LECTURE

NOTES

ON

INFORMATION SECURITY

Computer Science
&
Engineering

UNIT – I

Computer Security concepts, The OSI Security Architecture, Security attacks, Security services and Security mechanisms, A model for Network Security Classical encryption techniques- symmetric cipher model, substitution ciphers, transposition ciphers, Steganography. Modern Block Ciphers: Block ciphers principles, Data encryption standard (DES), Strength of DES, linear and differential cryptanalysis, block cipher modes of operations, AES, RC4.

Introduction:

Computer Security concepts:

What is computer security?

Computer security is refers to techniques for ensuring that data stored in a computer cannot be read or compromised by any individuals without authorization.

Most computer security measures involve data encryption and passwords.
We are addressing three important aspects of any computer-related system such as confidentiality, integrity, and availability.

Confidentiality: ensures that computer-related assets are accessed only by authorized parties. Confidentiality is sometimes called secrecy or privacy.

Integrity: it means that assets can be modified only by authorized parties or only in authorized ways.

Availability: it means that assets are accessible

Vulnerability

Vulnerability is a weakness in the security system.
Weaknesses can appear in any element of a computer, both in the hardware, operating system, and the software. The types of vulnerabilities we might find as they apply to the assets of hardware, software, and data

Threats

A threat to a computing system is a set of circumstances that has the potential to cause loss or harm. •There are many threats to a computer system, including human-initiated and computer- initiated ones. •A threat is blocked by control of a vulnerability.

We can view any threat as being one of four

1. An interception means that some unauthorized party has gained access to an asset. The outside party can be a person, a program, or a computing system.
2. In an interruption is an asset of the system becomes lost, unavailable, or unusable.
3. If an unauthorized party not only accesses but tampers with an asset, is called as a modification.
4. An unauthorized party might create a fabrication of counterfeit objects on a computing system. The intruder may insert spurious transactions to a network communication system or add records to an existing database.

The OSI Security Architecture

To assess effectively the security needs of an organization and to evaluate and choose various security products and policies, the manager responsible for security needs some systematic way of defining the requirements for security and characterizing the approaches to satisfying those requirements. This is difficult enough in a centralized data processing environment; with the use of local and wide area networks, the problems are compounded.

Network Security: It can be defined as "measures adopted to prevent the unauthorized use, misuse, modification or denial of use of knowledge, facts, data or capabilities". Three aspects of IS are:

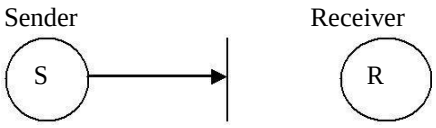
- **Security Attack:**
  Any action that comprises the security of information
- **Security Mechanism:**
  A mechanism that is designed to detect, prevent, or recover from a security.
- **Security Service:**
  It is a processing or communication service that enhances the security of the data processing systems and information transfer. The services are intended to counter

Security Attacks

Security attacks can be classified in terms of Passive attacks and Active attacks as per X.800 and RFC 2828

Different kinds of attacks are:

Interruption

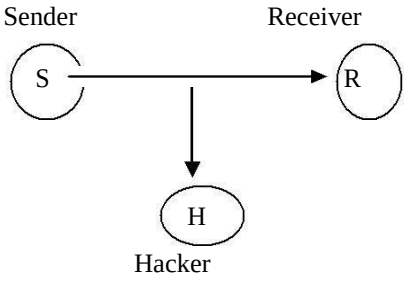


An asset of the system is destroyed or becomes unavailable or unusable. It is an attack on availability

Examples:

i. Destruction of some hardware
ii. Jamming wireless signals
iii. Disabling file management systems

Interception

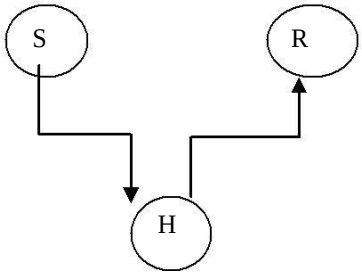


An unauthorized party gains access to an asset. Attack on confidentiality.

Examples:

i. Wire tapping to capture data in a network.
ii. Illicitly copying data or programs
iii. Eavesdropping

Modification:

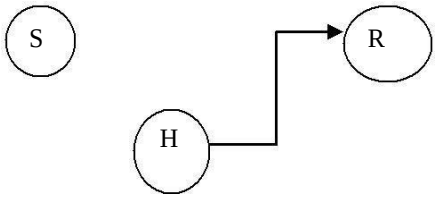


When an unauthorized party gains access and tampers an asset. Attack is on Integrity.

Examples:

i. Changing data file
ii. Altering a program and the contents of a message

Fabrication



An unauthorized party inserts a counterfeit object into the system. Attack on Authenticity. Also called impersonation

Examples:

i. Hackers gaining access to a personal email and sending message
ii. Insertion of records in data files
iii. Insertion of spurious messages in a network

Passive Attacks

A Passive attack attempts to learn or make use of information from the system, but does not affect system resources.

Two types:

Release of message content

It may be desirable to prevent the opponent from learning the contents (i.e sensitive or confidential info) of the transmission

Traffic analysis

A more subtle technique where the opponent could determine the location and identity of communicating hosts and could observe the frequency & length of encrypted messages being exchanged there by guessing the nature of communication taking place.

Passive attacks are very difficult to detect because they do not involve any alternation of the data. As the communications take place in a very normal fashion, neither the sender nor receiver is aware that a third party has read the messages or observed the traffic pattern. So, the emphasis in dealing with passive attacks is on prevention rather than detection.

Active Attacks

Active attacks involve some modification of the data stream or creation of a false stream. An active attack attempts to alter system resources or affect their operation.

Four types:

- **Masquerade:** Here, an entity pretends to be some other entity. It usually includes one ofthe other forms of active attack.
- **Replay:** It involves the passive capture of a data unit and its subsequent retransmission toproduce an unauthorized effect.
- **Modification of messages:** It means that some portion of a legitimate message is altered, orthat messages are delayed to produce an unauthorized effect.

  Ex: “John’s acc no is 2346” is modified as “John’s acc no is 7892”
- **Denial of service:** This attack prevents or inhibits the normal use or management ofcommunication facilities.

  Ex: a: Disruption of entire network by disabling it

  b: Suppression of all messages to a particular destination by a third party.

Security Services:

It is a processing or communication service that is provided by a system to give a specific kind of production to system resources. Security services implement security policies and are implemented by security mechanisms.

Confidentiality

Confidentiality is the protection of transmitted data from passive attacks. It is used to prevent the disclosure of information to unauthorized individuals or systems.".

The other aspect of confidentiality is the protection of traffic flow from analysis. **Ex:** A credit card number has to be secured during online transaction.

Authentication

This service assures that a communication is authentic. For a single message transmission, its function is to assure the recipient that the message is from intended source. Two specific authentication services defines in X.800 are

- **Peer entity authentication:** Verifies the identities of the peer entities involved in communication. Provides use at time of connection establishment and during data transmission. Provides confidence against a masquerade or a replay attack
- **Data origin authentication:** Assumes the authenticity of source of data unit, but does not provide protection against duplication or modification of data units. Supports applications like electronic mail, where no prior interactions take place between communicating entities.

Integrity

Integrity means that data cannot be modified without authorization two types of integrity services are available. They are

- **Connection-Oriented Integrity Service:** This service deals with a stream of messages, assures that messages are received as sent, with no duplication, insertion, modification, reordering or replays.
- **Connectionless-Oriented Integrity Service:** It deals with individual messages regardless of larger context, providing protection against message modification only.

An integrity service can be applied with or without recovery. Because it is related to active attacks, major concern will be detection rather than prevention. If a violation is detected and the service reports it, either human intervention or automated recovery machines are required to recover.

Access Control

This refers to the ability to control the level of access that individuals or entities have to a network or system and how much information they can receive..

Availability

It is defined to be the property of a system or a system resource being accessible and usable upon demand by an authorized system entity.

Security Mechanisms:

According to X.800, the security mechanisms are divided into those implemented in a specific protocol layer and those that are not specific to any particular protocol layer or security service. X.800 also differentiates reversible & irreversible encipherment mechanisms. A reversible encipherment mechanism is simply an encryption algorithm that allows data to be encrypted and subsequently decrypted, where as irreversible encipherment include hash algorithms and message authentication codes used in digital signature and message authentication applications

- **Specific Security Mechanisms:**

  Incorporated into the appropriate protocol layer in order to provide some of the OSI security services,

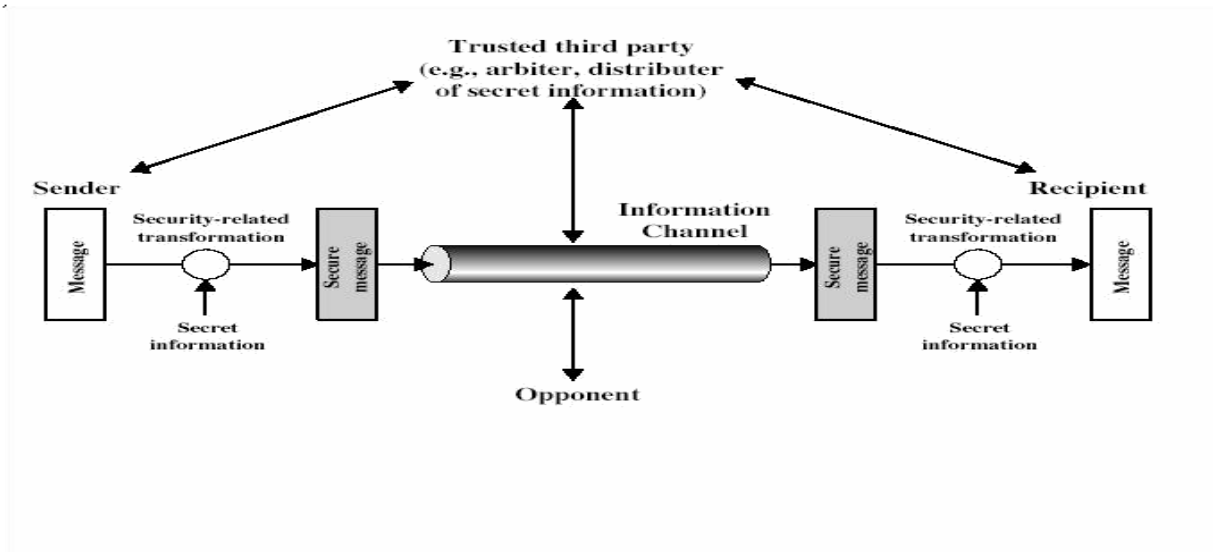
  - **Encipherment:** It refers to the process of applying mathematical algorithms forconverting data into a form that is not intelligible. This depends on algorithm used and encryption keys.
  - **Digital Signature:** The appended data or a cryptographic transformation applied to anydata unit allowing to prove the source and integrity of the data unit and protect against forgery.
  - **Access Control:** A variety of techniques used for enforcing access permissions to thesystem resources.
  - **Data Integrity:** A variety of mechanisms used to assure the integrity of a data unit orstream of data units.
  - **Authentication Exchange:** A mechanism intended to ensure the identity of an entity by means of information exchange
  - **Traffic Padding:** The insertion of bits into gaps in a data stream to frustrate trafficanalysis attempts.
  - **Routing Control:** Enables selection of particular physically secure routes for certain dataand allows routing changes once a breach of security is suspected.
  - **Notarization:** The use of a trusted third party to assure certain properties of a dataexchange

