secured.

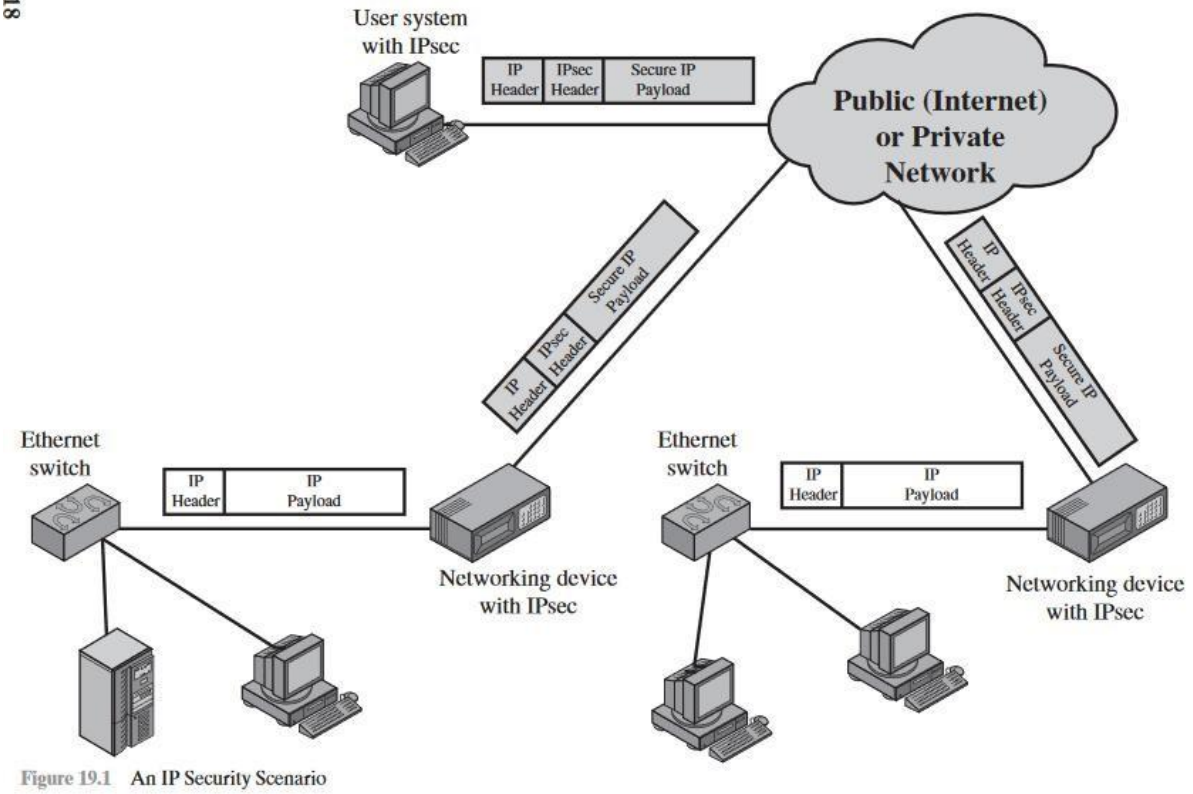


Figure 19.1 An IP Security Scenario

Figure 19.1 is a typical scenario of IPsec usage. An organization maintains LANs at dispersed locations. Non-secure IP traffic is conducted on each LAN.

Fortraffic 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.

Benefits of IPsec

Some of the 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. Traffic within a company or workgroup does not incur the overhead of security-related processing.
- IPsec in a firewall is resistant to bypass if all traffic from the outside must useIP and the firewall is the only means of entrance from the Internet into theorganization.
- IPsec is below the transport layer (TCP, UDP) and so is transparent toapplications.
- IPsec can be transparent to end users.
- IPsec can provide security for individual users if needed.

Routing Applications

- Router advertisement (a new router advertises its presence) comes from anauthorized router.
- A neighbor advertisement (a router seeks to establish or maintain a neighbour relationship with a router in another routing domain) comes from an authorized router.
- A redirect message comes from the router to which the initial IP packet was sent.
- A routing update is not forged.

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.

- Access control
- Connectionless integrity
- Data origin authentication
- Rejection of replayed packets (a form of partial sequence integrity)
- Confidentiality (encryption)
- Limited traffic flow confidentiality

IP SECURITY ARCHITECTURE

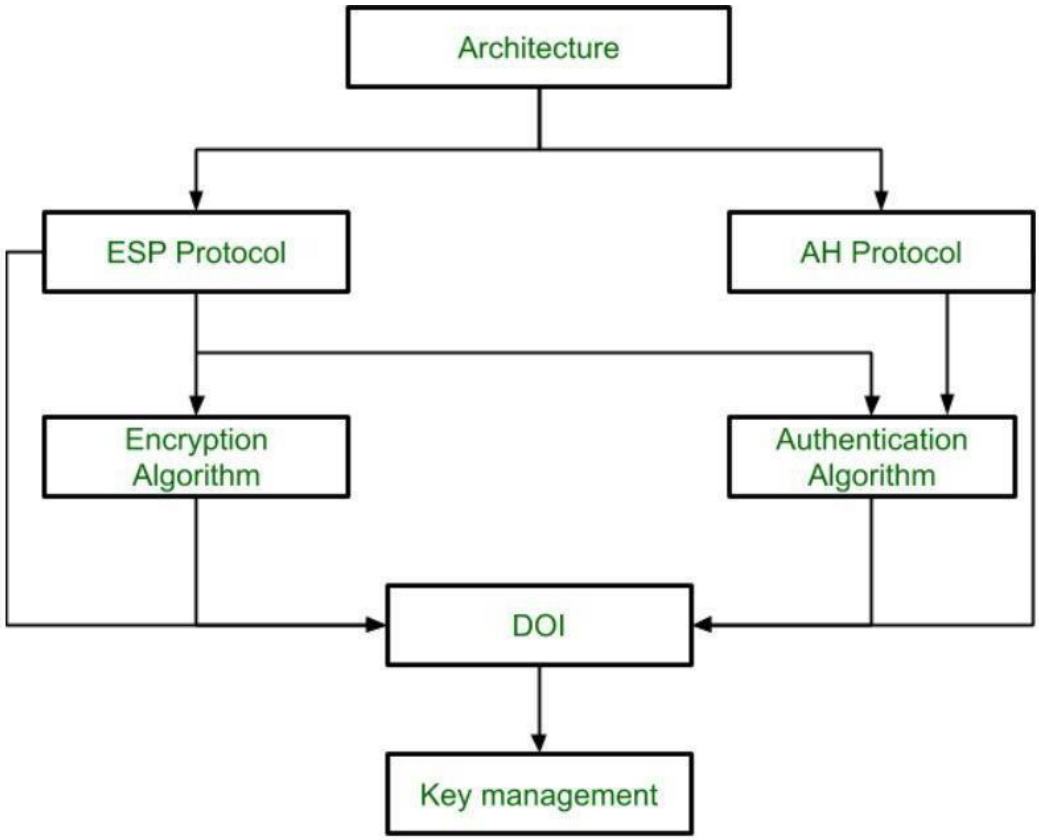
IPSec (IP Security) architecture uses two protocols to secure the traffic or data flow. These protocols are ESP (Encapsulation Security Payload) and AH (Authentication Header). IPSec Architecture includes protocols, algorithms, DOI, and Key Management. All these components are very important in order to provide the three main services:

- Confidentiality
- Authentication
- Integrity

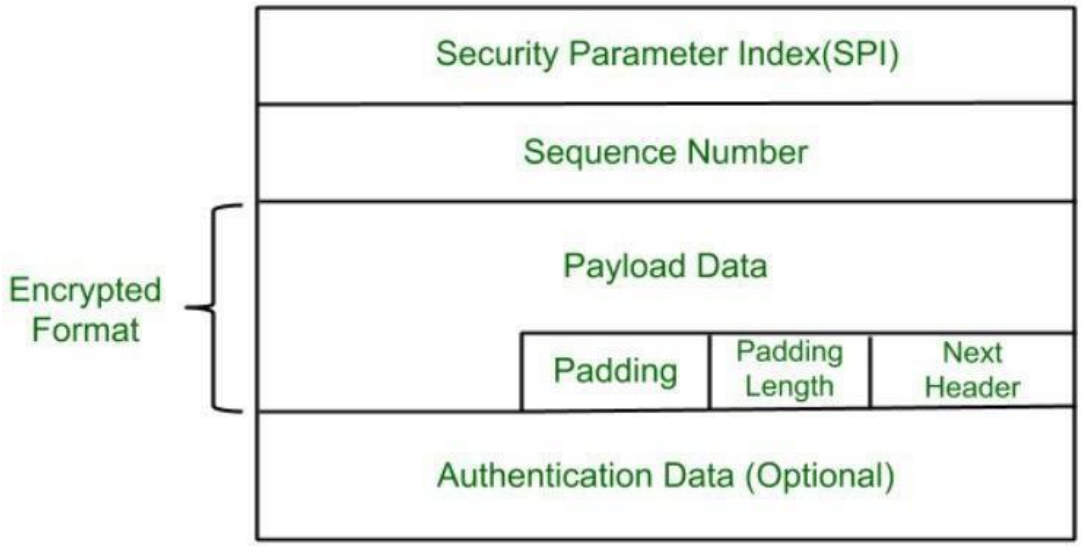
1. Architecture: Architecture or IP Security Architecture covers the general concepts, definitions, protocols, algorithms, and security requirements of IP Security technology.

2. ESP Protocol:ESP(Encapsulation Security Payload) provides a confidentiality service. Encapsulation Security Payload is implemented in either two ways:

- ESP with optional Authentication.
- ESP with Authentication.



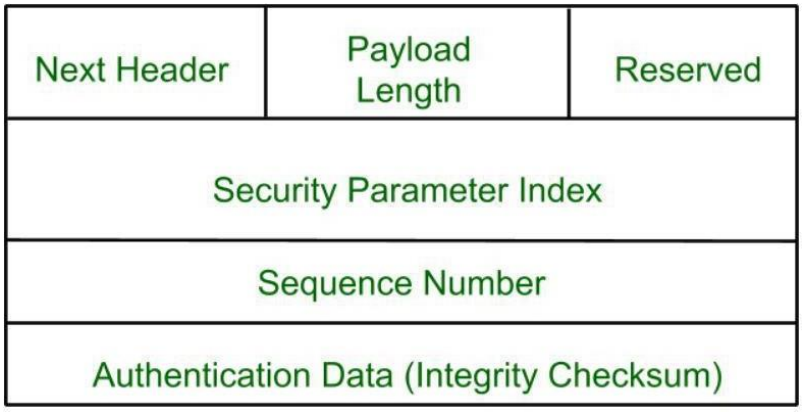
Packet Format:



- **Security Parameter Index(SPI):** This parameter is used by Security Association. It is used to give a unique number to the connection built between the Client and Server.
- **Sequence Number:** Unique Sequence numbers are allotted to every packet so that on the receiver side packets can be arranged properly.
- **Payload Data:** Payload data means the actual data or the actual message. The Payload data is in an encrypted format to achieve confidentiality.
- **Padding:** Extra bits of space are added to the original message in order to ensure confidentiality. Padding length is the size of the added bits of space in the original message.
- **Next Header:** Next header means the next payload or next actual data.
- **Authentication Data** This field is optional in ESP protocol packet format.

