

# Cryptography and System Security (CSS)

## Course Code: CSC 604



### Subject Incharge

Ankita Karia

Assistant Professor

Room No. 421

email: [ankitakaria@sfit.ac.in](mailto:ankitakaria@sfit.ac.in)



# COURSE OBJECTIVES

---

1. To introduce classical encryption techniques and concepts of modular arithmetic and number theory.
2. To explore the working principles and utilities of various cryptographic algorithms including secret key cryptography, hashes and message digests, and public key algorithms
3. To explore the design issues and working principles of various authentication protocols, PKI standards and various secure communication standards including Kerberos, IPsec, and SSL/TLS and email.
4. To develop the ability to use existing cryptographic utilities to build programs for secure communication.



# Syllabus Module-wise

<b>Module No.</b>	<b>Unit No.</b>	<b>Topics</b>	<b>Course Outcome</b>
1.0		<b>Introduction &amp; Number Theory</b>	<b>CO 1</b>
	1.1	Security Goals, Services, Mechanisms and attacks, The OSI security architecture, Network security model, Classical Encryption techniques, Symmetric cipher model, mono-alphabetic and poly- alphabetic substitution techniques: Vigenere cipher, playfair cipher, Hill cipher, transposition techniques: keyed and keyless transposition ciphers, steganography.	To explain system security goals and concepts, classical encryption techniques and acquire fundamental knowledge on the concepts of modular arithmetic and number theory.
	1.2	Modular Arithmetic and Number Theory:- Euclid's algorithm—Prime numbers-Fermat's and Euler's theorem- Testing for primality –The Chinese remainder theorem, Discrete logarithms.	



# Syllabus Module-wise

<b>Module No.</b>	<b>Unit No.</b>	<b>Topics</b>	<b>Course Outcome</b>
2.0		<b>Symmetric and Asymmetric key Cryptography and key Management</b>	<b>CO 2</b>
	2.1	Block cipher principles, block cipher modes of operation, DES, Double DES, Triple DES, Advanced Encryption Standard (AES), Stream Ciphers: RC5 algorithm.	To discuss, compare and apply different encryption and decryption techniques to solve problems related to confidentiality and authentication
	2.2	Public key cryptography: Principles of public key cryptosystems-The RSA algorithm, The knapsack algorithm, ElGamal Algorithm.	
	2.3	Key management techniques: using symmetric and asymmetric algorithms and trusted third party. Diffie Hellman Key exchange algorithm.	



# Syllabus Module-wise

Module No.	Unit No.	Topics	Course Outcome
3.0		<b>Hashes, Message Digests and Digital Certificates</b>	<b>CO 3</b>
	3.1	Cryptographic hash functions, Properties of secure hash function, MD5, SHA-1, MAC, HMAC, CMAC.	To apply the knowledge of cryptographic checksums and evaluate the performance of different message digest algorithms for verifying the integrity of varying message sizes.
	3.2	<b>Digital Certificate: X.509, PKI</b>	



# Syllabus Module-wise

<b>Module No.</b>	<b>Unit No.</b>	<b>Topics</b>	<b>Course Outcome</b>
4.0		<b>Authentication Protocols &amp; Digital signature schemes</b>	<b>CO 4</b>
	4.1	User Authentication and Entity Authentication, One-way and mutual authentication schemes, Needham Schroeder Authentication protocol, Kerberos Authentication protocol.	To apply different digital signature algorithms to achieve authentication and design secure applications
	4.2	Digital Signature Schemes – RSA, ElGamal and Schnorr signature schemes.	

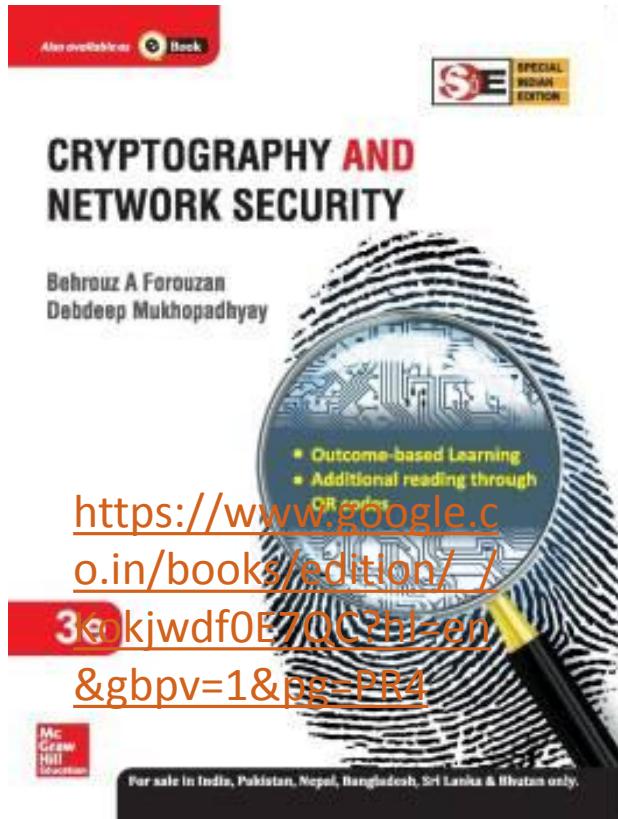


# Syllabus Module-wise

<b>Module No.</b>	<b>Unit No.</b>	<b>Topics</b>	<b>Course Outcome</b>
5.0		<b>Network Security and Applications</b>	<b>CO 5</b>
	5.1	Network security basics: TCP/IP vulnerabilities (Layer wise), Packet Sniffing, ARP spoofing, port scanning, IP spoofing, TCP syn flood, DNS Spoofing.	To discuss network security basics, analyze different attacks on networks and evaluate the performance of firewalls and security protocols like SSL, IPsec, and PGP.
	5.2	Denial of Service: Classic DOS attacks, Source Address spoofing, ICMP flood, SYN flood, UDP flood, Distributed Denial of Service, Defenses against Denial of Service Attacks.	
	5.3	Internet Security Protocols: SSL, IPSEC, Secure Email: PGP, Firewalls, IDS and types, Honey pots	
6.0	6.1	<b>System Security</b>	<b>CO 6</b>
		Software Vulnerabilities: Buffer Overflow, Format string, cross-site scripting, SQL injection, Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, Rootkits.	To analyze and apply system security concept to recognize malicious code



# Text Books to Refer



# TOPICS TO COVER

---

## 1. Background

## 2. Basic Definitions

## 3. Security Goals

- ✓ Confidentiality
- ✓ Integrity
- ✓ Availability



*The art of war teaches us to rely not on the likelihood of the enemy's not coming, but on our own readiness to receive him; not on the chance of his not attacking, but rather on the fact that we have made our position unassailable.*

**—The Art of War, Sun Tzu**

# Background

1. Human being from ages had two inherent needs –
  - (a) to communicate and share information and
  - (b) to communicate selectively.
2. These two needs gave rise to the art of coding the messages in such a way that only the intended people could have access to the information.
3. Unauthorized people could not extract any information, even if the scrambled messages fell in their hand.

1. Information is an asset that has a value associated to it.
2. As an asset, information needs to be secured from unauthorized access (attacks).
3. With the advent of computers and distributed systems, there should be some mechanism to protect the data which is stored on computers or transmitted through network.



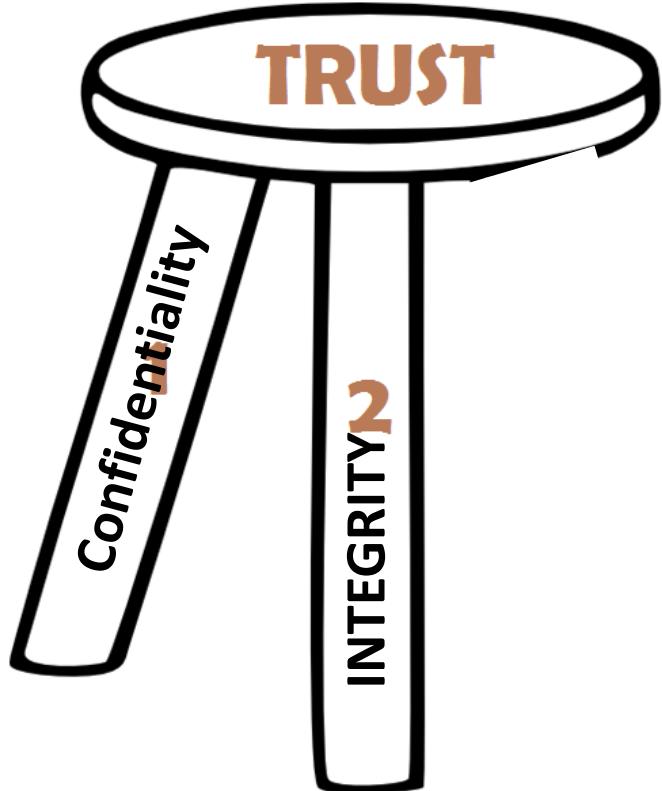
# Basic Definitions

THREAT	VULNERABILITY	ATTACK
A possible security violation that might exploit the vulnerability of a system or asset	Refers to a known weakness of an asset that can be exploited by one or more attackers	An action that exploits a vulnerability or enacts a threat
Its origin may be accidental, environmental, human negligence or human failure	It is a weakness that makes threat possible	It is an deliberate unauthorized action on a system or asset.
Types of threat: 1. Interruption 2. Interception 3. Fabrication 4. Modification	This may be because of:- 1. Poor design 2. Configuration mistakes 3. Inappropriate or insecure coding techniques	It will have a motive and will follow a method when opportunity arises.

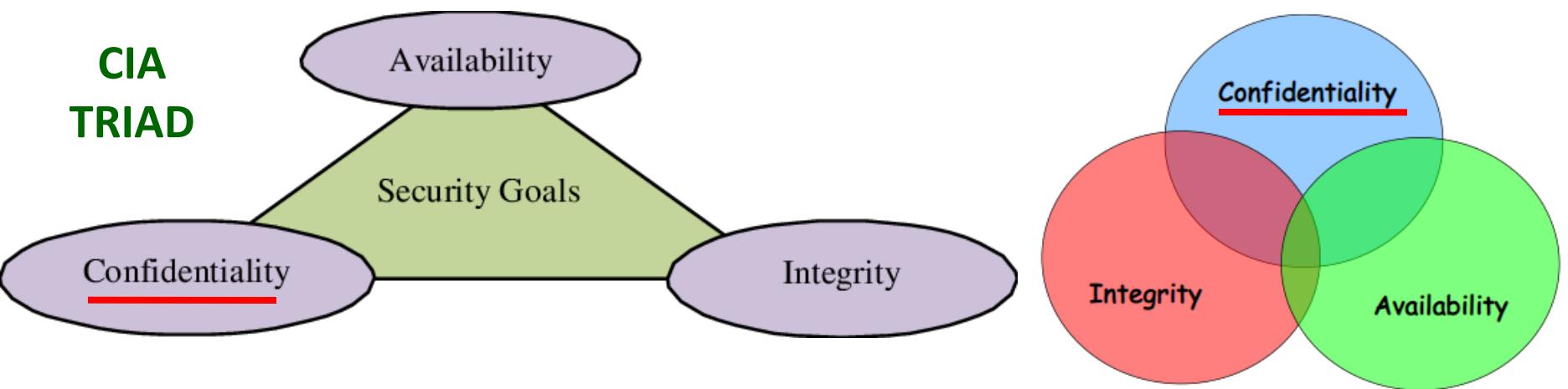
An attack can be classified as : **ACTIVE or PASSIVE** Attack



# Security Goals

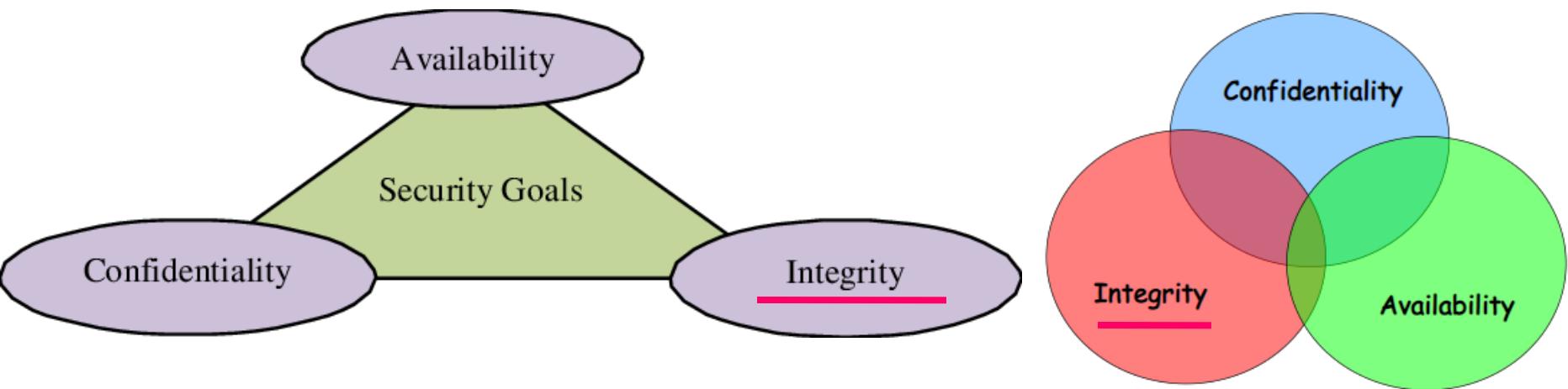


# Security Goals



1. Its function is to protect precious/sensitive information from unauthorized persons
2. It ensures that the data is available only to intended and authorized persons
3. Encryption techniques can protect data at rest or in transit and prevents unauthorized access to protected data.

# Security Goals

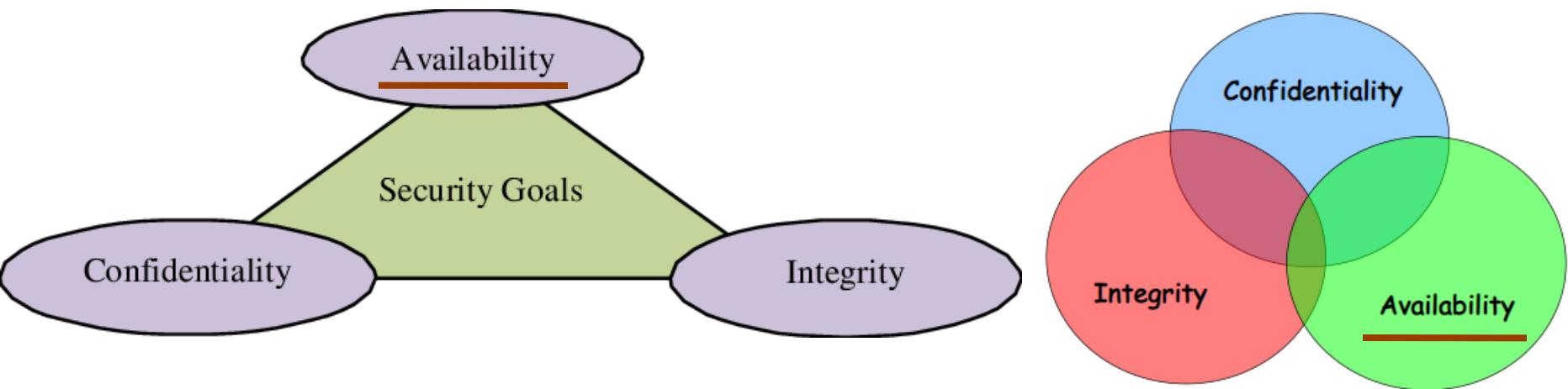


1. Aims at maintaining and assuring the accuracy and consistency of data
2. Its function is to make sure that the data is accurate and reliable and is not allowed by unauthorized persons or hackers
3. Integrity violation is not necessarily the result of malicious act; an interruption in the system may also create unwanted changes in some information.

## HASHING AND CHECKSUMS



# Security Goals

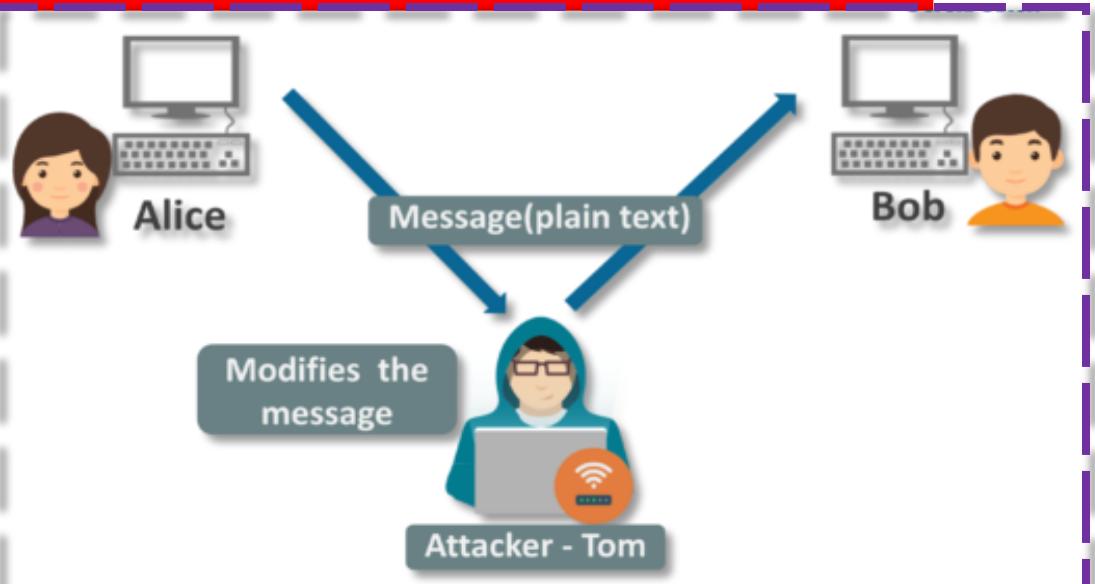


1. Information is useless if it is not available.
2. Function:- To make sure that the data , network resources are continuously available to the legitimate users whenever required.
3. Resolving hardware and software conflicts, along with regular maintenance is crucial to keep the system up and available.

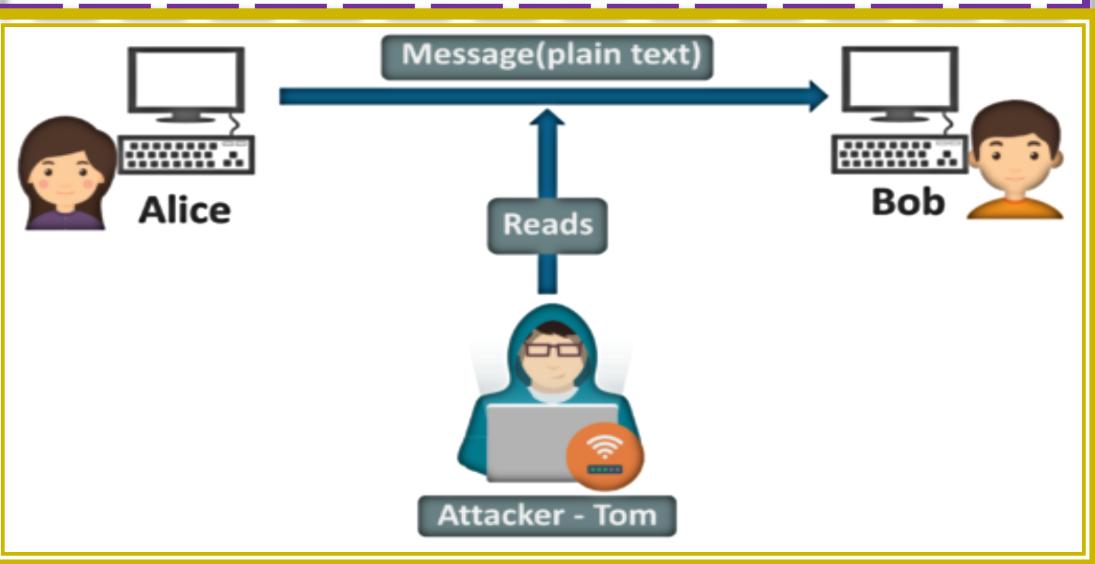
**LOAD BALANCER, RAID or SERVER CLUSTERING**



# Types of Attacks



**ACTIVE ATTACK**



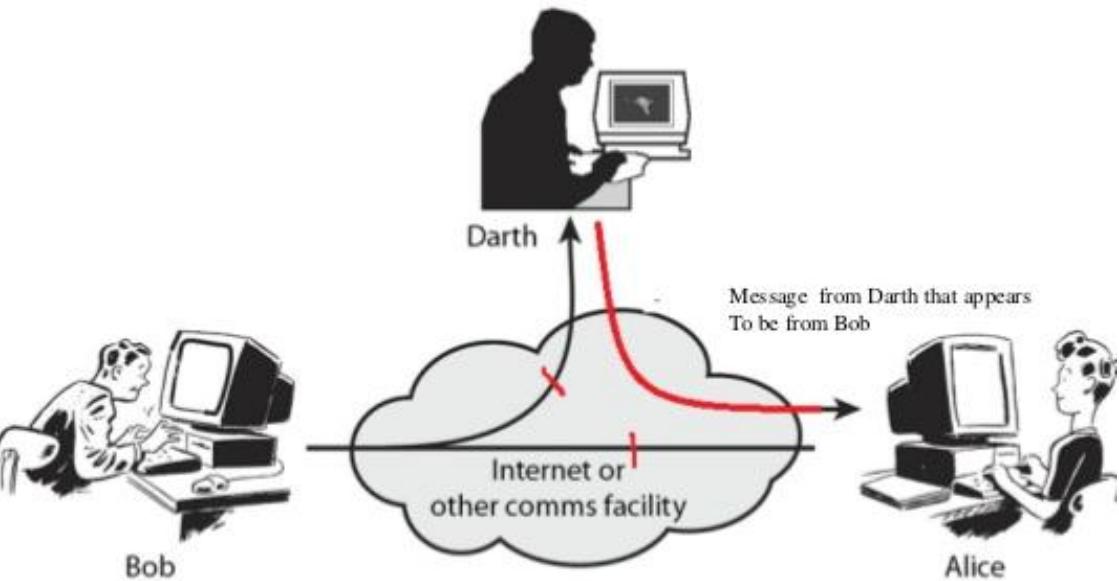
**PASSIVE ATTACK**

# Types of Attacks

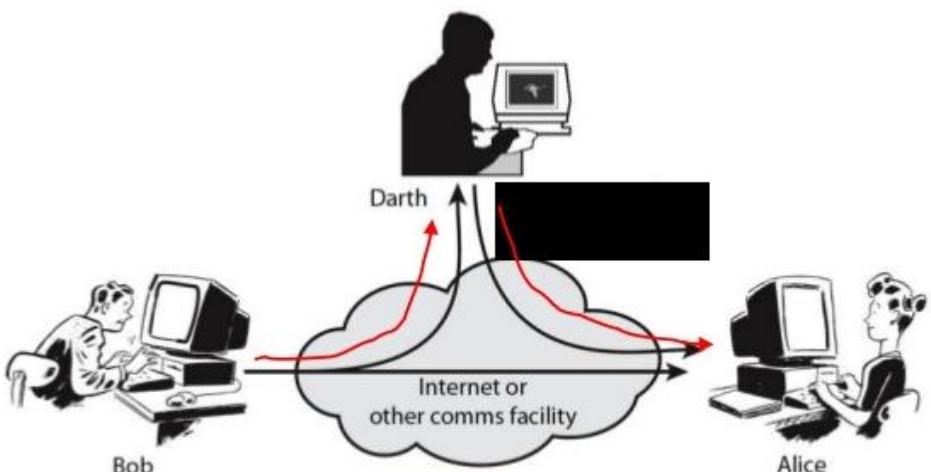
## ACTIVE ATTACK

1. Are attacks in which the hacker attempts to change or transform the content of the message or information
2. An active attack attempts to alter system resources or effect their operations
3. These attacks are threat to **THE INTEGRITY** and **AVAILABILITY** of the system.
- 4. EASIER TO DETECT THAN TO PREVENT**
5. Thus, instead of preventing, emphasis should be on detection of attack and recovery from any disruption or delay caused by it.

# Types of ACTIVE Attacks

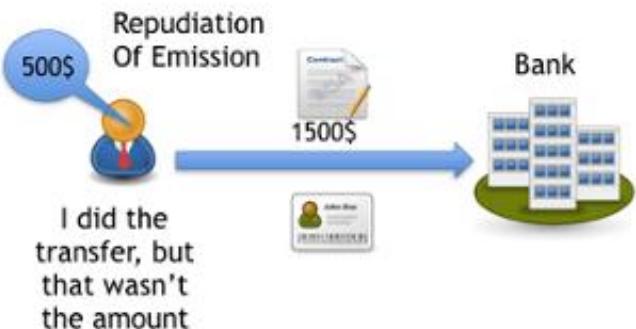
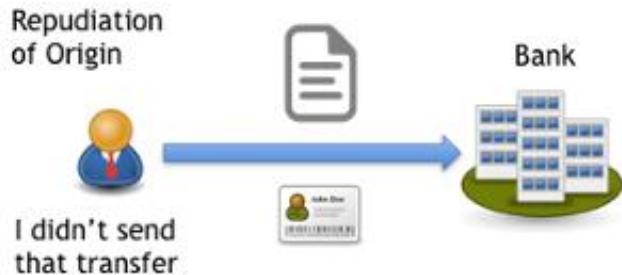


**MASQUERADE**



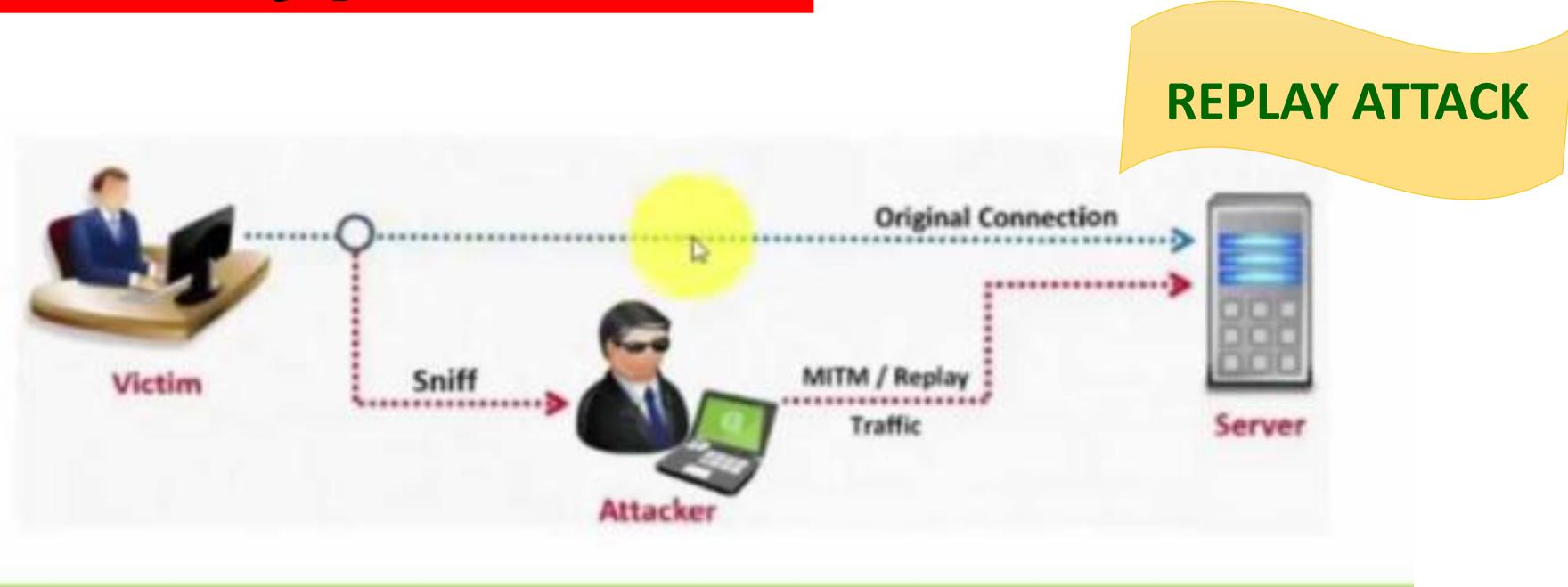
**MODIFICATION OF MESSAGES**

# Types of ACTIVE Attacks



**REPUDIATION**

# Types of ACTIVE Attacks



**Denial of Service (DoS):** DoS is one of the oldest forms of attack. It may slow down or totally interrupt the service of a system

**EXAMPLE:** If a railway website is brought down, it fails to serve the people who want to book tickets.

# Types of Attacks

---

## PASSIVE ATTACK

1. Are the ones in which the attacker observes all the messages and copies the content of messages
2. They focus on monitoring all the transmission and gaining the data
3. No potential harm to the system.
4. But significant danger to your data's **CONFIDENTIALITY**.
- 5. DIFFICULT TO DETECT.**
- 6. PREVENTION IS POSSIBLE** using some **ENCRYPTION** techniques

# Types of PASSIVE ATTACK

## 1. RELEASE OF MESSAGE CONTENT (SNOOPING)

Snooping refers to unauthorized access to or interception of data.

## 2. TRAFFIC ANALYSIS

Traffic analysis refers to obtaining some other type of information by monitoring online traffic.

### SUMMARY OF ATTACKS

#### ACTIVE ATTACKS

1. Masquerade
2. Modification of messages
3. Repudiation
4. Replaying
5. Denial of Service

#### PASSIVE ATTACKS

1. Release of messages
2. Traffic Analysis

# Active vs. Passive Attack

ACTIVE ATTACK	PASSIVE ATTACK
Tries to change the system resources or affect their operation.	Tries to read or make use of information from the system but does not influence system resources.
Modification in information take place.	Modification in the information does not take place.
It is danger for Integrity as well as availability.	Passive Attack is danger for Confidentiality.
In active attack attention is on detection.	While in passive attack attention is on prevention.
Due to active attack system is always damaged.	While due to passive attack, there is no any harm to the system.
Victim gets informed about the attack.	Victim does not get informed about the attack.
information collected through passive attacks are used during executing.	While passive attack are performed by collecting the information such as passwords, messages by itself.

# Active vs. Passive Attack

Attacks	Passive/Active	Threatening
Snooping Traffic analysis	Passive	Confidentiality
Modification Masquerading Replaying Repudiation	Active	Integrity
Denial of service	Active	Availability



# Security Service

- ITU-T provides some security services and some mechanisms to implement those services
- Security and mechanisms are closely related.
- A security service makes use **of one or more security mechanisms**



# **TYPES OF SECURITY SERVICES**

---

- 1. Data Confidentiality**
- 2. Data Integrity**
- 3. Authentication**
- 4. Non-Repudiation**
- 5. Access Control**



# Security Services (X.800)

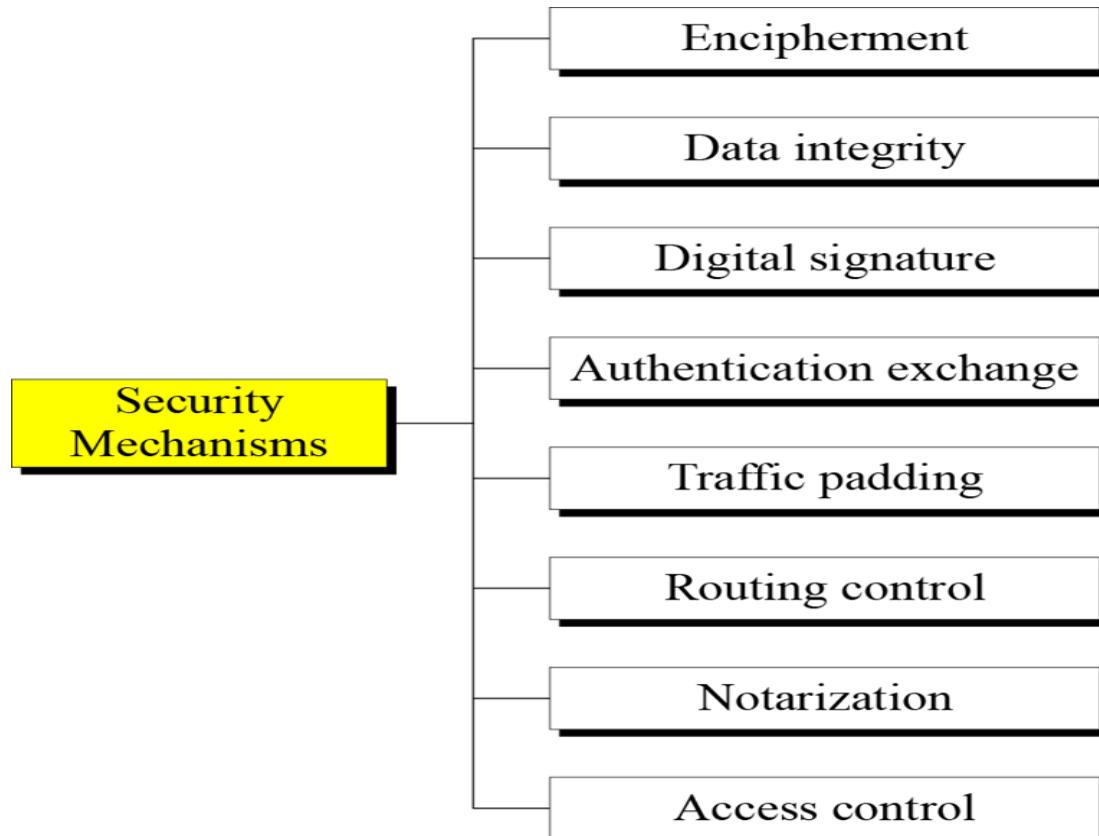
**X.800 defines security services in 5 major categories**

- **Data Confidentiality** – protection of data from unauthorized disclosure
- **Data Integrity** - assurance that data received is as sent by an authorized entity
- **Authentication** - assurance that the communicating entity is the one claimed
- **Non-Repudiation** - protection against denial by one of the parties in a communication
- **Access Control** - prevention of the unauthorized use of a resource



# Security Mechanism

A mechanism that is designed to detect, prevent, or recover from a security attack



# Security Mechanisms

---

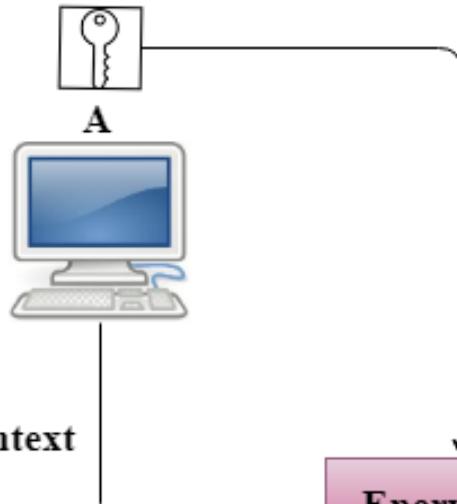
1. **Encipherment:** This security mechanism deals with hiding and covering of data which helps data to become confidential.
2. **Data Integrity:** This security mechanism is used by appending value to data to which is created by data itself.
3. **Digital Signature:** This security mechanism is achieved 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.
4. **Authentication Exchange:** Two entities exchange some messages to prove their identity to each other.



A's public



A's private

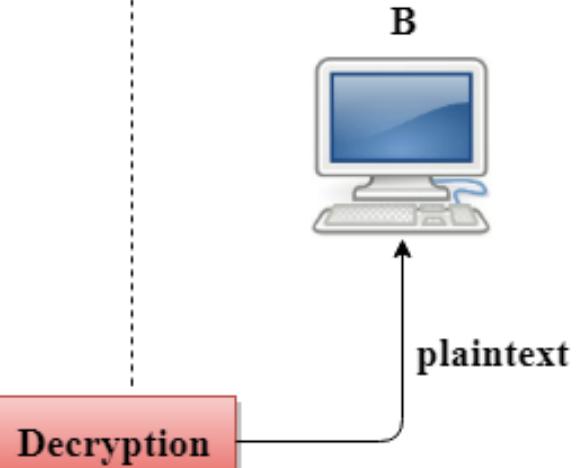


\*

ciphertext

Network

ciphertext



Decryption

B

plaintext

# Security Mechanisms

---

5. **Traffic Padding:** Means inserting some bogus data into the data traffic to thwart the adversary's attempt to use the traffic analysis
6. **Routing Control:** Means selecting and continuously changing different available router between sender and receiver to prevent the opponent from eavesdropping on a particular route.
7. **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.
8. **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.



# Relation between Services and Mechanisms

---

<i>Security Service</i>	<i>Security Mechanism</i>
Data confidentiality	Encipherment and routing control
Data integrity	Encipherment, digital signature, data integrity
Authentication	Encipherment, digital signature, authentication exchanges
Nonrepudiation	Digital signature, data integrity, and notarization
Access control	Access control mechanism



# Relation between Services and Mechanisms

---

<i>Security Service</i>	<i>Security Mechanism</i>
Data confidentiality	Encipherment and routing control
Data integrity	Encipherment, digital signature, data integrity
Authentication	Encipherment, digital signature, authentication exchanges
Nonrepudiation	Digital signature, data integrity, and notarization
Access control	Access control mechanism

# Cryptography

## Basic terms

1. **Plaintext:** Original text
2. **Cipher text:** Coded or scrambled message produced as output
3. **Encryption or Enciphering:** converts original text into ciphertext.
4. **Decryption or Deciphering:** Restoring the plaintext from ciphertext
5. **Secret Key:** Set of values that is provided as an input to encryption algorithm in order to produce cipher text
6. **Cryptography:** Various schemes used for encryption constitute the area of study known as Cryptography
7. **Cryptanalysis:** breaking of “secret codes”
8. **Cryptology: Cryptography + Cryptanalysis**

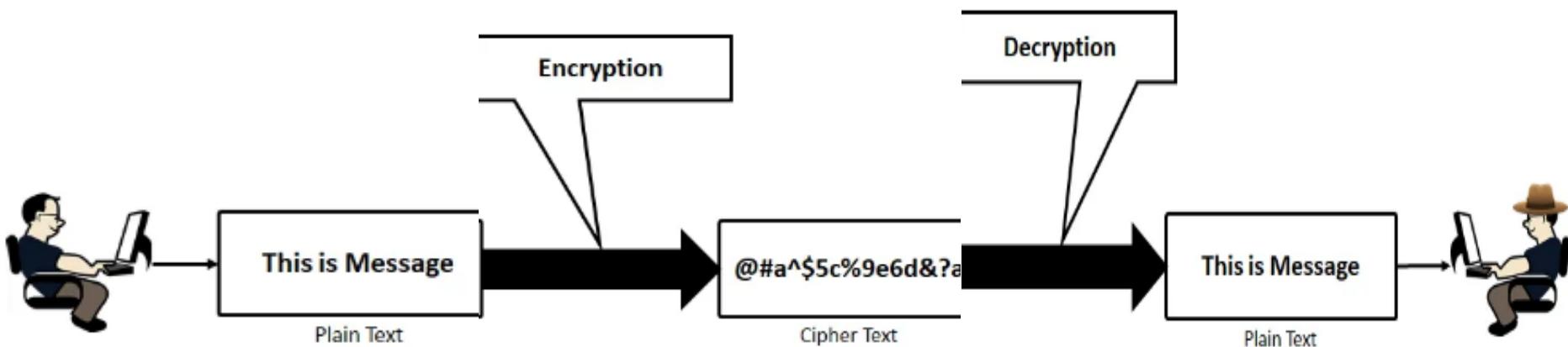
Cryptology is the art and science of making and breaking “secret codes”



# TYPES OF CRYPTOGRAPHY



- Crypt – ‘hidden’ Graphy – ‘writing’.
- In layman language, hiding information from the outside world and let only the right receiver know how to see it.
- 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

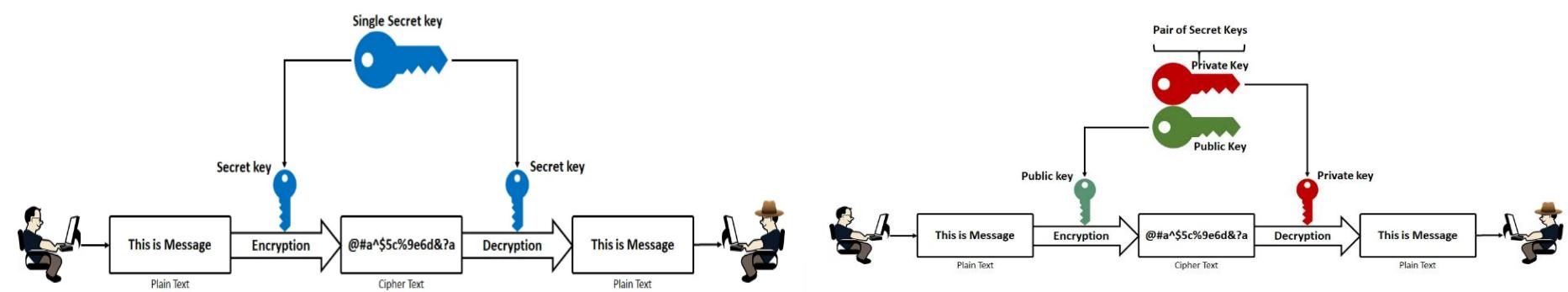


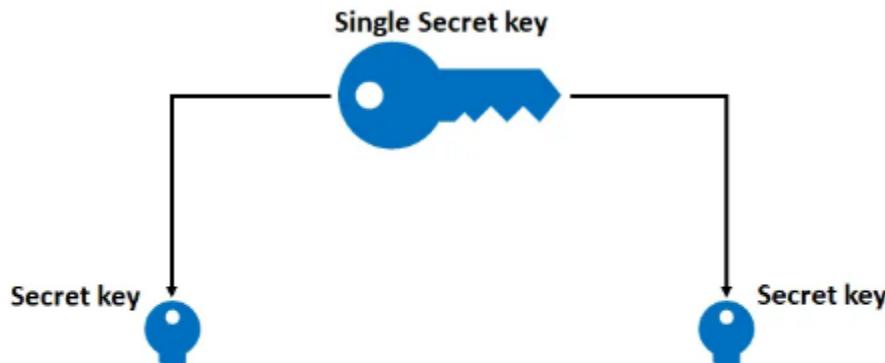
# TYPES OF CRYPTOGRAPHY



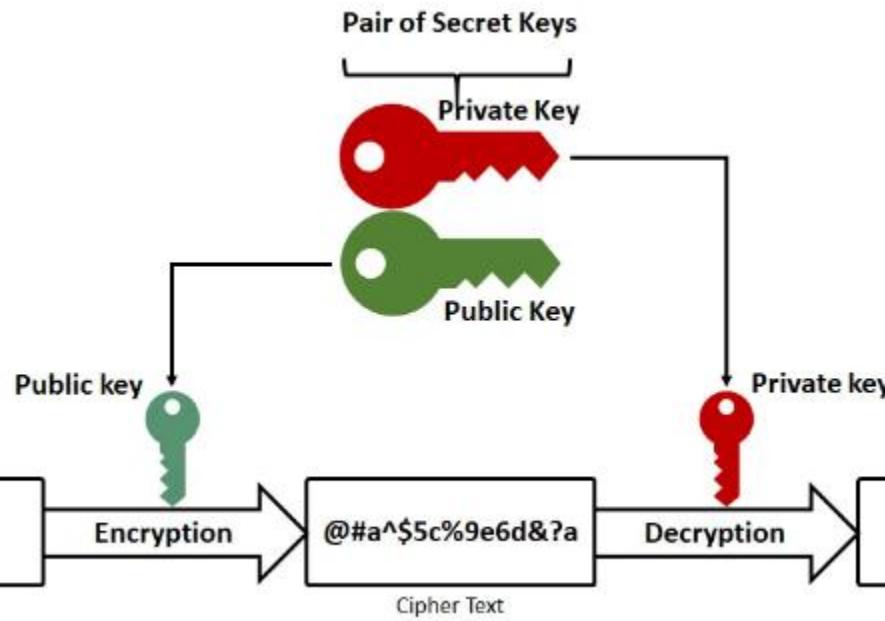
Based on **types of keys** used for encryption and decryption operations, there are **two** types of cryptography

Symmetric Key Cryptography	Asymmetric Key Cryptography
Also called Private key Cryptography	Also called as Public-key cryptography
Both the sender and receiver will use the same key for encrypting and decrypting the message.	A public key is used for encryption and a private key is used for decryption





Symmetric Cryptography



# Cryptography

## Basic terms

1. **Plaintext:** Original text
2. **Cipher text:** Coded or scrambled message produced as output
3. **Encryption or Enciphering:** converts original text into ciphertext.
4. **Decryption or Deciphering:** Restoring the plaintext from ciphertext
5. **Secret Key:** Set of values that is provided as an input to encryption algorithm in order to produce cipher text
6. **Cryptography:** Various schemes used for encryption constitute the area of study known as Cryptography
7. **Cryptanalysis:** breaking of “secret codes”
8. **Cryptology: Cryptography + Cryptanalysis**

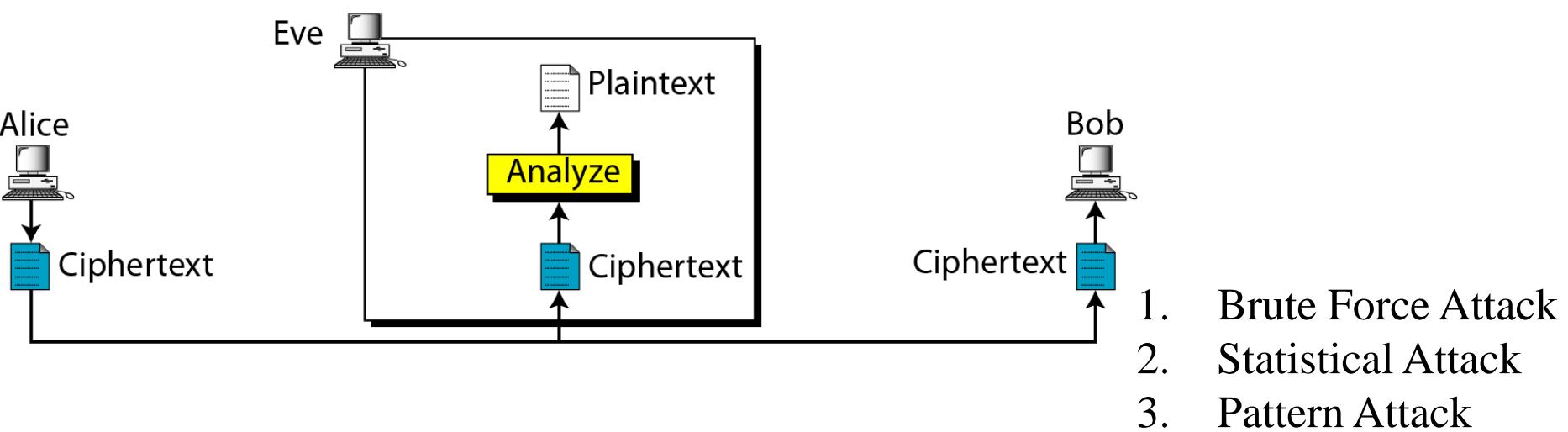
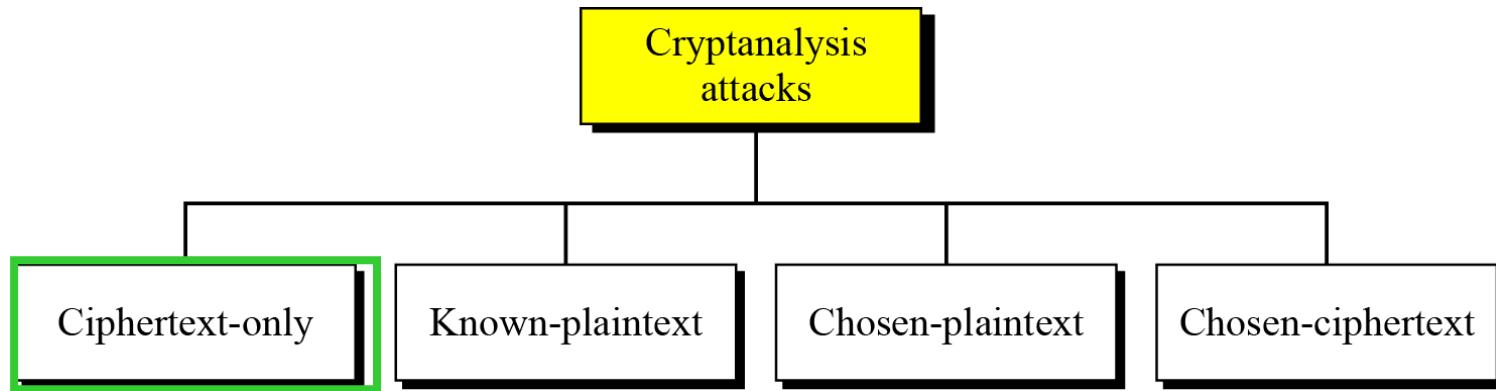
Cryptology is the art and science of making and breaking “secret codes”



# TYPES OF CRYPTANALYSIS ATTACK

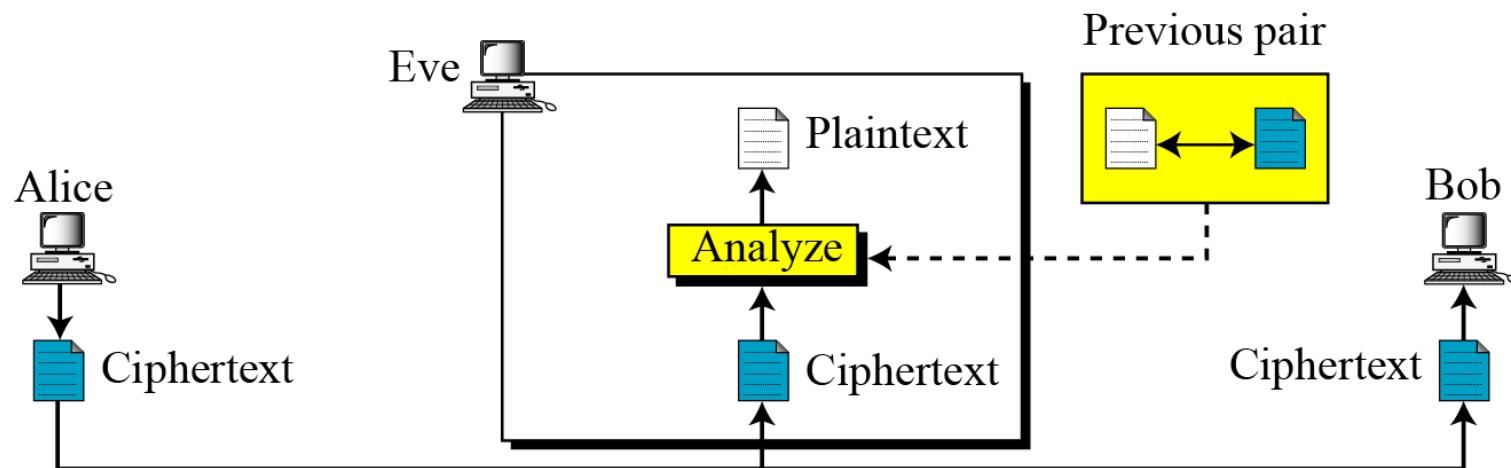
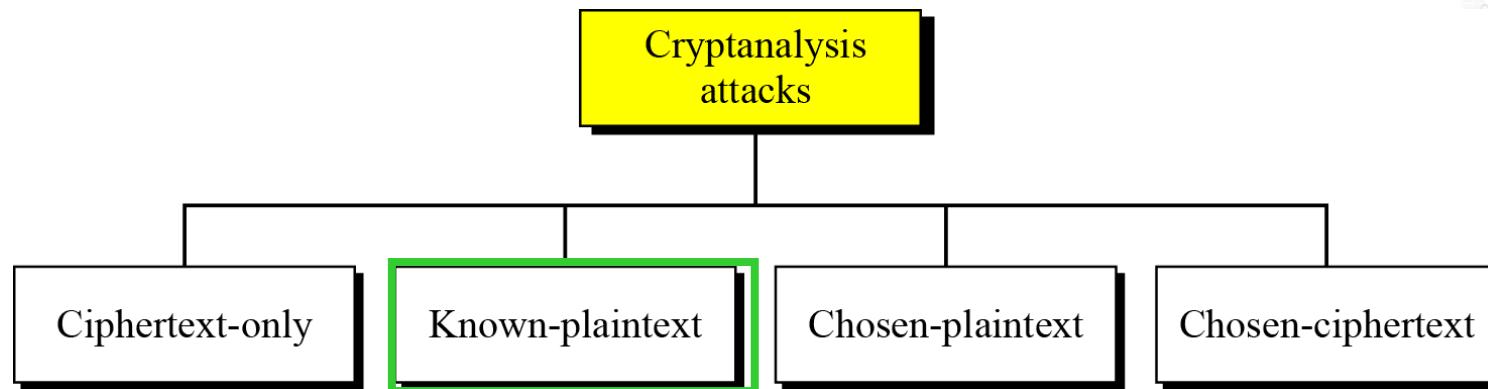


As cryptography is the science and art of creating secret codes, cryptanalysis is the science and art of breaking those codes

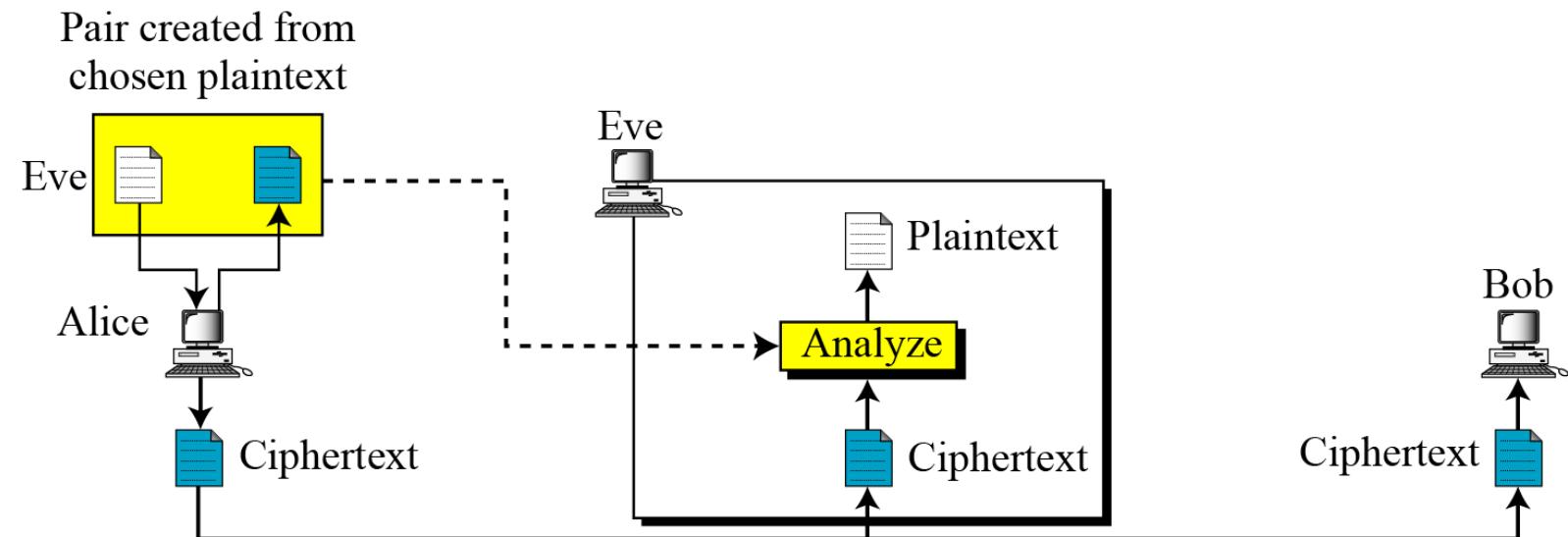
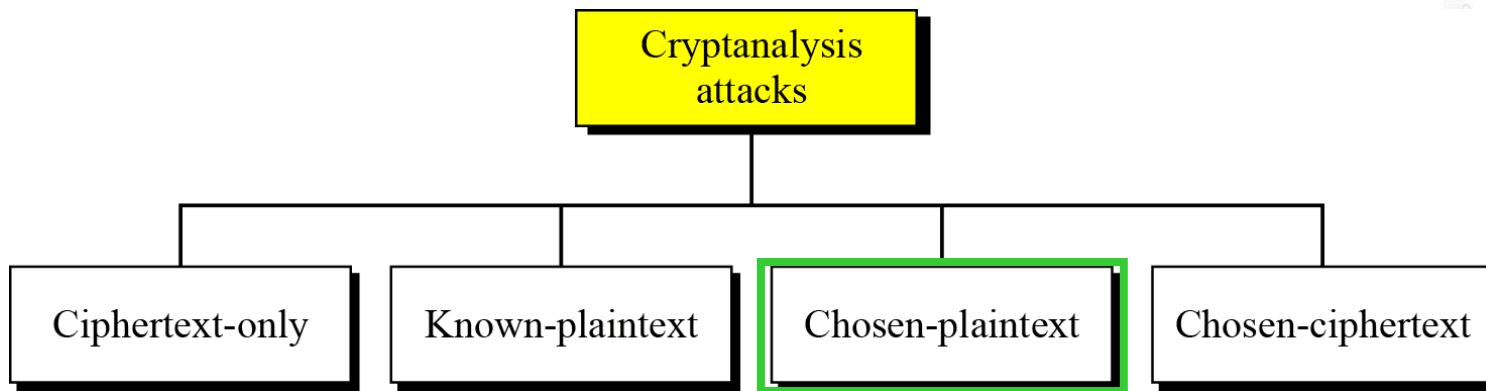


<i>Letter</i>	<i>Frequency</i>	<i>Letter</i>	<i>Frequency</i>	<i>Letter</i>	<i>Frequency</i>	<i>Letter</i>	<i>Frequency</i>
E	12.7	H	6.1	W	2.3	K	0.08
T	9.1	R	6.0	F	2.2	J	0.02
A	8.2	D	4.3	G	2.0	Q	0.01
O	7.5	L	4.0	Y	2.0	X	0.01
I	7.0	C	2.8	P	1.9	Z	0.01
N	6.7	U	2.8	B	1.5		
S	6.3	M	2.4	V	1.0		

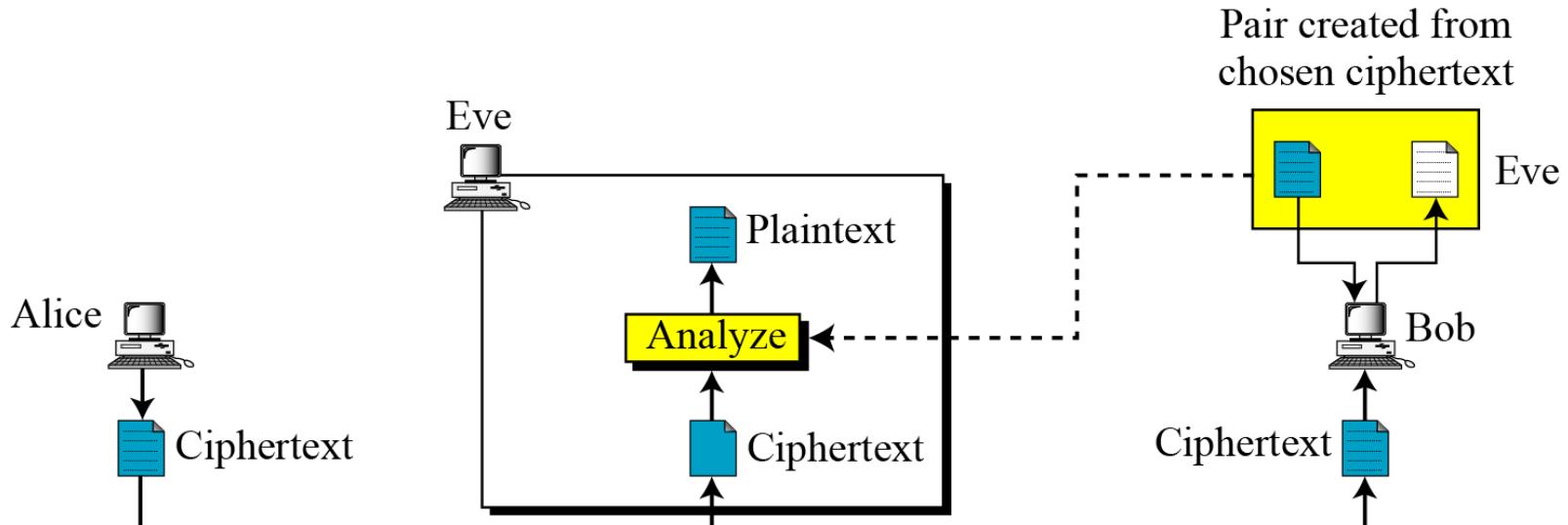
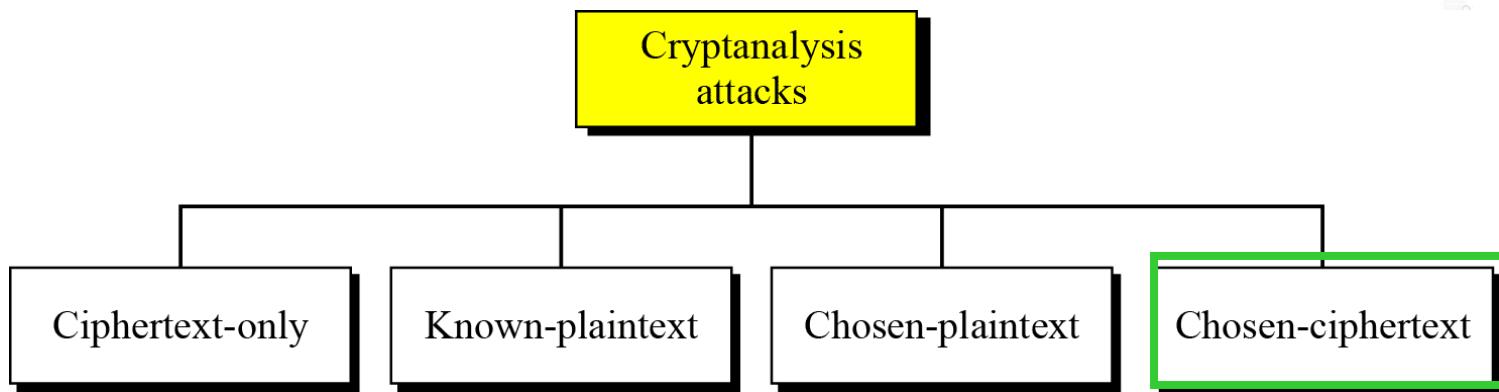
# TYPES OF CRYPTANALYSIS ATTACK



# TYPES OF CRYPTANALYSIS ATTACK



# TYPES OF CRYPTANALYSIS ATTACK



# CATEGORIES OF TRADITIONAL CIPHERS

## Classic Cryptosystems

### Substitution Cipher

- Mono alphabetic Cipher
  - Additive Cipher
  - Multiplicative Cipher
  - Affine Cipher

- Poly Alphabetic cipher
  - Autokey Cipher
  - Playfair Cipher
  - Vigenere Cipher
  - Hill Cipher
  - One time pad Cipher

### Transposition Cipher

- Keyless
- Keyed



# SUBSTITUTION CIPHER

---

1. A substitution cipher replaces one symbol with another.
2. Substitution ciphers can be categorized as either **monoalphabetic ciphers** or **polyalphabetic ciphers**.

## *Monoalphabetic Ciphers*

In monoalphabetic substitution, the relationship between a symbol in the plaintext to a symbol in the ciphertext is always one-to-one.

**Plaintext:** hello

**Ciphertext:** KHOOR

**Plaintext:** hello

**Ciphertext:** ABNZF



# CATEGORIES OF TRADITIONAL CIPHERS

## Classic Cryptosystems

### Substitution Cipher

- Mono alphabetic Cipher
  - Additive Cipher
  - Multiplicative Cipher
  - Affine Cipher

- Poly Alphabetic cipher
  - Autokey Cipher
  - Playfair Cipher
  - Vigenere Cipher
  - Hill Cipher
  - One time pad Cipher

### Transposition Cipher

- Keyless
- Keyed



# ADDITIVE CIPHER

1. Also called as SHIFT CIPHER and sometimes as “CAESAR CIPHER”.
2. The term additive cipher better reveals its mathematical nature.

## ENCRYPTION

$$C = (P + k) \bmod 26$$

## DECRYPTION

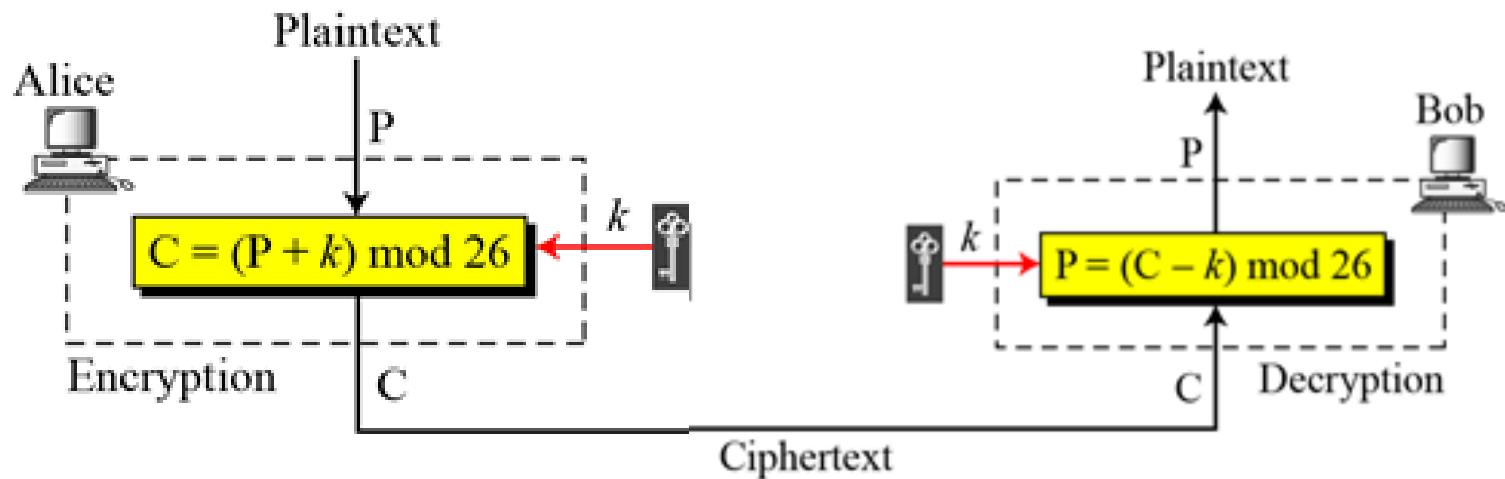
$$P = (C - k) \bmod 26$$

Plaintext →	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
Ciphertext →	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
Value →	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

When the cipher is additive, the plaintext, ciphertext, and key are integers in  $\mathbb{Z}_{26}$ .



# Additive Cipher



Plaintext →	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
Ciphertext →	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
Value →	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

Use the additive cipher with key = 15 to encrypt the message “hello”.

## Solution

We apply the encryption algorithm to the plaintext, character by character:

PLAIN TEXT	ENCRYPTION	CIPHERTEXT
<b>h → 07</b>	$(07 + 15) \bmod 26$	<b>22 → W</b>
<b>e → 04</b>	$(04 + 15) \bmod 26$	<b>19 → T</b>
<b>l → 11</b>	$(11 + 15) \bmod 26$	<b>00 → A</b>
<b>l → 11</b>	$(11 + 15) \bmod 26$	<b>00 → A</b>
<b>o → 14</b>	$(14 + 15) \bmod 26$	<b>03 → D</b>

hello

ENCRYPTED

WTAAD

ENCRYPTION  
 $C = (P + k) \bmod 26$



Plaintext →	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
Ciphertext →	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
Value →	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

## DECRYPTION

$$P = (C - k) \bmod 26$$

### Solution:

We apply the decryption algorithm to the plaintext character by character:

Ciphertext: W → 22	Decryption: $(22 - 15) \bmod 26$	Plaintext: 07 → h
Ciphertext: T → 19	Decryption: $(19 - 15) \bmod 26$	Plaintext: 04 → e
Ciphertext: A → 00	Decryption: $(00 - 15) \bmod 26$	Plaintext: 11 → l
Ciphertext: A → 00	Decryption: $(00 - 15) \bmod 26$	Plaintext: 11 → l
Ciphertext: D → 03	Decryption: $(03 - 15) \bmod 26$	Plaintext: 14 → o



# SHIFT CIPHER and CAESAR CIPHER

---

- Historically, additive ciphers are called **Shift Ciphers**.
- Julius Caesar used an additive cipher to communicate with his officers.
- For this reason, additive ciphers are sometimes referred to as the **Caesar cipher**.
- Caesar used a **key of 3** for his communications.

**Note**

**Additive ciphers are sometimes referred to as shift ciphers or Caesar cipher.**



# **CRYPTANALYSIS OF ADDITIVE CIPHER**

---

- Are vulnerable to CIPHER TEXT ONLY Attack using Exhaustive key searches.
- The key domain of Additive cipher is very small i.e. only 26 keys
- Since one of the key is ZERO, we are left with only 25 keys, due to which an attacker can easily launch a BRUTE – FORCE ATTACK on Ciphertext

## **ADDITIVE CIPHER are Subject to**

1. BRUTE FORCE ATTACK
2. STATISTICAL ATTACK



## Example 3.5

Eve has intercepted the ciphertext “UVACLYFZLJBYL”. Show how she can use a brute-force attack to break the cipher.

### Solution

Eve tries keys from 1 to 7. With a key of 7, the plaintext is “not very secure”, which makes sense.

**Ciphertext:** UVACLYFZLJBYL

<b>K = 1</b>	→	<b>Plaintext:</b> tuzbkxeykiaxk
<b>K = 2</b>	→	<b>Plaintext:</b> styajwdxjhzwj
<b>K = 3</b>	→	<b>Plaintext:</b> rsxzivcwigyvi
<b>K = 4</b>	→	<b>Plaintext:</b> qrwyhubvhfxuh
<b>K = 5</b>	→	<b>Plaintext:</b> pqvxgtaugewtg
<b>K = 6</b>	→	<b>Plaintext:</b> opuwfsztfdvsf
<b>K = 7</b>	→	<b>Plaintext:</b> notverysecure



# *Statistical attack on Additive Cipher*

**Table 3.1 Frequency of characters in English**

Letter	Frequency	Letter	Frequency	Letter	Frequency	Letter	Frequency
E	12.7	H	6.1	W	2.3	K	0.08
T	9.1	R	6.0	F	2.2	J	0.02
A	8.2	D	4.3	G	2.0	Q	0.01
O	7.5	L	4.0	Y	2.0	X	0.01
I	7.0	C	2.8	P	1.9	Z	0.01
N	6.7	U	2.8	B	1.5		
S	6.3	M	2.4	V	1.0		

**Table 3.2 Frequency of diagrams and trigrams**

Digram	TH, HE, IN, ER, AN, RE, ED, ON, ES, ST, EN, AT, TO, NT, HA, ND, OU, EA, NG, AS, OR, TI, IS, ET, IT, AR, TE, SE, HI, OF
Trigram	THE, ING, AND, HER, ERE, ENT, THA, NTH, WAS, ETH, FOR, DTH



## 3.2.1 *Continued*

### Example 3.6

Eve has intercepted the following ciphertext. Using a statistical attack, find the plaintext.

XLILSYWIMWRSAJSVWEPIJSVJSYVQMPPMSRHSPEVWMXMWASVX-LQSVILY-  
VVCFIJSVIXLIWIPPIVIGIMZIWQSVISJJIVW

### Solution

When Eve tabulates the frequency of letters in this ciphertext, she gets: I =14, V =13, S =12, and so on. The most common character is I with 14 occurrences. This means key = 4.

the house is now for sale for four million dollars it is worth more hurry before the seller receives more offers

# CATEGORIES OF TRADITIONAL CIPHERS

## Classic Cryptosystems

### Substitution Cipher

#### Mono alphabetic Cipher

→ Additive Cipher

→ Multiplicative Cipher

→ Affine Cipher

#### Poly Alphabetic cipher

→ Autokey Cipher

→ Playfair Cipher

→ Vigenere Cipher

→ Hill Cipher

→ One time pad Cipher

### Transposition Cipher

→ Keyless  
→ Keyed



**The following word was encrypted using a Caesar cipher with a shift of 2: ecguct.  
What word is it?**

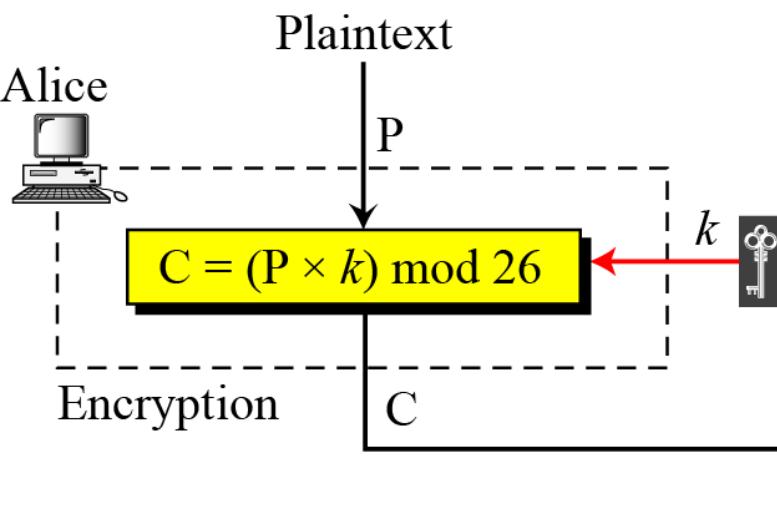
**Who was the first know user of the Caesar Cipher?**

# MULTIPLICATIVE CIPHER

1. Encryption algorithm specifies multiplication of the plaintext by the key
2. Decryption algorithm specifies division of the ciphertext by the key
3. Key belongs to  $Z_{26}^*$

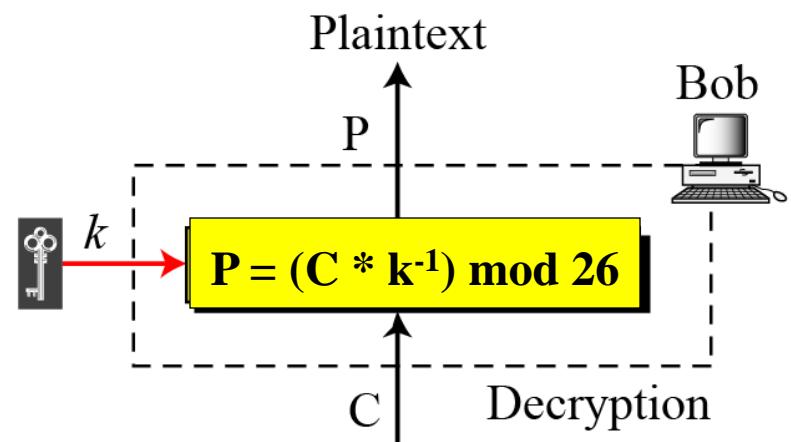
## ENCRYPTION

$$C = (P * k) \bmod 26$$



## DECRYPTION

$$P = (C * k^{-1}) \bmod 26$$



$Z_{26}^*$  include 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, 25

Use a multiplicative cipher to encrypt the message “ankita” with a key of 4

<b>PLAIN TEXT</b>	<b>ENCRYPTION</b>	<b>CIPHERTEXT</b>
a - 00	$(00 * 4) \text{ mod } 26$	00 – A
n – 13	$(13 * 4) \text{ mod } 26$	00 – A
k – 10	$(10 * 4) \text{ mod } 26$	14 -- O
i – 08	$(08 * 4) \text{ mod } 26$	06 -- G
t – 19	$(19 * 4) \text{ mod } 26$	24 -- Y
a -- 00	$(00 * 4) \text{ mod } 26$	00 -- A

Plaintext →	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
Ciphertext →	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
Value →	00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

## Example

---

1. What is the key domain for any multiplicative cipher?
2. We use a multiplicative cipher to encrypt the message “hello” with a key of 7

Plaintext: h → 07

Encryption:  $(07 \times 07) \bmod 26$

ciphertext: 23 → X

Plaintext: e → 04

Encryption:  $(04 \times 07) \bmod 26$

ciphertext: 02 → C

Plaintext: l → 11

Encryption:  $(11 \times 07) \bmod 26$

ciphertext: 25 → Z

Plaintext: l → 11

Encryption:  $(11 \times 07) \bmod 26$

ciphertext: 25 → Z

Plaintext: o → 14

Encryption:  $(14 \times 07) \bmod 26$

ciphertext: 20 → U



# Multiplicative Inverse

Inverses mod 26

$x$	1	3	5	7	9	11	15	17	19	21	23	25
$x^{-1}$	1	9	21	15	3	19	7	23	11	5	17	25



# Multiplicative Inverse

1. Find Multiplicative Inverse of 11 in  $Z_{26}$

$$\begin{aligned}r_1 &= 26 \\r_2 &= 11\end{aligned}$$

Terms used:

1.  $r_1$  – greatest of 2 nos.
2.  $r_2$  - Smallest no.  
 $(r_2, r_1)$
3.  $t_1 = 0$
4.  $t_2 = 1$
5.  $t = t_1 - t_2 * q$

<b>q</b>	<b>r1</b>	<b>r2</b>	<b>r</b>	<b>t1</b>	<b>t2</b>	<b>t</b>
2	26	11	4	0	1	-2
2	11	4	3	1	-2	5
1	4	3	1	-2	5	-7
3	3	1	0	5	-7	26
	1	0		-7	26	



# AFFINE CIPHER

---

1. It is the combination of ADDITIVE and MULTIPLICATIVE cipher.
2. It uses a combination of both ciphers with a pair of keys.
3. First key --- Multiplicative Cipher
4. Second Key --- Additive Cipher

**ENCRYPTION**

$$C = (P * k1 + k2) \text{ mod } 26$$

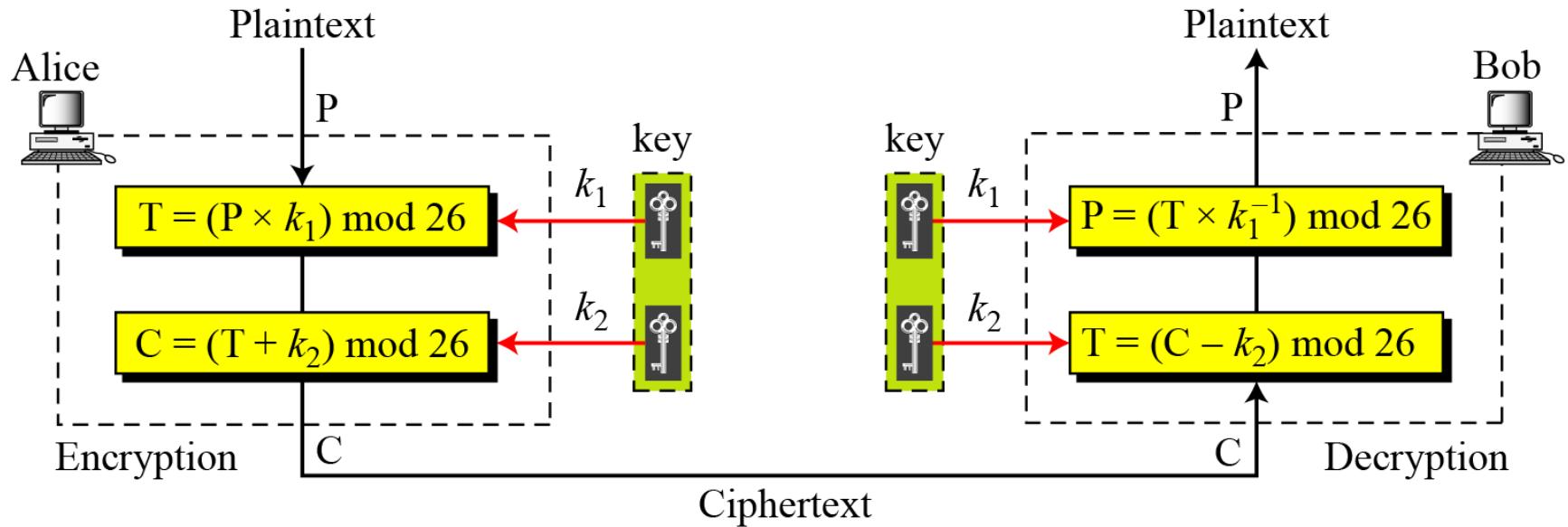
**DECRYPTION**

$$P = ([C - k2] * k^{-1}) \text{ mod } 26$$



# AFFINE CIPHER

## Affine Ciphers



$$C = (P \times k_1 + k_2) \text{ mod } 26$$

$$P = ((C - k_2) \times k_1^{-1}) \text{ mod } 26$$

where  $k_1^{-1}$  is the multiplicative inverse of  $k_1$  and  $-k_2$  is the additive inverse of  $k_2$

# AFFINE CIPHER

---

1. The Affine Cipher uses a pair of keys in which;

First key is from  $Z_{26}^*$

Second Key is from  $Z_{26}$

Thus, the size of the key domain is  $26 * 12 = 312$ .