- **Pervasive Security Mechanisms:**

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

  - **Trusted Functionality:** That which is perceived to b correct with respect to some criteria
  - **Security Level:** The marking bound to a resource (which may be a data unit) that namesor designates the security attributes of that resource.
  - **Event Detection:** It is the process of detecting all the events related to network security.
  - **Security Audit Trail:** Data collected and potentially used to facilitate a security audit,which is an independent review and examination of system records and activities.
  - **Security Recovery:** It deals with requests from mechanisms, such as event handling andmanagement functions, and takes recovery actions.

A Model Of Inter Network Security



Data is transmitted over network between two communicating parties, who must cooperate for the exchange to take place. A logical information channel is established by defining a route through the internet from source to destination by use of communication protocols by the two parties. Whenever an opponent presents a threat to confidentiality, authenticity of information, security aspects come into play. Two components are present in providing security

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

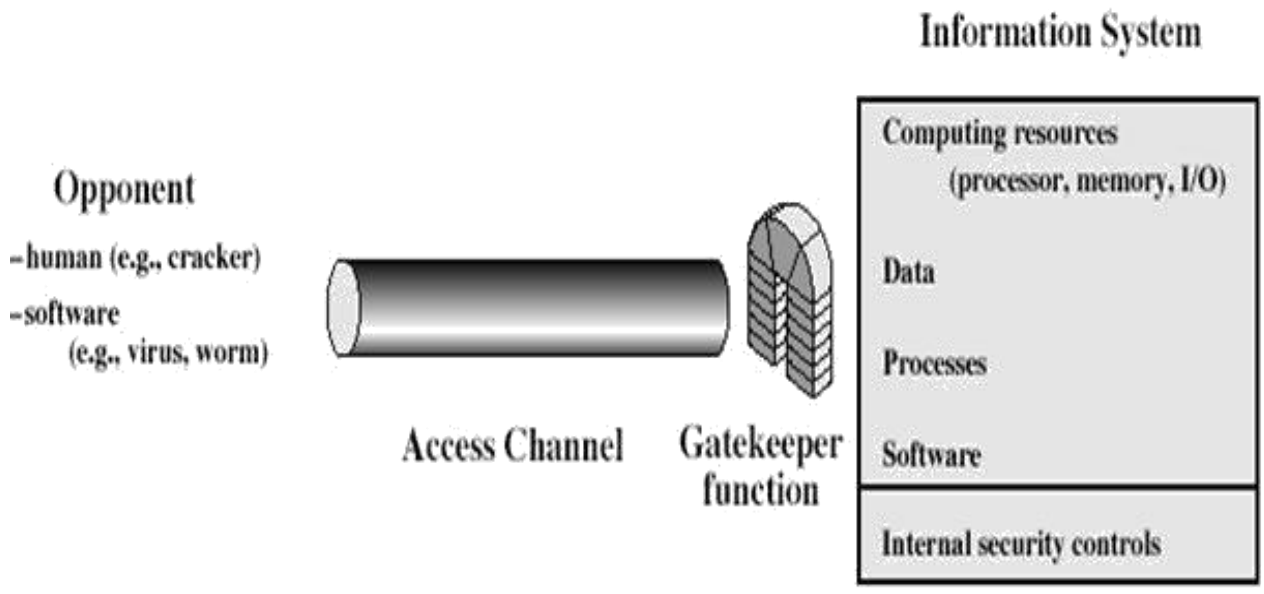
A trusted third party may be needed to achieve secure transmission. It is responsible for distributing the secret information to the two parties, while keeping it away from any opponent. It also may be needed to settle disputes between the two parties regarding authenticity of a message transmission. The general model shows that there are four basic tasks in designing a particular security service:

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

Threat is placement of some logic in computer system affecting various applications and utility programs. This inserted code presents two kinds of threats.

- **Information access threats** intercept or modify data on behalf of users who should nothave access to that data
- **Service threats** exploit service flaws in computers to inhibit use by legitimate users

Viruses and worms are two examples of software attacks inserted into the system by means of a disk or also across the network. The security mechanisms needed to cope with unwanted access fall into two broad categories



- Placing a gatekeeper function, which includes a password-based login methods that provide access to only authorized users and screening logic to detect and reject worms, viruses etc
- An internal control, monitoring the internal system activities analyzes the stored information and detects the presence of unauthorized users or intruders.

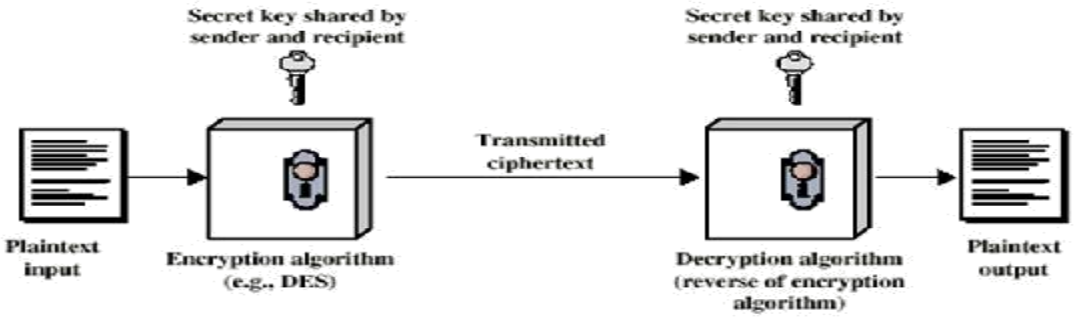
Conventional Encryption principles

A Symmetric encryption scheme has five ingredients

1. ***Plain Text***: This is the original message or data which is fed into the algorithm as input.

2. ***Encryption Algorithm***: This encryption algorithm performs various substitutions and transformations on the plain text.

3. ***Secret Key***: The key is another input to the algorithm. The substitutions and transformations performed by algorithm depend on the key.



Simplified Model of Conventional Encryption

4. ***Cipher Text*:** This is the scrambled (unreadable) message which is output of the encryption algorithm. This cipher text is dependent on plaintext and secret key. For a given plaintext, two different keys produce two different cipher texts.

5. ***Decryption Algorithm***: This is the reverse of encryption algorithm. It takes the cipher text and secret key as inputs and outputs the plain text.

Two main requirements are needed for secure use of conventional encryption:

(i). A strong encryption algorithm is needed. It is desirable that the algorithm should be in such a way that, even the attacker who knows the algorithm and has access to one or more cipher texts would be unable to decipher the cipher text or figure out the key.

(ii). The secret key must be distributed among the sender and receiver in a very secured way. If in any way the key is discovered and with the knowledge of algorithm, all communication using this key is readable.

Cryptography

A cipher is a secret method of writing, as by code. **Cryptography**, in a very broad sense, is the study of techniques related to aspects of information security. Hence cryptography is concerned with the writing (ciphering or encoding) and deciphering (decoding) of messages in secret code. Cryptographic systems are classified along three independent dimensions:

- ***The type of operations used for performing plaintext to ciphertext***

  All the encryption algorithms make use of two general principles; substitution and transposition through which plaintext elements are rearranged. Important thing is that no information should be lost.

- ***The number of keys used***

  If single key is used by both sender and receiver, it is called symmetric, single-key, secret-key or conventional encryption. If sender and receiver each use a different key, then it is called asymmetric, two-key or public-key encryption.

- ***The way in which plaintext is processed***

  A block cipher process the input as blocks of elements and generated an output block for each input block. Stream cipher processes the input elements continuously, producing output one element at a time as it goes along.

Cryptanalysis

The process of attempting to discover the plaintext or key is known as cryptanalysis. It is very difficult when only the cipher text is available to the attacker as in some cases even the encryption algorithm is not known. The most common attack under these circumstances is brute-force approach of trying all the possible keys. This attack is made impractical when the key size is considerably large. The table below gives an idea on types of attacks on encrypted messages.

Type of Attack	Known to Cryptanalyst
Ciphertext only	• Encryption algorithm • Ciphertext to be decoded
Known plaintext	• Encryption algorithm • Ciphertext to be decoded • One or more plaintext-ciphertext pairs formed with the secret key
Chosen plaintext	• Encryption algorithm • Ciphertext to be decoded • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key
Chosen ciphertext	• Encryption algorithm • Ciphertext to be decoded • Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key
Chosen text	• Encryption algorithm • Ciphertext to be decoded • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key • Purported ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key

Cryptology covers both cryptography and cryptanalysis. *Cryptology* is a constantly evolving science; ciphers are invented and, given time, are almost certainly breakable. *Cryptanalysis* is the best way to understand the subject of cryptology. *Cryptographers* are constantly searching for the perfect security system, a system that is both fast and hard and a system that encrypts quickly but is hard or impossible to break. *Cryptanalysts* are always looking for ways to break the security provided by a cryptographic system, mostly though mathematical understanding of the cipher structure.

Cryptography can be defined as the conversion of data into a scrambled code that can be deciphered and sent across a public or a private network.

- A ***Ciphertext-only attack*** is an attack with an attempt to decrypt ciphertext when only the ciphertext itself is available.
- A ***Known-plaintext attack*** is an attack in which an individual has the plaintext samples and its encrypted version (ciphertext) thereby allowing him to use both to reveal further secret information like the key
- A ***Chosen- plaintext attack*** involves the cryptanalyst be able to define his own plaintext, feed it into the cipher and analyze the resulting ciphertext.
- A ***Chosen-ciphertext attack*** is one, where attacker has several pairs of plaintext-ciphertext and ciphertext chosen by the attacker.