3. Encryption algorithm: The encryption algorithm is the document that describes various encryption algorithms used for Encapsulation Security Payload.

4. AH Protocol: AH (Authentication Header) Protocol provides both Authentication and Integrity service. Authentication Header is implemented in one way only: Authentication along with Integrity.



Authentication Header covers the packet format and general issues related to the use of AH for packet authentication and integrity.

5. Authentication Algorithm: The authentication Algorithm contains the set of documents that describe the authentication algorithm used for AH and for the authentication option of ESP.

6. DOI (Domain of Interpretation): DOI is the identifier that supports both AH and ESP protocols. It contains values needed for documentation related to each other.

7. Key Management: Key Management contains the document that describes how the keys are exchanged between sender and receiver.

AUTHENTICATION HEADER

Authentication Header (AH) is used to provide integrity and authentication to IP datagrams. Replay protection is also possible. The services are connectionless, that means they work on a per-packet basis.

AH is used in two modes as follows –

- Transport mode
- Tunnel mode

AH authenticates are the same as IP datagram. In transport mode, some fields in the IP header change en-route and their value cannot be predicted by the receiver. These fields are called mutable and they are not protected by AH.

Mutable IPv4 fields

The mutable IPv4 fields are as follows –

- Type of service (TOS)
- Flags
- Fragment offset
- Time to live (TTL)
- Header checksum

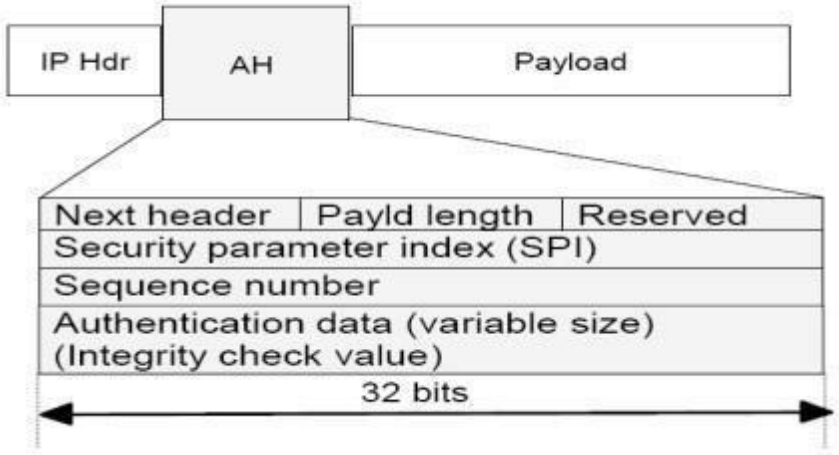
To protect these fields, tunnelling must be used. The payload of the IP packet is considered immutable and is always protected by AH.

- AH processing is applied only to non-fragmented IP packets. Whereas an IP packet with AH applied can be fragmented by intermediate routers.

- In this case, the destination first reassembles the packet and then applies AH processing to it.
- If an IP packet that appears to be a fragment is input to AH processing, and it is discarded.
- This prevents the overlapping fragment attack, which misuses the fragment reassembly algorithm to create forged packets and force them through a firewall.
- Packets that fail authentication are discarded and never delivered to upper layers.
- This mode of operation greatly reduces the chances of successful denial-of-service attacks.

AH format

The AH format is described in RFC 2402. The below shows the position of the Authentication Header fields in the IP packet.



The fields are as follows –

Next header

It is an 8-bit field which identifies the type of what follows. The value of this field is chosen from the set of IP header protocol fields, which is set to 51, and the value that would have gone in the protocol field goes in the AH next header field.

Payload length

It is an 8 bits long field and contains the length of the AH header expressed in 32-bit words, minus 2. It does not relate to the actual payload length of the IP packet. Suppose if default options are used, the value is 4 (three 32-bit fixed words plus three 32-bit words of authentication data minus two).

Reserved

It is reserved for future use. Its length is 16 bits and it is set to zero.

Security parameter index (SPI)

It is 32 bits in length.

Sequence number

This 32-bit field is a monotonically increasing counter, which is used for replay protection. It is an optional field. The sender always includes this field, and it is at the discretion of the receiver to process it or not. Starting the sequence number is initialized to zero. The first packet transmitted using the SA has a sequence number of 1. Sequence numbers are not allowed to repeat.

Authentication data

This is a variable-length field containing the Integrity Check Value (ICV), and is padded to 32 bits for IPv4 or 64 bits for IPv6.

ENCAPSULATING SECURITY PAYLOAD (ESP)

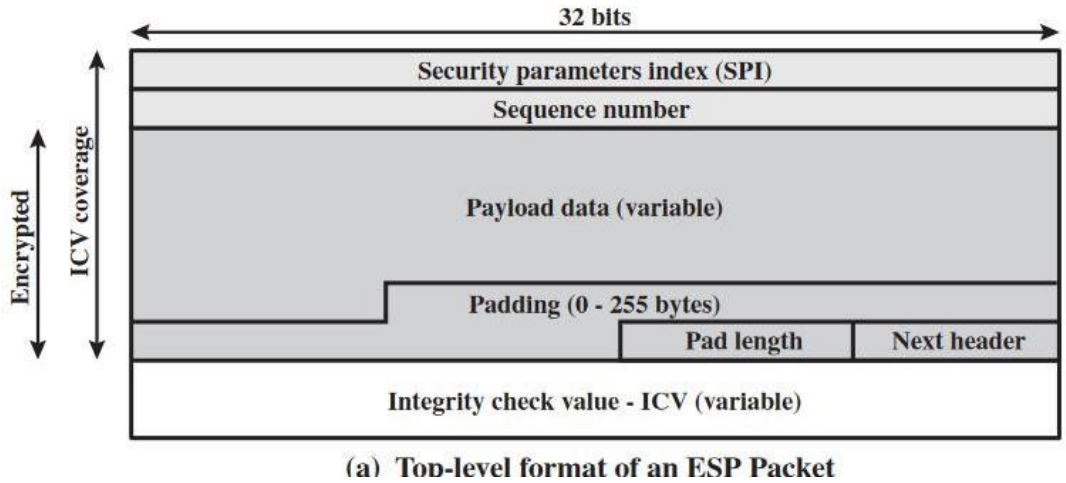
ESP can be used to provide confidentiality, data origin authentication, connection-less integrity, an anti-replay service (a form of partial sequence integrity), and (limited) traffic flow confidentiality. The set of services provided depends on options selected at the time of Security Association (SA) establishment and on the location of the implementation in a network topology.

ESP Format

Figure 19.5a shows the top-level format of an ESP packet. It contains the followingfields.

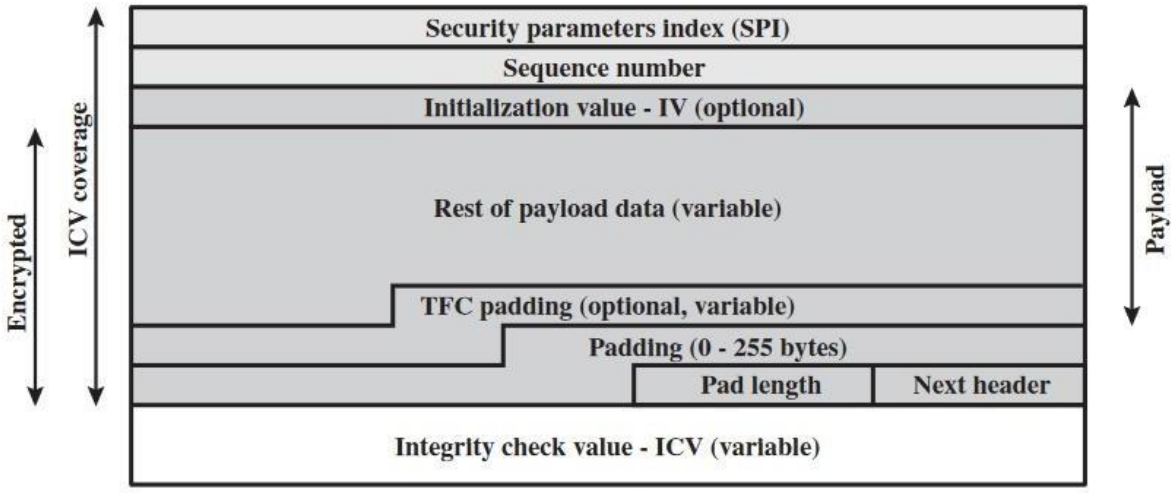
- **Security Parameters Index (32 bits):** Identifies a security association.
- **Sequence Number (32 bits):** A monotonically increasing counter value; this provides an anti-replay function, as discussed for AH.
- **Payload Data (variable):** This is a transport-level segment (transport mode)or IP packet (tunnel mode) that is protected by encryption.
- **Pad Length (8 bits):** Indicates the number of pad bytes immediately precedingthis field.
- **Next Header (8 bits):** Identifies the type of data contained in the payload datafield by identifying the first header in that payload (for example, an extensionheader in IPv6, or an upper-layer protocol such as TCP).

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



(a) Top-level format of an ESP Packet

\



(b) Substructure of payload data

Figure 19.5 ESP Packet Format

When any combined mode algorithm is employed, the algorithm itself isexpected to return both decrypted plaintext and a pass/fail indication for the integrity check.

Padding

The Padding field serves several purposes:

If an encryption algorithm requires the plaintext to be a multiple of somnumber of bytes (e.g., the multiple of a single block for a block cipher), the Padding field is used to expand the plaintext (consisting of the Payload Data, Padding, Pad Length, and Next Header fields) to the required length.

Anti-Replay Service

A replay attack is one in which an attacker obtains a copy of an authenticated packet and later transmits it to the intended destination. The receipt of duplicate, authenticated IP packets may disrupt service in some way or may have some other undesired consequence. The Sequence Number field is designed to thwart such attacks. First, we discuss sequence number generation by the sender, and then we look at how it is processed by the recipient.

Transport and Tunnel Modes

Figure 19.7 shows two ways in which the IPsec ESP service can be used. In the upperpart of the figure, encryption (and optionally authentication) is provided directly between two hosts. Figure 19.7b shows how tunnel mode operation can be used to set up a virtual private network.