# AFFINE CIPHER (ENCRYPTION)

Use an affine cipher to encrypt the message “hello” with the key pair (7, 2).

PLAIN TEXT	ENCRYPTION	CIPHERTEXT
<b>h → 07</b>	$(07 * 7 + 2) \text{ mod } 26$	<b>25 → Z</b>
<b>e → 04</b>	$(04 * 7 + 2) \text{ mod } 26$	<b>04 → E</b>
<b>l → 11</b>	$(11 * 7 + 2) \text{ mod } 26$	<b>01 → B</b>
<b>l → 11</b>	$(11 * 7 + 2) \text{ mod } 26$	<b>01 → B</b>
<b>o → 14</b>	$(14 * 7 + 2) \text{ mod } 26$	<b>22 → W</b>



# AFFINE CIPHER (DECRYPTION)

Use the affine cipher to decrypt the message “ZEBBW” with the key pair (7, 2) in modulus 26.

## Solution

$$C: Z \rightarrow 25$$

$$\text{Decryption: } ((25 - 2) \times 7^{-1}) \bmod 26$$

$$P: 07 \rightarrow h$$

$$C: E \rightarrow 04$$

$$\text{Decryption: } ((04 - 2) \times 7^{-1}) \bmod 26$$

$$P: 04 \rightarrow e$$

$$C: B \rightarrow 01$$

$$\text{Decryption: } ((01 - 2) \times 7^{-1}) \bmod 26$$

$$P: 11 \rightarrow 1$$

$$C: B \rightarrow 01$$

$$\text{Decryption: } ((01 - 2) \times 7^{-1}) \bmod 26$$

$$P: 11 \rightarrow 1$$

$$C: W \rightarrow 22$$

$$\text{Decryption: } ((22 - 2) \times 7^{-1}) \bmod 26$$

$$P: 14 \rightarrow o$$



# Monoalphabetic Substitution Cipher

Because additive, multiplicative, and affine ciphers have small key domains, they are very vulnerable to brute-force attack.

A better solution is to create a mapping between each plaintext character and the corresponding ciphertext character. Alice and Bob can agree on a table showing the mapping for each character.

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

*An example key for monoalphabetic substitution cipher*



### 3.2.1 *Continued*

#### Example 3.13

We can use the key in Figure 3.12 to encrypt the message

this message is easy to encrypt but hard to find the key

The ciphertext is

ICFVQRVVNEFVRNVSIYRGAHSLIOJICNHTIYBFGTICRXRS

# POLYALPHABETIC CIPHER

1. In polyalphabetic substitution, each occurrence of a character may have a different substitute.
2. The relationship between a character in the plaintext to a character in the ciphertext is one-to-many.
3. Hides the letter frequency of the underlying language
4. Each Ciphertext character is dependent on
  - Corresponding Plaintext Character.
  - Position of Plaintext Character.
5. A stream of keys,  $k = (k_1, k_2, k_3, \dots)$  is used in which  $k_i$  is used to encipher  $i^{\text{th}}$  character in the plaintext to create  $i^{\text{th}}$  character in Ciphertext

## Poly Alphabetic cipher

- Autokey Cipher
- Playfair Cipher
- Vigenere Cipher
- Hill Cipher
- One time pad Cipher



# AUTOKEY CIPHER

Plaintext: followwolf  
Autokey: P

1. The key is stream of subkeys.
2. Each subkey is used to encrypt the corresponding character in the plaintext.
3. The 1<sup>st</sup> subkey is predetermined value secretly agreed upon by the sender and receiver
4. The 2<sup>nd</sup> subkey is the value of the first plaintext character [ 0 - 25 ]
5. The name itself implies that the subkeys are automatically created from the plaintext character during the encryption process.

Plaintext	f	o	l	l	o	w	w	o	l	f
P's value	5	14	11	11	14	22	22	14	11	5
Keys	15	05	14	11	11	14	22	22	14	11
C's Value	20	19	25	22	25	10	18	10	25	16
Ciphertext	U	T	Z	W	Z	K	S	K	Z	Q



# AUTOKEY CIPHER

---

$$K = (k_1, P_1, P_2, \dots)$$

## ENCRYPTION

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

## DECRIPTION

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

## CRYPTANALYSIS

1. Vulnerable to Brute-Force Attack.
2. The first sub-key can be only one of the 25 values .
3. Polyalphabetic ciphers should not only hide the characteristic of the language but should also have large domain.

Example:  $k_1 = 12$

Plaintext: "Attack is today".



## 3.2.2 *Continued*

### Example 3.14

$k_1 = 12$

Plaintext: “Attack is today”.

Plaintext:	a	t	t	a	c	k	i	s	t	o	d	a	y
P's Values:	00	19	19	00	02	10	08	18	19	14	03	00	24
Key stream:	12	00	19	19	00	02	10	08	18	19	14	03	00
C's Values:	12	19	12	19	02	12	18	00	11	7	17	03	24
Ciphertext:	M	T	M	T	C	M	S	A	L	H	R	D	Y

# PLAYFAIR CIPHER

1. Was used by British Army during World War I
2. The secret key in this cipher is made of 25 alphabet letters arranged in  $5 \times 5$  matrix
3. Different arrangements of the letters in the matrix can create many different secret keys.



If the letters in a pair are same then a bogus letter is inserted to separate them.



If odd characters in plaintext, then add extra character at the end to make no. of characters EVEN.

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

# PLAYFAIR CIPHER

## RULES FOR ENCRYPTION

Plain Text: hello      Cipher Text: **ECQZBX**

Inserting Bogus Character: helxlo

### PAIRING CHARACTERS:

he	lx	lo
EC	QZ	BX

 If two letters in a pair are located in the same row of the secret key, then the corresponding encrypted character for each letter is the next letter to one right in the same row.

 If two letters in a pair are located in the same column of the secret key, then the corresponding encrypted character for each letter is the letter beneath it.

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

Not in same row or column, the corresponding encrypted character for each letter is a letter that is in its own row but in the same column as the other letter



# PLAYFAIR CIPHER

---

$$K = [ (k_1, k_2), (k_3, k_4), \dots ]$$

**ENCRYPTION**

$$C_i = k_i$$

**DECRYPTION**

$$P_i = k_i$$

## CRYPTANALYSIS

1. The size of the key domain is  $25!$ .
2. Brute – Force attack is very difficult
3. Frequency of diagrams are preserved, so a cryptanalyst can use a ciphertext only attack based on the diagram frequency test to find a key.

Example: Plaintext: instruments  
Key: monarchy



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: instruments

**CIPHER TEXT: GATLMZCLRQTX**

# PLAYFAIR CIPHER (DECRYPTION)

Cipher text: CTQVSIRF

Cipher text: CT QV SI RF

Secret Key: INFORMATION

Plain text: ba lx lo on

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

# VIGENERE CIPHER

---

1. Was designed by Blaise de Vigenere
2. The key stream is a repetition of an initial secret key stream of length  $m$ , where  $1 \leq m \leq 26$ .
3.  $(k_1, k_2, k_3, \dots, k_m)$  is the initial secret key agreed upon by sender and receiver.
4. Vigenere key stream does not depend on the plaintext characters, but only on the position of the character in the plaintext.
5. Thus, key stream can be created without knowing the plaintext.

$$P = P_1 P_2 P_3 \dots$$

$$C = C_1 C_2 C_3 \dots$$

$$K = [(k_1, k_2, \dots, k_m), (k_1, k_2, \dots, k_m), \dots]$$

$$\text{Encryption: } C_i = P_i + k_i$$

$$\text{Decryption: } P_i = C_i - k_i$$



# VIGENERE CIPHER (ENCRYPTION)

Plain text: attackatos

Keyword: Pascal

The initial key stream is = (15, 0, 18, 2, 0, 11).

Plaintext	a	t	t	a	c	k	a	t	o	s
P's value	00	19	19	00	02	10	00	19	14	18
Keys	15	00	18	02	00	11	15	00	18	02
C's Value	15	19	11	02	02	21	15	19	06	20
Ciphertext	P	T	L	C	C	V	P	T	G	U

Encrypt the message “She is listening” using the 6-character keyword  
“PASCAL”



# SOLUTION

<b>Plaintext:</b>	s	h	e	i	s	l	i	s	t	e	n	i	n	g
<b>P's values:</b>	18	07	04	08	18	11	08	18	19	04	13	08	13	06
<b>Key stream:</b>	<i>15</i>	<i>00</i>	<i>18</i>	<i>02</i>	<i>00</i>	<i>11</i>	<i>15</i>	<i>00</i>	<i>18</i>	<i>02</i>	<i>00</i>	<i>11</i>	<i>15</i>	<i>00</i>
<b>C's values:</b>	07	07	22	10	18	22	23	18	11	6	13	19	02	06
<b>Ciphertext:</b>	H	H	W	K	S	W	X	S	L	G	N	T	C	G

# HILL CIPHER

Note

The key matrix in the Hill cipher needs to have a multiplicative inverse.

1. Was invented by Lester S. Hill
2. In this, the plain text is divided into equal size blocks
3. The blocks are encrypted one at a time in such a way that each character in the block contributes to the encryption of other characters in the block.
4. Due to above property, Hill cipher belongs to the category of **BLOCK CIPHERS**
5. The key is a square matrix of size  $m \times m$ , in which  $m$  is the size of the block.

$$K = \begin{bmatrix} k_{11} & k_{12} & \dots & k_{1m} \\ k_{21} & k_{22} & \dots & k_{2m} \\ \vdots & \vdots & & \vdots \\ k_{m1} & k_{m2} & \dots & k_{mm} \end{bmatrix}$$

$$\begin{aligned} C_1 &= P_1 k_{11} + P_2 k_{21} + \dots + P_m k_{m1} \\ C_2 &= P_1 k_{12} + P_2 k_{22} + \dots + P_m k_{m2} \\ &\dots \\ C_m &= P_1 k_{1m} + P_2 k_{2m} + \dots + P_m k_{mm} \end{aligned}$$



# HILL CIPHER (ENCRYPTION EXAMPLE)

Plain text: safemessages

Keyword: ciphering

**KEY IS A BLOCK OF 3\*3 matrix**

Thus,  $k =$

C	I	P
H	E	R
I	N	G

Integer  
Equivalent

2	8	15
7	4	17
8	13	6



No. of columns in plaintext matrix should be equal to no of rows in key matrix

Arrange the plaintext in 4\*3 matrix

s	a	f
e	m	e
s	s	a
g	e	s

Integer  
Equivalent

18	00	05
04	12	04
18	18	00
06	04	18



# HILL CIPHER (ENCRYPTION EXAMPLE)

ENCRYPTION :  $C = P * k$

18	00	05
04	12	04
18	18	00
06	04	18

\*

2	8	15
7	4	17
8	13	6

C

24	01	14
20	02	02
06	08	04
02	12	06

=

Y	B	O
U	C	C
G	I	E
C	M	G



# HILL CIPHER (DECRYPTION EXAMPLE)

DECRYPTION :  $P = C * k^{-1}$

24	01	14
20	02	02
06	08	04
02	12	06

\*

2	8	15
7	4	17
8	13	6

-1

$$A^{-1} = \frac{1}{|A|} \text{Adj}(A)$$



# HILL CIPHER (DECRYPTION EXAMPLE)

To find  $k^{-1}$

2	8	15
7	4	17
8	13	6

-1

$$A^{-1} = \frac{1}{|A|} \text{Adj}(A)$$

Now,  $\frac{1}{|k|} = |k|^{-1}$

$$k^{-1} = 1243 * 5 \pmod{26} = 1$$

To find  $|k|$

$$|k| = 2(24 - 221) - 8(42 - 136) + 15(91 - 32)$$

$$|k| = 2(-197) - 8(-94) + 15(59)$$

$$\begin{aligned}|k| &= -394 + 752 + 885 \\ &= 1243\end{aligned}$$



# HILL CIPHER (DECRYPTION EXAMPLE)

To find  $k^{-1}$

2	8	15
7	4	17
8	13	6

-1

To find  $\text{Adj}(A)$ :

$$A^{-1} = \frac{1}{|A|} \text{Adj}(A)$$

2	8	15	2	8
7	4	17	7	4
8	13	6	8	13
2	8	15	2	8
7	4	17	7	4

-197	147	76
94	-108	71
59	38	-48

55	735	380
470	110	355
295	190	20

11	147	76
94	22	71
59	38	4



$$\begin{aligned} -197 + 26(8) &= 11 \\ -108 + 26(5) &= 22 \\ -48 + 26(2) &= 4 \end{aligned}$$

# HILL CIPHER (DECRYPTION EXAMPLE)

To find  $k^{-1}$

2	8	15
7	4	17
8	13	6

$-1$

55	735	380
470	110	355
295	190	20

$\text{mod } 26 =$

3	7	16
2	6	17
9	8	20



# HILL CIPHER (DECRYPTION EXAMPLE)

DECRYPTION :  $P = C * k^{-1}$

24	01	14
20	02	02
06	08	04
02	12	06

\*

3	7	16
2	6	17
9	8	20

18	00	05
04	12	04
18	18	00
06	04	18



s	a	f
e	m	e
s	s	a
g	e	s

Plain text: safemessages



# Example:

---

- Plaintext: “We live in an insecure world”

- Key: 
$$\begin{bmatrix} \underline{\phantom{0}} & \underline{\phantom{0}} \\ 03 & 02 \\ \underline{\phantom{0}} & \underline{\phantom{0}} \\ 05 & 07 \\ \underline{\phantom{0}} & \underline{\phantom{0}} \end{bmatrix}$$

# ONE TIME PAD CIPHER

---

1. Also known as Vernam Cipher (invented by Vernam) or Perfect Cipher.
2. Plaintext is combined with a random key
3. It is the only existing mathematically unbreakable encryption
4. A study by Shannon has shown that perfect secrecy can be achieved if each plaintext symbol is encrypted with a key randomly chosen from a key domain.
5. To encrypt each character a key is randomly chosen from the key domain (00, 01, 02, ...., 25)
6. The key has the same length as the plaintext and is chosen completely random
7. Even if it is a perfect cipher, it is almost impossible to implement commercially.



# ONE TIME PAD CIPHER (ENCRYPTION)

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

hello

ENCRYPTED

EQNVZ

# ONE TIME PAD CIPHER (DECRYPTION)

E	Q	N	V	Z	ciphertext
4 (E)	16 (Q)	13 (N)	21 (V)	25 (Z)	ciphertext
- 23 (X)	12 (M)	2 (C)	10 (K)	11 (L)	key
= -19	4	11	11	14	ciphertext - key
= 7 (H)	4 (E)	11 (L)	11 (L)	14 (O)	ciphertext - key (mod 26)
H	E	L	L	O	→ message

EQNVZ

DECRYPTED

hello

# CATEGORIES OF TRADITIONAL CIPHERS

## Classic Cryptosystems

### Substitution Cipher

#### Mono alphabetic Cipher

→ Additive Cipher

→ Multiplicative Cipher

→ Affine Cipher

#### Poly Alphabetic cipher

→ Autokey Cipher

→ Playfair Cipher

→ Vigenere Cipher

→ Hill Cipher

→ One time pad Cipher

### Transposition Cipher



Keyless  
Keyed



# TRANSPOSITION CIPHER

---

1. A transposition cipher does not substitute one symbol for another, instead it changes the location of the symbols.
2. A symbol in the 1<sup>st</sup> position of the plaintext may appear in the 10<sup>th</sup> position of the ciphertext
3. It transposes the symbols

**Note**

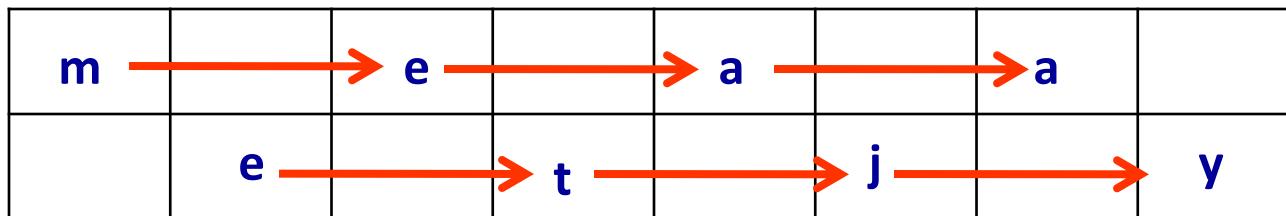
**A transposition cipher reorders symbols.**



# KEYLESS TRANSPOSITION CIPHER

1. Simple transposition ciphers, which were used in the past, are keyless.
2. There are 2 methods for permutation of characters

Text is written into a table column by column and then transmitted row by row

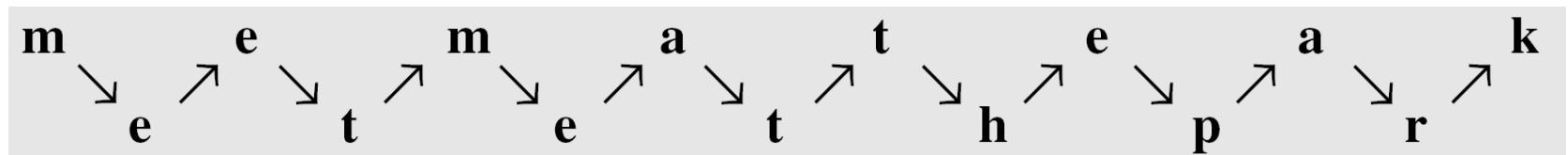


**Plaintext: meet ajay**

**Ciphertext: m e a a e t j y**

## Rail fence cipher:

Plaintext: “Meet me at the park”



Ciphertext: “**MEMATEAKETETHPR**”.

# KEYLESS TRANSPOSITION CIPHER

1. Simple transposition ciphers, which were used in the past, are keyless.
2. There are 2 methods for permutation of characters

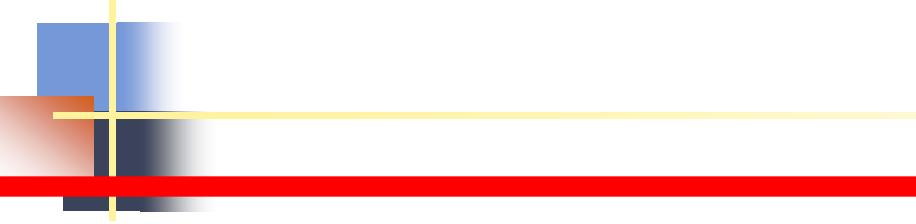
m	e	e	t
a	j	a	y

Text is written into a table row by row and then transmitted column by column

**Plaintext: meet ajay**

**Ciphertext: m a e j e a t y**





Alice and Bob can agree on the number of columns and use the second method. Alice writes the same plaintext, row by row, in a table of four columns.

<b>m</b>	<b>e</b>	<b>e</b>	<b>t</b>
<b>m</b>	<b>e</b>	<b>a</b>	<b>t</b>
<b>t</b>	<b>h</b>	<b>e</b>	<b>p</b>
<b>a</b>	<b>r</b>	<b>k</b>	

She then creates the ciphertext “MMTAEEHREAEKTP”.

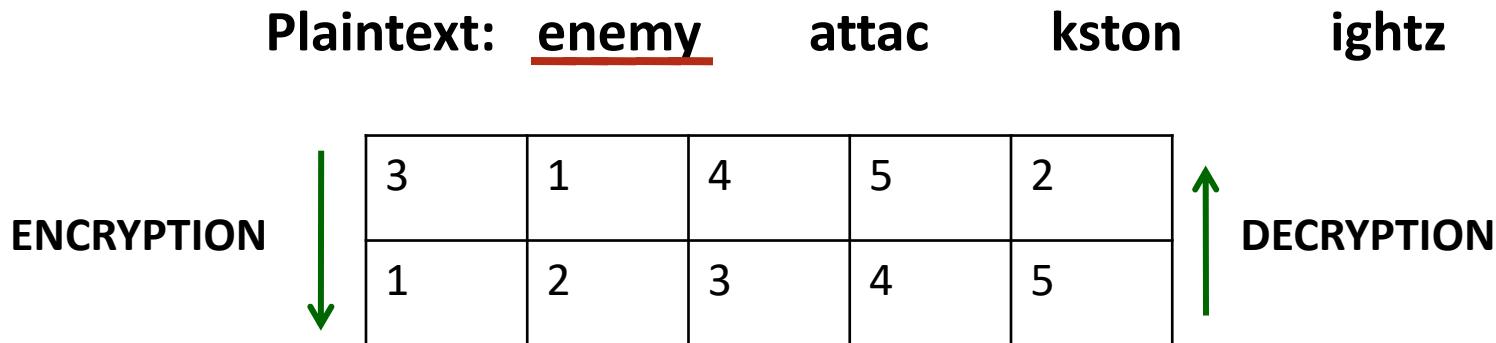


# KEYED TRANSPOSITION CIPHER

1. Divide the plaintext into groups of predetermined size, called blocks
2. A key is used to permute the character in each block separately

**Plaintext: enemy attacks tonight**

Sender and receiver has agreed to divide the text into group of 5 characters



The key used for encryption and decryption is a permutation key

**Ciphertext: E E M Y N TAACT TKONS HITZG**



# COMBINING TWO APPROACHES

1. To achieve better scrambling
2. Encryption or Decryption is done in 3 steps
  - 1.Text is written row by row
  2. Use key for reordering
  3. Read column by column

**Plaintext: enemy attacks tonight**

e	n	e	m	y
a	t	t	a	c
k	s	t	o	n
i	g	h	t	z

ENCRYPTION

3	1	4	5	2
1	2	3	4	5

E	E	M	Y	N
t	a	a	c	t
t	k	o	n	s
h	i	t	z	g

Read by  
column

**Ciphertext: etth eaki maot ycnz ntsg**

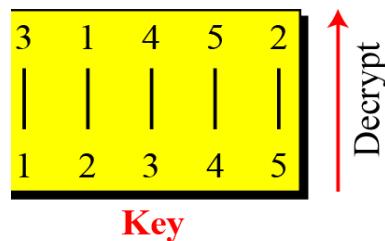




e n e m y a t t a c k s t o n i g h t z

Read row by row

e	n	e	m	y
a	t	t	a	c
k	s	t	o	n
i	g	h	t	z



Decrypt

E	E	M	Y	N
T	A	A	C	T
T	K	O	N	S
H	I	T	Z	G

Write column by column

Transmission

E T T H E A K I M A O T Y C N Z N T S G

Ciphertext

# **CRYPTANALYSIS OF TRANSPOSITION CIPHER**

---

1. These are vulnerable to several kinds of Cipher-text only attack

## A. Statistical Attack

The frequency of the letters is not changed. Thus, single-letter frequency analysis is the 1st attack

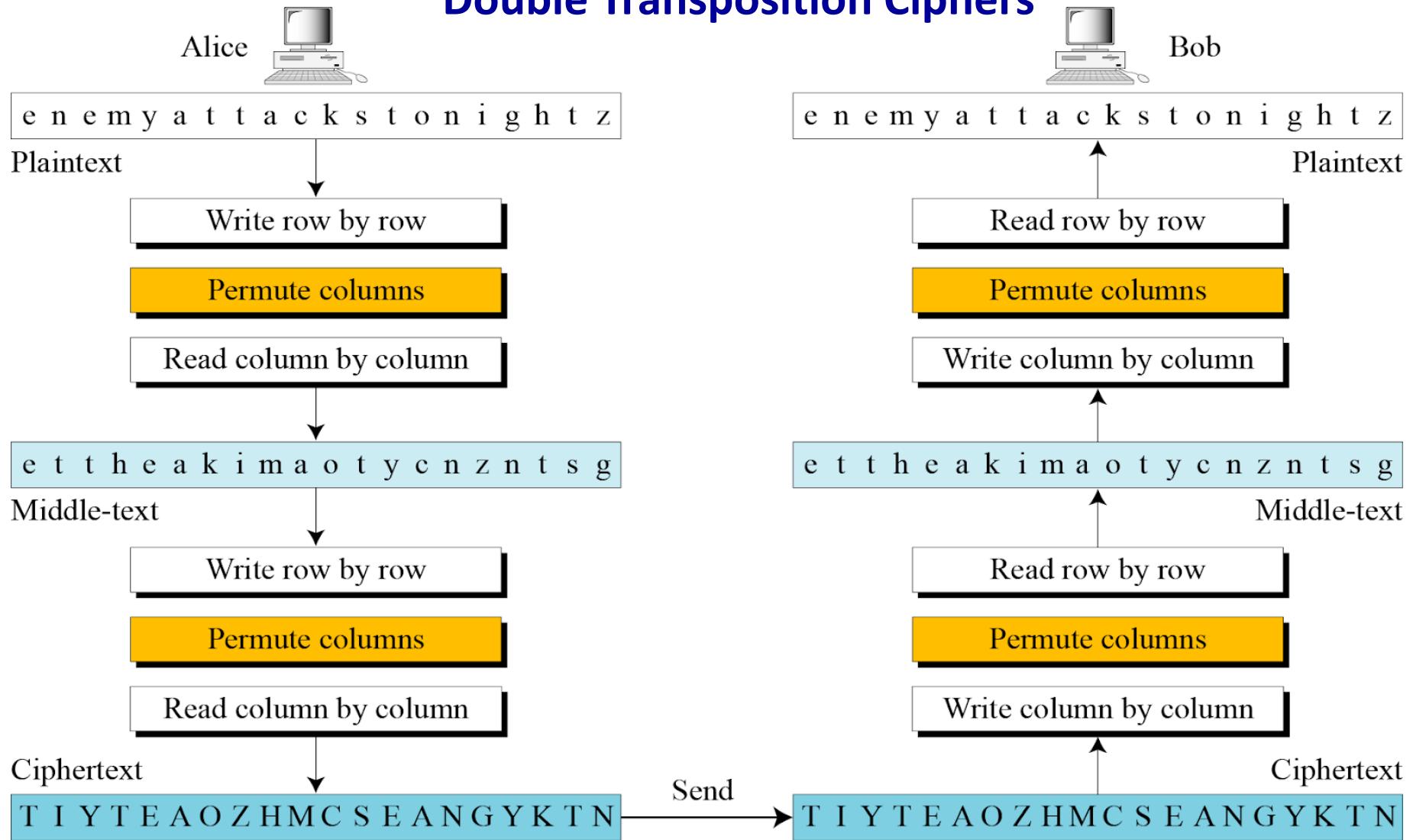
## B. Brute Force Attack

- Attacker can use all possible keys to decrypt the message. Since the no. of keys is huge ( $1! + 2! + 3! + \dots + L!$ )
- A better approach is to guess the number of columns

## C. Pattern Attack



# Double Transposition Ciphers



# STREAM CIPHER

1. Encryption and decryption are done typically on one symbol at a time.

Call the plaintext stream P, the ciphertext stream C, and the key stream K.

$$P = P_1 P_2 P_3, \dots$$

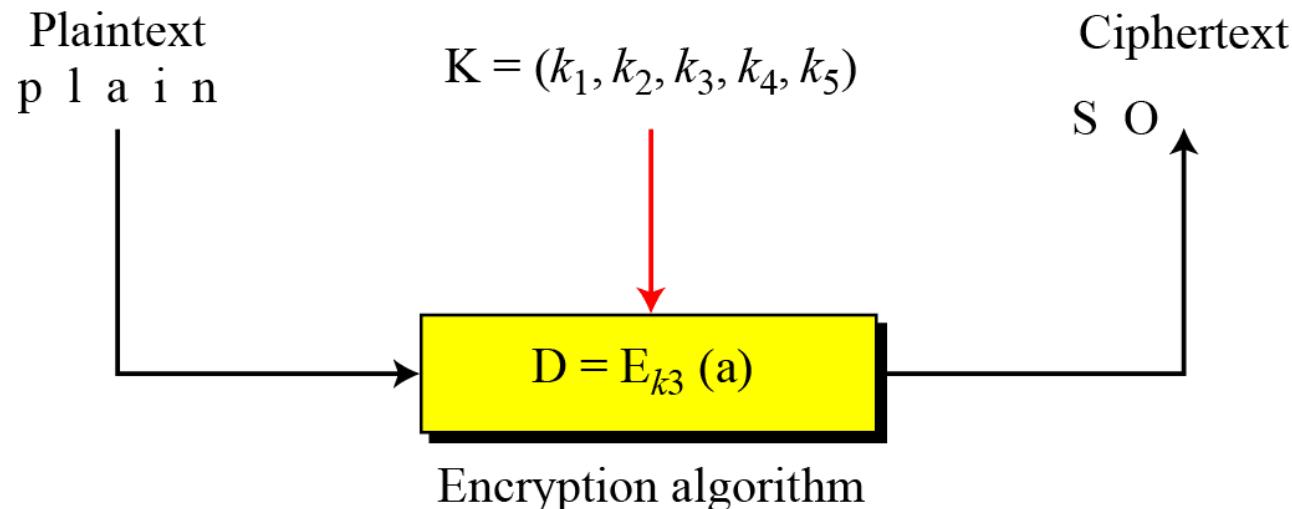
$$C = C_1 C_2 C_3, \dots$$

$$K = (k_1, k_2, k_3, \dots)$$

$$C_1 = E_{k1}(P_1)$$

$$C_2 = E_{k2}(P_2)$$

$$C_3 = E_{k3}(P_3) \dots$$



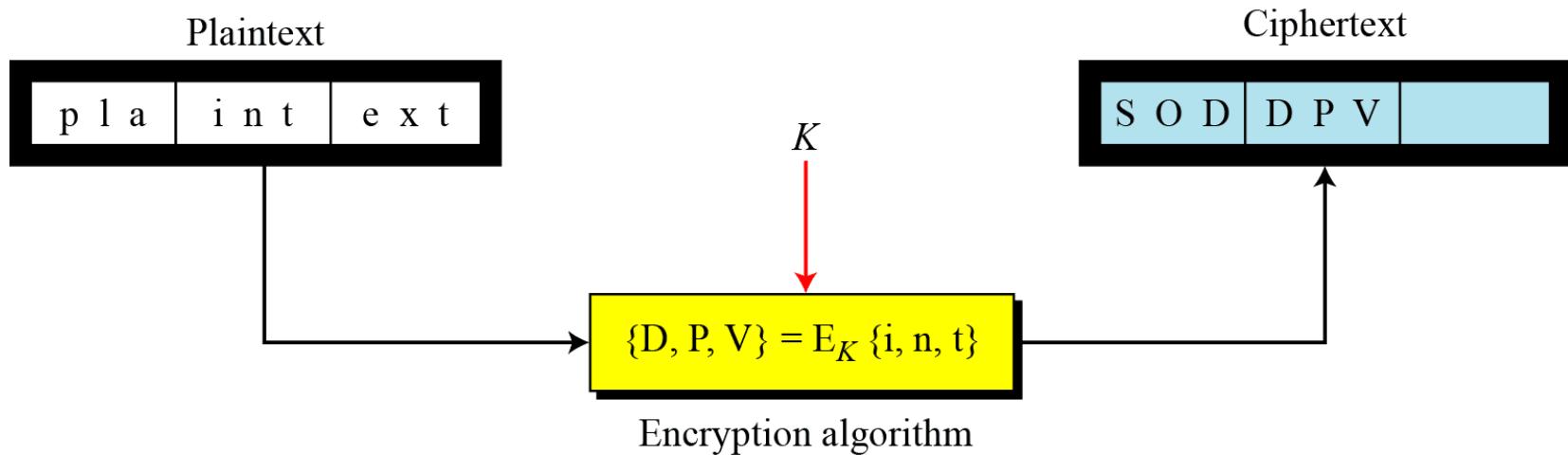
# STREAM CIPHER

---

1. Additive ciphers can be categorized as stream ciphers in which the key stream is the repeated value of the key.
2. The monoalphabetic substitution ciphers discussed in this chapter are also stream ciphers.
3. Vigenere ciphers are also stream ciphers according to the definition.
4. a stream cipher is a monoalphabetic cipher if the value of  $k_i$  does not depend on the position of the plaintext character in the plaintext stream; otherwise, the cipher is polyalphabetic.

# BLOCK CIPHER

In a block cipher, a group of plaintext symbols of size  $m$  ( $m > 1$ ) are encrypted together creating a group of ciphertext of the same size. A single key is used to encrypt the whole block even if the key is made of multiple values. Figure 3.27 shows the concept of a block cipher.



# BLOCK CIPHER

---

1. Playfair ciphers are block ciphers.
2. Hill ciphers are block ciphers.
3. From the definition of the block cipher, it is clear that every block cipher is a polyalphabetic cipher because each character in a ciphertext block depends on all characters in the plaintext block.



We add the bogus character, "z" to the end of the plaintext to make the number of characters multiple of 2. The plaintext matrix, the key matrix, and ciphertext matrix are shown below:

$$\begin{matrix}
 \begin{bmatrix} 8 & 20 \\ 21 & 0 \\ 5 & 18 \\ 11 & 3 \\ 13 & 13 \\ 11 & 3 \\ 22 & 12 \\ 2 & 14 \\ 19 & 10 \\ 6 & 12 \\ 2 & 7 \\ 4 & 25 \end{bmatrix} & = & \begin{bmatrix} 22 & 4 \\ 11 & 8 \\ 21 & 4 \\ 8 & 13 \\ 0 & 13 \\ 8 & 13 \\ 18 & 4 \\ 2 & 20 \\ 17 & 4 \\ 22 & 14 \\ 17 & 11 \\ 3 & 25 \end{bmatrix} & \times & \begin{bmatrix} 3 & 2 \\ 5 & 7 \end{bmatrix} \\
 \mathbf{C} & & \mathbf{P} & & \mathbf{K}
 \end{matrix}$$

The ciphertext is then "IUVAFSLDNNLDWMCOTKGMCHEZ", in which the last character is a bogus character.

"TEKOOHRACIRMNREATANFTETYTGHH", encrypted with a key of 4,

T					E					E
	-				-					
		-		-						
			-							

# Modular Arithmetic and Number Theory

## *Euler's Phi-Function $\phi(n)$*

1. Is sometimes called as Euler's totient function.
2. It plays a very important role in cryptography
3. The function finds the number of integers that are both smaller than  $n$  and relatively prime to  $n$
4. The function  $\phi(n)$  calculates the number of elements in this set

**Use the following to find the value of  $\phi(n)$**

1.  $\phi(1) = 0$ .
2.  $\phi(p) = p - 1$  if  $p$  is a prime.
3.  $\phi(m \times n) = \phi(m) \times \phi(n)$  if  $m$  and  $n$  are relatively prime.  $m \neq n$
4.  $\phi(p^e) = p^e - p^{e-1}$  if  $p$  is a prime.

**Note**

The difficulty of finding  $\phi(n)$  depends on the difficulty of finding the factorization of  $n$ .



# Euler's Phi-Function $\phi(n)$ - Examples

Note

1. What is the value of  $\phi(13)$ ?

Interesting point: If  $n > 2$ , the value of  $\phi(n)$  is even.

**Solution:** Because 13 is a prime,  $\phi(13) = (13 - 1) = 12$ .

2. What is the value of  $\phi(10)$ ?

**Solution:**  $\phi(10) = \phi(2) \times \phi(5)$   
=  $1 * 4$   
= 4

What is the number of elements in  $Z_{14}^*$ ?

**Solution:** 6

3. What is the value of  $\phi(240)$ ?

**Solution:**  $240 = 2^4 \times 3^1 \times 5^1$

$$\begin{aligned}\phi(240) &= (2^4 - 2^3) \times (3^1 - 3^0) \times (5^1 - 5^0) \\ &= 64\end{aligned}$$

4. Can we say that  $\phi(49) = \phi(7) \times \phi(7) = 6 \times 6 = 36$ ?

**Solution:** No. The third rule applies when  $m$  and  $n$  are relatively prime. Here  $49 = 7^2$ .

We need to use the fourth rule:  $\phi(49) = 7^2 - 7^1 = 42$ .



<b>n</b>	<b><math>\phi(n)</math></b>	<b>numbers coprime to n</b>
1	1	1
2	1	1
3	2	1, 2
4	2	1,3
5	4	1,2,3,4
6	2	1,5
7	6	1,2,3,4,5,6
8	4	1,3,5,7
9	6	1,2,4,5,7,8
10	4	1,3,7,9
11	10	1,2,3,4,5,6,7,8,9,10
12	4	1,5,7,11
13	12	1,2,3,4,5,6,7,8,9,10,11,12
14	6	1,3,5,9,11,13
15	8	1,2,4,7,8,11,13,14

# *Euler's Phi-Function $\varphi(n)$ – Examples for practice*

Calculate

- a)  $\Phi(29)$
- b)  $\Phi(32)$
- c)  $\Phi(80)$
- d)  $\Phi(100)$
- e)  $\Phi(101)$

**ANSWERS**

- a) 28
- b) 16
- c) 32
- d) 40
- e) 100

$$\begin{aligned}\Phi(80) &= 2 * 2 * 2 * 2 * 5 \\&= 2^4 * 5 \\&= (2^4 - 2^3) * 4 \\&= (16 - 8) * 4 \\&= 8 * 4 \\&= 32\end{aligned}$$

**Note**

**Interesting point: If  $n > 2$ , the value of  $\varphi(n)$  is even.**



# Fermat's Little Theorem

---

## First Version

*If  $p$  is prime number*

*$a$  is positive integer not divisible by  $p$*

*$a, p$  are co-prime*

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

**Find the result of  $6^{10} \pmod{11}$ .**

**Solution:**  $p = 11$ ,  $a = 6$

$$6^{11-1} \equiv 1 \pmod{11}$$

$$6^{10} \equiv 1 \pmod{11}$$

$$6^{10} \pmod{11} \equiv 1$$



# Fermat's Little Theorem

## Second Version

If  $p$  is prime number

~~a is positive integer not divisible by  $p$~~

$a, p$  are co-prime

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

Find the result of  $3^{12} \pmod{11}$ .

Solution:  $p = 11, a = 3$

Now,  $3^{12} \pmod{11} = (3^{11} * 3^1) \pmod{11}$

$$= (3^{11} \pmod{11}) * (3^1 \pmod{11})$$

$$= 3 * 3$$

$$= 9$$



## Multiplicative Inverse

---

A very interesting application of Fermat's theorem is in finding some multiplicative inverses quickly if the modulus is prime

If  $p$  is prime and  $a$  is an integer such that  $p$  does not divide  $\underline{a}$  then,

$$a^{-1} \bmod p = a^{p-2} \bmod p$$

- a.  $8^{-1} \bmod 17 = 8^{17-2} \bmod 17 = 8^{15} \bmod 17 = 15 \bmod 17$
- b.  $5^{-1} \bmod 23 = 5^{23-2} \bmod 23 = 5^{21} \bmod 23 = 14 \bmod 23$
- c.  $60^{-1} \bmod 101 = 60^{101-2} \bmod 101 = 60^{99} \bmod 101 = 32 \bmod 101$
- d.  $22^{-1} \bmod 211 = 22^{211-2} \bmod 211 = 22^{209} \bmod 211 = 48 \bmod 211$



# Euler's Theorem

---

1. Generalization of Fermat's Theorem
2. The modulus in the Fermat's theorem is prime, whereas the modulus in Euler's theorem is an integer.

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

## First Version

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

## Second Version

$$a^{k \times \varphi(n) + 1} \equiv a \pmod{n}$$

**Note**

The second version of Euler's theorem is used in the RSA cryptosystem in Chapter 4.



## *Chinese Remainder theorem*

---

1. Is used to solve a set of congruent equations with one variable but different moduli, which are relatively prime

$$x \equiv a_1 \pmod{m_1}$$

$$x \equiv a_2 \pmod{m_2}$$

...

$$x \equiv a_k \pmod{m_k}$$

It states that the above equations have a unique solution if the moduli are relatively prime.



# *Chinese Remainder theorem*

---

## Solution To Chinese Remainder Theorem

1. Find  $M = m_1 \times m_2 \times \dots \times m_k$ . This is the common modulus.
2. Find  $M_1 = M/m_1, M_2 = M/m_2, \dots, M_k = M/m_k$ .
3. Find the multiplicative inverse of  $M_1, M_2, \dots, M_k$  using the corresponding moduli  $(m_1, m_2, \dots, m_k)$ . Call the inverses  $M_1^{-1}, M_2^{-1}, \dots, M_k^{-1}$ .
4. The solution to the simultaneous equations is

$$x = (a_1 \times M_1 \times M_1^{-1} + a_2 \times M_2 \times M_2^{-1} + \dots + a_k \times M_k \times M_k^{-1}) \bmod M$$



## *Chinese Remainder theorem (Example)*

---

$$a_1 = 2, \quad a_2 = 3, \quad a_3 = 2$$

$$m_1 = 3, \quad m_2 = 5, \quad m_3 = 7$$

$$x \equiv 2 \pmod{3}$$

$$x \equiv 3 \pmod{5}$$

$$x \equiv 2 \pmod{7}$$

1. Find  $M = m_1 * m_2 * m_3$

$$M = 3 * 5 * 7 = 105$$

$$M_2 = 21$$

2. Find  $M_1 = M / m_1$

$$M_3 = 15$$

$$M_1 = \frac{m_1 * m_2 * m_3}{m_1}$$

$$M_1 = m_2 * m_3$$

$$= 35$$



## *Chinese Remainder theorem (Example)*

---

$$M = 3 * 5 * 7 = 105$$

$$M_1 = 35, M_2 = 21, M_3 = 15$$

$$x \equiv 2 \pmod{3}$$

$$x \equiv 3 \pmod{5}$$

$$x \equiv 2 \pmod{7}$$

3. Find the multiplicative inverse of  $M_1, M_2, \dots, M_k$  using the corresponding moduli  $(m_1, m_2, \dots, m_k)$ . Call the inverses  $M_1^{-1}, M_2^{-1}, \dots, M_k^{-1}$ .

$$M_1^{-1} \pmod{3}$$

$$= 35^{-1} \pmod{3}$$

$$= 35^{3-2} \pmod{3}$$

$$= 35^1 \pmod{3}$$

$$= 2$$

$$M_2^{-1} \pmod{5}$$

$$= 21^{-1} \pmod{5}$$

$$= 21^{5-2} \pmod{5}$$

$$= 21^3 \pmod{5}$$

$$= 1$$

$$\text{Similarly } M_3^{-1} = 1$$



# *Chinese Remainder theorem*

## Solution To Chinese Remainder Theorem

1. Find  $M = m_1 \times m_2 \times \dots \times m_k$ . This is the common modulus.
2. Find  $M_1 = M/m_1, M_2 = M/m_2, \dots, M_k = M/m_k$ .
3. Find the multiplicative inverse of  $M_1, M_2, \dots, M_k$  using the corresponding moduli  $(m_1, m_2, \dots, m_k)$ . Call the inverses  $M_1^{-1}, M_2^{-1}, \dots, M_k^{-1}$ .
4. The solution to the simultaneous equations is

$$x = (a_1 \times M_1 \times M_1^{-1} + a_2 \times M_2 \times M_2^{-1} + \dots + a_k \times M_k \times M_k^{-1}) \bmod M$$

$$a_1=2, \quad a_2=3, \quad a_3=2$$

$$M_1 = 35, \quad M_2 = 21, \quad M_3 = 15$$

$$M_1^{-1} = 2, \quad M_2^{-1} = 1, \quad M_3^{-1} = 1$$

$$x = 233 \bmod 105$$

$$= 23$$



## *Chinese Remainder theorem (example to practise)*

---

$$X \equiv 1 \pmod{5}$$

$$X \equiv 1 \pmod{7}$$

$$X \equiv 3 \pmod{11}$$

Find an integer that has remainder 3 when divided by 7 and 13 ,  
but is divisible by 12.

$$X \equiv 3 \pmod{7}$$

$$X \equiv 3 \pmod{13}$$

$$X \equiv 0 \pmod{12}$$



# Cryptography and System Security (CSS)

## Course Code: CSC 604



### Subject Incharge

Ankita Karia

Assistant Professor

Room No. 421

email: [ankitakaria@sfit.ac.in](mailto:ankitakaria@sfit.ac.in)



# Module 2: Symmetric & Asymmetric Key Cryptography and Key Management

---

## 2.1

- Block Cipher
  - ✓ Block Cipher Principles
  - ✓ Block Cipher Modes of Operation
  - ✓ Data Encryption Standard (DES)
  - ✓ Double DES, Triple DES
  - ✓ Advanced Encryption Standard (AES)
- Stream Cipher
  - ✓ RC5 algorithm



# Block Cipher

---

1. Operates on the blocks of plain text, instead of operating on each bit of plain text separately.
2. Each block is of equal size and has fixed no of bits.
3. The generated ciphertext has blocks equal to the number of blocks in plaintext and also has the same number of bits in each block as of plain text.
4. Block cipher uses the same key for encryption and decryption.



# Block Cipher (Contd...)

---

1. Block cipher is an encryption method which divides the plain text into blocks of fixed size.
2. Each block has an equal number of bits.
3. At a time, block cipher operates only on one block of plain text and applies key on it to produce the corresponding block of ciphertext.
4. While decryption also only one block of ciphertext is operated to produce its corresponding plain text.
5. Data Encryption Standard (DES) is the best example of it.



# Block Cipher Principles

---

The design of the block cipher is based on the three principles, which are

1. Number of rounds
2. Design of function F
3. Key schedule algorithm

- 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.



# Block Cipher Principles

---

The design of the block cipher is based on the three principles, which are

1. Number of rounds

2. Design of function F

3. Key schedule algorithm

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



# Block Cipher Principles

---

The design of the block cipher is based on the three principles, which are

1. Number of rounds
2. Design of function F
3. Key schedule algorithm

This suggests that the schedule key should be able to confirm the strict effect of avalanche and the criterion of bit independence.



# Module 2: Symmetric & Asymmetric Key Cryptography and Key Management

---

## 2.1

- Block Cipher
  - ✓ Block Cipher Principles
  - ✓ Block Cipher Modes of Operation
  - ✓ Data Encryption Standard (DES)
  - ✓ Double DES, Triple DES
  - ✓ Advanced Encryption Standard (AES)
- Stream Cipher
  - ✓ RC5 algorithm



# Block Cipher Modes of Operation

---

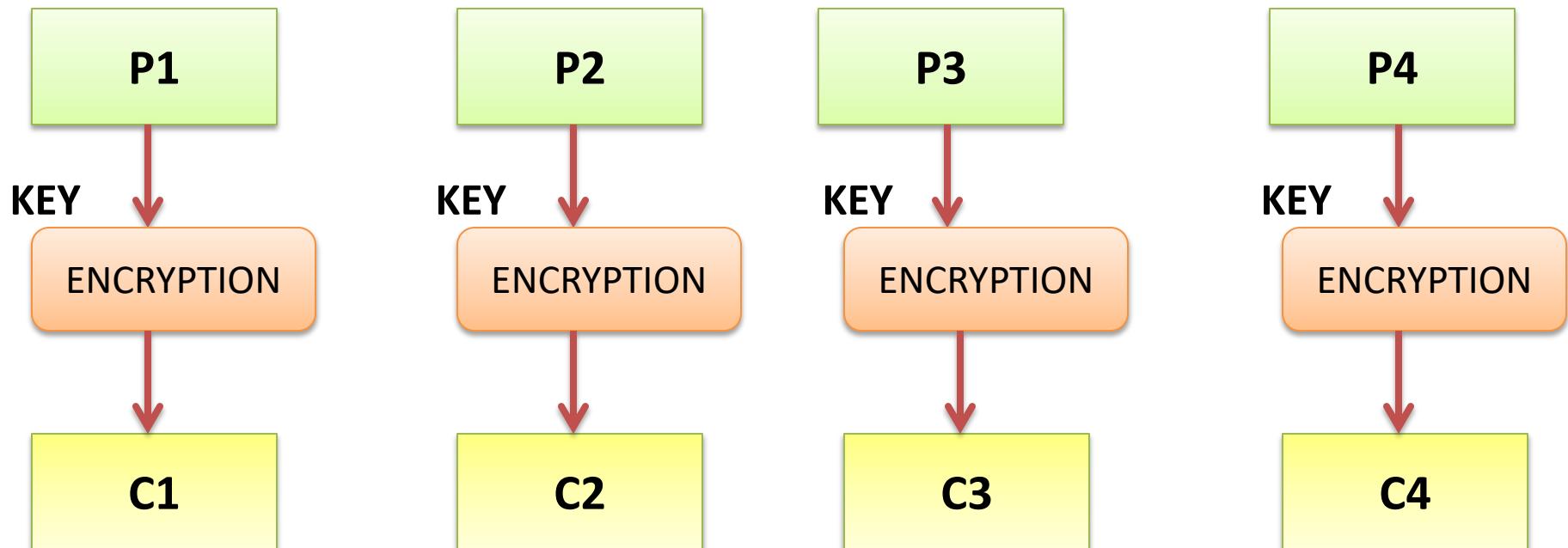
There are five important block cipher modes of operation defined by NIST. These five modes of operation enhance the algorithm so that it can be adapted by a wide range of applications which uses block cipher for encryption.

1. Electronic Code Book Mode (ECB)
2. Cipher Block Chaining Mode (CBC)
3. Cipher Feedback Mode (CFB)
4. Output Feedback Mode (OFB)
5. Counter Mode



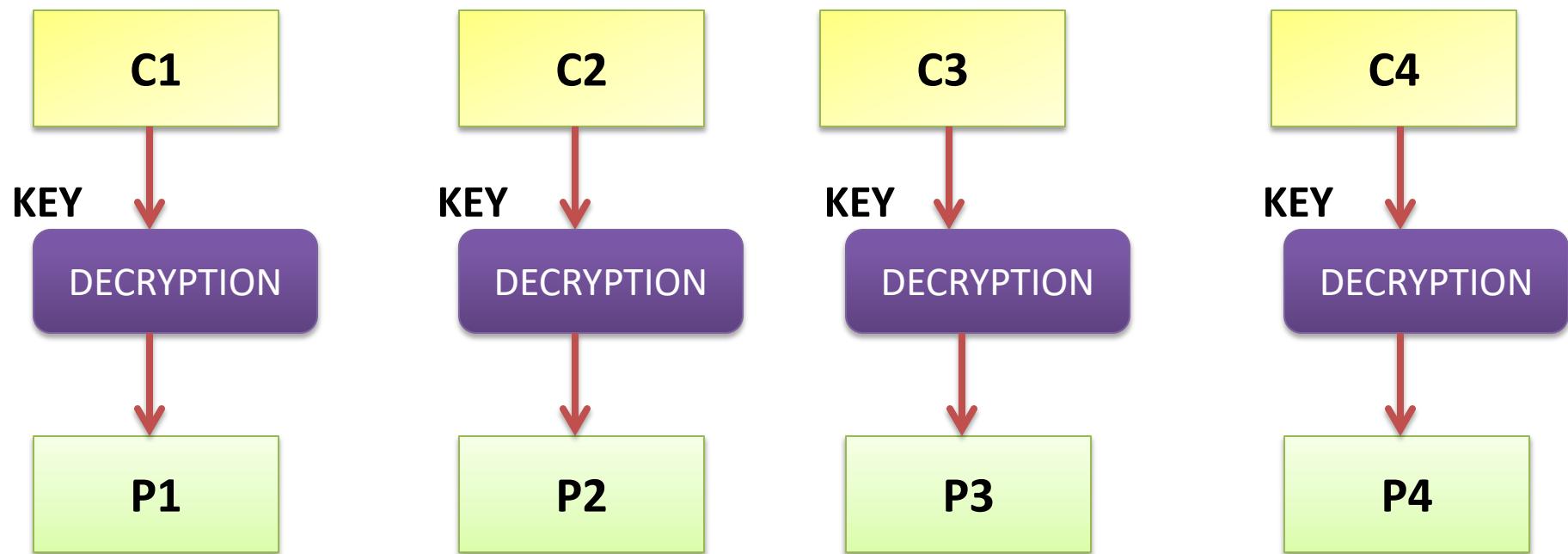
# Electronic Code Book Mode (ECB)

- Easiest block cipher mode of operation.
- The plain text is divided into the blocks, each of 64-bit.
- Each block is encrypted one at a time to produce the cipher block.
- The same key is used to encrypt each block.



# Electronic Code Book Mode (ECB)

- This ciphertext is again divided into blocks, each of 64-bit and each block is decrypted independently one at a time to obtain the corresponding plain text block.
- The same key is used to decrypt each block which was used to encrypt each block..



# **Electronic Code Book Mode (ECB)**

---

## **ADVANTAGES:**

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

## **DISADVANTAGES:**

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



# Cipher Block Chaining Mode (CBC)

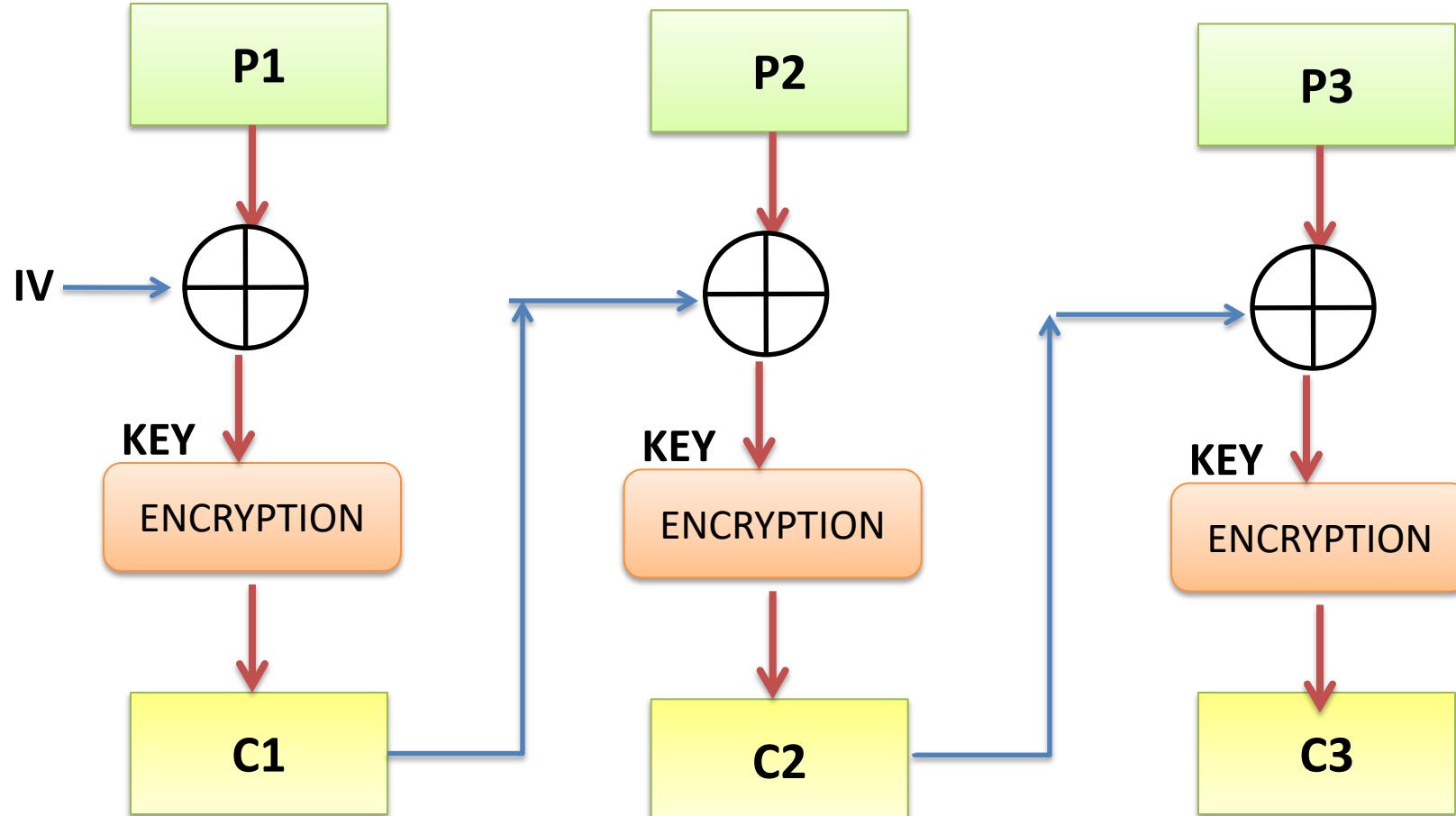
---

1. CBC confirms that even if the plain text has repeating blocks its encryption won't produce same cipher block.
2. Chaining has been added to the block cipher.
3. For this, the result obtained from the encryption of the first plain text block is fed to the encryption of the next plaintext box.
4. Since, during the encryption of first plain text block, no previous plain text block is available so a random block of text is generated called Initialization vector.



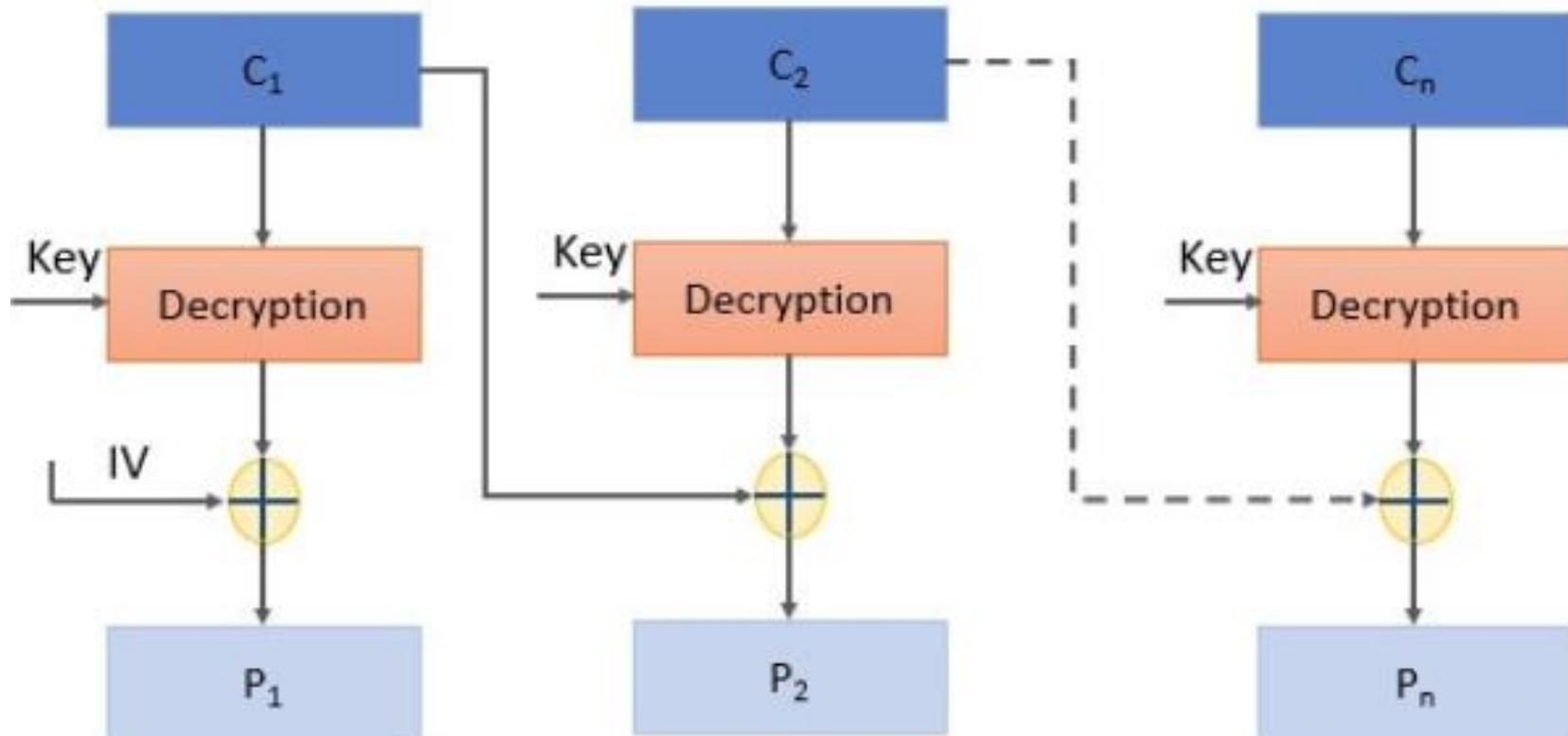
# Cipher Block Chaining Mode (CBC)

ENCRYPTION



# Cipher Block Chaining Mode (CBC)

## DECRYPTION



# Cipher Block Chaining Mode (CBC)

---

## ADVANTAGES:

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

## DISADVANTAGES:

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



# Cipher Feedback Mode (CFB)

---

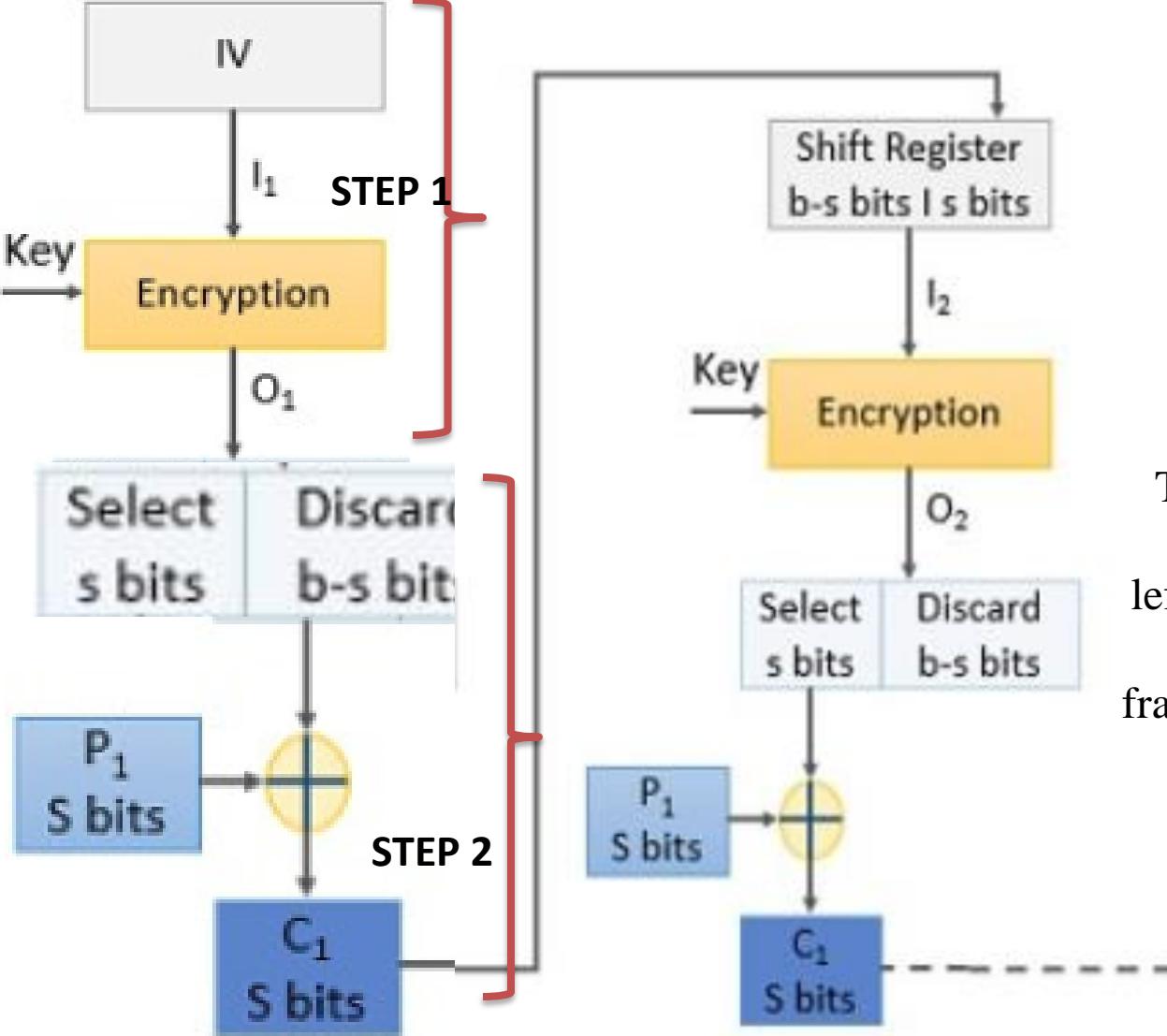
1. All applications may not be designed to operate on the blocks of data, some may be character or bit-oriented.
2. Cipher feedback mode is used to operate on smaller units than blocks.

## The encryption steps in cipher feedback mode:

**Step 1:** Here also we use initialization vector, IV is kept in the shift register and it is encrypted using the key.

**Step 2:** The left most s bits of the encrypted IV is then XORed with the first fragment of the plain text of s bits. It produces the first ciphertext C1 of s bits.





Then again, the encryption is performed on IV and the leftmost s bit of encrypted IV is XORed with the second fragment of plain text to obtain s bit ciphertext C2.

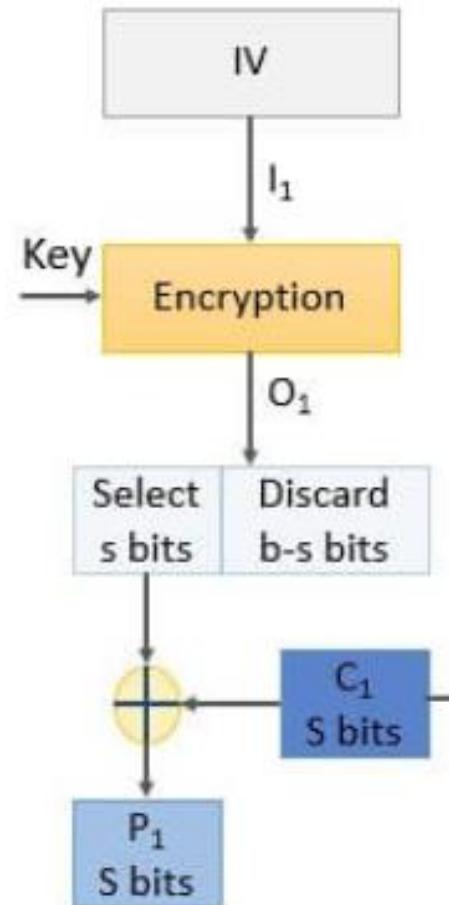
**Step 3:** Now the shift register containing initialization vector performs left shift by s bits and s bits  $C_1$  replaces the rightmost s bits of the initialization vector.

# Cipher Feedback Mode Mode (CFB)

## DECRYPTION PROCESS

### Step 1:

The initialization vector is placed in the shift register. It is encrypted using the same key. Then from the encrypted IV s bits are XORed with the s bits ciphertext C1 to retrieve s bits plain text P1.

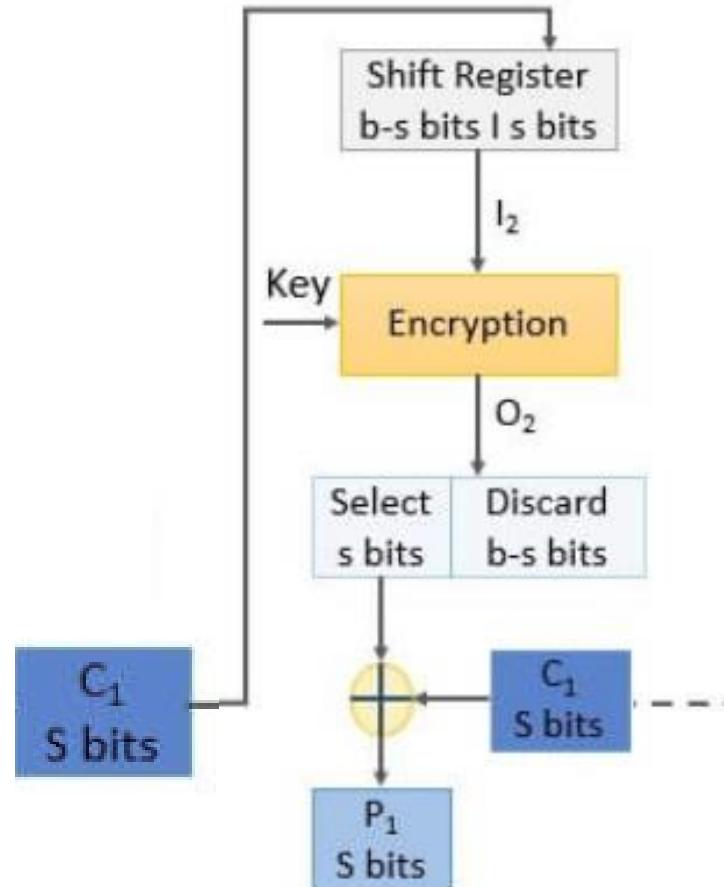


In the **decryption process**,  
the **encryption** algorithm is implemented instead of  
the decryption algorithm.

# Cipher Feedback Mode Mode (CFB)

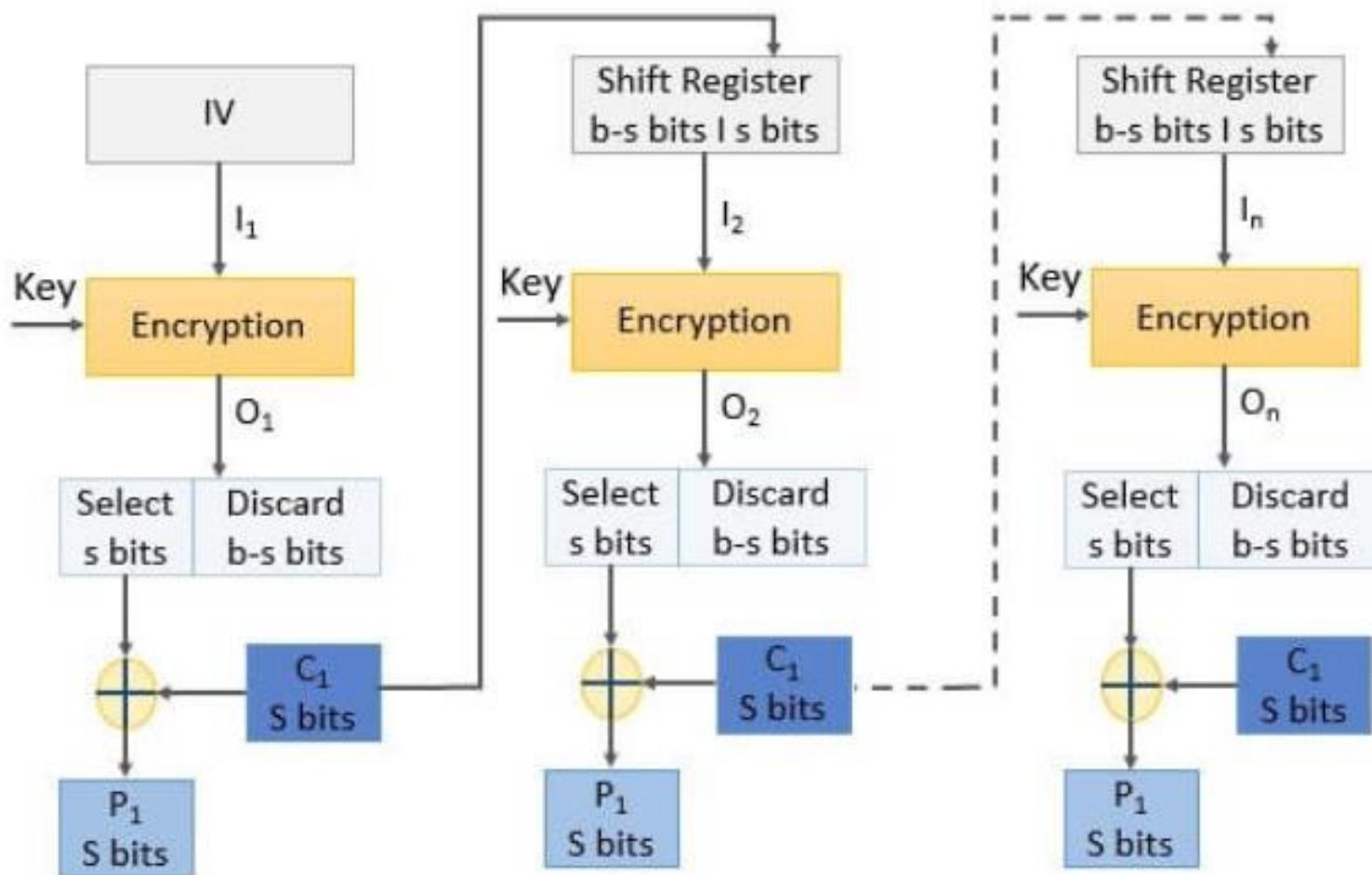
## DECRYPTION PROCESS

*Step 2:* The IV in the shift register is left-shifted by s bits and the s bits C1 replaces the rightmost s bits of IV.



# Cipher Feedback Mode Mode (CFB)

## DECRYPTION PROCESS



# Output Feedback Mode (OFB)

---

1. The output feedback (OFB) mode is almost similar to the CFB.
2. In OFB the encrypted IV is fed to the encryption of next plain text block.
3. The other difference is that CFB operates on a stream of bits whereas OFB operates on the block of bits.

## **The encryption steps in cipher feedback mode:**

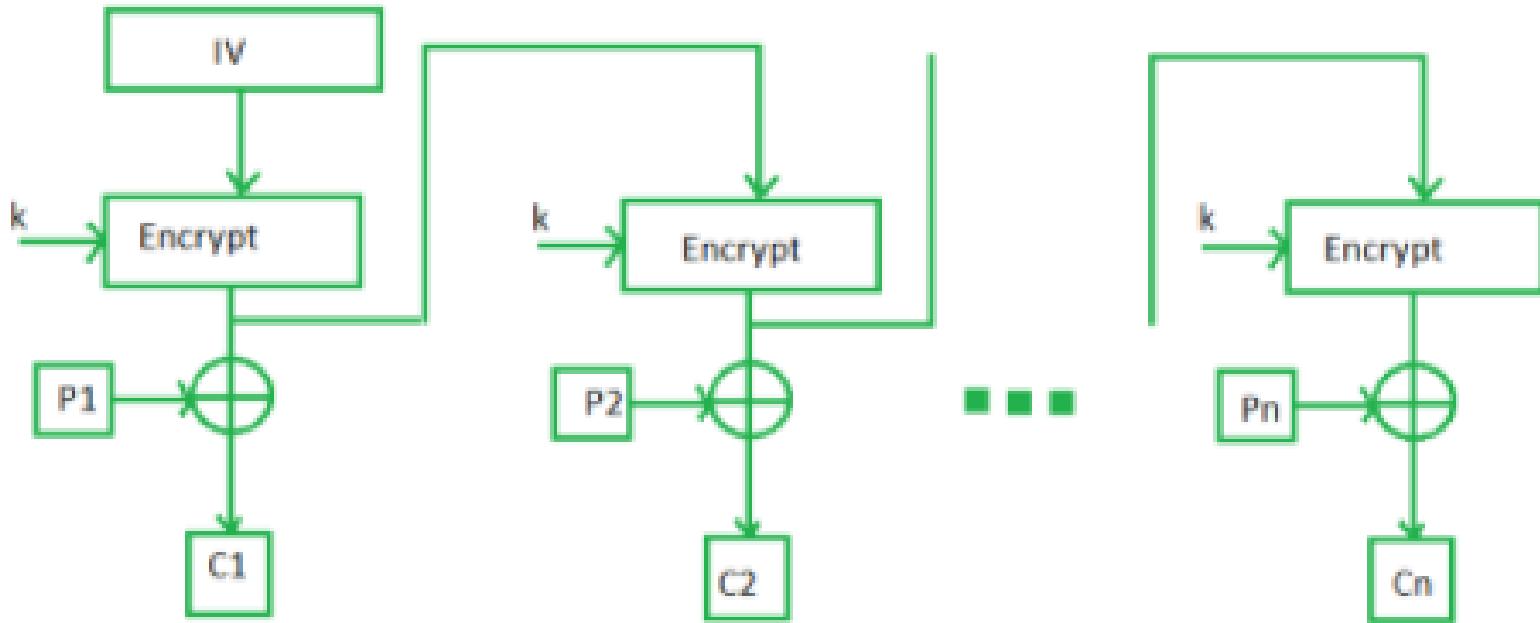
***Step 1:*** The initialization vector is encrypted using the key.

***Step 2:*** The encrypted IV is then XORed with the plain text block to obtain the ciphertext block. The encrypted IV is fed to the encryption of next plain text block



# Output Feedback Mode Mode (OFB)

## Encryption



ENCRYPTION



# Output Feedback Mode (OFB)

---

## DECRYPTION PROCESS

**Step 1:** The initialization vector is encrypted using the same key used for encrypting all plain text blocks.

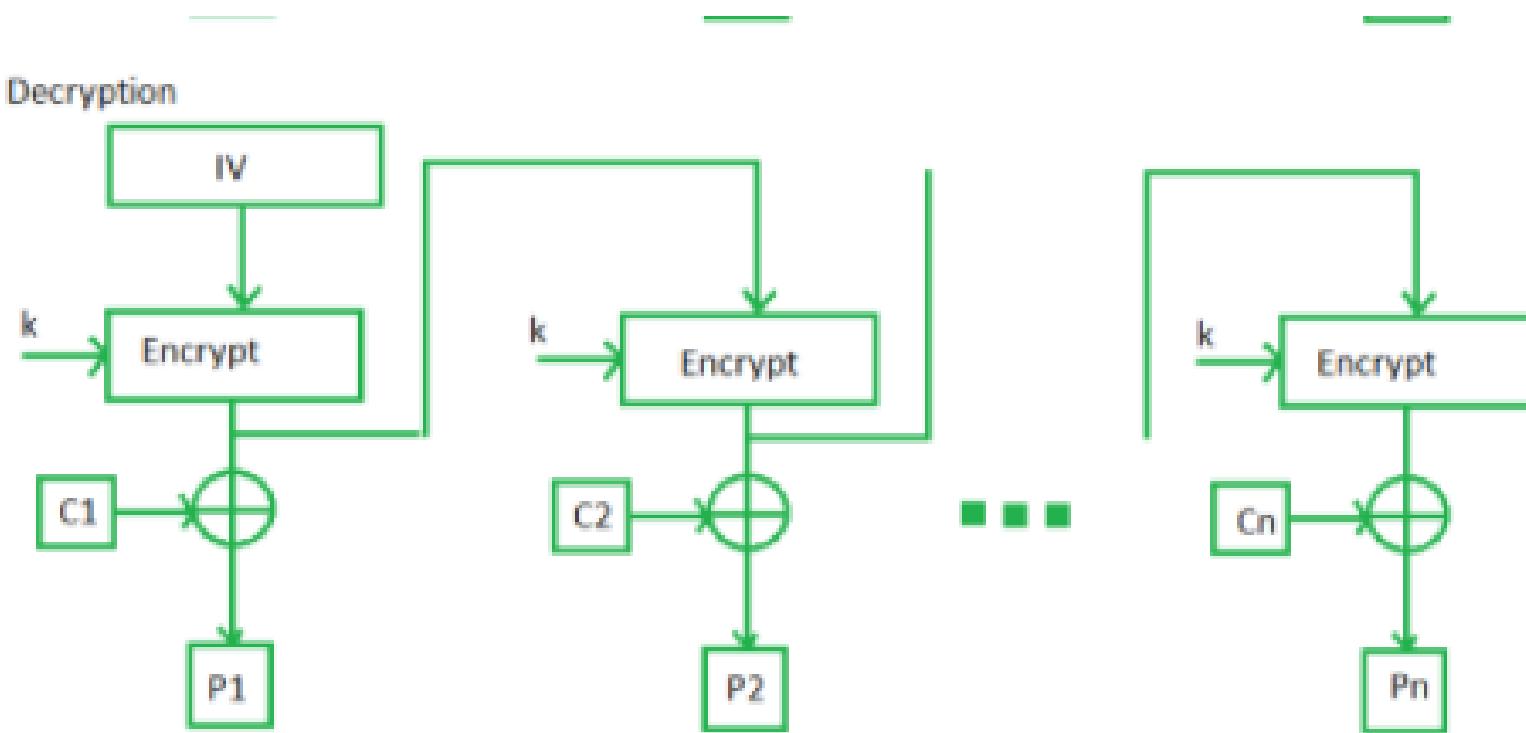
**Step2:** The encrypted IV is then XORed with the ciphertext block to retrieve the plain text block. The encrypted IV is also fed to the decryption process of the next ciphertext block.