An encryption scheme is **unconditionally secure** if the ciphertext generated by the scheme does not contain enough information to determine uniquely the corresponding plaintext, no matter how much ciphertext and time is available to the opponent. Example for this type is One-time Pad.

An encryption scheme is **computationally secure** if the ciphertext generated by the scheme meets the following criteria:

- Cost of breaking cipher exceeds the value of the encrypted information.
- Time required to break the cipher exceeds the useful lifetime of the information. The average time required for exhaustive key search is given below:

Key Size (bits)	*Number of Alternative Keys*	*Time required at 1 decryption/µs*	*Time required at 10^6 decryptions/µs*
32	**2^{32} = 4.3 10^9**	**2^{31} µs = 35.8 minutes**	**2.15 milliseconds**
56	**2^{56} = 7.2 10^{16}**	**2^{55} µs = 1142 years**	**10.01 hours**
128	**2^{128} = 3.4 10^{38}**	**2^{127} µs = 5.4 10^{24} years**	**5.4 10^{18} years**
168	**2^{168} = 3.7 10^{50}**	**2^{167} µs = 5.9 10^{36} years**	**5.9 10^{30} years**

Substitution Ciphers

A substitution technique is one in which the letters of plaintext are replaced by other letters or by numbers or symbols.[1] If the plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext bit patterns with ciphertext bit patterns.

Caesar Cipher

The earliest known use of a substitution cipher, and the simplest, was by Julius Caesar. The Caesar cipher involves replacing each letter of the alphabet with the letter standing three places further down the alphabet.
For example,
plain: meet me after the toga party
cipher: PHHW PH DIWHU WKH WRJD SDUWB

Note that the alphabet is wrapped around, so that the letter following Z is A.
We can define the transformation by listing all possibilities, as follows:

plain: a b c d e f g h i j k l m n o p q r s t u v w x y z
cipher: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

Let us assign a numerical equivalent to each letter:
a b c d e f g h i j k l m

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

Then the algorithm can be expressed as follows.
For each plaintext letter p, substitute the ciphertext letter C

C = E(3, p) = (p + 3) mod 26
A shift may be of any amount,
so that the general Caesar algorithm is C = E(k, p) = (p + k) mod 26
where k takes on a value in the range 1 to 25.
The decryption algorithm is simply
p = D(k, C) = (C k) mod 26
If it is known that a given ciphertext is a Caesar cipher, then a brute-force cryptanalysis is easily performed: Simply try all the 25 possible keys. Figure shows the results of applying this strategy to the example ciphertext. In this case, the plaintext leaps out as occupying the third line.

Monoalphabetic Ciphers

With only 25 possible keys, the Caesar cipher is far from secure. A dramatic increase in the key space can be achieved by allowing an arbitrary substitution. Recall the assignment for the Caesar cipher:

plain: a b c d e f g h i j k l m n o p q r s t u v w x y z
cipher: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

If, instead, the "cipher" line can be any permutation of the 26 alphabetic characters, then there are 26! or greater than 4 x 1026 possible keys. This is 10 orders of magnitude greater than the key space for DES and would seem to eliminate brute-force techniques for cryptanalysis. Such an approach is referred to as a monoalphabetic substitution cipher, because a single cipher alphabet (mapping from plain alphabet to cipher alphabet) is used per message.
There is, however, another line of attack. If the cryptanalyst knows the nature of the plaintext (e.g., noncompressed English text), then the analyst can exploit the regularities of the language. To see how such a cryptanalysis might proceed, we give a partial example here . The ciphertext to be solved is

UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ
VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX
EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ

As a first step, the relative frequency of the letters can be determined and compared to a standard frequency distribution for English, such as is shown in Figure (based on [LEWA00]). If the message were long enough, this technique alone might be sufficient, but because this is a relatively short message, we cannot expect an exact match. In any case, the relative frequencies of the letters in the ciphertext (in percentages) are as follows:

P 13.33	H 5.83	F 3.33	B 1.67	C 0.00
Z 11.67	D 5.00	W 3.33	G 1.67	K 0.00
S 8.33	E 5.00	Q 2.50	Y 1.67	L 0.00
U 8.33	V 4.17	T 2.50	I 0.83	N 0.00
O 7.50	X 4.17	A 1.67	J 0.83	R 0.00
M 6.67				

Fig: Relative Frequency of Letters in English Text

Eg:
UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ
t a e e te a that e e a a
VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX

EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ

Only four letters have been identified, but already we have quite a bit of the message. Continued analysis of frequencies plus trial and error should easily yield a solution from this point. The complete plaintext, with spaces added between words, follows:

it was disclosed yesterday that several informal but
direct contacts have been made with political
representatives of the viet cong in moscow

Monoalphabetic ciphers are easy to break because they reflect the frequency data of the original alphabet. A countermeasure is to provide multiple substitutes, known as homophones, for a single letter. For example, the letter e could be assigned a number of different cipher symbols, such as 16, 74, 35, and 21, with each homophone used in rotation, or randomly. If the number of symbols assigned to each letter is proportional to the relative frequency of that letter, then single-letter frequency information is completely obliterated. The great mathematician Carl Friedrich Gauss believed that he had devised an unbreakable cipher using homophones. However, even with homophones, each element of plaintext affects only one element of ciphertext, and multiple-letter patterns (e.g., digram frequencies) still survive in the ciphertext, making cryptanalysis relatively straightforward.

Playfair Cipher

The best-known multiple-letter encryption cipher is the Playfair, which treats digrams in the plaintext as single units and translates these units into ciphertext digrams.

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

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

In this case, the keyword is monarchy. The matrix is constructed by filling in the letters of the keyword (minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetic order. The letters I and J count as one letter. Plaintext is encrypted two letters at a time, according to the following rules:

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

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

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

4. Otherwise, each plaintext letter in a pair is replaced by the letter that lies in its own row and the column occupied by the other plaintext letter. Thus, hs becomes BP and ea becomes IM (or JM, as the encipherer wishes).

Hill Cipher

Another interesting multiletter cipher is the Hill cipher, developed by the mathematician Lester Hill in 1929. The encryption algorithm takes m successive plaintext letters and substitutes for them m ciphertext letters. The substitution is determined by m linear equations in which each character is assigned a numerical value (a = 0, b = 1 ... z = 25). For m = 3, the system can be described as follows:

$$c1 = (k11P1 + k12P2 + k13P3) \bmod 26$$

$$c2 = (k21P1 + k22P2 + k23P3) \bmod 26$$

$$c3 = (k31P1 + k32P2 + k33P3) \bmod 26$$

This can be expressed in term of column vectors and matrices:

$$C = KP \bmod 26$$

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

For example, consider the plaintext "paymoremoney" and use the encryption key

the ciphertext for the entire plaintext is LNSHDLEWMTRW.

Polyalphabetic Ciphers :

Another way to improve on the simple monoalphabetic technique is to use different monoalphabetic substitutions as one proceeds through the plaintext message. The general name for this approach is polyalphabetic substitution cipher. All these techniques have the following features in common:

1. A set of related monoalphabetic substitution rules is used.

2. A key determines which particular rule is chosen for a given transformation. The best known, and one of the simplest, such algorithm is referred to as the Vigenère cipher. In this scheme, the set of related monoalphabetic substitution rules consists of the 26 Caesar ciphers, with shifts of 0 through 25. Each cipher is denoted by a key letter, which is the ciphertext letter that substitutes for the plaintext letter a. Thus, a Caesar cipher with a shift of 3 is denoted by the key value d

To aid in understanding the scheme and to aid in its use, a matrix known as the Vigenère tableau is constructed (Table 2.3). Each of the 26 ciphers is laid out horizontally, with the key letter for each cipher to its left. A normal alphabet for the plaintext runs across the top. The process of encryption is simple: Given a key letter x and a plaintext letter y, the ciphertext letter is at the intersection of the row labeled x and the column labeled y; in this case the ciphertext is V.

Encryption:

key: deceptivedeceptivedeceptive

plaintext: wearediscoveredsaveyourself

ciphertext: ZICVTWQNGRZGVTWAVZHCQYGLMGJ

Decryption;

key: deceptivewearediscoveredsav

plaintext: wearediscoveredsaveyourself

ciphertext: ZICVTWQNGKZEIIGASXSTSLVVWLA

Transposition Techniques

All the techniques examined so far involve the substitution of a ciphertext symbol for a plaintext symbol. A very different kind of mapping is achieved by performing some sort of permutation on the plaintext letters. This technique is referred to as a transposition cipher. The simplest such cipher is the rail fence technique, in which the plaintext is written down as a sequence of diagonals and then read off as a sequence of rows.

For example, to encipher the message "meet me after the toga party" with a rail fence of depth 2, we write the following:

m e m a t r h t g p r y

e t e f e t e o a a t

The encrypted message is MEMATRHTGPRYETEFETEOAAT

Encryption:

Key: 4 3 1 2 5 6 7

Plaintext: a t t a c k p

o s t p o n e

d u n t i l t

w o a m x y z

Ciphertext: TTNAAPTMTSUOAODWCOIXKNLYPETZ

Decryption:

Key: 4 3 1 2 5 6 7

Input: t t n a a p t

m t s u o a o

d w c o i x k

n l y p e t z

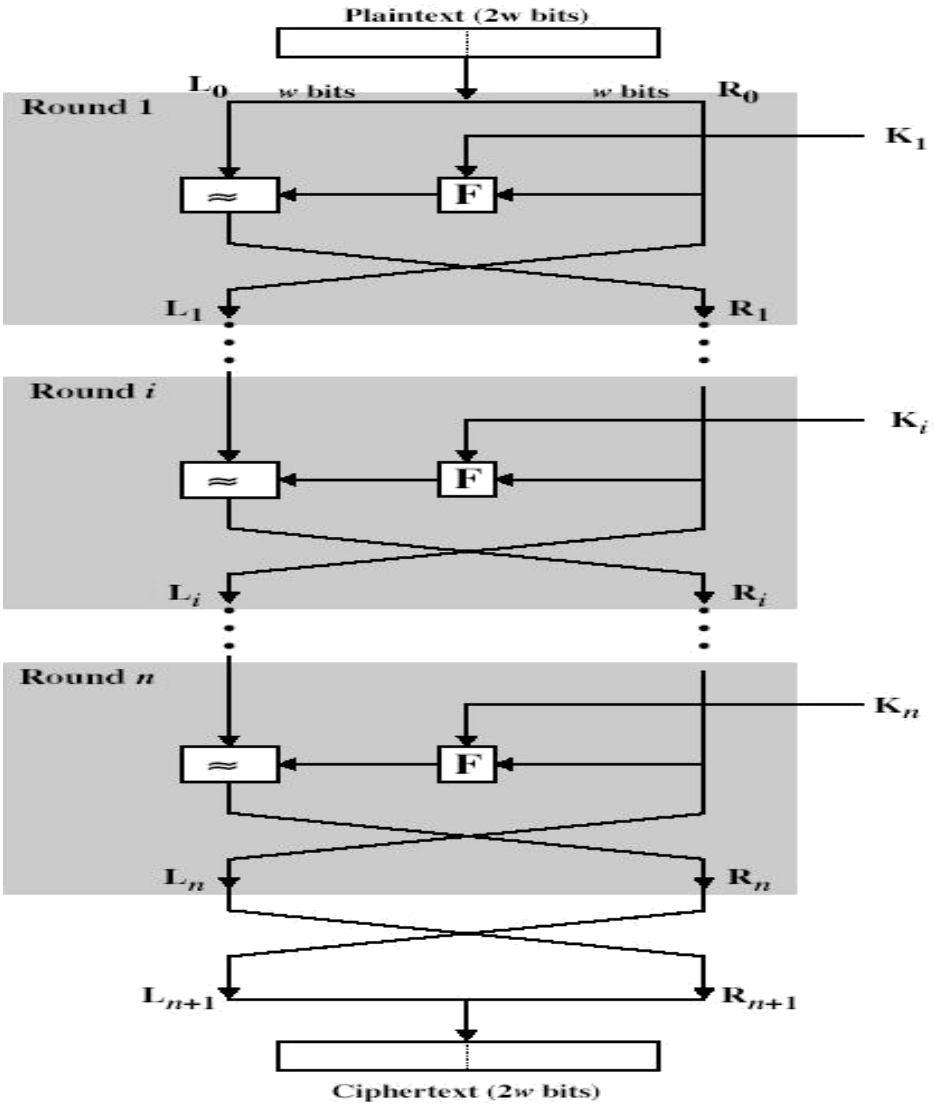
Output: NSCYAUOPTTWLTMDNAOIEPAXTTOKZ

Steganography:

Steganography (pronounced STEHG-uh-NAH-gruhf-ee, from Greek *steganos*, or "covered," and *graphie*, or "writing") is the hiding of a secret message within an ordinary message and the extraction of it at its destination. Steganography takes cryptography a step farther by hiding an encrypted message so that no one suspects it exists. Ideally, anyone scanning your data will fail to know it contains encrypted data.

Feistel Cipher Structure

Most symmetric block ciphers are based on a *Feistel Cipher Structure.* It was first described by Horst Feistel of IBM in 1973 and is still forms the basis for almost all conventional encryption schemes. It makes use of two properties namely *diffusion* and *confusion*; identified by Claude Shannon for frustrating statistical cryptanalysis. Confusion is basically defined as the concealment of the relation between the secret key and the cipher text. On the other hand, diffusion is regarded as the complexity of the relationship between the plain text and the cipher text.



The function of Feistel Cipher is shown in the above figure and can be explained by following steps: The input to the encryption algorithm is a plaintext block of length 2w bits and a key K.

- The plaintext block is divided into two halves: L_i and R_i.
- The two halves pass through n rounds of processing and then combine to produce the cipher text block Each Round i has inputs L_{i-1} and R_{i-1}, derived from the previous round, as well as a unique subkey K_i generated by a sub-key generation algorithm.
- All rounds have the same structure which involves substitution (mapping) on left half of data, which is done by applying a round function F to right half of data and then taking

  XOR of the output of that function and left half of data. The round function F is common to every round but parameterized by round subkey K_i.
- Then a permutation is performed that consists of interchange of the two halves of data.

For each round $i = 0, 1, \ldots, n$, compute

$$L_{i+1} = R_i$$
$$R_{i+1} = L_i \oplus \mathrm{F}(R_i, K_i)$$

. Then the ciphertext is ($Rn + 1$,$Ln + 1$).

Decryption of a ciphertext (R_{n+1},L_{n+1}) is accomplished by computing for

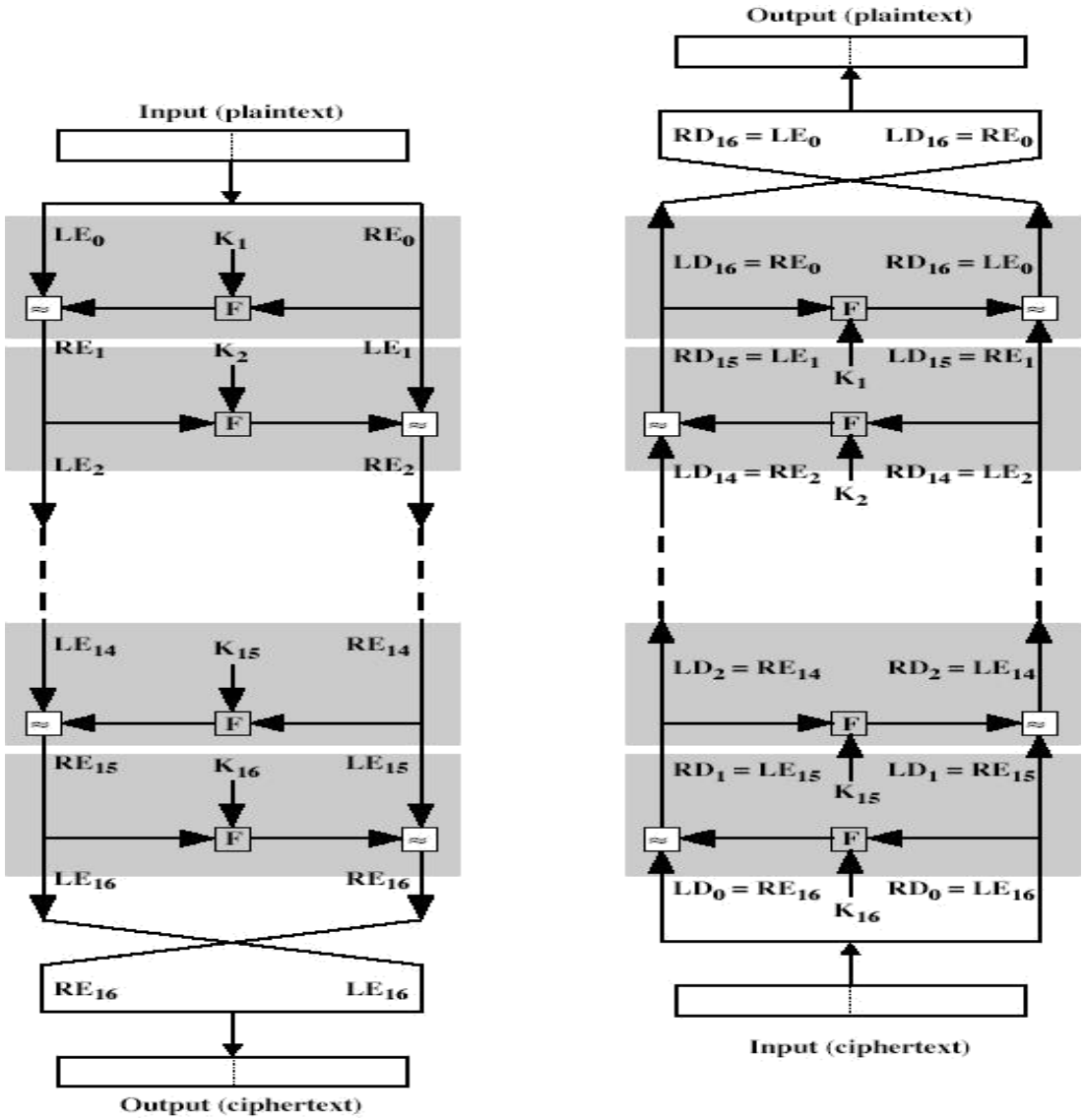
$i = n, n-1, \ldots, 0$

$$R_i = L_{i+1}$$
$$L_i = R_{i+1} \oplus \mathrm{F}(L_{i+1}, K_i)$$

. Then (L_0,R_0) is the plaintext again.

The structure is a particular form of substitution-permutation network (SPN) proposed by Shannon. The realization or development of a Feistel encryption scheme depends on the choice of the following parameters and design features:

- **Block size:** larger block sizes mean greater security but slower processing. Block size of 64 bits has been nearly universal in block cipher design.
- **Key Size:** larger key size means greater security but slower processing. Most common key length in modern algorithms is 128 bits.
- **Number of rounds:** multiple rounds offer increasing security but slows cipher. Typical size is 16 rounds.
- **Subkey generation algorithm:** greater complexity will lead to greater difficulty of cryptanalysis.
- **Round Function:** greater complexity will make cryptanalysis harder.
- **Fast software en/decryption & ease of analysis:** are more recent concerns for practical use and testing.



Feistel Cipher Decryption

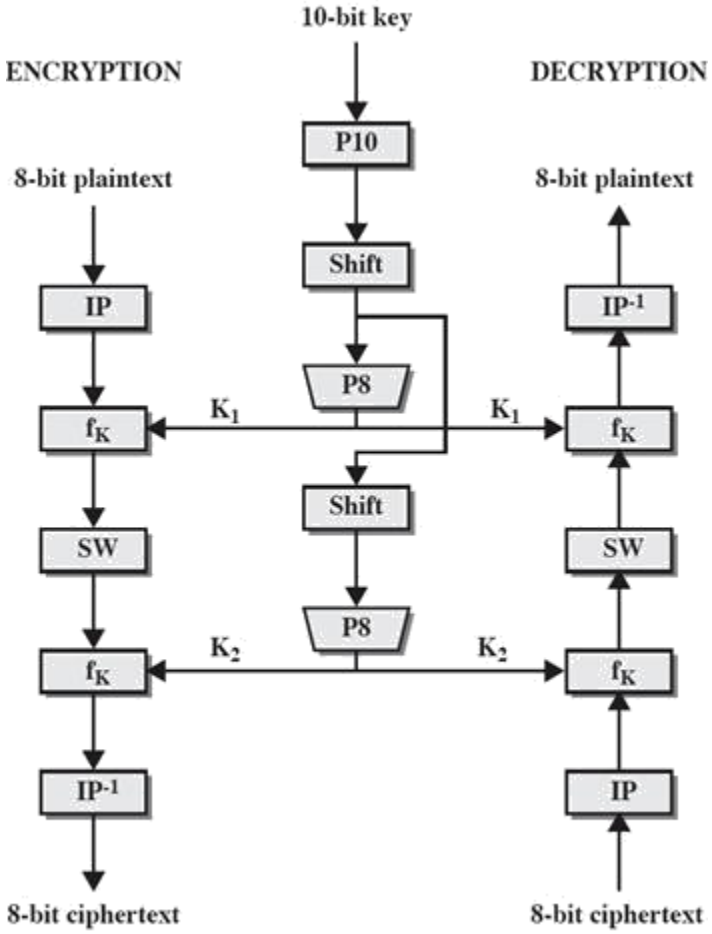
The process of decryption with a Fiestel cipher is same as the encyption process. Use the ciphertext as input to the algorithm, but use the subkeys K_i in the reverse order. Use K_n in the first round and K_{n-1} in the second round and so on until k_1 is used in the last round. Main advantage is we need not implement two different algorithms for encryption and decryption.