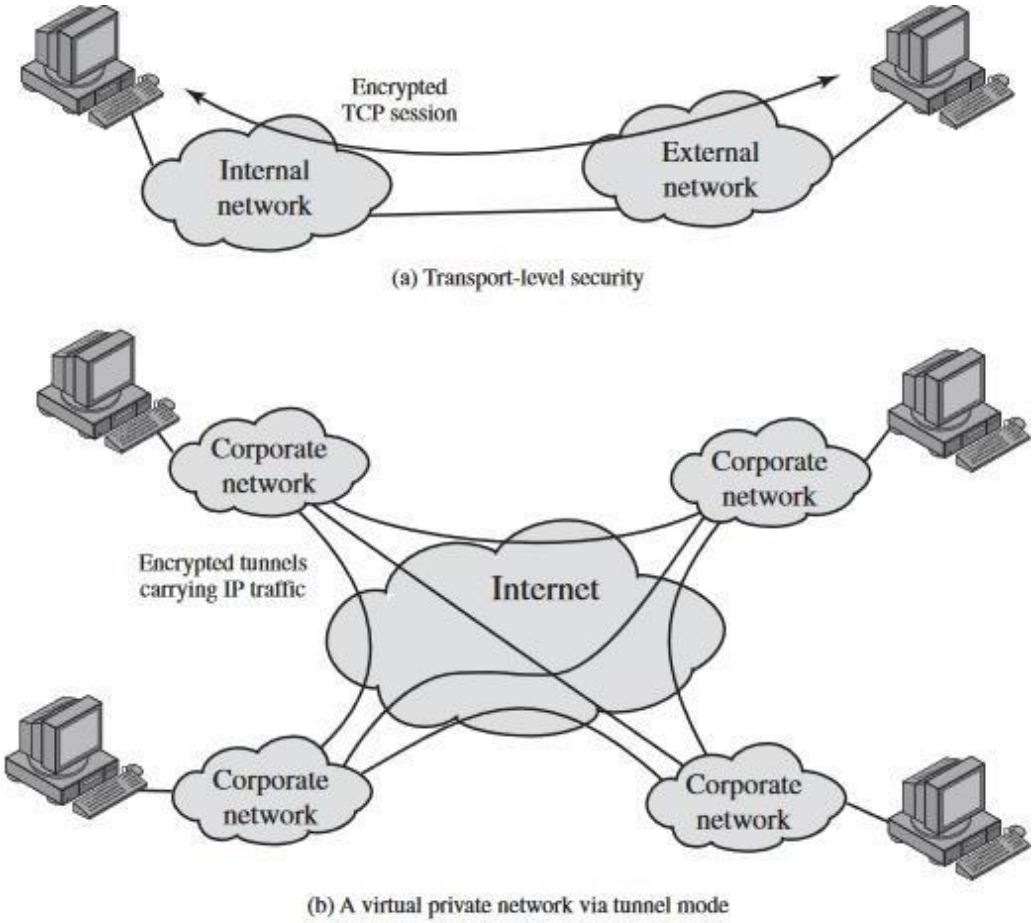


Figure 19.7 Transport-Mode versus Tunnel-Mode Encryption

In this example, an organization has four private networks interconnected across the Internet. Hosts on the internal networks use the Internet for transport of data but do not interact with other Internet-based hosts. By terminating the tunnels at the security gateway to each internal network, the configuration allows the hosts to avoid implementing the security capability. The former technique is supported by a transport mode SA, while the latter technique uses a tunnel mode SA.

COMBINING SECURITY ASSOCIATIONS

An individual SA can implement either the AH or ESP protocol but not both. Sometimes a particular traffic flow will call for the services provided by both AH and ESP.

Security associations may be combined into bundles in two ways:

• **Transport adjacency:** Refers to applying more than one security protocol to the same IP packet without invoking tunneling. This approach to combining AH and ESP allows for only one level of combination; further nesting yields no added benefit since the processing is performed at one IPsec instance: the(ultimate) destination.

• **Iterated tunneling:** Refers to the application of multiple layers of security protocols effected through IP tunneling. This approach allows for multiple levels of nesting, since each tunnel can originate or terminate at a different IPsec site along the path.

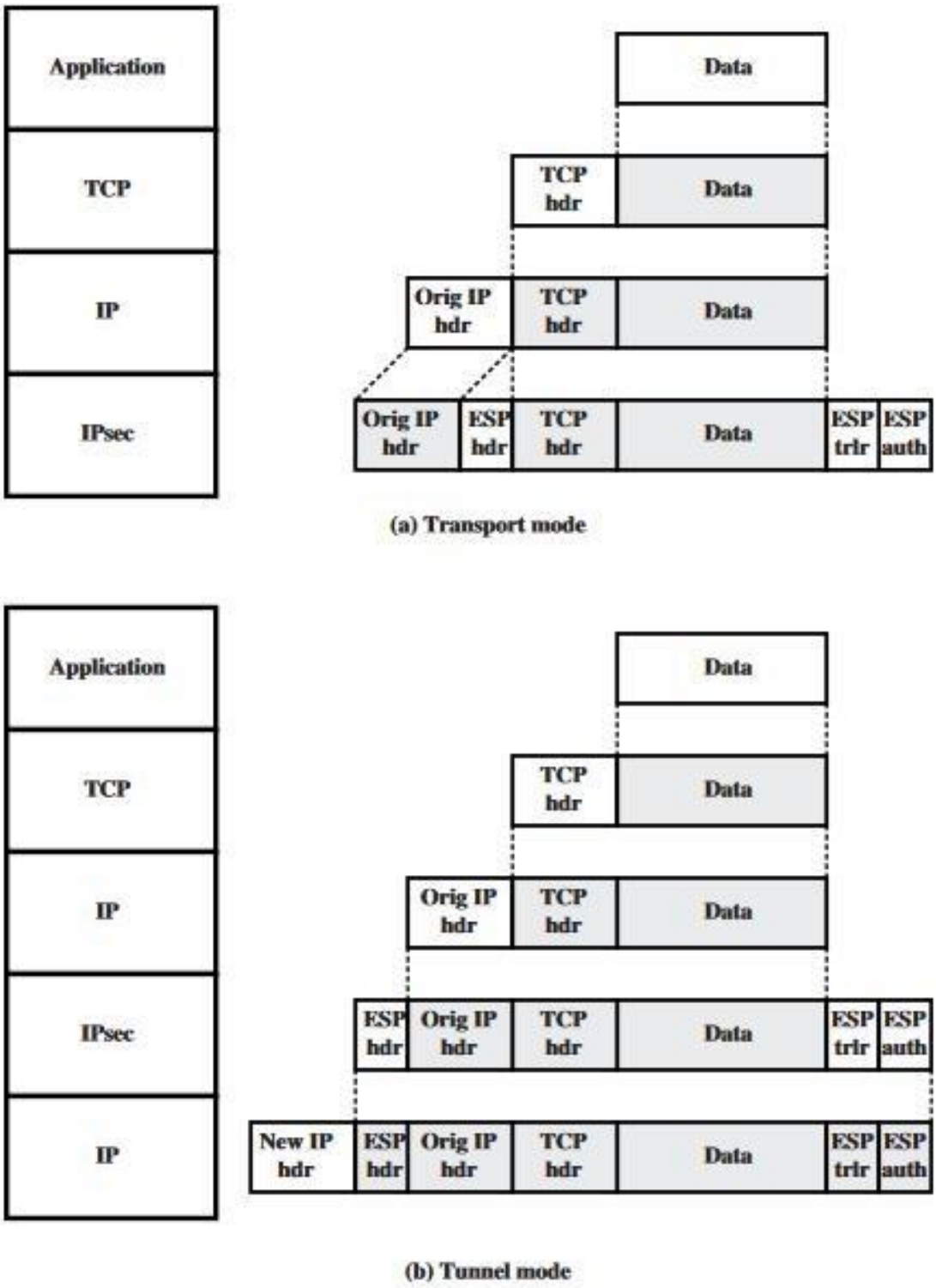


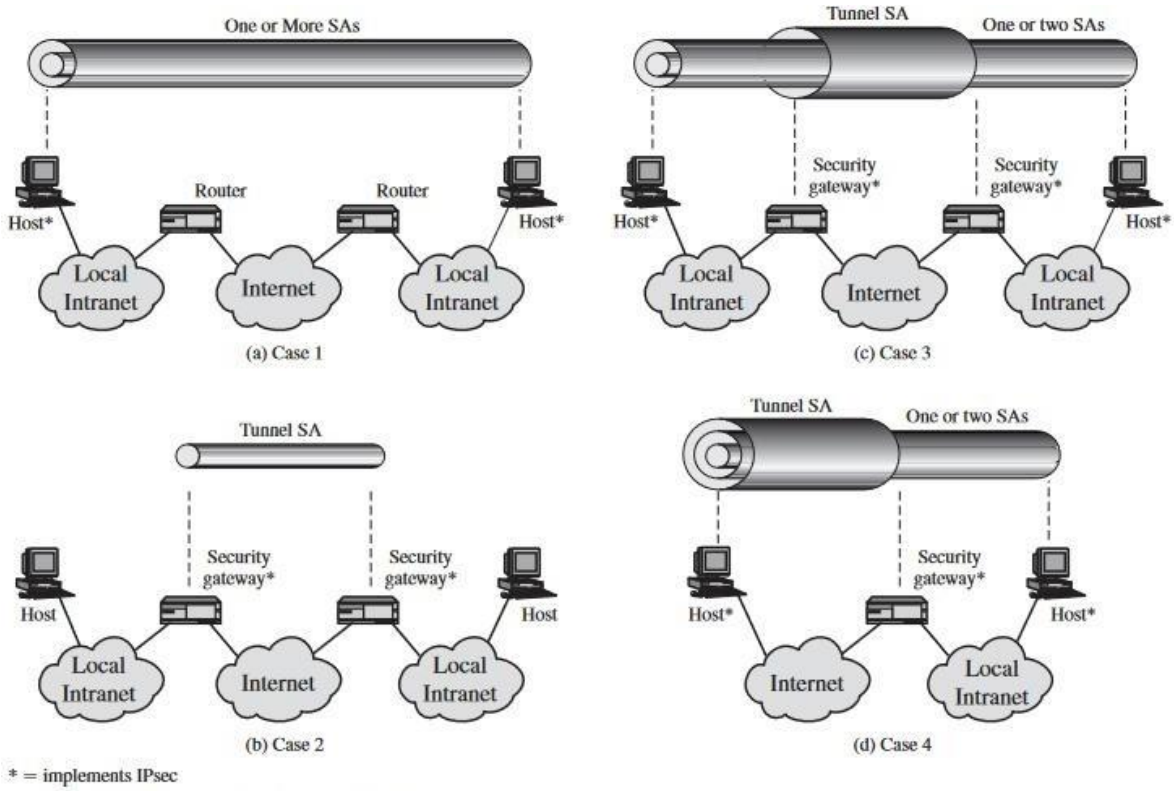
Figure 19.9 Protocol Operation for ESP

The two approaches can be combined, for example, by having a transport SA between hosts travel part of the way through a tunnel SA between security gateways.

Basic Combinations of Security Associations

The IPsec Architecture document lists four examples of combinations of SAs that must be supported by compliant IPsec hosts (e.g., workstation, server) or

security gateways (e.g. firewall, router). These are illustrated in Figure 19.10. The lower partof each case in the figure represents the physical connectivity of the elements; the upper part represents logical connectivity via one or more nested SAs. Each SA can be either AH or ESP. For host-to-host SAs, the mode may be either transport or tunnel; otherwise it must be tunnel mode.



* = implements IPsec

Figure 19.10 Basic Combinations of Security Associations

Case 1. All security is provided between end systems that implement IPsec. For any two end systems to communicate via an SA, they must share the appropriate secret keys.

Case 2. Security is provided only between gateways (routers, firewalls, etc.) and no hosts implement IPsec. This case illustrates simple virtual private network support.

Case 3. This builds on case 2 by adding end-to-end security. The same combinations discussed for cases 1 and 2 are allowed here. The gateway-to-gateway tunnel provides either authentication, confidentiality, or both for all traffic between end systems.

Case 4. This provides support for a remote host that uses the Internet to reach anorganization's firewall and then to gain access to some server or workstation behindthe firewall.

KEY MANAGEMENT

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.