The process continues until all the plain text blocks are retrieved.



# Output Feedback Mode (OFB)

## DECRYPTION PROCESS



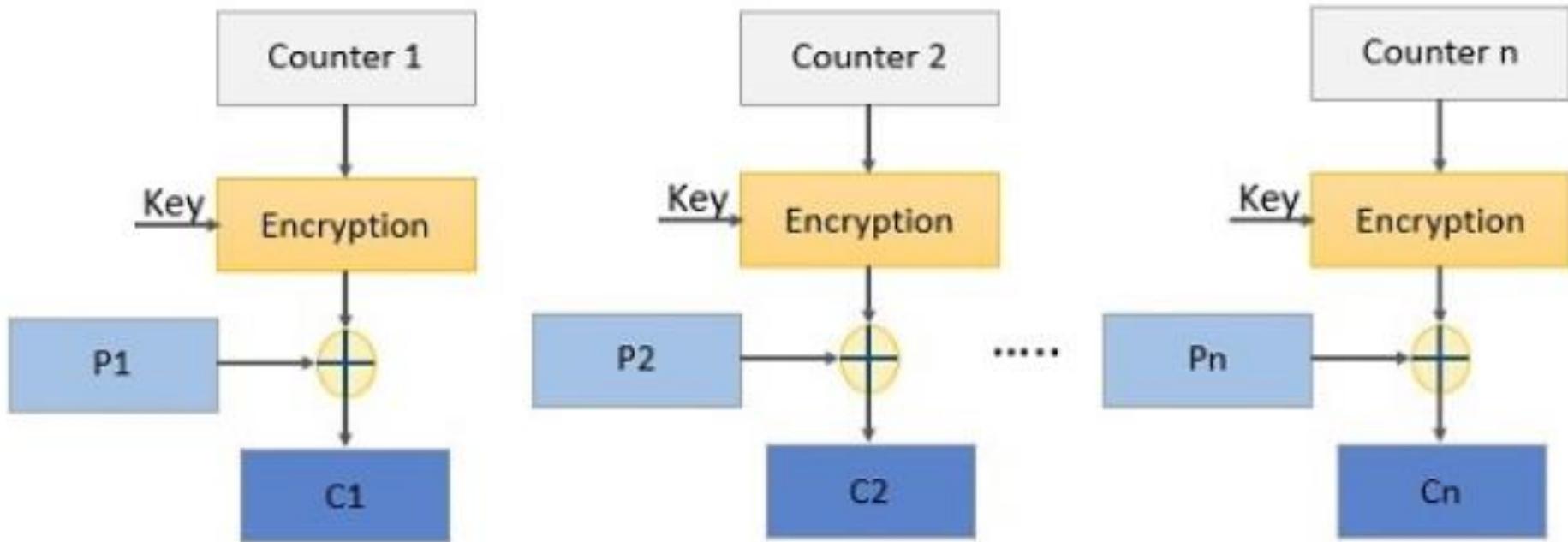
# Counter Mode

---

1. It is similar to OFB but there is no feedback mechanism in counter mode.
2. Nothing is being fed from the previous step to the next step instead it uses a sequence of number which is termed as a **counter** which is input to the encryption function along with the key.
3. After a plain text block is encrypted the counter value increments by 1.



# Counter Mode



**Step 1:** The counter value is encrypted using a key.

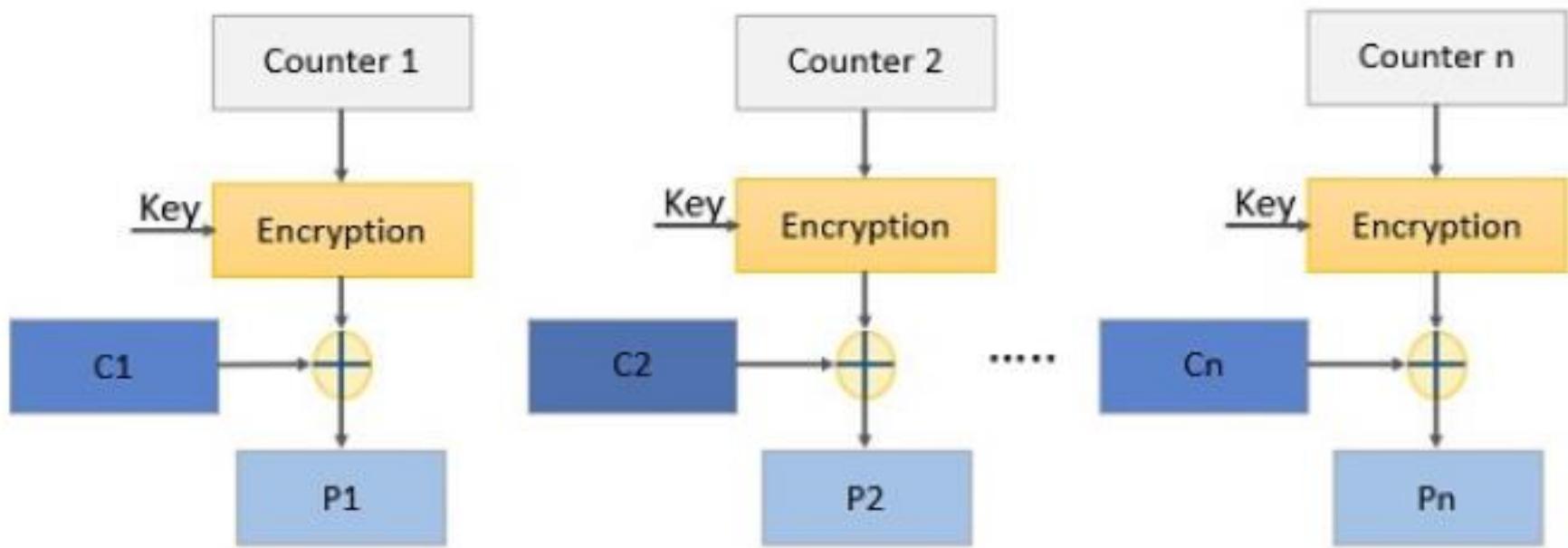
**Step 2:** The encrypted counter value is XORed with the plain text block to obtain a ciphertext block.

ENCRYPTION



# Counter Mode

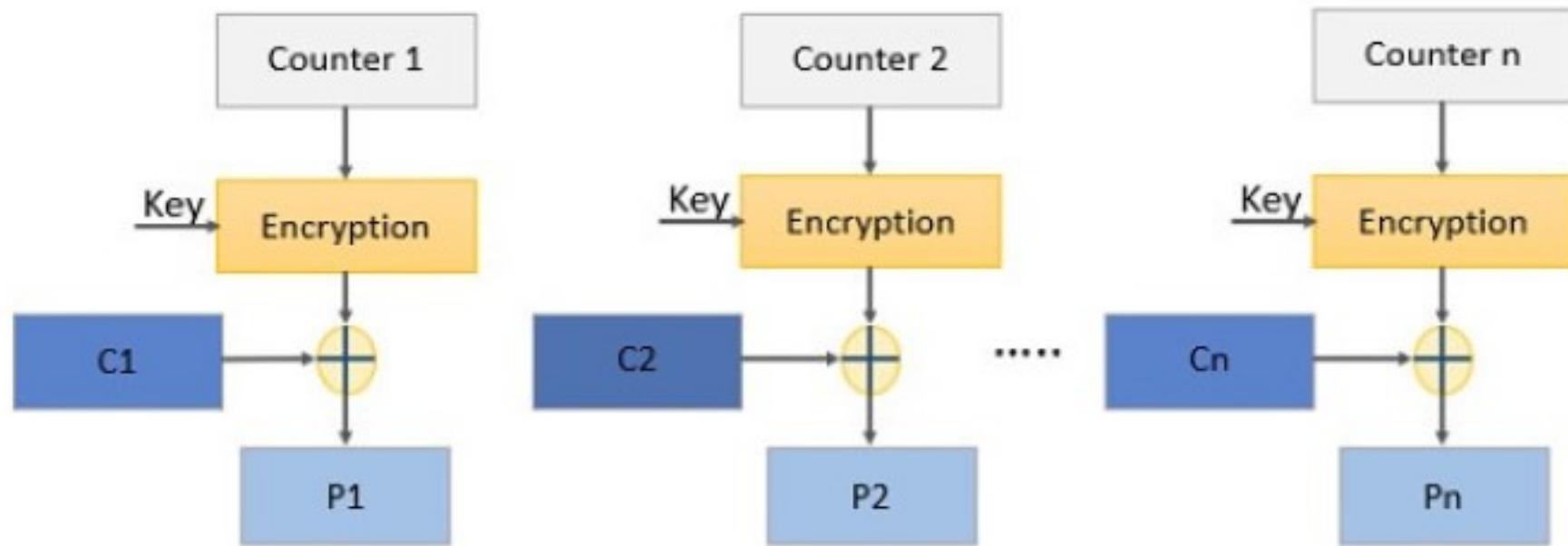
## DECRYPTION PROCESS



Encryption function is used in the decryption process

# Counter Mode

## DECRYPTION PROCESS



Encryption function is used in the decryption  
process



# Module 2: Symmetric & Asymmetric Key Cryptography and Key Management

---

## 2.1

- Block Cipher
  - ✓ Block Cipher Principles
  - ✓ Block Cipher Modes of Operation
  - ✓ Data Encryption Standard (DES)
  - ✓ Double DES, Triple DES
  - ✓ Advanced Encryption Standard (AES)
- Stream Cipher
  - ✓ RC5 algorithm



# Data Encryption Standard (DES)

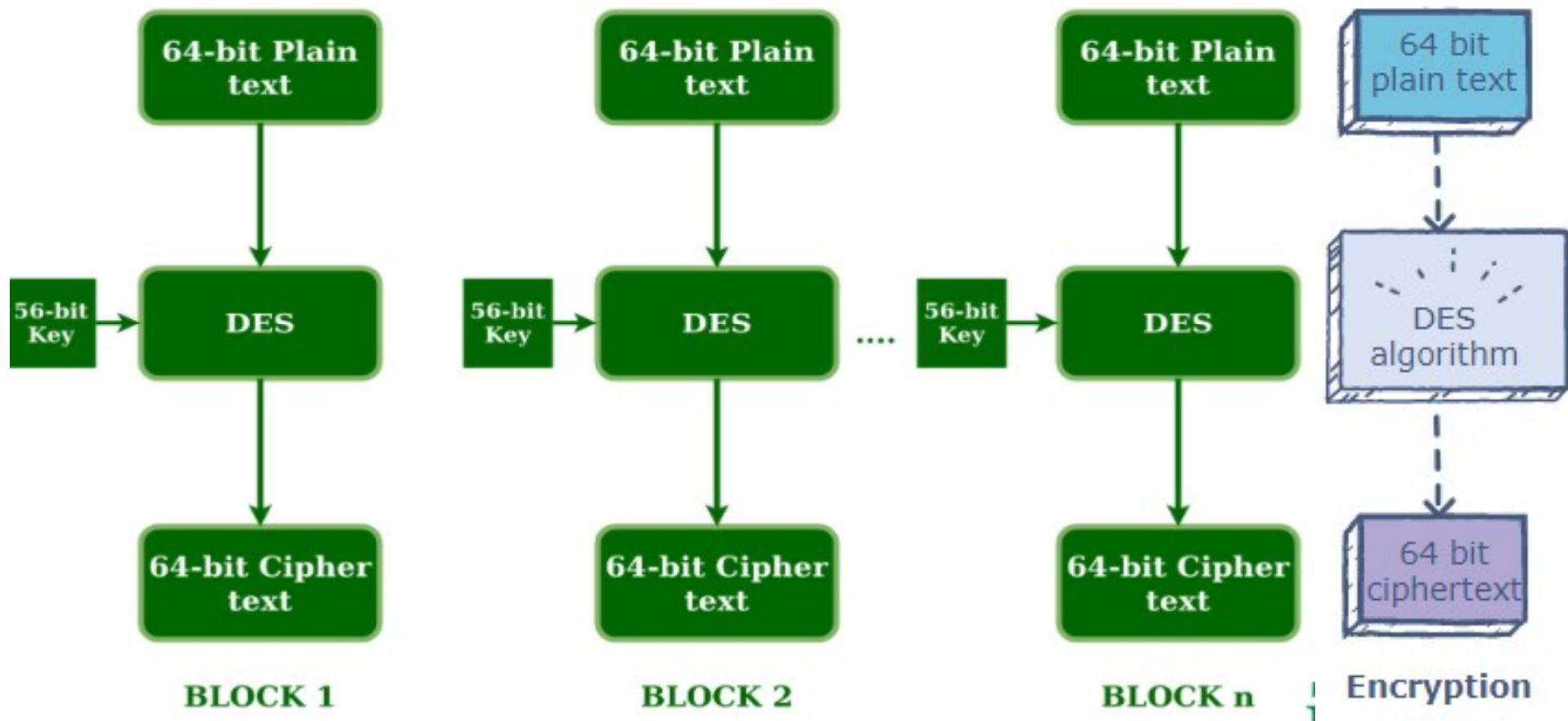
---

1. Is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST).
2. DES encrypts data in blocks of size of 64 bit each.
3. The same algorithm and key are used for encryption and decryption, with minor differences.
4. DES is an implementation of a Feistel Cipher. It uses 16 round Feistel structure.
5. 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).



# Data Encryption Standard (DES)

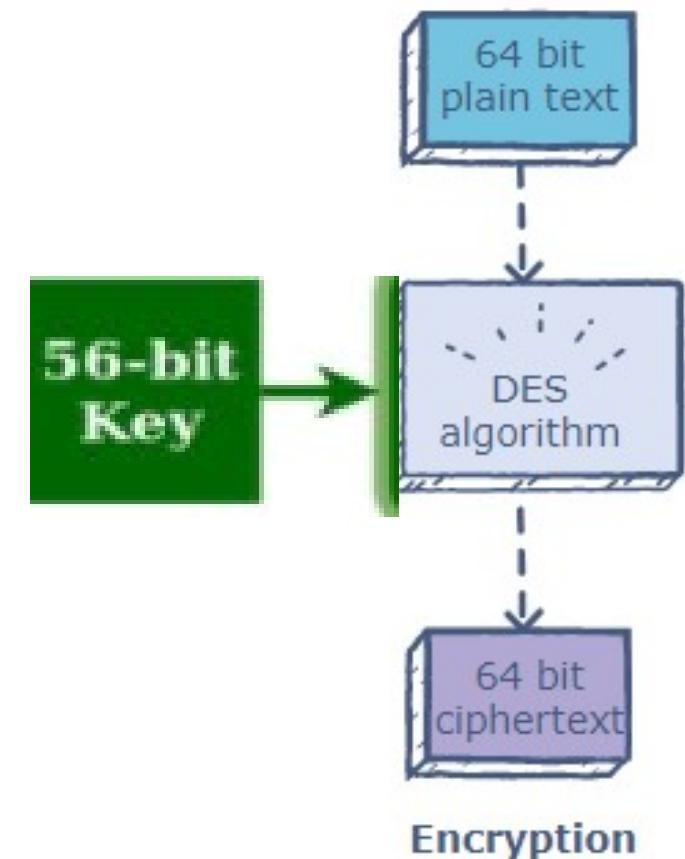
1. The basic idea of DES is shown below:



# Data Encryption Standard (DES)

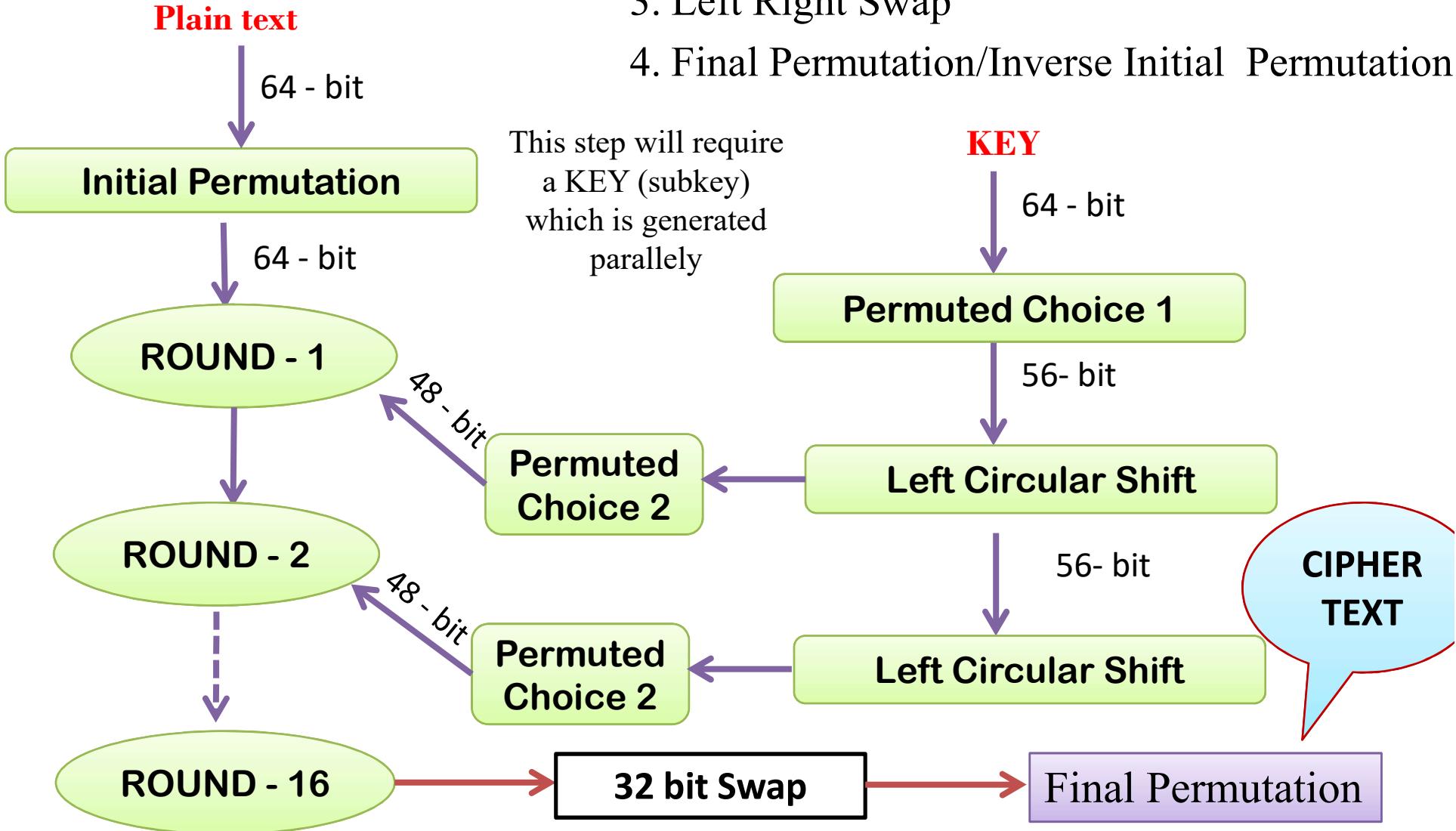
1. The basic idea of DES is shown below:

Even before the DES process starts, every 8th bit of the key is discarded to produce a 56 bit key. That is bit position 8, 16, 24, 32, 40, 48, 56 and 64 are discarded.



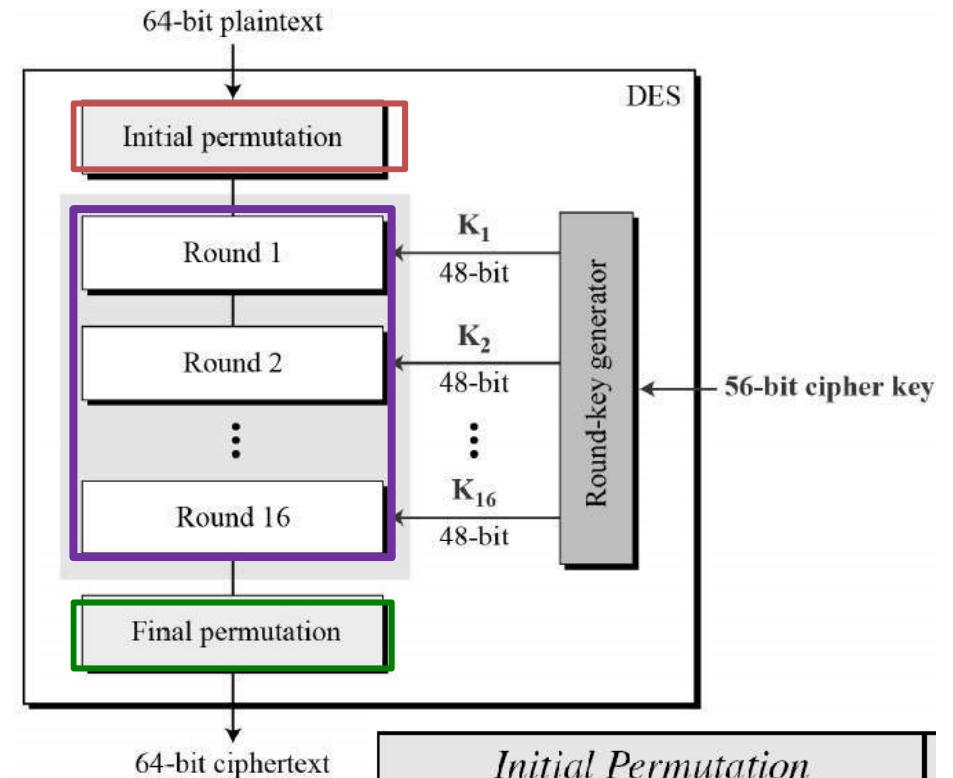
# Steps involved in DES

1. Initial Permutation
2. 16 Feistel Rounds
3. Left Right Swap
4. Final Permutation/Inverse Initial Permutation



# BLOCK DIAGRAM OF DES

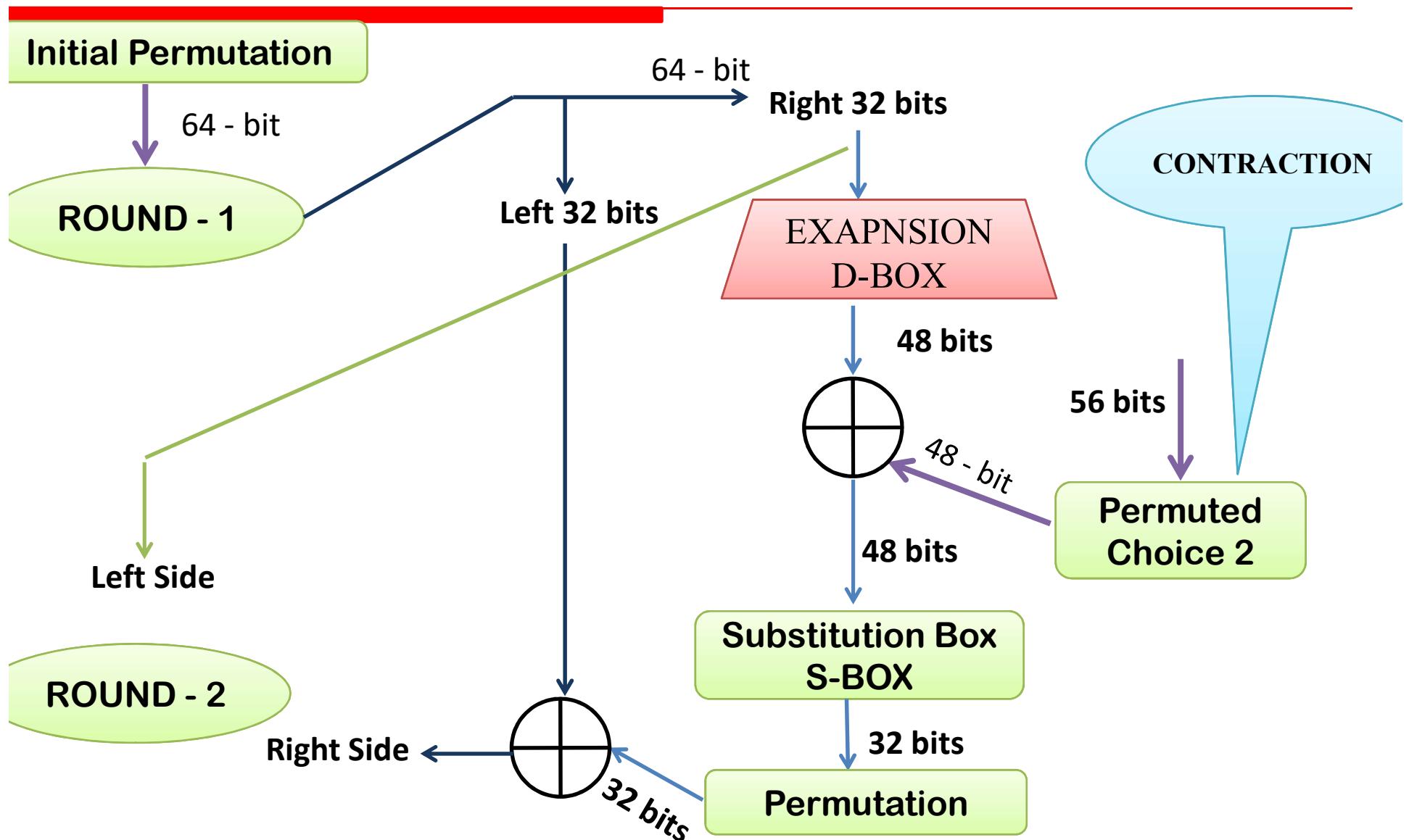
1. Permutations are referred as P-BOX.
2. Takes 64-bit input and permutes them according to a predefined rule.
3. These are Keyless Straight permutations that are inverse of each other.



Initial Permutation									
58	50	42	34	26	18	10	02		
60	52	44	36	28	20	12	04		
62	54	46	38	30	22	14	06		
64	56	48	40	32	24	16	08		
57	49	41	33	25	17	09	01		
59	51	43	35	27	19	11	03		
61	53	45	37	29	21	13	05		
63	55	47	39	31	23	15	07		

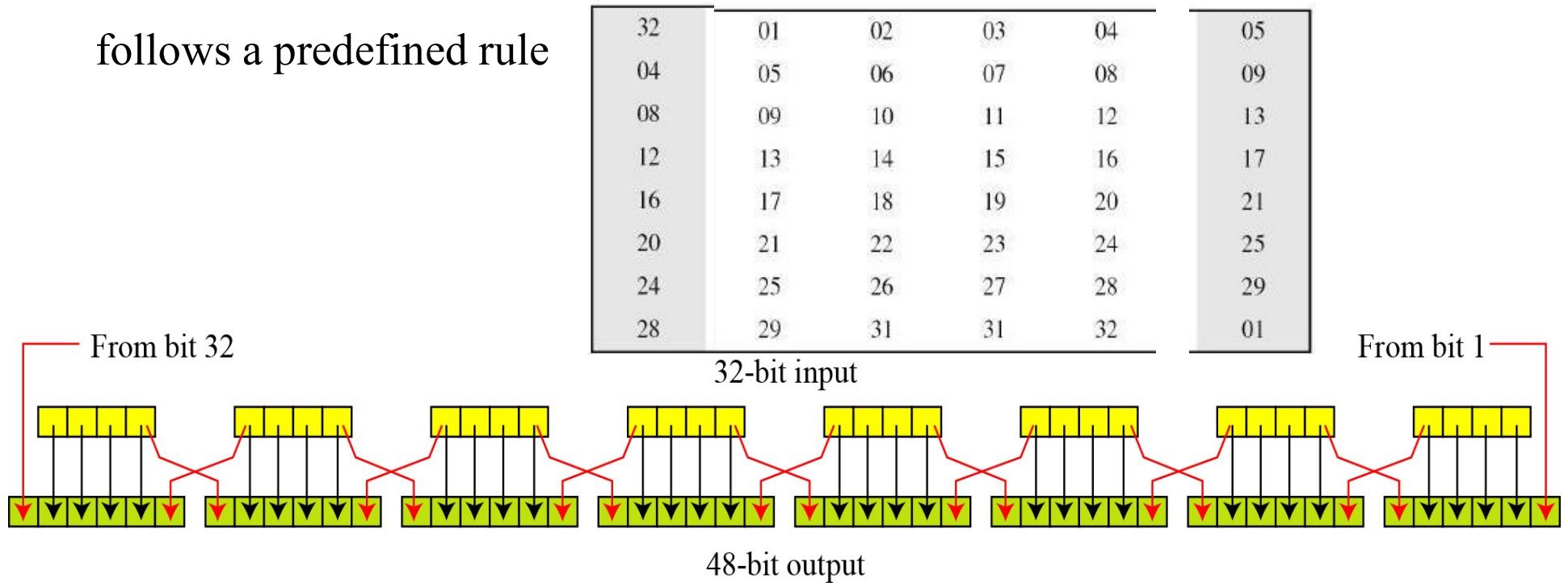


# ROUND FUNCTION of DES

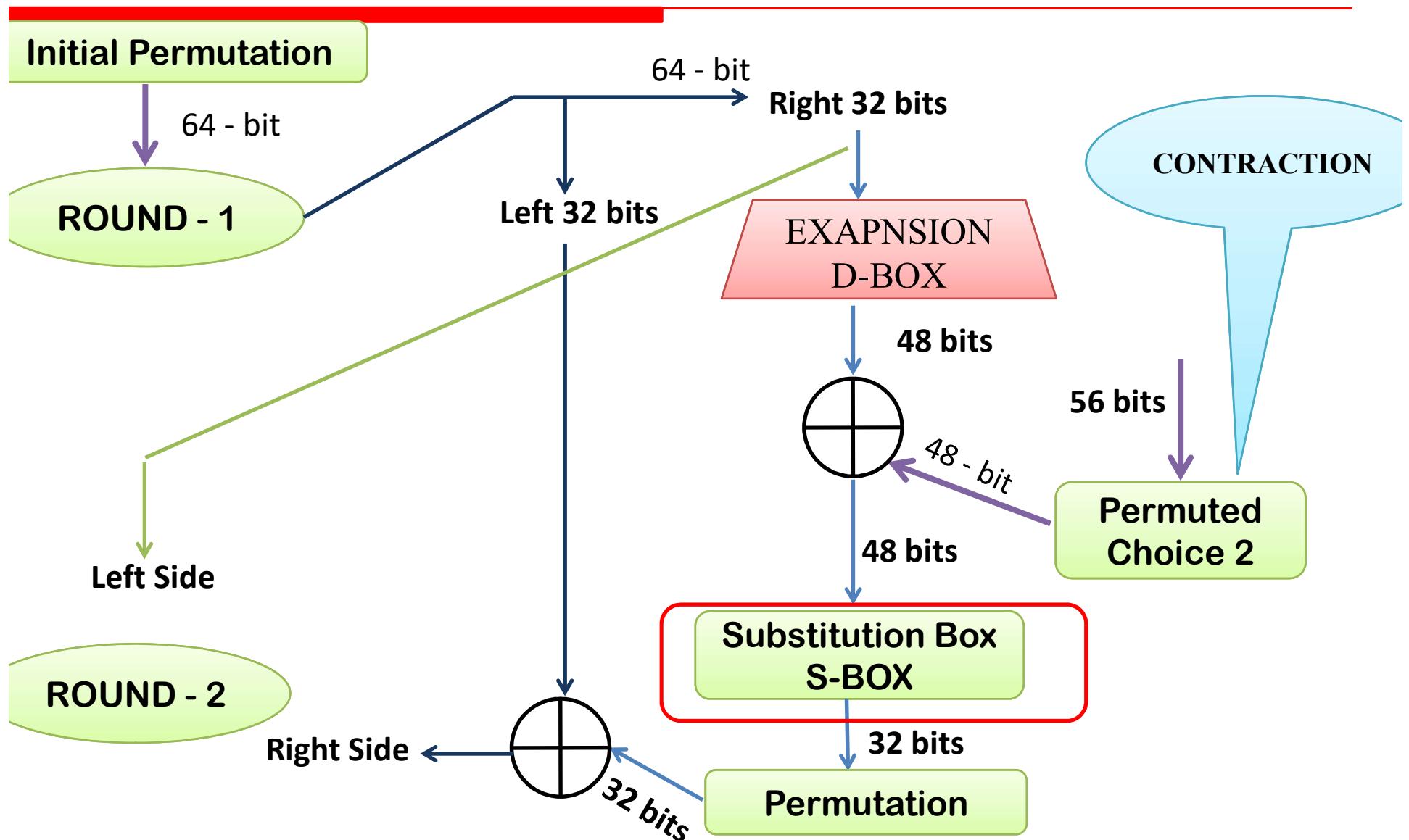


# Expansion D – BOX of Round

1. The RIGHT HALF of plaintext (64 bits) contains 32 bits, uses a key of 48-bits.
2. RIGHT HALF needs to be expanded to 48 bits.
3. The 32 – bits of RIGHT HALF into 8, 4-bit sections.
4. Each 4-bit section is then expanded to 6-bits. This expansion permutations follows a predefined rule

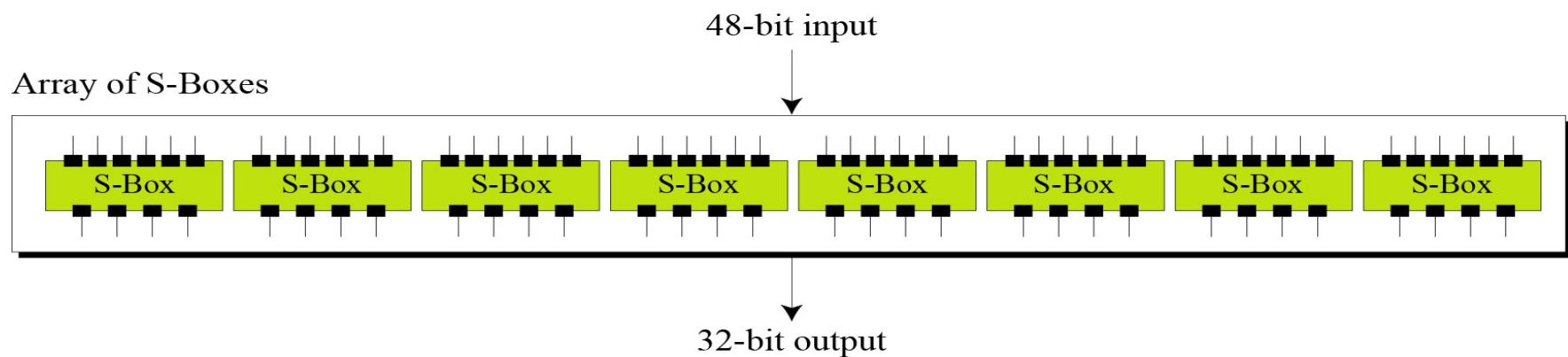
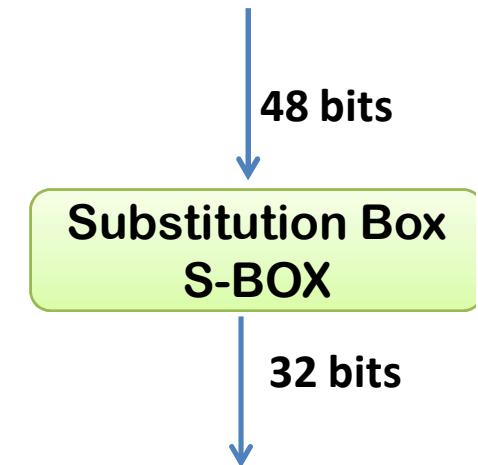


# ROUND FUNCTION of DES



# SUBSTITUTION BOX (S-BOX)

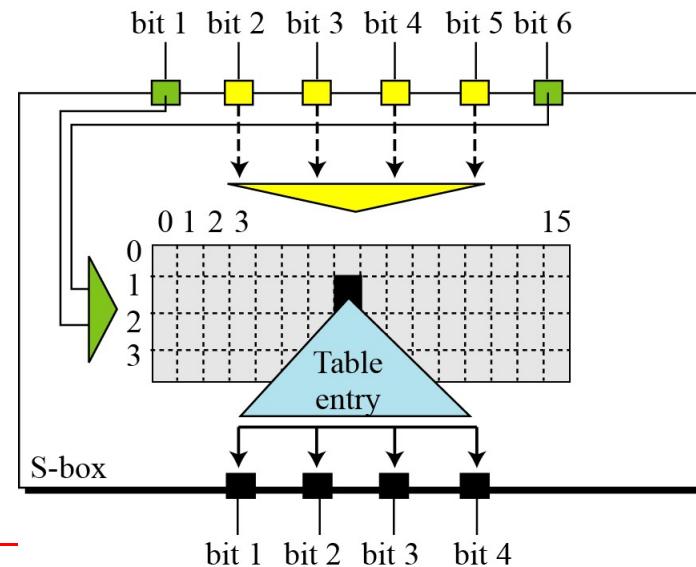
1. Performs the real mixing.
2. DES uses 8 S-boxes, each with a 6-bit input and a 4-bit output.
3. 48-bit data is divided into EIGHT 6-bit chunks, and each chunk is fed into a box
4. The result of each box is 4-bit, which when combined the 32-bit result is obtained.



# SUBSTITUTION BOX (S-BOX)

---

1. The substitution in each box follows a predetermined rule based on 4-ROW and 16-COLUMN table.
2. The combination of 1<sup>st</sup> and 6<sup>th</sup> bit of the input defines one of the 4 ROWS
3. The combination of bits 2<sup>nd</sup> through 5<sup>th</sup> of the input defines one of the 16 COLUMNS
4. Because each S-box has its own table, we will need 8 tables



# SUBSTITUTION BOX (S-BOX)

---

*The following table shows the permutation for S-box 1.*

***S-box 1***

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

*For the rest of the boxes see the textbook.*



The input to S-box 1 is 100011. What is the output?

***S-box 1***

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

1|00011|

11 – 03  
0001 - 01

Solution

The input to S-box 1 is **001100**. What is the output?

***S-box 1***

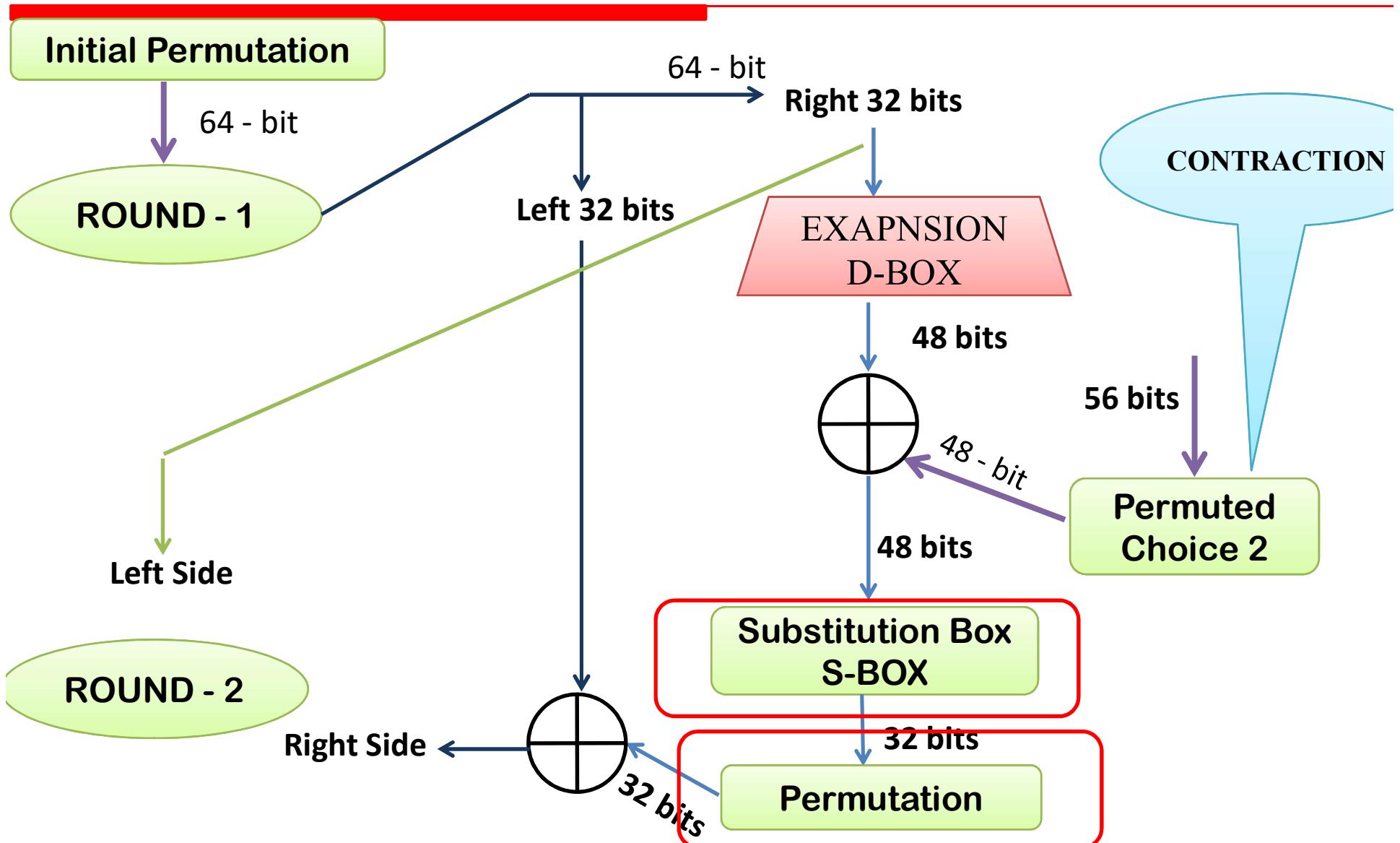
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	10	03	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

0|01100

00 – 00  
0110 - 06

**Solution**

# ROUND FUNCTION of DES



# **ROUND FUNCTION of DES**

---

16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

---

**Last Operation**



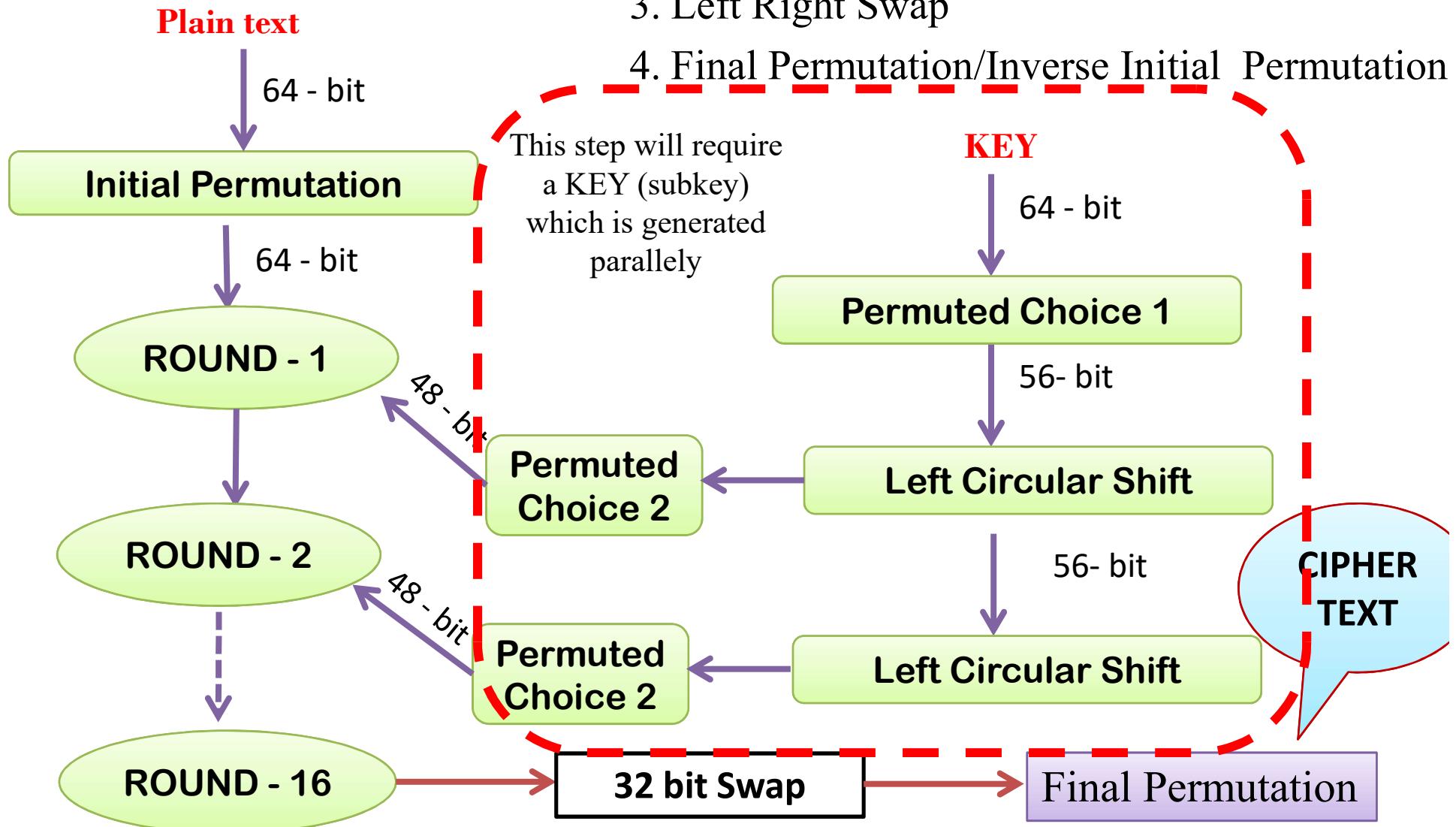
# Steps involved in DES

1. Initial Permutation

2. 16 Feistel Rounds

3. Left Right Swap

4. Final Permutation/Inverse Initial Permutation

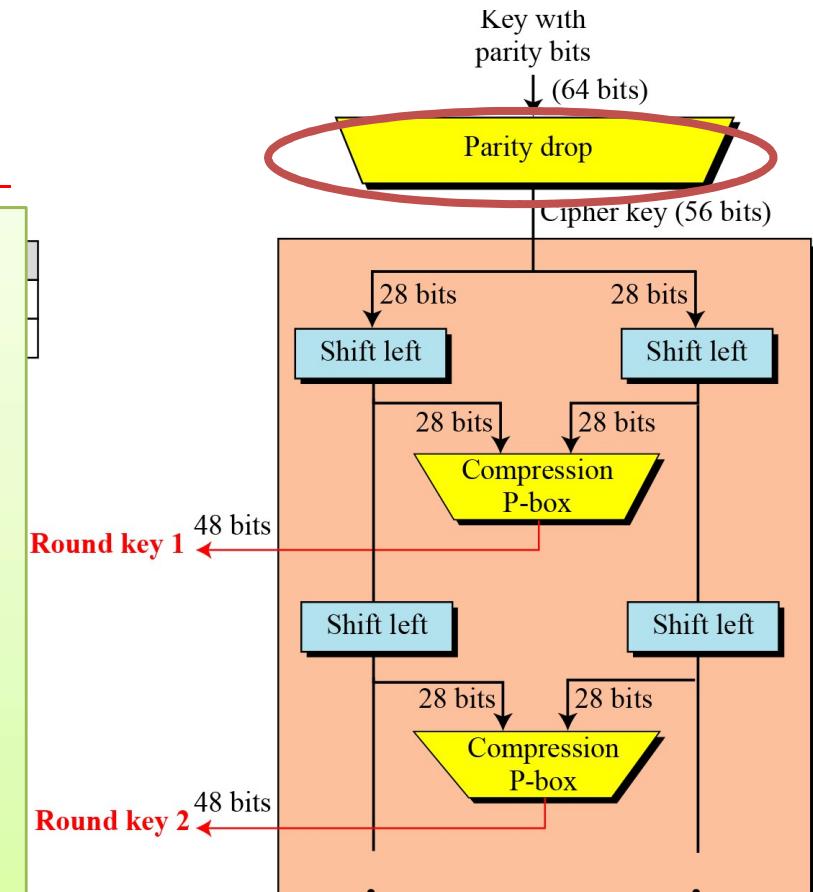


# KEY GENERATION

## PARITY DROP:

- The preprocess before key expansion is a compression transposition step
- It drop parity bits (8, 16, 24, 32, ... 64)
- Permutes the rest of the bits according to following Parity Drop Table.
- The remaining 56-bit value is the actual CIPHER KEY which is used to generate rounds.

57	49	41	33	25	17	09	01
58	50	42	34	26	18	10	02
59	51	43	35	27	19	11	03
60	52	44	36	63	55	47	39
31	23	15	07	62	54	46	38
30	22	14	06	61	53	45	37
29	21	13	05	28	20	12	04

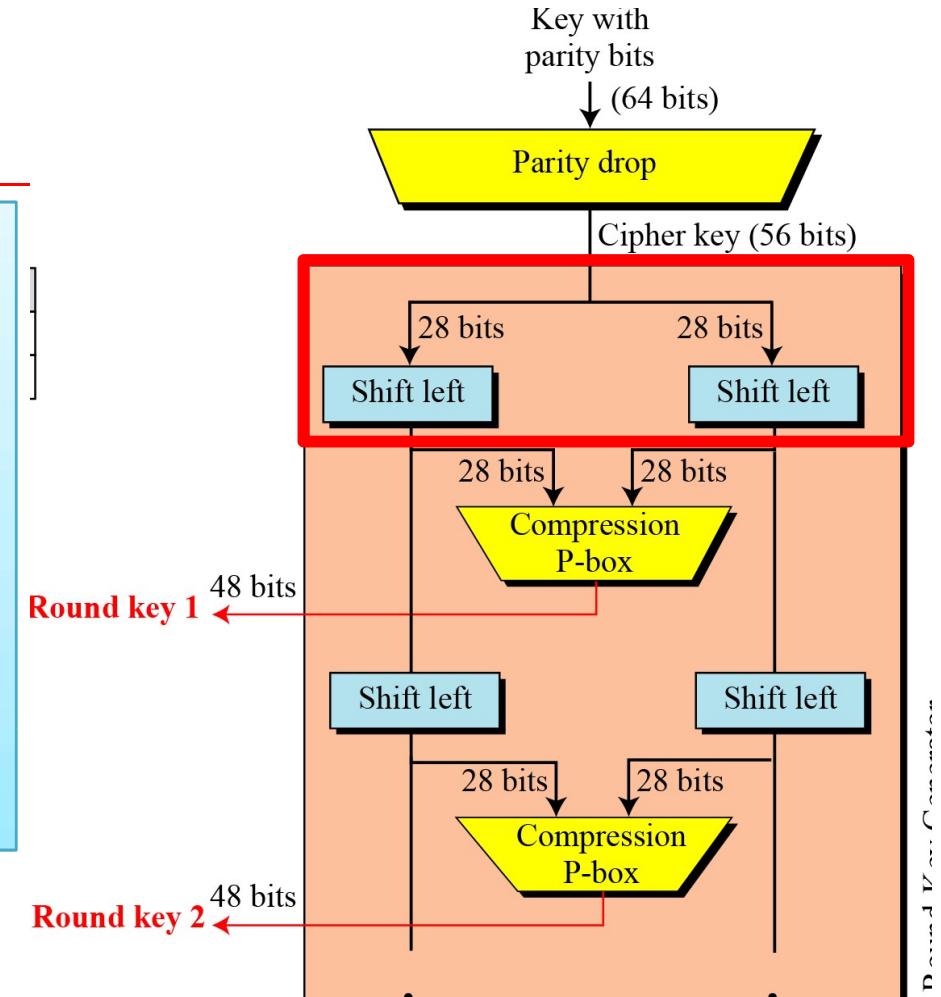


**Permuted  
Choice - 1**



# KEY GENERATION

- These 56-bits are again divided into Left and Right Half (28-bits each).
- Left Circular Shift is performed on each half
- In Round 1, 2, 9 and 16, shifting is 1 bit
- In the remaining Rounds it is 2-bits



**Table 6.13 Number of bits shifts**

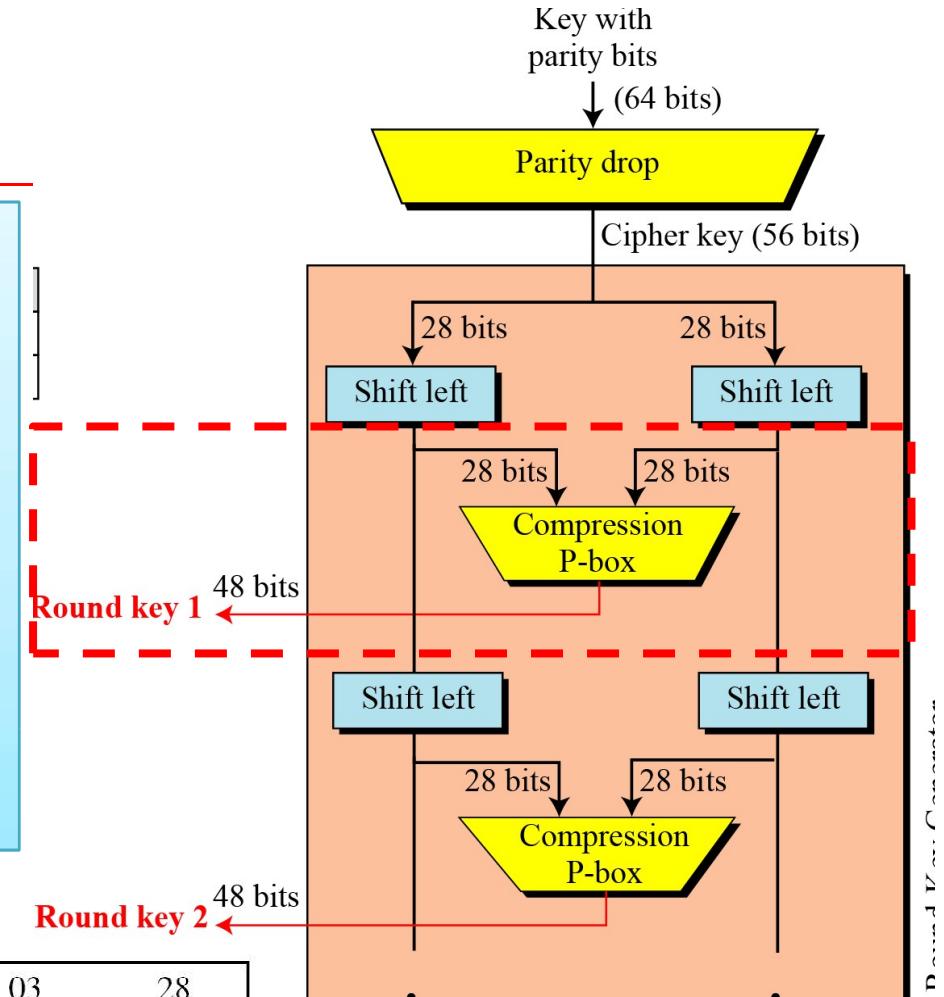
Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bit shifts	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1



# KEY GENERATION

## COMPRESSION D-Box

The Compression D-box changes the 56-bits to 48-bits which are used as a key for a round.



14	17	11	24	01	05	03	28
15	06	21	10	23	19	12	04
26	08	16	07	27	20	13	02
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32



# DES ANALYSIS

---

1. *Critics have used a strong magnifier to analyze DES.*
2. *Tests have been done to measure the strength of some desired properties in a block cipher.*

**1. Properties:** Two desired properties of a block cipher are the

**AVALANCHE EFFECT and THE COMPLETENESS**

2. **Design Criteria:** The design of DES was revealed by IBM in 1994. Many test on DES have proved that it satisfies some of the required criteria as claimed.
3. **DES Weaknesses:** Weakness in Cipher Design and in Cipher Key



## ***PROPERTIES***

---

***Avalanche effect*** means a small change in plaintext(or key) should create a significant change in cipher text

### **Example**

To check the avalanche effect in DES, let us encrypt two plaintext blocks (with the same key) that differ only in one bit and observe the differences in the number of bits in each round.

Plaintext: 0000000000000000

Key: 22234512987ABB23

Ciphertext: 4789FD476E82A5F1

Plaintext: 0000000000000001

Key: 22234512987ABB23

Ciphertext: 0A4ED5C15A63FEA3

### ***Completeness effect***

*Completeness effect means that each bit of the ciphertext needs to depend on many bits on the plaintext*



## **MULTIPLE DES**

---

1. Because of its vulnerability to brute-force attack, DES, once the most widely used symmetric cipher, has been largely replaced by stronger encryption schemes.
2. Two approaches have been taken.
3. One approach is to design a completely new algorithm that is resistant to both cryptanalytic and brute-force attacks, of which AES is a prime example.
4. Another alternative, which preserves the existing investment in software and equipment, is to use multiple encryption with DES and multiple keys.

- 1. Double DES (2DES)**
- 2. Triple DES (3DES)**



## ***DOUBLE DES (2DES)***

---

1. The simplest form of multiple encryption has two encryption stages and two keys
2. Given a plaintext P and two encryption keys K<sub>1</sub> and K<sub>2</sub>, ciphertext C is generated as

$$P \longrightarrow E(K_1, P) \quad C = E(K_2, E(K_1, P))$$

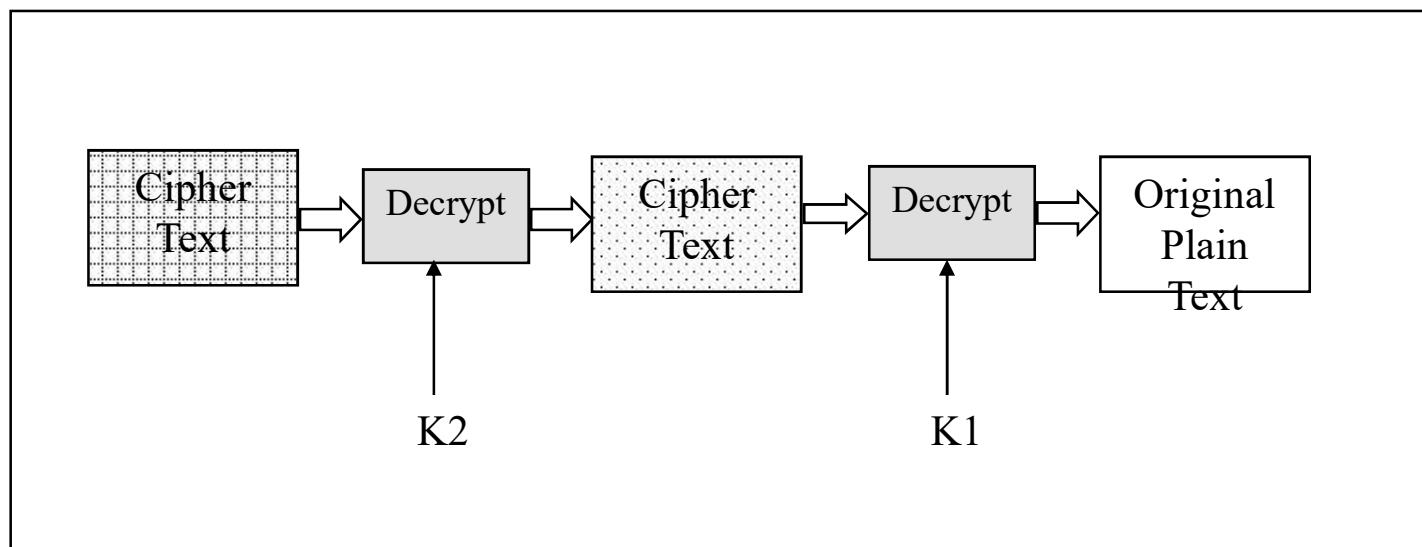
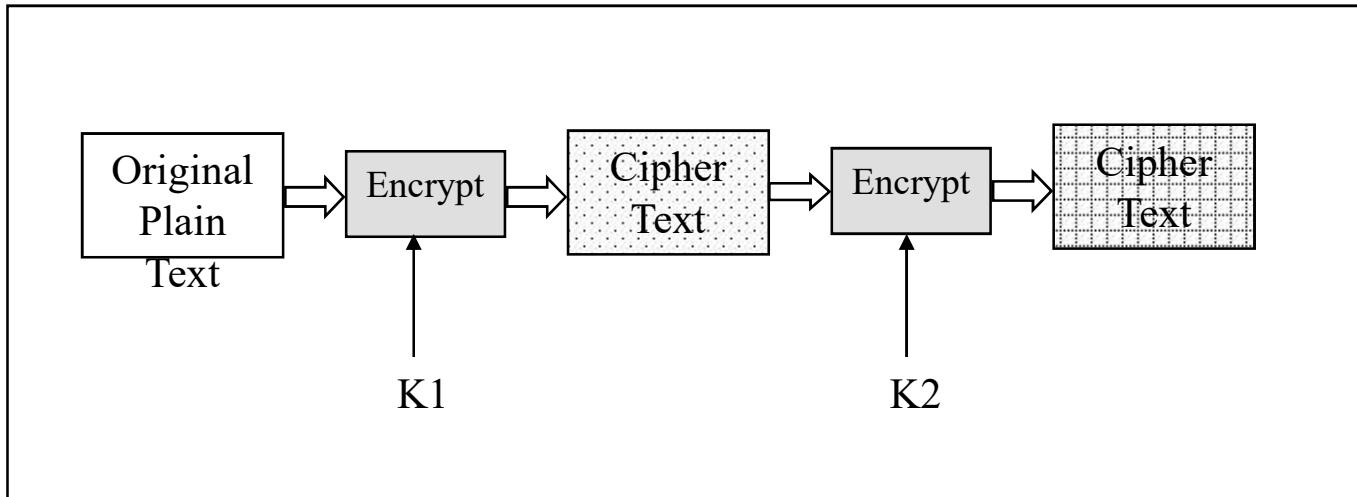
Decryption requires that the keys be applied in reverse order:

$$\text{Plaintext} = D(K_1, D(K_2, C))$$

For DES, this scheme apparently involves a key length of **56 \* 2 = 112 bits**



## ***DOUBLE DES (2DES)***



## ***TRIPLE DES (3DES)***

---

1. Double DES was prone to MEET-IN-THE-MIDDLE-ATTACK
2. An obvious counter to the meet-in-the-middle attack is to use three stages of encryption with three different keys.
3. Using DES as the underlying algorithm, this approach is commonly referred to as 3DES, or Triple Data Encryption Standard

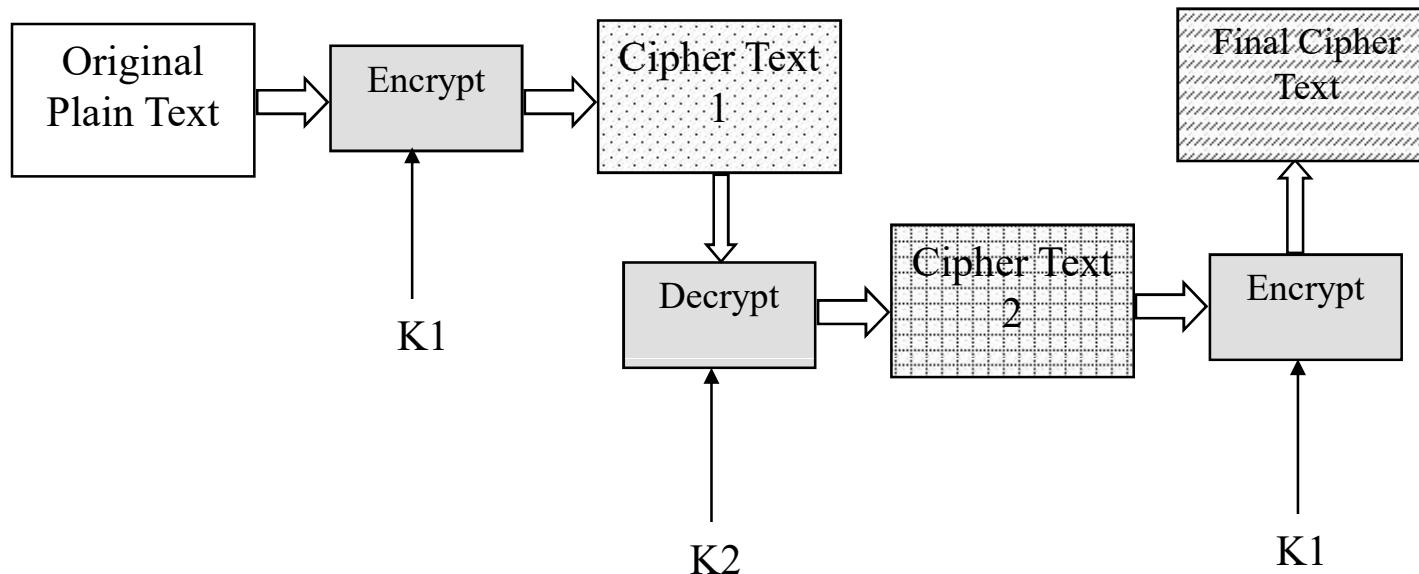
**Triple DES with Three Different Keys**

**Triple DES with Two Different Keys**

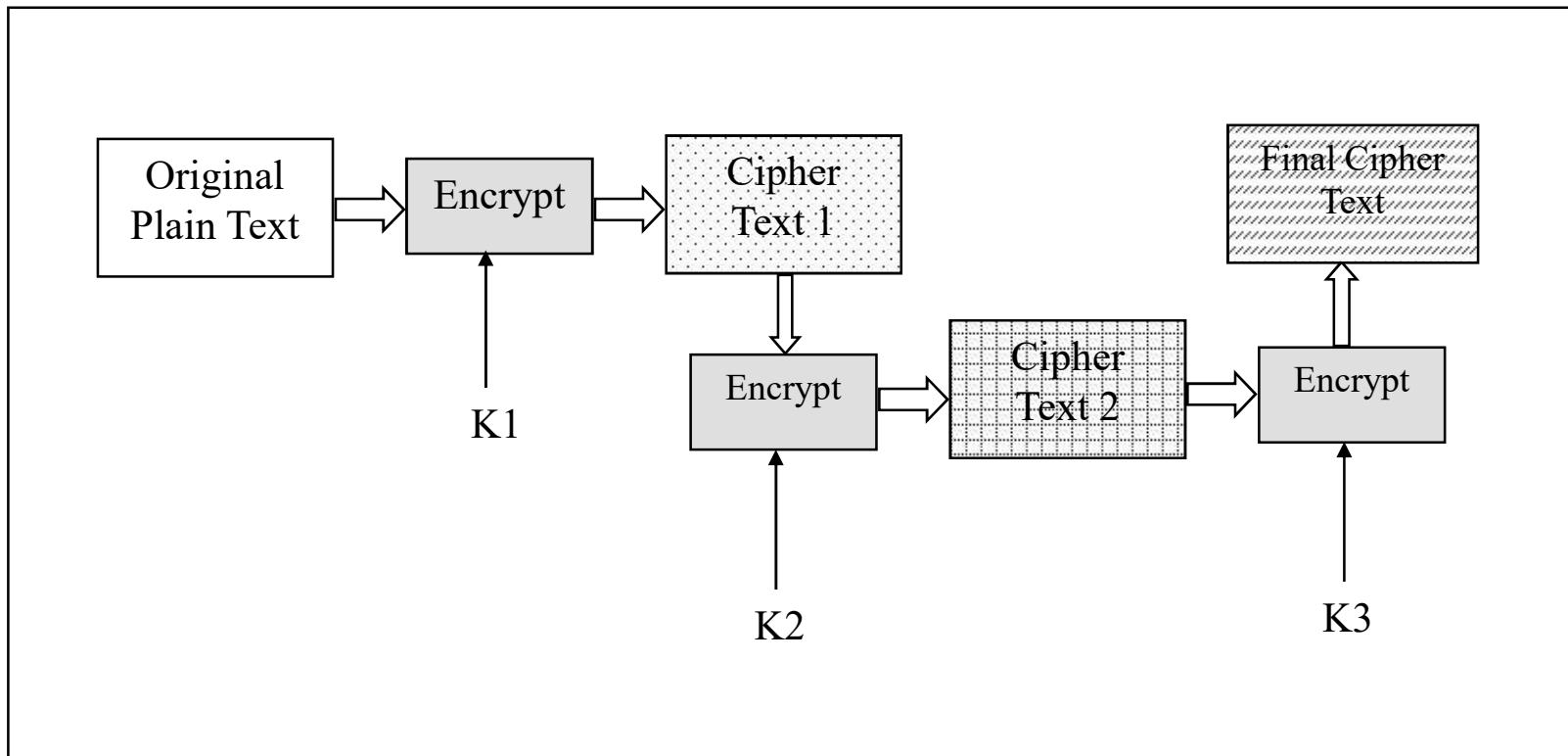


# Triple DES with Two Different Keys

---



# Triple DES with Three Different Keys



# Module 2: Symmetric & Asymmetric Key Cryptography and Key Management

---

## 2.1

- Block Cipher
  - ✓ Block Cipher Principles
  - ✓ Block Cipher Modes of Operation
  - ✓ Data Encryption Standard (DES)
  - ✓ Double DES, Triple DES
  - ✓ Advanced Encryption Standard (AES)
- Stream Cipher
  - ✓ RC5 algorithm



# Advanced Encryption Standard (AES)

1. Is a symmetric-key block cipher published by the National Institute of Standards and Technology (NIST). Proposed by **Rijndael**
2. AES encrypts data in blocks of size of 128 bit each.
3. AES is a NON - Feistel Cipher. It uses 10 rounds
4. The key size is 128 bits. Here, the key is processed in terms of **WORDS**
5. 128 bits are processed in 4 words or 16 bytes

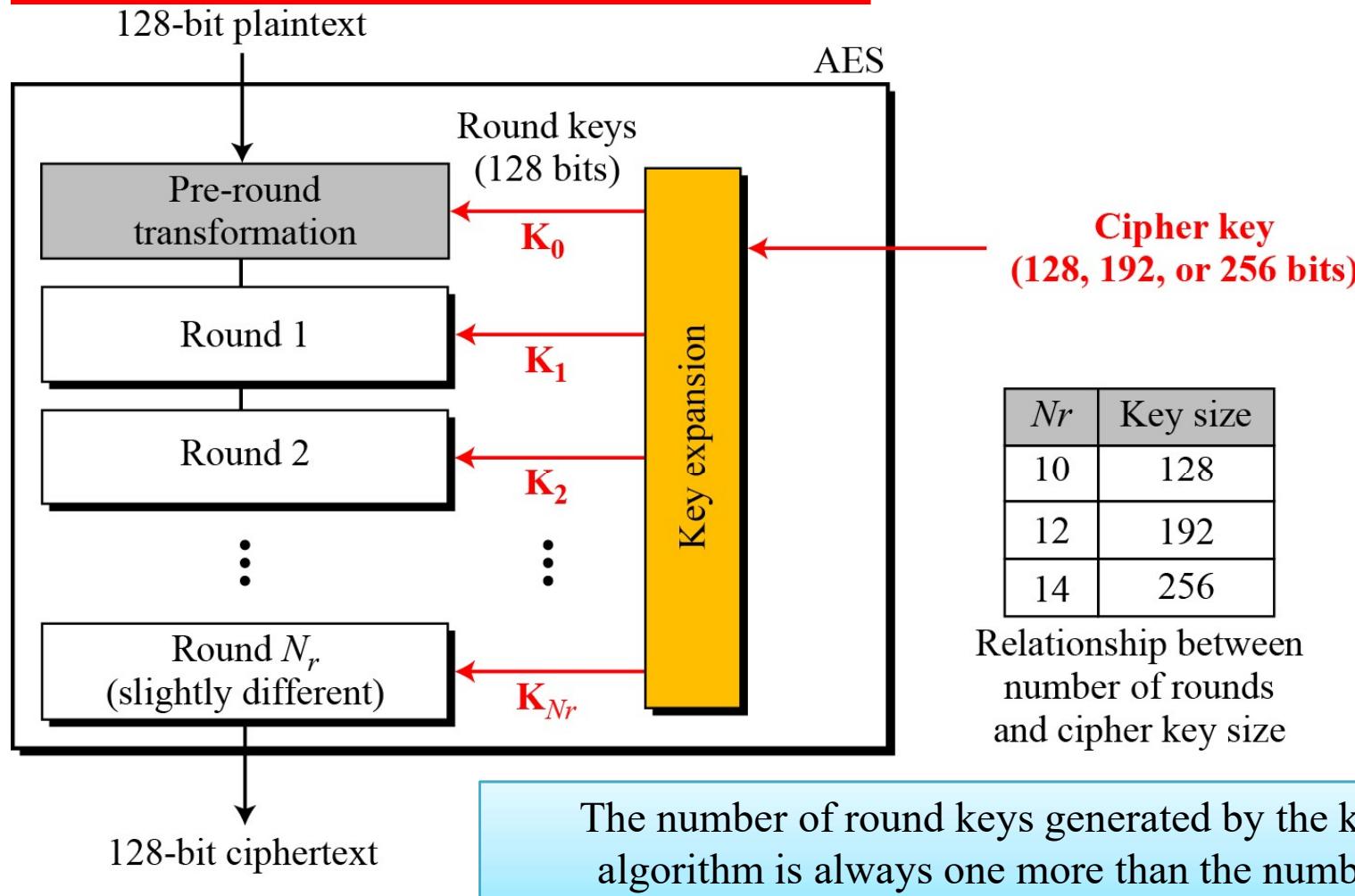
**1 WORD = 32 bits  
1 byte = 8 bits**

## Note

AES has defined three versions, with 10, 12, and 14 rounds. Each version uses a different cipher key size (128, 192, or 256), BUT the round keys are always 128 bits.