The Fiestel cipher has the advantage that encryption and decryption operations are very similar, even identical in some cases requiring only a reversal in the key schedule. Therefore, the size of the code or circuitry required to implement such a cipher is nearly halved.

Conventional Encryption Algorithms

Simplified DES

S-DES is a reduced version of the DES algorithm. It has similar properties to DES but deals with a much smaller block and key size (operates on 8-bit message blocks with a 10-bit key). The S-DES decryption algorithm takes an 8-bit block of ciphertext and the same 10-bit key used to produce that ciphertext as input and produces the original 8-bit block of plaintext. S-DES scheme is shown below:



Simplified DES Scheme

The encryption algorithm involves five functions: and initial permuatation(IP), a complex function labeled f_k, which involves both permutations and substitution operations and depends on a key input, a single permutation function (SW) that switches the two halves of the data, the function f_k again and finally a permutation function that is inverse of the IP i.e. IP^{-1}.

As shown in figure, the function f_k takes the data from encryption function along with 8-bit key. The key is choosen to be 10-bit length from which two 8-bit subkeys are generated. The initial 10-bit key is subjected to a permutation (P10) followed by a shift operation. The output of this shift operation then passes through a permutation function that produces an 8-bit output (P8) for the first key (k_1) and also feeds into another shift and another instance of P8 to produce the second subkey (k_2). The encryption algorithm can be written as:

$$\textbf{Ciphertext} = IP^{-1} (f_{k2}(SW(f_{k1}(IP(\text{plaintext})))))$$

Where K_1 = P8(shift(p10(key)))

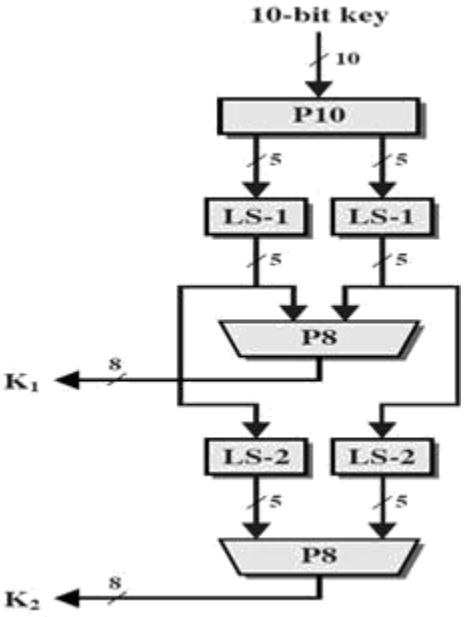
K_2 = P8(shift(shift(p10(key))))

Decryption is also shown in the above figure and can be given as:

$$\textbf{Plaintext} = IP^{-1} (f_{k1}(SW(f_{k2}(IP(\text{ciphertext})))))$$

Key Generation:

The key generation process is shown below:



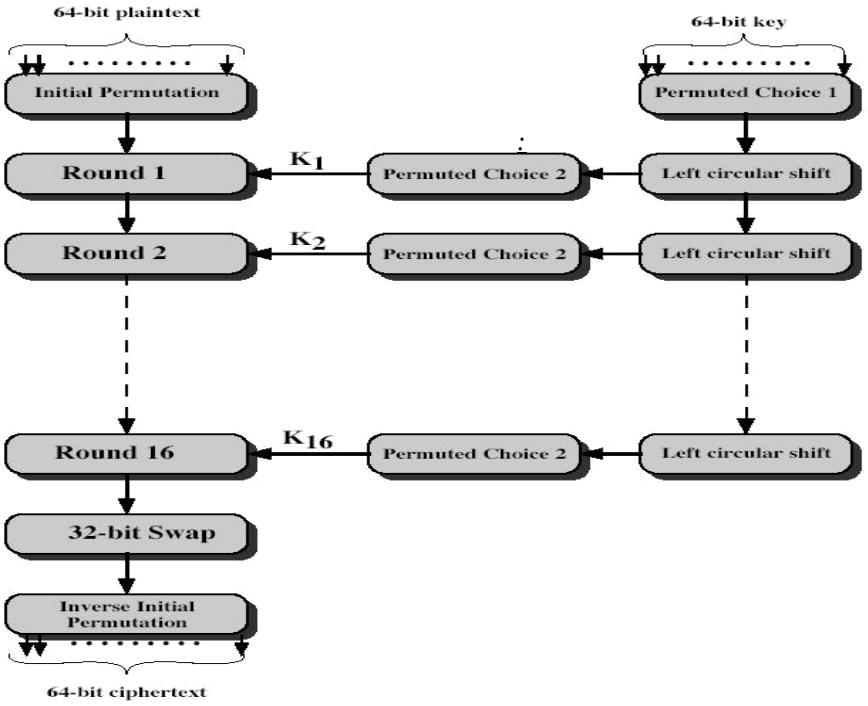
As shown above, a 10-bit key shared between sender and receiver is used and fist passed through a permutation P10.

The Switch Function:

This function interchanges the left and right 4 bits so that the second instance of $\mathbf{f_K}$ operates on a different 4 bits. For second instance all other parameters remain same, but the key is K_2. The S-boxes operates as follows:- The first and fourth input bits are treated as 2-bit numbers that specify a row of the S-box, and the second and third input bits specify a column of S-box. The entry in that row and column in base2 is the 2-bit output.

Data Encryption Standard

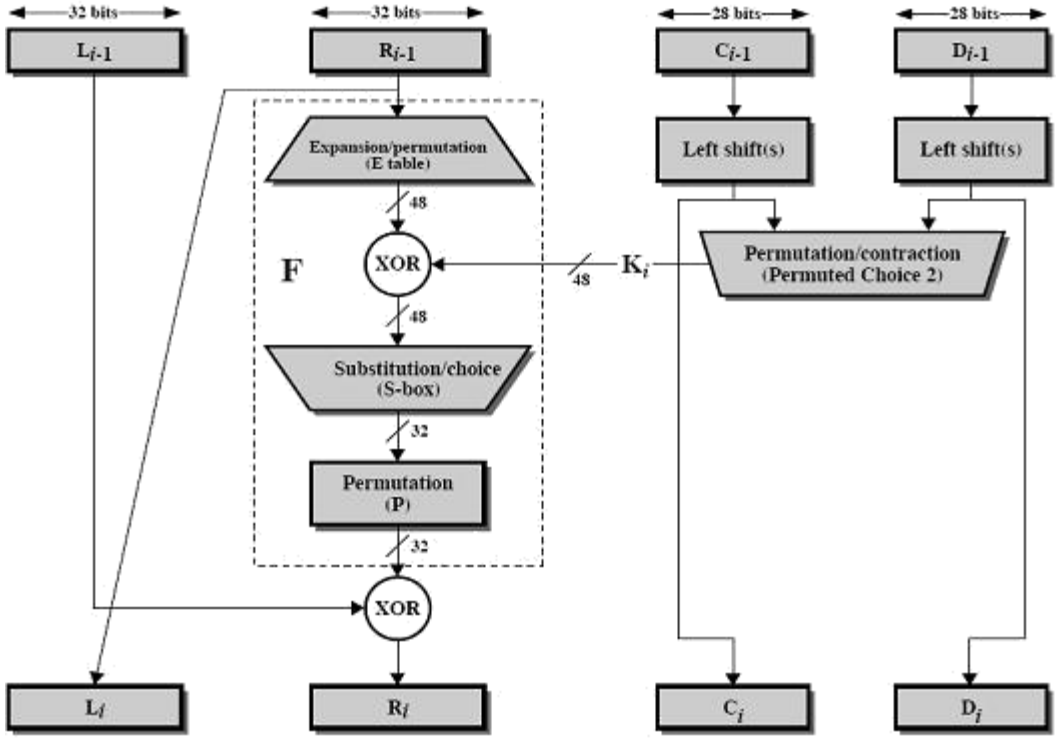
In 1974, IBM proposed "Lucifer", an encryption algorithm using 64-bit keys. Two years later (1977), NBS (now NIST) in consultation with NSA made a modified version of that algorithm into a standard. DES uses the two basic techniques of cryptography - confusion and diffusion. At the simplest level, diffusion is achieved through numerous permutations and confusion is achieved through the XOR operation and the S-Boxes. This is also called an S-P network The DES encryption scheme can be explained by the following figure



The plain text is 64 bits in length and the key in 56 bits in length. Longer plain text amounts are processed in 64-bit blocks. The main phases in the left hand side of the above figure i.e. processing of the plain text are,

- **Initial Permutation (IP):** The plaintext block undergoes an initial permutation. 64 bits of the block are permuted.
- **A Complex Transformation:** 64 bit permuted block undergoes 16 rounds of complex transformation. Subkeys are used in each of the 16 iterations.
- **32-bit swap:** The output of 16th round consists of 64bits that are a function of input plain text and key.32 bit left and right halves of this output is swapped.
- **Inverse Initial Permutation (IP^{-1}):** The 64 bit output undergoes a permutation that is inverse of the initial permutation.
- **A Complex Transformation:** 64 bit permuted block undergoes 16 rounds of complex transformation. Subkeys are used in each of the 16 iterations.
- **32-bit swap:** The output of 16th round consists of 64bits that are a function of input plain text and key.32 bit left and right halves of this output is swapped.
- **Inverse Initial Permutation (IP^{-1}):** The 64 bit output undergoes a permutation that is inverse of the initial permutation.