FEATURES OF IKE KEY DETERMINATION The IKE key determination algorithm ischaracterized by five important features:

1. It employs a mechanism known as cookies to thwart clogging attacks.
2. It enables the two parties to negotiate a group; this, in essence, specifies the global parameters of the Diffie-Hellman key exchange.
3. It uses nonces to ensure against replay attacks.
4. It enables the exchange of Diffie-Hellman public key values.
5. It authenticates the Diffie-Hellman exchange to thwart man-in-the-middle attacks.

Header and Payload Formats

IKE defines procedures and packet formats to establish, negotiate, modify, and delete security associations. As part of SA establishment, IKE defines payloads for exchanging key generation and authentication data. These payload formats provide a consistent framework independent of the specific key exchange protocol, encryption algorithm, and authentication mechanism.

It consists of the following fields.

- **Initiator SPI (64 bits):** A value chosen by the initiator to identify a uniqueIKE security association (SA).
- **Responder SPI (64 bits):** A value chosen by the responder to identify aunique IKE SA.
- **Next Payload (8 bits):** Indicates the type of the first payload in the message; payloads are discussed in the next subsection.
- **Major Version (4 bits):** Indicates major version of IKE in use.
- **Minor Version (4 bits):** Indicates minor version in use.

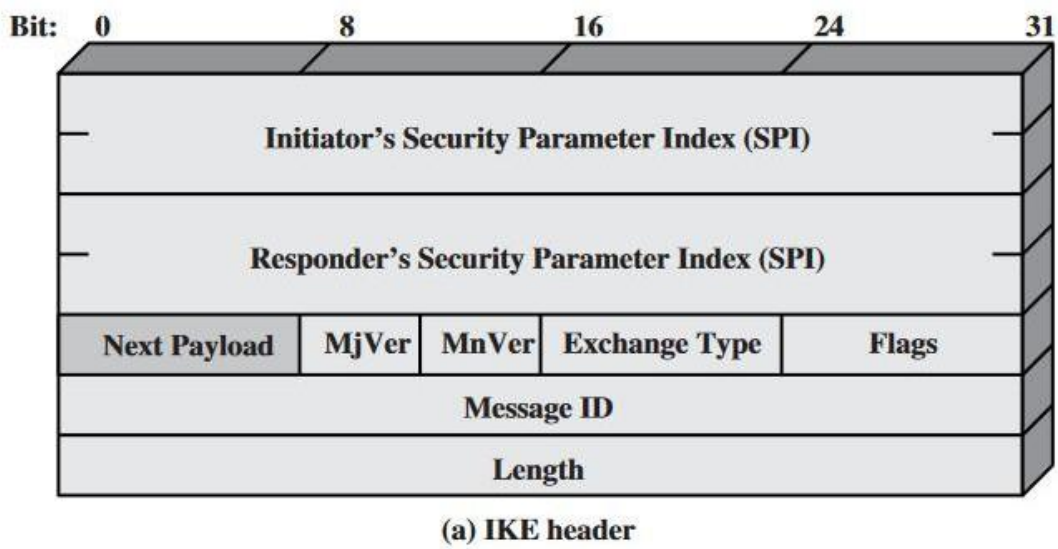


Figure 19.12 IKE Formats

Exchange Type (8 bits): Indicates the type of exchange; these are discussed later in this section.

• **Flags (8 bits):** Indicates specific options set for this IKE exchange. Three bits are defined so far. The initiator bit indicates whether this packet is sent by the SA initiator. The version bit indicates whether the transmitter is capable of using a higher major version number than the one currently indicated. The response bit indicates whether this is a response to a message containing the same message ID.

• **Message ID (32 bits):** Used to control retransmission of lost packets and matching of requests and responses.

• **Length (32 bits):** Length of total message (header plus all payloads) in octets

UNIT-5
WEB SECURITY

WEB SECURITY CONSIDERATIONS

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

- The Internet is two-way.
- The Web is increasingly serving as a highly visible outlet for corporate and product information and as the platform for business transactions.
- Although Web browsers are very easy to use, Web servers are relatively easy to configure and manage, and Web content is increasingly easy to develop, the underlying software is extraordinarily complex.
- A Web server can be exploited as a launching pad into the corporation's or agency's entire computer complex
- Casual and untrained (in security matters) users are common clients for Web-based services

Web Security Threats

Table 16.1 A Comparison of Threats on the Web

	Threats	Consequences	Countermeasures
Integrity	• Modification of user data • Trojan horse browser • Modification of memory • Modification of message traffic in transit	• Loss of information • Compromise of machine • Vulnerabilty to all other threats	Cryptographic checksums
Confidentiality	• Eavesdropping on the net • Theft of info from server • Theft of data from client • Info about network configuration • Info about which client talks to server	• Loss of information • Loss of privacy	Encryption, Web proxies
Denial of Service	• Killing of user threads • Flooding machine with bogus requests • Filling up disk or memory • Isolating machine by DNS attacks	• Disruptive • Annoying • Prevent user from getting work done	Difficult to prevent
Authentication	• Impersonation of legitimate users • Data forgery	• Misrepresentation of user • Belief that false information is valid	Cryptographic techniques

One way to group these threats is in terms of passive and active attacks.

Another way to classify Web security threats is in terms of the location of the threat: Web server, Web browser, and network traffic between browser and server.

Web Traffic Security Approaches

One way to provide Web security is to use IP security (IPsec) (Figure 16.1a). The advantage of using IPsec is that it is transparent to end users and applications and provides a general-purpose solution.

Another relatively general-purpose solution is to implement security just above TCP.

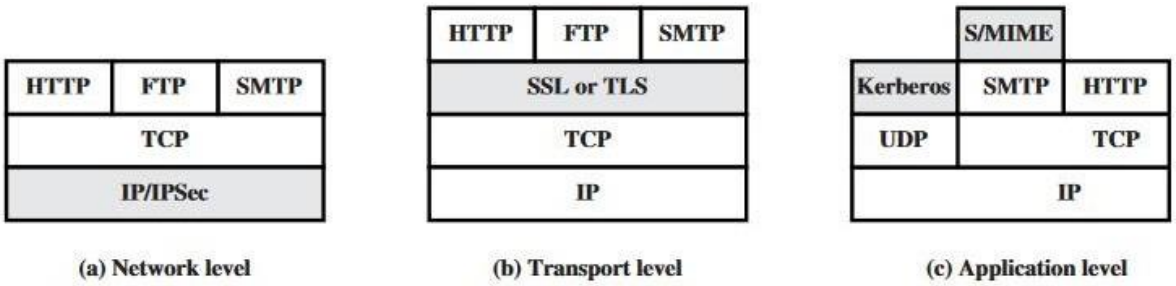


Figure 16.1 Relative Location of Security Facilities in the TCP/IP Protocol Stack

Application-specific security services are embedded within the particular application.

SECURE SOCKET LAYER AND TRANSPORT LAYER SECURITY

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

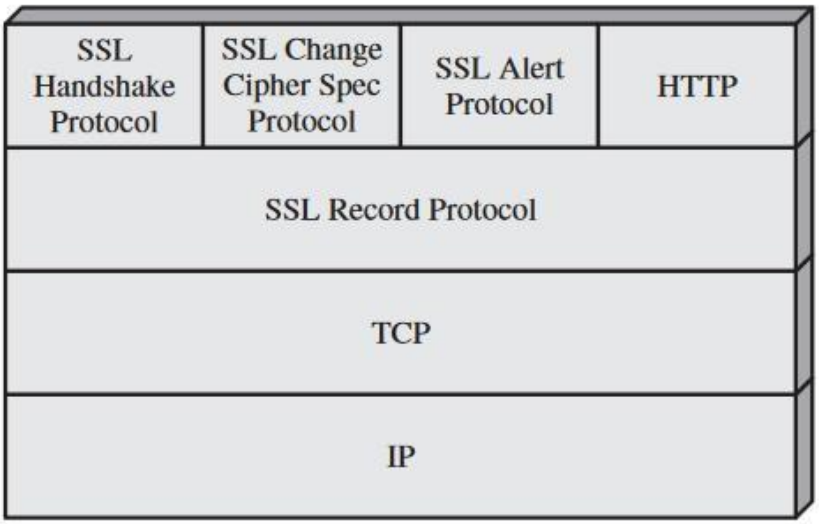


Figure 16.2 SSL Protocol Stack

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

- **Connection:** 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:** 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

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.
- **Master secret:** 48-byte secret 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 secret encryption key for data encrypted by the server and decrypted by the client.
- **Client write key:** The symmetric 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 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).

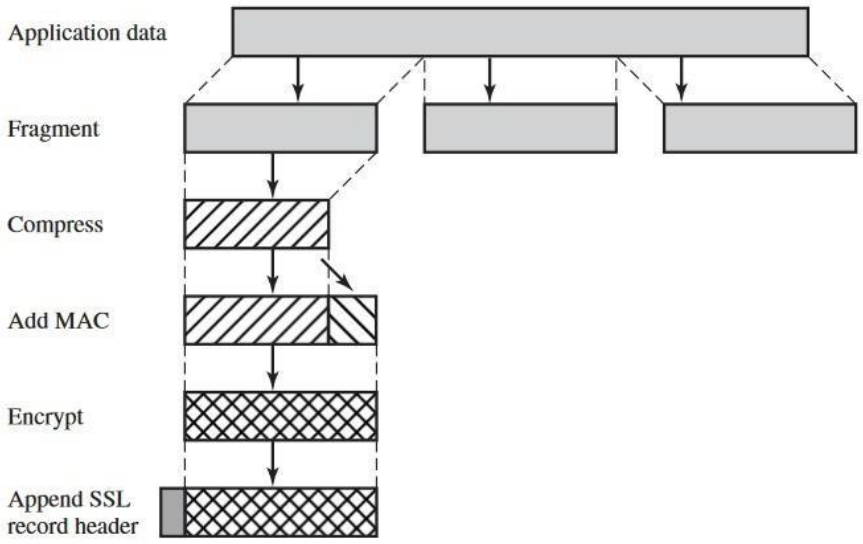


Figure 16.3 SSL Record Protocol Operation

The final step of SSL Record Protocol processing is to prepare 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^{14} +2048.

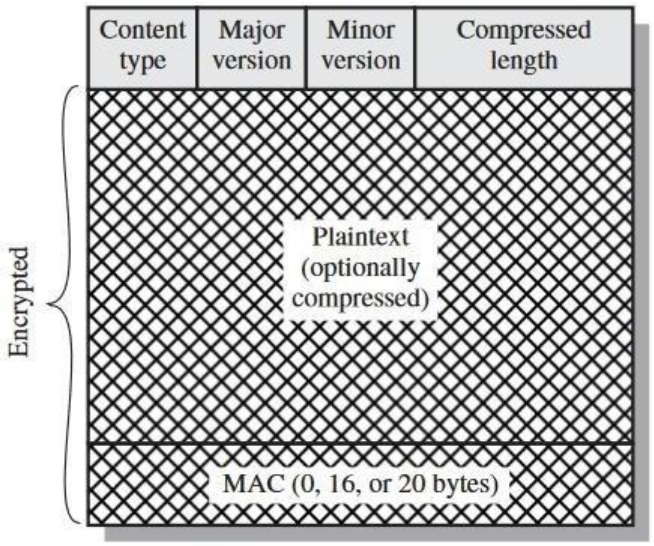


Figure 16.4 SSL Record Format

TRANSPORT LAYER SECURITY

TLS is an IETF standardization initiative whose goal is to produce an Internet standard version of SSL.