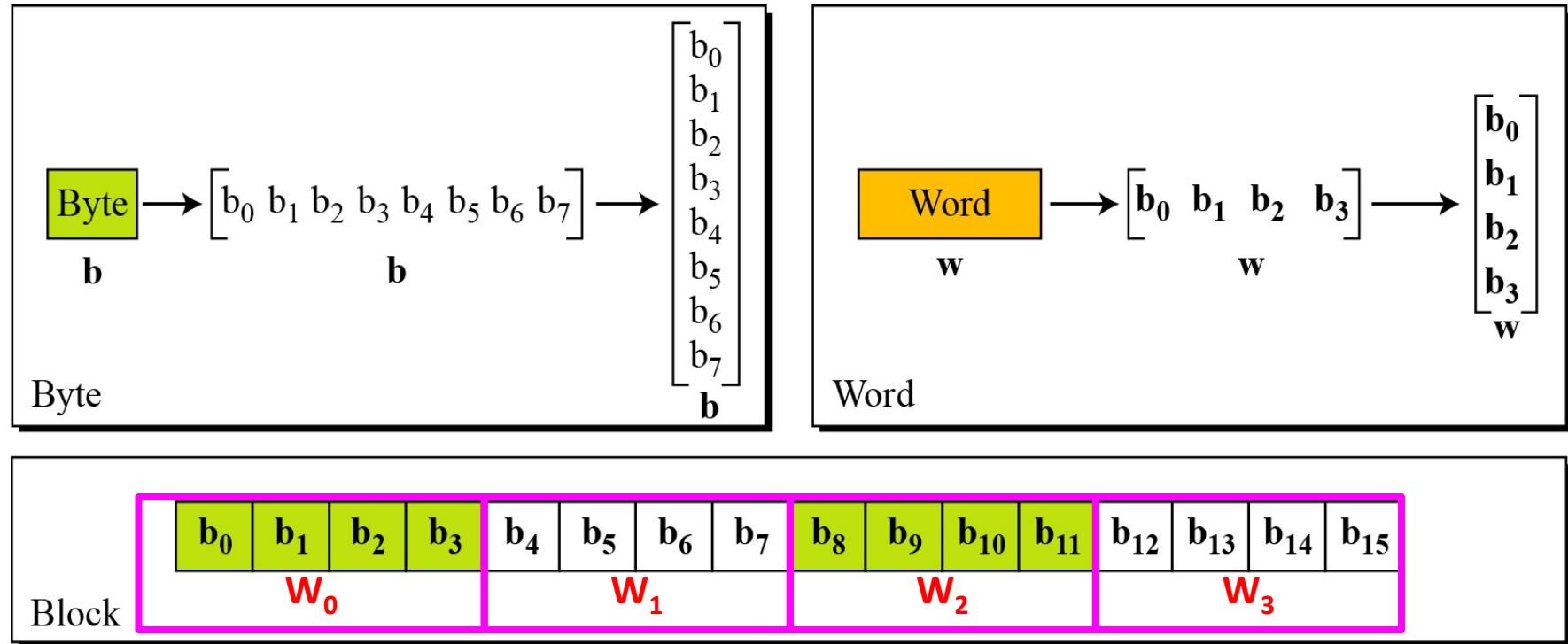
# Advanced Encryption Standard (AES)



$$\text{Number of Round Keys} = N_r + 1$$



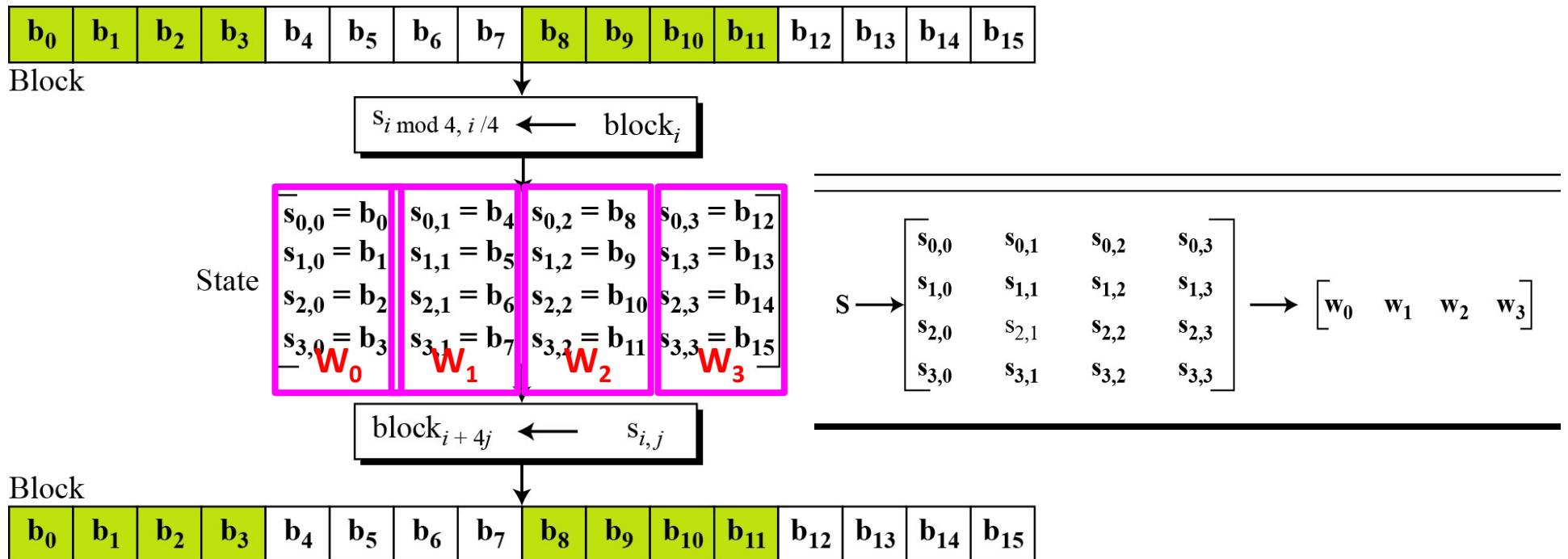
# AES data units



# AES data units

## State

1. Data Block is referred to as a STATE.
2. States are made of 16 bytes, but normally are treated as matrices of 4 \* 4 bytes
3. Each element is referred to as  $s_{r,c}$  where  $r(0$  to  $3)$  defines the row and  $c(0$  to  $3)$  defines the column



# Example *Changing plaintext to state*

Text A E S U S E S A M A T R I X Z Z

Hexadecimal 00 04 12 14 12 04 12 00 0C 00 13 11 08 23 19 19  
 $w_0$

00	12	0C	08
04	04	00	23
12	12	13	19
14	00	11	19

	DEC	HEX		DEC	HEX
A	00	00	N	13	0D
B	01	01	O	14	0E
C	02	02	P	15	0F
D	03	03	Q	16	10
E	04	04	R	17	11
F	05	05	S	18	12
G	06	06	T	19	13
H	07	07	U	20	14
I	08	08	V	21	15
J	09	09	W	22	16
K	10	0A	X	23	17
L	11	0B	Y	24	18
M	12	0C	Z	25	19

b<sub>0</sub> b<sub>1</sub> b<sub>2</sub> b<sub>3</sub> b<sub>4</sub> b<sub>5</sub> b<sub>6</sub> b<sub>7</sub> b<sub>8</sub> b<sub>9</sub> b<sub>10</sub> b<sub>11</sub> b<sub>12</sub> b<sub>13</sub> b<sub>14</sub> b<sub>15</sub>  
 $w_0$   $w_1$   $w_2$   $w_3$



# Advanced Encryption Standard (AES)

**Plaintext** (128 – bits)

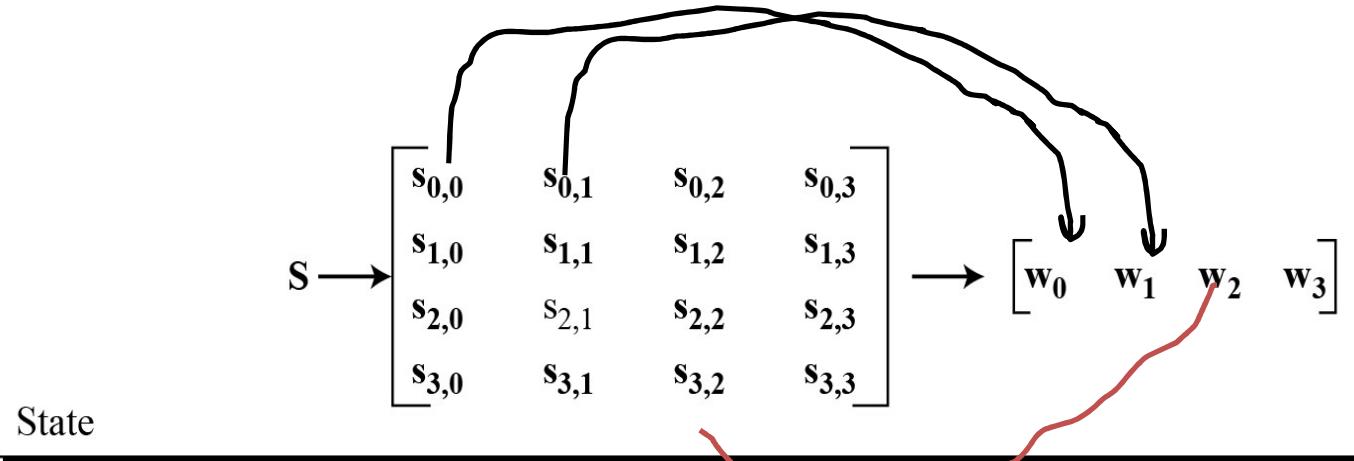
IN <sub>0</sub>	IN <sub>4</sub>	IN <sub>8</sub>	IN <sub>12</sub>
IN <sub>1</sub>	IN <sub>5</sub>	IN <sub>9</sub>	IN <sub>13</sub>
IN <sub>2</sub>	IN <sub>6</sub>	IN <sub>10</sub>	IN <sub>14</sub>
IN <sub>3</sub>	IN <sub>7</sub>	IN <sub>11</sub>	IN <sub>15</sub>

Is represented  
as Input Array  
( 4 \* 4 ) each  
cell has 8 bits  
data i.e. 1 byte

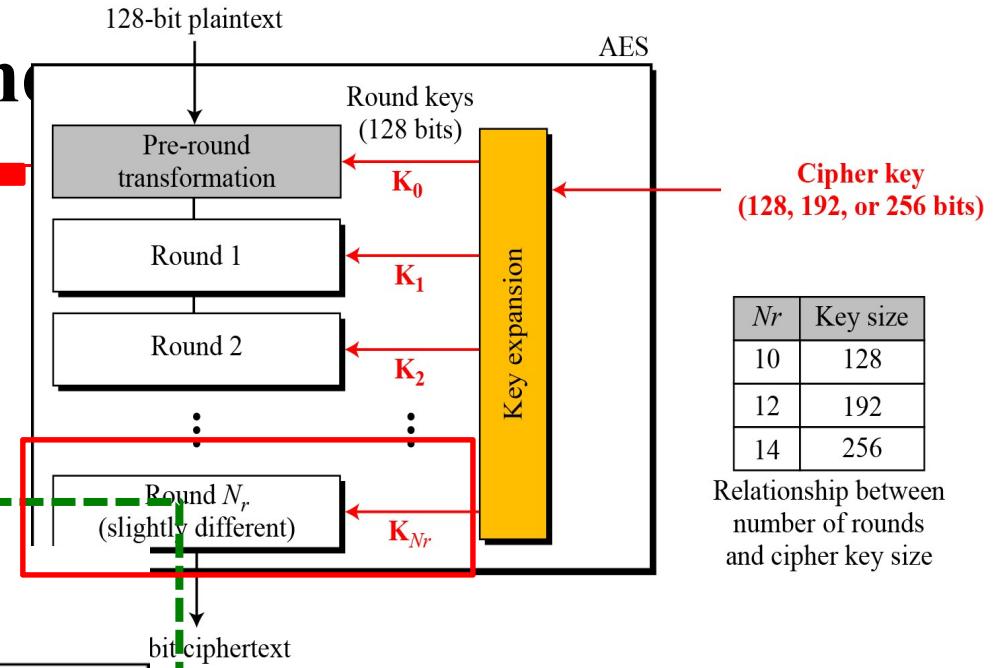
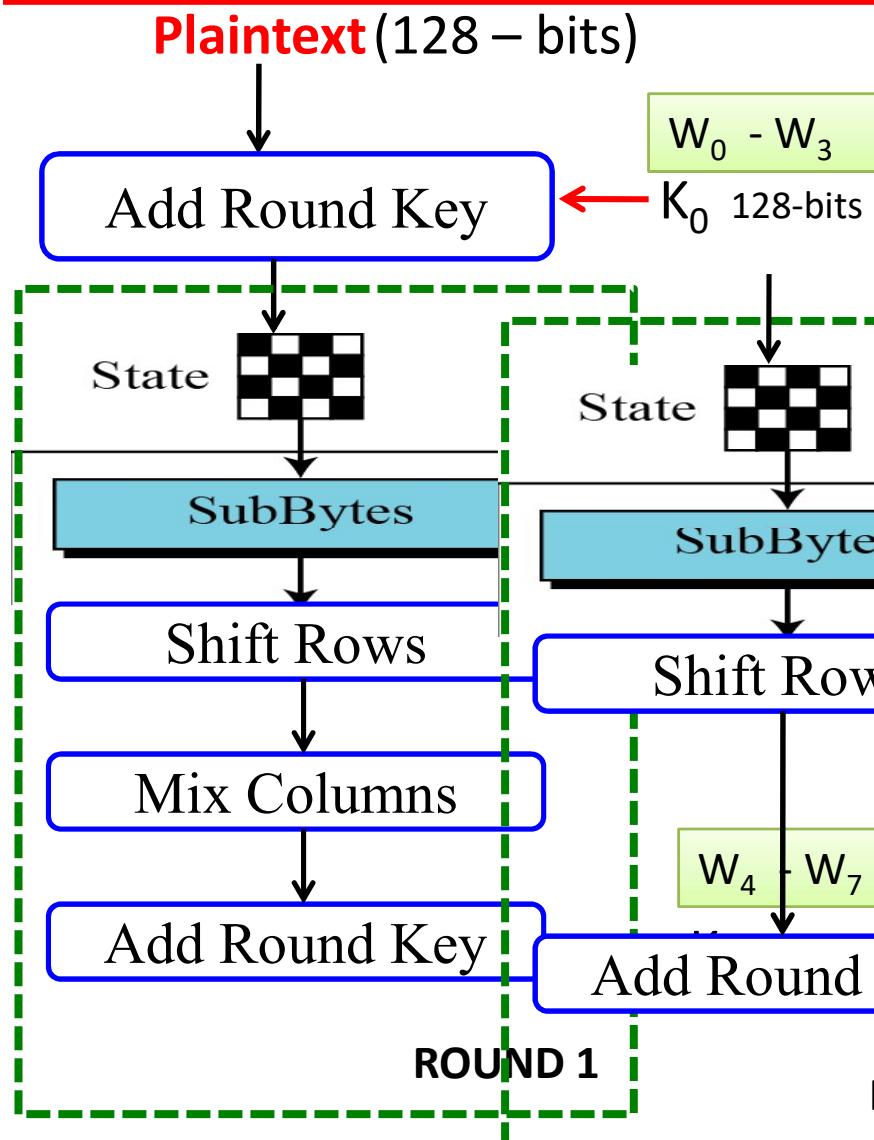
**KEY** (128 – bits)

KEY <sub>0</sub>	KEY <sub>4</sub>	KEY <sub>8</sub>	KEY <sub>12</sub>
KEY <sub>1</sub>	KEY <sub>5</sub>	KEY <sub>9</sub>	KEY <sub>13</sub>
KEY <sub>2</sub>	KEY <sub>6</sub>	KEY <sub>10</sub>	KEY <sub>14</sub>
KEY <sub>3</sub>	KEY <sub>7</sub>	KEY <sub>11</sub>	KEY <sub>15</sub>

Intermediate Results  
are saved in STATE  
ARRAY



# Advanced Encryption Standard

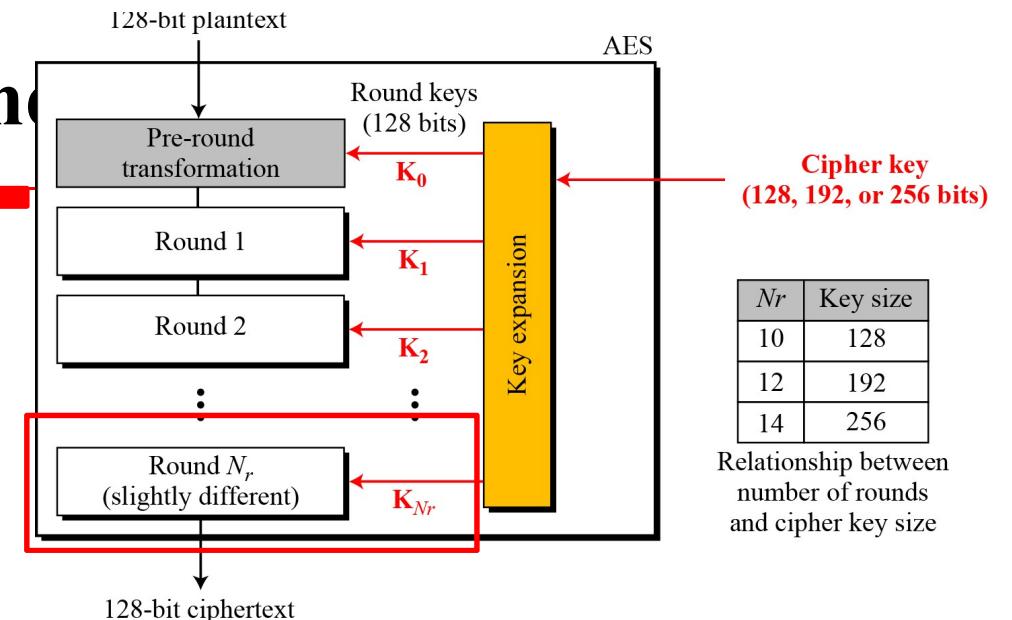
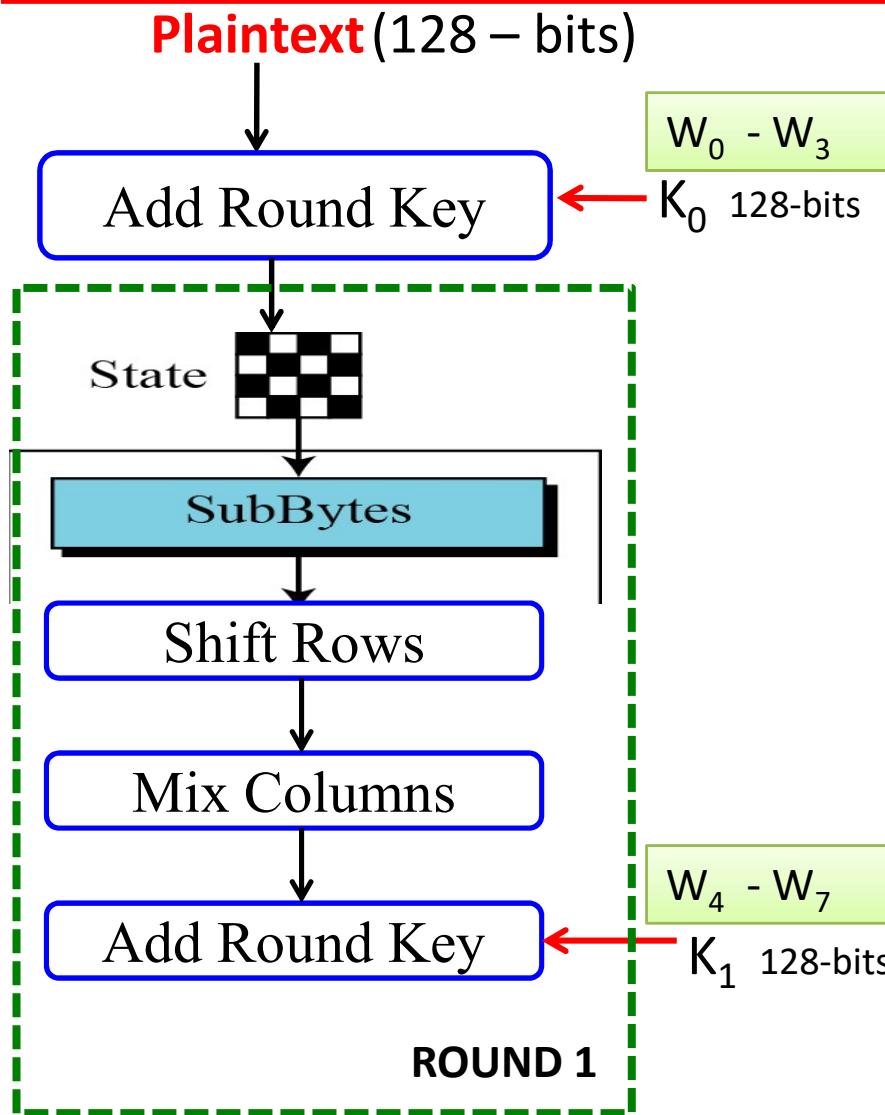


Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size



# Advanced Encryption Standard



<i>Nr</i>	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size



# Advanced Encryption Standard (AES)

---

1. Each transformation takes a state and creates another state to be used for the next round or transformation
2. The pre-round section uses only one transformation (AddRoundKey)
3. The last round uses only 3 transformations (MixColumns transformation is missing)
4. At DECRYPTION side, the Inverse Transformations are used - InvSubByte, InvShiftRows, InvMixColumns and AddRoundKey (self-invertible)

SUBSTITUTION

PERMUTATION

MIXING

KEY ADDING



# Advanced Encryption Standard (AES)

## SUBSTITUTION

AES uses two invertible transformations.

*SubBytes*

*InvSubBytes*

*SubBytes*

The first transformation, SubBytes, is used at the encryption site. To substitute a byte, we interpret the byte as two hexadecimal digits.

58

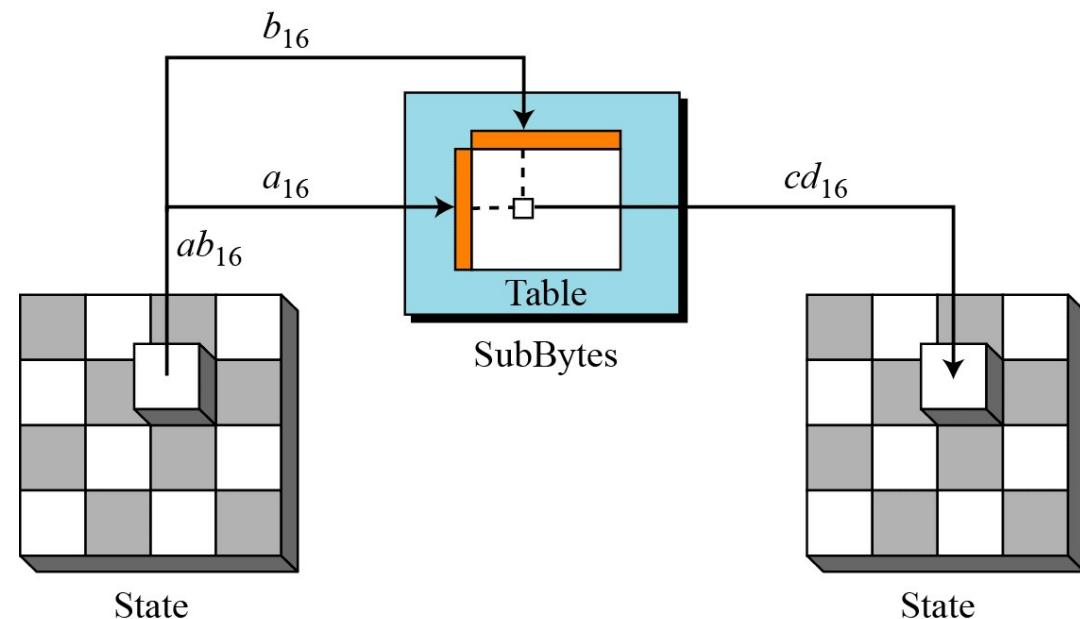


TABLE 3.4. AES ByteSub.

00	12	0C	08
04	04	00	23
12	12	13	19
14	00	11	19

63	C9	FE	30
F2	F2	63	26
C9	C9	7D	D4
FA	63	82	D4

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

**Table 7.2** *InvSubBytes transformation table*

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

# Advanced Encryption Standard (AES)

---

SUBSTITUTION

PERMUTATION

MIXING

KEY ADDIING



# Advanced Encryption Standard (AES)

PERMUTATION

Another transformation found in a round is SHIFTING.

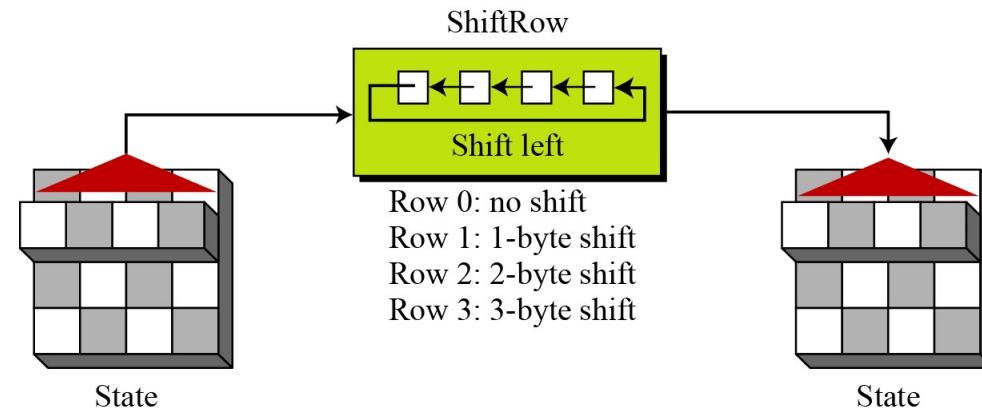
*ShiftRows*

*InvShiftRows*

*ShiftRows*

The shifting is done to left.

The number of shifts depends on the row number of the state matrix



*InvShiftRows*

In decryption, the transformation is **InvShiftRows**

**Shifting is done to RIGHT.**



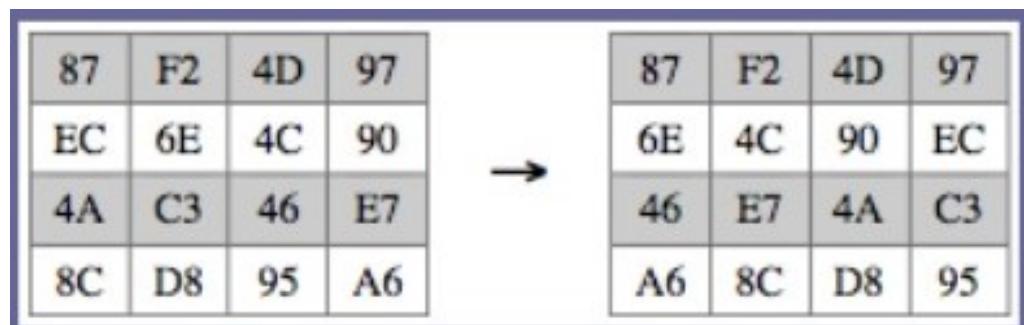
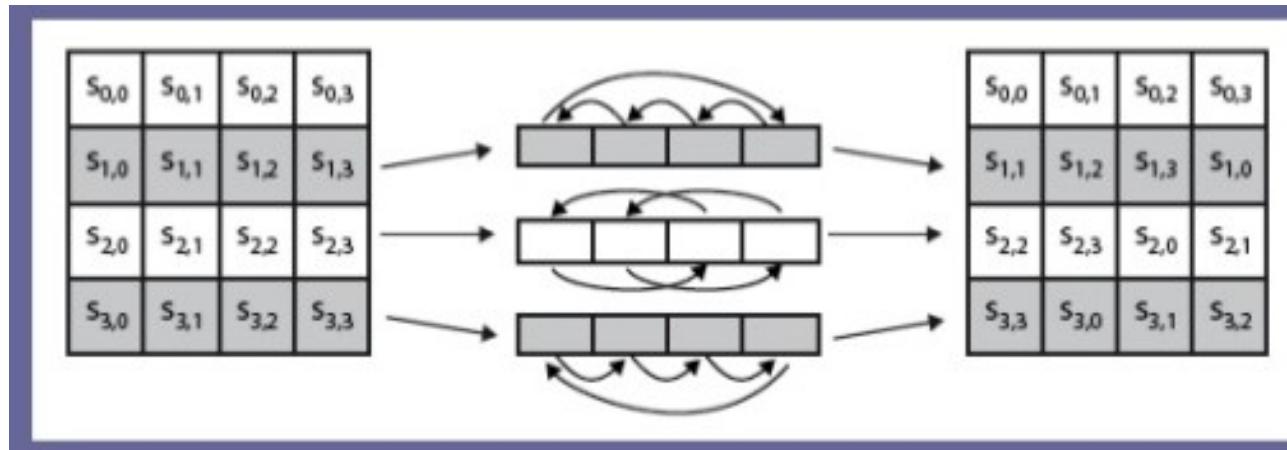
# Advanced Encryption Standard (AES)

## PERMUTATION

Another transformation found in a round is SHIFTING.

*ShiftRows*

*InvShiftRows*



# Advanced Encryption Standard (AES)

---

SUBSTITUTION

PERMUTATION

MIXING



# Advanced Encryption Standard (AES)

MIXING

$$\begin{array}{l} ax + by + cz + dt \\ ex + fy + gz + ht \\ ix + jy + kz + lt \\ mx + ny + oz + pt \end{array} \xrightarrow{\text{New matrix}} = \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ t \end{bmatrix}$$

**Constant matrix**



# Advanced Encryption Standard (AES)

MIXING

1. An interbyte transformation is required that changes the bits inside a byte, based on the bits inside the neighboring bytes.
2. We need to mix bytes to provide diffusion at the bit level.
3. The mixing transformation changes the contents of each byte by taking 4 bytes at a time and combining them to recreate 4 new bytes.
4. To guarantee that each byte is different, the combination process first multiplies each byte with a different constant and then mixes them
5. Mixing is provided by **MATRIX MULTIPLICATION**



# Advanced Encryption Standard (AES)

MIXING

1. AES defines a transformation called **MixColumns**, to achieve this goal
2. *There is also an inverse transformation called InvMixColumns.*
3. The following Constant Matrices are used for these transformations.

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \xleftarrow{\text{Inverse}} \begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix}$$

$C$      $C^{-1}$



# Advanced Encryption Standard (AES)

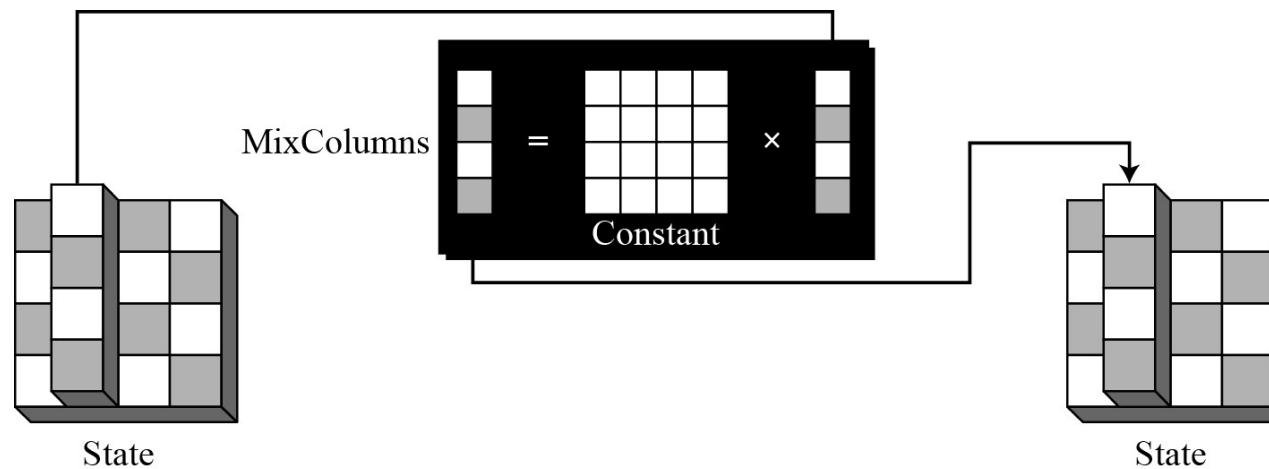
MIXING

## *MixCoulmns*

Transformation operates at the column level

It transforms each column of the state to a new column

The transformation is actually the MATRIX MULTIPLICATION of a state column by a constant square matrix



The bytes in the state column and constants matrix are interpreted as 8-bit words



# Advanced Encryption Standard (AES)

---

SUBSTITUTION

PERMUTATION

MIXING

KEY ADDIING



# Advanced Encryption Standard (AES)

KEY ADDING

## AddRoundKey

AddRoundKey proceeds one column at a time.

AddRoundKey adds a round key word with each state column matrix

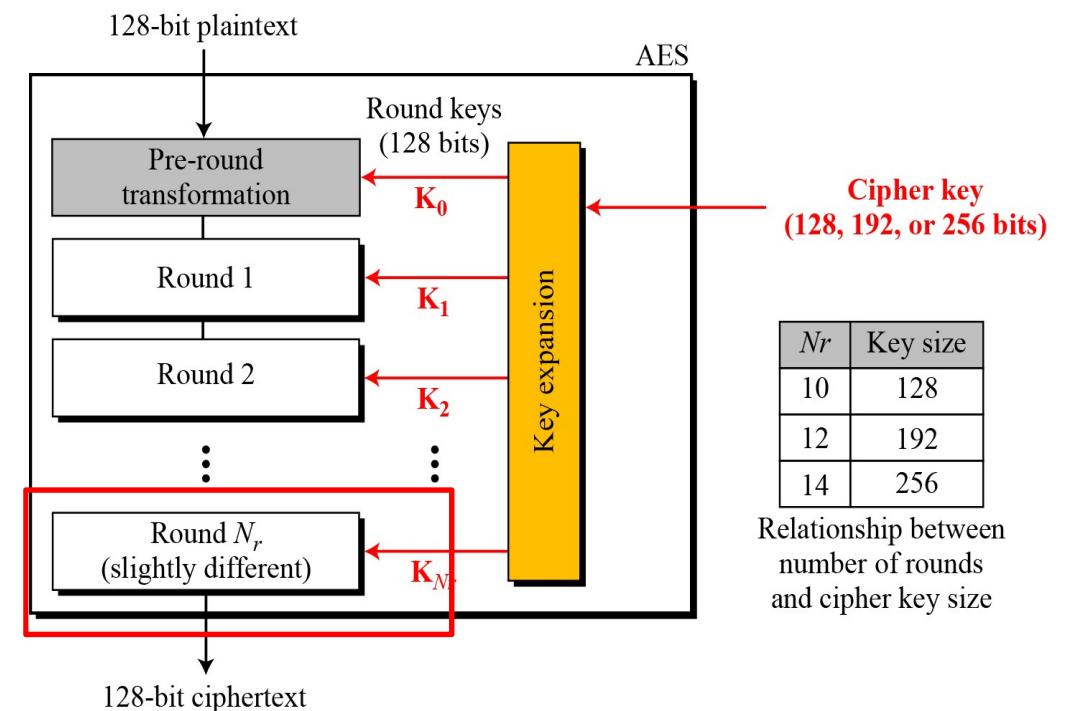
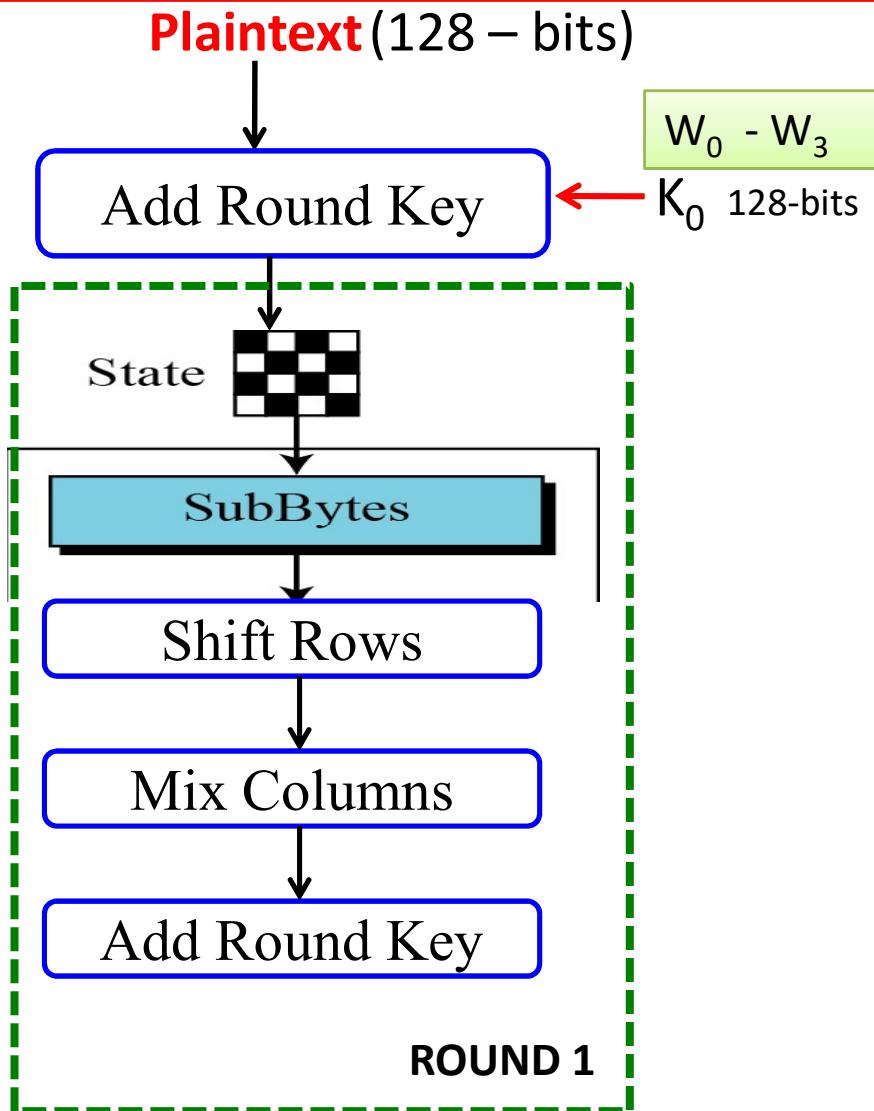
It can be thought as XORing of each column of the state, with the corresponding keyword

47	40	A3	4C		AC	19	28	57		EB	59	8B	1B
37	D4	70	9F	⊕	77	FA	D1	5C	=	40	2E	A1	C3
94	E4	3A	42		66	DC	29	00		F2	38	13	42
ED	A5	A6	BC		F3	21	41	6A		1E	84	E7	D6

Cipher Key is expanded into a set of keywords



# Advanced Encryption Standard (AES)

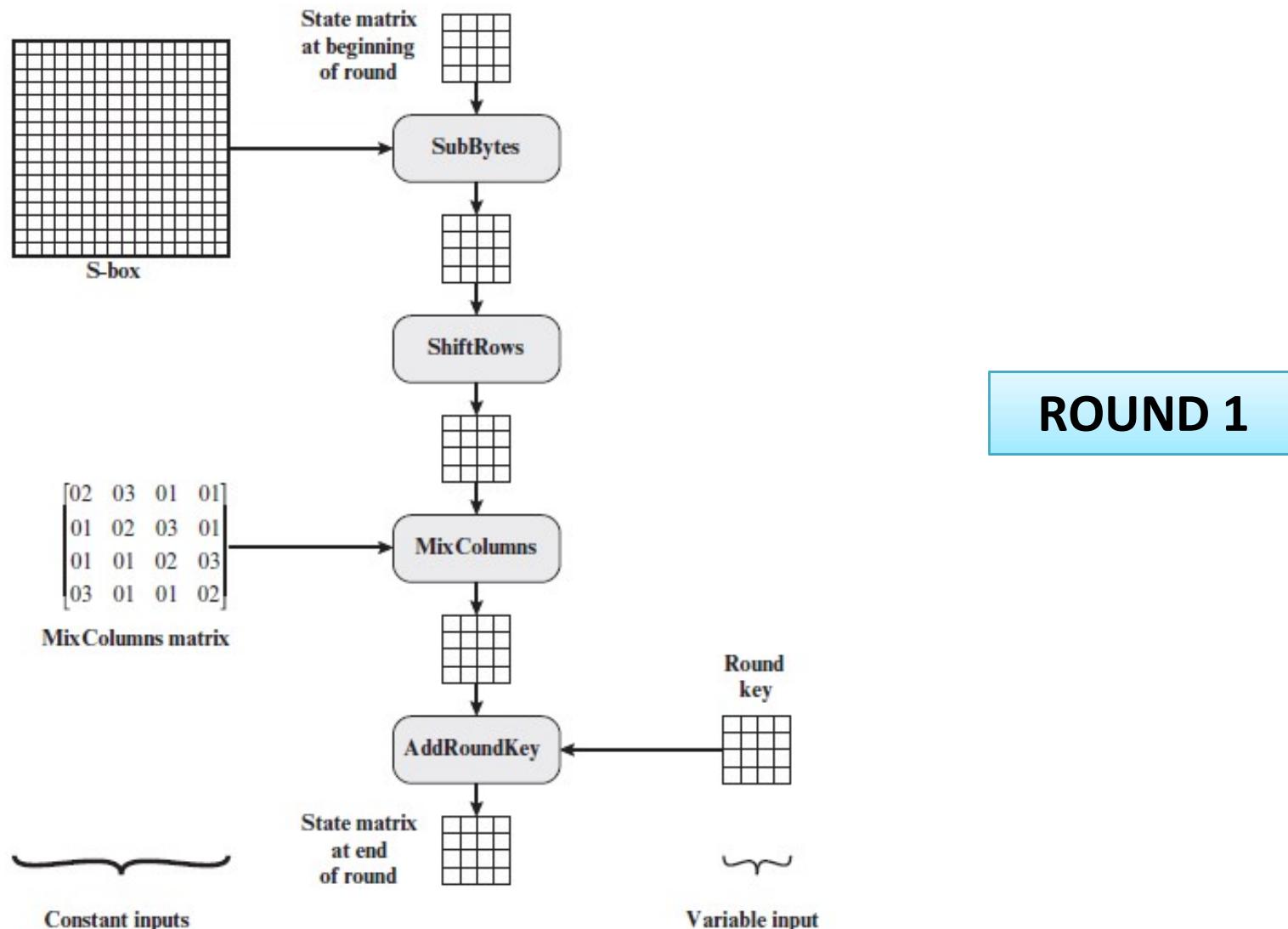


<b>Nr</b>	<b>Key size</b>
10	128
12	192
14	256

Relationship between  
number of rounds  
and cipher key size



# Advanced Encryption Standard (AES)



# Advanced Encryption Standard (AES)

## KEY EXPANSION

1. To create round keys for each round, AES uses a key-expansion process.
2. If the number of rounds is  $N_r$ , the key-expansion routine creates  $N_r + 1$
3. 128-bit round keys are generated from one single 128-bit cipher key.
4. The 1<sup>st</sup> round key is used for pre-round transformation(AddRoundKey)
5. The remaining round keys 2<sup>nd</sup> to 10<sup>th</sup> are used for last transformation at the end of each round
6. The KEY EXPNASION routine creates Round Keys word by word, where word is an array of 4 bytes
7. Total number of words created are  $4*(N_r+1)$



# Advanced Encryption Standard (AES)

KEY  
EXPANSION

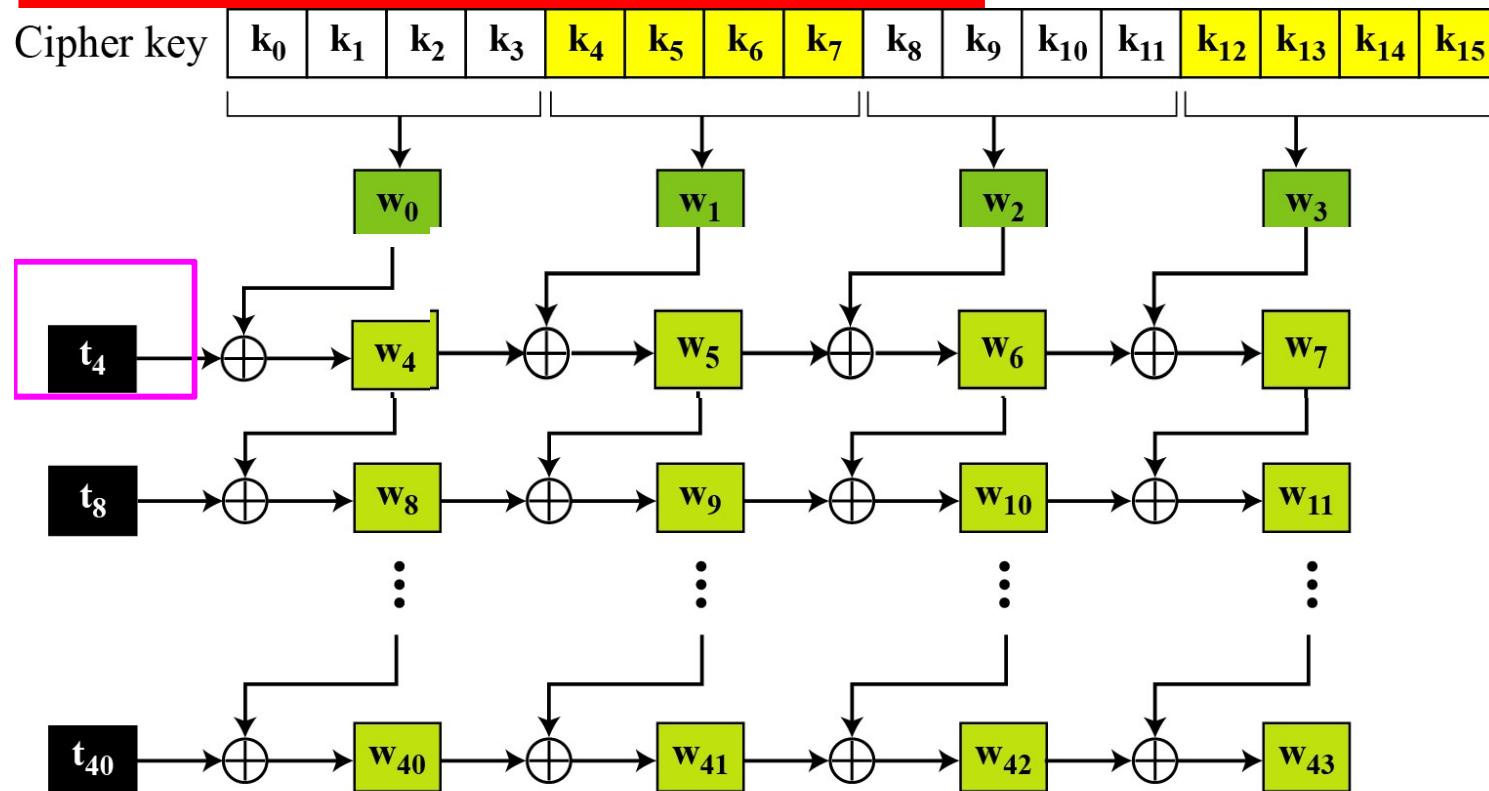
**Table 7.3** *Words for each round*

<i>Round</i>	<i>Words</i>			
Pre-round	$w_0 \quad w_1 \quad w_2 \quad w_3$			
1	$w_4 \quad w_5 \quad w_6 \quad w_7$			
2	$w_8 \quad w_9 \quad w_{10} \quad w_{11}$			
...	...			
$N_r$	$w_{4N_r} \quad w_{4N_r+1} \quad w_{4N_r+2} \quad w_{4N_r+3}$			



# Advanced Encryption Standard (AES)

## KEY EXPANSION



The rest of the words  $w_i$  for  $i = 4$  to  $43$  are made as follows:

1. If  $(i \bmod 4) \neq 0$ ,  $w_i = w_{i-1} \oplus w_{i-4}$
2. If  $(i \bmod 4) = 0$ ,  $w_i = t \oplus w_{i-4}$

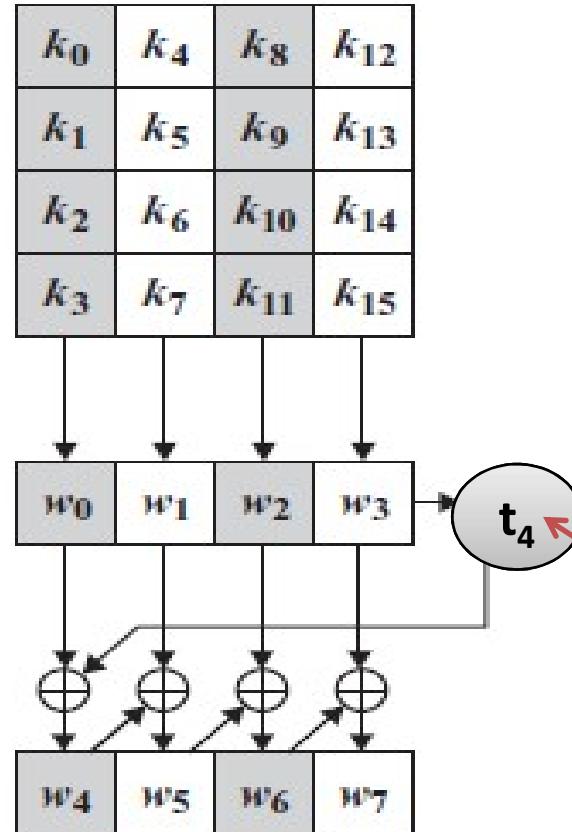
The 1<sup>st</sup> four words

$(w_0, w_1, w_2, w_3)$  are made  
from cipher key



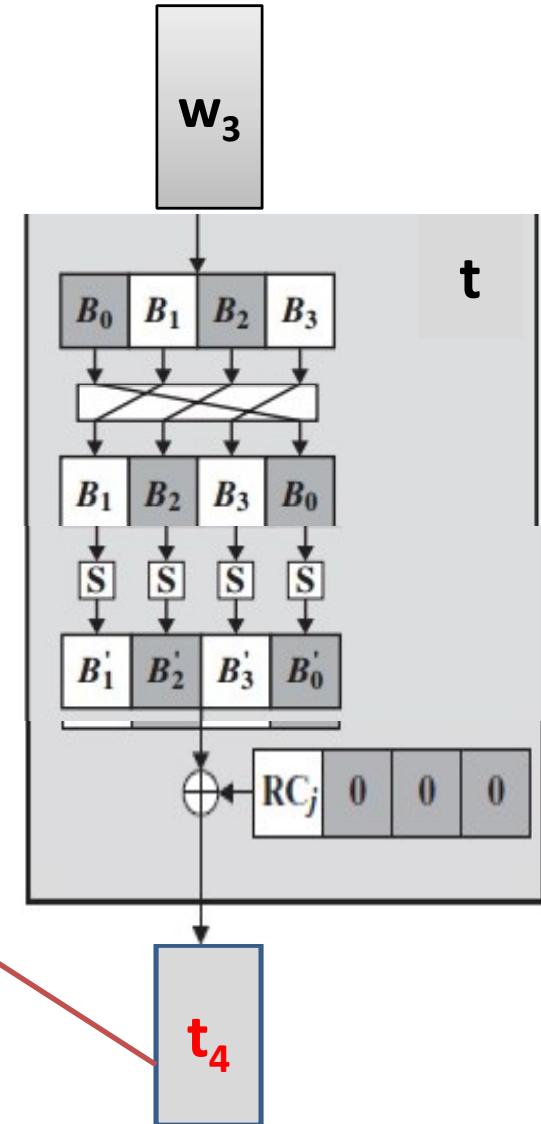
# Advanced Encryption Standard (AES)

## KEY EXPANSION



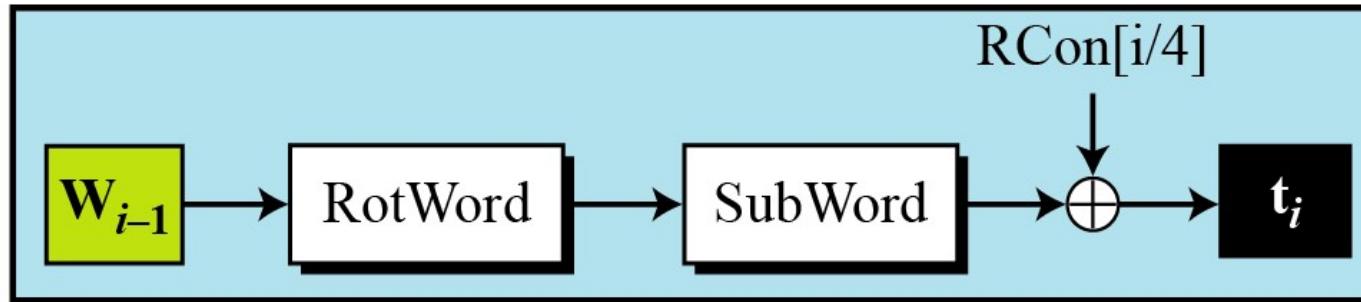
**The rest of the words  $w_i$  for  $i = 4$  to  $43$  are made as follows:**

1. If  $(i \bmod 4) = 0$ ,  $w_i = t \oplus w_{i-4}$



# Advanced Encryption Standard (AES)

## KEY EXPANSION



Making of  $t_i$  (temporary) words  $i = 4 N_r$

1. RotWord performs a one-byte circular left shift on a word. This means that an input word  $[B_0, B_1, B_2, B_3]$  is transformed into  $[B_1, B_2, B_3, B_0]$ .
2. SubWord performs a byte substitution on each byte of its input word, using the S-box
3. The result of steps 1 and 2 is XORed with a round constant,  $Rcon[j]$ .



# Advanced Encryption Standard (AES)

## KEY EXPANSION

**Table 7.4** *RCon constants*

Round	Constant (RCon)	Round	Constant (RCon)
1	( <u>01</u> 00 00 00) <sub>16</sub>	6	( <u>20</u> 00 00 00) <sub>16</sub>
2	( <u>02</u> 00 00 00) <sub>16</sub>	7	( <u>40</u> 00 00 00) <sub>16</sub>
3	( <u>04</u> 00 00 00) <sub>16</sub>	8	( <u>80</u> 00 00 00) <sub>16</sub>
4	( <u>08</u> 00 00 00) <sub>16</sub>	9	( <u>1B</u> 00 00 00) <sub>16</sub>
5	( <u>10</u> 00 00 00) <sub>16</sub>	10	( <u>36</u> 00 00 00) <sub>16</sub>

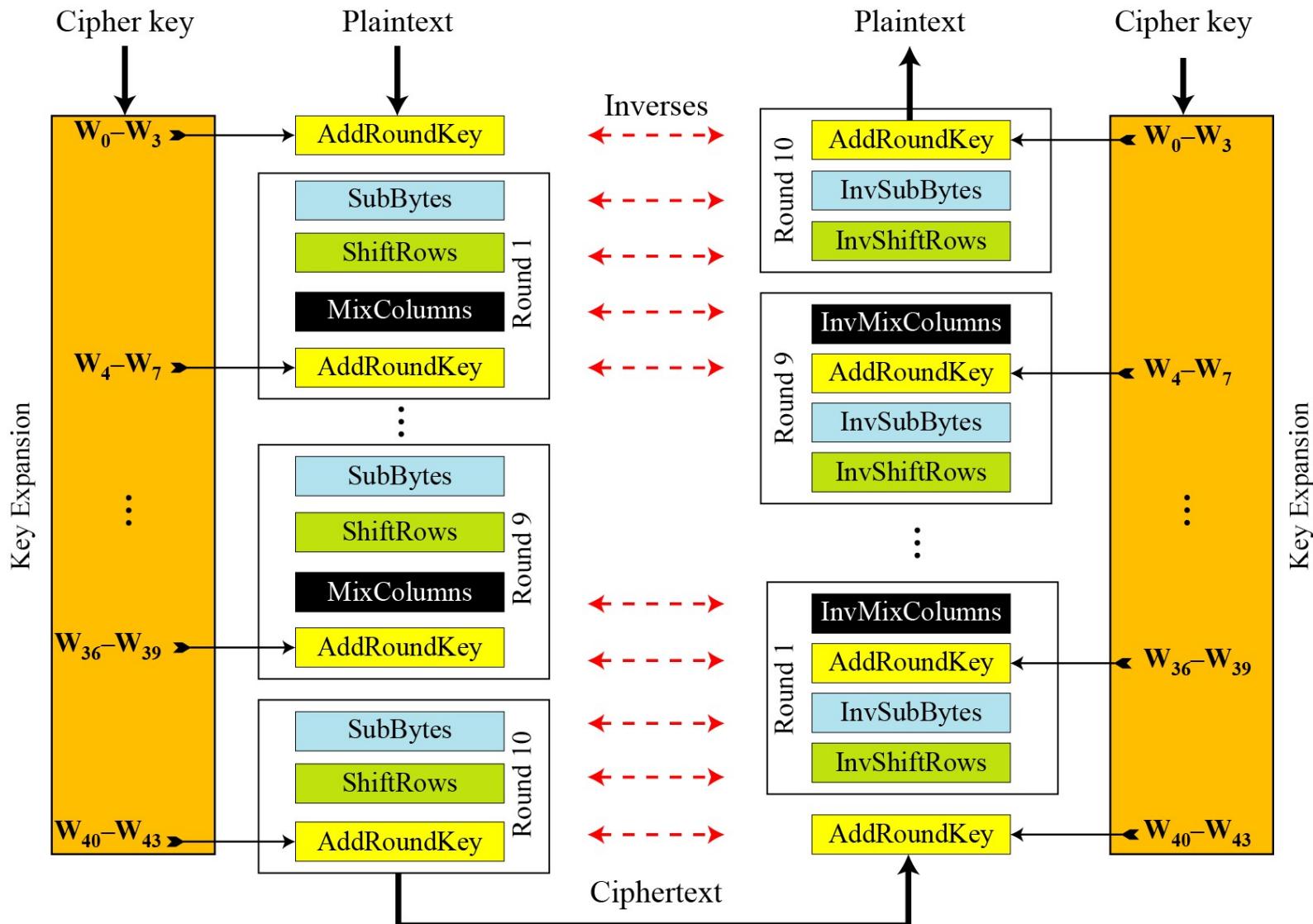
1. The round constant is a word in which the three rightmost bytes are always 0.
2. Thus, the effect of an XOR of a word with Rcon is to only perform an XOR on the leftmost byte of the word.
3. The round constant is different for each round and is defined as

$$Rcon[j] = (RC[j], 0, 0, 0), \text{ with } RC[1] = 1, RC[j] = 2 \# RC[j - 1]$$

AND

With multiplication defined over the field GF(28).





# Module 2: Symmetric & Asymmetric Key Cryptography and Key Management

---

## 2.1

- Block Cipher
  - ✓ Block Cipher Principles
  - ✓ Block Cipher Modes of Operation
  - ✓ Data Encryption Standard (DES)
  - ✓ Double DES, Triple DES
  - ✓ Advanced Encryption Standard (AES)
- Stream Cipher
  - ✓ RC5 algorithm



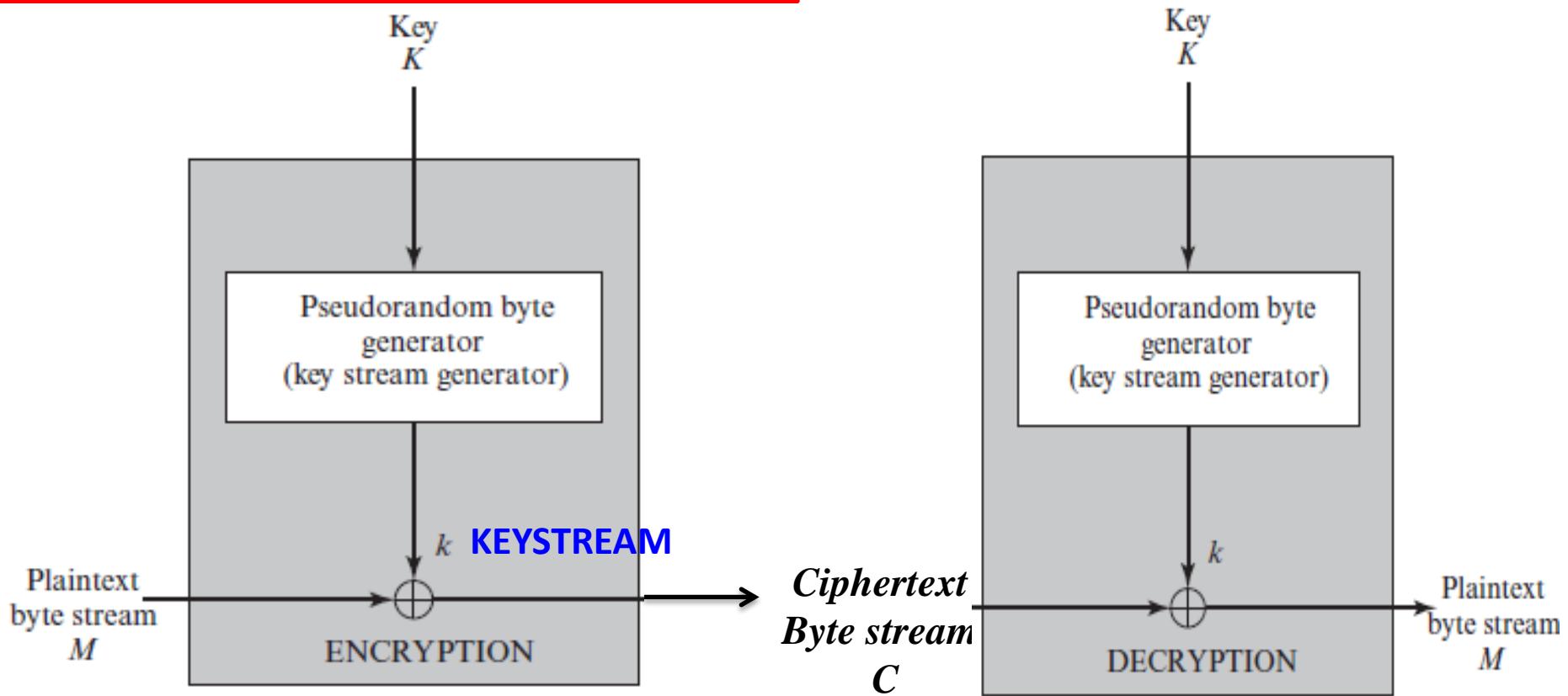
# **STREAM CIPHER**

---

1. Stream ciphers are a type of encryption algorithm that process an individual bit, byte, or character of plaintext at a time.
2. A Stream Cipher is used for symmetric key cryptography
3. Stream ciphers are often faster than block ciphers in hardware and require circuitry that is less complex.
4. They are also useful when transmission errors are likely to occur because they have little or no error propagation.



# STREAM CIPHER (Block Diagram)



A **keystream** is a sequence of pseudorandom digits which extend to the length of the plaintext in order to uniquely encrypt each character based on the corresponding digit in the keystream

# Types of Stream cipher

---

1. Stream ciphers can be classified into
  - a) **Synchronous:** A Synchronous Stream Cipher generates a keystream based on internal states not related to the plaintext or ciphertext. This means that the stream is generated pseudorandomly outside of the context of what is being encrypted. A binary additive stream cipher is the term used for a stream cipher which XOR's the bits with the bits of the plaintext.
  - b) **Self-synchronizing:** A Self-synchronizing Stream Cipher, also known as an asynchronous stream cipher or ciphertext autokey (CTAK), is a stream cipher which uses the previous N digits in order to compute the keystream used for the next N characters.



# Design considerations for a stream cipher

---

1. The encryption sequence should have a large period. A pseudorandom number generator uses a function that produces a deterministic stream of bits that eventually repeats. The longer the period of repeat the more difficult it will be to do cryptanalysis.
2. The keystream should approximate the properties of a true random number stream as close as possible. The more random-appearing the keystream is, the more randomized the ciphertext is, making cryptanalysis more difficult.
3. To guard against brute-force attacks, the key needs to be sufficiently long



# RC5 Stream Cipher

---

1. Symmetric key algorithm developed by Ron Rivest
2. Its main features are that it is quite fast and uses primitive operations like (Addition, XOR, Shift).
3. It allows for a variable number of rounds and a variable bit-size key to add flexibility
4. RC5 requires less memory for execution and hence it is not only suitable for desktop computers but also for smart cards.
5. It has been incorporated into RSA data security



# RC5 Stream Cipher

1. RC5 allows for variable values in 3 parameters and is denoted as  $\text{RC5} - w/r/b$

w:- Word size in bits

r:- Number of Rounds

b:- Number of 8-bits byte in the key

***RC5 uses 2 word blocks***

***RC5 – 32/16/16***

Parameter	Allowed Values
Word size in bits (RC5 encrypts 2-word blocks at a time)	16, 32, 64
Number of rounds	0-255
Number of 8-bit bytes (octets) in the key	0-255

The following conclusions emerge from the table:

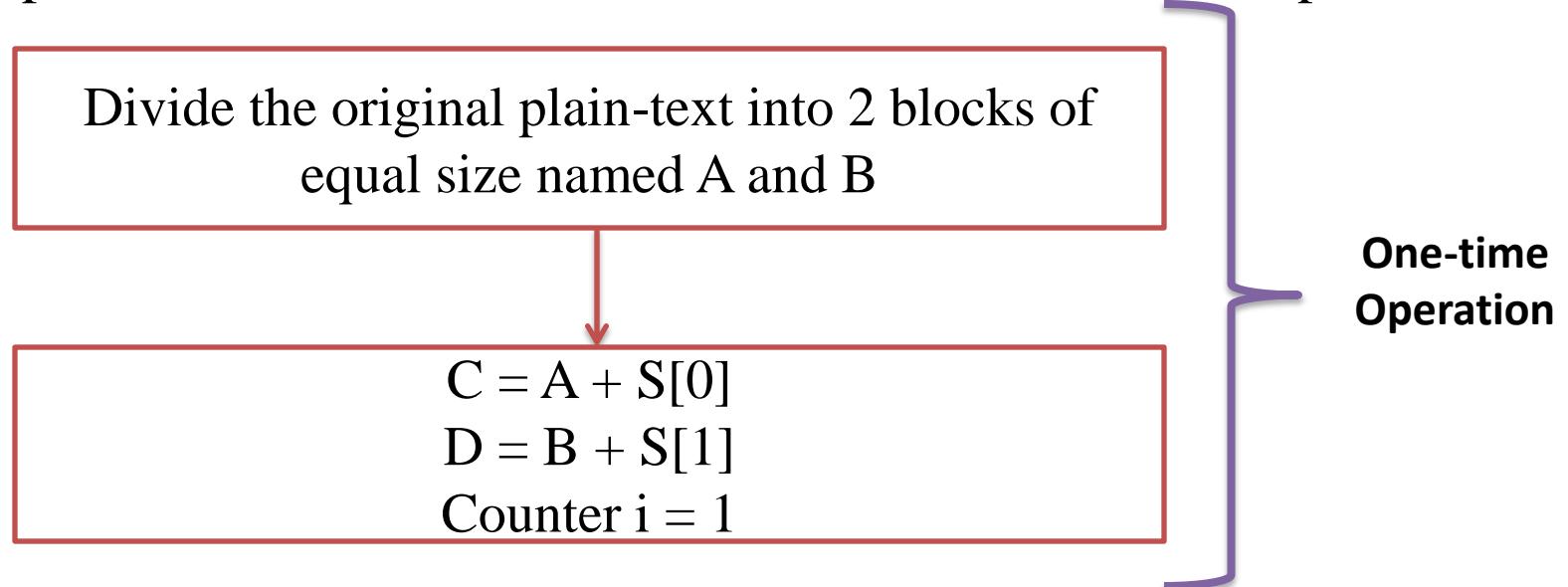
- The plain-text block size can be of 32, 64 or 128 bits (since 2-word blocks are used).
- The key length can be 0 to 2040 bits (since we have specified the allowed values for 8-bit keys).

Rivest has suggested RC5-32/12/16 as the *minimum safety version.*



# RC5 Stream Cipher

1. Assume an input plain block with size 64-bit.
2. In the first two steps of the one-time initial operation, the input plain text is divided into two 32 bits blocks A and B.
3. The 1<sup>st</sup> two sub-keys ( S[0] and S[1] ) are added to A and B respectively
4. This produces C and D and marks the end of the one-time operation



# RC5 Stream Cipher

Divide the original plain-text into 2 blocks of equal size named A and B

$$C = A + S[0], D = B + S[1], \text{Counter } i = 1$$

Note: First perform all the left-hand side steps, and then come to the right hand side steps, as indicated by the step numbers.

1. XOR C and D to produce E.

4. XOR D and F to produce G.

2. Circular-left shift E by D bits.

5. Circular-left shift G by F bits.

3. Add E and  $S[2i]$  to produce F.

6. Add G and  $S[2i + 1]$  to produce H.

Increment i by 1

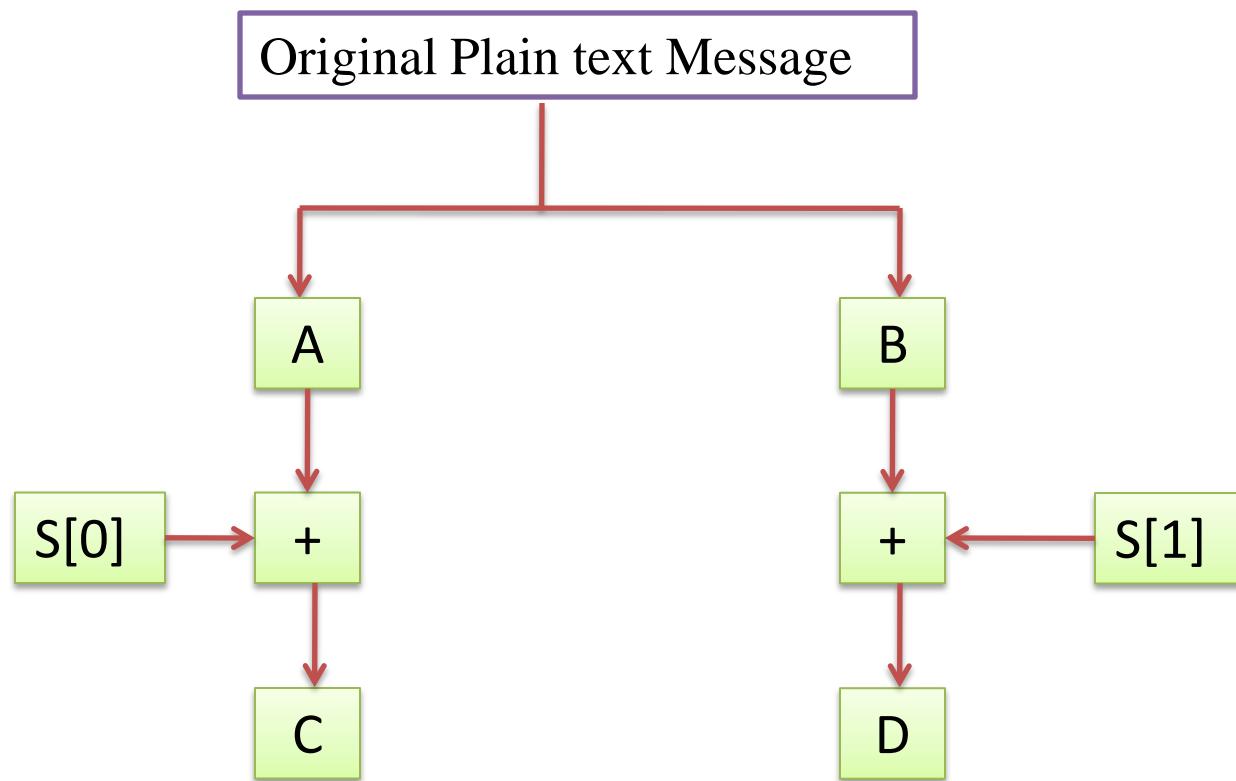
Call F as C  
(i.e.  $C = F$ )  
Call H as D  
(i.e.  $D = H$ )

Check:  
Is  $i > r$ ?

Yes  
Stop



# RC5 Stream Cipher: One Time Initial Operation



# RC5 Stream Cipher: Details of ONE Round

Step 1: XOR C and D

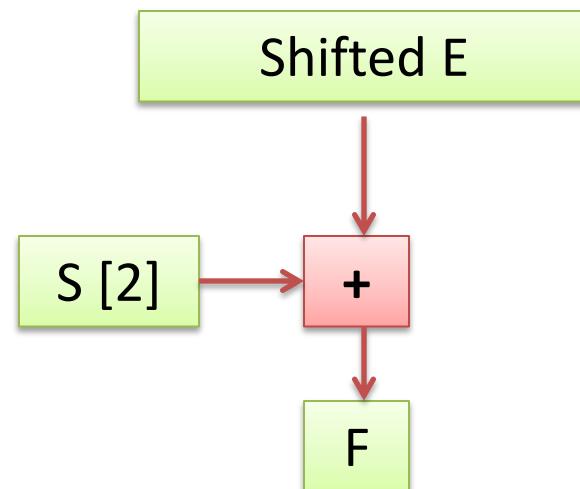
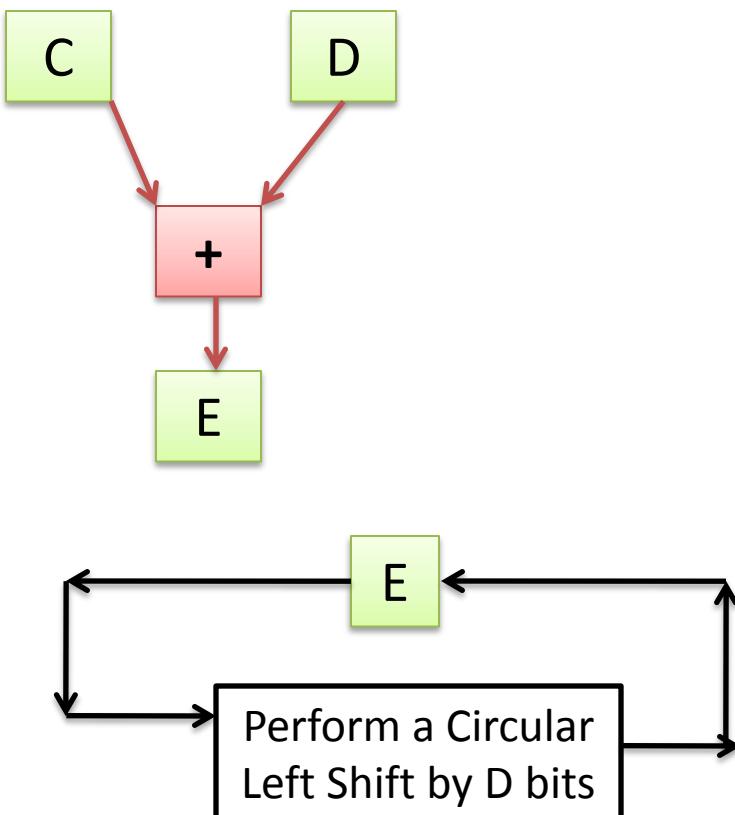
Step 4: XOR D and F

Step 2: Circular Left Shift E

Step 5: Circular Left Shift G

Step 3: Add E and Next Sub-key

Step 6: Add G and Next Subkey



# RC5 Stream Cipher: Details of ONE Round

**Miscellaneous Tasks:** In this step, we check to see if all the rounds are over or not. For this we perform the following steps

1. Increment by  $i$
2. Check to see if  $i < r$

Assume that  $i$  is still less than  $r$ , we rename F as C and H as D and return back to step 1.

$$\begin{aligned} A &= A + S[0] \\ B &= B + S[1] \end{aligned}$$

## Mathematical Representation of RC5 ENCRYPTION

For  $i = 1$  to  $r$

$$\begin{aligned} A &= ((A \text{ XOR } B) \lll B) + S[2i] \\ B &= ((B \text{ XOR } A) \lll A) + S[2i + 1] \end{aligned}$$

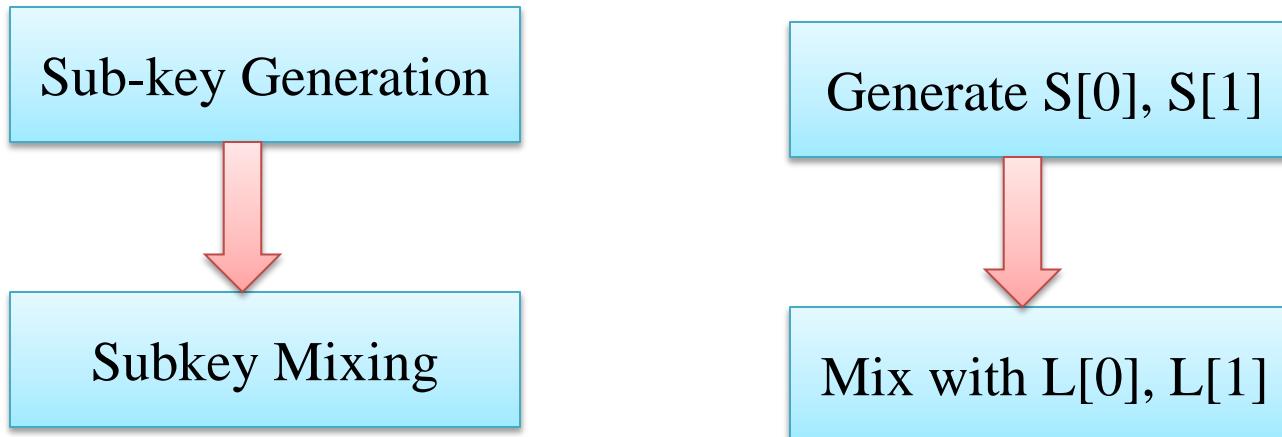
Next i



# RC5 Stream Cipher: Subkey Generation

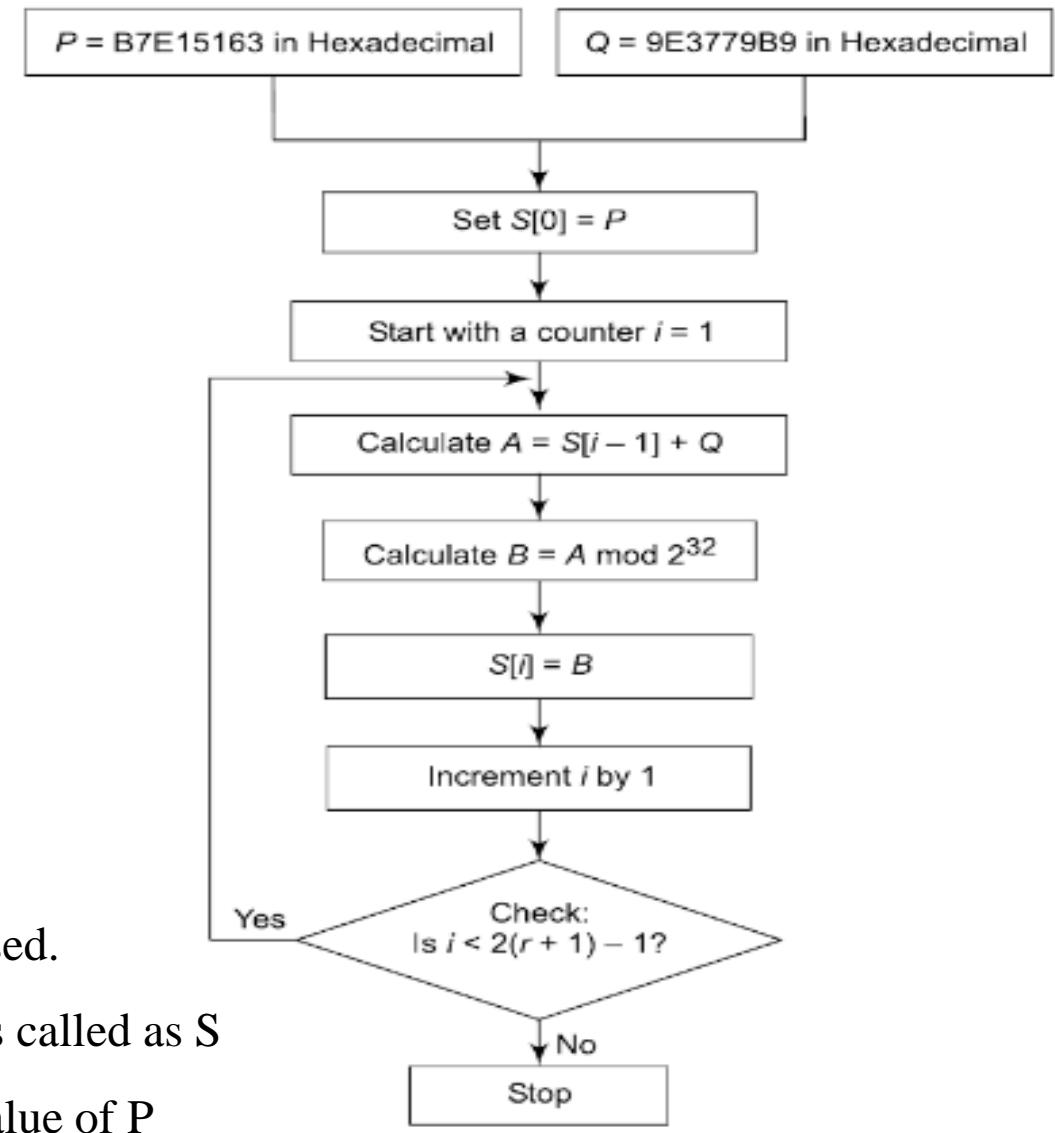
This is a 2-step process

1. The sub-keys denoted by  $S[0], S[1], S[2], \dots$  Are generated
2. The original key is called  $L$ . In the 2<sup>nd</sup> step the subkeys ( $S[0], S[1], \dots$ ) are mixed with the corresponding sub-portions of the original key (i.e.  $L[0], L[1], L[2], \dots$ )



# RC5 Stream Cipher: Subkey Generation

1. In this step, 2 constants P and Q are used.
2. The array of subkeys to be generated is called as S
3. The 1<sup>st</sup> subkey is initialized with the value of P
4. Each next sub-key is calculated on the basis of the previous sub-key and the constant value Q



# Module 2: Symmetric & Asymmetric Key Cryptography and Key Management

---

2.2

- Public Key Cryptography

- ✓ Principles of public key cryptosystems
- ✓ The RSA algorithm
- ✓ The knapsack algorithm
- ✓ ElGamal Algorithm



# RSA Algorithm

- RSA algorithm is asymmetric (public key) cryptography algorithm.
- Asymmetric actually means that it works on two different keys i.e. Public Key and Private Key.
- As the name describes that the Public Key is given to everyone and Private key is kept private.
- The RSA algorithm is named after those who invented it in 1978: Ron Rivest, Adi Shamir, and Leonard Adleman.

The RSA algorithm holds the following features –

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



# RSA Algorithm

The following steps highlight how it works:

1. Generating the keys
2. Encryption      The pair of numbers  $(n,e)$  makes up the public key.
3. Decryption      The pair  $(n,d)$  makes up the private key.

## 1. Generating the keys

Select two large prime numbers,  $p$  and  $q$ .

Calculate  $n=p * q$ .

Calculate the *totient* function;  $\phi(n)$

Select an integer  $e$ , such that  $e$  is **co-prime** to  $\phi(n)$  and  $1 < e < \phi(n)$ .

Calculate  $d$  such that  $d = e^{-1} \bmod \phi(n)$ . i.e  $e.d = 1 \bmod \phi(n)$

$d$  can be found using the **extended euclidean algorithm**.



# RSA Algorithm

---

The following steps highlight how it works:

1. Generating the keys
2. Encryption   The pair of numbers  $(n,e)$  makes up the public key.
3. Decryption   The pair  $(n,d)$  makes up the private key.

## Encryption

Given a plaintext  $P$ , represented as a number, the ciphertext  $C$  is calculated as:

$$C = P^e \bmod n$$

## Decryption

Given a Ciphertext  $C$ , represented as a number, the plaintext  $P$  is calculated as:

$$P = C^d \bmod n$$



# RSA Algorithm (Example)

Let  $p = 3$  and  $q = 11$

$$n = p * q$$

$$= 3 * 11$$

$$= 33$$

$$\phi(33) = 11 * 3$$

$$= (11-1) * (3-1)$$

$$= 10 * 2 = 20$$

Select  $e$  such that  $e$  is co-prime to  $\phi(n)$  and  $1 < e < \phi(n)$ .

$$1 < e < 20$$

$$e = 7$$

Calculate  $d$  such that  $d = e^{-1} \pmod{\phi(n)}$ . i.e  $e.d = 1 \pmod{\phi(n)}$

$$d = 3$$

## 1. Generating the keys

Select two large prime numbers,  $p$  and  $q$ .

Calculate  $n = p * q$ .

Calculate the **totient** function;  $\phi(n)$

Select an integer  $e$ , such that  $e$  is **co-prime** to  $\phi(n)$  and  $1 < e < \phi(n)$ .

Calculate  $d$  such that  $d = e^{-1} \pmod{\phi(n)}$ . i.e  $e.d = 1 \pmod{\phi(n)}$



# RSA Algorithm

---

The pair of numbers  $(n,e)$  makes up the public key.

$$\text{Public Key} = (33, 7)$$

$$\begin{aligned} C &= P^e \bmod n \\ &= 31^7 \bmod 33 \\ &= 4 \end{aligned}$$

The pair  $(n,d)$  makes up the private key.

$$\text{Private Key} = (33, 3)$$

$$\begin{aligned} P &= C^d \bmod n \\ &= 4^3 \bmod 33 \\ &= 31 \end{aligned}$$



# The Knapsack Algorithm

---

1. Knapsack Encryption Algorithm is the first general public key cryptography algorithm.
2. It is developed by Ralph Merkle and Martin Hellman in 1978.
3. As it is a Public key cryptography, it needs two different keys.
4. One is Public key which is used for Encryption process and the other one is Private key which is used for Decryption process.



# The Knapsack Algorithm – Steps to follow

---

## STEPS TO FOLLOW:

1. Choose a super increasing knapsack {1, 2, 4, 10, 20, 40} as the private key.
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.
3. Calculate the values of Public key using m and n.

Super Increasing knapsack problem. Super increasing knapsack is a sequence in which every next term is greater than the sum of all preceding terms.



# The Knapsack Algorithm – Example

Bob and Alice wants to use knapsack for their secure communication. Bob has a Superincreasing knapsack  $\{1, 2, 4, 10, 20, 40\}$  and he chooses  $m=110$  and  $n=31$ . Calculate the public and private keys of Bob.

## Calculating Public Key

Multiply all the values of private key by the number n and then find modulo m

$$1 * 31 \text{ mod } 110 = 31$$

$$2 * 31 \text{ mod } 110 = 62$$

$$4 * 31 \text{ mod } 110 = 14$$

$$10 * 31 \text{ mod } 110 = 90$$

$$20 * 31 \text{ mod } 110 = 70$$

$$40 * 31 \text{ mod } 110 = 30$$

Thus, our public key is  $\{31, 62, 14, 90, 70, 30\}$

And Private key is  $\{1, 2, 4, 10, 20, 40\}$ .

Calculate the values of Public key using m and n.