The following figure shows a closer view of algorithms for a single iteration. The 64bit permuted input passes through 16 iterations, producing an intermediate 64-bit value at the conclusion of each iteration.



Single Round of DES Algorithm

The left and right halves of each 64 bit intermediate value are treated as separated 32-bit quantities labeled L (left) and R (Right). The overall processing at each iteration is given by following steps, which form one round in an S-P network.

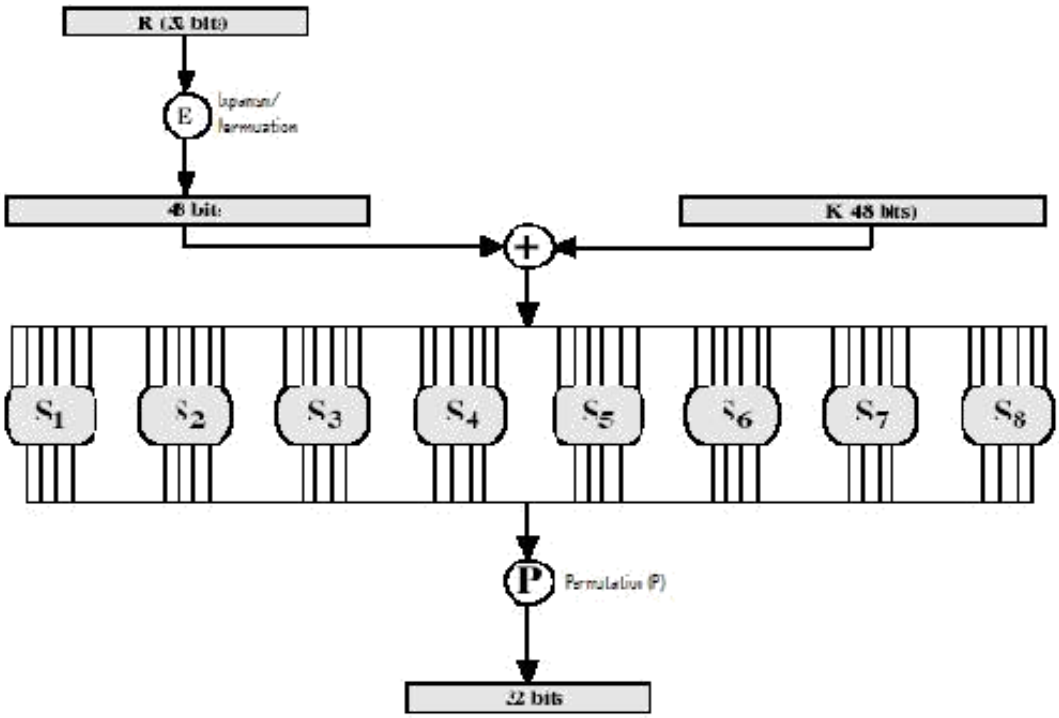
Li = Ri-1.

Ri = L i-1 ⊕ F(R i-1, Ki)

Where Function F can be described as **P(S(E(R(i-1)) □ K(i)))**

The left hand output of an iteration (Li) is equal to the right hand input to that iteration Ri-1. The right hand output Ri is exclusive OR of Li-1 and a complex function F of Ri-1 and Ki. The fucntion F can be depicted by the following figure. S1, S2 S8 represent the "S-boxes" , which maps each combination of 48 input bits into a particular 32 bit pattern. For the generation of subkey of length 48 bits, a 56bit key is used which is first passed through a permutation funciton and then halved to get two 28 bit quantities labeled C0 and D0. At each iteration, these two C and D are subjected to a circular left shift or rotation of 1 or 2 bits. These shifted values serve as input to the next iteration and also to another permutation function which produces a 48-bit output. This output is fed as input to function

F(R $_{i-1}$, Ki).



The first and last bits of the input to the box Si form a 2-bit binary number to select one of four substitutions defined by the four rows in the table for Si. The middle 4-bits select a particular column. The decimal value in the cell selected by the row and column is converted to its 4-bit representation to produce the output.

The substitution consists of a set of eight S-boxes, each of which accepts 6 bits as input and produces 4 bits as output. The process of decryption with DES is essentially the same as the encryption process: no different algorithm is used. The ciphertext is used as input to the DES algorithm and the keys are used in the reverse order i.e. K16 in the first iteration, K15 on the second iteration and so on until k1 is used on the sixteenth and last iteration.

Strength of DES:

Avalanche Effect: An effect in DES and other secret key ciphers where each small change in plaintext implies that somewhere around half the ciphertext changes. The avalanche effect makes it harder to successfully cryptanalyze the ciphertext. DES exhibits a strong Avalanche effect.

Concern about the strength of DES falls into two categories i.e. strength of algorithm itself and use of 56-bit key. Though many attempts were made over the years to find and exploit weaknesses in the algorithm, none of them were successful in discovering any fatal weakness in DES. A serious concern is with the key size as the time passed the security in DES became getting compromised by the advent of supercomputers which succeeded in breaking the DES quickly using a brute-force attack. If the only form of attack that could be made on an encryption algorithm is brute force, the way of countering it is obviously using long keys. If a key of size 128 bits is used, it takes approximately 10^{18} years to break the code making the algorithm unbreakable by brute-force approach.

The two analytical attacks on DES are Differential cryptanalysis and Linear cryptanalysis. Both make use of Known plaintext-ciphertext pairs and try to attack the round structure and the S-Boxes. Recent advancements showed that using Differential cryptanalysis, DES can be broken using 2^{47} plaintext-ciphertext pairs and for linear cryptanalysis, the number is even reduced to 2^{41}.

Triple DES

The first answer to problems of DES is an algorithm called Double DES which includes double encryption with two keys. It increases the key size to 112 bits, which seems to be secure. But, there are some problems associated with this approach.

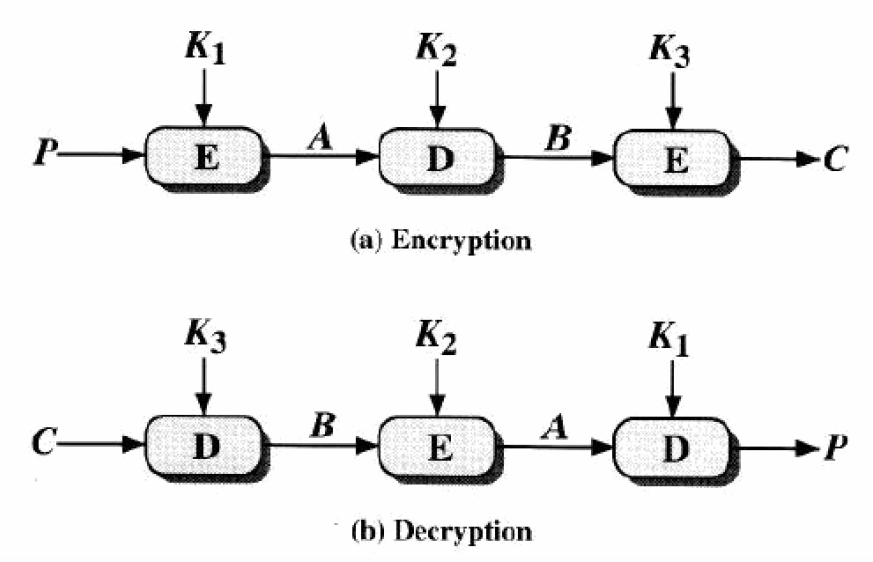
issue of reduction to single stage:

In other words, could there be a key K3 such that $E_{K2}(E_{K21}(P)) = E_{K3}(P)$?

"meet-in-the-middle" attack:

- Works when given a known (P,C) pair
- since $\mathbf{X = E_{K1}(P) = D_{K2}(C)}$
- attack by encrypting P with all 2^{56} keys $K_1$and store
- then decrypt C with all possible 2^{56} keys K_2 and match X value

Triple DES was the answer to many of the shortcomings of DES. Since it is based on the DES algorithm, it is very easy to modify existing software to use Triple DES. 3DES was developed in 1999 by IBM – by a team led by Walter Tuchman. 3DES prevents a meet-in-the-middle attack. 3DES has a 168-bit key and enciphers blocks of 64 bits. It also has the advantage of proven reliability and a longer key length that eliminates many of the shortcut attacks that can be used to reduce the amount of time it takes to break DES. 3DES uses three keys and three executions of the DES algorithm. The function follows an encrypt-decrypt-encrypt (EDE) sequence.



$$C = E_{K3}[D_{K2}[E_{K1}[P]]]$$

Where C= ciphertext, P= plaintext and
$E_K[X]$ = encryption of X using key K
$D_K[Y]$ = decryption of Y using key K

Decryption is simply the same operation with the keys reversed

$$P = D_{K1}[E_{K2}[D_{K3}[C]]]$$

Triple DES runs three times slower than standard DES, but is much more secure if used properly. With three distinct keys, TDEA has an effective key length of 168 bits making it a formidable algorithm. As the underlying algorithm is DEA, it offers the same resistance to cryptanalysis as is DEA.

Triple DES can be done using 2 keys or 3 keys.

Advanced Encryption Standard

AES is a symmetric block cipher that is intended to replace DES as the approved standard for a wide range of applications. The drawbacks of 3DES being it is very slow and also it uses 64-bit block size same as DES. For reasons of both efficiency and security, a larger key size is desirable. So, NIST (National Institute of Standards and Technology) has called for proposals for a new AES, which should have security strength equal to or better than 3DES and significantly, improved efficiency. NIST specified that AES must be a symmetric block cipher with a block length of 128 bits and support for key lengths of 128, 192, and 256 bits.