Version Number

The TLS Record Format is the same as that of the SSL Record Format For the current version of TLS, the major version is 3 and the minor version is 3.

Message Authentication Code

There are two differences between the SSLv3 and TLS MAC schemes: the actualalgorithm and the scope of the MAC calculation. TLS makes use of the

HMAC algorithm defined in RFC 2104. Recall from Chapter 12 that HMAC is defined as

$$HMAC_K(M) = H[(K^+ \oplus opad)||H[(K^+ \oplus ipad)||M]$$

Pseudorandom Function

TLS makes use of a pseudorandom function referred to as PRF to expand secretsinto blocks of data for purposes of key generation or validation. The objective is tomake use of a relatively small shared secret value but to generate longer blocks ofdata in a way that is secure from the kinds of attacks made on hash functions and*MACs*.

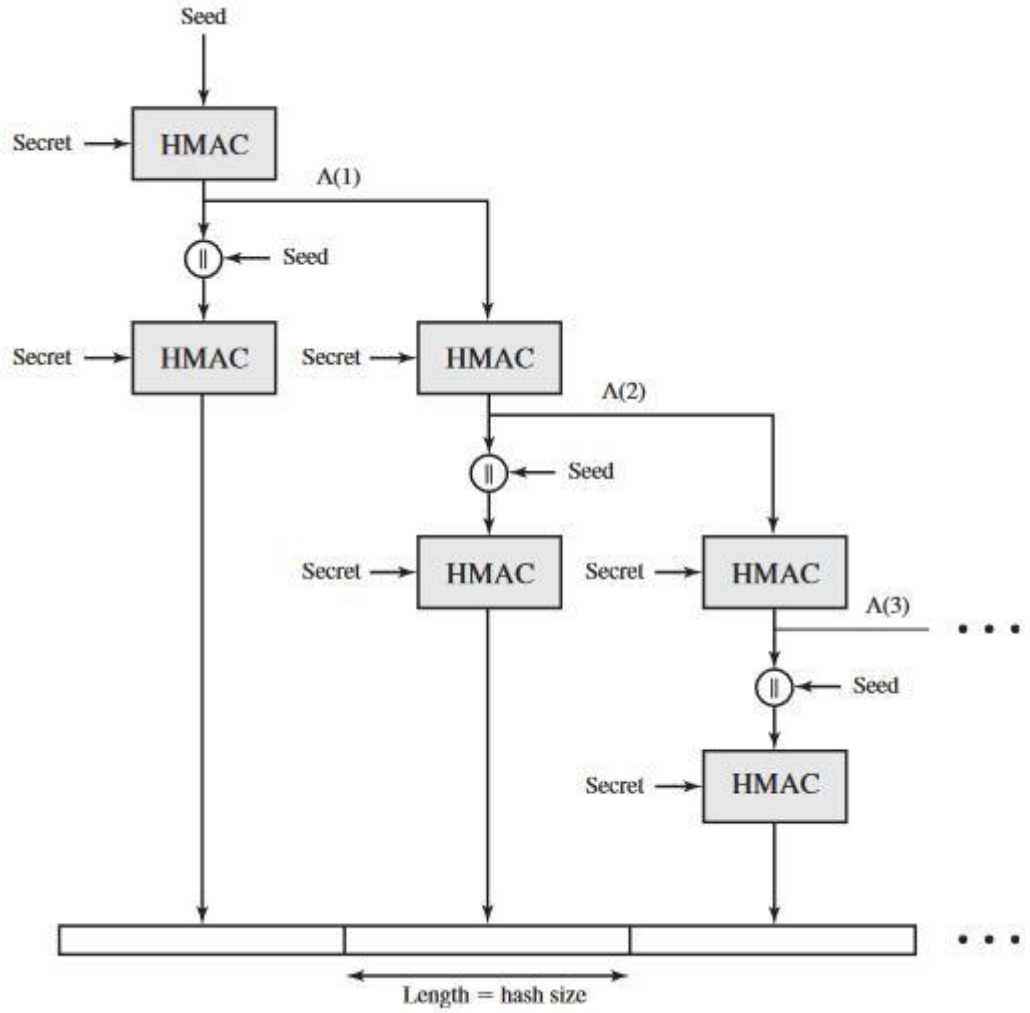


Figure 16.7 TLS Function P_hash(secret, seed)

Alert Codes

TLS supports all of the alert codes defined in SSLv3 with the exception of no_certificate. A number of additional codes are defined in TLS; of these, the following are always fatal.

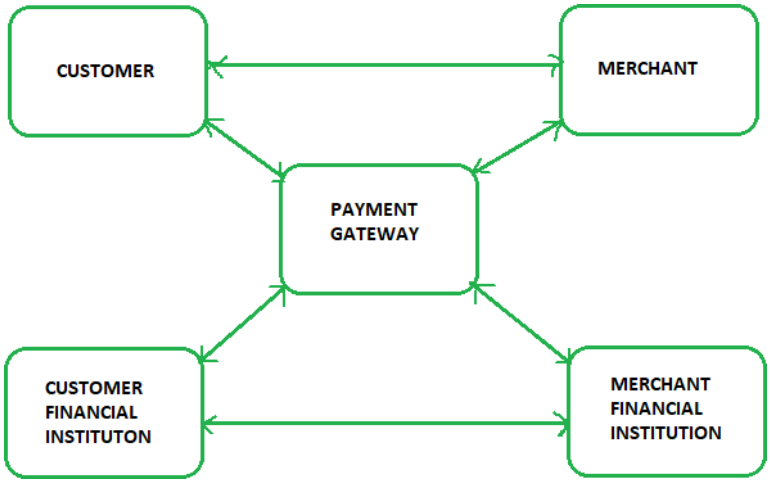
- **record_overflow:** A TLS record was received with a payload (ciphertext) whose length exceeds 2^{14}+2048bytes, or the ciphertext decrypted to a length of greater than 2^{14}+1024 bytes
- **unknown_ca:** A valid certificate chain or partial chain was received, but the certificate was not accepted because the CA certificate could not be located or could not be matched with a known, trusted CA.
- **access_denied:** A valid certificate was received, but when access control was applied, the sender decided not to proceed with the negotiation.
- **decode_error:** A message could not be decoded, because either a field was out of its specified range or the length of the message was incorrect.
- **protocol_version:** The protocol version the client attempted to negotiate is recognized but not supported.
- **insufficient_security:** Returned instead of handshake_failure when a negotiation has failed specifically because the server requires ciphers more secure than those supported by the client.
- **unsupported_extension:** Sent by clients that receive an extended server hello containing an extension not in the corresponding client hello.
- **internal_error:** An internal error unrelated to the peer or the correctness of the protocol makes it impossible to continue.
- **decrypt_error:** A handshake cryptographic operation failed, including being unable to verify a signature, decrypt a key exchange, or validate a finished message.

SECURE ELECTRONIC TRANSACTION (SET)

Secure Electronic Transaction or SET is a system that ensures the security and integrity of electronic transactions done using credit cards in a scenario. SET is not some system that enables payment but it is a security protocol applied to those payments. It uses different encryption and hashing techniques to secure payments over the internet done through credit cards. The SET protocol was supported in development by major organizations like Visa, Mastercard, Microsoft which provided its Secure Transaction Technology (STT), and Netscape which provided the technology of Secure Socket Layer (SSL).

SET protocol restricts the revealing of credit card details to merchants thus keeping hackers and thieves at bay. The SET protocol includes Certification Authorities for making use of standard Digital Certificates like X.509 Certificate.

Before discussing SET further, let's see a general scenario of electronic transactions, which includes client, payment gateway, client financial institution, merchant, and merchant financial institution.



- **Cardholder:** A cardholder is an authorized holder of the payment card. The card can be a Master card or a Visa which an issuer has issued.
- **Merchant:** A merchant is any person or organization who wants to sell its goods and services to cardholders. Note that a merchant must have a relationship with the acquirer to accept the payment through the internet.
- **Issuer:** An issuer is a financial organization such as a bank that issues payment card – Master card or visa to user or cardholder. The issuer is responsible for the cardholder's debt payment.
- **Acquirer:** This is a financial organization with a relationship with the merchant for processing the card payment authorization and all the payments. An acquirer is part of this process because the merchant can accept credit cards of more than one brand. It also provides an electronic fund transfer to the merchant account.
- **Payment Gateway:** For payment authorization, the payment gateway acts as an interface between secure electronic transactions and existing card payment networks. The merchant exchanges the Secure Electronic Transaction message with the payment gateway through the internet. In response to that, the payment gateway connects to the acquirer's system by using a dedicated network line.
- **Certification Authority:** It is a trusted authority that provides public-key certificates to cardholders, payment gateways, and merchants.
-

How Secure Electronic Transaction Works?

Secure Electronic Transaction works as follows:

Step 1: Customer Open an Account

The customer opens a credit card account like a master card or visa with a bank, i.e. issuer that supports electronic payment transactions and the secure electronic transaction protocol.

Step 2: Customer Receive a Certificate

Once the customer identity is verified (Verification can be done by using a passport, business documents or other documents), it receives a digital certificate which is issued by CA (Certificate Authority). This certificate contains customer details like name, public key, expiry date, certificate number, etc.

Step 3: Merchant Receives a Certificate

The merchant who wants to accept certain credit card brands must process a digital certificate for trustworthiness.

Step 4: Customer Place an Order

It is a shopping cart process where customers borrow an item from the available list, search for the specific item according to requirements, and place the order. Once the customer places the orders, the merchant, in return, sends the details of the order, such as a list of items selected, their quantity and price, total bill, etc., to maintain a record of the order at the customer site.

Step 5: Merchant is Verified

Merchant also sends a digital certificate to the customer to ensure the customers that they are dealing with an authorized or valid merchant.

Step 6: The Order and Payment Details Are Sent

Along with the customer's digital certificate customer also sends an order and payment details to the merchant. The order part is used to confirm the transaction with the reference of items that are mentioned in the order form. The payment part contains the credit card(master card or visa) details. This payment information is in encrypted form; even the merchant cannot read it. The customer certificate ensures the merchant of a customer's identity.

Step 7: Merchant Requests Payment Authorization

Once the merchant gets the customer's payment details, it transfers them to the payment gateway via the acquirer and requests the payment gateway to authorize the payment details. This process ensures start the customer credit card is valid, and the credit limit is not breached.

Step 8: Payment Gateway Authorizes the Payment

Using the credit card information received from the merchant, the payment gateway cross verify the customer's credit card with the help of the issuer. Based on the verification result, it either authorizes the payment or rejects the payment.

Step 9: Merchant Confirm the Order

Assuming that the payment gateway authorizes the payment, merchants send confirmation of the order to the customer.

Step 10: Merchant Provides a Goods and Services

Now the merchant provides goods and services according to the customer's order.

Step 11: Merchant Request Payment

The merchant sends a request to the payment gateway for making payment. After that, the payment gateway interacts with various financial organizations such as the issuer, acquirer and the clearinghouse to effect the payment from the customer's account to the merchant's account.