# The Knapsack Algorithm – Example

## ENCRYPTION

Plain text is **100100111100101110**

- 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      Public Key Is {31, 62, 14, 90, 70, 30}**

$$1 * 31 + 0 * 62 + 0 * 14 + 1 * 90 + 0 * 70 + 0 * 30 = 121$$

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.**



# The Knapsack Algorithm – Example

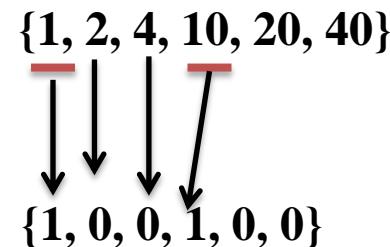
## DECRYPTION

Cipher text is 121 197 205.

$$\begin{aligned} n \times n^{-1} \bmod(m) \\ = 31 \times \bmod(110) \\ = 71 \end{aligned}$$

Multiply 71 With Each Block Of Cipher Text Take Modulo m.

$$\begin{aligned} 121 \times 71 \bmod(110) \\ = 11 \end{aligned}$$



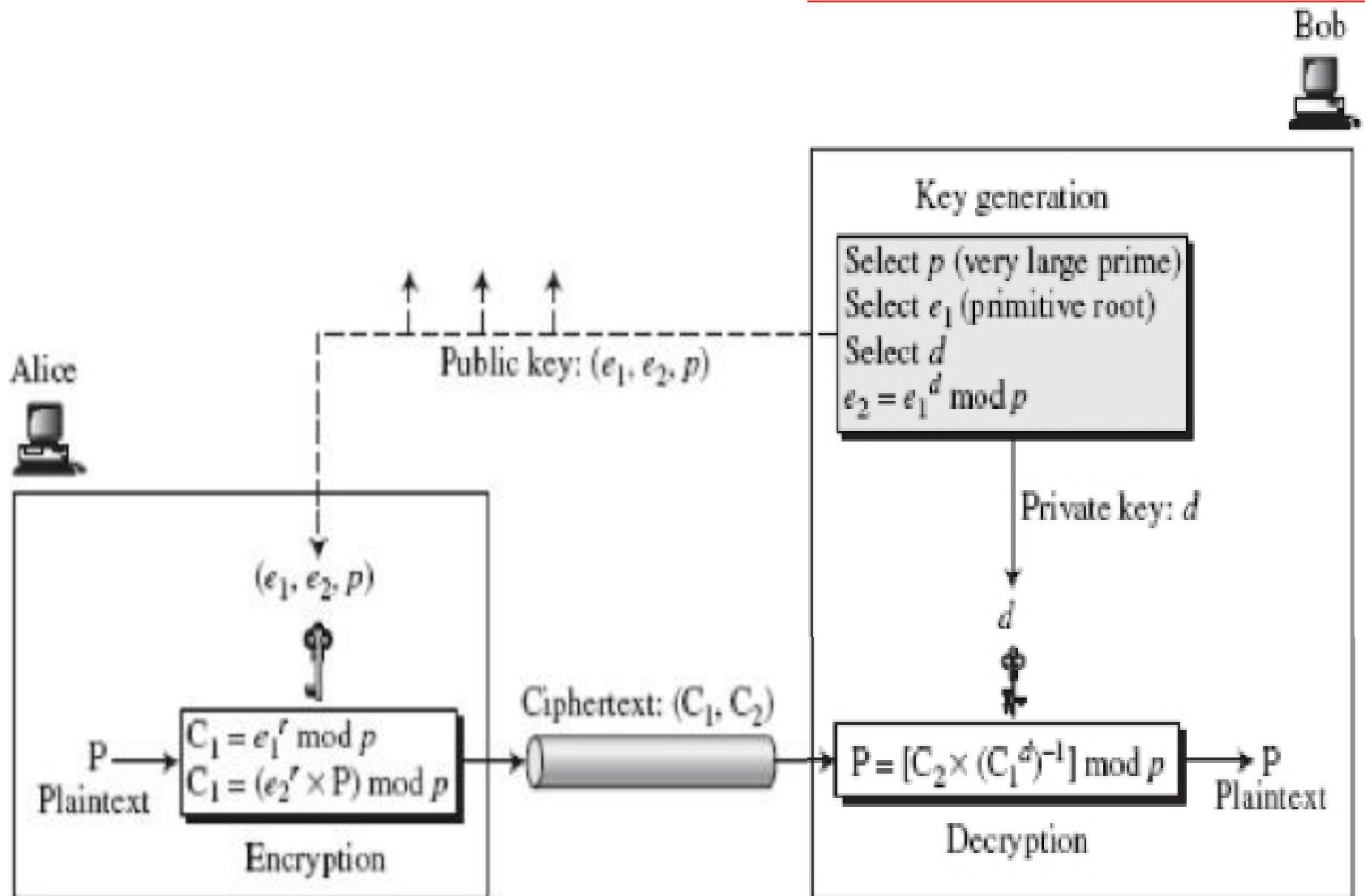
# ElGamal Algorithm – Public Key Cryptography

---

1. ElGamal encryption is an public-key cryptosystem. It uses asymmetric key encryption for communicating between two parties and encrypting the message.
2. It is named after its inventor, Taher Elgamal.
3. This cryptosystem is based on the difficulty of finding discrete logarithm in a cyclic group that is even if we know  $g^a$  and  $g^k$ , it is extremely difficult to compute  $g^{ak}$ .



# ElGamal Algorithm – Procedure



# ElGamal Algorithm – Public Key Cryptography

---

## 1. Generating Keys

- a) Select a large Prime Number (p)
- b) Select  $e_1$  (Primitive Root)
- c) Select Decryption key  $d$  (Private Key)
- d)  $e_2 = e_1^d \text{ mod } p$
- e) Encryption Key ( Public Key ) =  $(e_1, e_2, p)$

## 2. Encryption

- a) Select Random Integer  $r$
- b)  $C_1 = e_1^r \text{ mod } p$
- c)  $C_2 = (\text{Plain Text} * e_2^r) \text{ mod } p$
- d) Cipher Text =  $(C_1, C_2)$

## 3. Decryption

- a) Plain Text =  $[C_2 * (C_1^d)^{-1}] \text{ mod } p$



# ElGamal Algorithm – Public Key Cryptography

---

## 1. Generating Keys

- a) Select a large Prime Number (p) **11**
- b) Select e1 (Primitive Root)
- c) Select Decryption key d (Private Key)
- d)  $e2 = e1^d \text{ mod } p$
- e) Encryption Key ( Public Key ) = (e1, e2, p)

## 2. Encryption

- a) Select Random Integer r
- b)  $C1 = e1^r \text{ mod } p$
- c)  $C2 = (\text{Plain Text} * e2^r) \text{ mod } p$
- d) Cipher Text = (C1, C2)

## 3. Decryption

- a) Plain Text =  $[C2 * (C1^d)^{-1}] \text{ mod } p$



# ElGamal Algorithm – Public Key Cryptography

$$2^5 = 2 * 2^4$$

## 1. Generating Keys

a) Select a large Prime Number ( $p$ )

b) Select  $e_1$  (Primitive Root)

11

POWERS

$$2^5 = 2 * 5$$

$$2^5 = 10$$

	1	2	3	4	5	6	7	8	9	10
1	1	1	1							
2	2	4	8	5	10	9	7	3	6	1
3										
4										
10										

a



## Primitive Roots

- If  $x^n = a$  then  $a$  is called the  $n$ -th root of  $x$
- For any prime number  $p$ , if we have a number  $a$  such that powers of  $a \bmod p$  generate all the numbers between 1 to  $p-1$  then  $a$  is called a **Primitive Root** of  $p$ .

# ElGamal Algorithm – Public Key Cryptography

---

## 1. Generating Keys

a) Select a large Prime Number (p) **11**

b) Select e1 (Primitive Root) **2**

**Assume Plaintext**

**7**

c) Select Decryption key d (Private Key) **3**

d)  $e2 = e1^d \text{ mod } p = 2^3 \text{ mod } 11 = 8$

e) Encryption Key ( Public Key ) = (e1, e2, p) **(2, 8, 11)**

## 2. Encryption

a) Select Random Integer r **4**

b)  $C1 = e1^r \text{ mod } p = 2^4 \text{ mod } 11 = 5$

c)  $C2 = (\text{Plain Text} * e2^r) \text{ mod } p = 7 * 8^4 \text{ mod } 11 = 6$

d) Cipher Text = (C1, C2) = (5, 6)

## 3. Decryption

a) Plain Text =  $[C2 * (C1^d)^{-1}] \text{ mod } p = [6 * (5^3)^{-1}] \text{ mod } 11 = 7$



# Module 2: Symmetric & Asymmetric Key Cryptography and Key Management

---

## 2.3

- Key management techniques:
  - ✓ Using symmetric and asymmetric algorithms and trusted third party.
  - ✓ Diffie Hellman Key exchange algorithm



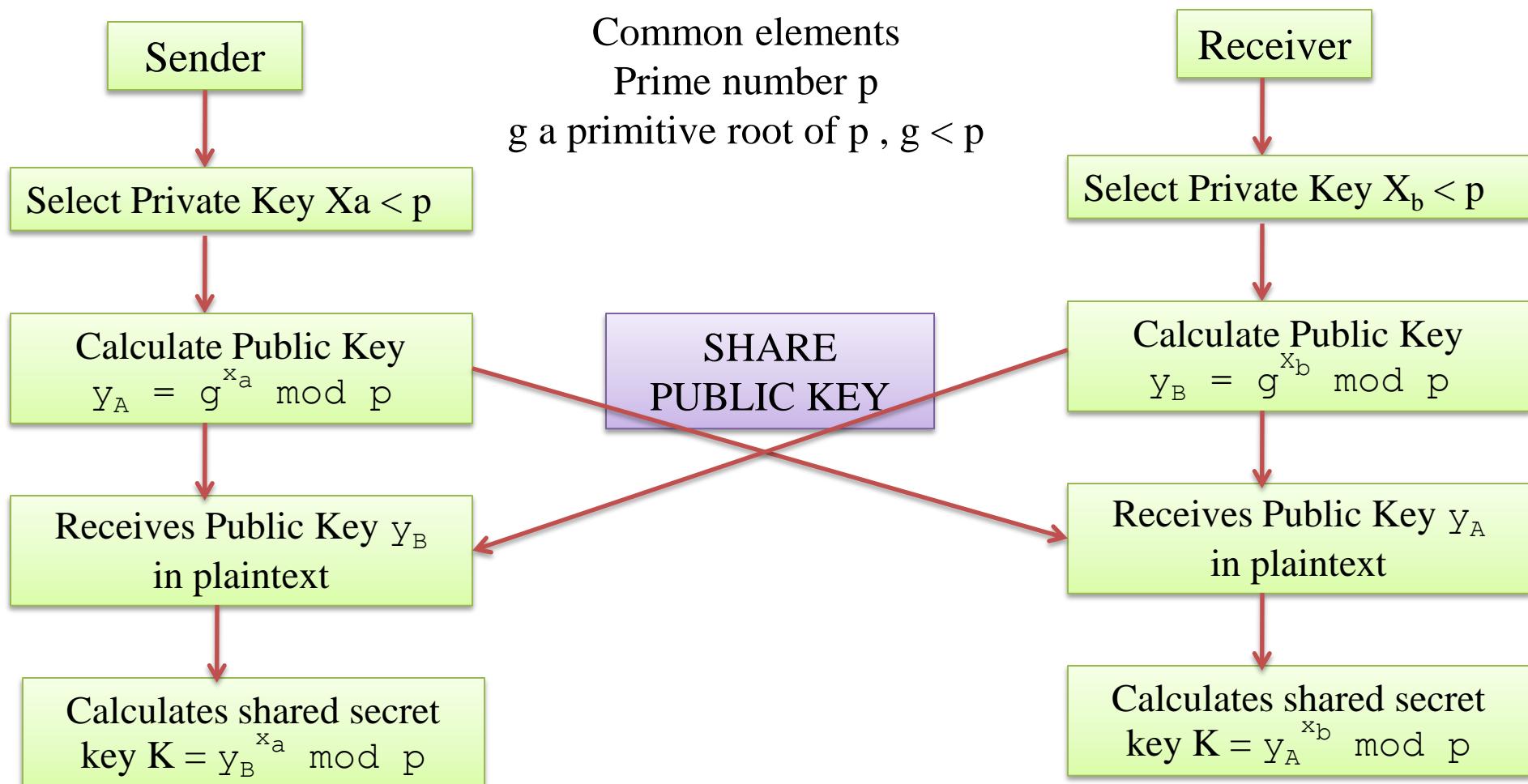
# Diffie-Hellman Key Exchange Algorithm

---

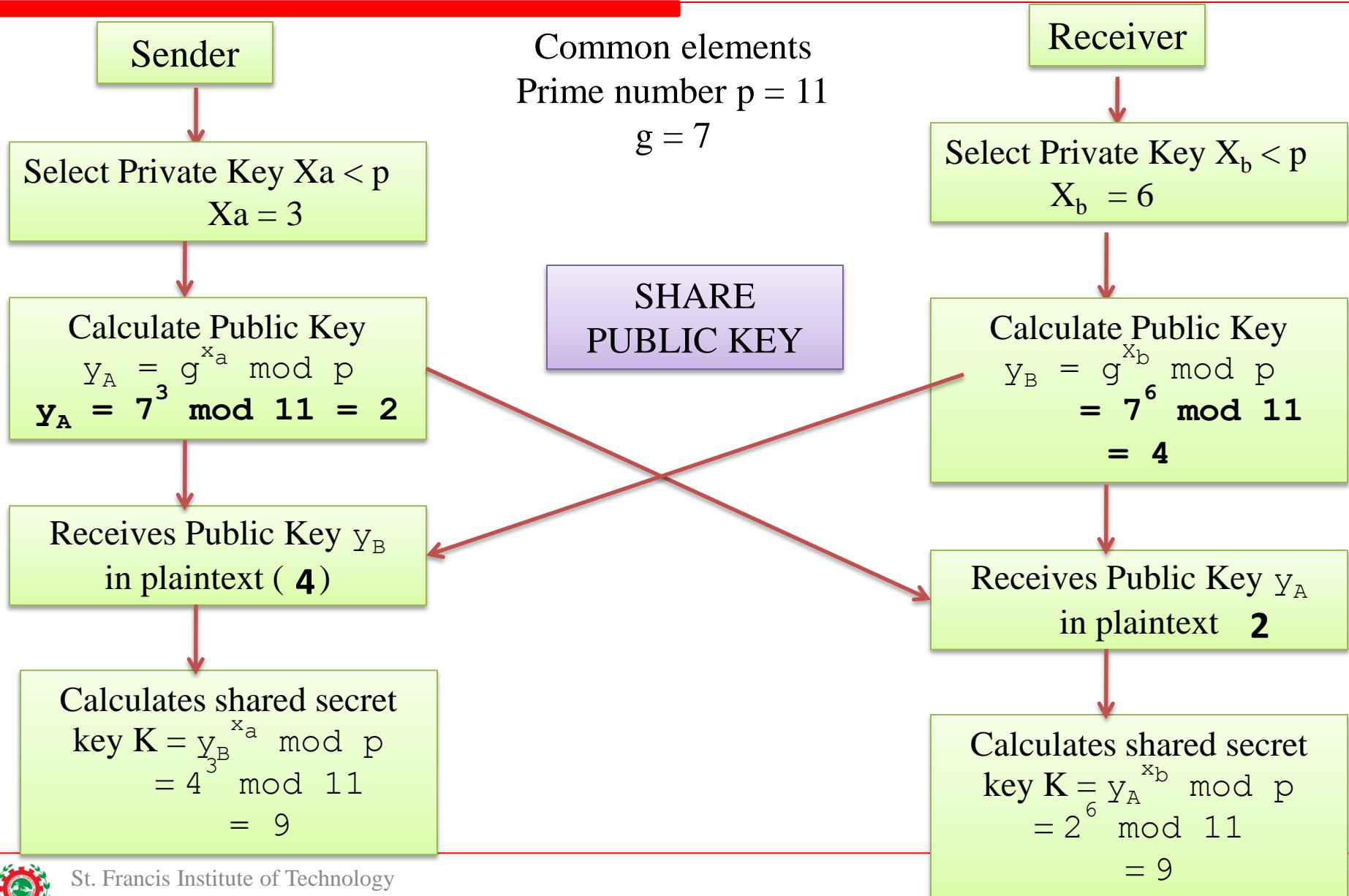
1. Diffie Hellman (DH) key exchange algorithm is a method for securely exchanging cryptographic keys over a public communications channel.
2. Keys are not actually exchanged – they are jointly derived.
3. It is named after their inventors Whitfield Diffie and Martin Hellman.
4. The purpose of the algorithm is to enable two users to securely exchange keys that can then be used for subsequent symmetric encryption of messages
5. The algorithm itself is limited to the exchange of secret values
6. The DH algorithm depends for its effectiveness on the difficulty of computing discrete logarithms



# Diffie-Hellman Key Exchange Algorithm - Procedure



# Diffie-Hellman Key Exchange Algorithm - Example



# Diffie-Hellman Key Exchange Algorithm - Example

---

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



# Diffie-Hellman Key Exchange Algorithm - Example

---

Alice and Bob agrees on  $q=23$  and  $\alpha=7$ . Alice chooses secret key as 3 and Bob chooses as 6. Find their public and private keys.

Public Keys:  $Y_A$  and  $Y_B$  Private Key:  $K$



# Diffie-Hellman Key Exchange Algorithm - Example

---

Users A and B use the Diffie Hellman key exchange technique with a common prime  $q = 71$  and primitive root  $= 7$

1. If user A has private key  $X_a = 5$ , then what is A's Public Key  $Y_a$
2. If user B has private key  $X_b = 12$ , what is B's public key  $Y_b$ 
  3. What is the shared secret key?



# Cryptography and System Security (CSS)

## Course Code: CSC 604



### Subject Incharge

Ankita Karia  
Assistant Professor  
Room No. 421  
email: [ankitakaria@sfit.ac.in](mailto:ankitakaria@sfit.ac.in)



# Module 3:Hashes, Message Digests and Digital Certificates

---

## 3.1

- Cryptographic Hash Functions
- Properties of secure hash function
- Various Cryptographic Hash Functions:
  - ✓ MD5,
  - ✓ SHA-1,
  - ✓ MAC, HMAC, CMAC

## 3.2

- Digital Certificate
- X.509, PKI



# Hash Functions

---

1. A hash function  $H$  accepts a variable-length block of data  $M$  as input and produces a fixed-size hash value  $h = H(M)$ .
2. A “good” hash function produces output that are evenly distributed and apparently random.
3. The principal object of a hash function is **data integrity**. → Detects changes in message
4. A change to any bit or bits in  $M$  results, with high probability, in a change to the hash value.
5. Values returned by a hash function are called message digest or simply hash values
6. The kind of hash function needed for security applications is referred to as a **Cryptographic Hash Function**.



# Cryptographic Hash Functions

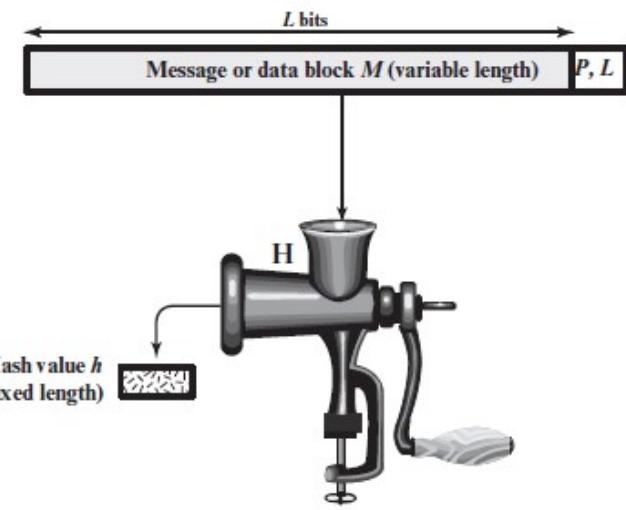
1. A cryptographic hash function is an algorithm for which it is computationally infeasible to find either
  - (a) a data object that maps to a pre-specified hash result (**the one-way property**)
  - (b) two data objects that map to the same hash result (**the collision-free property**).
2. Hash functions are often used to determine whether or not data has changed.

For a hash value  $h = H(x)$ ,

$x$  is said to be the **PREIMAGE** of  $h$ .



Because  $H$  is a many-to-one mapping, for any given hash value  $h$ , there will in general be multiple preimages.



$P, L$  = padding plus length field



# Cryptographic Hash Functions - PROPERTIES

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

1. **Variable input size:** H can be applied to a block of data of any size.
2. **Fixed output size :** H produces a fixed-length output.
3. **Efficiency :**  $H(x)$  is relatively easy to compute for any given  $x$ , making both hardware and software implementations practical.

A **COLLISION** occurs if we have  $x \neq y$  and  $H(x) = H(y)$ . Because we are using hash functions for data integrity, collisions are clearly undesirable.



# Cryptographic Hash Functions - PROPERTIES

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

## 4. Pre-Image Resistance (**the one-way property**)

This property means that it should be computationally hard to reverse a hash function.

## 5. Second Pre-Image Resistance (**weak collision resistant**)

This property means given an input and its hash, it should be hard to find a different input with the same hash.

## 6. 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.

A hash function that satisfies the first five properties is referred to as a **Weak Hash Function**.

If the sixth property, collision resistant, is also satisfied, then it is referred to as a **Strong Hash Function**.



# The Resistant Properties Required For Various Hash Function Applications.

---

	Preimage Resistant	Second Preimage Resistant	Collision Resistant
Hash + digital signature	yes	yes	yes*
Intrusion detection and virus detection		yes	
Hash + symmetric encryption			
One-way password file	yes		
MAC	yes	yes	yes*

\*Resistance required if attacker is able to mount a chosen message attack



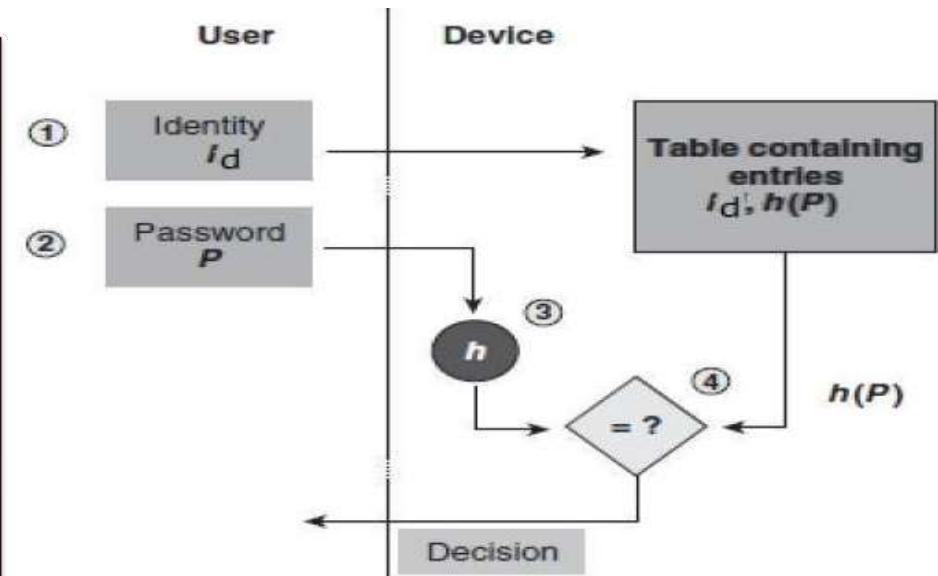
# Applications of Hash Functions

There are two direct applications of hash function based on its cryptographic properties.

## 1. Password Storage

- ✓ All login processes store the hash values of passwords in the file.
- ✓ The Password file consists of a table of pairs which are in the form (user id,  $h(P)$ ).
- ✓ The process of logon is depicted in the following illustration –

An intruder can only see the hashes of passwords, even if he accessed the password. He can neither logon using hash nor can he derive the password from hash value since hash function possesses the property of pre-image resistance.

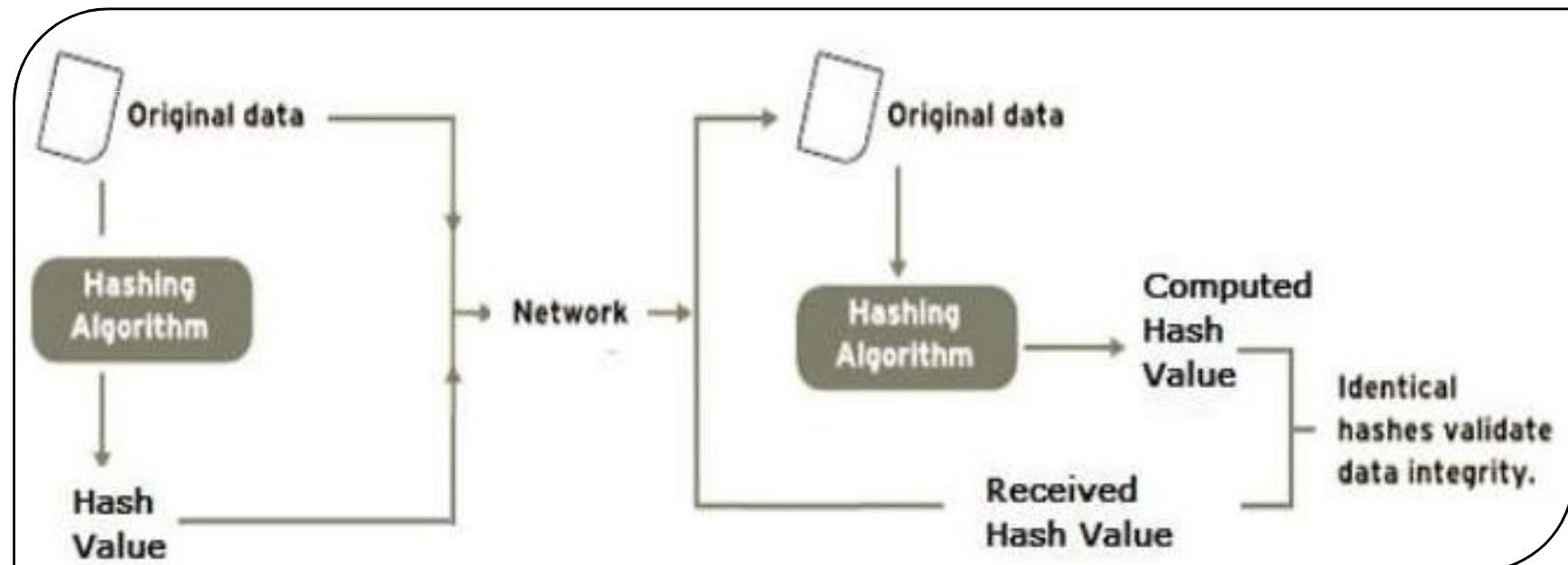


# Applications of Hash Functions

There are two direct applications of hash function based on its cryptographic properties.

## 2. Data Integrity Check

- ✓ It is used to generate the checksums on data files.
- ✓ This application provides assurance to the user about correctness of the data.



# Applications of Hash Functions

---

- Message Authentication:
  - Verifies the integrity of the message
- Digital signatures
- Intrusion detection and virus detection
  - keep & check hash of files on system
- Pseudorandom function (PRF) or pseudorandom number generator (PRNG)
  - PRF is used for generating symmetric keys



# Various Cryptographic Hash Functions

---

- **Message Digest (MD)**
  - ✓ MD5 algorithm was developed by Professor Ronald L. Rivest in 1991.
  - ✓ MD5 Message Digest Algorithm takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input.
  - ✓ MD5 digests have been widely used in the software world to provide assurance about integrity of transferred file.
  - ✓ In 2004, collisions were found in MD5. An analytical attack was reported to be successful only in an hour by using computer cluster. This collision attack resulted in compromised MD5 and hence it is no longer recommended for use.
- **Secure Hash Function (SHA)**



# MD 5

The padding consists of a single 1 bit followed by the necessary number of 0 bits.

## Steps to follow:

### 1. Append Padding Bits

### 2. Append Length

After padding, 64 bits are inserted at the end which is used to record the length of the original input

### 3. Initialize MD buffer

### 4. Processing message in 16-word block

512	- 64 = 448
1024	- 64 = 960
1536	- 64 = 1472
2048	- 64 = 1984
2560	- 64 = 2496
3072	- 64 = 3008
3584	- 64 = 3520
4096	- 64 = 4032

## Step 1: APPPEND PADDING BITS

Padding means adding extra bits to the original message. Padding is done such that the total bits are 64 less than a multiple of 512 bits length.

### EXAMPLE:

Plaintext size = 1200 bits

Padding bits = 272

Plaintext + Padding = 1200 + 272

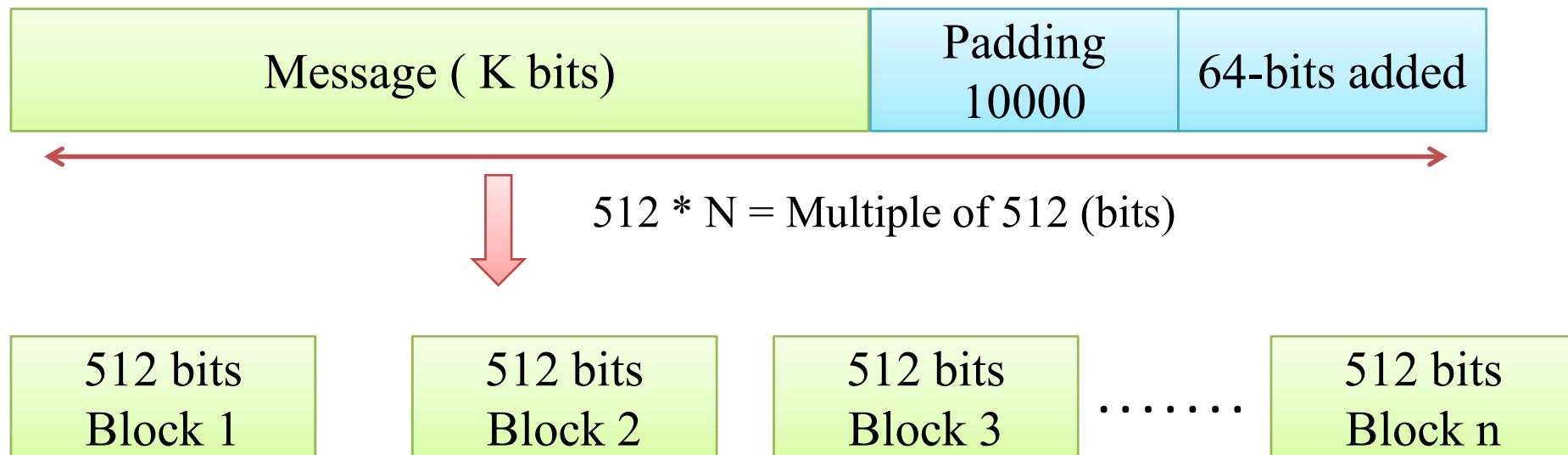
$$= 1472$$



# MD 5

**Steps to follow:**

- 1. Append Padding Bits**
- 2. Append Length**



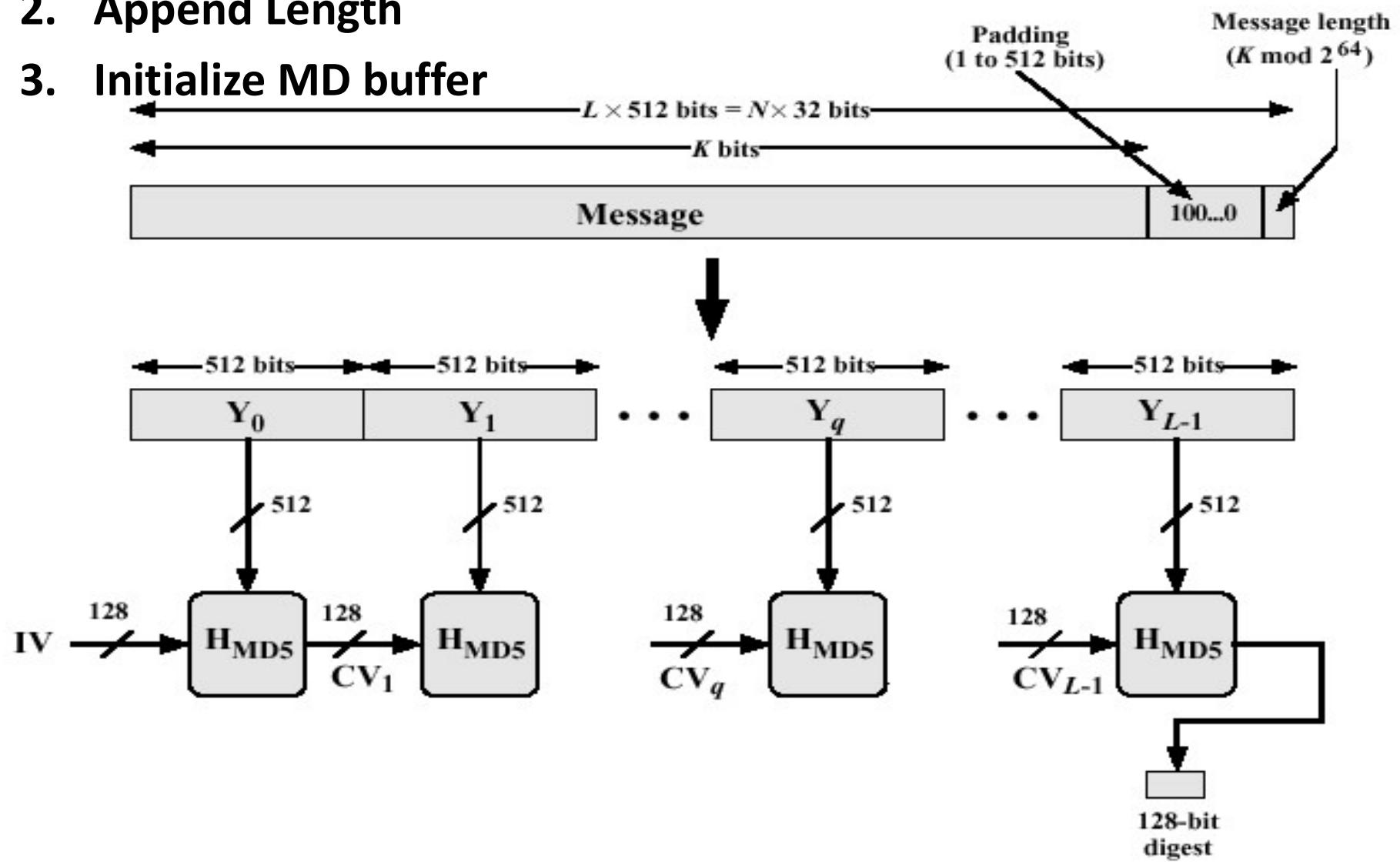
## Steps to follow:

MD 5

1. Append Padding Bits

2. Append Length

3. Initialize MD buffer



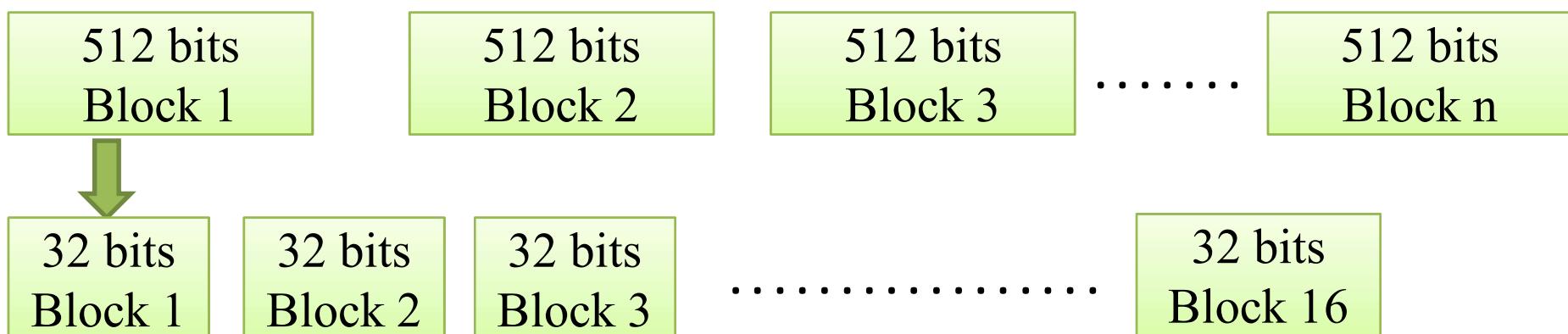
# MD 5

## Steps to follow:

### 3. Initialize MD buffer

- ✓ A 128-bit buffer is used to store intermediate and final result
- ✓ A buffer is represented as 4 32-bit register viz. A, B, C,D

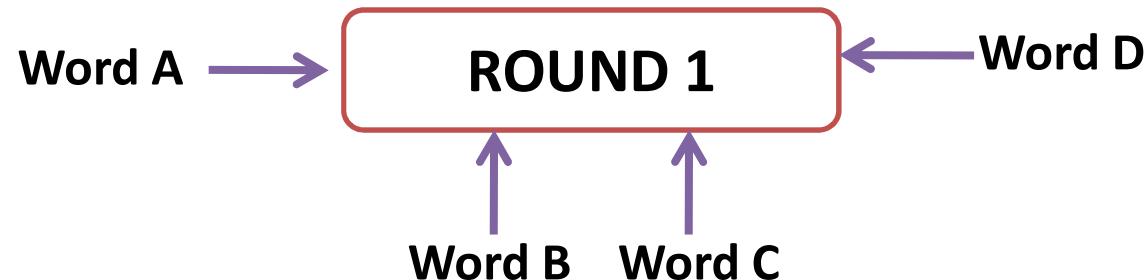
word A:	01	23	45	67
word B:	89	ab	cd	ef
word C:	fe	dc	ba	98
word D:	76	54	32	10



# MD 5

**Steps to follow:**

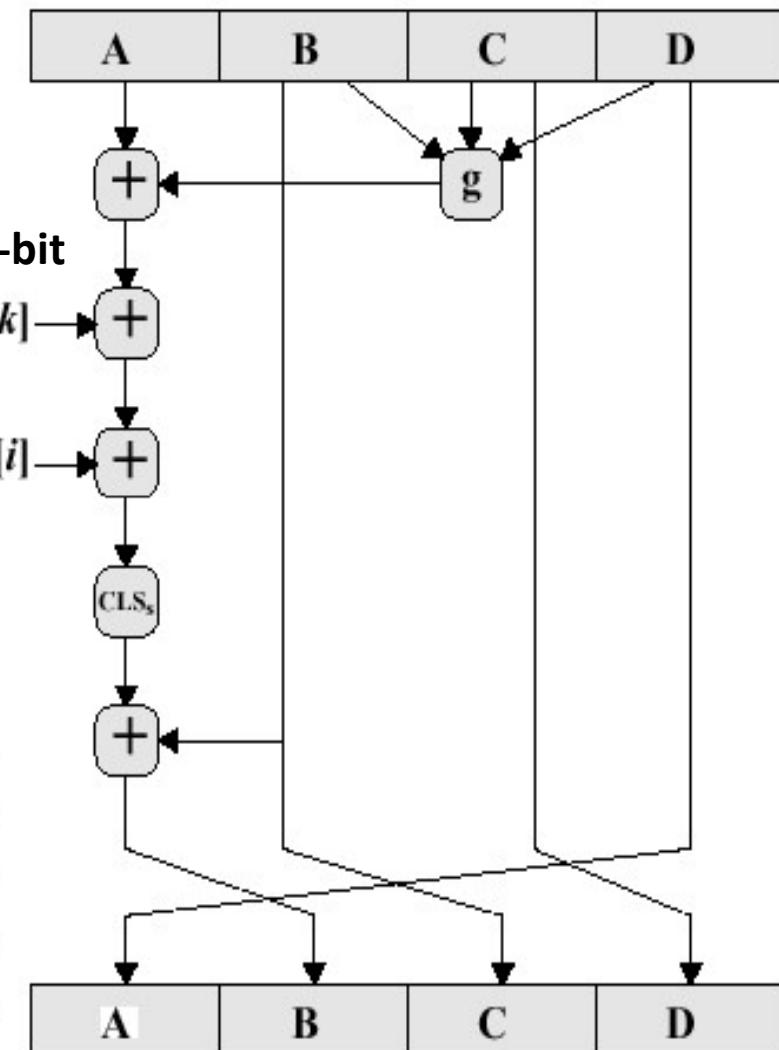
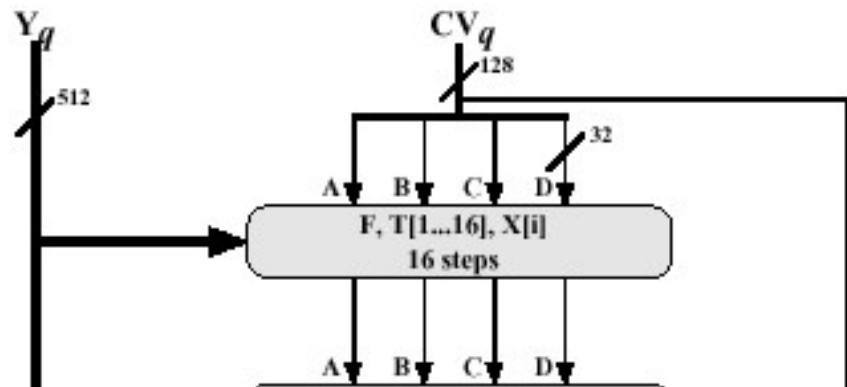
## 4. Processing message in 16-word block



Steps to follow:

**MD 5**

#### 4. Processing message in 16-word block



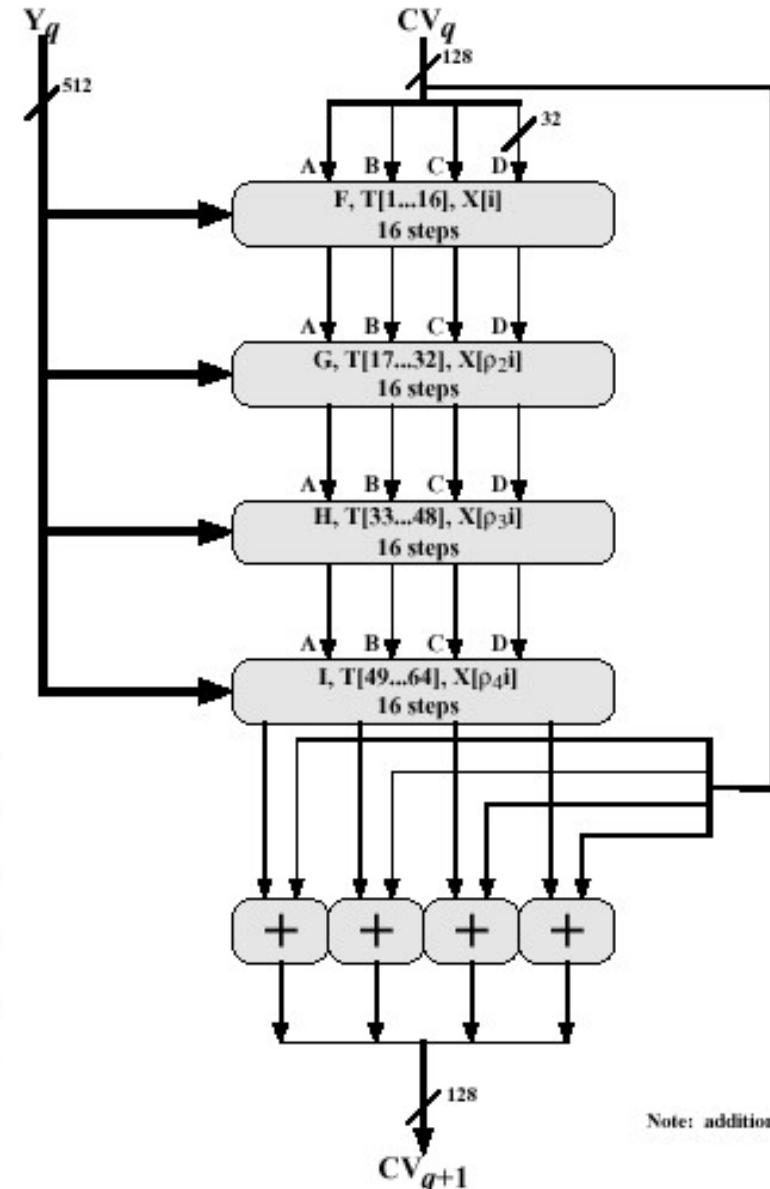
Round	Primitive function $g$	$g(b,c,d)$
1	$F(b,c,d)$	$(b \wedge c) \vee (\neg b \wedge d)$
2	$G(b,c,d)$	$(b \wedge d) \vee (c \wedge \neg d)$
3	$H(b,c,d)$	$b \oplus c \oplus d$
4	$I(b,c,d)$	$c \oplus (b \vee \neg d)$



## Steps to follow:

**MD 5**

### 4. Processing message in 16-word block



Round	Primitive function $g$	$g(b,c,d)$
1	$F(b,c,d)$	$(b \wedge c) \vee (\neg b \wedge d)$
2	$G(b,c,d)$	$(b \wedge d) \vee (c \wedge \neg d)$
3	$H(b,c,d)$	$b \oplus c \oplus d$
4	$I(b,c,d)$	$c \oplus (b \vee \neg d)$



# Table T, consturcted from the sine function

---

T[1] = D76AA478	T[17] = F61E2562	T[33] = FFFA3942	T[49] = F4292244
T[2] = E8C7B756	T[18] = C040B340	T[34] = 8771F681	T[50] = 432AFF97
T[3] = 242070DB	T[19] = 265E5A51	T[35] = 699D6122	T[51] = AB9423A7
T[4] = C1BDCEEE	T[20] = E9B6C7AA	T[36] = FDE5380C	T[52] = FC93A039
T[5] = F57COFAF	T[21] = D62F105D	T[37] = A4BEEA44	T[53] = 655B59C3
T[6] = 4787C62A	T[22] = 02441453	T[38] = 4BDECFA9	T[54] = 8F0CCC92
T[7] = A8304613	T[23] = D8A1E681	T[39] = F6BB4B60	T[55] = FFEFF47D
T[8] = FD469501	T[24] = E7D3FBC8	T[40] = BEBFBC70	T[56] = 85845DD1
T[9] = 698098D8	T[25] = 21E1CDE6	T[41] = 289B7EC6	T[57] = 6FA87E4F
T[10] = 8B44F7AF	T[26] = C33707D6	T[42] = EAA127FA	T[58] = FE2CE6E0
T[11] = FFFF5BB1	T[27] = F4D50D87	T[43] = D4EF3085	T[59] = A3014314
T[12] = 895CD7BE	T[28] = 455A14ED	T[44] = 04881D05	T[60] = 4E0811A1
T[13] = 6B901122	T[29] = A9E3E905	T[45] = D9D4D039	T[61] = F7537E82
T[14] = FD987193	T[30] = FCEFA3F8	T[46] = E6DB99E5	T[62] = BD3AF235
T[15] = A679438E	T[31] = 676F02D9	T[47] = 1FA27CF8	T[63] = 2AD7D2BB
T[16] = 49B40821	T[32] = 8D2A4C8A	T[48] = C4AC5665	T[64] = EB86D391

---

# Various Cryptographic Hash Functions

---

- **Message Digest (MD)**
- **Secure Hash Function (SHA)**
  - ✓ The Secure Hash Algorithm (SHA) is the most widely used hash function.
  - ✓ Indeed, because virtually every other widely used hash function had been found to have substantial cryptanalytic weaknesses.
  - ✓ SHA was more or less the last remaining standardized hash algorithm by 2005.
  - ✓ SHA was developed by the National Institute of Standards and Technology (NIST)
  - ✓ When weaknesses were discovered in SHA, now known as SHA-0, a revised version was issued as FIPS 180-1 in 1995 and is referred to as SHA-1.



# Versions of SHA

---

## Secure Hash Function (SHA)

- ✓ SHA-1 produces a hash value of 160 bits.
- ✓ Later, three new versions of SHA, with hash value lengths of 256, 384, and 512 bits, known as SHA-256, SHA-384, and SHA-512, respectively.

Collectively, these hash algorithms are known as SHA-2.

Algorithm	Message Size	Block Size	Word Size	Message Digest Size
SHA-1	$< 2^{64}$	512	32	160
SHA-224	$< 2^{64}$	512	32	224
SHA-256	$< 2^{64}$	512	32	256
SHA-384	$< 2^{128}$	1024	64	384
SHA-512	$< 2^{128}$	1024	64	512
SHA-512/224	$< 2^{128}$	1024	64	224
SHA-512/256	$< 2^{128}$	1024	64	256

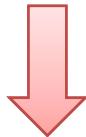
*Note:* All sizes are measured in bits.



# SHA-512

---

- Plaintext size block size – 1024 bits
- No. of Rounds/Steps – 80
- In each round a constant “k” is used
- A 512-bit buffer is used to hold intermediate and final results



8 Buffers of 64-bit each

- Size of Hash Value – 512 bits

## Steps to follow:

1. Append padding bits.
2. Append length
3. Initialize hash buffer.
4. Process message in 1024-bit (128-byte) blocks.



# **SHA 512**

---

## **Steps to follow:**

- 1. Append Padding Bits**
- 2. Append Length**

After padding, **128 bits** are inserted at the end which is used to record the length of the original input

- 3. Initialize hash buffer**
- 4. Processing message in 1024-bit block**

## **Step 1: APPPEND PADDING BITS**

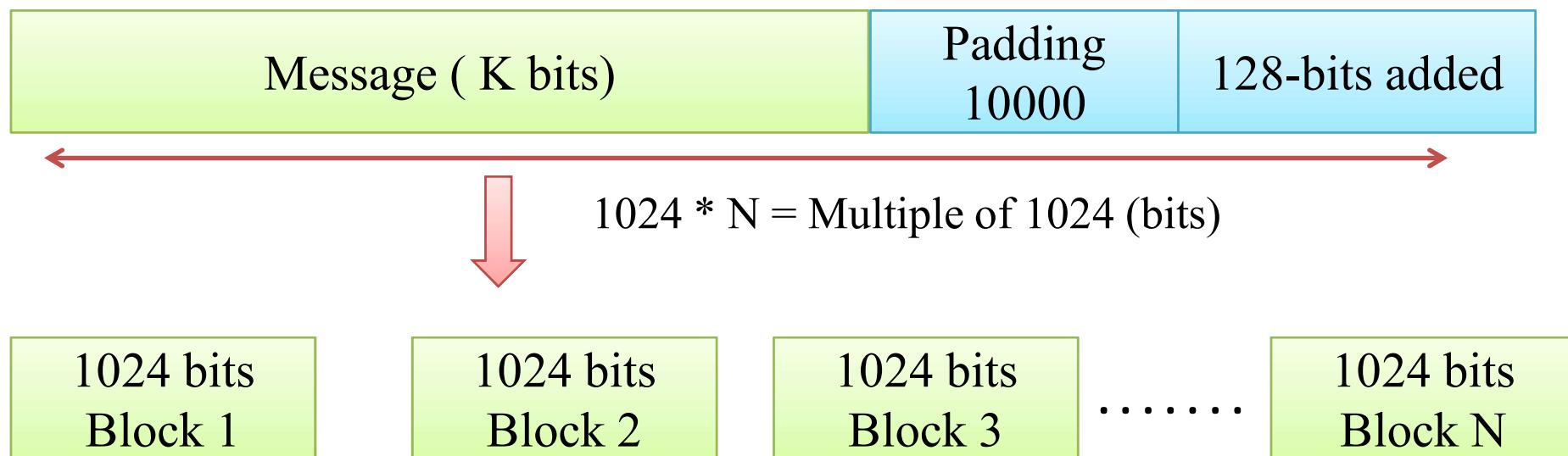
Padding means adding extra bits to the original message. Padding is done such that the total bits are 128 less than a multiple of 1024 bits length.



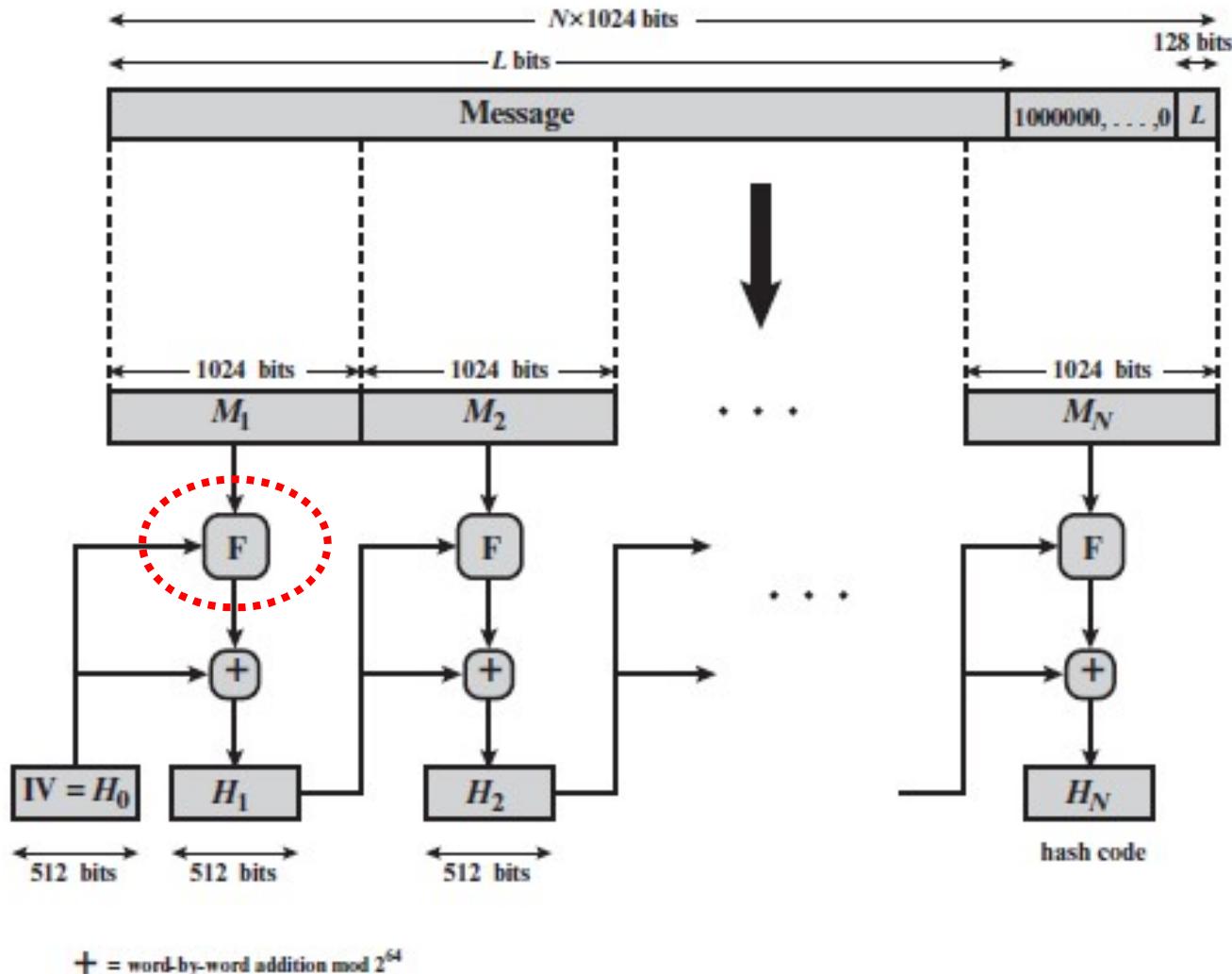
# SHA 512

## Steps to follow:

1. Append Padding Bits
2. Append Length



# SHA 512



# SHA 512

---

## Steps to follow:

### 3. Initialize hash buffer

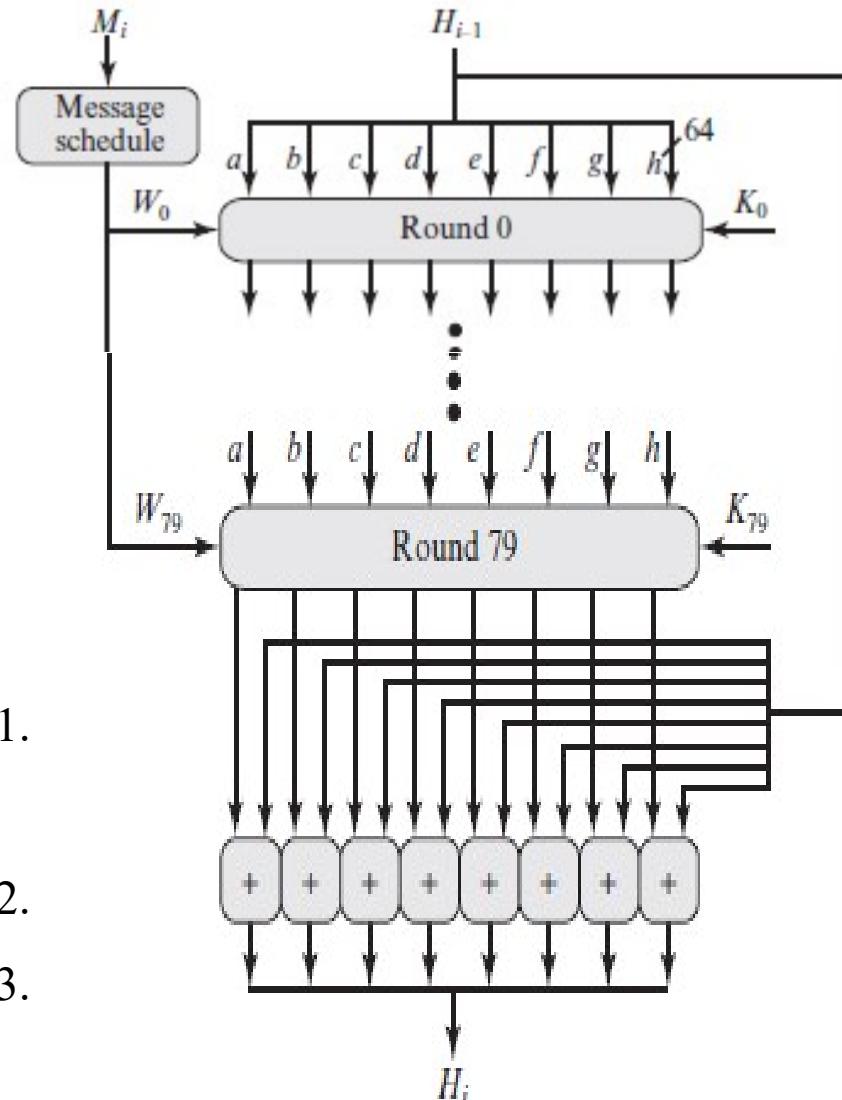
- ✓ A 512-bit buffer is used to hold intermediate and final results of the hash function.
- ✓ The buffer can be represented as eight 64-bit registers (a, b, c, d, e, f, g, h).
- ✓ These registers are initialized to the following 64-bit integers (hexadecimal values)

### 4. Process message in 1024-bit (128-byte) blocks.

- ✓ The heart of the algorithm is a module that consists of 80 rounds.
- ✓ This module is labeled **F**

These words were obtained by taking the first sixty-four bits of the fractional parts of the square roots of the first eight prime numbers.

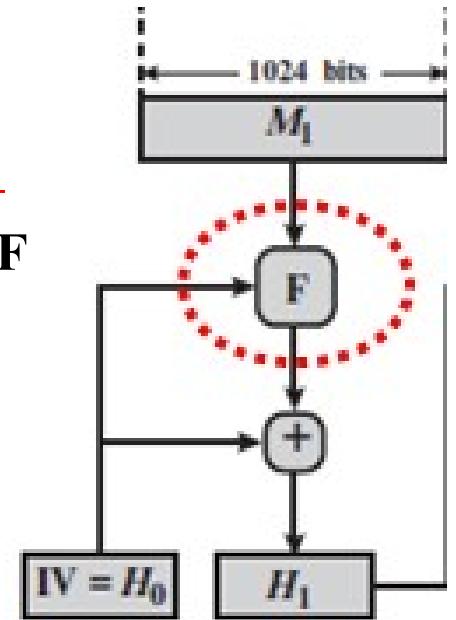




1. Iffer value, [a b c d e f g h], and updates the
2. The output of the eightieth round is added to the
3. input to the first round ( $H_{i-1}$ ) to produce  $H_i$ .
4. he value of the intermediate hash value,  $H_{i-1}$ .
5. e  $W_t$ , derived from the current 1024-bit block
6. Each round also makes use of an additive constant  $K_t$ , where  $0 \dots t \dots 79$  indicates one
7. of the 80 rounds.

# SHA 512

## Logic of Module F



# SHA 512

## 5. Output

After all  $N$  1024-bit blocks have been processed, the output from the  $N^{th}$  stage is the 512-bit **MESSAGE DIGEST**.

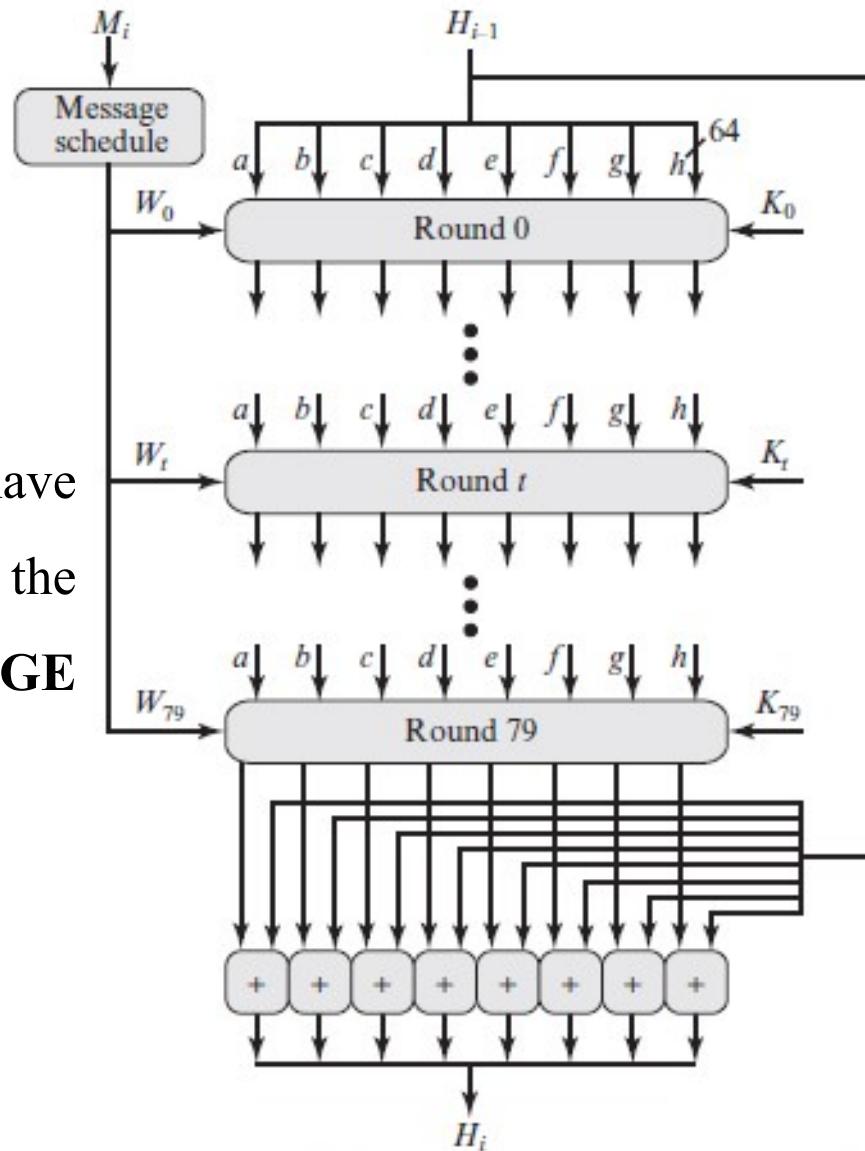


Figure 11.10 SHA-512 Processing of a Single 1024-Bit Block



# SHA 512 – ROUND FUNCTION

Each round is defined by the following set of equations:

$$T_1 = h + \text{Ch}(e, f, g) + (\sum_0^{512} e) + W_t + K_t$$

$$T_2 = (\sum_0^{512} a) + \text{Maj}(a, b, c)$$

Where;

t = step number; 0 ... t ... 79

$$h = g$$

$$g = f$$

$$f = e$$

$$e = d + T1$$

$$d = c$$

$$c = b$$

$$b = a$$

$$a = T1 + T2$$

$\text{Ch}(e, f, g) = (e \text{ AND } f) \oplus (\text{NOT } e \text{ AND } g)$ .

The conditional function: If e then f else g

$\text{Maj}(a, b, c) = (a \text{ AND } b) \oplus (a \text{ AND } c) \oplus (b \text{ AND } c)$ : The function is true only if the majority (two or three) of the arguments are true

$$(\sum_0^{512} a) = \text{ROTR28}(a) \oplus \text{ROTR34}(a) \oplus \text{ROTR39}(a)$$

$$(\sum_0^{512} e) = \text{ROTR14}(e) \oplus \text{ROTR18}(e) \oplus \text{ROTR41}(e)$$

$\text{ROTR}^n(x)$  = circular right shift (rotation) of the 64-bit argument x by n bits



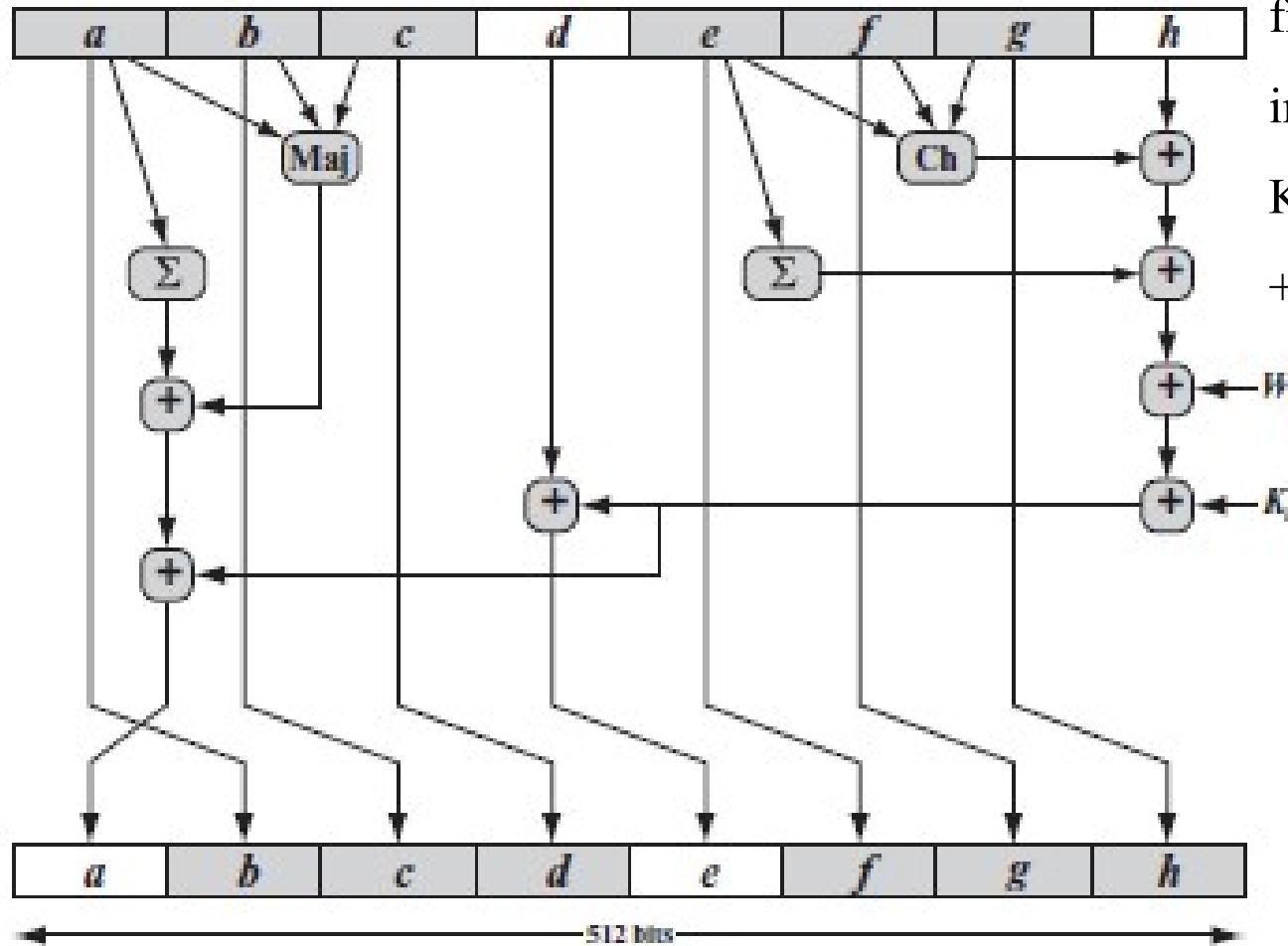
# SHA 512 – ROUND FUNCTION

Each round is defined by the following set of equations:

$W_t$  = a 64-bit word derived

from the current 1024-bit  
input block

$K_t$  = a 64-bit additive constant  
 $+$  = addition modulo  $2^{64}$



# SHA 512 – Deriving Word $W_t$

---

The first 16 values of  $W_t$  are taken directly from the 16 words of the current block.

$$W_t = \sigma_1^{512}(W_{t-2}) + W_{t-7} + \sigma_0^{512}(W_{t-15}) + W_{t-16}$$

where

$$\sigma_0^{512}(x) = \text{ROTR}^1(x) \oplus \text{ROTR}^8(x) \oplus \text{SHR}^7(x)$$

$$\sigma_1^{512}(x) = \text{ROTR}^{19}(x) \oplus \text{ROTR}^{61}(x) \oplus \text{SHR}^6(x)$$

$\text{ROTR}^n(x)$  = circular right shift (rotation) of the 64-bit argument  $x$  by  $n$  bits

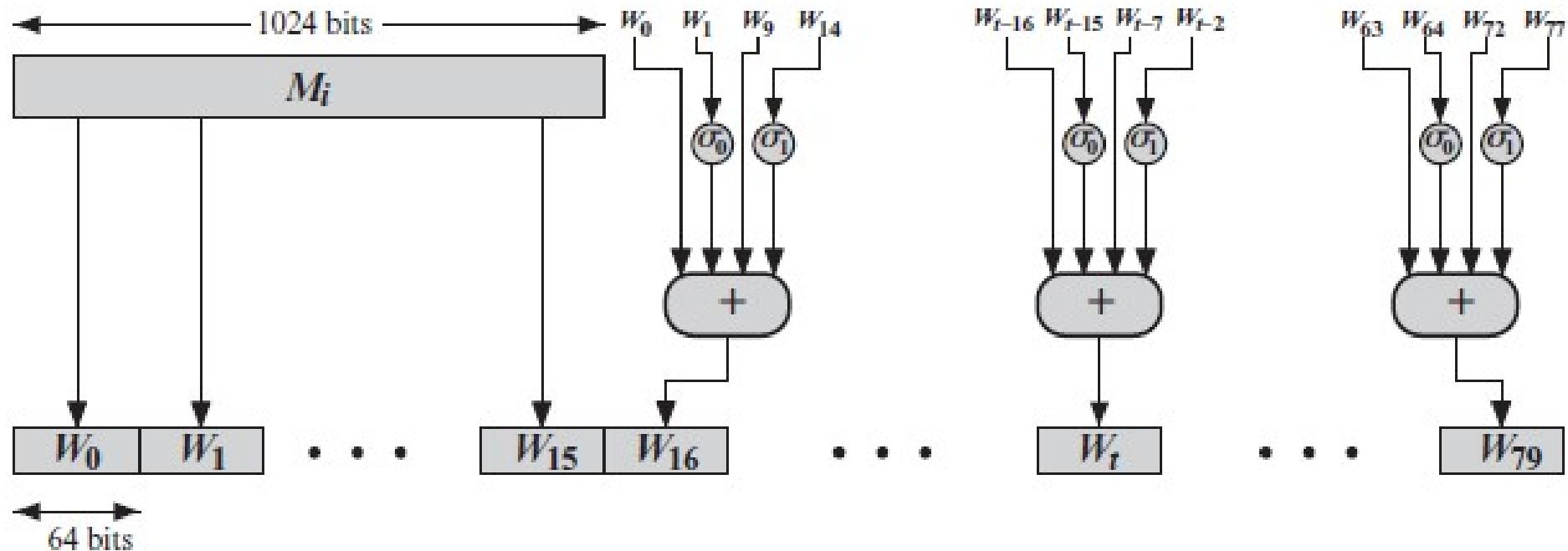
$\text{SHR}^n(x)$  = right shift of the 64-bit argument  $x$  by  $n$  bits with padding by zeros on the left

$+$  = addition modulo  $2^{64}$



# SHA 512 – Deriving Word $W_t$

The first 16 values of  $W_t$  are taken directly from the 16 words of the current block.



# Module 3:Hashes, Message Digests and Digital Certificates

---

## 3.1

- Cryptographic Hash Functions
- Properties of secure hash function
- Various Cryptographic Hash Functions:
  - ✓ MD5,
  - ✓ SHA-1,
  - ✓ MAC, HMAC, CMAC



# Message Authentication

1. Is a mechanism or service used to verify the integrity of a message
2. Message Authentication assures that data received are exactly as sent (i.e no Modification, Insertion, Deletion or Replay).
3. In many cases, there is a requirement that the authentication mechanism **ASSURES** that the purported IDENTITY of the SENDER IS VALID.

1. Message Authentication mechanism has 2 levels of functionality.
  - a) At lower level, there must be some sort of function that produces an authenticator → **A value to be used to Authenticate a message**
  - b) This is then used as a primitive in a Higher-Level authentication protocol that enables a receiver to verify the authenticity of a message



# **Message Authentication Code (MAC)**

---

## **FUNCTIONS USED TO PRODUCE AN AUTHENTICATOR**

1. HASH FUNCTION
2. MESSAGE ENCRYPTION
3. MESSAGE AUTHENTICATION CODE (MAC)

A function of the message and a secret key that produces a **FIXED-LENGTH** value as the authenticator



# Message Authentication Code (MAC)

---

1. Involves the use of a secret key to generate a small-fixed size block of data, known as **CRYPTOGRAPHIC CHECKSUM or MAC**, that is appended to the message.
2. This technique assumes that two communicating parties, say A and B, share a common secret key K.
3. When A has a message to send to B, it calculates the MAC as a function of the message and the key:

$$\mathbf{MAC} = \mathbf{C}(K, M)$$

where

$M$  = *input message*

C = MAC function

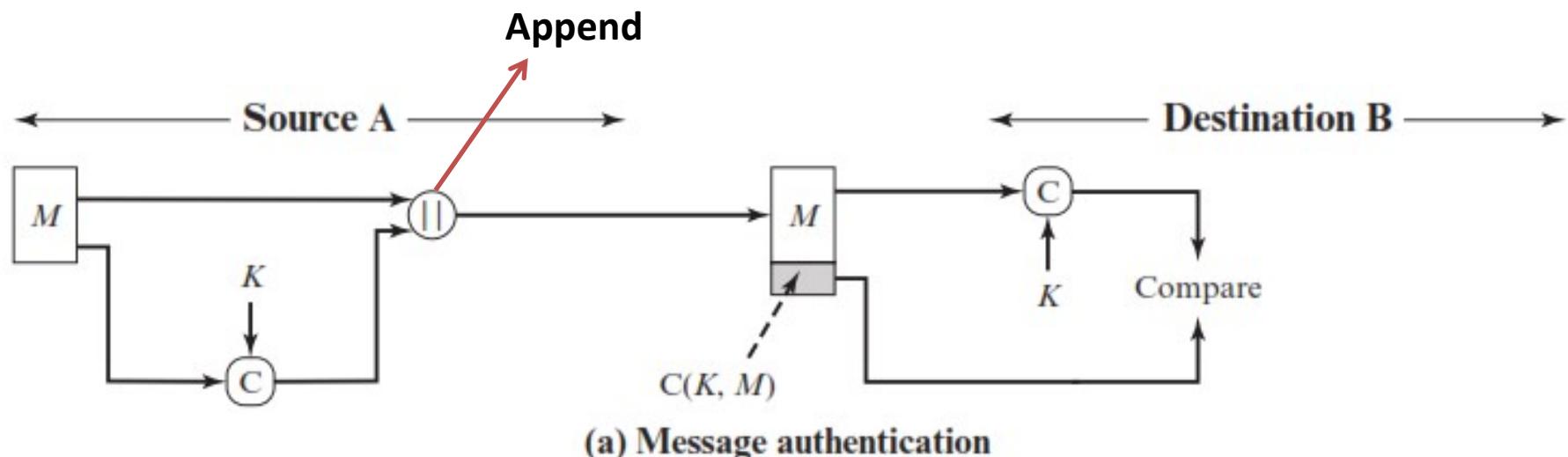
$K$  = *shared secret key*

MAC = message authentication code



# Message Authentication Code (MAC)

1. The message plus MAC are transmitted to the intended recipient.
2. The recipient performs the same calculation on the received message, using the same secret key, to generate a new MAC.
3. The received MAC is compared to the calculated MAC



$$\text{MAC} = C(K, M)$$



# Message Authentication Code (MAC)

---

**Assumption:** The receiver and the sender know the identity of the secret key.

If the received MAC matches the calculated MAC, then

- 1. The receiver is assured that the message has not been altered.**

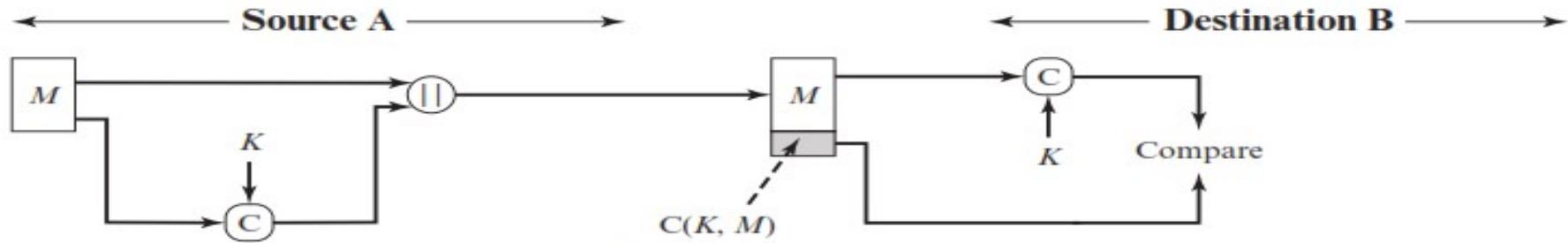
If an attacker alters the message but does not alter the MAC, then the receiver's calculation of The MAC will differ from the received MAC. Because the attacker is assumed not to know the secret key, the attacker cannot alter the MAC to correspond to the alterations in the message.

- 2. The receiver is assured that the message is from the alleged sender.**

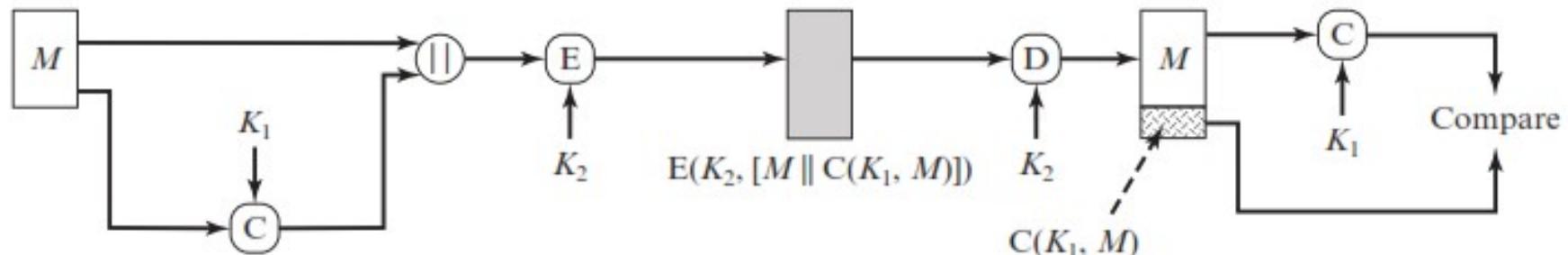
Because no one else knows the secret key, no one else could prepare a message with a proper MAC.