Out of all the algorithms that were submitted, five were shortlisted and upon final evaluation, NIST selected **Rijndael** as the proposed AES algorithm. The two researchers who developed and submitted Rijndael for the AES are both cryptographers from Belgium: **Dr. Joan Daemen and Dr. Vincent Rijmen.**

AES Evaluation:

- There are three main categories of criteria used by NIST to evaluate potential candidates.
- *Security:* Resistance to cryptanalysis, soundness of math, randomness of output, etc
- *Cost:* Computational efficiency (speed), Memory requirements
- *Algorithm/Implementation Characteristics:* Flexibility, hardware and software suitability, algorithm simplicity

Simplified AES

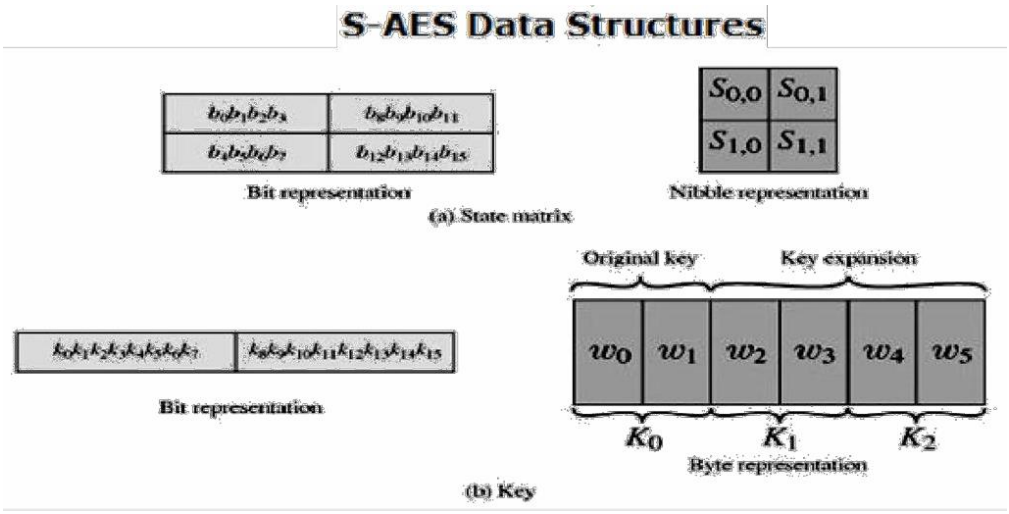
The encryption algorithm takes a 16-bit block of plaintext as input and a 16-bit key and produces a 16-bit block of ciphertext as output. The S-AES decryption algorithm takes a 16-bit block of ciphertext and the same 16-bit key used to produce that ciphertext as input and produces the original 16-bit block of plaintext as output. The encryption algorithm involves the use of four different functions, or transformations: add key (A_K) nibble substitution (NS), shift row (SR), and mix column (MC).

The encryption algorithm can be expressed as:

$$A_{K_2} \circ SR \circ NS \circ A_{K_1} \circ MC \circ SR \circ NS \circ A_{K_0}$$, so that A_{K0} is applied first.

The encryption algorithm is organized into three rounds. Round 0 is simply an add key round; round 1 is a full round of four functions; and round 2 contains only 3 functions. Each round includes the add key function, which makes use of 16 bits of key. The initial 16-bit key is expanded to 48 bits, so that each round uses a distinct 16-bit round key. S- AES encryption and decryption scheme is shown below.

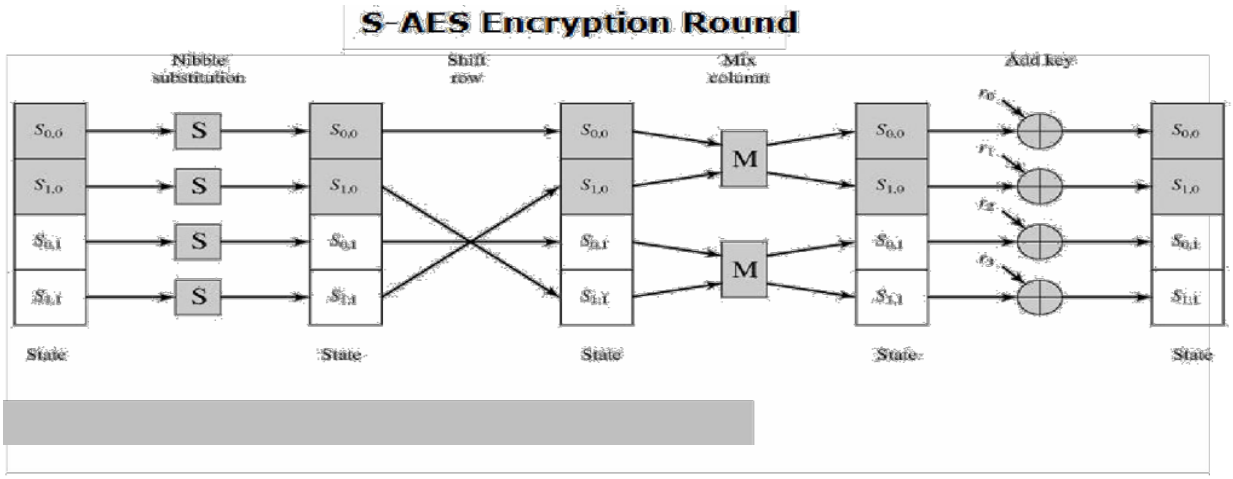
Each function operates on a 16-bit state, treated as a 2 x 2 matrix of nibbles, where one nibble equals 4 bits. The initial value of the state matrix is the 16-bit plaintext; the state matrix is modified by each subsequent function in the encryption process, producing after the last function the 16-bit ciphertext. The following figure shows the ordering of nibbles within the matrix is by column. So, for example, the first eight bits of a 16-bit plaintext input to the encryption cipher occupy the first column of the matrix, and the second eight bits occupy the second column. The 16-bit key is similarly organized, but it is somewhat more convenient to view the key as two bytes rather than four nibbles The expanded key of 48 bits is treated as three round keys, whose bits are labelled as follows: $K_0 = k_0...k_{15}$; $K_1 = k_{16}...k_{31}$; $K_2 = k_{32}...k_{47}$.



The following figure shows the essential elements of a full round of S-AES. The decryption as shown above can be given as:

$$A_{K_0} \circ INS \circ ISR \circ IMC \circ A_{K_1} \circ INS \circ ISR \circ A_{K_2}$$

in which three of the functions have a corresponding inverse function: inverse nibble substitution (INS), inverse shift row (ISR), and inverse mix column (IMC).



S-AES Encryption and Decryption

The individual functions that are part of the encryption algorithm are given below.

Add Key

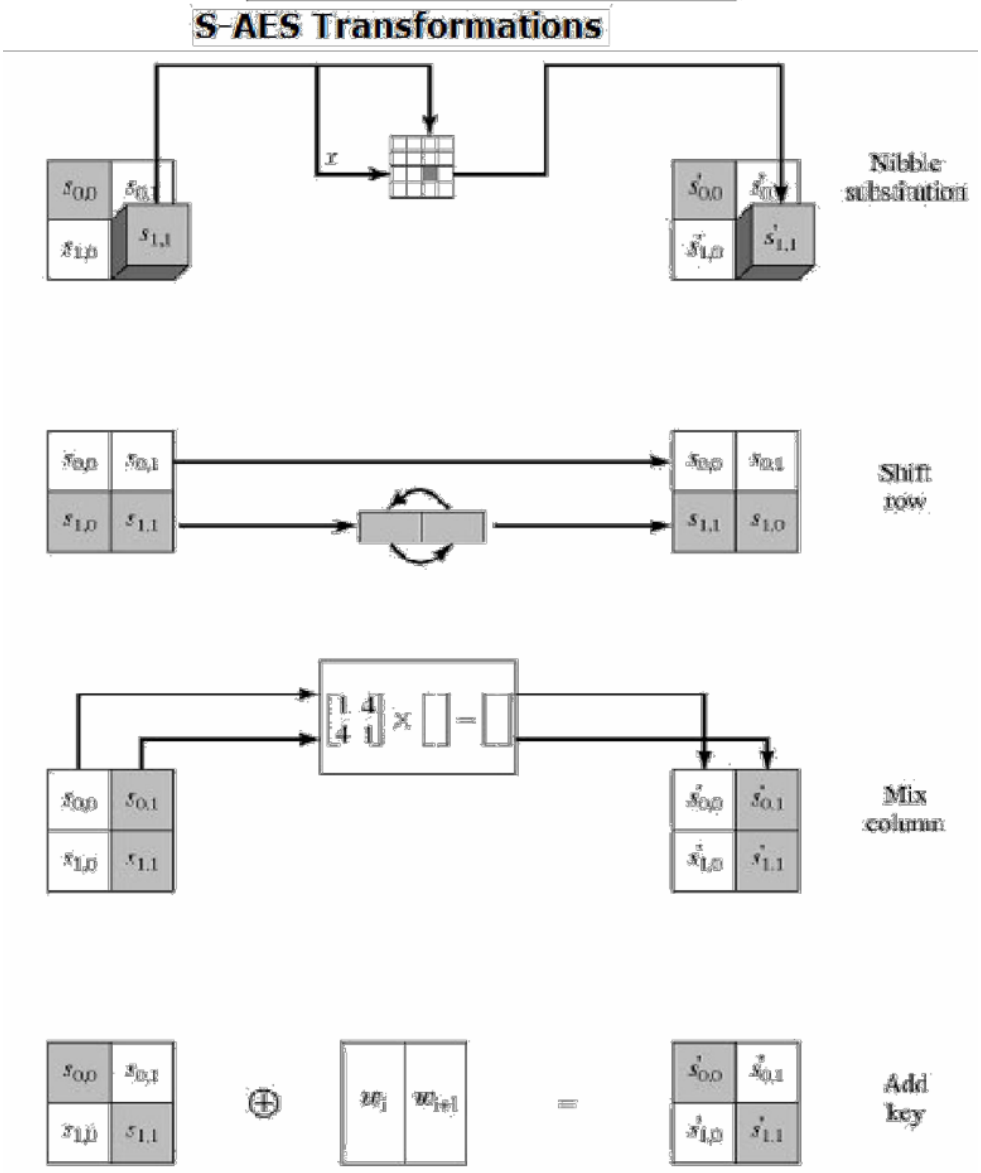
A	4	⊕	2	5	=	8	1
7	9		D	5		A	C
state matrix			key				

The add key function consists of the bitwise XOR of the 16-bit state matrix and the 16-bit round key. As shown in the above example, it can also be viewed as a nibble-wise or bitwise operation. The inverse of the add key function is identical to the add key function, because the XOR operation is its own inverse.

Nibble Substitution

The nibble substitution function is a simple table lookup. AES defines a 4 x 4 matrix of nibble values, called an S-box that contains a permutation of all possible 4-bit values. Each individual nibble of the state matrix is mapped into a new nibble in the following way: The leftmost 2 bits of the nibble are used as a row value and the rightmost 2 bits are used as a column value. These row and column values serve as indexes into the S-box to select a unique 4-bit output value. For example, the hexadecimal value A references row 2, column 2 of the S-box, which contains the value 0. Accordingly, the value A is mapped into the value 0.