INTRUDER

The most common threat to security is the attack by the intruder. Intruders are often referred to as hackers and are the most harmful factors contributing to the vulnerability of security. They have immense knowledge and an in-depth understanding of technology and security. Intruders breach the privacy of users and aim at stealing the confidential information of the users. The stolen information is then sold to third-party, which aim at misusing the information for their own personal or professional gains.

Intruders are divided into three categories:

- **Masquerader:** The category of individuals that are not authorized to use the system but still exploit user's privacy and confidential information by possessing techniques that give them control over the system, such category of intruders is referred to as Masquerader. Masqueraders are outsiders and hence they don't have direct access to the system, their aim is to attack unethically to steal data/ information.
- **Misfeasor:** The category of individuals that are authorized to use the system, but misuse the granted access and privilege. These are individuals that take undue advantage of the permissions and access given to them, such category of intruders is referred to as Misfeasor. Misfeasors are insiders and they have direct access to the system, which they aim to attack unethically for stealing data/ information.
- **Clandestine User:** The category of individuals those have supervision/administrative control over the system and misuse the authoritative power given to them. The misconduct of power is often done by superlative authorities for financial gains, such a category of intruders is referred to as Clandestine User. A Clandestine User can be any of the two, insiders or outsiders, and accordingly, they can have direct/ indirect

access to the system, which they aim to attack unethically by stealing data/ information.

INTRUSION DETECTION

An illegal entrance into your network or an address in your assigned domain is referred to as a *network intrusion*. An intrusion can be passive (in which access is achieved quietly and undetected) or *aggressive* (in which access is gained overtly and without detection) (in which changes to network resources are effected).

Intrusions might occur from the outside or from within your network structure (an employee, customer, or business partner). Some intrusions are just aimed to alert you that an intruder has entered your site and is defacing it with various messages or obscene graphics. Others are more malevolent, attempting to harvest sensitive data on a one-time basis or as part of a long-term parasitic connection that will continue to siphon data until it is identified.

Some intruders will try to implant code that has been carefully developed. Others will infiltrate the network, stealthily siphoning out data on a regular basis or altering public-facing Web sites with varied messages.

An attacker can acquire physical access to your system (by physically accessing a restricted computer and its hard drive and/or BIOS), externally (by assaulting your Web servers or breaching your firewall), or internally (by physically accessing a restricted machine and its hard disc and/or BIOS) (your own users, customers, or partners).

Any of the following can be considered an intrusion –

- Malware, sometimes known as ransomware, is a type of computer virus.
- Attempts to obtain unauthorized access to a system
- DDOS (Distributed Denial of Service) attacks
- Destruction of cyber-enabled equipment
- Employee security breaches that are unintentional (like moving a secure file into a shared folder)
- Untrustworthy users, both within and external to your company
- Phishing campaigns and other methods of deceiving consumers with ostensibly genuine communication are examples of social engineering assaults.

Network Intrusion Attack Techniques

When it comes to compromising networks, attackers are increasingly relying on existing tools and procedures as well as stolen credentials. Operating system utilities, commercial productivity software, and scripting languages, for example, are clearly not malware and have a wide range of lawful applications.

- **Asymmetric Routing** – Attackers will typically employ several routes to gain access to the targeted device or network if the network allows for asymmetric routing.
- **Buffer Overwriting** – Attackers can substitute regular data in specified parts of computer memory on a network device with a barrage of commands that can subsequently be utilized as a part of a network incursion by overwriting certain memory locations.
- **Covert CGI Scripts** – The Common Gateway Interface (CGI), which allows servers to relay user requests to appropriate programs and get data back to then forward to users, unfortunately, provides an easy mechanism for attackers to gain access to network system files.
- **Enormous traffic loads** – Attackers can cause chaos and congestion in network settings by producing traffic loads that are too enormous for systems to fully filter, allowing them to carry out assaults without being discovered.
- **Worms** – The typical, isolated computer virus, or worm, is one of the easiest and most dangerous network penetration tools. Worms, which are commonly distributed by email attachments or instant messaging, use a considerable amount of network resources, preventing permitted activities from taking place.

How Does Intrusion Detection Work?

An intrusion detection system (IDS) is a monitor-only program that detects and reports irregularities in your network architecture before hackers may do damage. IDS can be set up on your network or on a client system (host-based IDS).

Intrusion detection systems often seek known attack signatures or aberrant departures from predetermined standards. These anomalous network traffic patterns are then transmitted up the stack to the OSI (Open Systems Interconnection) model's protocol and application layers for further investigation.

An IDS is a detection system that is positioned outside of the real-time communication band (a channel between the information transmitter and receiver) within your network infrastructure. Instead, it uses a SPAN or TAP

port to watch the network and examines a copy of inline network packets (acquired through port mirroring) to ensure that the streaming traffic is not fraudulent or faked in any manner.

The IDS can readily identify malformed information packets, DNS poisonings, Xmas scans, and other polluted materials, which can have a severe impact on your overall network performance.

Intrusion detection systems employ two detection methods –

- *Signature-based detection* matches data activity to a signature or pattern in a signatures database. A new harmful behavior that is not in the database, for example, is overlooked when using signature-based detection.
- Unlike signature-based detection, *behavior-based detection* recognizes any abnormality and issues alarms, making it capable of identifying new sorts of threats. It's referred to as an expert system since it learns what regular behavior looks like in your system.

PASSWORD MANAGEMENT

Passwords are a set of strings provided by users at the authentication prompts of web accounts. Although passwords still remain as one of the most secure methods of authentication available to date, they are subjected to a number of security threats when mishandled. The role of password management comes in handy there. Password management is a set of principles and best practices to be followed by users while storing and managing passwords in an efficient manner to secure passwords as much as they can to prevent unauthorized access.

What are the challenges in password management?

There are many challenges in securing passwords in this digital era. When the number of web services used by individuals are increasing year-over-year on one end, the number of cyber crimes is also skyrocketing on the other end. Here are a few common threats to protecting our passwords:

- **Login spoofing** - Passwords are illegally collected through a fake login page by cybercriminals.
- **Sniffing attack** - Passwords are stolen using illegal network access and with tools like key loggers.
- **Shoulder surfing attack** - Stealing passwords when someone types them, at times using a micro-camera and gaining access to user data.
- **Brute force attack** - Stealing passwords with the help of automated tools and gaining access to user data.

- **Data breach** - Stealing login credentials and other confidential data directly from the website database.

All of these threats create an opportunity for attackers to steal user passwords and enjoy unlimited access benefits. Let's take a look at how individuals and businesses typically manage their passwords.

Traditional methods of password management

- Writing down passwords on sticky notes, post-its, etc.
- Sharing them via spreadsheets, email, telephone, etc.
- Using simple and easy to guess passwords
- Reusing them for all web applications
- Often forgetting passwords and seeking the help of 'Forgot Password' option

While hackers are equipped with advanced tools and attacks, individuals and businesses still rely on traditional methods of password management. This clearly raises the need for the best password management practices to curb security threats.

How to manage passwords

- Use strong and unique passwords for all websites and applications
- Reset passwords at regular intervals
- Configure two-factor authentication for all accounts
- Securely share passwords with friends, family, and colleagues
- Store all enterprise passwords in one place and enforce secure password policies within the business environment
- Periodically review the violations and take necessary actions.

Virus and related threats

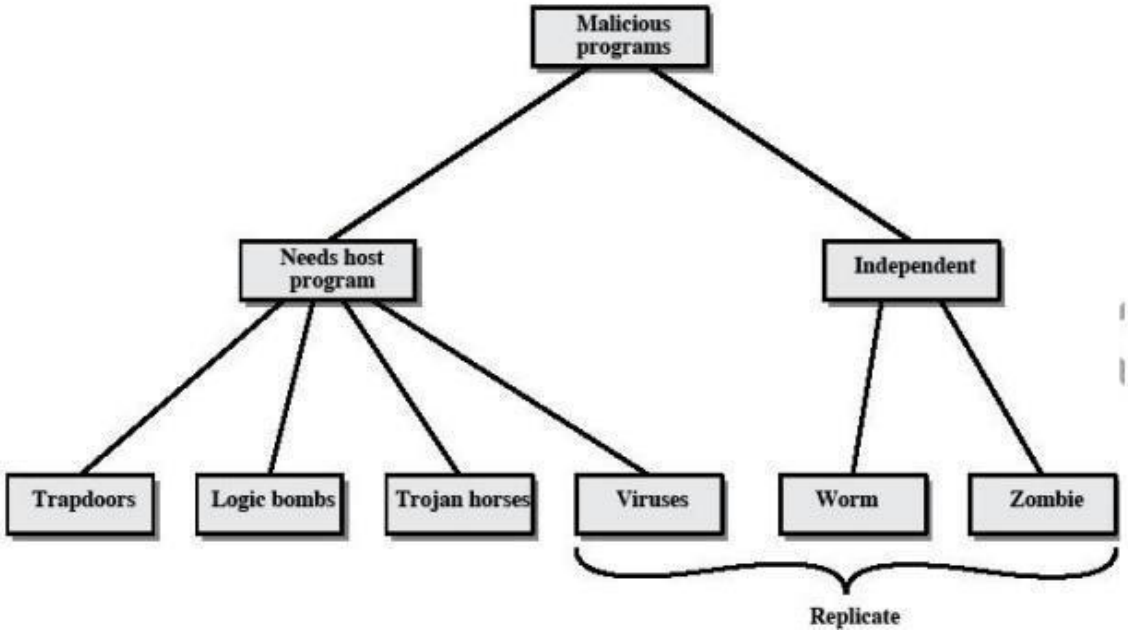
The most sophisticated types of threats to computer systems are presented by programs that exploit vulnerabilities in computing systems.

Malicious Programs

Malicious software can be divided into two categories: those that need a host program, and those that are independent.

The former are essentially fragments of programs that cannot exist independently of some actual application program, utility, or system program. Viruses, logic bombs, and backdoors are examples. The latter are self-contained

programs that can be scheduled and run by the operating system. Worms and zombie programs are examples.



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

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.

A virus can do anything that other programs do. The only difference is that it attaches itself to another program and executes secretly when the host program is run. Once a virus is executing, it can perform any function, such as erasing files and 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.

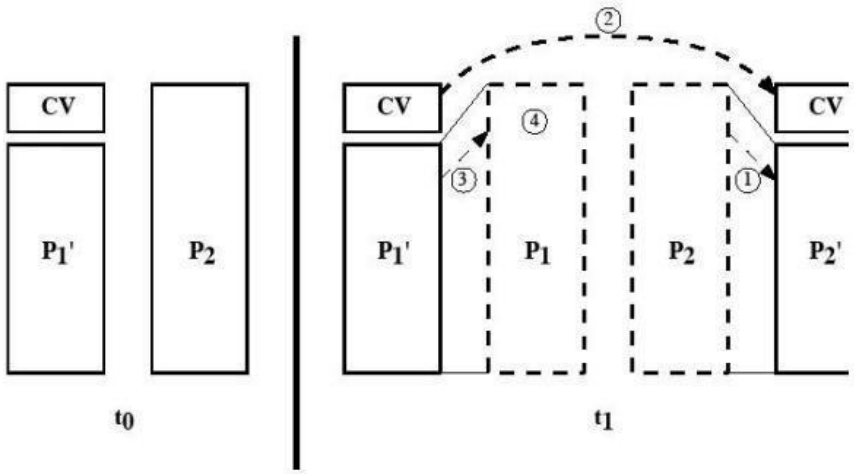
Execution phase: The function is performed. The function may be harmless, such as a message on the screen, or damaging, such as the destruction of programs and data files.