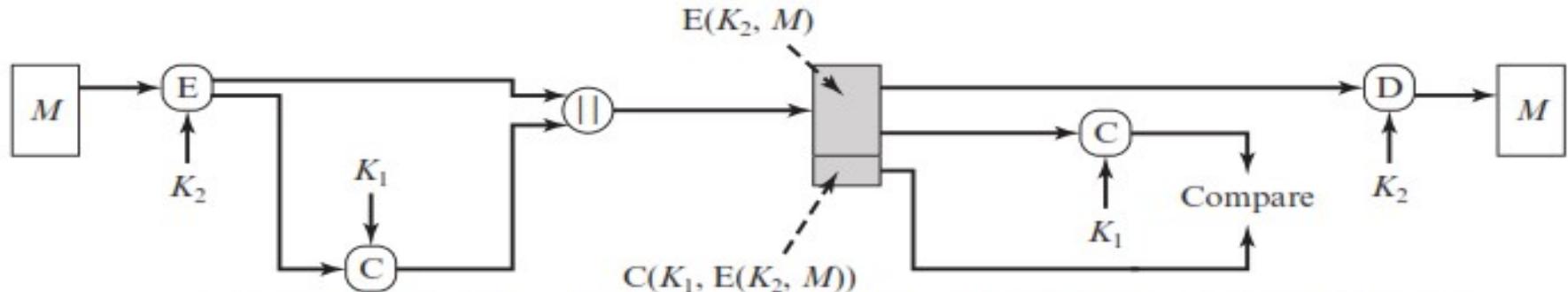
# Message Authentication Code (MAC)



(a) Message authentication



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



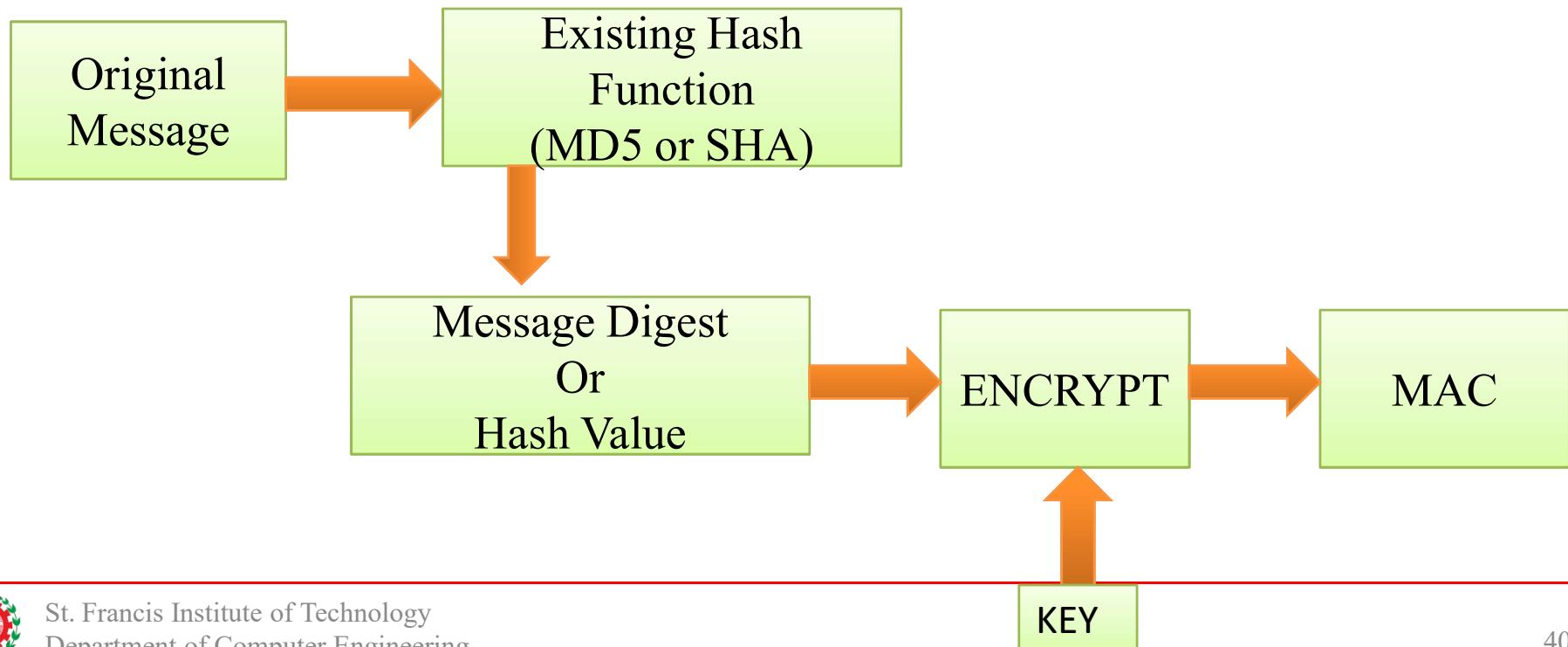
(c) Message authentication and confidentiality; authentication tied to ciphertext



# MAC Based on Hash Function: HMAC

---

1. HMAC stands for HASH Message Authentication Code
2. It is a specific technique for calculating a Message Authentication Code (MAC) involving a combination of cryptographic hash function and a secret key cryptography.



# HMAC Design Objectives

---

The following are design objectives for HMA

- a) To use, without modifications, available hash functions.
- b) To allow for easy replaceability of the embedded hash function in case faster or more secure hash functions are found or required.
- c) To preserve the original performance of the hash function without incurring significant degradation
- d) To use and handle keys in a simple way.
- e) To have a well understood cryptographic analysis of the strength of the authentication mechanism based on reasonable assumptions about the embedded hash function.



# HMAC ALGORITHM

---

Terms used in 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 =  $i^{\text{th}}$  block of M,  $0 \leq i \leq (L - 1)$

L = number of blocks in M

b = number of bits in a block



# HMAC ALGORITHM

---

Terms used in HMAC Algorithm:

$n$  = 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

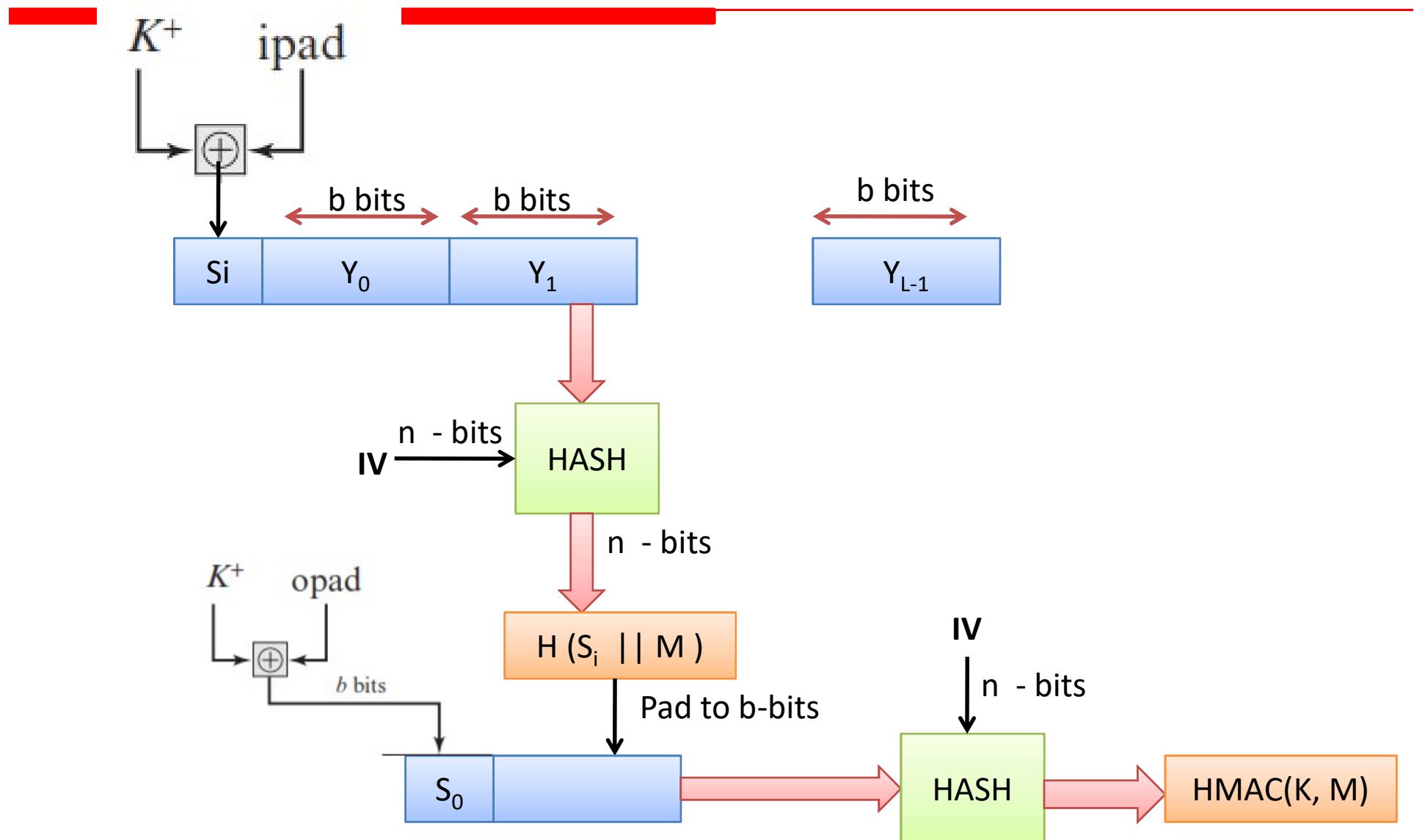
$opad$  = 01011100 (5C in hexadecimal) repeated  $b/8$  times

HMAC can be expressed as

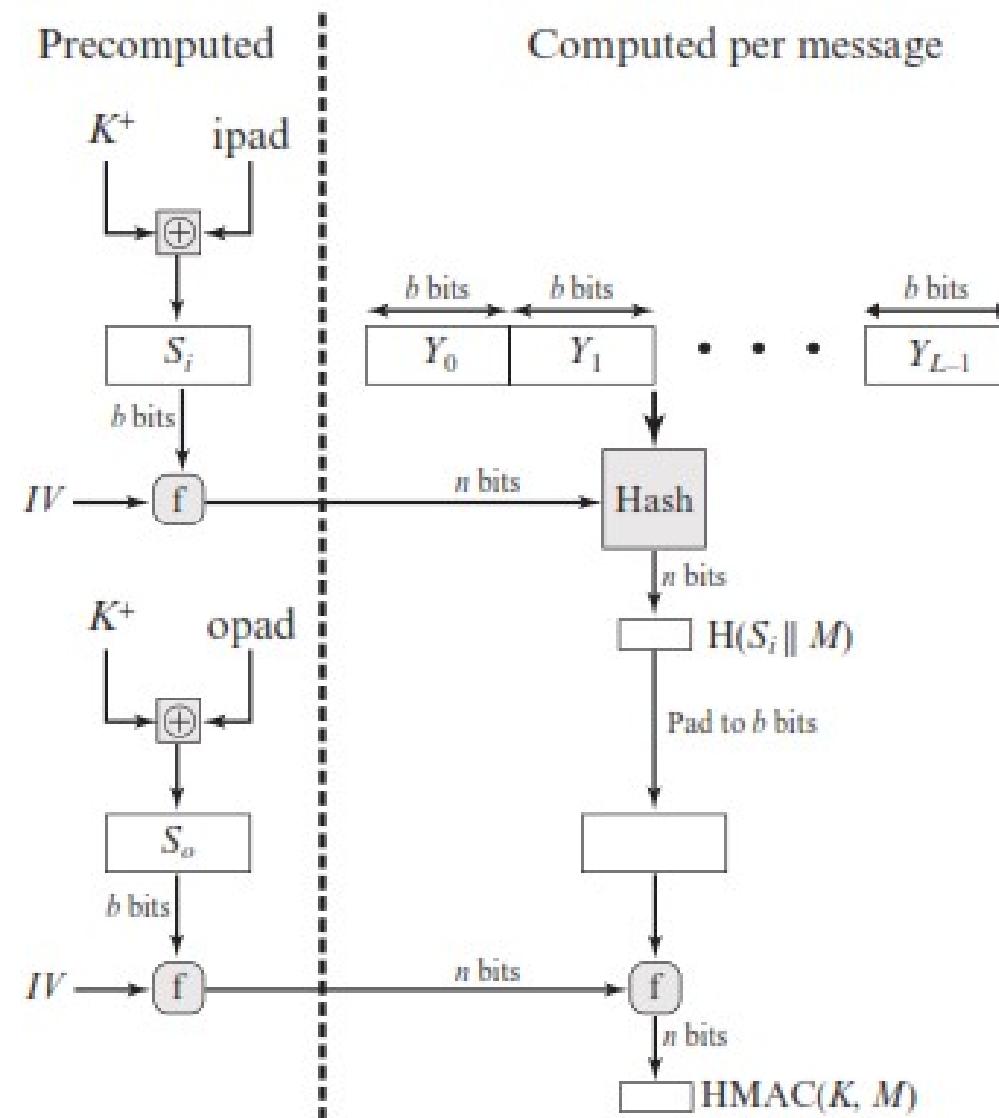
$$\text{HMAC}(K, M) = H[(K^+ \oplus opad) \parallel H[(K^+ \oplus ipad) \parallel M]]$$



# HMAC STRUCTURE



# HMAC STRUCTURE



# CMAC : Cipher based Message Authentication Code

1. Cipher-based message authentication codes (or CMACs) are a tool for calculating message authentication codes using a block cipher coupled with a secret key.
2. CMAC is used to verify both the integrity and authenticity of a message.
3. It is a mode of operation for use with AES and Triple DES

## Operation of CMAC:

- For AES, the cipher block length,  $b = 128$  bits
- For Triple DES, the cipher block length,  $b = 64$  bits
- The message is divided into  $n$  blocks ( $M_1, M_2, M_3, \dots, M_n$ )
- $k$ -bit encryption key  $K$  is used **Triple DES key size is 112 or 168 bits**  
**AES key size is 128, 192, 256 bits**
- A constant  $K_1$  is used which is length  $b$ -bits

Message  
is integer  
multiple  
of cipher  
block  
length



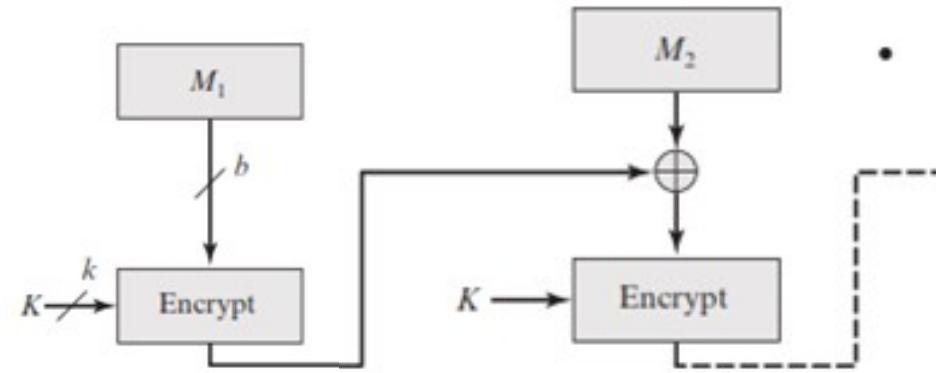
# CMAC : Cipher based Message Authentication Code

CMAC is calculated as follows:

$$C_1 = E(K, M_1)$$

$$C_2 = E(K, [M_2 \oplus C_1])$$

$$C_3 = E(K, [M_3 \oplus C_2])$$



$$C_n = E(K, [M_n \oplus C_{n-1} \oplus \dots \oplus C_1 \oplus K])$$

$$T = \text{MSB}_{T\text{len}}(C_n)$$

Where,

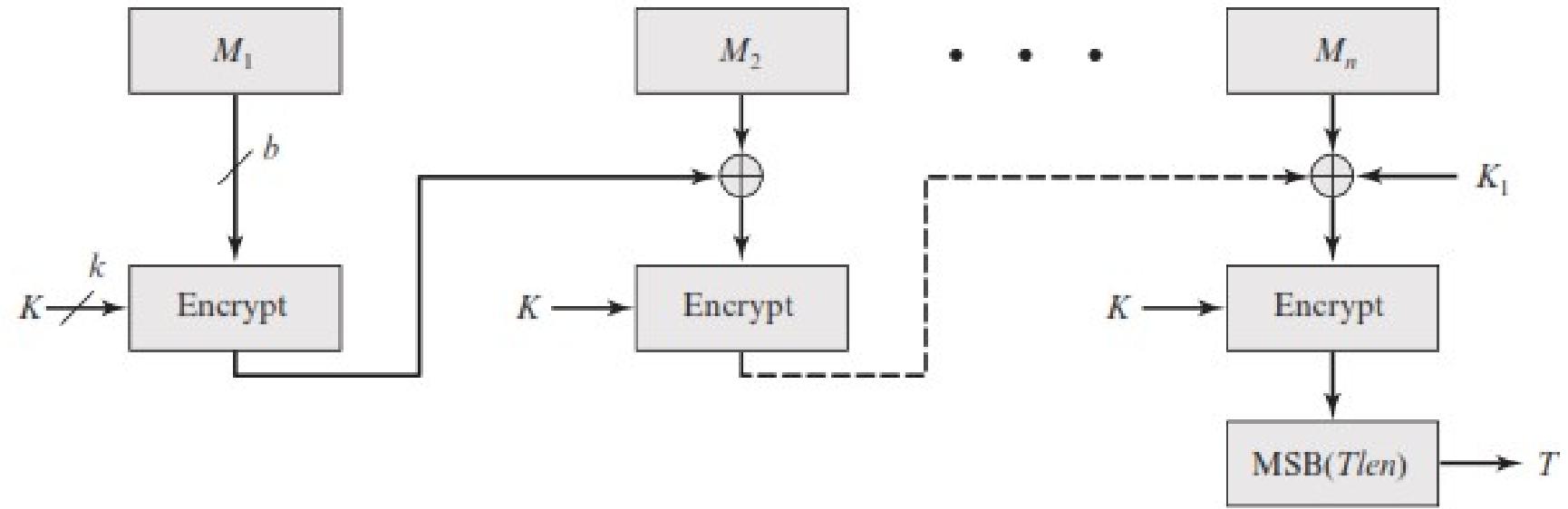
T = MAC, also referred to as the TAG.

Tlen = bit length of T

MSBs(X) = the s leftmost bits of the bit string X



# CMAC : Cipher based Message Authentication Code



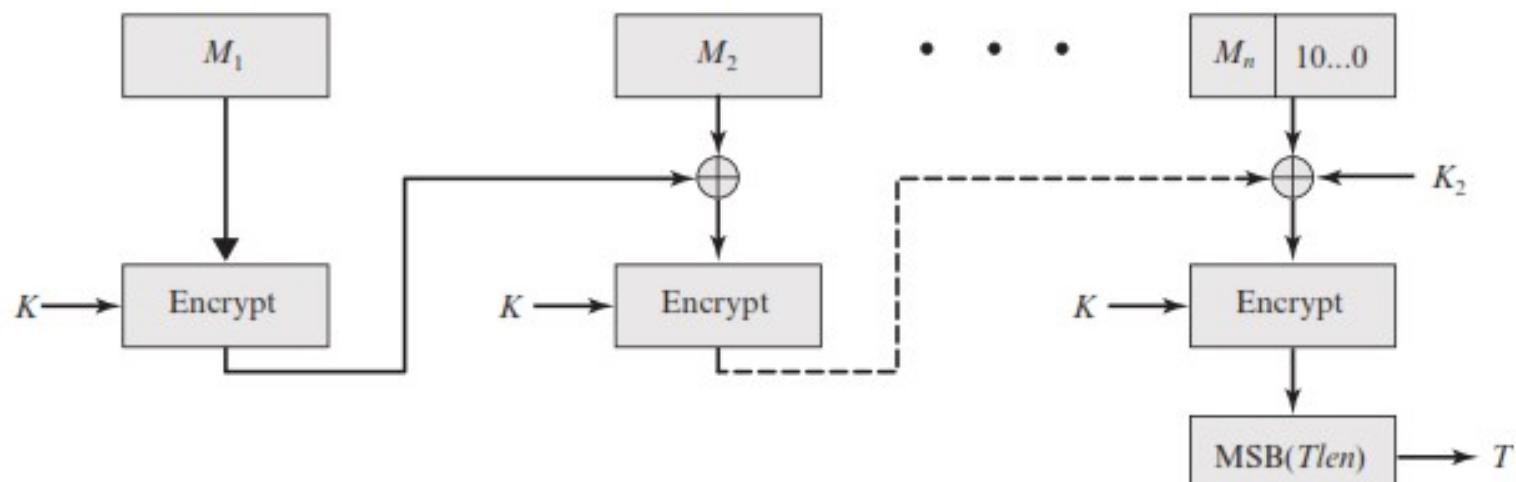
(a) Message length is integer multiple of block size



# CMAC : Cipher based Message Authentication Code

*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 proceeds as before, except that a different b-bit key  $K_2$  is used instead of  $K_1$*



(b) Message length is not integer multiple of block size



# CMAC : Cipher based Message Authentication Code

---

The two b-bit keys are derived from the k-bit encryption key as follows:

$$L = E(K, 0^b)$$

$$K_1 = L \cdot x$$

$$K_2 = L \cdot x^2 = (L \cdot x) \cdot x$$

1. To generate K1 and K2 , the block cipher is applied to the block that consists entirely of 0 bits.
2. The first subkey is derived from the resulting ciphertext by a left shift of one bit and, conditionally, by XORing a constant that depends on the block size.
3. The second subkey is derived in the same manner from the first subkey.



# Module 3:Hashes, Message Digests and Digital Certificates

---

## 3.1

- Cryptographic Hash Functions
- Properties of secure hash function
- Various Cryptographic Hash Functions:
  - ✓ MD5,
  - ✓ SHA-1,
  - ✓ MAC, HMAC, CMAC

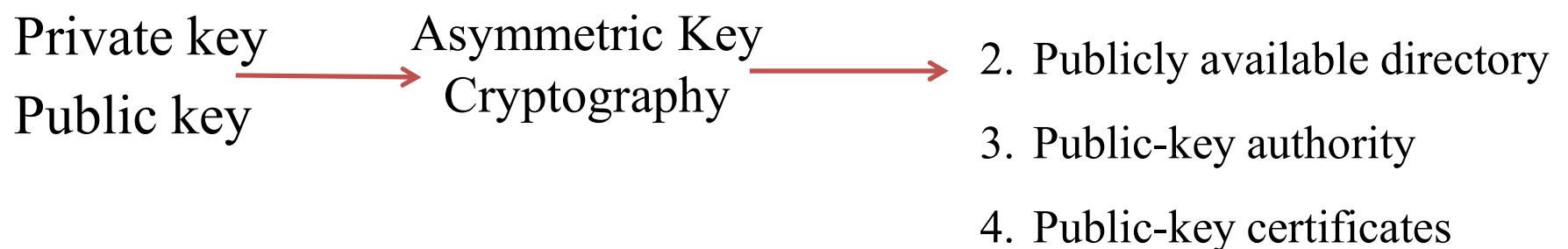
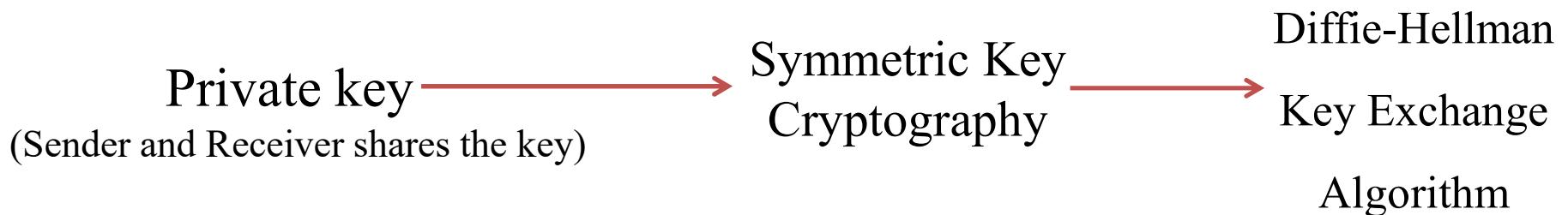
## 3.2

- Digital Certificate
- X.509, PKI



# Distribution of Public Key

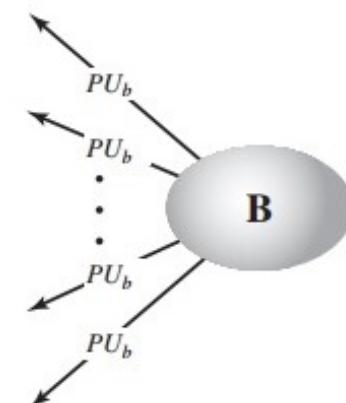
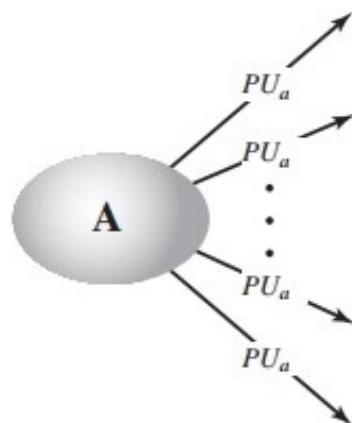
---



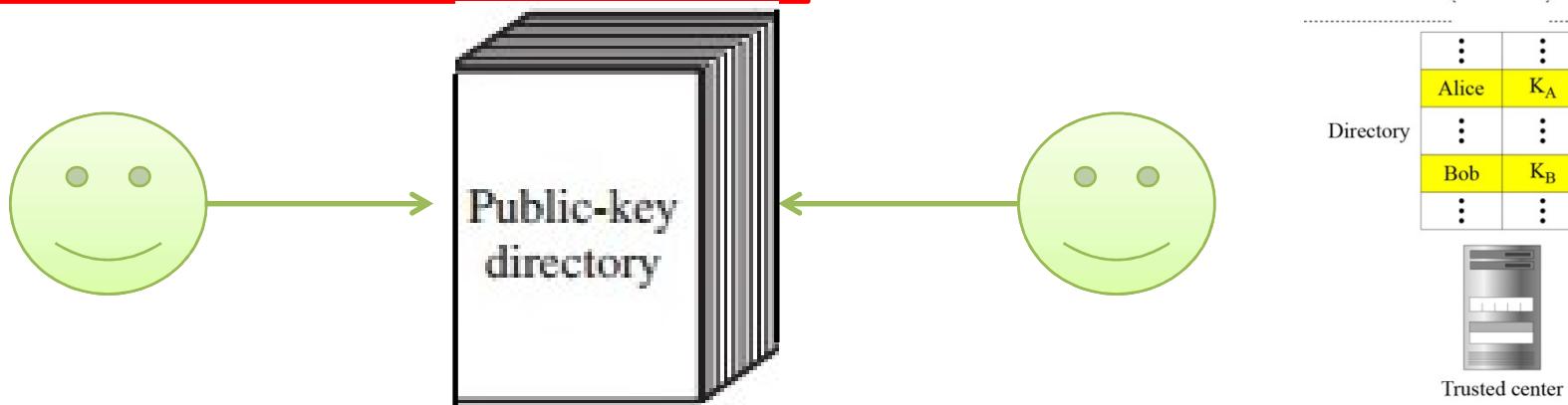
# 1. Public announcement

---

- Any participant can send his or her public key to any other participant or broadcast the key to the community at large.
- Although this approach is convenient, it has a major weakness.
- Anyone can forge such a public announcement.  
(That is, some user could pretend to be user A and send a public key to another participant or broadcast such a public key. Until such time as user A discovers the forgery and alerts other participants, the forger is able to read all encrypted messages intended for A and can use the forged keys for authentication)



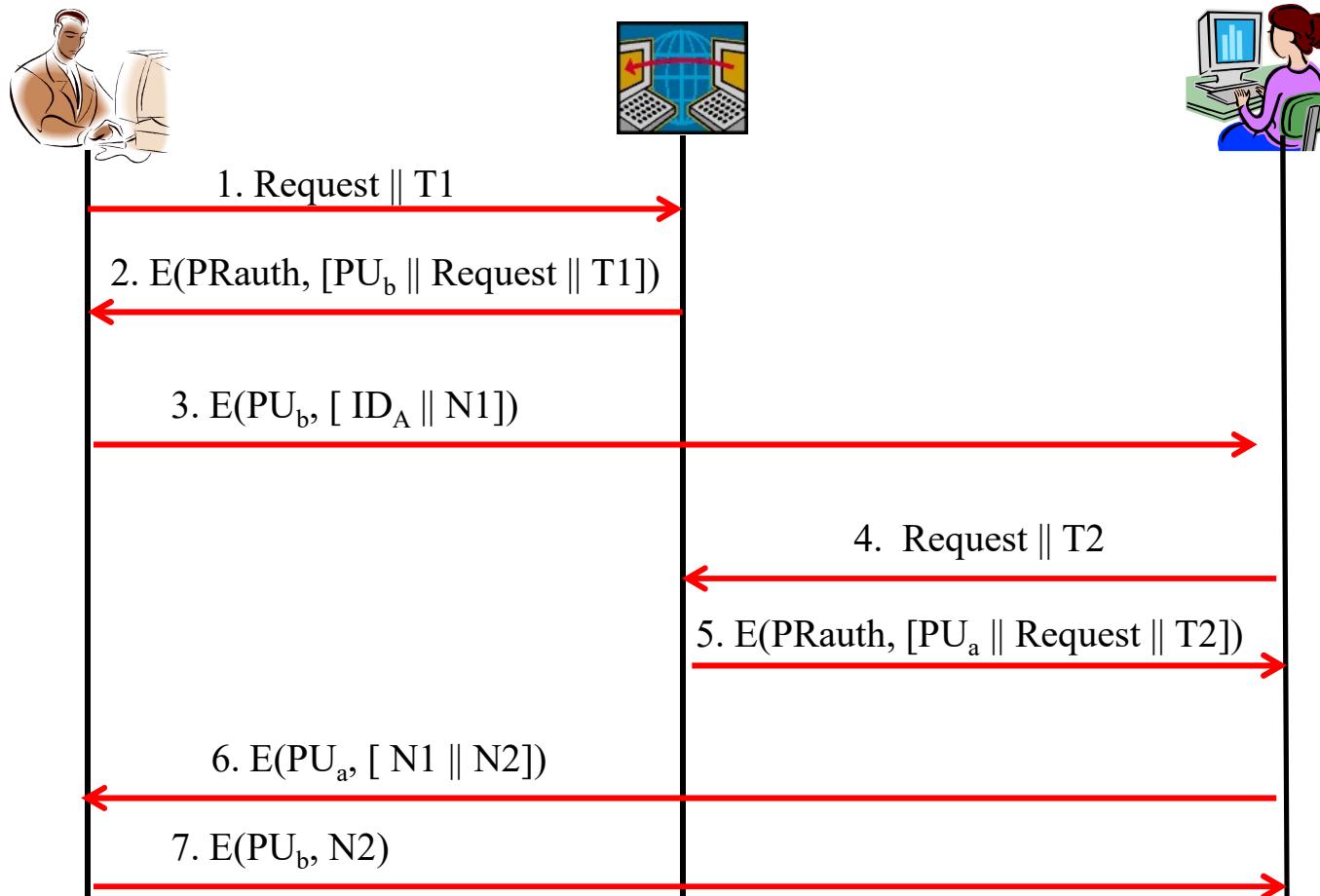
## 2. Publicly Available Directory



1. The authority maintains a directory with a {name, public key} entry for each participant.
2. Each participant registers a public key with the directory authority. Registration would have to be in person or by some form of secure authenticated communication.
3. A participant may replace the existing key with a new one at any time,
  - ✓ A public key has already been used for a large amount of data,
  - ✓ The corresponding private key has been compromised in some way.
4. Participants could also access the directory electronically. For this purpose, secure, authenticated communication from the authority to the participant is mandatory.

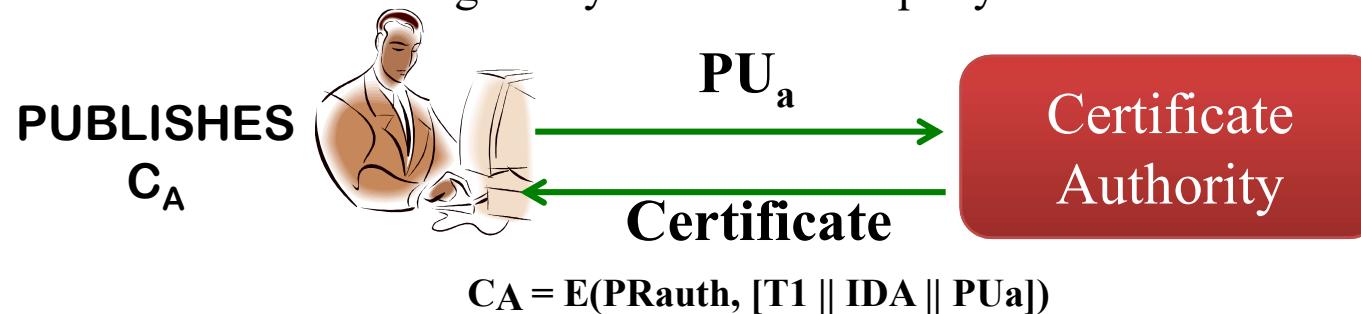


### 3. Public-Key Authority



# 4. Public-key Certificates

1. An alternative approach, first suggested by Kohnfelder is to use certificates that can be used by participants to exchange keys without contacting a public-key authority, in such a way that is as reliable as if the keys were obtained directly from a public-key authority.
2. A certificate consists of a
  - public key,
  - an identifier of the key owner,
  - The whole block signed by a trusted third party.



The third party is a certificate authority, such as a **GOVERNMENT AGENCY** or a **FINANCIAL INSTITUTION**, that is trusted by the user community.



# 4. Public-key Certificates

---

4. Anyone needing this user's public key can obtain the certificate and verify that it is valid by way of the attached trusted signature.
5. A participant can also convey its key information to another by transmitting its certificate.
6. Other participants can verify that the certificate was created by the authority.

## REQUIREMENTS

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



# 4. Public-key Certificates

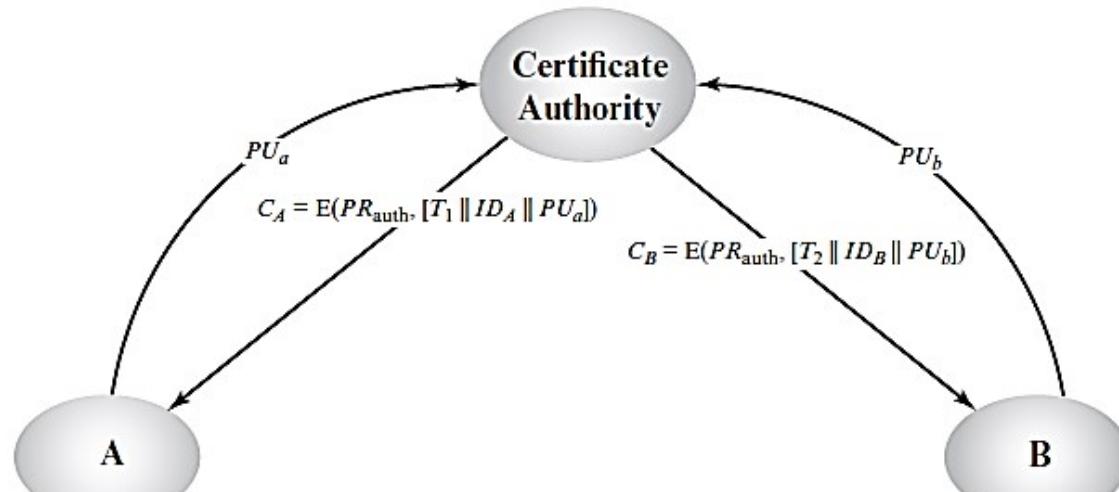
1. Each participant applies to the certificate authority, supplying a public key and requesting a certificate.
2. For participant A, the authority provides a certificate of the form

$$C_A = E(PR_{\text{auth}}, [T \parallel ID_A \parallel PU_a])$$

where,

$PR_{\text{auth}}$  is the private key used by the authority

T is a timestamp.



(a) Obtaining certificates from CA



# 4. Public-key Certificates

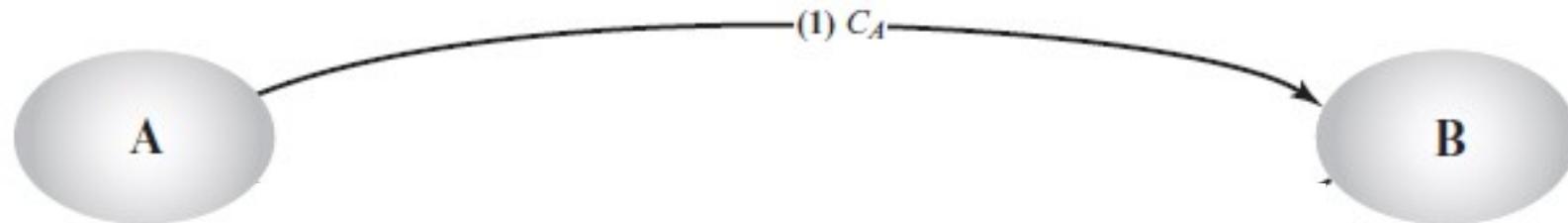
1. A may then pass this certificate on to any other participant, who reads and verifies the certificate as follows:

$$\begin{aligned} D(PU_{auth}, CA) &= D(PU_{auth}, E(PR_{auth}, [T \parallel ID_A \} PU_a])) \\ &= (T \parallel ID_A \parallel PU_a) \end{aligned}$$

The recipient uses the authority's public key,  $PU_{auth}$ , to decrypt the certificate.

The elements  $ID_A$  and  $PU_a$  provide the recipient with the name and public key of the certificate's holder.

The timestamp  $T$  validates the currency of the certificate.



One scheme has become universally accepted for formatting public-key certificates: the **X.509 standard**. These are used in most network security applications, including IP security, transport layer security (TLS)



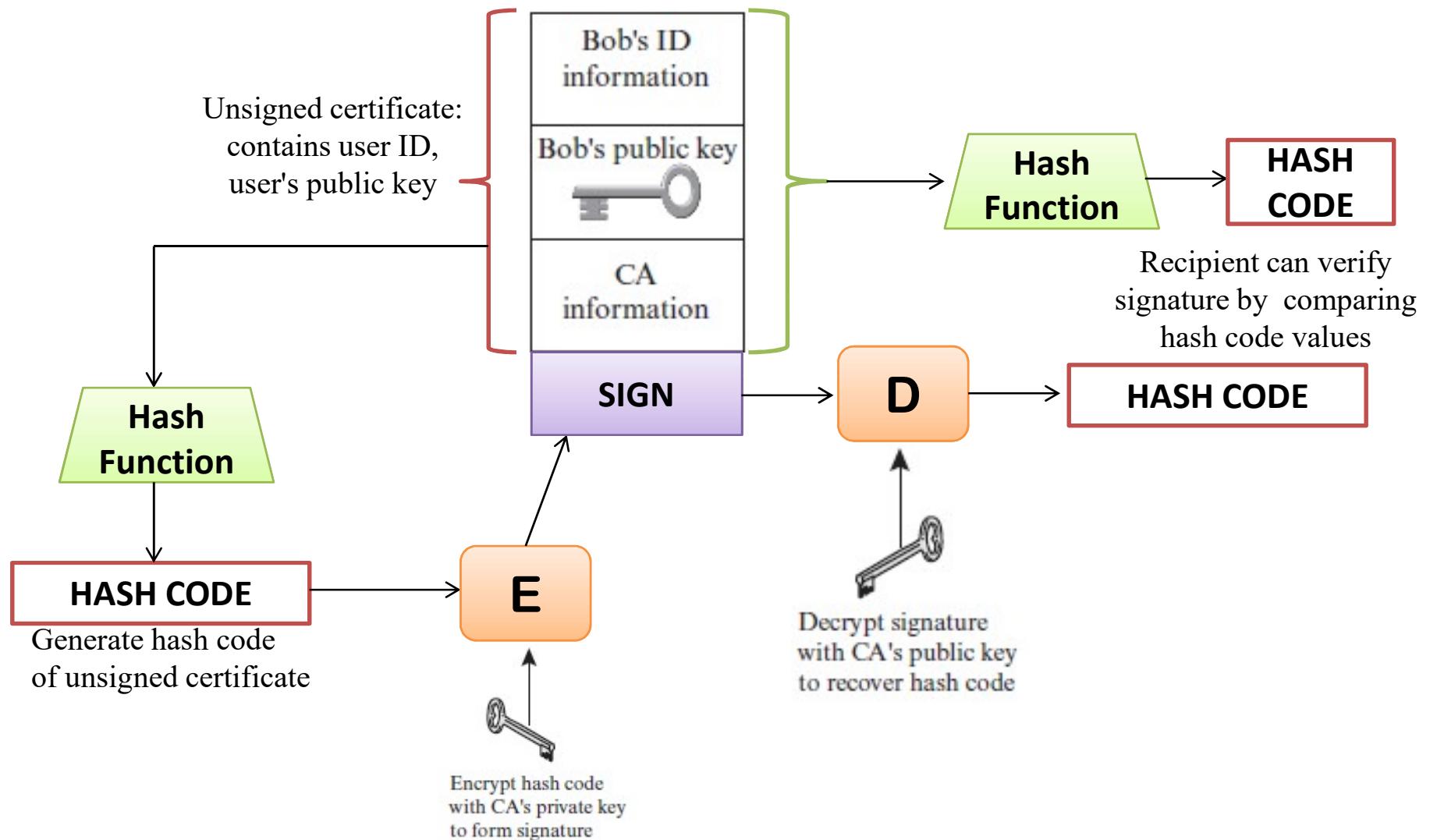
# X. 509 Certificates

---

1. X.509 was initially issued in 1988. The standard was subsequently revised in 1993 to address some of the security concerns. The standard is currently at version 7, issued in 2012.
2. X.509 is based on the use of public-key cryptography and digital signatures.
3. The standard does not dictate the use of a specific digital signature algorithm nor a specific hash function.



# X. 509 Certificates



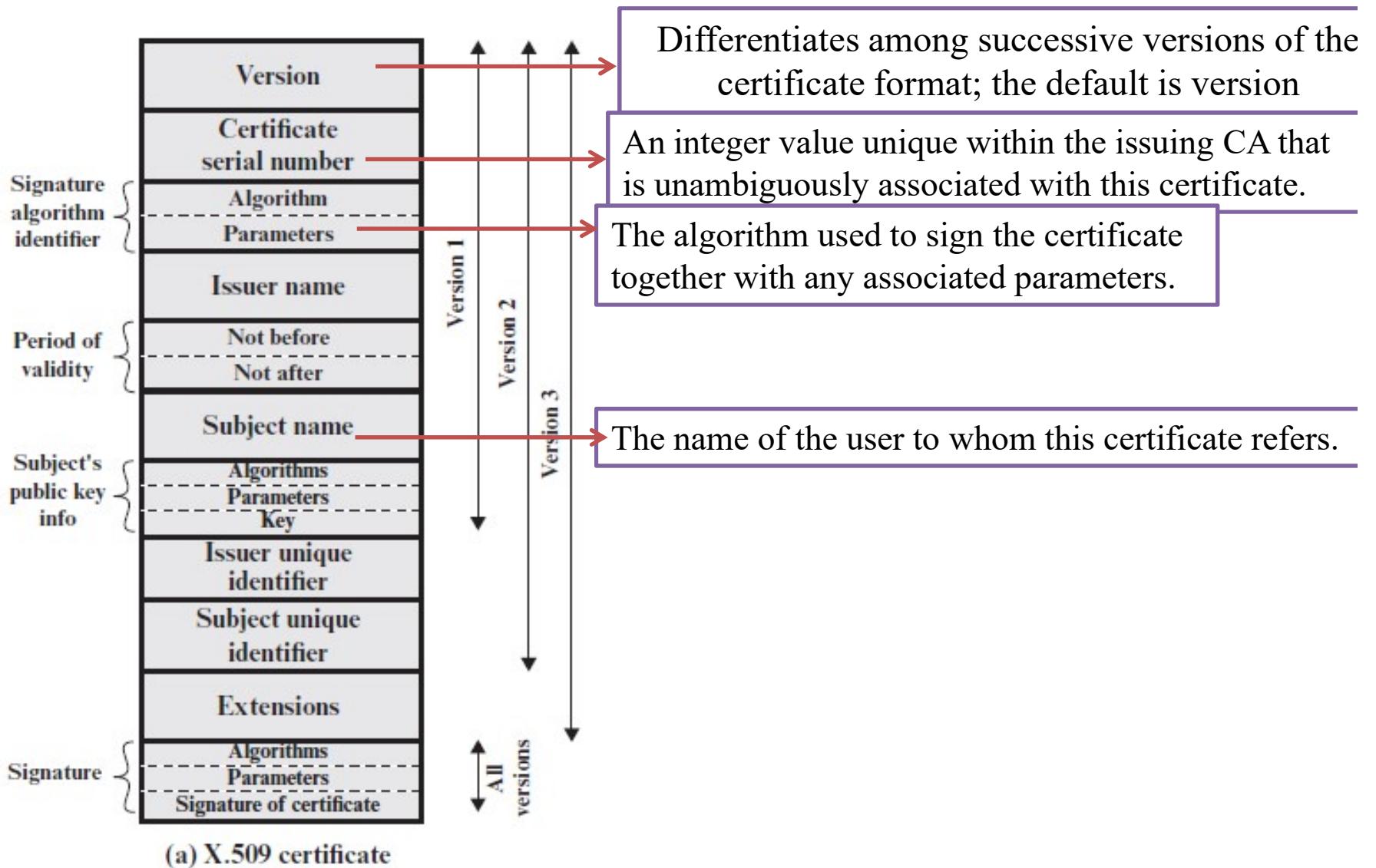
# X. 509 Certificates

---

1. The heart of the X.509 scheme is the public-key certificate associated with each user.
2. These user certificates are assumed to be created by some trusted certification authority (CA) and placed in the directory by the CA or by the user.
3. The directory server itself is not responsible for the creation of public keys or for the certification function.
4. It merely provides an easily accessible location for users to obtain certificates.



# X. 509 Certificates – General Format



# X. 509 Certificates – General Format

---

The standard uses the following notation to define a certificate:

$$\text{CA} \ll \text{A} \gg = \text{CA} \{ \text{V}, \text{SN}, \text{AI}, \text{CA}, \text{UCA}, \text{A}, \text{UA}, \text{Ap}, \text{TA} \}$$

where,

$\text{Y} \ll \text{X} \gg$  = the certificate of user X issued by certification authority Y

CA = name of certificate authority

V = version of the certificate

UCA = optional unique identifier of the CA

SN = serial number of the certificate

A = name of user A

AI = identifier of the algorithm used  
to sign the certificate

UA = optional unique identifier of the user A

CA = name of certificate authority

Ap = public key of user A

TA = period of validity of the certificate



# X. 509 Certificates – Revocation

---

1. Each certificate includes a period of validity, much like a credit card.
2. Typically, a new certificate is issued just before the expiration of the old one.
3. In addition, it may be desirable on occasion to revoke a certificate before it expires, for one of the following reasons.
  - The user's private key is assumed to be compromised.
  - The user is no longer certified by this CA. Reasons for this include that the subject's name has changed, the certificate is superseded, or the certificate was not issued in conformance with the CA's policies.
  - The CA's certificate is assumed to be compromised.

Each CA must maintain a list consisting of all revoked but not expired certificates issued by that CA, including both those issued to users and to other CAs. These lists should also be posted on the directory.



# PKI – Public Key Infrastructure

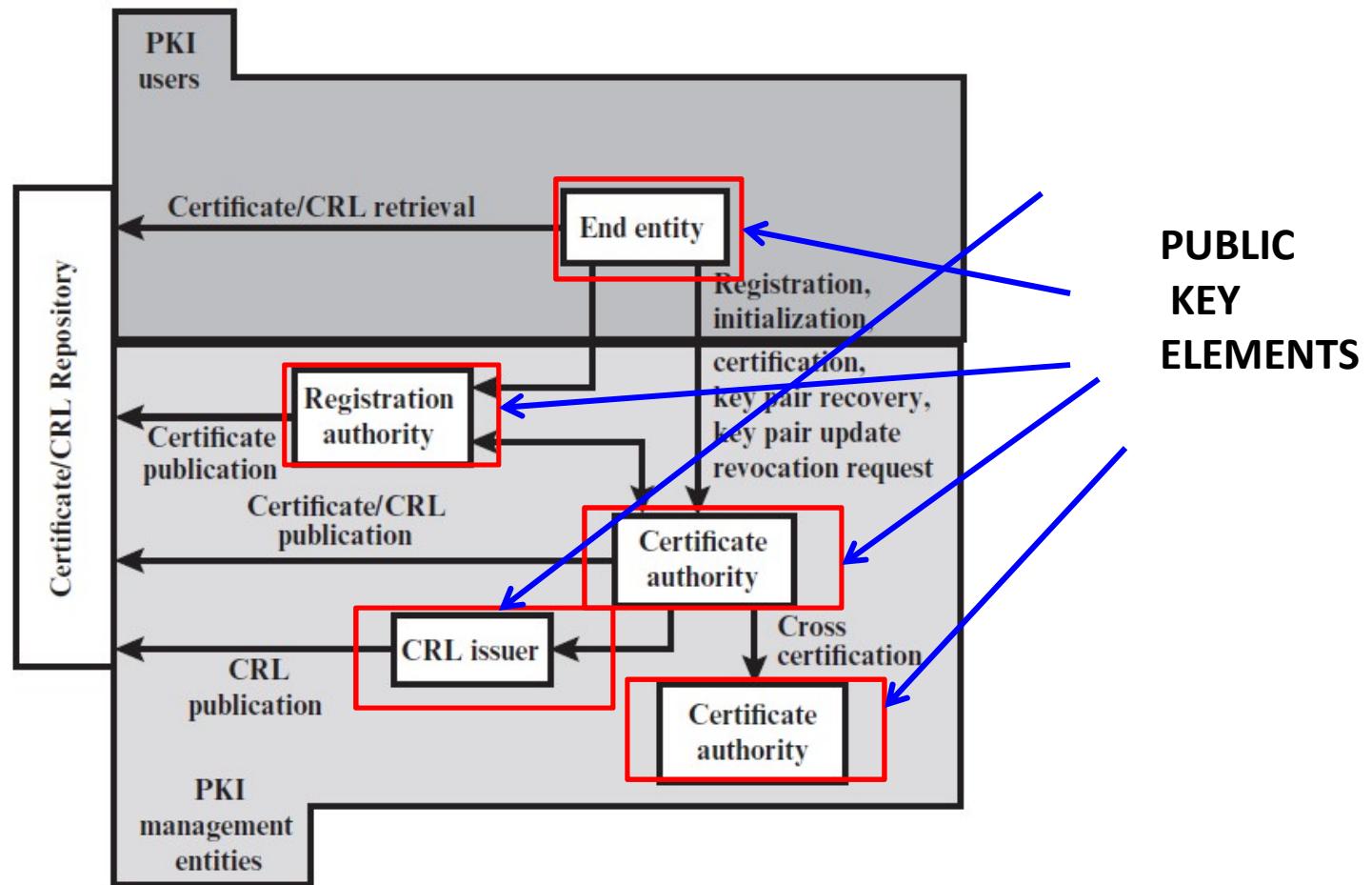
---

1. RFC 4949 (Internet Security Glossary) defines public-key infrastructure (PKI) as the set of hardware, software, people, policies, and procedures needed to create, manage, store, distribute, and revoke digital certificates based on asymmetric cryptography.
2. The principal objective for developing a PKI is to enable secure, convenient, and efficient acquisition of public keys.



# PKI – Public Key Infrastructure

The Formal Model of PKI based on X.509 that is suitable for deploying a certificate-based architecture on the Internet is Public Key Infrastructure X.509 (PKIX)



# PKI – Public Key Infrastructure

---

The key elements of the PKIX model are:

- **End entity:** A generic term used to denote end users, devices. End entities typically consume and/or support PKI related services.
- **Certification authority (CA):** The issuer of certificates and (usually) certificate revocation lists (CRLs).
- **Registration authority (RA):** An optional component that can assume a number of administrative functions from the CA. The RA is often associated with the end entity registration process but can assist in a number of other areas as well.
- **CRL issuer:** An optional component that a CA can delegate to publish CRLs.
- **Repository:** A generic term used to denote any method for storing certificates and CRLs so that they can be retrieved by end entities.



# PKIX – Management Functions

---

PKIX identifies a number of management functions that potentially need to be supported by management protocols.

- **Registration:** This is the process whereby a user first makes itself known to a CA (directly or through an RA), prior to that CA issuing a certificate or certificates for that user.
- **Initialization:** Before a client system can operate securely, it is necessary to install key materials that have the appropriate relationship with keys stored elsewhere in the infrastructure.
- **Certification:** This is the process in which a CA issues a certificate for a user's public key, returns that certificate to the user's client system, and/or posts that certificate in a repository.



# PKIX – Management Functions

---

- **Key pair recovery:** It is important to provide a mechanism to recover the necessary decryption keys when normal access to the keying material is no longer possible, otherwise it will not be possible to recover the encrypted data.
  - **Key pair update:** All key pairs need to be updated regularly (i.e., replaced with a new key pair) and new certificates issued. Update is required when the certificate lifetime expires and as a result of certificate revocation.
  - **Revocation request:** An authorized person advises a CA of an abnormal situation requiring certificate revocation. Reasons for revocation include private key compromise, change in affiliation, and name change.
  - **Cross certification:** Two CAs exchange information used in establishing a cross-certificate. A cross-certificate is a certificate issued by one CA to another CA that contains a CA signature key used for issuing certificates.
- 



# Cryptography and System Security (CSS)

## Course Code: CSC 604



### Subject Incharge

Ankita Karia  
Assistant Professor  
Room No. 421  
email: [ankitakaria@sfit.ac.in](mailto:ankitakaria@sfit.ac.in)



St. Francis Institute of Technology  
Department of Computer Engineering

08 March 2021

Cryptography and System Security  
Ms. Ankita Karia 1

# Module 4: Authentication Protocols & Digital signature schemes

---

## 4.1

- User Authentication - One-way and mutual
- Needham Schroeder Authentication protocol
- Kerberos Authentication protocol.

## 4.2

- Digital Signature Schemes
  - a) RSA
  - b) EIGamal
  - c) Schnorr



# Digital Signature

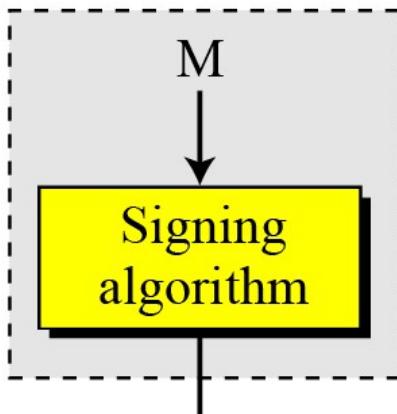
---

Conventional Signature	Digital Signature
A conventional signature is included in the document; it is part of the document.	When we sign a document digitally, we send the signature as a separate document.
When the recipient receives a document, she compares the signature on the document with the signature on file.	The recipient receives the message and the signature. The recipient needs to apply a verification technique to the combination of the message and the signature to verify the authenticity.
There is normally a one-to-many relationship between a signature and documents.	There is a one-to-one relationship between a signature and a message.
A copy of the signed document can be distinguished from the original one on file	There is no such distinction unless there is a factor of time on the document.



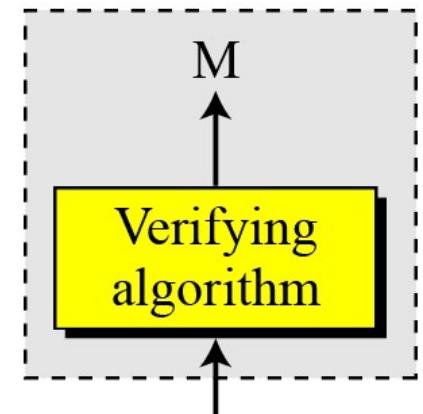
# Digital Signature Process

Alice



M: Message  
S: Signature

Bob

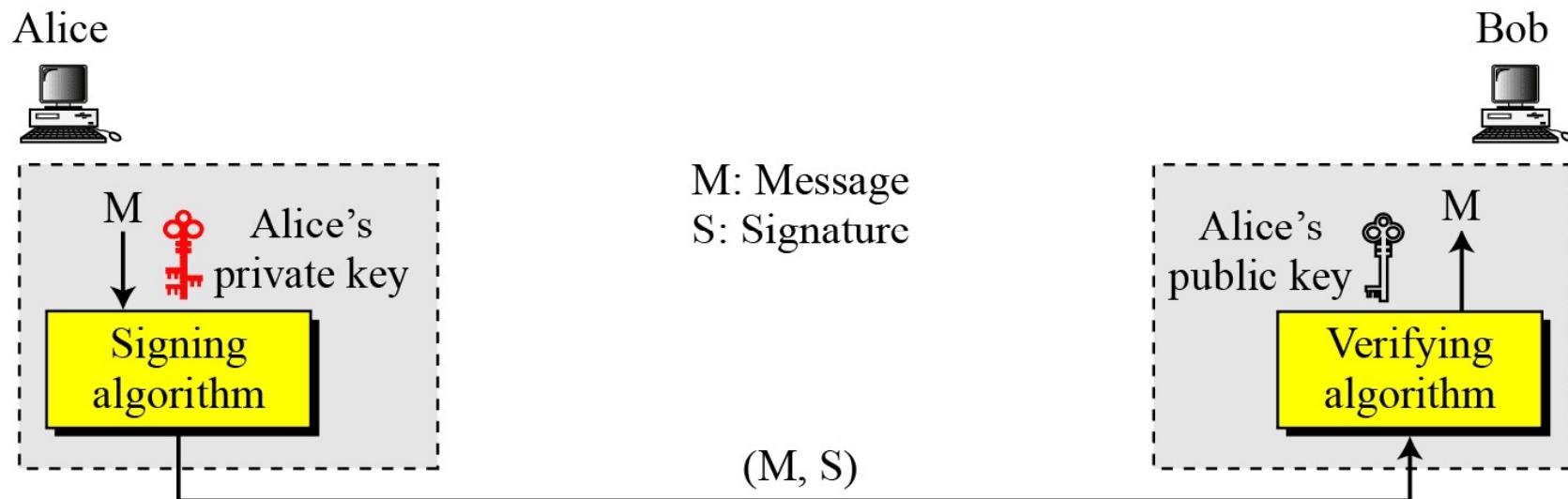


1. The sender uses a signing algorithm to sign the message.
2. The message and the signature are sent to the receiver.
3. The receiver receives the message and the signature and applies the verifying algorithm to the combination.
4. If the result is true, the message is accepted; otherwise, it is rejected.



# Digital Signature – Need for keys

1. The signer uses her Private key, applied to a signing algorithm, to sign the document.
2. The verifier uses the public key of the signer, applied to the verifying algorithm, to verify the document



When a document is signed, anyone receiving the signed document can verify it because everyone has access to sender's public key

Sender must not use her public key to sign the document, because then anyone could forge her signature.



# Digital Signature – Need for keys

---

**Note**

A digital signature needs a public-key system.  
The signer signs with her private key; the verifier  
verifies with the signer's public key.

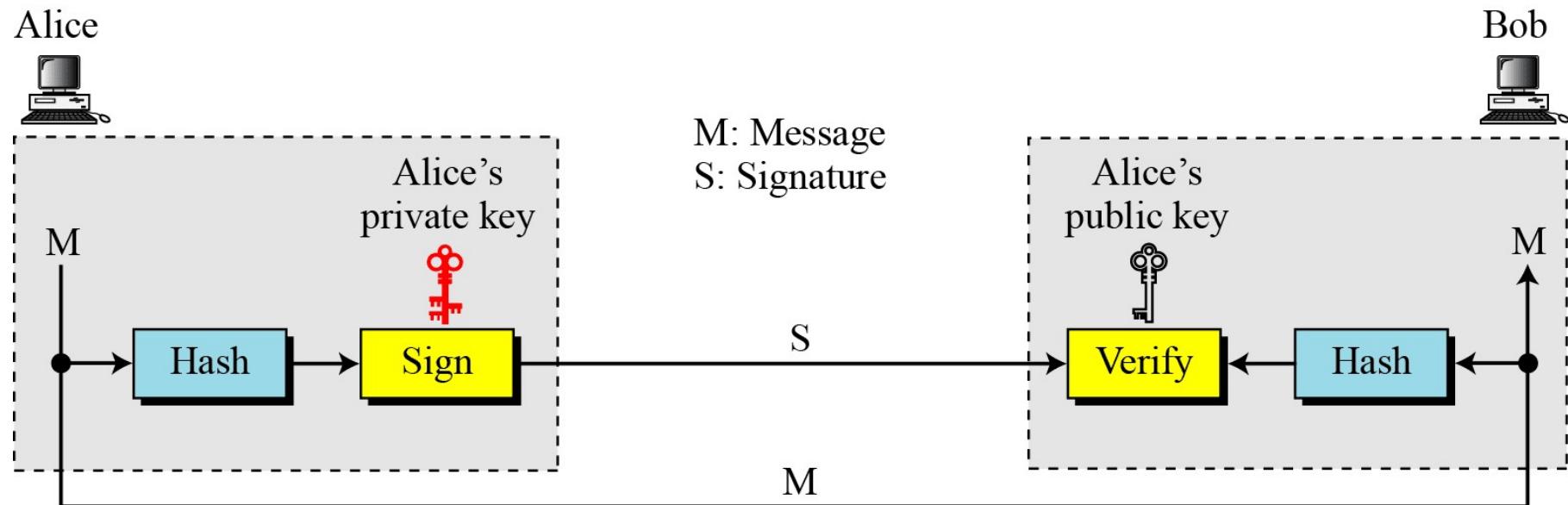
**Note**

A cryptosystem uses the private and public keys of  
the receiver: a digital signature uses  
the private and public keys of the sender.



# Signing the digest

1. Asymmetric key cryptosystem are very inefficient when dealing with long messages.
2. In digital signature system, the messages are normally long, but we have to use asymmetric key schemes
3. The solution is to sign a digest of the message, which is much shorter than the message
4. The sender can sign the message digest and the receiver can verify the message digest



# SERVICES

---

1. Various services provided by Digital Signature are: Message Confidentiality, Message Authentication, Message Integrity, and Non-repudiation.
2. A digital signature can directly provide the last three; for message confidentiality we still need encryption/decryption.

## MESSAGE AUTHENTICATION

A secure digital signature scheme, like a secure conventional signature can provide message authentication.

## MESSAGE INTEGRITY

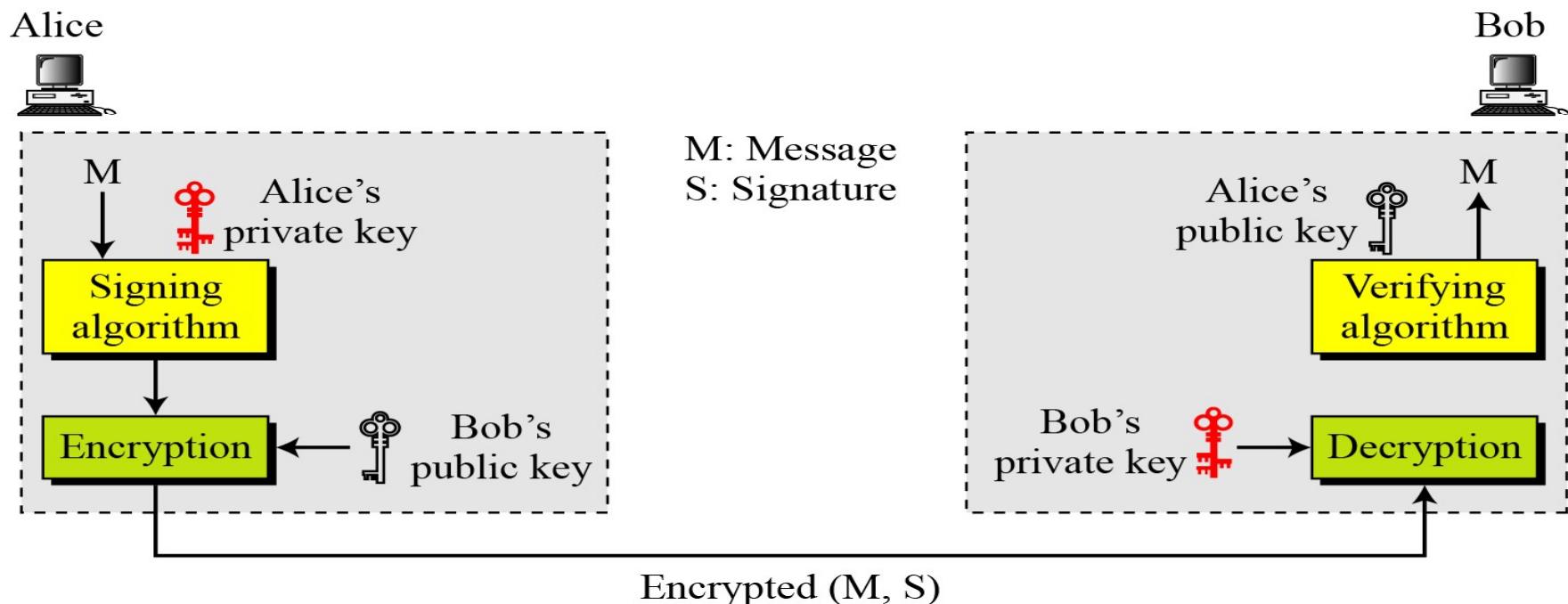
- The integrity of the message is preserved even if we sign the whole message because we cannot get the same signature if the message is changed.
- The digital signature uses a hash function in the signing and verifying algorithms that preserve the integrity of the message.



# SERVICES (Contd...)

## CONFIDENTIALITY

- ✓ A digital signature does not provide confidential communication.
- ✓ If confidential communication is required, then the message and the signature must be encrypted using either a Secret key or Public key cryptosystem.



Attacker has access to one or more message-signature pair. In other words, attacker has access to some documents previously signed by Sender. Attacker tries to create another message and forge attacker's signature on it.

## Digital Signature

### Attacks

1. Key-Only Attack
2. Known-Message Attack
3. Chosen-Message Attack

### Forgery

1. Existential Forgery
2. Selective Forgery

Attacker has access only to the public information released by the sender. To forge a message, attacker needs to create Sender's signature to convince receiver that is message is coming from authentic sender.

Attacker somehow makes the sender sign one or messages for her. Attacker will now have a chosen message-signature pair. Attacker later creates another message, with the content it wants and forges Sender's signature on it.



# ATTACKS ON DIGITAL SIGNATURE

---

## Digital Signature



### Attacks

1. Key-Only Attack
2. Known-Message Attack
3. Chosen-Message Attack

### Forgery

1. Existential Forgery
2. Selective Forgery

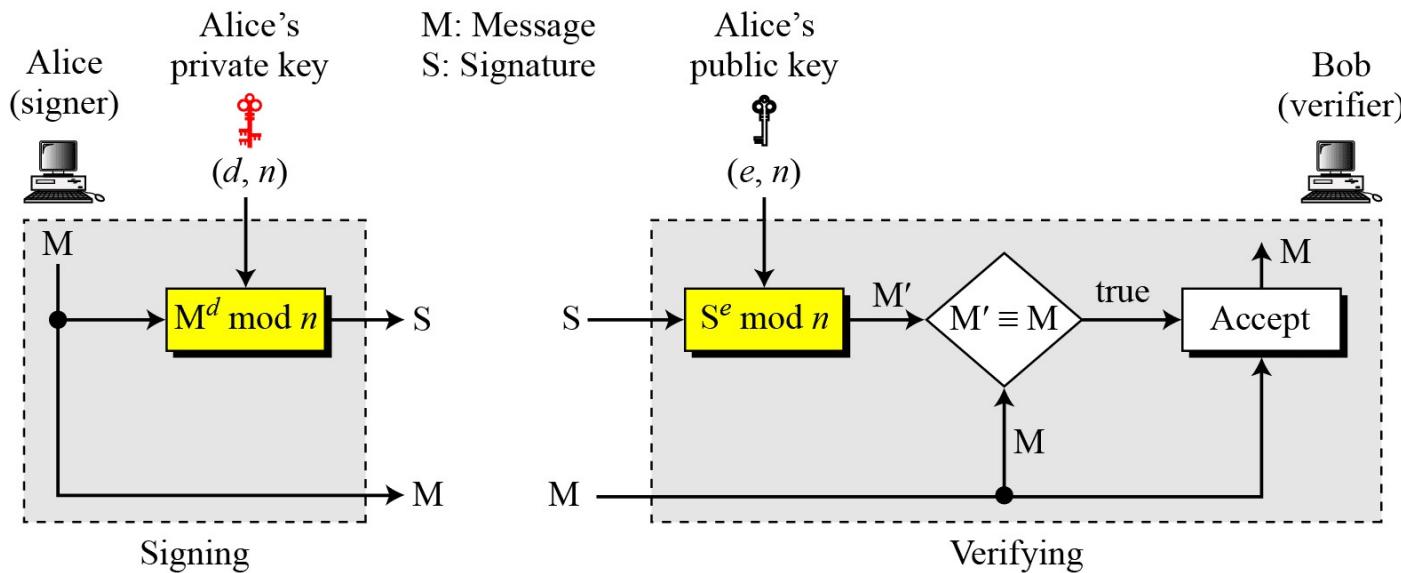
Attacker may be able to create a valid message-signature pair, but cannot really use

Attacker may be able to forge sender's signature on a message with the content selectively chosen by attacker. This is beneficial to the attacker but the probability of such forgery is very low, but not negligible.



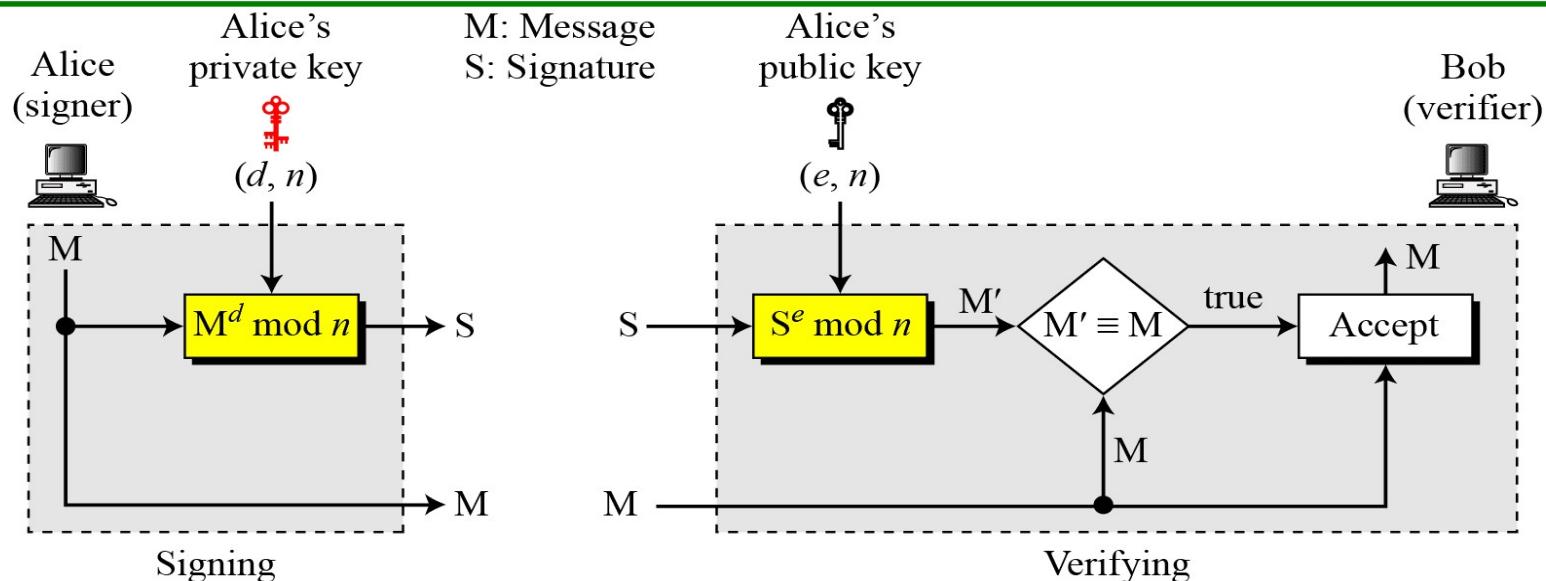
# RSA Digital Signature Scheme

1. The idea of RSA can also be used for signing and verifying a message.
2. This technique is known as **RSA Digital Signature Scheme**.
3. The digital signature scheme changes the role of PRIVATE and PUBLIC key
4. Here, the Private and Public key of the SENDER is used.
5. SENDER uses her own PRIVATE KEY to sign the document.
6. Receiver used the sender's PUBLIC KEY to verify it.

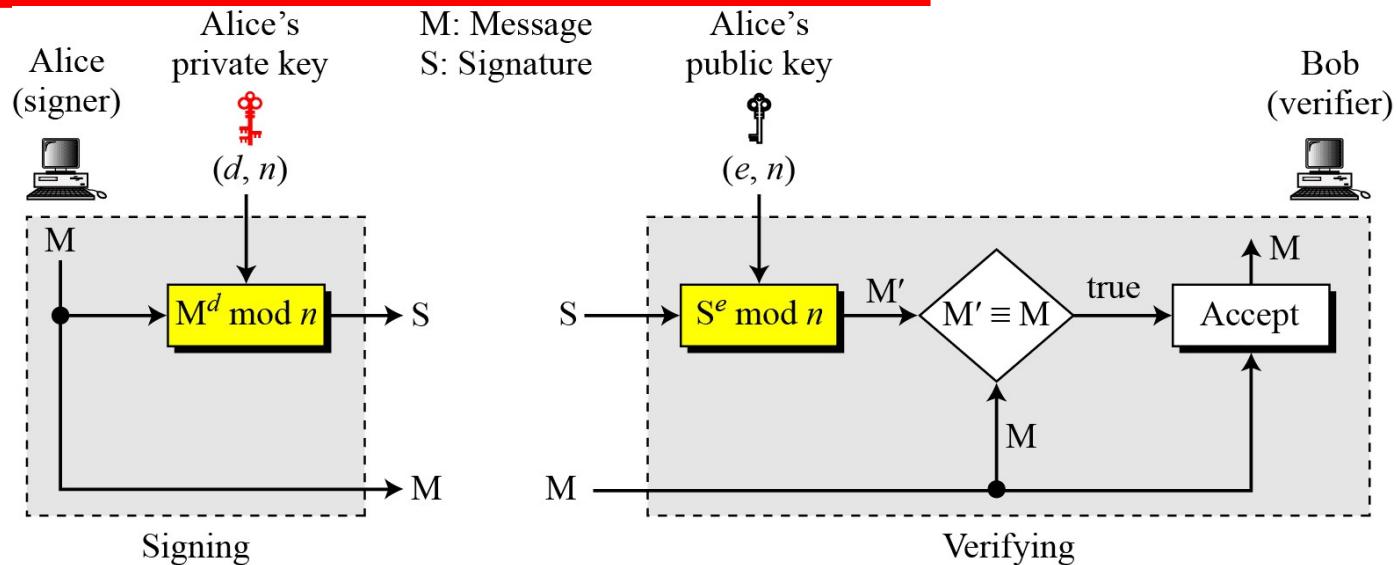


# RSA Digital Signature Scheme – KEY GENERATION

1. Key generation in the RSA digital signature scheme is exactly the same as key generation in the RSA
2. Sender selects 2 prime numbers  $p$  and  $q$
3. Calculate  $\phi(n) = (p-1)(q-1)$
4. Sender then chooses  $e$ , calculates  $d$  such that  $e * d = 1 \text{ mod } \phi(n)$
5. Sender keeps  $d$  and **announces  $n$  and  $e$**



# RSA Digital Signature Scheme – Example



For  $p = 7$ ,  $q = 13$  and  $e = 5$

Calculate  $d$  and Signature  $S$  for Message  $m = 35$

Verify the signature on receiver side



# Elgamal Digital Signature Scheme-General Idea

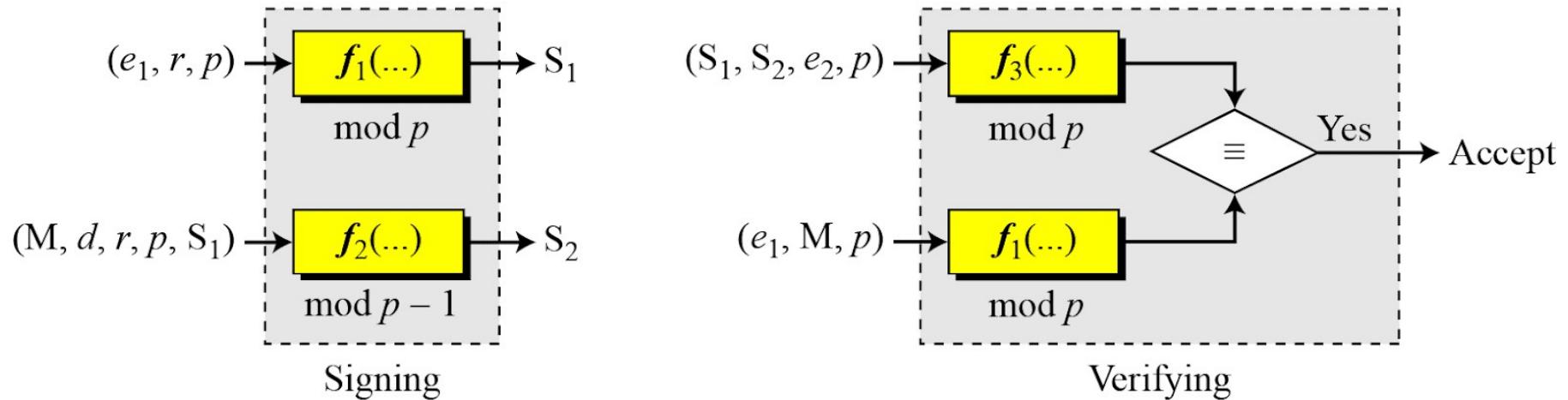
$S_1, S_2$ : Signatures

M: Message

$(e_1, e_2, p)$ : Alice's public key

$d$ : Alice's private key

r: Random secret



**Note**

In ElGamal digital signature scheme,  $(e_1, e_2, p)$  is Alice's public key;  $d$  is her private key.