For the example, after nibble substitution, the output is



Shift Row

The shift row function performs a one-nibble circular shift of the second row of the state matrix; the first row is not altered. Our example is shown below:

6	4
0	C

→

6	4
C	0

The inverse shift row function is identical to the shift row function, because it shifts the second row back to its original position.

Mix Column

The mix column function operates on each column individually. Each nibble of a column is mapped into a new value that is a function of both nibbles in that column. The transformation can be defined by the following matrix multiplication on the state matrix.

$$\begin{bmatrix} 1 & 4 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} S_{0,0} & S_{0,1} \\ S_{1,0} & S_{1,1} \end{bmatrix} = \begin{bmatrix} S'_{0,0} & S'_{0,1} \\ S'_{1,0} & S'_{1,1} \end{bmatrix}$$

Where arithmetic is performed in GF(2^4), and the symbol · refers to multiplication in GF(2^4). The example is shown below:

$$\begin{bmatrix} 1 & 4 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} 6 & 4 \\ C & 0 \end{bmatrix} = \begin{bmatrix} 3 & 4 \\ 7 & 3 \end{bmatrix}$$

The inverse mix column function is defined as follows:

$$\begin{bmatrix} 9 & 2 \\ 2 & 9 \end{bmatrix} \begin{bmatrix} S_{0,0} & S_{0,1} \\ S_{1,0} & S_{1,1} \end{bmatrix} = \begin{bmatrix} S'_{0,0} & S'_{0,1} \\ S'_{1,0} & S'_{1,1} \end{bmatrix}$$

Key Expansion

For key expansion, the 16 bits of the initial key are grouped into a row of two 8-bit words. The following figure shows the expansion into 6 words, by the calculation of 4 new words from the initial 2 words. The algorithm is as follows:

$$w_2 = w_0 \oplus g(w_1) = w_0 \oplus \text{RCON}(1) \oplus \text{SubNib}(\text{RotNib}(w_1))$$

$$w_3 = w_2 \oplus w_1$$

$$w_4 = w_2 \oplus g(w_3) = w_2 \oplus \text{RCON}(2) \oplus \text{SubNib}(\text{RotNib}(w_3))$$

$$w_5 = w_4 \oplus w_3$$

RCON is a round constant, defined as follows: **RC[i] = x^{i+2}, so that RC[1]=x^3=1000** and **RC[2]=x^4 mod (x^4 + x + 1) = x + 1 = 0011**. RC[i] forms the leftmost nibble of a byte, with the rightmost nibble being all zeros. Thus, RCON(1) = 10000000 and RCON(2) = 00110000.

For example, suppose the key is 2D55 = 0010 1101 0101 0101 = w_0w_1. Then,

$$w_2 = 00101101 \oplus 10000000 \oplus \text{SubNib}(01010101)$$

$$= 00101101 \oplus 10000000 \oplus 00010001 = 10111100$$

$$w_3 = 10111100 \oplus 01010101 = 11101001$$

$$w_4 = 10111110 \oplus 00110000 \oplus \text{SubNib}(10011110)$$

$$= 10111100 \oplus 00110000 \oplus 00101111 = 10100011$$

$$w_5 = 10100011 \oplus 11101001 = 01001010$$

The S-Box

The S-box is constructed as follows:

- Initialize the S-box with the nibble values in ascending sequence row by row. The first row contains the hexadecimal values 0, 1, 2, 3; the second row contains 4, 5, 6, 7; and so on. Thus, the value of the nibble at row i, column j is 4i + j.
- Treat each nibble as an element of the finite field GF(2^4) modulo x^4 +x + 1. Each nibble $a_0a_1a_2a_3$ represents a polynomial of degree 3.
- Map each byte in the S-box to its multiplicative inverse in the finite field GF(2^4) modulo x^4 + x + 1; the value 0 is mapped to itself.
- Consider that each byte in the S-box consists of 4 bits labeled (b_0, b_1, b_2, b_3). Apply the following transformation to each bit of each byte in the S-box: The AES standard depicts this transformation in matrix form as follows:

$$\begin{bmatrix} b_0' \\ b_1' \\ b_2' \\ b_3' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \oplus \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

The prime (') indicates that the variable is to be updated by the value on the right. Remember that addition and multiplication are being calculated modulo 2.

RC4

The RC4 Algorithm:

The algorithm is based on the use of a random permutation. Analysis shows that the period of the cipher is overwhelmingly likely to be greater than 10100 [ROBS95a]. Eight to sixteen machine operations are required per output byte, and the cipher can be expected to run very quickly in software.
RC4 is used in the SSL/TLS (Secure Sockets Layer/Transport Layer Security) standards that have been defined for communication between Web browsers and servers. It is also used in the WEP (Wired Equivalent Privacy) protocol and the newer WiFi Protected Access (WPA) protocol that are part of the IEEE 802.11 wireless LAN standard. RC4 was kept as a trade secret by RSA Security.

The RC4 algorithm is remarkably simply 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-byte state vector S, with elements S[0], S[1],..., S [255]. At all times, S contains a permutation of all 8-bit numbers from 0 through 255. For encryption and decryption, a byte k is generated from S by selecting one of the 255 entries in a systematic fashion. As each value of k is generated, the entries in S are once again permuted.

Initialization of S

To begin, the entries of S are set equal to the values from 0 through 255 in ascending order; that is; S [0] = 0, S[1] = 1,..., S[255] = 255. A temporary vector, T, is also created. If the length of the key K is 256 bytes, then K 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 preliminary operations can be summarized as follows:

```
/* 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]);
```

Because the only operation on S is a swap, the only effect is a permutation. S still contains all the numbers from 0 through 255.

Stream Generation

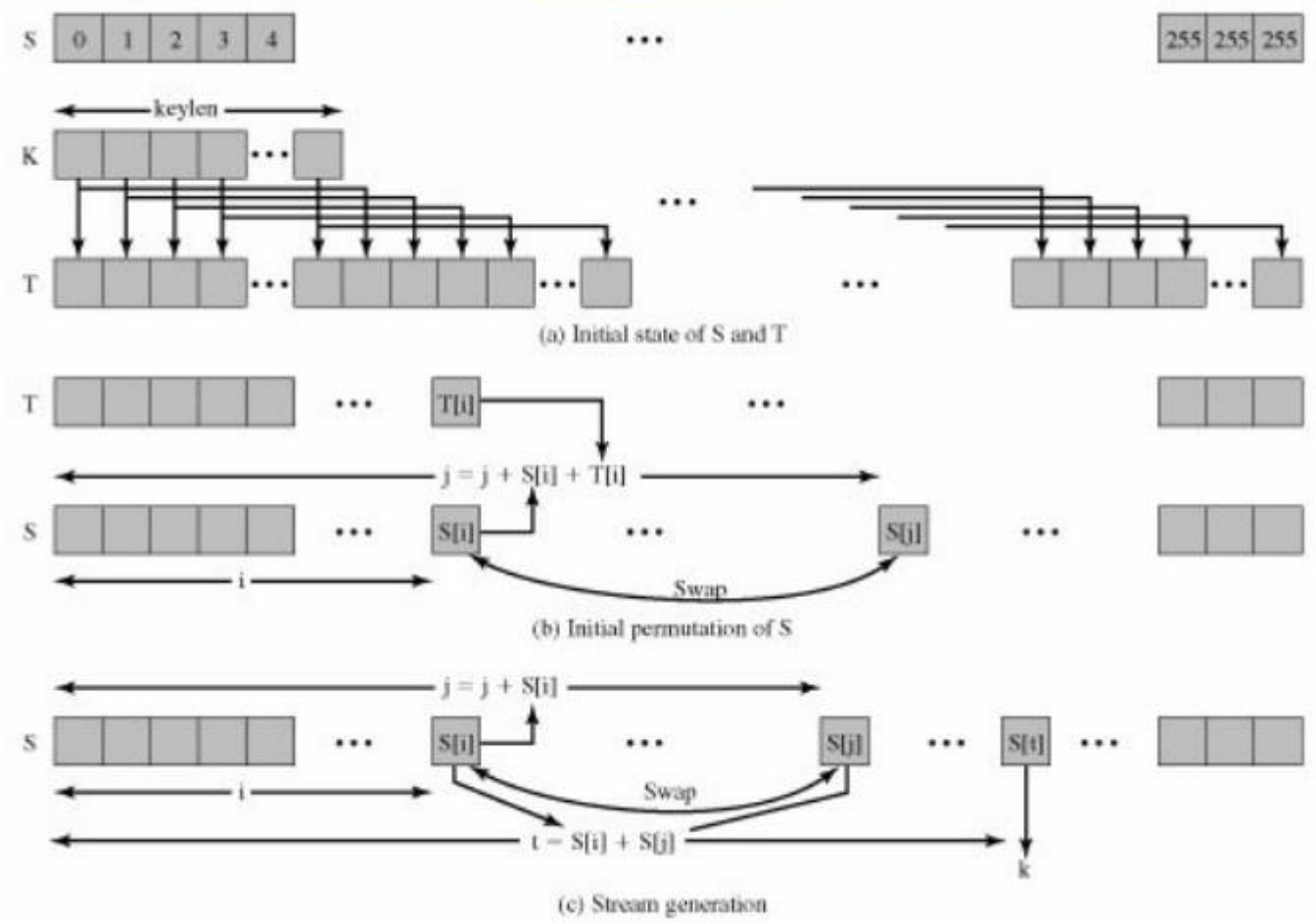
Once the S vector is initialized, the input key is no longer used. Stream generation involves cycling through all the elements of S[i], and, for each S[i], swapping S[i] with another byte in S according to a scheme dictated by the current configuration of S. After S[255] is reached, the process continues, 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];
```

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

Figure RC4

[View full size image]



Strength of RC4

A number of papers have been published analyzing methods of attacking RC4 [e.g., [KNUD98], [MIST98], [FLUH00], [MANT01]). None of these approaches is practical against RC4 with a reasonable key length, such as 128 bits. A more serious problem is reported in [FLUH01]. The authors demonstrate that the WEP protocol, intended to provide confidentiality on 802.11 wireless LAN networks, is vulnerable to a particular attack approach. In essence, the problem is not with RC4 itself but the way in which keys are generated for use as input to RC4. This particular problem does not appear to be relevant to other applications using RC4 and can be remedied in WEP by changing the way in which keys are generated. This problem points out the difficulty in designing a secure system that involves both cryptographic functions and protocols that make use of them.