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.

A virus such as the one just described is easily detected because an infected version of a program is longer than the corresponding uninfected one. A way to thwart such a simple means of detecting a virus is to compress the executable file so that both the infected and uninfected versions are of identical length.. The key lines in this virus are numbered. We assume that program P1 is infected with the virus CV. When this program is invoked, control passes to its virus, which performs the following steps:

1. For each uninfected file P2 that is found, the virus first compresses that file to produce P'2, which is shorter than the original program by the size of the virus.

2. A copy of the virus is prepended to the compressed program.

3. The compressed version of the original infected program, P'1, is uncompressed.

4. The uncompressed original program is executed.

FIREWALL DESIGN PRINCIPLES

A **Firewall** is hardware or software to prevent a private computer or a network of computers from, it acts as a filter to avoid unauthorized users from accessing private computers and networks. It is a vital component of network security. It is the first line of defense for network security. It filters network packets and stops malware from entering the user's computer or network by blocking access and preventing the user from being infected.

Characteristics of Firewall

1. **Physical Barrier:** A firewall does not allow any external traffic to enter a system or a network without its allowance. A firewall creates a choke point for all the external data trying to enter into the system or network and hence can easily block the access if needed.
2. **Multi-Purpose:** A firewall has many functions other than security purposes. It configures domain names and Internet Protocol (IP) addresses. It also acts as a network address translator. It can act as a meter for internet usage.
3. **Flexible Security Policies:** Different local systems or networks need different security policies. A firewall can be modified according to the requirement of the user by changing its security policies.
4. **Security Platform:** It provides a platform from which any alert to the issue related to security or fixing issues can be accessed. All the queries related to security can be kept under check from one place in a system or network.
5. **Access Handler:** Determines which traffic needs to flow first according to priority or can change for a particular network or system. specific action requests may be initiated and allowed to flow through the firewall.

Need and Importance of Firewall Design Principles

1. **Different Requirements:** Every local network or system has its threats and requirementswhich needs different structure and devices. All this can only be identified while designing a firewall. Accessing the current security outline of a company can help to create a better firewall design.
2. **Outlining Policies:** Once a firewall is being designed, a system or network doesn't need to be secure. Some new threats can arise and if we have proper paperwork of policies then the security system can be modified again and the network will become more secure.
3. **Identifying Requirements:** While designing a firewall data related to threats, devices needed to be integrated, Missing resources, updating the security devices. All the information collected is combined to get the best results. Even if one of these things is misidentified leads to security issues.
4. **Setting Restrictions:** Every user has its limitations to access different level of data or modify it and it needed to be identified and taken action

accordingly. After retrieving and processing data, priority is set to people, devices, and applications.

5. **Identify Deployment Location:** Every firewall has its strengths and to get the most use out of it, we need to deploy each of them at the right place in a system or network. In the case of a packet filter firewall, it needs to be deployed at the edge of your network in between the internal network and webserver to get the most out of it.

Firewall Design Principles

1. Developing Security Policy

Security policy is a very essential part of firewall design. Security policy is designed according to the requirement of the company or client to know which kind of traffic is allowed to pass. Without a proper security policy, it is impossible to restrict or allow a specific user or worker in a company network or anywhere else. A properly developed security policy also knows what to do in case of a security breach. Without it, there is an increase in risk as there will not be a proper implementation of security solutions.

2. Simple Solution Design

If the design of the solution is complex.then it will be difficult to implement it. If the solution is easy.then it will be easier to implement it. A simple design is easier to maintain. we can make upgrades in the simple design according to the new possible threats leaving it with an efficient but more simple structure. The problem that comes with complex designs is a configuration error that opens a path for external attacks.

3. Choosing the Right Device

Every network security device has its purpose and its way of implementation. if we use the wrong device for the wrong problem, the network becomes vulnerable. if the outdated device is used for a designing firewall, it exposes the network to risk and is almost useless. Firstly the designing part must be done then the product requirements must be found out, if the product is already available then it is tried to fit in a design that makes security weak.

4. Layered Defense

A network defense must be multiple layered in the modern world because if the security is broken, the network will be exposed to external attacks. Multilayer security design can be set to deal with different levels of threat. It gives an edge to the security design and finally neutralizes the attack over the system.

5. Consider Internal Threats

While giving a lot of attention to safeguarding the network or device from external attacks. The security becomes weak in case of internal attacks and most of the attacks are done internally as it is easy to access and designed weakly. Different levels can be set in network security while designing internal security. Filtering can be added to keep track of the traffic moving from lower-level security to higher level.

TYPES OF FIREWALL

The major purpose of the network firewall is to protect an inner network by separating it from the outer network. Inner Network can be simply called a network created inside an organization and a network that is not in the range of inner network can be considered as Outer Network.

Types of Firewall:

Packet Filters

It is a technique used to control network access by monitoring outgoing and incoming packets and allowing them to pass or halt based on the source and destination Internet Protocol (IP) addresses, protocols, and ports. This firewall is also known as a static firewall.

Stateful Inspection Firewalls

It is also a type of packet filtering which is used to control how data packets move through a firewall. It is also called dynamic packet filtering. These firewalls can inspect that if the packet belongs to a particular session or not. It only permits communication if and only if, the session is perfectly established between two endpoints else it will block the communication.

Application Layer Firewalls

These firewalls can examine application layer (of OSI model) information like an HTTP request. If finds some suspicious application that can be responsible for harming our network or that is not safe for our network then it gets blocked right away.

Next-generation Firewalls

These firewalls are called intelligent firewalls. These firewalls can perform all the tasks that are performed by the other types of firewalls that we learned previously but on top of that, it includes additional features like application

awareness and control, integrated intrusion prevention, and cloud-delivered threat intelligence.

Circuit-level gateways

A circuit-level gateway is a firewall that provides User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) connection security and works between an Open Systems Interconnection (OSI) network model's transport and application layers such as the session layer.

Software Firewall

The software firewall is a type of computer software that runs on our computers. It protects our system from any external attacks such as unauthorized access, malicious attacks, etc. by notifying us about the danger that can occur if we open a particular mail or if we try to open a website that is not secure.

Hardware Firewall

A hardware firewall is a physical appliance that is deployed to enforce a network boundary. All network links crossing this boundary pass-through this firewall, which enables it to perform an inspection of both inbound and outbound network traffic and enforce access controls and other security policies.

Cloud Firewall

These are software-based, cloud-deployed network devices. This cloud-based firewall protects a private network from any unwanted access. Unlike traditional firewalls, a cloud firewall filters data at the cloud level.

Advantages of Network Firewall :

- **Monitors network traffic**
  A network firewall monitors and analyzes traffic by inspecting whether the traffic or packets passing through our network is safe for our network or not. By doing so, it keeps our network away from any malicious content that can harm our network.

- **Halt Hacking**

  In a society where everyone is connected to technology, it becomes more important to keep firewalls in our network and use the internet safely.

- **Stops viruses**

Viruses can come from anywhere, such as from an insecure website, from a spam message, or any threat, so it becomes more important to have a strong defense system (i.e. firewall in this case), a virus attack can easily shut off a whole network. In such a situation, a firewall plays a vital role.

- **Better security**
  If it is about monitoring and analyzing the network from time to time and establishing a malware-free, virus-free, spam-free environment so network firewall will provide better security to our network.
- **Increase privacy**
  By protecting the network and providing better security, we get a network that can be trusted.

Disadvantages of Network Firewall :

- **Cost**
  Depending on the type of firewall, it can be costly, usually, the hardware firewalls are more costly than the software ones.
  Restricts User
  Restricting users can be a disadvantage for large organizations, because of its tough security mechanism. A firewall can restrict the employees to do a certain operation even though it's a necessary operation.
- **Issues with the speed of thenetwork**
  Since the firewalls have to monitor every packet passing through the network, this can slow down operations needed to be performed, or it can simply lead to slowing down the network.
- **Maintenance**
  Firewalls require continuous updates and maintenance with every change in the networking technology. As the development of new viruses is increasing continuously that can damage your system.

CASE STUDIES ON CRYPTOGRAPHY AND NETWORK SECURITY

Secure Inter-branch Payment

General Bank Of India (GBI) has implemented an Electronic Payment System called as EPS in about1200 branches across the country. This system transfers payment instructions between twocomputerized branches of GBI. A central server is maintained at the EPS office located in Mumbai. Thebranch offices connect to the Local VSAT of a private network by using dial-up connection. The localVSAT has a connectivity established with the EPS office. GBI utilizes its

proprietary messaging servicecalled as GBI-Transfer to exchange payment instructions.

Currently, EPS has minimal data security. As the system operates in a closed network, the currentsecurity infrastructure may suffice the need. The data moving across the network is in encryptedformat.

Current EPS Architecture EPS is used to transmit payment details from the payer branch to thepayee branch via the central server in Mumbai. Fig. 10.5 depicts the flow, which is also described step-by-step

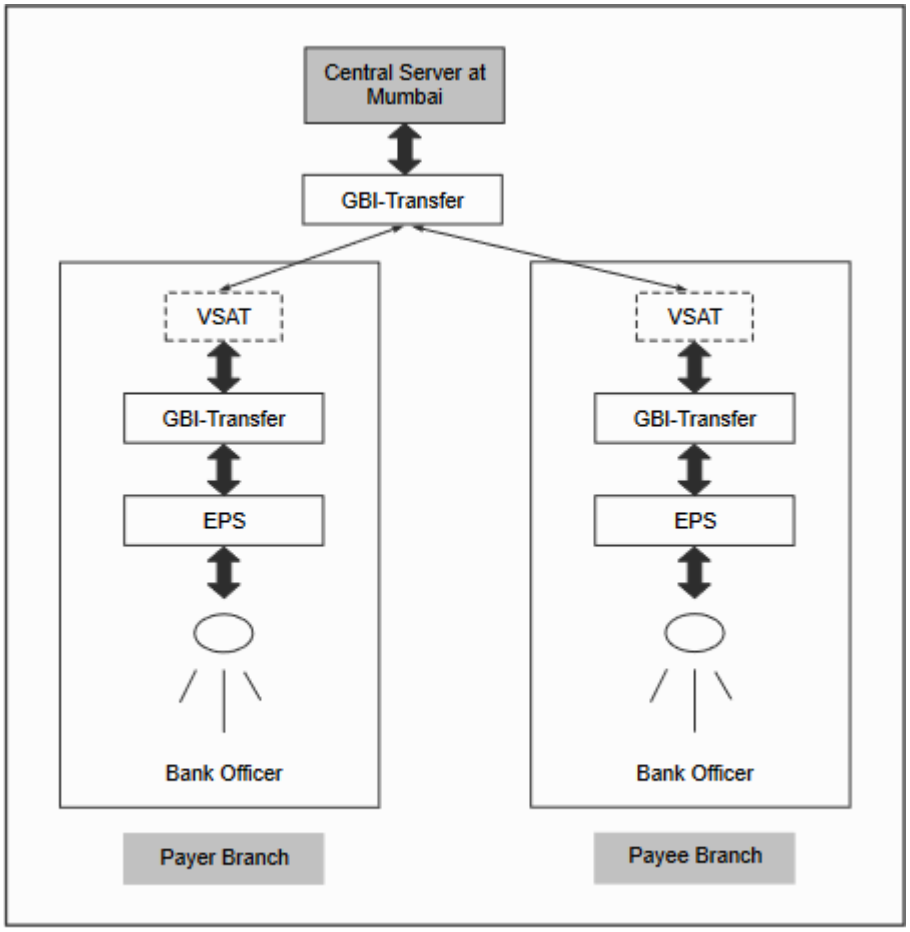


Fig. 10.5 *EPS transaction flow*

A typical payment transfer takes the following steps:

1. A data-entry person in the Payer Branch enters transaction details through the EPS interface.
2. A Bank Officer checks the validity of the transaction through the EPS interface.
3. After validating the transaction, the Bank Officer authorizes the transaction. Authorizedtransaction is stored in a local Payment Master (PM) database.