# Elgamal Digital Signature Scheme

## Verifying and Signing

M: Message

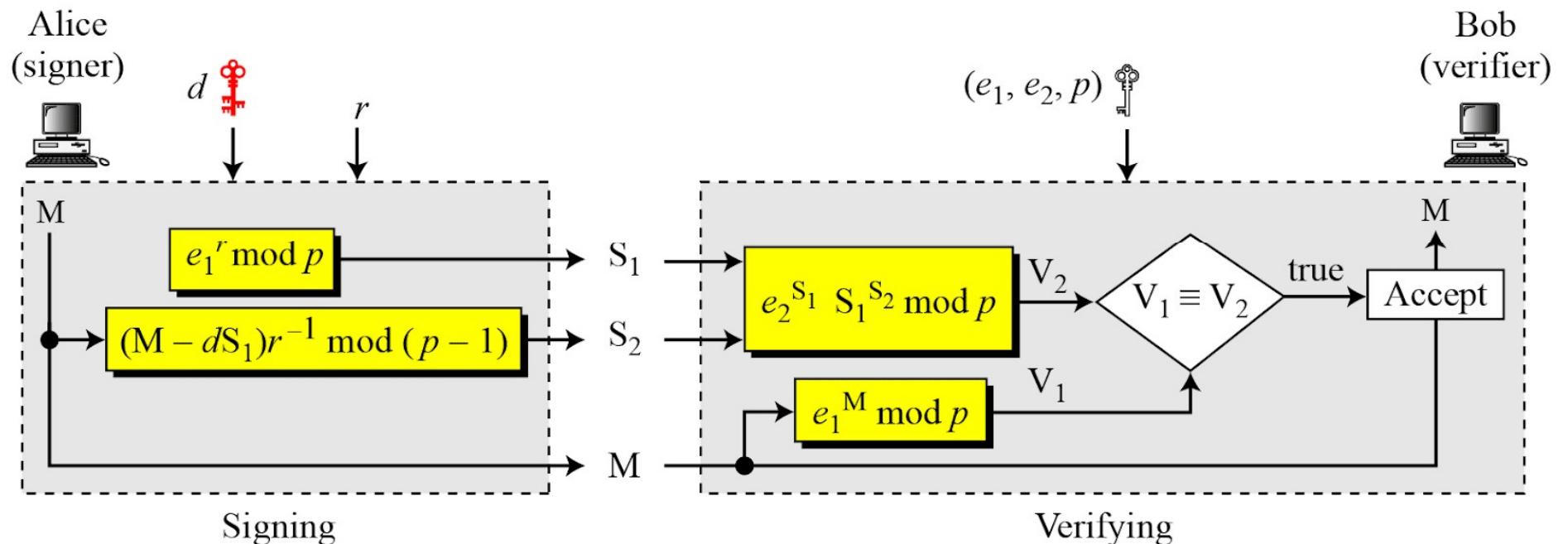
S<sub>1</sub>, S<sub>2</sub>: Signatures

V<sub>1</sub>, V<sub>2</sub>: Verifications

r: Random secret

d: Alice's private key

(e<sub>1</sub>, e<sub>2</sub>, p): Alice's public key



# Elgamal Digital Signature Scheme

## Verifying and Signing

SIGNING	VERIFYNG
Alice chooses a secret random number r	Bob checks if $0 < S1 < p$
Alice calculates the first Signature $S1 = e1^r \text{ mod } p$	Bob checks if $0 < S2 < p-1$
Alice calculates the Second Signature $S2 = (M - d * S1) * r^{-1} \text{ mod } (p-1)$ Where $r^{-1}$ is the multiplicative inverse of r modulo $p - 1$	Bob calculates $V1 = e1^M \text{ mod } p$
Alice Sends M, S1, S2 to Bob	Bob calculates $V2 = e2^{S1} * S1^{S2} \text{ mod } p$

$$e2 = e1^d \text{ mod } p$$

ALICE: SENDER

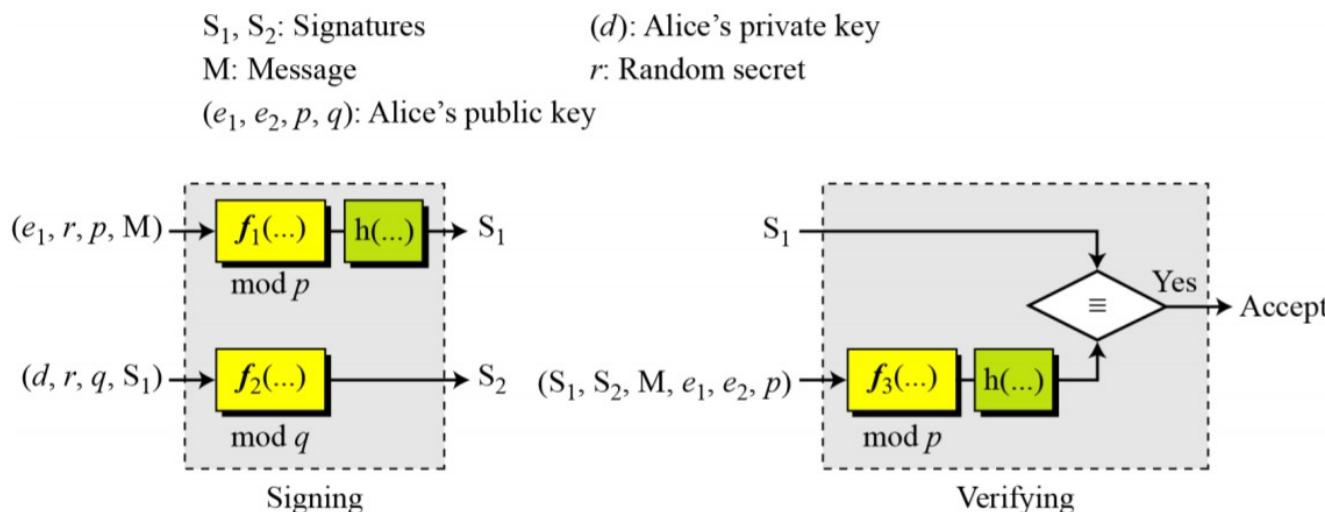
BOB : RECEIVER

Example: Using the Elgamal Scheme, let  $p = 19$  and  $d = 16$ . Find  $e2$  if  $e1 = 10$ . Choose  $r = 5$ . Find the values of  $S1$  and  $S2$  if  $M = 14$



# Schnorr Digital Signature Scheme

1. The problem with the ElGamal digital signature scheme is that  $p$  needs to be very large to guarantee that the discrete log problem is intractable in  $Z_p^*$ .
2. The recommendation is a  $p$  of atleast 1024 bits, which could make the signature as large as 2048 bits.
3. To reduce the size of the signature, Schnorr proposed a new scheme based on ElGamal, but with reduced signature size.



# Schnorr Digital Signature Scheme – Key Generation

---

Before signing a message, Alice needs to generate keys and announce the public comes to the public.

1. Alice selects a **prime p**
2. Alice selects **another prime q**, which is the same size as the digest created by the cryptographic has functions (**p-1 should be divisible by q**)
3. Calculate  $e1 = e_0^{(p-1)/q} \text{ mod } p$  ( $e_0$  is a primitive element in  $Z_p$ )
4. Alice chooses an integer **d** as her private key.
5. Alice calculates  $e2 = e_1^d \text{ mod } p$
6. **Alice public key is (e1, e2, p, q)**
7. **Alice private key is (d)**



# Schnorr Digital Signature Scheme

## Signing and Verifying

---

### SIGNING

Alice chooses a secret random number r

Alice calculates the first Signature

$$S1 = h(M | e1^r \bmod p)$$

Alice calculates the Second Signature

$$S2 = r + d * S1 \bmod q$$

Alice Sends M, S1, S2 to Bob

### VERIFYNG

Bob calculates  $V = h(M | e1^{S2} e2^{-S2} \bmod p)$

If  $S1$  is congruent to  $V$  modulo  $p$ , the message is accepted; otherwise it is rejected.



# Module 4: Authentication Protocols & Digital signature schemes

---

## 4.1

- User Authentication - One-way and mutual
- Needham Schroeder Authentication protocol
- Kerberos Authentication protocol.

## 4.2

- Digital Signature Schemes
  - a) RSA
  - b) ELGamal
  - c) Schnorr



# User Authentication

---

1. User authentication is the fundamental building block and the primary line of defence.
2. User authentication is the basis for most types of access control and for user accountability.
3. User Authentication: Is the process of verifying an identity claimed by or for a system entity.
4. An authentication process consists of two steps:
  - a) Identification Step: Presenting an identifier to the security system.
  - b) Verification step: Presenting or generating authentication information that corroborates the binding between the entity and the identifier.



# Means of authenticating User's Identity

---

There are 4 means of authenticating a user's identity, which can be used alone or in combination:

- A. Something the individual knows
- B. Something the individual possesses
- C. Something the individual is (Static Biometrics)
- D. Something the individual does (Dynamic Biometrics)

All of the above methods, if properly implemented and used, can provide secure user authentication.

However, each method has problems



# MUTUAL AUTHENTICATION

---

- Mutual Authentication protocols enable communicating parties to satisfy themselves mutually about each other's identity and to exchange session keys.
- Central to the problem of authenticated key exchange are two issues:
  1. **Confidentiality:** To prevent masquerade and to prevent compromise of session keys, essential identification and session-key must be communicated in an encrypted form
  2. **Timeliness:** Is important because of the threat of MESSAGE REPLAYS. Such replays could allow an opponent to compromise a session key or successfully impersonate another party
- To achieve mutual authentication following two general approaches are used;
  - a) **Timestamps**
  - b) **Challenge/Response**



# MUTUAL AUTHENTICATION

---

## a) Timestamps

Party A accepts a message as fresh only if the message contains a TIMESTAMP that, In A's judgement, is close enough to A's knowledge of current time. This approach requires that clocks among the various participants be synchronized.

## b) Challenge/Response

Party A, expecting a fresh message from B, first sends B a NONCE and requires that the subsequent message (Response) received from B contain the correct NONCE value.



# Difficulties involved in following approach.....

---

## a) Timestamps

- Should not be used for CONNECTION-ORIENTED Applications
- **Reasons:**
  - a) Some sort of protocol is needed to maintain synchronization among the various processor clocks. This protocol must be Fault Tolerant, should be able to cope with network errors and hostile attacks.
  - b) Because of variable and unpredictable nature of network delays, distributed clocks cannot be expected to maintain precise synchronization.

## b) Challenge/Response

- Should not be used for CONNECTION-LESS Applications
- Since it requires the overhead of a handshake before any connectionless transmission, effectively negating the chief characteristic of a connectionless transaction



# Key Distribution Centre : KDC

---

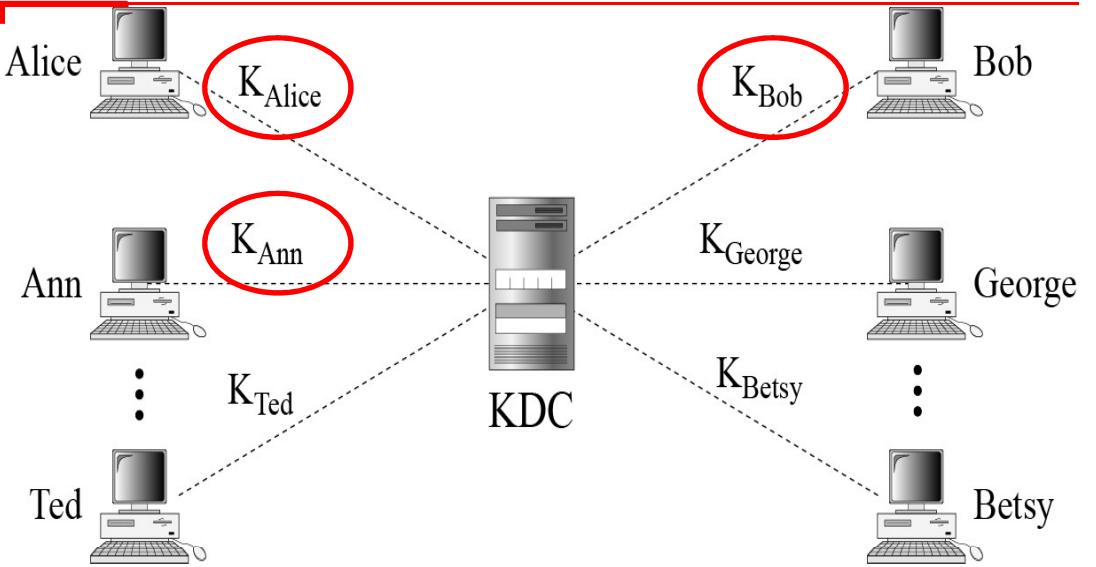
- Symmetric-key cryptography is more efficient than asymmetric-key cryptography for enciphering large messages.
- Symmetric-key cryptography, however, needs a shared secret key between two parties.
- If Alice needs to exchange confidential messages with N people, she needs N different keys.
- Thus, if N people needs to communicate to with each other, then a total of  $N(N-1)$  keys are required.
- An efficient way is required to maintain and distribute Secret Keys.
- **To reduce the number of keys and to distribute the secret keys securely, Trusted third party called as KDC is used**



# Key Distribution Centre : KDC

The following process is taken when Alice wants to send a confidential message to Bob.

1. Alice sends a request to the KDC, that she needs a **session key** between herself and Bob.
2. The KDC informs Bob about Alice's request.
3. If Bob agrees a session key is created between the two



**The SESSION KEY between Alice and Bob that is established with the KDC is used to authenticate Alice and Bob to the KDC and to prevent Attacker from impersonating either of them**



# Key Distribution Centre : KDC

---

## Secret Keys :

A KDC creates a secret key for each member.

This secret key can be used only between the member and the KDC, not between two members.

## Session Keys:

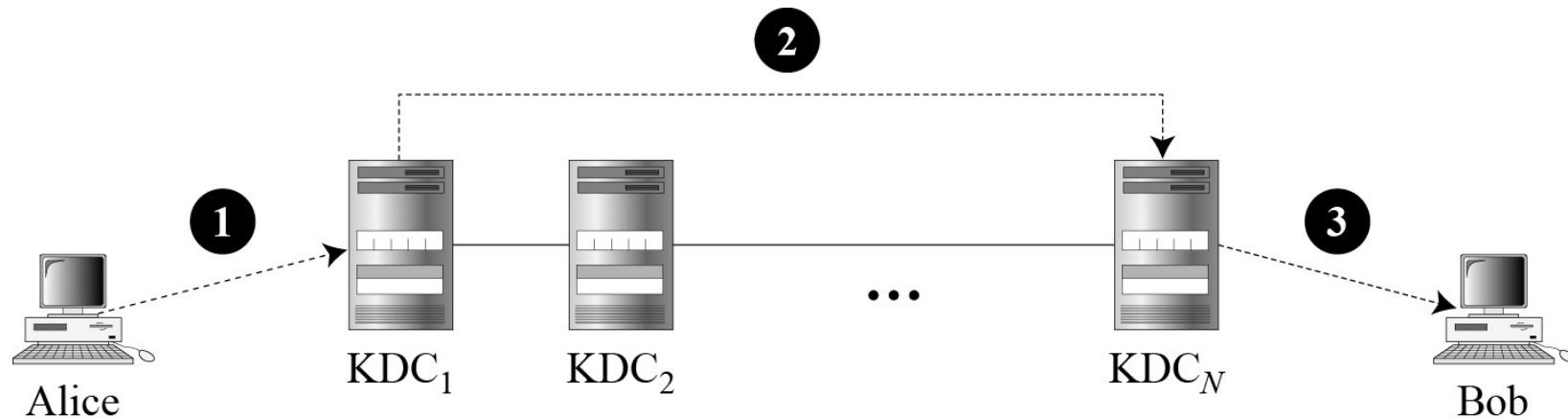
Session Keys is created between 2 members ( Eg. Alice and Bob)

A session symmetric key between two parties is used only once.



# Flat Multiple KDCs.

- When the number of people using KDC increases, the system becomes unmanageable and a bottleneck can result.
- To solve this problem, we need to have multiple KDC's
- We can divide the world into domains, each domain can have one or more KDC's (for redundancy in case of failure)

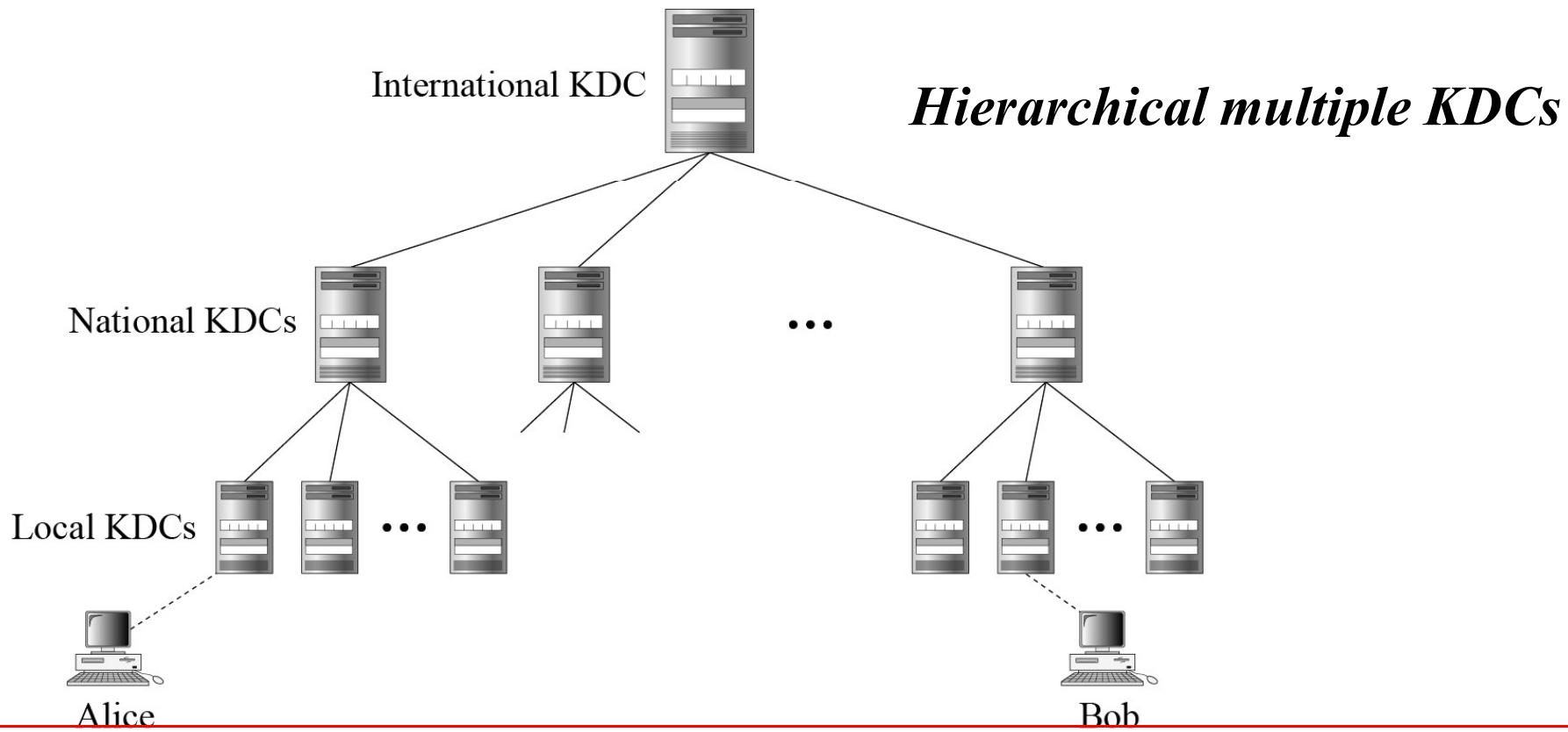


**The above concept is referred as FLAT MULTIPLE KDC**



# Hierarchical Multiple KDCs

- The concept of flat multiple KDC's can be extended to a hierarchical system of KDC's, with one or more KDC's at the top of the hierarchy.



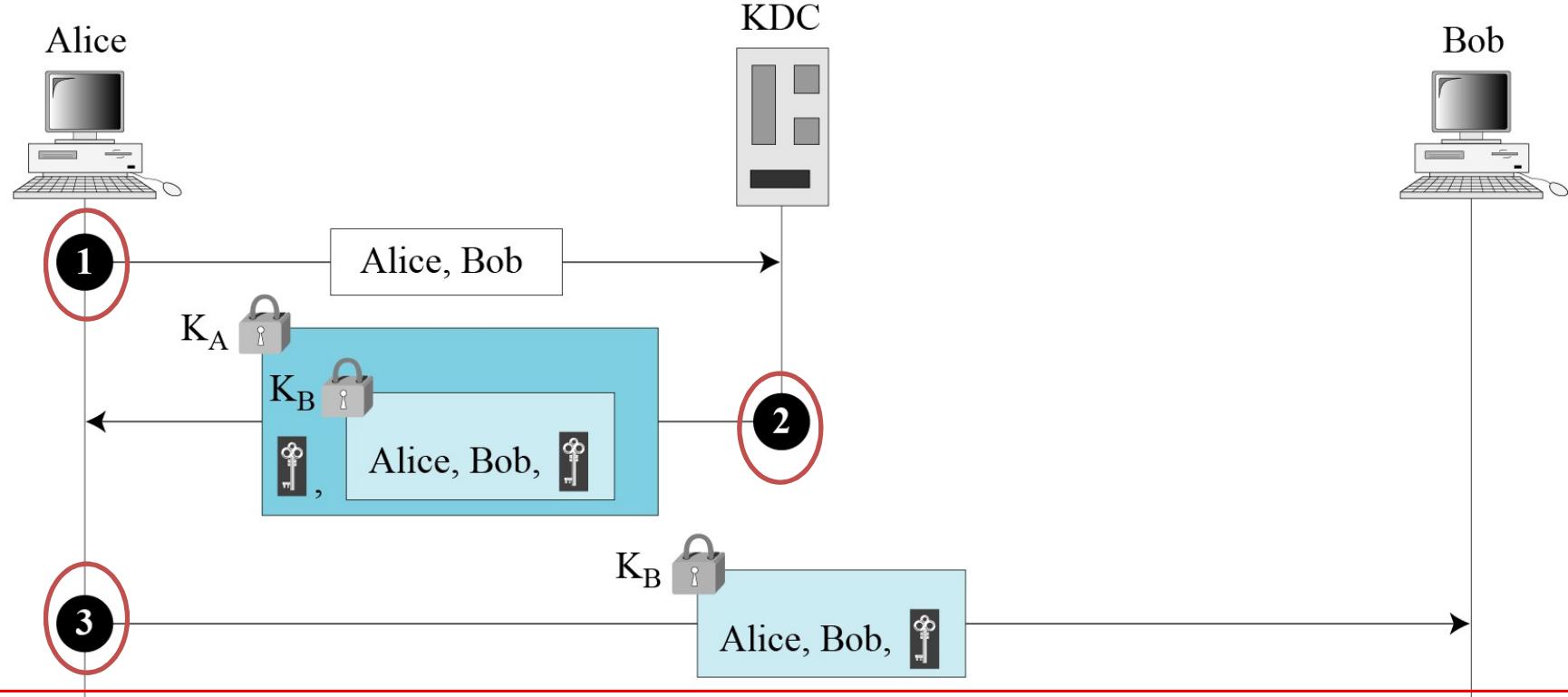
KDC receives the message and creates what is called a **TICKET**

## *A Simple Protocol Using a KDC*

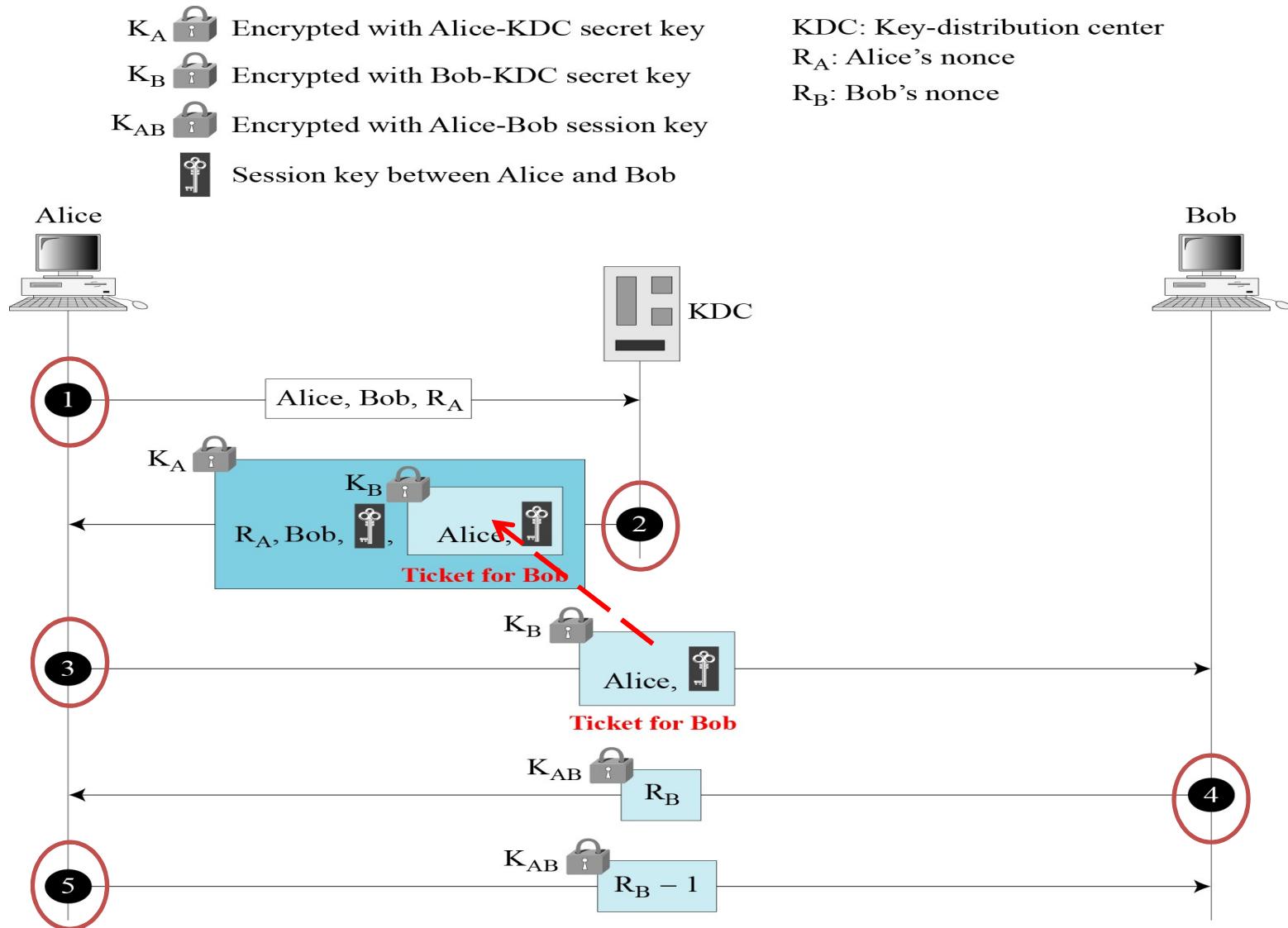
### *First approach using KDC*

$K_A$  Encrypted with Alice-KDC secret key      Session key between Alice and Bob

$K_B$  Encrypted with Bob-KDC secret key      KDC: Key-distribution center



# Needham-Schroeder Protocol



# KERBEROS

---

- Kerberos is an authentication protocol, and at the same time a KDC, that has become very popular.
- Several systems, including Windows 2000, use Kerberos.
- Originally designed at MIT, it has gone through several versions.

## **SERVERS**

Three servers are involved in the Kerberos protocol:

- **Authentication Server ( AS):**

The authentication server (AS) is the KDC in the Kerberos protocol. Each user registers with the AS and is granted a user identity and a password. The AS has a database with these identities and the corresponding passwords. The AS verifies the user, issues a session key to be used between Alice and TGS, and send a ticket for the TGS

- **Ticket Granting Server (TGS):**

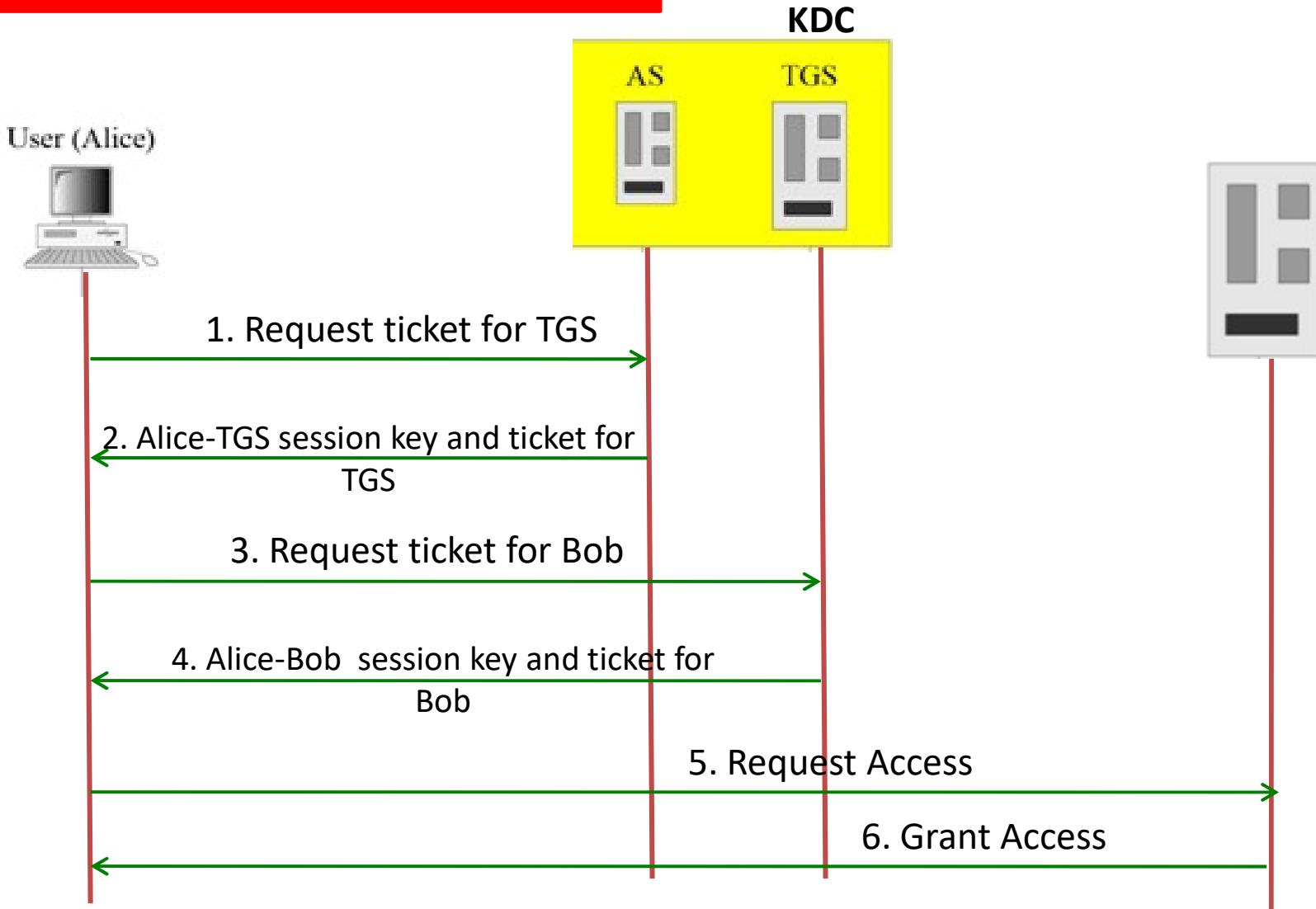
The **TGS** issues a ticket for the real server (Bob). It also provides the session key KAB between Alice and Bob. Kerberos has separated user verification from the issuing of tickets. In this way, though Alice verifies her ID just once with AS, she can contact TGS multiple times to obtain tickets for different real servers

- **A Real (data) Server :**

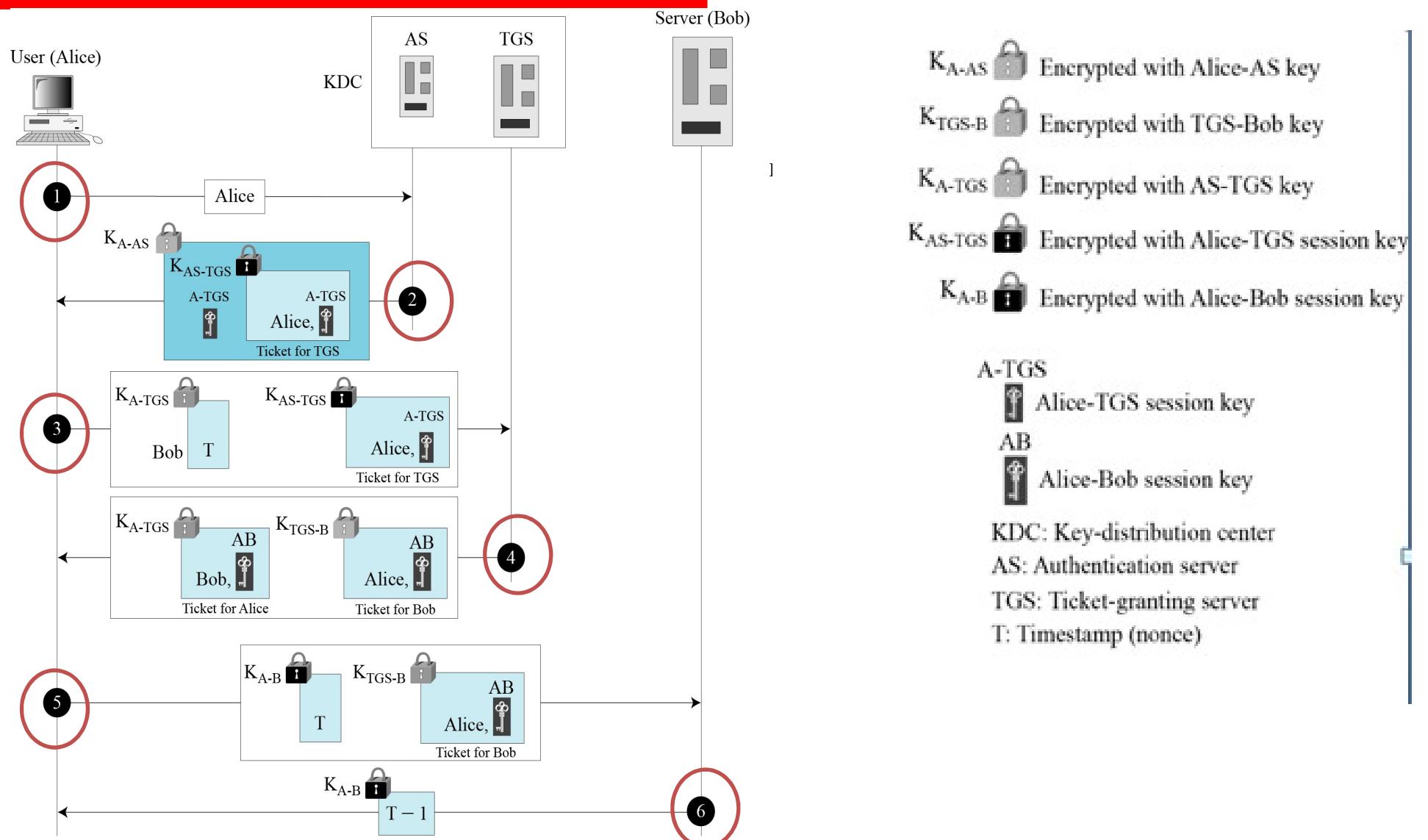
The real server (Bob) provides services for the user (Alice).



# KERBEROS



# KERBEROS



# Cryptography and System Security (CSS)

## Course Code: CSC 604



### Subject Incharge

Ankita Karia  
Assistant Professor  
Room No. 421  
email: [ankitakaria@sfit.ac.in](mailto:ankitakaria@sfit.ac.in)



# Module 5: Network Security and Applications

---

5.1

- Network Security Basics
- TCP/IP Vulnerabilities
- Packet Sniffing
- ARP Spoofing, DNS Spoofing
- Port Scanning, TCP Syn flood

5.3

- Internet Security Protocols: SSL, IPSEC
- Secure Email: PGP, Firewalls
- IDS and Its types
- Honey pots

5.2

- Denial of Service (DOS)
- Classic DOS attacks
- Source Address Spoofing
- ICMP Flood, SYN flood, UDP flood
- Distributed DOS, Defences against DOS Attacks



# Network Security Basics

---

1. A **network** is defined as two or more computing devices connected together for sharing resources efficiently.
2. Further, connecting two or more networks together is known as internetworking. Thus, the Internet is just an internetwork – a collection of interconnected networks.
3. **Network security** is the process of taking preventative measures to protect the underlying networking infrastructure from unauthorized access, misuse, malfunction, modification, destruction or improper disclosure.

For setting up its internal network, an organization has various options. It can use a wired network or a wireless network to connect all workstations. Nowadays, organizations are mostly using a combination of both **wired and wireless networks**.



# Components of Computer Network

---

**1. Nodes:** A network node is a connection point that can receive, create, store or send data along distributed network routes

- **End Nodes:** Starting point or End Point in the communication

Example: Computer, Network Printers, VoIP Phones, Security Cameras, Mobile Handheld Devices

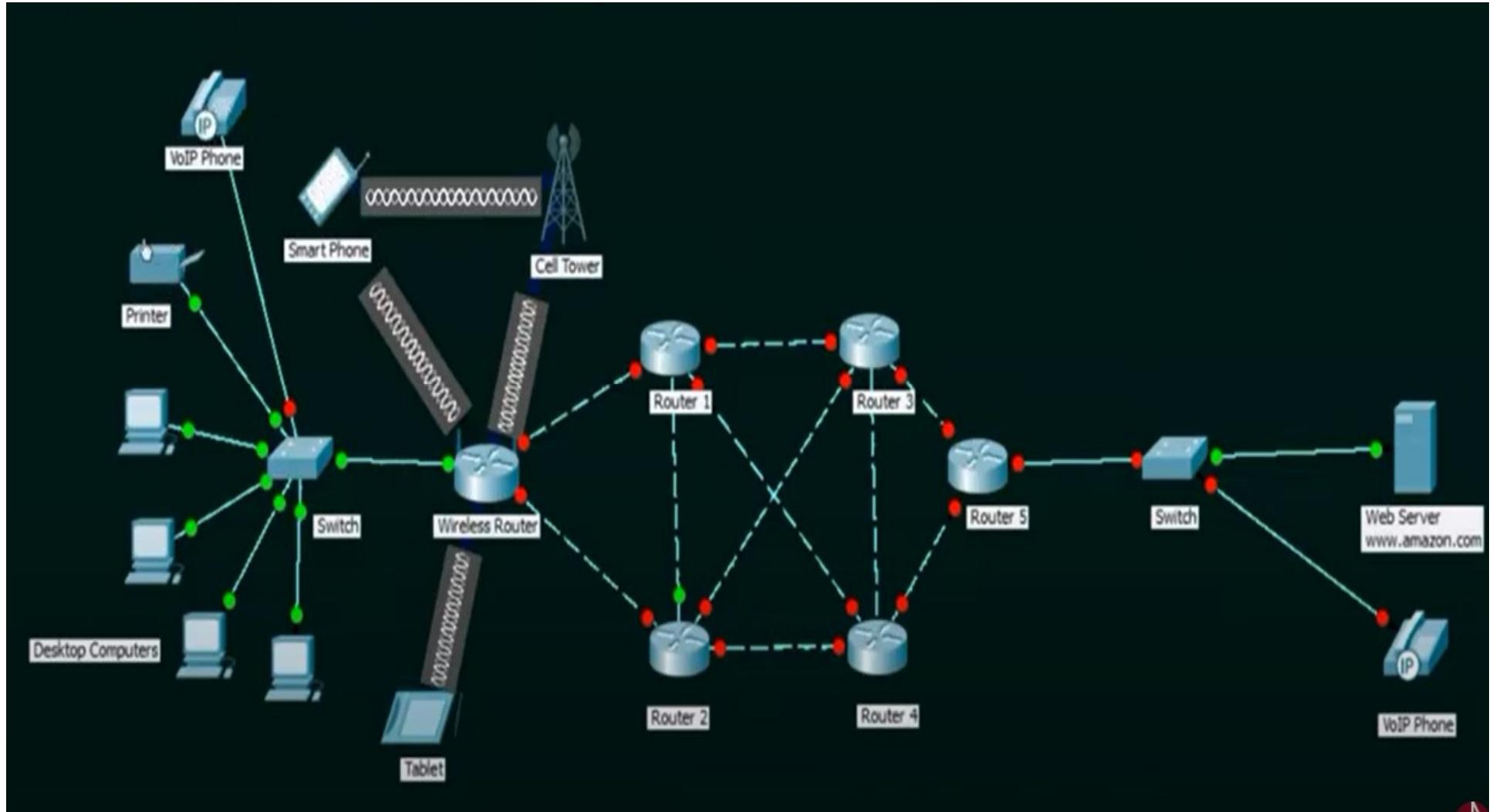
- **Intermediary Nodes:** Forwards data from one node to another. They are in between end nodes. Example: Switches, Router, Bridges, Hubs, Repeater, Cell Tower, Wireless Access Point etc.

**2. Media:** Wired or Wireless Medium

**3. Services:** E-mail, Storage Services, File Sharing, Online Game, Video Conferencing, Instant Messaging



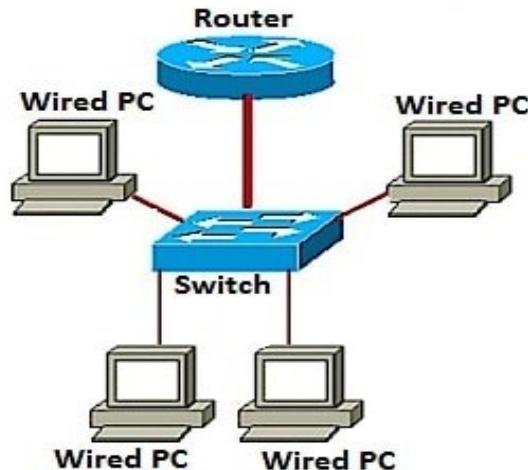
# Components of Computer Network



# Network Security Basics: Wired and Wireless Network.

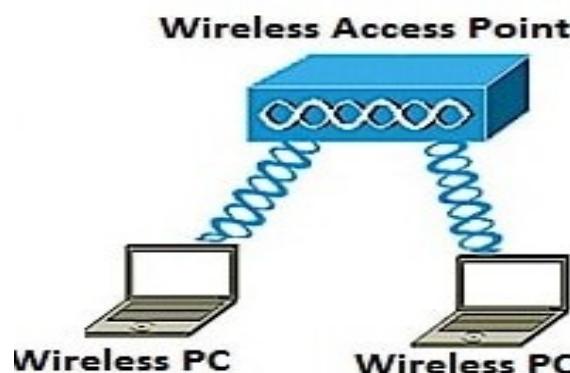
---

## WIRED NETWORK



- A wired setup uses physical cables to transfer data between different devices and computer systems.
- Wired network is used to carry different forms of electrical signals from one end to the other.
- Most wired networks use Ethernet cables to transfer data between connected PCs.

## WIRELESS NETWORK



- Wireless network refers to the use of infrared or radio frequency signals to share information and resources between devices.
- Wireless technologies are designed to reduce the time and different type of obstacles created by the cables.



# Network Security Basics: Vulnerabilities and Attack

---

- The common vulnerability that exists in both wired and wireless networks is an “unauthorized access” to a network. An attacker can connect his device to a network through unsecure hub/switch port.
- In this regard, **wireless network are considered less secure than wired network**, because wireless network can be easily accessed without any physical connection.

After accessing, an attacker can exploit this vulnerability to launch attacks such as –

- 1.Sniffing** the packet data to steal valuable information.
- 2.Denial of service** to legitimate users on a network by flooding the network medium with spurious packets.
- 3.Spoofing** physical identities (MAC) of legitimate hosts and then stealing data or further launching a ‘man-in-the-middle’ attack.



# Network Security Basics: Network Protocol

---

- Network Protocol is a set of rules that govern communications between devices connected on a network.
- They include mechanisms for making connections, as well as formatting rules for data packaging for messages sent and received.
- Several computer network protocols have been developed each designed for specific purposes.
- The popular and widely used protocols are TCP/IP with associated higher- and lower-level protocols.



# PROTOCOL STACK

---

- A **Protocol Stack** is a prescribed hierarchy of software layers, starting from the application layer at the top (the source of the data being sent) to the data link layer at the bottom (transmitting the bits on the wire).
- The protocols in a stack determine the interconnectivity rules for a layered network model such as in the OSI or TCP/IP models.

## TCP/IP

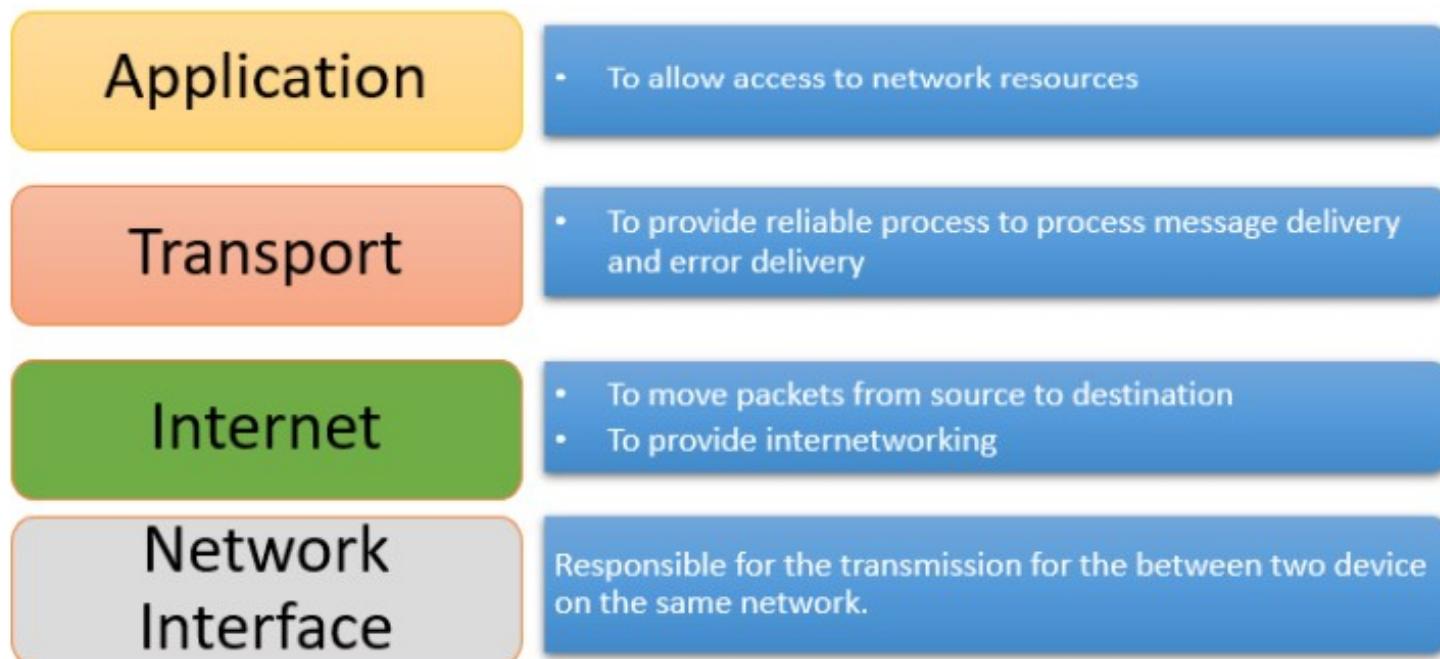
- ❖ IP stands for the Internet Protocol that deals with routing packets of data from one computer to another or from one router to another.
- ❖ TCP, which stands for Transmission Control Protocol, has the job of ensuring that the data packets delivered by the IP protocol did arrive at their destination.

TCP protocol sits on top of the IP protocol — in the sense that TCP asks IP to send a packet to its destination and then makes sure that the packet was actually received at the destination



# Four Layers of TCP/IP model

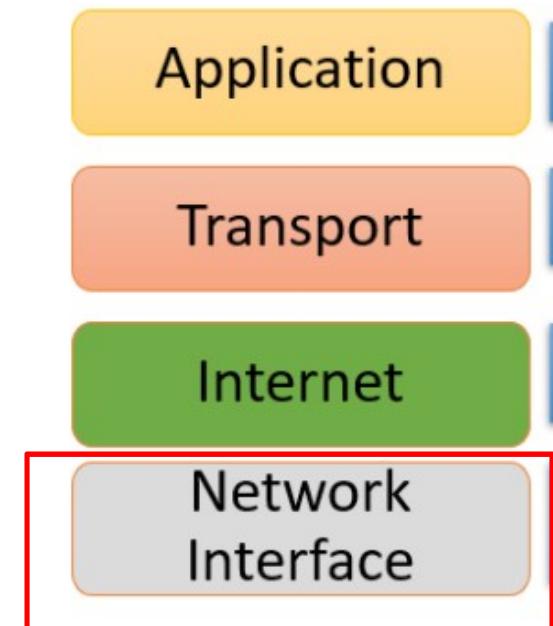
1. The functionality of the TCP IP model is divided into four layers, and each includes specific protocols.
2. TCP/IP is a layered server architecture system in which each layer is defined according to a specific function to perform. All these four TCP/IP layers work collaboratively to transmit the data from one layer to another.



# TCPIP Layers and Vulnerabilities

At the Network Interface layer, the packet of information that is placed on the wire is known as a frame. The packet is comprised of three areas: the header, the payload, and the FCS. Because the Network Interface layer is used for communications on a local network, the attacks that occur at this level would be carried out on local networks. Some of the ways the network layer can be exploited to compromise the C-I-A triad include the following:

- 1. MAC address spoofing**
- 2. Denial of service (DoS)**
- 3. ARP cache poisoning**



# TCPIP Layers and Vulnerabilities

Application

Transport

Internet

Network  
Interface

- 1. MAC address spoofing:** The header contains the MAC address of the source and destination computers and is required to successfully send a directed message from a source computer to a destination computer. Attackers can easily spoof the MAC address of another computer.
- 2. Denial of service (DoS):** A DoS attack overloads a single system so that it cannot provide the service it is configured to provide. An ARP protocol attack could be launched against a computer to overwhelm it, which would make it unavailable to support the C-I-A triad.
- 3. ARP cache poisoning:** The ARP cache stores MAC addresses of computers on the local network that have been contacted within a certain amount of time in memory. If incorrect, or spoofed, entries were added to the ARP cache, then the computer is not able to send information to the correct destination.

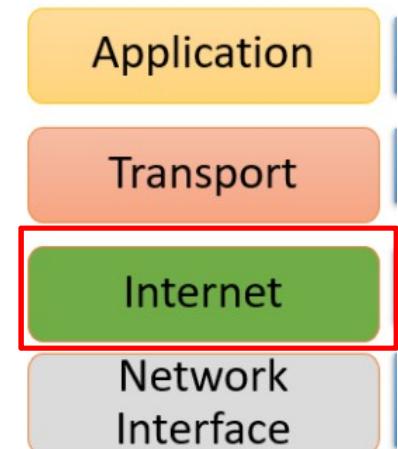


# TCPIP Layers and Vulnerabilities

At the **Internet layer**, IP datagrams are formed. The packet is comprised of two areas: the header and the payload. Some of the ways the Internet layer can be exploited to compromise the C-I-A triad include the following:

**IP address spoofing:** If the IP header fields and lengths are known, the IP address in the IP datagram can be easily discovered and spoofed.

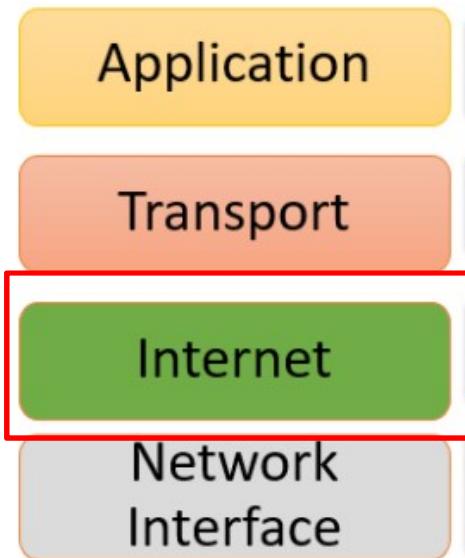
**Man-in-the-middle attacks:** This attack occurs when a hacker places himself or herself between the source and destination computer in such a way that neither notices his or her existence. Meanwhile, the attacker can modify packets or simply view their contents.



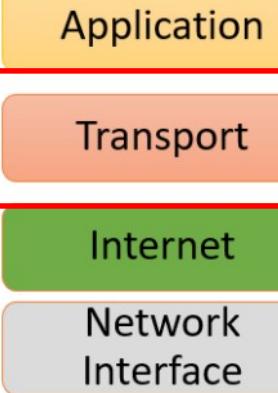
# TCPIP Layers and Vulnerabilities

---

**Corrupting packets:** Because IP datagrams can pass through several computers between the source and destination, the information in the IP header fields is read and sometimes modified, such as when the information reaches a router. If the packet is intercepted, the information in the header can be modified, corrupting the IP datagram. This could cause the datagram to never reach the destination computer, or it could change the protocols and payload information in the datagram.



# TCPIP Layers and Vulnerabilities



At the Transport layer, either a UDP header is added to the message or a TCP header is added. The application that is requesting the service determines what protocol will be used. Some of the ways the Transport layer can be exploited to compromise the C-I-A triad include the following:

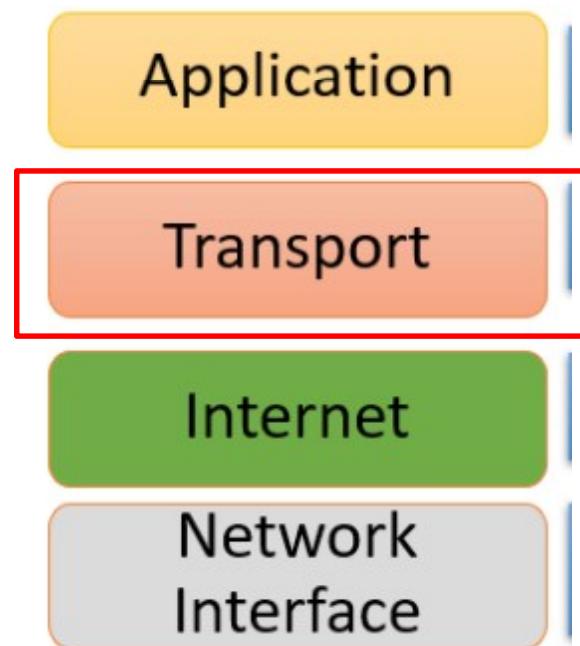
**DoS:** With a DoS attack at this level, simple IP-level protocols and utilities can be exploited to overload a computer, thus breaking the C-I-A triad. For instance, by knowing the steps involved in a three-way TCP handshake, a hacker or cracker might send the packets in the incorrect order and disrupt the availability of one of your servers. An example of this is a **SYN flood**



# TCPIP Layers and Vulnerabilities

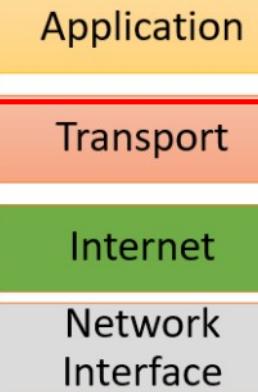
---

**Session hijacking:** This kind of attack occurs after a source and destination computer have established a communications link. A third computer disables the ability of one the computers to communicate, and then imitates that computer. Because the connection has already been established, the third computer can disrupt your C-I-A triad.



# TCPIP Layers and Vulnerabilities

**Application layer** attacks can be some of the most difficult to protect against because they take advantage of vulnerabilities in applications and lack of end-user knowledge of computer security. Some of the ways the Application layer can be exploited to compromise the C-I-A triad include the following:



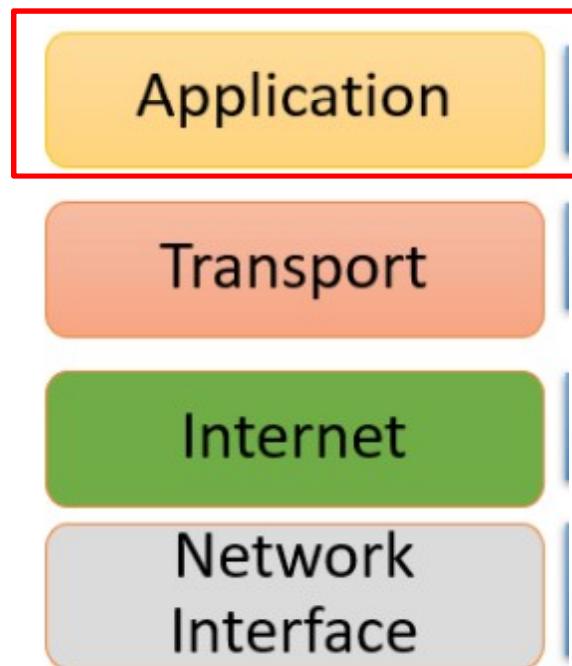
**E-mail application exploits:** Attachments can be added to e-mail messages and delivered to a user's inbox. The user can open the e-mail message and run the application. The attachment might do immediate damage, or might lay dormant and be used later. Similarly, hackers often embed malicious code in Hypertext Markup Language (HTML) formatted messages. Exploits of this nature might take advantage of vulnerability in the client's e-mail application or a lack of user knowledge about e-mail security concerns.



# TCPIP Layers and Vulnerabilities

---

**Web browser exploits:** When a client computer uses a Web browser to connect to a Web server and download a Web page, the content of the Web page can be active. That is, the content is not just static information, but can be executable code. If the code is malicious, it can be used to disrupt the C-I-A triad.



# PACKET SNIFFING

---

1. When any data has to be transmitted over the computer network, it is broken down into smaller units at the sender's node called **data packets** and reassembled at receiver's node in original format.
2. Data Packets is the smallest unit of communication over a computer network.
3. Data Packet is also called a block, a segment, a datagram or a cell.
4. The act of capturing data packet across the computer network is called **packet sniffing**.
5. It is similar to as wire tapping to a telephone network.

## Packet Sniffer

Packet sniffing is done by using tools called *packet sniffer*. It can be either ***filtered*** or ***unfiltered***. Filtered is used when only *specific data packets* have to be captured and Unfiltered is used when *all the packets* have to be captured.

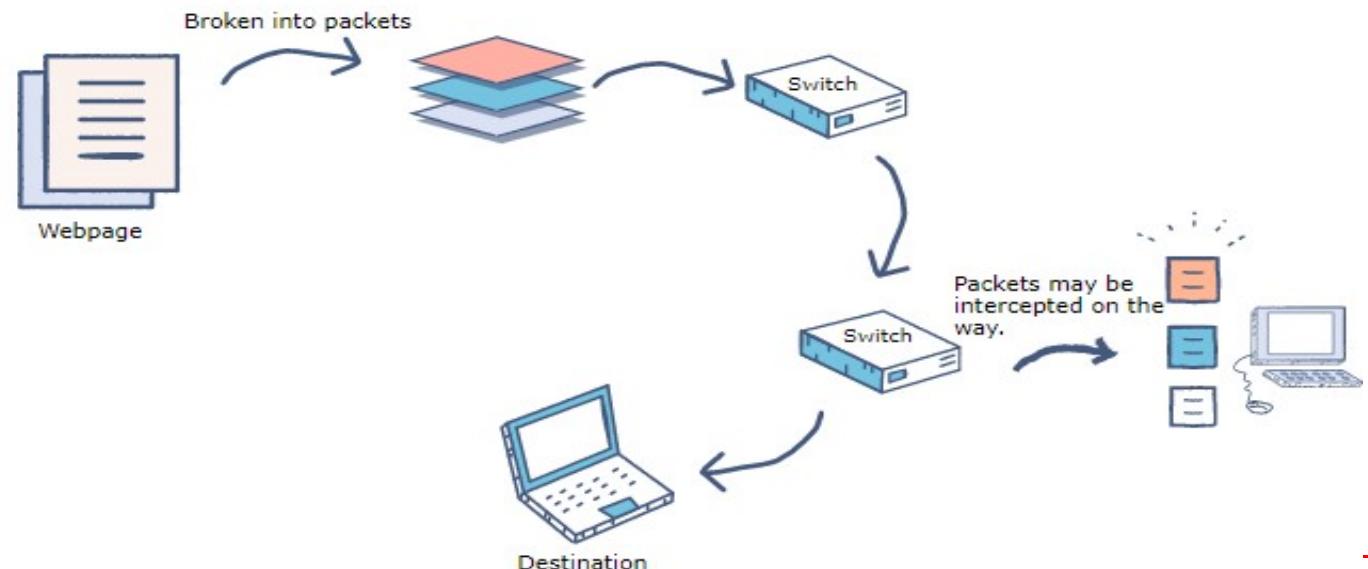
**WireShark, SmartSniff are examples of packet sniffing tools.**

---



# PACKET SNIFFING – How it Works

1. Web pages and emails are not sent through the internet as one document. Instead they are broken down into many little data packets.
2. These packets are then addressed to an IP address at the receiving end, which has to send back an acknowledgment of each packet it receives.
3. These packets are not transferred from the sender to the receiver through a single direct connection. Instead, as each packet traverses the internet en-route to its destination, it passes through a number of traffic control devices such as routers and switches. Each time a packet passes through one of these traffic control devices, it is susceptible to capture and analysis.



# PACKET SNIFFING – Who can sniff packets

---

1. It is mostly used by crackers and hackers to collect information illegally about network.
2. It is also used by ISPs, advertisers and governments.

**ISPs** use packet sniffing to track all your activities such as:

- ✓ who is receiver of your email
- ✓ what is content of that email
- ✓ what you download
- ✓ sites you visit
- ✓ what you looked on that website
- ✓ downloads from a site
- ✓ streaming events like video, audio, etc.

To achieve this target, these agencies use packet sniffing to *inject advertisements* into the flowing packets. Most of the time these ads *contain malware*.

**Advertising agencies** or internet advertising agencies are paid according to:

- ✓ number of ads shown by them.
- ✓ number of clicks on their ads also called PPC (pay per click).



\*Wireless Network Connection [Wireshark 1.10.2 (SVN Rev 51934 from /trunk-1.10)]

File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help

Filter: http Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Length	Info
172	10.8306270	192.168.43.42	69.195.124.112	HTTP	433	GET / HTTP/1.1
188	11.6480510	69.195.124.112	192.168.43.42	HTTP	1188	HTTP/1.1 200 OK (text/html)
325	23.5363370	108.160.162.52	192.168.43.42	HTTP	233	HTTP/1.1 200 OK (text/plai
326	23.5481440	192.168.43.42	108.160.162.52	HTTP	362	GET /subscribe?host_int=740
384	26.8239240	192.168.43.42	69.195.124.112	HTTP	724	POST /index.php HTTP/1.1 (
400	27.7360490	69.195.124.112	192.168.43.42	HTTP	1254	HTTP/1.1 302 Moved Temporar
402	27.7534960	192.168.43.42	69.195.124.112	HTTP	567	GET /dashboard.php HTTP/1.1
424	28.5163760	192.168.43.42	108.160.162.52	HTTP	362	[TCP Retransmission] GET /s
425	28.7380900	69.195.124.112	192.168.43.42	HTTP	1322	HTTP/1.1 200 OK (text/html)

Frame 384: 724 bytes on wire (5792 bits), 724 bytes captured (5792 bits) on interface 0

Ethernet II, Src: IntelCor\_a6:c5:43 (60:36:dd:a6:c5:43), Dst: Samsung\_E51:12:f3 (10:d5:42:51:1)

Internet Protocol Version 4, Src: 192.168.43.42 (192.168.43.42), Dst: 69.195.124.112 (69.195.1)

Transmission Control Protocol, Src Port: 57803 (57803), Dst Port: http (80), Seq: 1, Ack: 1, Len: 724

Hypertext Transfer Protocol

Line-based text data: application/x-www-form-urlencoded

email=admin%40google.com&password=Password2010&remember\_me=Remember+me

**all POST variables have been captured in plaintext**

Hex	Dec	ASCII
0000	10 d5 42 51 12 f3 60 36 dd a6 c5 43 08 00 45 00	. . BQ..`6 . . C..E.
0010	02 c6 3f 1a 40 00 80 06 0b 12 c0 a8 2b 2a 45 c3	..?.@.... . . .+*E.
0020	7c 70 e1 cb 00 50 03 e3 07 22 5f 45 14 e0 50 18	p....P.. . ." _E..P.
0030	11 1c 33 c1 00 00 50 4f 53 54 20 2f 69 6e 64 65	. . 3...PO ST /inde
0040	78 2e 70 68 70 20 48 54 54 50 2f 31 2e 31 0d 0a	x.php HT TP/1.1..
0050	49 6f 72 74 21 20 77 77 77 20 74 65 62 68 70 61	Host: www.w...techno

Frame (frame), 724 bytes Packets: 666 · Disp... Profile: Default

# SPOOFING

---

- Spoofing is a specific type of cyber-attack in which someone attempts to use a computer, device, or network to trick other computer networks by masquerading as a legitimate entity.
- It's one of many tools hackers use to gain access to computers to mine them for sensitive data, turn them into zombies (computers taken over for malicious use), or launch Denial-of-Service (DoS) attacks.

Three types of spoofing attacks exist:

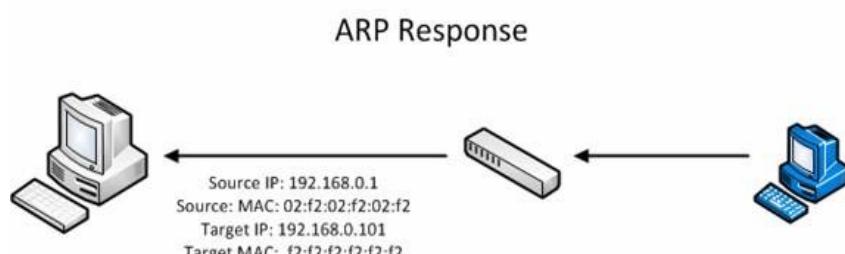
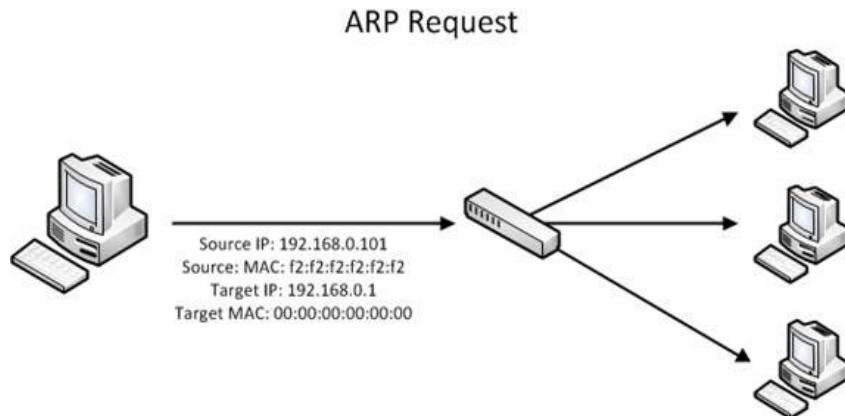
**DNS Server Spoofing:** Alters a DNS server to point a domain name to a different IP address, usually with the intent of spreading a virus.

**ARP Spoofing:** Connects hackers to an IP address through a spoofed address resolution protocol (ARP) message, usually to enable denial of service (DoS) and man-in-the-middle attacks.

**IP Spoofing:** Disguises one IP address to gain access as a trusted system, usually to enable a DDoS attack or redirect communications.



# ARP SPOOFING



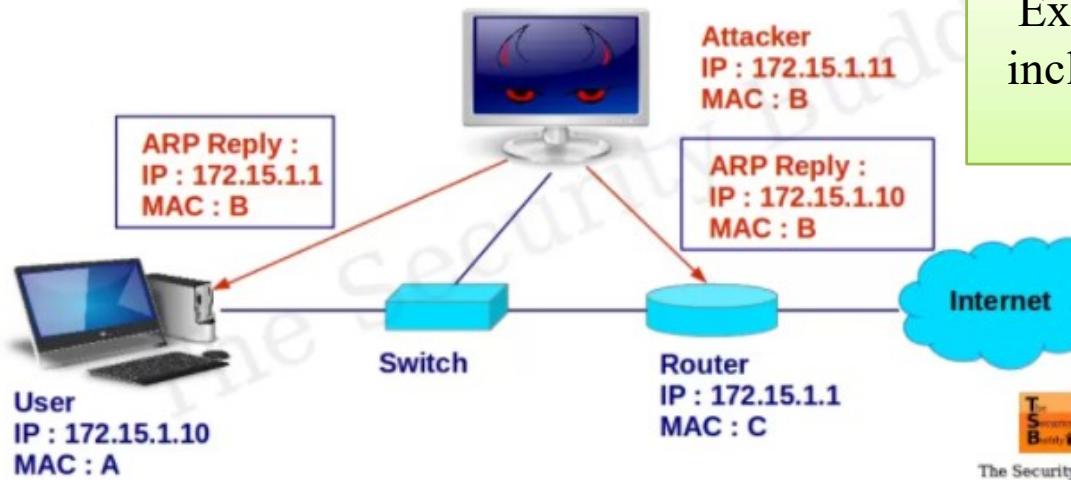
1. ARP spoofing is a type of attack in which a malicious actor sends falsified ARP (Address Resolution Protocol) messages over a local area network.
2. This results in the linking of an attacker's MAC address with the IP address of a legitimate computer or server on the network.
3. Once the attacker's MAC address is connected to an authentic IP address, the attacker will begin receiving any data that is intended for that IP address.
4. ARP spoofing can enable malicious parties to intercept, modify or even stop data in-transit. ARP spoofing attacks can only occur on local area networks that utilize the Address Resolution Protocol.



# ARP SPOOFING

1. An ARP spoofing, also known as ARP poisoning, is a Man in the Middle (MitM) attack that allows attackers to intercept communication between network devices.
2. It is an attack in which an attacker can send falsified ARP MESSAGES over a local area network and link the victim's IP address with the MAC address of the attacker's device .
3. As a result, all the traffic that is meant for the victim will reach the attacker first.
4. The attacker can afterward steal sensitive information or prepare for more attacks

## ARP Spoofing Attack

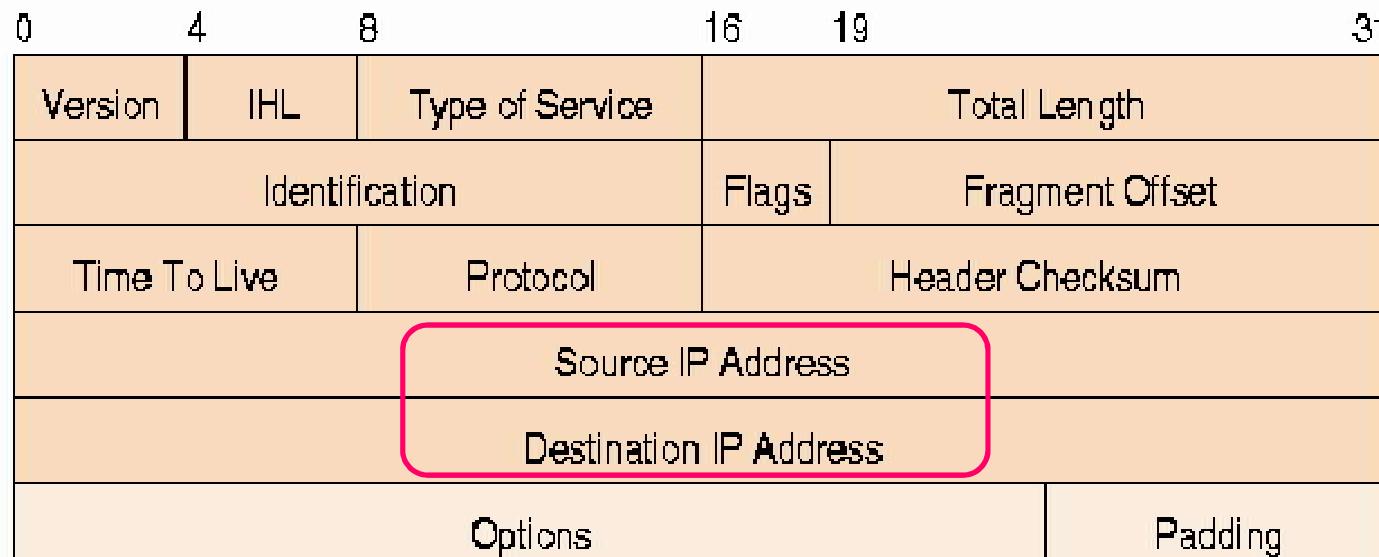


Examples of popular ARP spoofing software include Arpspoof, Cain & Abel, Arpoison and Ettercap.



# IP SPOOFING

1. The data transmitted over the internet is first broken into multiple packets, and those packets are transmitted independently and reassembled at the end.
2. Each packet has an IP (Internet Protocol) header that contains information about the packet, including the source IP address and the destination IP address.



In IP spoofing, a hacker uses tools to modify the source address in the packet header to make the receiving computer system think the packet is from a trusted source, such as another computer on a legitimate network, and accept it.



# IP SPOOFING

---

1. IP spoofing enables an attacker to replace a packet header's source IP address with a fake, or spoofed IP address.
2. The attacker does this by intercepting an IP packet and modifying it, before sending it on to its destination. This means that the IP address looks like it's from a trusted source – the original IP address – while masking its true source: an unknown third-party.
3. Once a hacker has successfully spoofed an IP address, they can access controlled systems and intercept communications intended for someone else (i.e., the person or device whose IP address they are impersonating).

IP spoofing commonly enables three different types of attacks:

1. DDoS Attacks
2. Botnet Attacks
3. Man in the Middle Attacks



# DNS SPOOFING

---

1. DNS Spoofing is a type of computer attack wherein a user is forced to navigate to a fake website disguised to look like a real one, with the intention of diverting traffic or stealing credentials of the users.
2. DNS spoofing and by extension, DNS cache poisoning are among the more deceptive cyberthreats



# DNS Concept

---

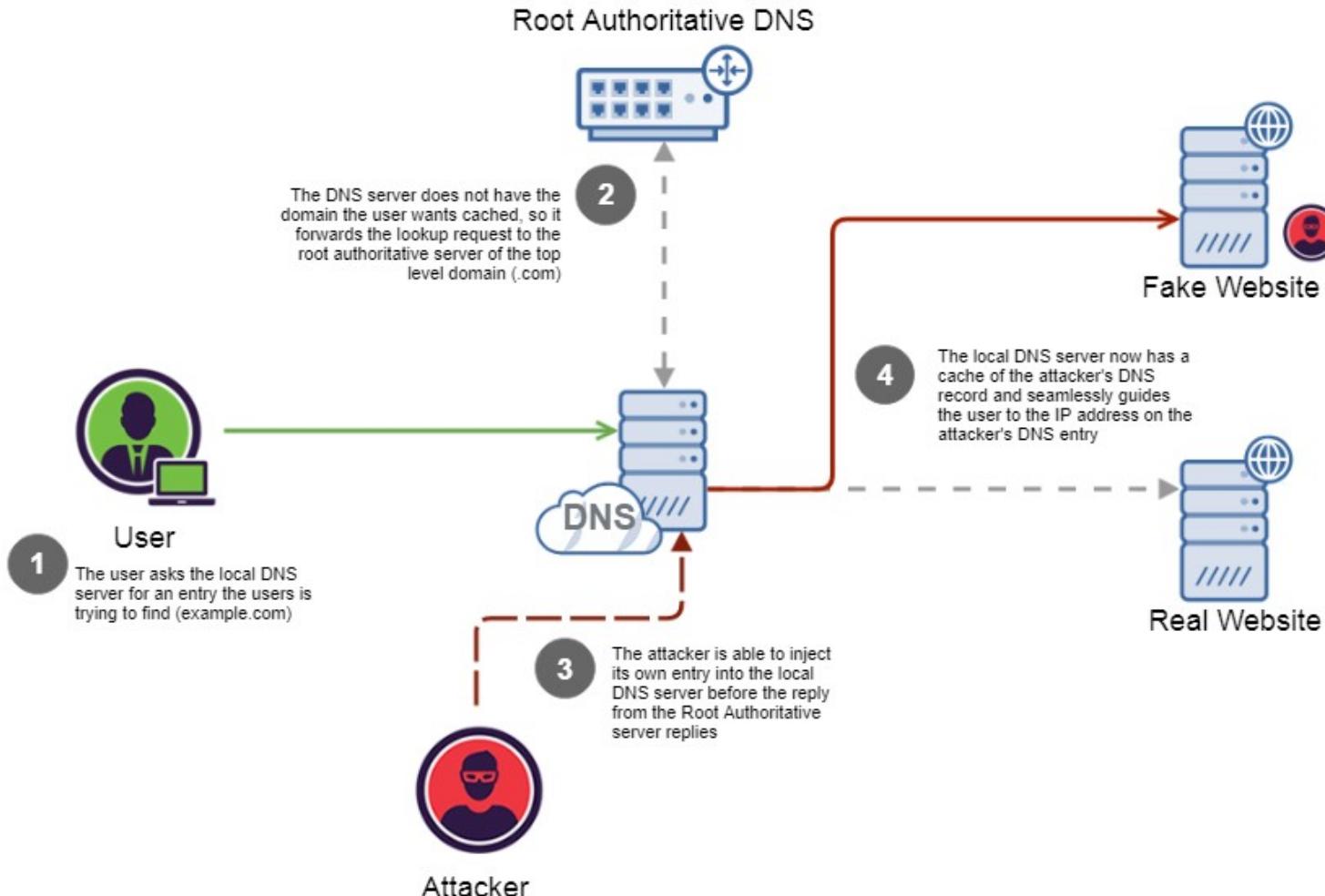
1. DNS stands for “Domain Name System.”
2. The Domain Name System (DNS) is the phonebook of the Internet.
3. Humans access information online through domain names, like [economictimes.com](http://economictimes.com) or [espn.com](http://espn.com). Web browsers interact through Internet Protocol (IP) addresses.
4. DNS translates domain names to IP addresses so browsers can load Internet resources.

## DNS Resolver

A DNS resolver, also called a **recursive resolver**, is a server designed to receive DNS queries from web browsers and other applications. The resolver receives a hostname - for example, [www.example.com](http://www.example.com) - and is responsible for tracking down the IP address for that hostname.

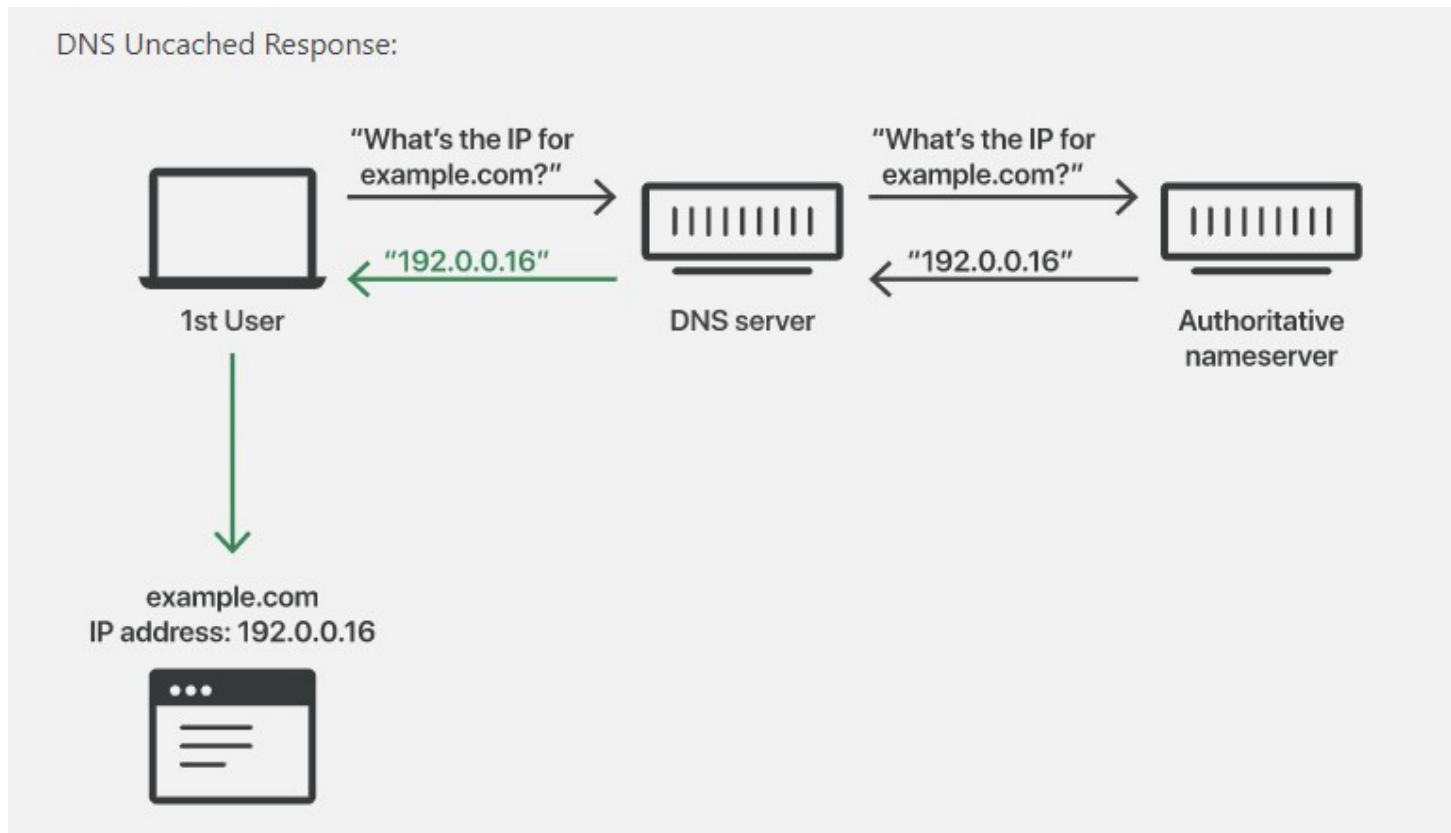


# DNS SPOOFING

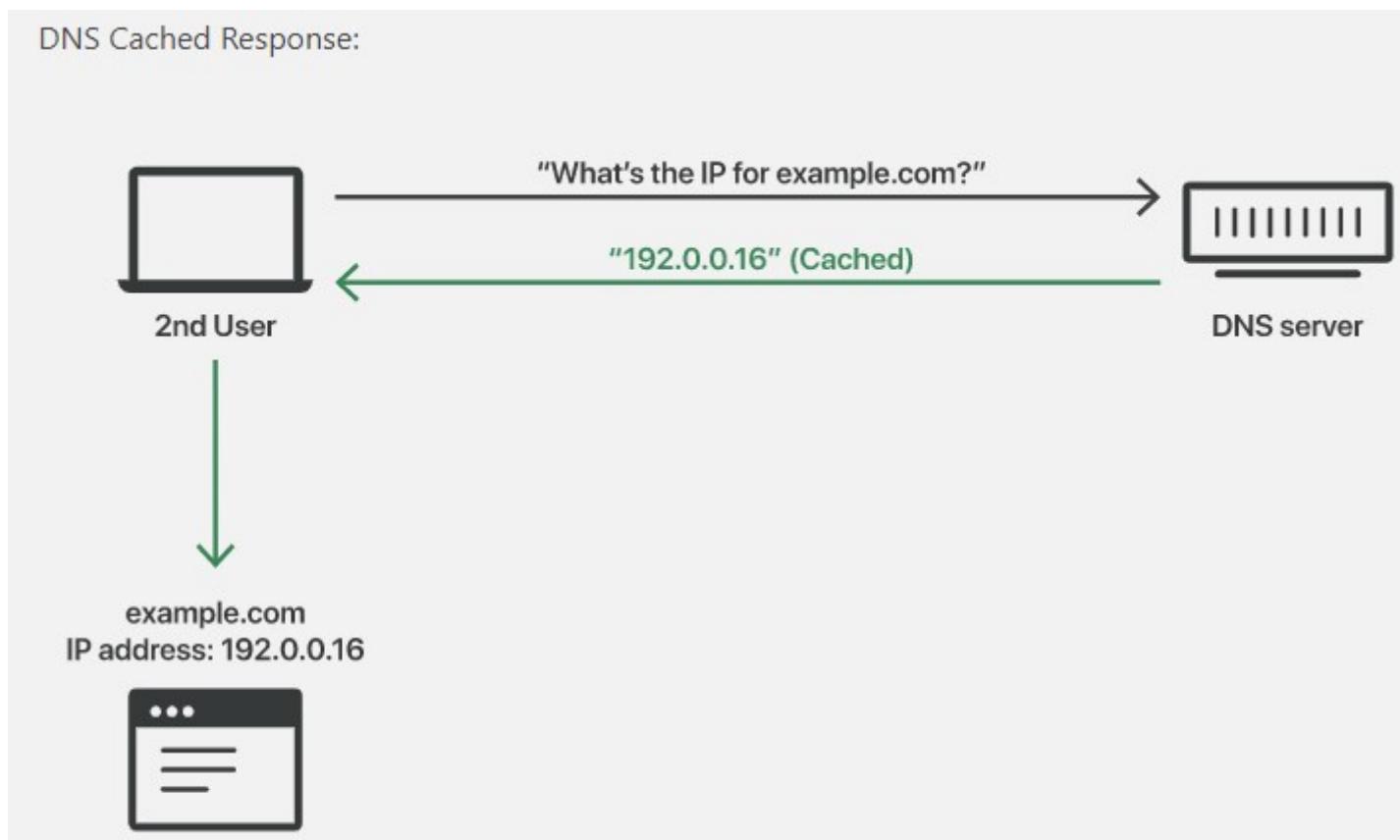


# DNS SPOOFING

1. DNS Spoofing is also known as DNS cache poisoning
2. DNS cache poisoning is the act of entering false information into a DNS cache, so that DNS queries return an incorrect response and users are directed to the wrong websites

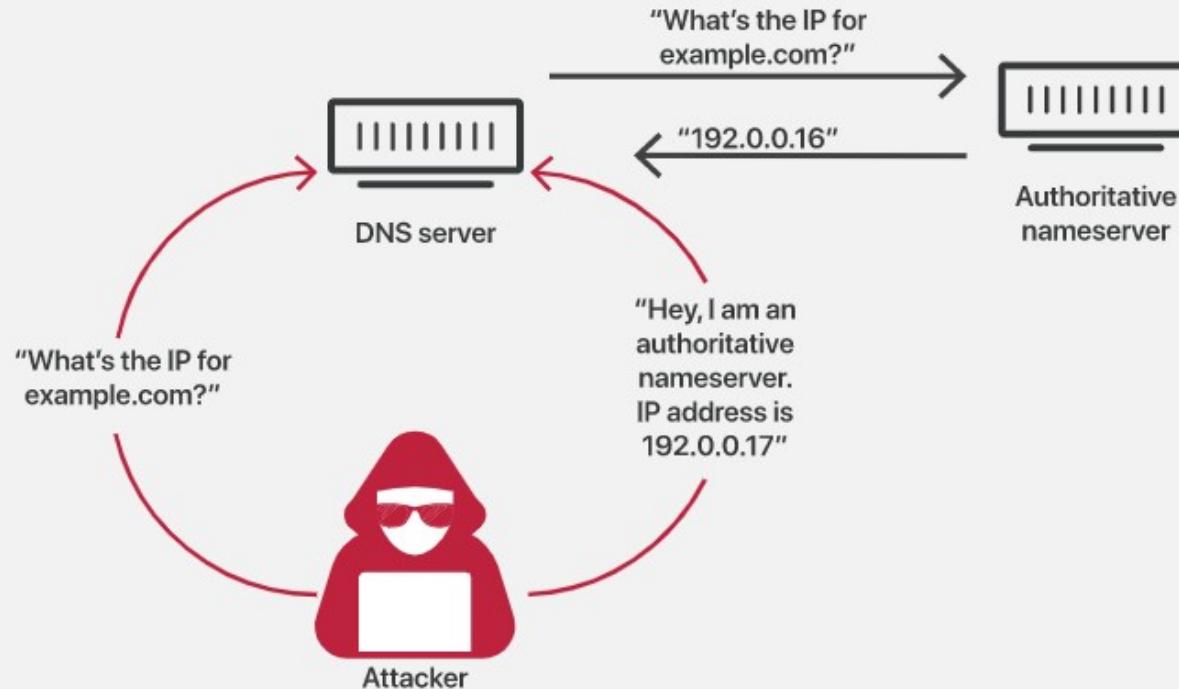


# DNS SPOOFING



# DNS SPOOFING

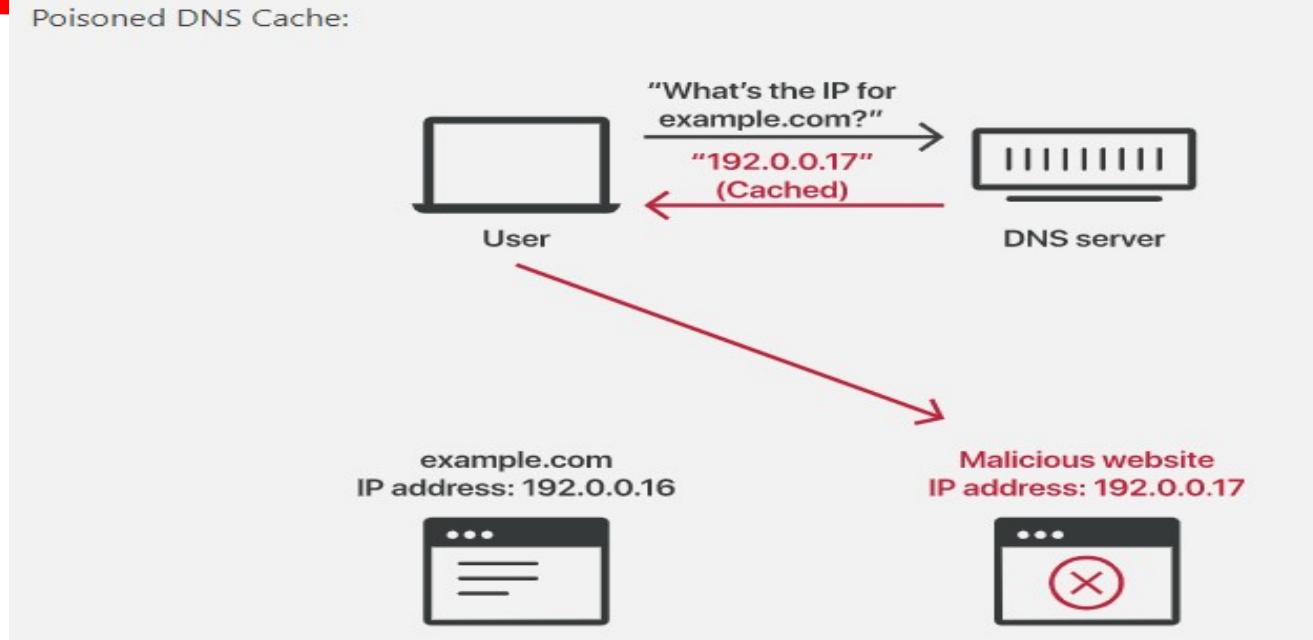
DNS Cache Poisoning Process:



Attackers can poison DNS caches by impersonating [DNS nameservers](#), making a request to a DNS resolver, and then forging the reply when the DNS resolver queries a nameserver. This is possible because DNS servers use [UDP](#) instead of [TCP](#), and because currently there is no verification for DNS information.



# DNS SPOOFING



Instead of using TCP, which requires both communicating parties to perform a 'handshake' to initiate communication and verify the identity of the devices, DNS requests and responses use UDP, or the User Datagram Protocol. With UDP, there is no guarantee that a connection is open, that the recipient is ready to receive, or that the sender is who they say they are. UDP is vulnerable to forging for this reason – an attacker can send a message via UDP and pretend it's a response from a legitimate server by forging the header data.



# PORT SCANNING

---

1. A port scan is a process which identifies “open doors” to a computer.
2. Ports are points at which information comes and goes from a computer, so by scanning for open ports, attackers can find weakened pathways with which to enter your computer.
3. Port scanning is one of the most popular techniques attackers use to discover services they can exploit to break into your computer system
4. Port scanning provides the following information to attackers:
  - What services are running
  - Which users own the services
  - If anonymous logins are allowed
  - What network services require authentication
5. During a port scan, hackers send a message to each port, one at a time. The response they receive from each port determines whether it's being used and reveals potential weaknesses.



# PORT SCANNING – How does it work

---

1. Port scans send requests to every port, asking to connect to a network. The scan then makes note of the ports that respond and which seem vulnerable.
2. Once the attacker has determined vulnerable ports in a network, the scan will classify ports into three categories:
  - **Open:** The host responds, announcing it is listening and open to requests. An open port means it's a path to attack the network.
  - **Closed:** The host responds, but notes there is no application listening. Often, hackers will come back to scan again in case it opens up.
  - **Filtered:** The host does not respond to a request. This could mean the packet was dropped due to congestion or a firewall.



# Module 5: Network Security and Applications

---

5.1

- Network Security Basics
- TCP/IP Vulnerabilities
- Packet Sniffing
- ARP Spoofing, DNS Spoofing
- Port Scanning, TCP Syn flood

5.3

- Internet Security Protocols: SSL, IPSEC
- Secure Email: PGP, Firewalls
- IDS and Its types
- Honey pots

5.2

- Denial of Service (DOS)
- Classic DOS attacks
- Source Address Spoofing
- ICMP Flood, SYN flood, UDP flood
- Distributed DOS, Defences against DOS Attacks



# DENIAL OF SERVICE (DoS)

---

1. A Denial-of-Service (DoS) attack is an attack meant to shut down a machine or network, making it inaccessible to its intended users.
2. DoS attacks accomplish this by flooding the target with traffic, or sending it information that triggers a crash.
3. In both instances, the DoS attack deprives legitimate users (i.e. employees, members, or account holders) of the service or resource they expected.



# DENIAL OF SERVICE (DoS)

---

1. Victims of DoS attacks often target web servers of high-profile organizations such as banking, commerce, and media companies, or government and trade organizations.
2. Though DoS attacks do not typically result in the theft or loss of significant information or other assets, they can cost the victim a great deal of time and money to handle.
3. There are two general methods of DoS attacks: **flooding services or crashing services**.
4. Flood attacks occur when the system receives too much traffic for the server to buffer, causing them to slow down and eventually stop.
5. Popular flood attacks include: **ICMP Flood, SYN Flood.**



# Internet Control Message Protocol (ICMP) Flood

---

1. ICMP is mainly used to determine whether or not data is reaching its intended destination in a timely manner.
2. Commonly, the ICMP protocol is used on network devices, such as routers.
3. ICMP is crucial for error reporting and testing, but it can also be used in distributed denial-of-service (DDoS) attacks.
4. Ping flood, also known as ICMP flood, is a common Denial of Service (DoS) attack in which an attacker takes down a victim's computer by overwhelming it with ICMP echo requests, also known as pings.
5. The ICMP echo-request and echo-reply messages are commonly used for the purpose of performing a ping.



# Ping Flood Attack (ICMP Attack)

---

1. Ping Flood is a Denial of Service Attack. In this attack, the attacker sends a large number of ICMP Echo request or ping packets to the targeted victim's IP address.
2. As a result, victim's machine starts responding to each ICMP packet by sending an ICMP Echo Reply Packet
3. Now, the victim's machine takes twice the bandwidth of the attacker- Once for receiving the packets and once for sending replies.
4. Thus, if the attacker has a much higher bandwidth than the victim, the victim's machine will get flooded with network Traffic
5. The victim's machine will consume a large number of CPU Cycles and notices a significant slow down
6. This attack is called Ping Flood



# SYN Flood

---

1. TCP SYN flood (a.k.a. SYN flood) is a type of Distributed Denial of Service (DDoS) attack that exploits part of the normal TCP three-way handshake to consume resources on the targeted server and render it unresponsive.
  2. Under normal conditions, TCP connection exhibits three distinct processes in order to make a connection.
    - ✓ First, the client sends a SYN packet to the server in order to initiate the connection.
    - ✓ The server then responds to that initial packet with a SYN/ACK packet, in order to acknowledge the communication.
    - ✓ Finally, the client returns an ACK packet to acknowledge the receipt of the packet from the server. After completing this sequence of packet sending and receiving, the TCP connection is open and able to send and receive data.
- 

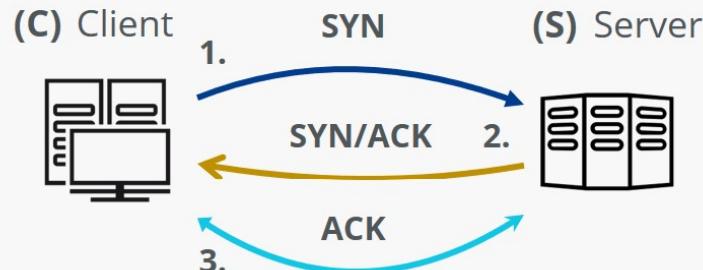


# SYN Flood

## SYN Flood

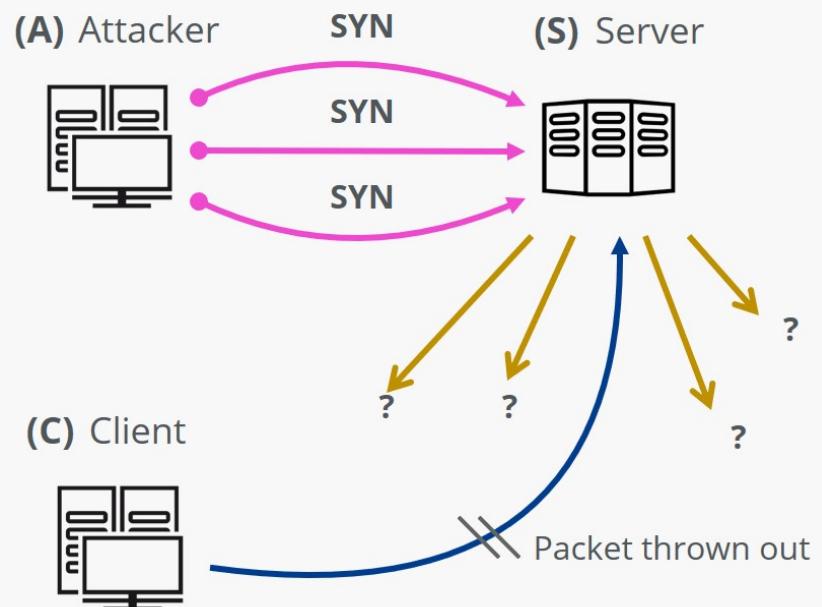
### How it works

TCP three-way handshake



Legend:  
→ = SYN packet  
→ = SYN/ACK packet  
→ = ACK packet

SYN Flood attack



# SYN Flood

---

1. The attacker sends a high volume of SYN packets to the targeted server, often with spoofed IP addresses.
2. The server then responds to each one of the connection requests and leaves an open port ready to receive the response.
3. While the server waits for the final ACK packet, which never arrives, the attacker continues to send more SYN packets. The arrival of each new SYN packet causes the server to temporarily maintain a new open port connection for a certain length of time, and once all the available ports have been utilized the server is unable to function normally.



# Distributed Denial of Service Attack

---

1. A distributed denial-of-service (DDoS) attack is one of the most powerful weapons on the internet.
2. When we hear about a website being “brought down by hackers,” it generally means it has become a victim of a DDoS attack.
3. In short, this means that hackers have attempted to make a website or computer unavailable by flooding or crashing the website with too much traffic.
4. In a DoS attack, it’s one system that is sending the malicious data or requests; a DDoS attack comes from multiple systems.



# Distributed Denial of Service Attack

---

1. DDoS attacks are carried out with networks of Internet-connected machines.
  2. These networks consist of computers and other devices (such as IoT devices) which have been infected with malware, allowing them to be controlled remotely by an attacker. These individual devices are referred to as bots (or zombies), and a group of bots is called a botnet.
  3. Once a botnet has been established, the attacker is able to direct an attack by sending remote instructions to each bot.
  4. When a victim's server or network is targeted by the botnet, each bot sends requests to the target's IP address, potentially causing the server or network to become overwhelmed, resulting in a denial-of-service to normal traffic.
  5. Because each bot is a legitimate Internet device, separating the attack traffic from normal traffic can be difficult.
- 



# Distributed Denial of Service Attack

---

1. A DDoS is a cyberattack on a server, service, website, or network floods it with Internet traffic. If the traffic overwhelms the target, its server, service, website, or network is rendered inoperable.
2. Network connections on the Internet consist of different layers of the Open Systems Interconnection (OS) model. Different types of DDoS attacks focus on particular layers. A few examples:
  - Layer 3, the Network layer. Attacks are known as Smurf Attacks, ICMP Floods, and IP/ICMP Fragmentation.
  - Layer 4, the Transport layer. Attacks include SYN Floods, UDP Floods, and TCP Connection Exhaustion.
  - Layer 7, the Application layer. Mainly, HTTP-encrypted attacks.



# Module 5: Network Security and Applications

5.1

- Network Security Basics
- TCP/IP Vulnerabilities
- Packet Sniffing
- ARP Spoofing, DNS Spoofing
- Port Scanning, TCP Syn flood

5.2

- Denial of Service (DOS)
- Classic DOS attacks
- Source Address Spoofing
- ICMP Flood, SYN flood, UDP flood
- Distributed DOS, Defences against DOS Attacks

5.3

- Internet Security Protocols: SSL, IPSEC
- Secure Email: PGP, Firewalls
- IDS and Its types
- Honey pots



# Internet Security

---

1. Internet security refers to securing communication over the internet. It includes specific security protocols such as:
  - ✓ Internet Security Protocol (IPSec)
  - ✓ Secure Socket Layer (SSL)

## Secure Socket Layer (SSL)

It is a security protocol developed by Netscape Communications Corporation.). It provides security at transport layer. It addresses the following security issues:

1. Privacy
2. Integrity
3. Authentication

## Internet Security Protocol (IPSec)

It consists of a set of protocols designed by Internet Engineering Task Force (IETF). It provides security at network level and helps to create authenticated and confidential packets for IP layer.

---



# SSL (Secure Socket Layer) – How it works

---

1. In order to provide a high degree of privacy, SSL encrypts data that is transmitted across the web. This means that anyone who tries to intercept this data will only see a garbled mix of characters that is nearly impossible to decrypt.
2. SSL initiates an authentication process called a handshake between two communicating devices to ensure that both devices are really who they claim to be.
3. SSL also digitally signs data in order to provide data integrity, verifying that the data is not tampered with before reaching its intended recipient.



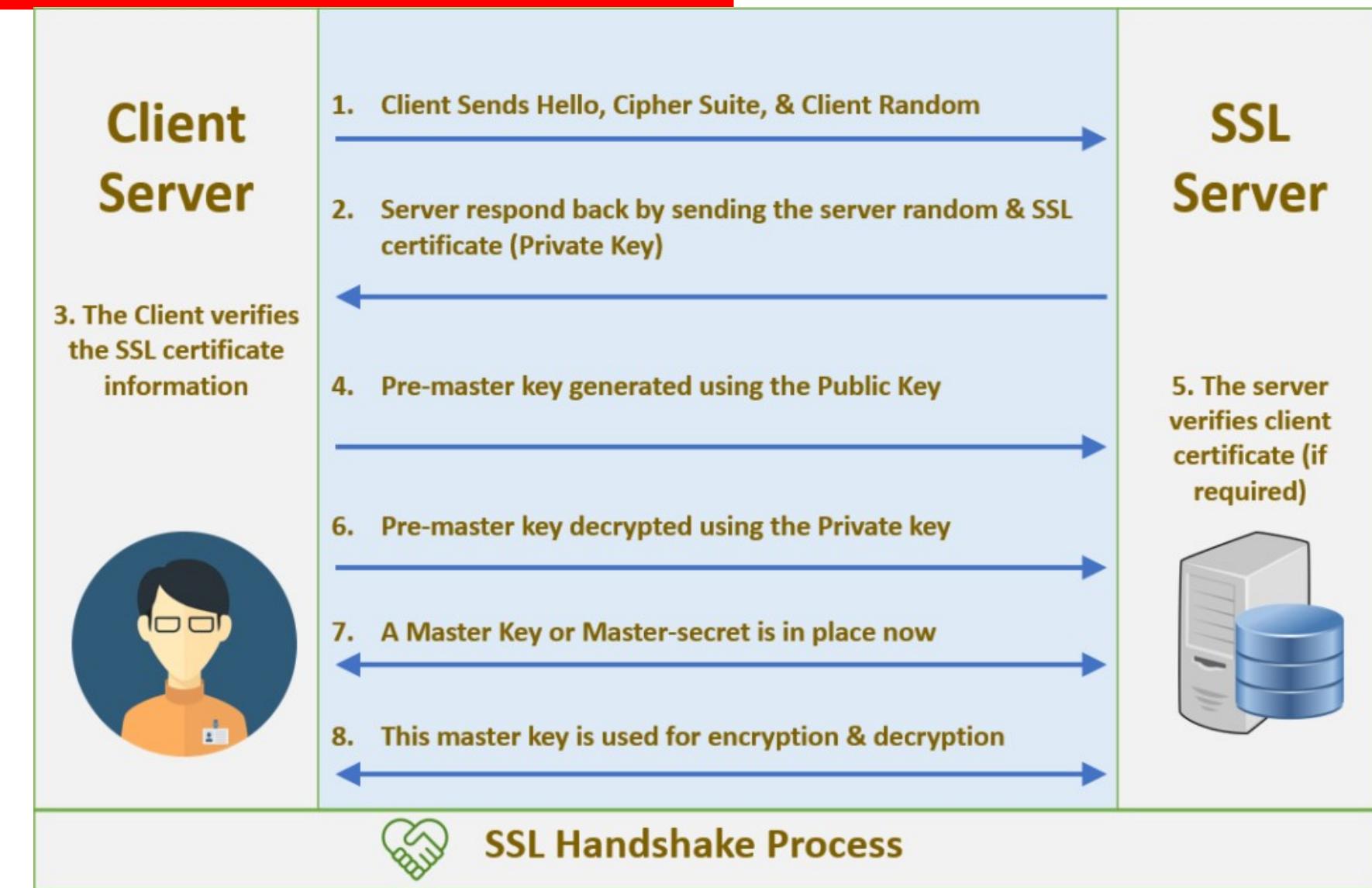
# SSL (Secure Socket Layer) – Handshake

---

1. The SSL/TLS handshake involves a series of steps through which both the parties – client and server, validate each other and start communicating through the secure SSL/TLS tunnel.
2. A TLS handshake takes place whenever a user navigates to a website over HTTPS and the browser first begins to query the website's origin server.
3. During the course of a TLS handshake, the client and server together will do the following:
  - ✓ Specify which version of TLS (TLS 1.0, 1.2, 1.3, etc.) they will use
  - ✓ Decide on which cipher suites (see below) they will use
  - ✓ Authenticate the identity of the server via the server's public key and the SSL certificate authority's digital signature
  - ✓ Generate session keys in order to use symmetric encryption after the handshake is complete



# SSL (Secure Socket Layer) – Handshake



# IPSec

---

1. Within the term "IPsec," "IP" stands for "Internet Protocol" and "sec" for "secure."
2. The Internet Protocol is the main routing protocol used on the Internet; it designates where data will go using IP addresses.
3. IPsec is secure because it adds encryption\* and authentication to this process.
4. The **IP security (IPSec)** is an Internet Engineering Task Force (IETF) standard suite of protocols between 2 communication points across the IP network that provide data authentication, integrity, and confidentiality.
5. It also defines the encrypted, decrypted and authenticated packets. The protocols needed for secure key exchange and key management are defined in it.



# IPSec - USES

---

1. To encrypt application layer data.
2. To provide security for routers sending routing data across the public internet.
3. To provide authentication without encryption, like to authenticate that the data originates from a known sender.
4. To protect network data by setting up circuits using IPsec tunneling in which all data is being sent between the two endpoints is encrypted, as with a Virtual Private Network(VPN) connection.



# IPSec – Protocols used

---

The following protocols make up the IPsec suite:

- 1. Authentication Header (AH):** The AH protocol ensures that data packets are from a trusted source and that the data has not been tampered with, like a tamper-proof seal on a consumer product. These headers do not provide any encryption; they do not help conceal the data from attackers.
- 2. Encapsulating Security Protocol (ESP):** ESP encrypts the IP header and the payload for each packet — unless transport mode is used, in which case it only encrypts the payload. ESP adds its own header and a trailer to each data packet.
- 3. Security Association (SA):** SA refers to a number of protocols used for negotiating encryption keys and algorithms. One of the most common SA protocols is Internet Key Exchange (IKE).



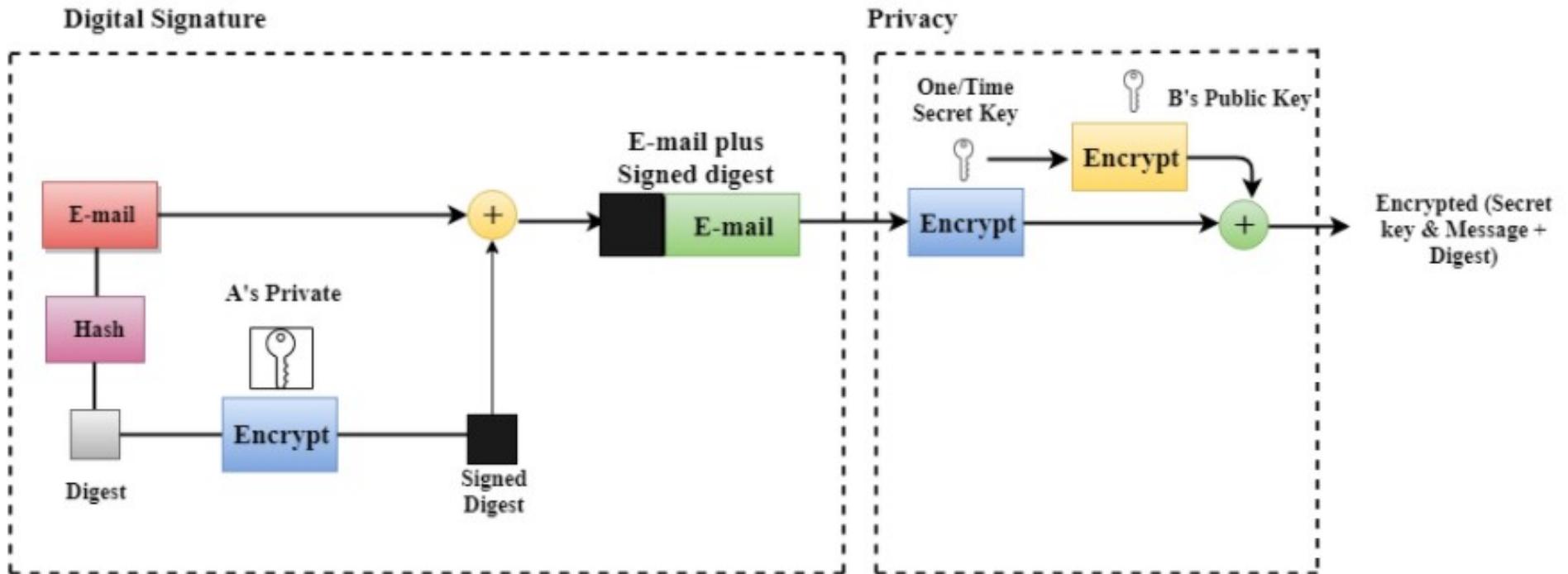
# PGP: Pretty Good Privacy

---

1. Pretty Good Privacy (PGP) is an encryption system used for both sending encrypted emails and encrypting sensitive files.
2. The popularity of PGP is based on two factors.
  - ✓ It is available as freeware, and so spread rapidly among users who wanted an extra level of security for their email messages.
  - ✓ PGP uses both symmetric encryption and public-key encryption, it allows users who have never met to send encrypted messages to each other without exchanging private encryption keys.



# Steps taken by PGP to create secure e-mail at the sender



# Steps taken by PGP to create secure e-mail at the sender

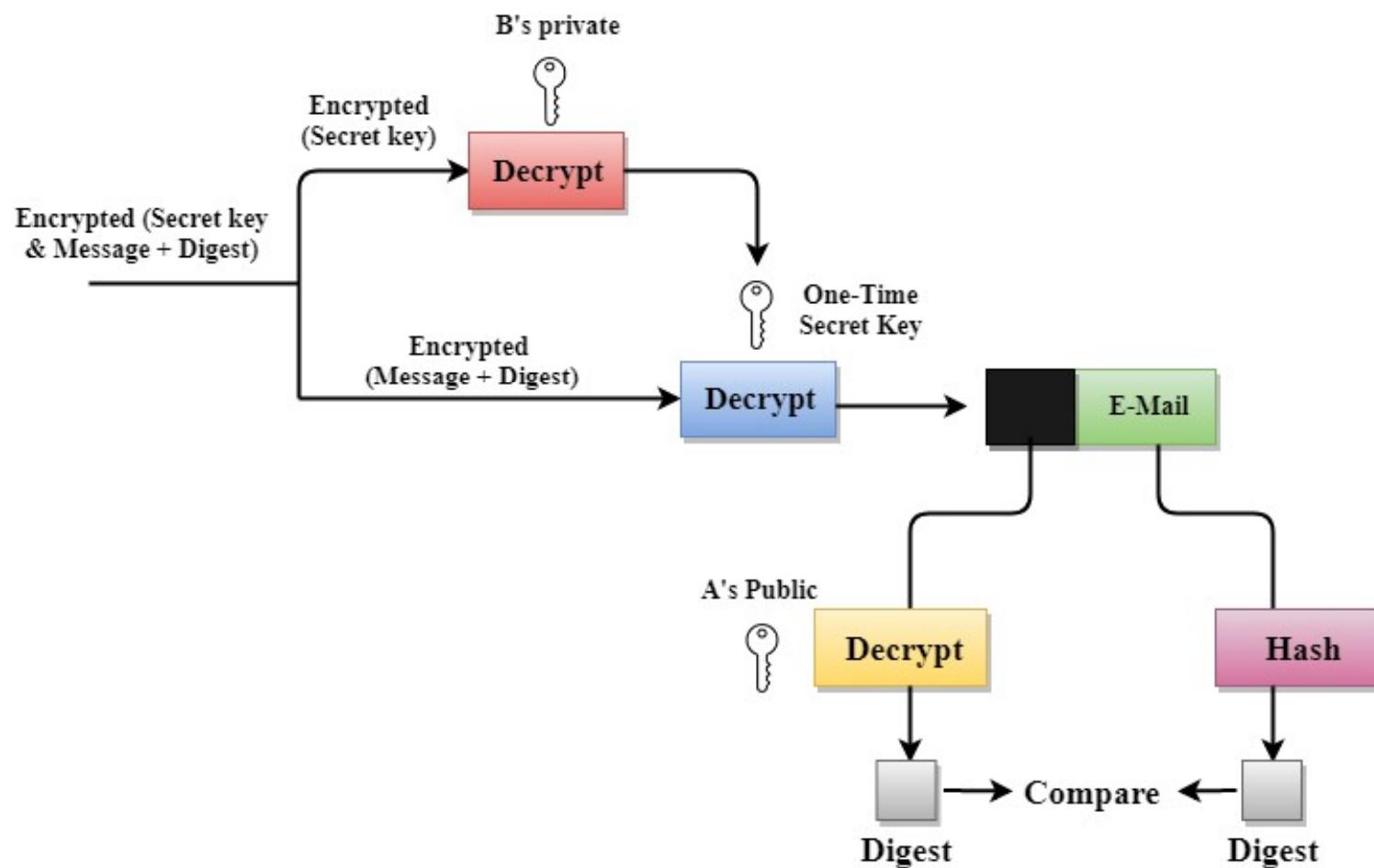
---

1. The e-mail message is hashed by using a hashing function to create a digest.
2. The digest is then encrypted to form a signed digest by using the sender's private key, and then signed digest is added to the original email message.
3. The original message and signed digest are encrypted by using a one-time secret key created by the sender.
4. The secret key is encrypted by using a receiver's public key.
5. Both the encrypted secret key and the encrypted combination of message and digest are sent together.



# Steps taken by PGP to create secure e-mail at the receiver

## PGP at the Receiver site (B)



# Steps taken by PGP to create secure e-mail at the receiver

---

1. The receiver receives the combination of encrypted secret key and message digest is received.
2. The encrypted secret key is decrypted by using the sender's private key to get the one-time secret key.
3. The secret key is then used to decrypt the combination of message and digest.
4. The digest is decrypted by using the sender's public key, and the original message is hashed by using a hash function to create a digest.
5. Both the digests are compared if both of them are equal means that all the aspects of security are preserved.



# Disadvantages of PGP Encryption

---

- 1. The Administration is difficult:** The different versions of PGP complicate the administration.
  - 2. Compatibility issues:** Both the sender and the receiver must have compatible versions of PGP.
  - 3. Complexity:** PGP is a complex technique. Other security schemes use symmetric encryption that uses one key or asymmetric encryption that uses two different keys. PGP uses a hybrid approach that implements symmetric encryption with two keys. PGP is more complex, and it is less familiar than the traditional symmetric or asymmetric methods.
  - 4. No Recovery:** PGP does not offer such a special program for recovery; encryption methods are very strong so, it does not retrieve the forgotten passwords results in lost messages or lost files.
- 



# Intrusion Detection System (IDS)

---

1. An Intrusion Detection System (IDS) is a system that monitors network traffic for suspicious activity and issues alerts when such activity is discovered.
2. It is a software application that scans a network or a system for harmful activity or policy breaching.
3. Any malicious venture or violation is normally reported either to an administrator or collected centrally using a Security Information and Event Management (SIEM) system.
4. A SIEM system integrates outputs from multiple sources and uses alarm filtering techniques to differentiate malicious activity from false alarms.



# Intrusion Detection System (IDS)

---

1. Although intrusion detection systems monitor networks for potentially malicious activity, they are also disposed to false alarms.
2. Hence, organizations need to fine-tune their IDS products when they first install them. It means properly setting up the intrusion detection systems to recognize what normal traffic on the network looks like as compared to malicious activity.
3. Intrusion prevention systems also monitor network packets inbound the system to check the malicious activities involved in it and at once sends the warning notifications.



# Intrusion Detection System (IDS) – How do they work

---

1. Intrusion detection systems are used to detect anomalies with the aim of catching hackers before they do real damage to a network. They can be either network- or host-based.
2. A host-based intrusion detection system is installed on the client computer, while a network-based intrusion detection system resides on the network.
3. Intrusion detection systems work by either looking for signatures of known attacks or deviations from normal activity. These deviations or anomalies are pushed up the stack and examined at the protocol and application layer. They can effectively detect events such as Christmas tree scans and domain name system (DNS) poisonings.
4. An IDS may be implemented as a software application running on customer hardware or as a network security appliance. Cloud-based intrusion detection systems are also available to protect data and systems in cloud deployments.

Is is a unique arrangement of information that can be used to identify an attacker's attempt to exploit a known operating system or application vulnerability.



# **Intrusion Detection System (IDS) vs. Intrusion Prevention System(IPS)**

---

1. An IDS can be contrasted with an **Intrusion Prevention System (IPS)**, which monitors network packets for potentially damaging network traffic, like an IDS, but has the primary goal of preventing threats once detected, as opposed to primarily detecting and recording threats.



# Intrusion Detection System (IDS) vs. Intrusion Prevention System(IPS)

## IDS vs. IPS

Most organizations have either an IDS or an IPS, and many have both as part of their security information and event management framework.

	IDS	IPS
NAME	Intrusion detection system	Intrusion prevention system
DESCRIPTION	A system that monitors network traffic for suspicious activity and alerts users when such activity is discovered.	A system that monitors network traffic and alerts for suspicious activity, like an IDS, but also takes preventative action against suspicious activity.
LOCATION	A host-based intrusion detection system is installed on the client computer. A network-based intrusion detection system resides on the network.	Located between a company's firewall and the rest of its network.
USE	Warns of suspicious activity taking place, but it doesn't prevent it.	Warns of suspicious activity taking place and prevents it.
FALSE POSITIVE	IDS false positives are usually just a minor inconvenience. Although the IDS incorrectly labels legitimate traffic as malicious, it does not prevent the traffic from entering the network.	IPS false positives can be more serious. When an IPS mistakes legitimate traffic for a threat, it stops the legitimate traffic from entering the network, which could impact any part of the organization, not just the IT team.



# **Types of Intrusion Detection System (IDS)**

---

## **1. Network Intrusion Detection System (NIDS)**

Network intrusion detection systems (NIDS) are set up at a planned point within the network to examine traffic from all devices on the network. It can monitor inbound and outbound traffic to and from all the devices on the network.. Once an attack is identified or abnormal behavior is observed, the alert can be sent to the administrator. An example of an NIDS is installing it on the subnet where firewalls are located in order to see if someone is trying crack the firewall.

## **2. Host Intrusion Detection System (HIDS):** runs on all computers or devices in the network with direct access to both the internet and the enterprise's internal network. A HIDS may also be able to identify malicious traffic that originates from the host itself, such as when the host has been infected with malware and is attempting to spread to other systems.

---



## **Types of Intrusion Detection System (IDS)**

---

- 3. Signature-based Intrusion Detection System (SIDS):** monitors all the packets traversing the network and compares them against a database of attack signatures or attributes of known malicious threats, much like antivirus software.
  
- 4. Anomaly-based Intrusion Detection System (AIDS):** monitors network traffic and compares it against an established baseline to determine what is considered normal for the network with respect to bandwidth, protocols, ports and other devices.



# HONEY POTS

---

1. A **honeypot** is a network-attached system set up as a decoy to lure cyber attackers and detect, deflect and study hacking attempts to gain unauthorized access to information systems.
2. The principle behind them is simple: **Don't go looking for attackers.** Prepare something that would attract their interest — the **honeypot** — **and then wait for the attackers to show up.**
3. Like mice to cheese-baited mousetraps, cybercriminals are attracted to honeypots — not because they're honeypots.
4. The bad guys think the honeypot is a legitimate target, something worthy of their time. That's because the bait includes applications and data that simulate a real computer system.



# HONEY POTS

---

1. The function of a honeypot is to represent itself on the internet as a potential target for attackers -- usually, a server or other high-value asset -- and to gather information and notify defenders of any attempts to access the honeypot by unauthorized users.
2. The honeypot looks like a real computer system, with applications and data, fooling cybercriminals into thinking it's a legitimate target.
3. For example, a honeypot could mimic a company's customer billing system - a frequent target of attack for criminals who want to find credit card numbers.
4. Once the hackers are in, they can be tracked, and their behavior assessed for clues on how to make the real network more secure.

In a nutshell, honeypots help organizations:

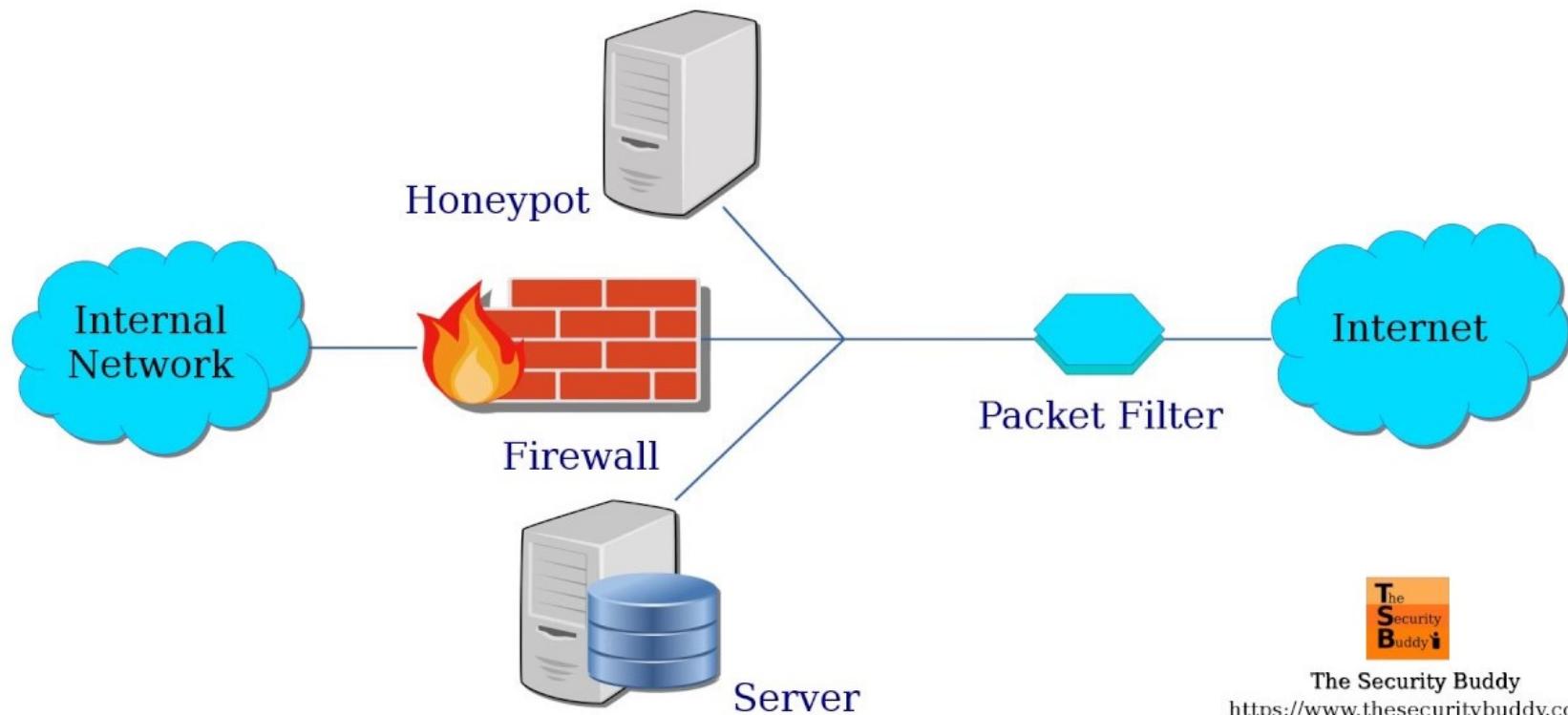
- a) Assess the latest trends in attacks,
- b) Understand where cyber attacks arise, and
- c) Better frame security policies to mitigate future risks.



# HONEY POTS

---

## Honeypot



The Security Buddy

<https://www.thesecuritybuddy.com/>



## HONEY POTS – How do they work

---

1. Generally, a honeypot operation consists of a computer, applications and data that simulate the behavior of a real system that would be attractive to attackers, such as a financial system, internet of things (IoT) devices, or a public utility or transportation network. It appears as part of a network but is actually isolated and closely monitored. Because there is no reason for legitimate users to access a honeypot, any attempts to communicate with it are considered hostile.
  
2. Honeypots are often placed in a demilitarized zone (DMZ) on the network. That approach keeps it isolated from the main production network, while still being a part of it. In the DMZ, a honeypot can be monitored from a distance while attackers access it, minimizing the risk of the main network being breached.



# HONEY POTS – TYPES

---

Honeypots are typically categorized in one of two ways — either based on their interaction levels or the types of threats they're able to detect.

## **Types of Honeypots Based on Interaction Level and Complexity**

### **High-Interaction Honeypots**

These honeypots imitate real-world systems and applications with actual services, functions, and operating systems involving high levels of interactivity (though less than pure honeypots). Setting up high-interaction honeypots is a complex and resource-intensive process. It gives extensive details about how an attack progresses and how payloads execute in a network. However, since there are actual operating systems and services involved, the chance of infection is higher if the hackers are able to compromise the honeypots and use them gain access to your organization's real production environment.

---



# HONEY POTS – TYPES

---

## Types of Honeypots Based on Interaction Level and Complexity

### Medium-Interaction Honeypots

Come with expanded capabilities compared to low interaction honeypots but reduced implementation complexities than high interaction honeypots. They imitate the application layer but don't have their own operating system. Organizations typically deploy these types of honeypots to stall attackers to give them time to respond to attacks.

### Low-Interaction Honeypots

Low-interaction honeypots allow partial interaction with systems since they run limited emulated services with restricted functionality as would be typically expected from a server. Though these are the easiest to set up and maintain, they run the risk of coming across as inauthentic targets to potential attackers. These types of honeypots serve as an early detection mechanism, and organizations commonly use them in production environments.



# HONEY POTS – TYPES

---

## Some Other Types of Honeypots

**Malware Honeypots** — These types of honeypots detect malware based on known replication techniques and propagation vectors.

**Database Honeypots** — Since attacks on databases like SQL injections are fairly common, you can use database honeypots to distract an attacker from your legitimate database servers by setting up decoy databases.

**Client Honeypots** — These honeypots typically act as servers, listening in for incoming connections. Client honeypots actively engage with malicious servers that attack clients. They pose as a client to monitor and record any modifications.

**Email Honeypots** — Email honeypots are a list of email addresses used by email service providers to detect spammers. Typically, accounts inactive over a long period of time are used for this purpose.

**Spider Honeypots** — These honeypots are used to trap web-crawlers by creating fake web pages and links only reachable by crawlers. Detecting these crawlers can be useful in blocking bot activity.



# HONEY POTS – TYPES

---

## Types of Honeypots Based on Purpose

### Research Honeypots

These honeypots are deployed and used by researchers to gain a better understanding of attack techniques, motivations, information about malware strains in the wild, and security vulnerabilities.

This is done to specifically use the knowledge gained to make informed decisions about:

- ✓ Defense strategies,
- ✓ Patching prioritizations,
- ✓ Future security investments, and
- ✓ Identifying and developing new security solutions.

### Production Honeypots

Production honeypots are placed within your organization's internal network with other production servers. Though the intention is similar in terms of gaining insights about active attacks, it is typically less complex than research honeypots with lesser data. It is primarily deployed to identify active attacks on the internal network and distract or misdirect hackers from attacking your legitimate servers

---



## FIREWALL

---

1. A firewall is a network security device that monitors incoming and outgoing network traffic and permits or blocks data packets based on a set of security rules.
  2. Its purpose is to establish a barrier between your internal network and incoming traffic from external sources (such as the internet) in order to block malicious traffic like viruses and hackers.
  3. Firewalls have been a first line of defense in network security
  4. They establish a barrier between secured and controlled internal networks that can be trusted and untrusted outside networks, such as the Internet.
  5. A firewall can be hardware, software, or both.
  6. A software firewall is a program installed on each computer and regulates traffic through port numbers and applications, while a physical firewall is a piece of equipment installed between your network and gateway.
- 



## **FIREWALL – How does they work**

---

1. Firewalls carefully analyze incoming traffic based on pre-established rules and filter traffic coming from unsecured or suspicious sources to prevent attacks. Firewalls guard traffic at a computer's entry point, called ports, which is where information is exchanged with external devices.
2. For example, "Source address 172.18.1.1 is allowed to reach destination 172.18.2.1 over port 22."



## FIREWALL – Types

---

1. **Proxy Firewall:** An early type of firewall device, a proxy firewall serves as the gateway from one network to another for a specific application. Proxy servers can provide additional functionality such as content caching and security by preventing direct connections from outside the network. However, this also may impact throughput capabilities and the applications they can support.
  
  2. **Stateful inspection firewall:** Now thought of as a “traditional” firewall, a stateful inspection firewall allows or blocks traffic based on state, port, and protocol. It monitors all activity from the opening of a connection until it is closed. Filtering decisions are made based on both administrator-defined rules as well as context, which refers to using information from previous connections and packets belonging to the same connection.
- 



## FIREWALL – Types

---

**3. Next Generation Firewall:** Combine traditional firewall technology with additional functionality, such as encrypted traffic inspection, intrusion prevention systems, anti-virus, and more. Most notably, it includes deep packet inspection (DPI). While basic firewalls only look at packet headers, deep packet inspection examines the data within the packet itself, enabling users to more effectively identify, categorize, or stop packets with malicious data.



# Cryptography and System Security (CSS)

## Course Code: CSC 604



### Subject Incharge

Ankita Karia  
Assistant Professor  
Room No. 421  
email: [ankitakaria@sfit.ac.in](mailto:ankitakaria@sfit.ac.in)



St. Francis Institute of Technology  
Department of Computer Engineering

16 April 2021

Cryptography and System Security  
Ms. Ankita Karia 1

# Module 6: System Security

---

## 6.1

- Software Vulnerabilities
- Buffer Overflow, Format string, cross-site scripting,
- SQL injection,
- Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, Rootkits



# Software Vulnerability

---

1. A software vulnerability is a glitch, flaw, or weakness present in the software or in an OS (Operating System).
2. The severity of software vulnerabilities advances at an exponential rate.
3. Of course, all systems include vulnerabilities. The thing is whether or not they're exploited to cause damage.

Software vulnerabilities are explained by three ideal factors. These are:

**Existence** – The existence of a vulnerability in the software.

**Access** – The possibility that hackers gain access to the vulnerability.

**Exploit** – The capability of the hacker to take advantage of that vulnerability via tools or with certain techniques.



# Most Common Software Vulnerabilities

---

According to the OWASP Top 10, here are the most common vulnerabilities:

## 1. Insufficient Logging and Monitoring

Insufficient logging and monitoring processes are dangerous. This leaves your data vulnerable to tampering, extraction, or even destruction.

## 2. Injection Flaws

Injection flaws occur when untrusted data is sent as part of a command or query. The attack can then trick the targeted system into executing unintended commands. An attack can also provide untrustworthy agents access to protected data.

## 3. Sensitive Data Exposure

Sensitive data — such as addresses, passwords, and account numbers — must be properly protected. If it isn't, untrustworthy agents take advantage of the vulnerabilities to gain access.

---



# Most Common Software Vulnerabilities

---

## 4. Using Components with Known Vulnerabilities

Components are made up of libraries, frameworks, and other software modules. Often, the components run on the same privileges as your application. If a component is vulnerable, it can be exploited by an untrustworthy agent. This causes serious data loss or server takeover.

## 5. Cross-Site Scripting (XSS) Flaws

Untrustworthy agents can take advantage of cross-site scripting flaws to execute their own scripts in the targeted system. In general, cross-site scripting flaws happen in one-of-two ways:

- a. Whenever an application includes untrusted data in a new web page without proper validation.
  - b. Whenever an existing webpage is updated with user-supplied data using a browser API that can create HTML or JavaScript.
- 



# Most Common Software Vulnerabilities

---

## 6. Broken Authentication

Authentication and session management application functions need to be implemented correctly. If they aren't, it creates a software vulnerability that can be exploited by untrustworthy agents to gain access to personal information.

## 7. Broken Access Control

User restrictions must be properly enforced. If they are broken, it can create a software vulnerability. Untrustworthy agents can exploit that vulnerability.

## 8. XML External Entities (XXE)

XML is a popular data format that is used in web services, documents, and image files. You need an XML parser to understand XML data. But if it's poorly configured and the XML input that contains a reference to an external entity, it's dangerous. An untrustworthy agent can cause a DoS.



# Most Common Software Vulnerabilities

---

**9. Security Misconfiguration:** Security misconfigurations are often the result of:

- a. Insecure default configurations.
- b. Incomplete or impromptu configurations.
- c. Open Cloud storage.
- d. Misconfigured HTTP headers.
- e. Wordy error messages that contain sensitive information.

## 10. Insecure Deserialization

Deserialization flaws often result in remote code execution. This enables untrustworthy agents to perform replay, [injection](#), and privilege escalation attacks.



# Buffer Overflow Attack – WHAT IS BUFFER

---

1. A buffer, or data buffer, is an area of physical memory storage used to temporarily store data while it is being moved from one place to another. These buffers typically live in RAM memory.
2. Computers frequently use buffers to help improve performance; most modern hard drives take advantage of buffering to efficiently access data, and many online services also use buffers.
3. For example, buffers are frequently used in online video streaming to prevent interruption. When a video is streamed, the video player downloads and stores perhaps 20% of the video at a time in a buffer and then streams from that buffer. This way, minor drops in connection speed or quick service disruptions won't affect the video stream performance.



**BUFFER  
OVERFLOW  
ATTACKS**



St. Francis Institute of Technology  
Department of Computer Engineering

16 April 2021

Cryptography and System Security  
Ms. Ankita Karia

# Buffer Overflow Attack – WHAT IS BUFFER

---

1. Buffers are designed to contain specific amounts of data. Unless the program utilizing the buffer has built-in instructions to discard data when too much is sent to the buffer, the program will overwrite data in memory adjacent to the buffer.
2. Buffer overflows can be exploited by attackers to corrupt software.
3. Despite being well-understood, buffer overflow attacks are still a major security problem that torment cyber-security teams.
4. In 2014 a threat known as ‘heartbleed’ exposed hundreds of millions of users to attack because of a buffer overflow vulnerability in SSL software.



# Buffer Overflow Attack – How do attackers exploit buffer overflows?

---

1. An attacker can deliberately feed a carefully crafted input into a program that will cause the program to try and store that input in a buffer that isn't large enough, overwriting portions of memory connected to the buffer space.
2. If the memory layout of the program is well-defined, the attacker can deliberately overwrite areas known to contain executable code.
3. The attacker can then replace this code with his own executable code, which can drastically change how the program is intended to work.

For example if the overwritten part in memory contains a pointer (an object that points to another place in memory) the attacker's code could replace that code with another pointer that points to an exploit payload. This can transfer control of the whole program over to the attacker's code.



# Buffer Overflow Attack – Who is vulnerable to buffer overflow attacks?

---

1. Certain coding languages are more susceptible to buffer overflow than others.
2. C and C++ are two popular languages with high vulnerability, since they contain no built-in protections against accessing or overwriting data in their memory.
3. Windows, Mac OSX, and Linux all contain code written in one or both of these languages.

More modern languages like Java, PERL, and C# have built-in features that help reduce the chances of buffer overflow, but cannot prevent it altogether.



# Format String Attack

---

1. A Format String attack can occur when an input string's submitted data is evaluated as a command by the application.
2. Taking advantage of a Format String vulnerability, an attacker can execute code, read the Stack, or cause a segmentation fault in the running application – causing new behavior that compromise the security or the stability of the system.
3. Format String attacks alter the flow of an application.
4. They use string formatting library features to access other memory space. Vulnerabilities occurred when the user-supplied data is deployed directly as formatting string input for certain C/C++ functions



# Format String Attack

To understand the attack, it's necessary to understand the components that constitute it.

1. The **Format Function** is an ANSI C conversion function, like printf, fprintf, which converts a primitive variable of the programming language into a human-readable string representation.
2. The **Format String is the argument of the Format Function** and is an ASCII string which contains text and format parameters, like: **printf (“The magic number is: %d\n”, 1911);**
3. The **Format String Parameter**, like **%x %s** defines the type of conversion of the format function.

The attack could be executed when the application doesn't properly validate the submitted input. In this case, if a Format String parameter, like %x, is inserted into the posted data, the string is parsed by the Format Function, and the conversion specified in the parameters is executed. However, the Format Function is expecting more arguments as input, and if these arguments are not supplied, the function could read or write the stack.



# Format String Attack

**Table 2. Common parameters used in a Format String Attack.**

Parameters	Output	Passed as
%%	% character (literal)	Reference
%p	External representation of a pointer to void	Reference
%d	Decimal	Value
%c	Character	
%u	Unsigned decimal	Value
%x	Hexadecimal	Value
%s	String	Reference
%n	Writes the number of characters into a pointer	Reference



# Format String Attack

```
#include <stdio.h>  
void main(int argc, char **argv)  
{  
    // This line is safe  
    printf("%s\n", argv[1]);
```

→ **SAFE CODE**

The line `printf("%s", argv[1]);` in the example is safe, if you compile the program and run it as follows:

`./example "Hello World %s%s%s%s%s%s"`

The `printf` in the first line will not interpret the “%s%s%s%s%s%s” in the input string, and the output will be: “Hello World %s%s%s%s%s%s”



# Format String Attack

```
#include <stdio.h>  
void main(int argc, char **argv)  
{  
    // This line is vulnerable  
    printf(argv[1]);  
}
```

 **VULNERABLE CODE**

The printf in the above line will interpret the %s%s%s%s%s%s in the input string as a reference to string pointers, so it will try to interpret every %s as a pointer to a string, starting from the location of the buffer (probably on the Stack).

At some point, it will get to an invalid address, and attempting to access it will cause the program to crash.



# Format String Attack

---

An attacker can also use this to get information, not just crash the software.

For example, running:

```
./example "Hello World %p %p %p %p %p %p"
```

Will print the lines:

Hello World %p %p %p %p %p %p → **Output from safe code**

Hello World 000E133E 000E133E 0057F000 CCCCCCCC CCCCCCCC CCCCCCCC

↑ **Output from vulnerable code**

The values printed after the “Hello World” text, are the values on the stack of computer at the moment of running this example.

Also reading and writing to any memory location is possible in some conditions, and even code execution.



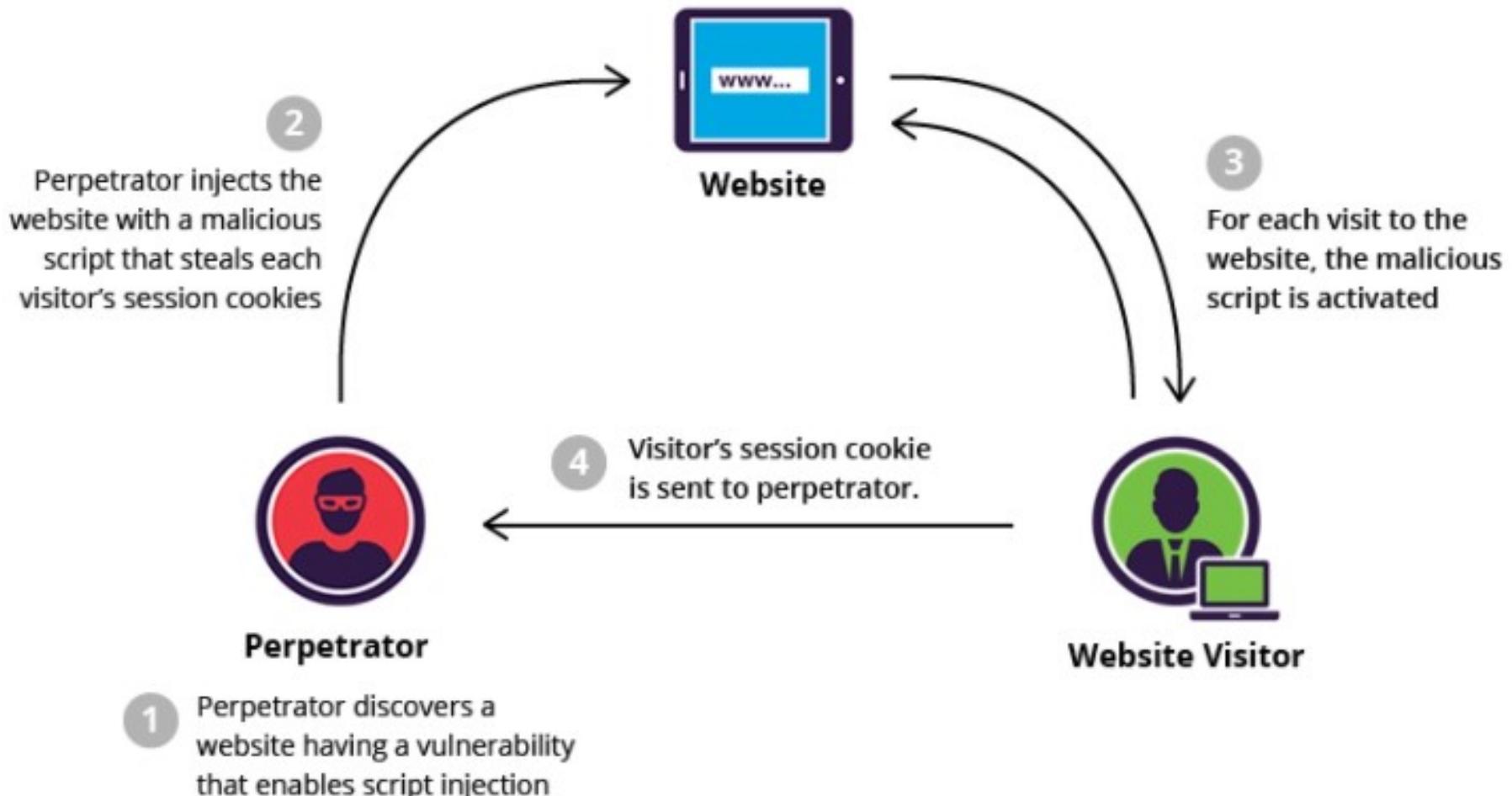
# Cross Site Scripting Attack (XSS Attack)

---

1. Cross site scripting (XSS) is a common attack vector that injects malicious code into a vulnerable web application.
  2. XSS differs from other web attack vectors (e.g., SQL injections), in that it does not directly target the application itself. Instead, the users of the web application are the ones at risk.
  3. Cross-site Scripting (XSS) is a client-side code injection attack.
  4. The attacker aims to execute malicious scripts in a web browser of the victim by including malicious code in a legitimate web page or web application.
  5. The actual attack occurs when the victim visits the web page or web application that executes the malicious code. The web page or web application becomes a vehicle to deliver the malicious script to the user's browser.
  6. Vulnerable vehicles that are commonly used for Cross-site Scripting attacks are forums, message boards, and web pages that allow comments.
- 



# Cross Site Scripting Attack (XSS Attack) – How does it work



# Cross Site Scripting Attack (XSS Attack) – How does it work

---

1. To successfully execute a stored XSS attack, a perpetrator has to locate a vulnerability in a web application and then inject malicious script into its server (e.g., via a comment field).
2. When attackers inject their own code into a web page, typically accomplished by exploiting a vulnerability on the website's software, they can then inject their own script, which is executed by the victim's browser.
3. Since the JavaScript runs on the victim's browser page, sensitive details about the authenticated user can be stolen from the session, essentially allowing a bad actor to target site administrators and completely compromise a website.
4. One of the most frequent targets are websites that allow users to share content, including blogs, social networks, video sharing platforms and message boards.
5. Every time the infected page is viewed, the malicious script is transmitted to the victim's browser.



# Cross Site Scripting Attack (XSS Attack) – Types

---

Depending on their goals, bad actors can use cross-site scripting in a number of different ways. Some of the most common types of attacks.

- 1. Stored (Persistent) Cross Site-Scripting**
- 2. Reflected Cross-Site Scripting**
- 3. Self Cross-Site Scripting**
- 4. Blind Cross-Site Scripting**
- 5. DOM-Based Cross-Site Scripting**



# Cross Site Scripting Attack (XSS Attack) – Types

---

**Stored (Persistent) Cross Site-Scripting:** Occurs when attackers stores their payload on a compromised server, causing the website to deliver malicious code to other visitors. This method only requires an initial action from the attacker and can compromise many visitors afterwards, this is the most dangerous and most commonly employed type of cross-site scripting. Examples of stored cross-site scripting attacks include the profile fields such as your username or email, which are saved on the server and displayed on your account page.



# Cross Site Scripting Attack (XSS Attack) – Types

---

## Reflected Cross-Site Scripting:

- Occurs when the payload is stored in the data sent from the browser to the server.
- Examples of reflected cross-site scripting attacks include when an attacker stores malicious script in the data sent from a **website's search or contact form**.
- A typical example of reflected cross-site scripting is a search form, where visitors sends their search query to the server, and only they see the result.

```
<input type="search" value="potatoes" />
```

- Attackers typically send victims custom links that direct unsuspecting users toward a vulnerable page. From this page, they often employ a variety of methods to trigger their proof of concept

```
<input type="search" value="Attacker "><script>StealCredentials()</script>" />
```



# Cross Site Scripting Attack (XSS Attack) – Types

---

**Self Cross-Site Scripting:** Self cross-site scripting occurs when attackers exploit a vulnerability that requires extremely specific context and manual changes. The only one who can be a victim is yourself. These specific changes can include things like cookie values or setting your own information to a payload.

**Blind Cross-Site Scripting:** Blind cross-site scripting attacks occur when an attacker can't see the result of an attack. In these attacks, the vulnerability commonly lies on a page where only authorized users can access. This method requires more preparation to successfully launch an attack; if the payload fails, the attacker won't be notified.



# Cross Site Scripting Attack (XSS Attack) – Types

---

**DOM-Based Cross-Site Scripting:** Occurs when the server itself isn't the one vulnerable to XSS, but rather the JavaScript on the page is. As JavaScript is used to add interactivity to the page, arguments in the URL can be used to modify the page after it has been loaded. By modifying the DOM when it doesn't sanitize the values derived from the user, attackers can add malicious code to a page. An example of DOM-based cross-site scripting attack would be when the website changes the language selection from the default one to one provided in the URL



# SQL Injection Attack

---

1. SQL Injection (SQLi) is a type of an injection attack that allows an attacker to interfere with the queries that an application makes to its database.
  
2. It generally allows an attacker to view data that they are not normally able to retrieve. This might include data belonging to other users, or any other data that the application itself is able to access.
  
3. In many cases, an attacker can modify or delete this data, causing persistent changes to the application's content or behavior.



# SQL Injection Attack

---

1. An SQL Injection vulnerability may affect any website or web application that uses an SQL database such as MySQL, Oracle, SQL Server, or others.
2. Criminals may use it to gain unauthorized access to your sensitive data: customer information, personal data, trade secrets, intellectual property, and more.
3. SQL Injection attacks are one of the oldest, most prevalent, and most dangerous web application vulnerabilities.
4. The OWASP organization (Open Web Application Security Project) lists injections in their OWASP Top 10 2017 document as the number one threat to web application security.



# Malware: **VIRUSES**, Worms, Trojans, Logic Bomb, Bots, Rootkits

---

1. A virus is a specific type of malware that self-replicates by inserting its code into other programs.
  2. Computer viruses have been prominent since almost the beginning of the commercial internet: The first one was created in 1982 for the Apple II, and other versions quickly followed.
  3. Viruses spread by attaching themselves to legitimate files and programs, and are distributed through infected websites, flash drives, emails, software installations or unsecured links..
  4. A victim activates a virus by opening the infected application or file.
  5. Once activated, a virus may delete or encrypt files, modify applications, or disable system functions.
- 



# Malware: **VIRUSES**, Worms, Trojans, Logic Bomb, Bots, Rootkits

---

1. A computer **Virus** is more dangerous than a computer **worm** as it makes changes or deletes your files while worms only replicates itself without making changes to your files/data.
2. Typical signs of computer virus infections include:
  - a. Ongoing crashes and blue screen errors
  - b. Slow performance
  - c. Missing files
  - d. Low storage
  - e. Unexpected behavior
  - f. Constant browser pop-ups
  - g. Unidentifiable programs
  - h. Increased network activity
  - i. Disabled security software



# Malware: **VIRUSES**, Worms, Trojans, Logic Bomb, Bots, Rootkits

## File-infecting Virus

A virus that attaches itself to an executable program. It is **also called a parasitic virus** which typically infects files with .exe or .com extensions. Some file infectors can overwrite host files and others can damage your hard drive's formatting.

## Browser Hijacker

This virus targets and alters your browser setting. It is often called a browser redirect virus because it redirects your browser to other malicious websites that you don't have any intention of visiting. This virus can pose other threats such as changing the default home page of your browser.

## Macro Virus

This type of virus is commonly found in programs such as Microsoft Word or Excel. These viruses are usually stored as part of a document and can spread when the files are transmitted to other computers, often through email attachments.

## Web Scripting Virus

A very sneaky virus that targets popular websites. What this virus does is overwrite code on a website and insert links that can install malicious software on your device. Web scripting viruses can steal your cookies and use the information to post on your behalf on the infected website.



# Malware: **VIRUSES**, Worms, Trojans, Logic Bomb, Bots, Rootkits

## Boot Sector Virus

These viruses are once common back when computers are booted from floppy disks. Today, these viruses are found distributed in forms of physical media such as external hard drives or USB. If the computer is infected with a boot sector virus, it automatically loads into the memory enabling control of your computer.

## Polymorphic Virus

This virus has the capability to evade anti-virus programs since it can change codes every time an infected file is performed.

## Resident Virus

A resident virus stores itself on your computer's memory which allows it to infect files on your computer. This virus can interfere with your operating system leading to file and program corruption.

## Multipartite Virus

A type of virus that is very infectious and can easily spread on your computer system. It can infect multiple parts of a system including memory, files, and boot sector which makes it difficult to contain.



# Malware: Viruses, **WORMS**, Trojans, Logic Bomb, Bots, Rootkits

---

1. A **computer worm** is a standalone malware computer program that replicates itself in order to spread to other computers.
2. It often uses a computer network to spread itself, relying on security failures on the target computer to access it. It will use this machine as a host to scan and infect other computers
3. A computer worm has the ability to operate autonomously, without the need for a host file .
4. The computer worm does not usually infect computer files, but rather infects another computer on the network.



# **Malware: Viruses, **WORMS**, Trojans, Logic Bomb, Bots, Rootkits**

---

## **How does a Computer Worm work?**

In order to spread, computer worms use vulnerabilities in networks. The worm is looking for a back door to penetrate the network unnoticed. To get computer worms into circulation for the first time, hackers often send phishing e-mails or instant messages with malicious attachments. Cyber criminals try to camouflage the worm so that the recipient is willing to run the program. For this purpose, for example, double file extensions are used and / or a data name that looks harmless or urgent, such as “invoice”. When the user opens the attachment or link, they will immediately download the malware (computer worm) into the system or be directed to a dangerous website. In this way, the worm finds its way into the user’s system without them noticing. Once executed, the worm seeks a way to replicate and penetrate other systems. One way of doing this, for example, is for the worm to send an email to all contacts on the infected computer, which contains replicas of the worm.

---



# Malware: Viruses, **WORMS**, Trojans, Logic Bomb, Bots, Rootkits

---

## How does a Computer Worm work?

Many worms now have what is known as a payload. Payload in this case is an attachment that the worm brings with it. The worm can, for example, **CARRY RANSOMWARE**, viruses or other malware, which then cause damage to the infected systems. These can then, for example, delete files on the PC or encrypt files in the event of a blackmail attack. A computer worm can also install a back door that can later be exploited by other malware programs. This vulnerability gives the worm's author control over the infected computer.



# Malware: Viruses, Worms, **TROJANS**, Logic Bomb, Bots, Rootkits

---

1. A Trojan horse, or Trojan, is a type of malicious code or software that looks legitimate but can take control of your computer.
2. A Trojan is designed to damage, disrupt, steal, or in general inflict some other harmful action on your data or network.
3. A Trojan acts like a genuine application or file to trick you. It seeks to deceive you into loading and executing the malware on your device. Once installed, a Trojan can perform the action it was designed for.
4. It can cause problems like killing background system processes, deleting hard drive data and corrupting file allocation systems.



# Malware: Viruses, Worms, **TROJANS**, Logic Bomb, Bots, Rootkits

---

## Trojan malware example

You might think you've received an email from someone you know and click on what looks like a legitimate attachment. But you've been fooled. The email is from a cybercriminal, and the file you clicked on — and downloaded and opened — has gone on to install malware on your device.

When you execute the program, the malware can spread to other files and damage your computer.



# Malware: Viruses, Worms, **TROJANS**, Logic Bomb, Bots, Rootkits

---

## Types of Trojans

**Backdoor Trojan:** This Trojan can create a “backdoor” on your computer. It lets an attacker access your computer and control it. Your data can be downloaded by a third party and stolen. Or more malware can be uploaded to your device.

**Distributed Denial of Service (DDoS) attack Trojan :**This Trojan performs DDoS attacks. The idea is to take down a network by flooding it with traffic. That traffic comes from your infected computer and others.

**Downloader Trojan:** This Trojan targets your already-infected computer. It downloads and installs new versions of malicious programs. These can include Trojans and adware.



# Malware: Viruses, Worms, **TROJANS**, Logic Bomb, Bots, Rootkits

---

## Types of Trojans

**Fake AV Trojan:** This Trojan behaves like antivirus software, but demands money from you to detect and remove threats, whether they're real or fake.

**Game-thief Trojan:** The losers here may be online gamers. This Trojan seeks to steal their account information.

**Infostealer Trojan:** As it sounds, this Trojan is after data on your infected computer.

**Mailfinder Trojan:** This Trojan seeks to steal the email addresses you've accumulated on your device.



# **Malware: Viruses, Worms, TROJANS, Logic Bomb, Bots, Rootkits**

---

## **Ransom Trojan**

This Trojan seeks a ransom to undo damage it has done to your computer. This can include blocking your data or impairing your computer's performance.

## **Remote Access Trojan**

This Trojan can give an attacker full control over your computer via a remote network connection. Its uses include stealing your information or spying on you.

## **Rootkit Trojan**

A rootkit aims to hide or obscure an object on your infected computer. The idea? To extend the time a malicious program runs on your device.

## **SMS Trojan**

This type of Trojan infects your mobile device and can send and intercept text messages. Texts to premium-rate numbers can drive up your phone costs.

## **Trojan IM**

This Trojan targets instant messaging. It steals your logins and passwords on IM platforms. That's just a sample. There are a lot more.



# Malware: Viruses, Worms, Trojans, **LOGIC BOMB**, Bots, Rootkits

---

1. A logic bomb virus is a computer virus that contains a logic bomb, which is malicious code that triggers an attack when specific conditions are met.
  
2. *Positive conditions* refer to something happening, like a program opening, while *negative conditions* refer to something not happening, like someone not logging in.
  
3. Logic bombs are often installed by someone with high-level access, such as a system administrator. Such a person can cause mayhem by setting up logic bombs on multiple systems and programming them to “blow up” simultaneously when a certain event occurs, like when an employee is removed from the company’s salary database.



# Malware: Viruses, Worms, Trojans, **LOGIC BOMB**, Bots, Rootkits

---

1. A logic bomb virus is a computer virus that contains a logic bomb, which is malicious code that triggers an attack when specific conditions are met.
  
  2. *Positive conditions* refer to something happening, like a program opening, while *negative conditions* refer to something not happening, like someone not logging in.
  
  3. Logic bombs are often installed by someone with high-level access, such as a system administrator. Such a person can cause mayhem by setting up logic bombs on multiple systems and programming them to “blow up” simultaneously when a certain event occurs, like when an employee is removed from the company’s salary database.
  
  4. Another name for a logic bomb is **slag code**, which refers to the manipulated code that makes an otherwise safe program harmful.
- 



# **Malware: Viruses, Worms, Trojans, LOGIC BOMB, Bots, Rootkits**

---

**All logic bombs share the following characteristics:**

1. They lie dormant until triggered.
2. They carry an unknown payload, which is the part of the code that performs the attack.
3. They deliver the payload when a certain condition is met.

**Among other things, a logic bomb can deliver its payload when:**

1. A specified amount of time elapses.
2. A specific date occurs.
3. A certain transaction is processed.
4. A particular program opens.
5. Someone (for example, an admin) fails to log in.

A logic bomb's potential payload  
may be designed to:

- Corrupt data.
- Wipe hard drives.
- Delete files.
- Siphon off funds.
- Gather sensitive data.



## Malware: Viruses, Worms, Trojans, **LOGIC BOMB**, Bots, Rootkits

---

A typical use case for a logic bomb is an insider attack. For example, let's say a privileged user has a grudge against his company and is afraid he might soon be fired. In advance of his firing, he sets up a scheduled job that checks to see if his user account has been active during the past 90 days; if no activity is found, the scheduled job deletes a critical database. Sure enough, the user is fired, and a few months later, once his account has been inactive for 90 days, the database is deleted. In most cases this would be a visible and obvious action, but what makes a logic bomb especially insidious is that it changes its code randomly, making it more difficult to detect and more damaging to the targeted organization.



# **Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, ROOTKITS**

---

1. A rootkit is a malicious software that allows an unauthorized user to have privileged access to a computer and to restricted areas of its software.
  
2. A rootkit may contain a number of malicious tools such as keyloggers, banking credential stealers, password stealers, antivirus disablers, and bots for DDoS attacks. This software remain hidden in the computer and allow the attacker remote access to the computer.



# **Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, ROOTKITS**

---

1. The term rootkit is derived from the combination of two words – "root" and "kit".
2. "Root" refers to the administrator account in Unix and Linux operating systems, which is an all-powerful account with full privileges and unrestricted access. It is equivalent to the administrator account in Windows systems.
3. The term "kit" refers to the programs that allow a threat actor to obtain unauthorized root/admin-level access to the computer and restricted areas.
4. The rootkit enables the threat actor to perform all these actions surreptitiously without the user's consent or knowledge.



# Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, **ROOTKITS**

---

## How the Attacker Installs Rootkits.

The attacker tries to obtain root/administrator access by exploiting known vulnerabilities, or by stealing administrator privilege credentials. Cyber criminals employ social engineering techniques to obtain credentials. Root access allows installation of rootkits or any other malware. Installation of the rootkit enables the threat actor to access the computer from remote to install other malware, steal data, observe activities and even control the computer. Rootkits are sophisticated malware, and most antivirus solutions and antimalware solutions do not detect rootkits. Rootkits are also able to hide their intrusion, and hence once they are in, they are practically undetectable.

Since rootkits have complete control over the system, they can modify software and the cyber security solutions such as the antivirus that could detect rootkits. As even the detection solutions are modified, it is difficult to detect and remove rootkits.



# Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, **ROOTKITS**

---

## What are Rootkits used for?

### Rootkits are used for many purposes:

**Stealth capabilities:** Modern rootkits add stealth capabilities to malicious software payloads (such as keyloggers and viruses) to make them undetectable.

**Backdoor access:** Rootkits permit unauthorized access through backdoor malware. The rootkit subverts the login mechanism to also accept a secret login access for the attacker. Standard authentication and authorization mechanisms are bypassed to provide admin privileges to the attacker.

**DDoS attacks:** Rootkits allow the compromised computer to be used as a bot for distributed-denial-of-service attacks. The attack would now be traced to the compromised computer and not to the attacker's system. These bots are also called as zombie computers and are used as part of bot networks to launch the DDoS attacks, and other malicious activities such as click fraud and spam email distribution.

---



# **Malware: Viruses, Worms, Trojans, Logic Bomb, Bots, ROOTKITS**

---

**The functionality of rootkits is also used for good causes, such as:**

- in a honeypot to detect attacks
- to enhance emulation software
- to enhance security software – it enables the software to secure itself from malicious actions
- digital rights management enforcement
- device anti-theft protection - BIOS-based rootkit software enables monitoring, disabling and wiping of data on mobile devices when they get lost or stolen

**There are five types of rootkits**

- 1. User-mode rootkits**
- 2. kernel-mode rootkits**
- 3. bootkits**
- 4. hypervisor rootkits**
- 5. firmware rootkits.**