4. Once the transaction is stored in PM, a copy of the same is encrypted and stored in a file. This transaction file is stored in OUT directory.
5. The GBI-Transfer application looks for any pending transactions (i.e. for the presence of any files in the OUT directory) by a polling mechanism and if it finds such transactions, it sends all these files one-by-one to the EPS central office located in Mumbai by dialing the local VSAT.
6. The local VSAT gets connectivity to the EPS central office and the transaction is transferred and stored in the IN directory at the EPS central office.
7. The interface program at the EPS central office collects the file pending in the IN directory and sends it to the PM application at that office.
8. In order to send the Credit Request to PM, the transaction headers are changed. The transaction with changed headers in encrypted format is then placed in OUT directory of the EPS central office.
9. The GBI-Transfer application at the EPS central office collects the transactions pending in the OUT directory and sends them to the Payee Bank through the VSAT.
10. The transaction is transferred and stored in the IN directory of the Payee Branch.
11. The interface program at the Payee Branch collects the transaction and posts it in PM.
12. PM marks the credit entry and returns back an acknowledgement of the same. The acknowledgement is placed in OUT directory of the Payee Branch.
13. The acknowledgement is picked by GBI-Transfer at the Payee Branch and sent to the EPS central office through the VSAT.
14. The EPS central office receives the credit acknowledgement and forwards it to Payer Branch.
15. The Payer Branch receives the credit acknowledgement receipt. This completes the transaction.

Requirements to Enhance EPS As GBI is in the process of complete automation and setting u connectivity over the Internet or a private network, they need to ensure stringent security measures, which demand the usage of a Public Key Infrastructure (PKI) framework.

As a part of implementing security, GBI wants the following aspects to be ensured:

- Non-repudiation (Digital Signatures)
- Encryption – 128-bit (Upgrade to the current 56-bit encryption)
- Smart card support for storing sensitive data & on-card digital signing
- Closed loop Public Key Infrastructure

The architecture for the Payer

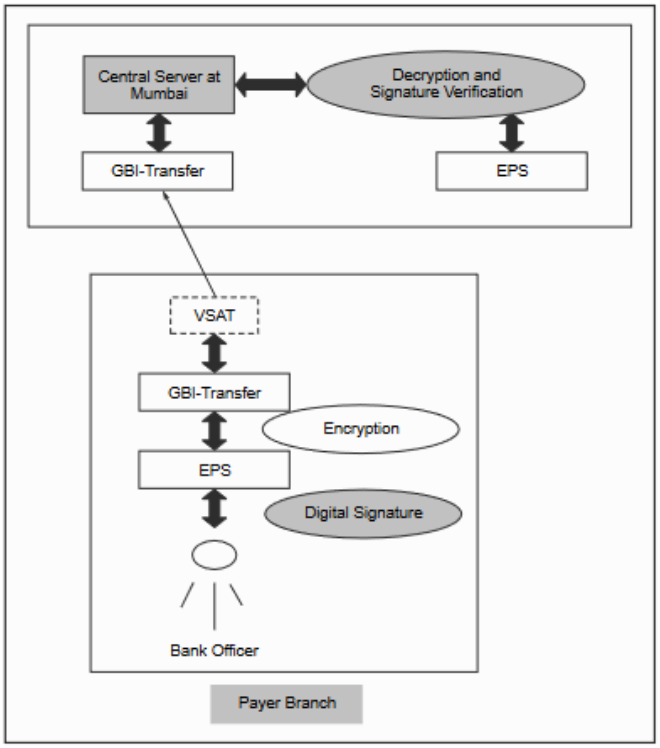


Fig. 10.6 *New EPS transaction flow at the Payer Branch*

On the Payee Leg, the EPS central office will create a Credit Request as before, sign and encrypt itwith the bank officer's digital certificate. This signed-and-encrypted request will be forwarded to thePayee Branch.

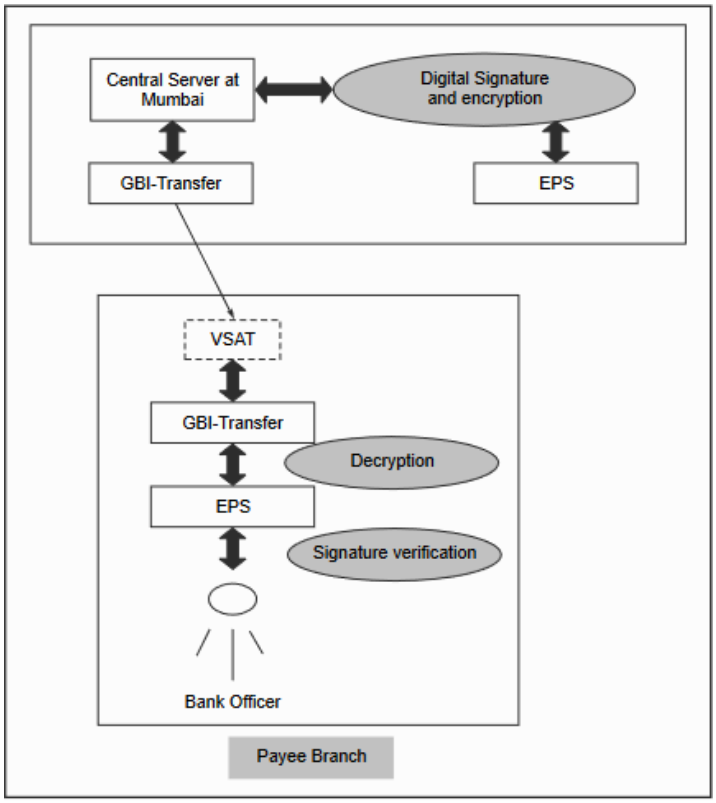


Fig. 10.7 *New EPS transaction flow at the Payee Branch*

CROSS SITE SCRIPTING VULNERABILITY (CSSV)

Cross Site Scripting Vulnerability (CSSV) is a relatively new form of attacks that exploits inadequatevalidations on the server-side. The term Cross Server Scripting Vulnerability (CSSV) is actually notcompletely correct. However, this term was coined when the problem was not completely understoodand has stuck ever since. Cross-site scripting happens when malicious tags and/or scripts attack a Webbrowser via another site's dynamically generated Web pages. The attacker's target is not a Website, butrather its users (i.e. clients or browsers). Suppose that the URL of the site sending this page is www.test.com and when the usersubmits this form, it would be processed by a server-side program called as address.asp.We would typically expect the user to enter the house number, street name, city, postal code andcountry, etc. However, imagine that the user enters the following weird string, instead:

<SCRIPT>Hello World</SCRIPT>

Fig. 10.9 *Sample HTML form*

As a result, the URL submitted would be something like www.test.com/address.asp?address=<SCRIPT>Hello World </SCRIPT>.

Now suppose that the server-side program address.asp does not validate the input sent by the userand simply sends the value of the field address to the next Web page. What would this translate to? Itwould mean that the next Web page would receive the value of address as <SCRIPT>Hello World</SCRIPT>.

As we know, this would most likely treat the value of the address field as a script, which would beexecuted as if it is written in a scripting language, such as JavaScript etc on the Web browser.Therefore, the user would get to see Hello World.

Obviously, no serious damage is done. However, extrapolate this possibility to other situations wherea user can actually send damaging scripts to the server.

VIRTUAL ELECTIONS

Computerized voting wouldbecome quite common in the next few decades. As such, it is important that the protocol for virtualelections should protect individual privacy and should also disallow cheating. Consider the followingprotocol in order that voters can send their votes electronically to the Election Authority (EA).

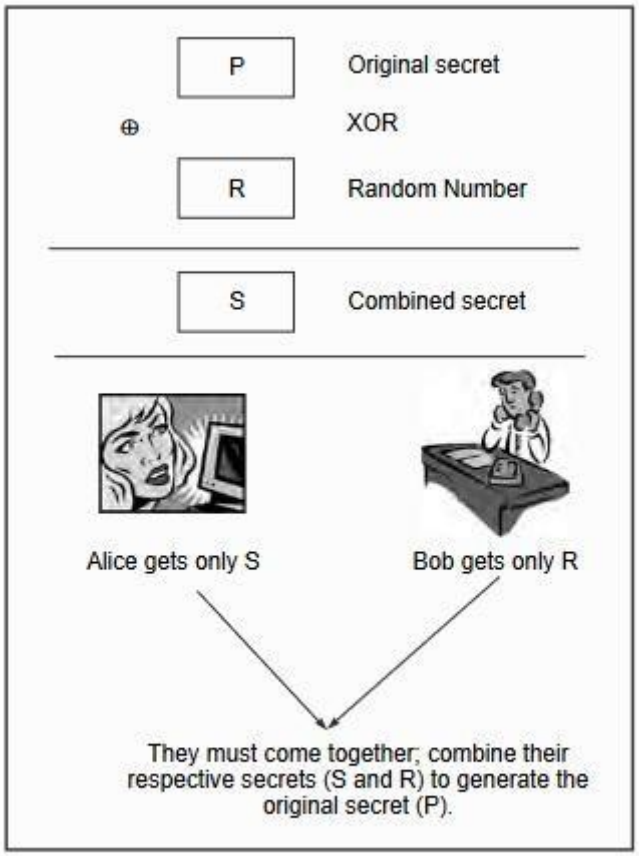


Fig. 10.10 *Secret splitting*

1. Each voter casts the vote and encrypts it with the public key of the EA.
2. Each voter sends the encrypted vote to the EA.
3. The EA decrypts all the votes to retrieve the original vote, tabulates all the votes and announces the result of the election.

Is this protocol secure and does it provide comfort both to the voters as well as to the EA? Not at all!There are following problems in this scheme:

1. The EA does not know whether the authorized voters have voted or it has received fake (bogus)votes.
2. Secondly, there is no mechanism to prevent duplicate voting.

What is the advantage of this protocol? Clearly, no one would be able to change another voter's vote,because it is first encrypted with the EA's public key and is then sent to the EA. However, if we observethis scheme carefully, an attacker need not change someone's vote at all. The attacker can simply send duplicate votes!

How can we improve upon this protocol to make it more robust? Let us rewrite it, as follows:

1. Each voter casts the vote and signs it with her private key
2. Each voter then encrypts the signed vote with the public key of the EA.

3. Each voter sends the vote to the EA.
4. The EA decrypts the voter with its private key and verifies the signature of the voter with the helpof the voter's public key.
5. The EA then tabulates all the votes and announces the result of the election

This protocol would now ensure that duplicate voting is disallowed.